Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 214 (2006) 829–857
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

Simultaneous space–time adaptive wavelet solution of nonlinear parabolic differential equations

Jahrul M. Alam [a], Nicholas K.-R. Kevlahan [a,*], Oleg V. Vasilyev [b]

[a] *Department of Mathematics & Statistics, McMaster University, 1280 Main St W, Hamilton, ON, Canada L8S 4K1*
[b] *Department of Mechanical Engineering, University of Colorado, 427 UCB, Boulder, CO 80309, USA*

Received 4 May 2005; received in revised form 29 August 2005; accepted 17 October 2005
Available online 1 December 2005

Abstract

Dynamically adaptive numerical methods have been developed to efficiently solve differential equations whose solutions are intermittent in both space and time. These methods combine an adjustable time step with a spatial grid that adapts to spatial intermittency at a fixed time. The same time step is used for all spatial locations and all scales: this approach clearly does not fully exploit space–time intermittency. We propose an adaptive wavelet collocation method for solving highly intermittent problems (e.g. turbulence) on a simultaneous space–time computational domain which naturally adapts both the space and time resolution to match the solution. Besides generating a near optimal grid for the full space–time solution, this approach also allows the global time integration error to be controlled. The efficiency and accuracy of the method is demonstrated by applying it to several highly intermittent $(1D + t)$-dimensional and $(2D + t)$-dimensional test problems. In particular, we found that the space–time method uses roughly 18 times fewer space–time grid points and is roughly 4 times faster than a dynamically adaptive explicit time marching method, while achieving similar global accuracy.

Keywords: Wavelets; Lifting scheme; Second generation wavelets; Partial differential equations; Elliptic problem; Adaptive grid; Numerical method; Multi-level method; Multi-grid method

1. Introduction

Mathematical modeling of problems in science and engineering (e.g. turbulence, reactive or non-reactive flows [1]) typically involves solving nonlinear partial differential equations (PDEs). A wide range of spatial and temporal scales must often be resolved in order to properly solve these equations [2]. However, in many situations the small spatial scales are highly localized, and thus efficient solution of the problem requires a locally adapted grid. A uniformly fine grid is clearly inefficient for such problems. Turbulence is a well-known example of a problem with high intermittency [3,4]. In high Reynolds number turbulence the number of degrees of freedom scales like the cube of Reynolds number, Re^3, in a uniform mesh that resolves the smallest

* Corresponding author.
E-mail addresses: alamj@math.mcmaster.ca (J.M. Alam), kevlahan@mcmaster.ca (N.K.-R. Kevlahan), Oleg.Vasilyev@Colorado.EDU (O.V. Vasilyev).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2005.10.009

active structures in space and time [5]. Since turbulence usually occurs at high Reynolds number (e.g. $Re \sim 10^6$ for a typical aeronautical flow), it is clear that any successful direct numerical simulation (DNS) of turbulence must take advantage of the flow's high intermittency [6,7]. Because of this intermittency, we expect that the minimum number of computational elements required is actually much smaller than Re^3.

Recently there has been increasing interest in developing adaptive [8–15] numerical methods for solving elliptic [16–20] and time-dependent [21–35] partial differential equations. Existing adaptive numerical methods fall into two classes: *error indicator* based (where the grid is refined to resolve gradients of a physically relevant quantity), and *error control* based (where the error is estimated and the grid is refined to ensure this error is less than a prescribed tolerance). The error-indicating strategy does not control the error directly, but instead controls the mesh coarsening and refinement. The error-estimating strategy minimizes the error as measured in an appropriate norm, which leads to an optimal mesh size distribution.

Wavelets have proved to be an efficient tool in developing adaptive numerical methods which control the global (usually L^2) approximation error [17,18,21,24,26,35]. The goal of the collocation-based nonlinear wavelet approximation is to obtain the best approximation of a function on a near optimal grid. The collocation approximation has a one-to-one correspondence between the wavelet expansion coefficients and grid points. Thus, nonlinear filtering of wavelet coefficients automatically refines the computational grid. Since functions and operators can be computed with a given accuracy, adaptive wavelet method provides global error control for the adaptive solution of differential equations.

Liandrat and Tchamitchian [21] proposed the first wavelet-based adaptive method for partial differential equations. Until the work of Sweldens [36], the research effort was focused on compressing both the differential operators and the solution using Galerkin projection. The early work found in [37–39] demonstrated the use of wavelets to find the numerical solution of PDEs with periodic boundary conditions. Galerkin-based wavelet methods for linear elliptic problems were studied in [16,40–42]. Schneider [43] used reliable and efficient a posteriori error estimates for adaptive multi-scale wavelet-Galerkin schemes for linear elliptic PDEs. The error achieved by adaptive wavelet schemes [16,17,19] is proportional to the smallest error realized by the wavelet approximation, i.e. these schemes are asymptotically optimal for elliptic problems [20]. In addition, adaptive wavelet methods are fast (at least for large problems) since the computational complexity scales like the number of wavelets retained in the approximation, $O(\mathcal{N})$.

Adaptive wavelet schemes have also been used for solving time dependent partial differential equations [21,22]. A more detailed derivation of fast and adaptive algorithms, projection of the solution and spatial derivatives on wavelet space, relationship between sparseness of the discretized system and the vanishing moment property of wavelets was developed by Beylkin and Keiser [29]. Debussche et al. [44] developed a multi-level Fourier–Galerkin method for homogeneous turbulence. The main difficulties of a Galerkin-based wavelet method are the efficient computation of nonlinear operators, and the implementation of general boundary conditions. These difficulties led to the development of collocation-based adaptive wavelet methods, e.g. [25,26,28]. Following the second-generation multi-resolution approximation of Sweldens [45], a multi-level adaptive wavelet collocation method was developed by Vasilyev and Bowman [34], which was applied to a wide variety of initial value problems problems [35]. The adaptive wavelet collocation method (AWCM) has since been used to construct a multi-level adaptive elliptic solver [46], two- and three-dimensional simulation of fluid–structure interaction [47–49], and a wavelet-based alternative to large eddy simulation [50].

To the best of our knowledge, all existing wavelet methods for time-dependent problems adapt the spatial grid dynamically in the region of intermittency. This means that mesh refinement or coarsening is automatic if the solution develops strong gradients, or if these gradients diffuse. If the solution is intermittent in both space and time, one adapts the spatial mesh to the solution at a fixed time and uses an adjustable time step to control the local error in time [22]. This approach enforces the same time step for all spatial locations, which is clearly not optimal for problems which are simultaneously intermittent in both space and time.

Following the classical time marching technique, an adaptive wavelet method discretizes the PDE to produce a system of ODEs with wavelet coefficients as time-dependent unknowns (in the Galerkin formulation), or nodal approximations as time-dependent unknowns in the collocation approach. This method adapts the spatial grid as shocks or any localized structures develop or move in a time-dependent solution. The dynamically adjusted time stepping procedure determines the maximum allowable time step for the spatially adapted grid, or for all the wavelet modes, at any instant. However, although the spatial error is controlled by the adaptive

wavelet approximation, the global error in time is uncontrolled. There is no guarantee that the temporal truncation error will not accumulate over time and eventually exceed the desired error tolerance. Often one needs to integrate the PDE for an arbitrarily long period of time. The spurious effect of accumulated error can cause the results of the calculation to become unreliable, even though the spatial error is controlled at each time step.

Two approaches have been proposed to control the global error in time. The first is the variational integrator approach developed by Marsden and his co-workers [51–56], where the global time integration error is reduced significantly so that the appropriate conservation laws are satisfied to within a desired tolerance for arbitrary times. An example of the second approach is the recent work of Tremblay et al. [57] who use a time-continuous space–time finite element formulation. This method, however, does not use automatic grid adaptation in the space–time computational domain.

In this paper, we develop a simultaneous space–time AWCM for time-dependent PDEs. Our aim is to address the two shortcomings of current numerical methods mentioned above: the inefficiency of using a single time step for all spatial locations, and the lack of control of the global error in time. The simultaneous space–time adaptive wavelet solution should produce accurate solutions for arbitrary times on a near-optimal space–time grid.

The paper is organized as follows. In Section 2, we briefly review the AWCM. Section 3 describes the proposed numerical method. The ideas behind the wavelet-based time integration technique for ODEs and the space–time integration technique for PDEs are explained in Sections 3.2 and 3.3. Section 3.6 outlines the wavelet-based full approximation scheme (FAS) we have developed for solving the nonlinear algebraic problem which results from full discretization of the time-dependent PDE. In Section 3.8, we describe a way of splitting the space–time domain into subdomains in the time dimension. This allows for calculation over arbitrary long times given the available computational resources. In Section 4, we present the results of numerical experiments using the $(1D + t)$ Burgers equation and $(2D + t)$ vorticity equation in order to verify the efficiency and accuracy of the proposed numerical method. We summarize the paper and discuss future research direction in Section 5.

2. Second generation adaptive wavelet discretization of PDEs

Our AWCM is based on the second generation wavelets developed by Sweldens [45,58,59], and we will consider general boundary value problems of the form:

$$\mathscr{L}u = f \quad \text{in } \Omega, \tag{1}$$

$$\mathscr{B}u = g \quad \text{in } \partial\Omega, \tag{2}$$

where $\mathscr{L}$ is a general partial differential operator and $\mathscr{B}$ is an operator that defines proper boundary conditions. Ω is a space–time domain such that

$$\Omega = D \times (0, T), \tag{3}$$

where $D \subset \mathbb{R}^n$, $T \in \mathbb{R}^+$, Ω is open, connected, and bounded set with boundary $\partial\Omega$, i.e. $\bar{\Omega} = \Omega \cup \partial\Omega$. A point in $\bar{\Omega}$ is denoted by $x = \{x_1, x_2, \ldots, x_n, x_{n+1}\}^{\mathscr{T}}$. If $n = 0$, then $\bar{\Omega} = [0, T]$ is a closed interval. When $n = 1$, we will be using $(x, t) \in \Omega$ to denote points in Ω. We will assume that all functions are from the function space $L^2(\Omega)$ unless otherwise stated.

2.1. Multi-scale decomposition

In the second generation multi-resolution approximation, functions are approximated using tensor product second generation wavelets that are constructed on a nested set ($G^j \subset G^{j+1}$) of collocation points

$$G^j = \left\{ x_k^j \in \bar{\Omega} : x_k^j = x_{2k}^{j+1}, k \in \mathscr{K}^j, j \in \mathscr{Z} \right\}, \tag{4}$$

where k denotes the position, j denotes the level or scale, and x_k^j are the collocation points in $\bar{\Omega}$ with $n = 0$. Here $\mathscr{Z}$ and $\mathscr{K}^j$ are some suitable index sets. Note that collocation points x_k^j can be distributed uniformly, $x_k^j = 2^{-j}k$, or non-uniformly [34].

To illustrate the second generation wavelet discretization of partial differential equations, let us consider the multi-scale decomposition of a function $u(x)$ at a certain level of resolution $J \geqslant 0$:

$$u^J(x) = \sum_{j=0}^{J} \sum_{k \in \mathscr{K}^j} d_k^j \psi_k^j(x), \quad J \to \infty, \tag{5}$$

where $\psi_k^j(x)$ are localized basis functions (wavelets) and d_k^j are expansion coefficients [60]. The major strength of the wavelet decomposition (5) is that the wavelet coefficient d_k^j measures the local variation of $u(x)$ at scale 2^{-j} near x_k^j. These coefficients should decay quickly to zero in the smooth regions, and should be large only in the region where the gradient of $u(x)$ is large. This behavior suggests that there exists a decreasing sequence of positive numbers ϵ_J such that $\forall j \geqslant J,\ |d_k^j| < \epsilon_J$. In other words, any truncation of the above infinite sum over j is an approximation of $u(x)$ at scale 2^{-j}. Secondly, if $u(x)$ is smooth, except at some isolated points, the above truncation requires a small number of coefficients to approximate $u(x)$. Thus, the multi-scale decomposition not only approximates a function, it also compresses it.

2.2. Adaptive approximation

For intermittent functions, the coefficients d_k^j's are large only for those positions k where the function has a steep gradient. Therefore, discarding coefficients whose magnitudes are smaller than a given threshold ϵ truncates the infinite sum (5) to a finite sum, as well as compressing the function. The truncated sum $u^J(x)$ is a good approximation of $u(x)$ at level J in the weighted residual sense, i.e.

$$\int_\Omega (u(x) - u^J(x))\delta(x - x_k^J)\,\mathrm{d}x = 0. \tag{6}$$

This restriction establishes a one-to-one correspondence between d_k^J and x_k^J. The grid adaptation strategy is based on the fact that discarding a wavelet coefficient is equivalent to discarding the corresponding collocation point. To construct a grid that adapts to the intermittent solution we collect all collocation points x_k^j such that $|d_k^j| \geqslant \epsilon$; i.e.

$$G_\epsilon^j = \left\{x_k^j \in \Omega : x_k^j = x_{2k}^{j+1},\ k \in \mathscr{K}^j,\ j \in \mathscr{L},\ |d_k^j| \geqslant \epsilon\right\}. \tag{7}$$

Adaptive second generation wavelet decomposition then takes the following form:

$$u_\epsilon^J(x) = \sum_{j=0}^{J} \sum_{\substack{k \in \mathscr{K}^j \\ |d_k^j| \geqslant \epsilon}} d_k^j \psi_k^j(x). \tag{8}$$

For functions that have localized structure in Ω, G_ϵ^j is much more compressed than G^j for all j. The decomposition (8) is usually known as nonlinear approximation in a wavelet basis.

2.3. Computing derivatives on adaptive grids

Approximating derivatives of a function in a collocation-based method is straightforward. Analogous to the continuous case, the derivative of a function is approximated from the approximation of the function on a grid G^J; i.e. from the nodal values of the function. The derivative $\mathscr{D}^m u$ of order m is computed on a grid G^j from the values $u(x_k^j)$ of the function $u(x)$,

$$\mathscr{D}^m u(x_k^j) = \sum_l \mathscr{D}_{kl}^{mj} u(x_k^j). \tag{9}$$

The entries $\mathscr{D}_{kl}^{mj}$ can be determined by applying a finite difference formula on $u^j(x)$; i.e. we can apply the finite difference approximation to $u^j(x)$ on grid G^j. When the grid is not uniform (e.g. G_ϵ^j), we can use a procedure in computing derivatives as explained by Vasilyev and Bowman [34] and Vasilyev and Kevlahan [46]. This procedure corresponds to a local finite difference operator that uses neighboring points of x_k^j to compute $\mathscr{D}^m u(x_k^j)$ at the appropriate level of resolution [34].

Thus, the adaptive wavelet method combines the fast second generation wavelet transform with finite difference approximation of derivatives. For intermittent functions, the number of active collocation points $\mathcal{N}$ is minimal (see [19,60,20]). The computational cost of calculating the derivatives is only $O(\mathcal{N})$, which is the same as the computational cost of wavelet transform. The accuracy and efficiency of the adaptive lifted interpolating wavelet differentiation is examined by Vasilyev [35]. On a uniform grid, this technique is consistent with standard finite difference stencils.

2.4. Analytical error estimates

Let us truncate the sum (5) at level J and define the residual of truncation by

$$r^J(x) = u(x) - u^J(x). \tag{10}$$

In multi-level wavelet approximation of functions and derivatives, the error depends on the wavelet thresholding parameter ϵ and the order of the wavelets [61]. Due to the one–to–one correspondence between the wavelets and the collocation points, one can relate the error with the active grid points. The most important feature is that the approximation error has a global control throughout the domain. For sufficiently smooth functions $u(x)$, we can find an ϵ such that $|d_k^j| < \epsilon \ \forall j \geqslant J$ and so the residual of approximation at level J is upper bounded as [61]

$$|r^J(x)| \sim \epsilon, \quad \epsilon \to 0. \tag{11}$$

Eq. (11) is independent of the dimensionality of the problem. Since the number of active collocation points depends on the dimension and the order of wavelets being used, one can show that the number of active coefficients satisfies

$$\mathcal{N} \sim \epsilon^{-n/p}, \quad \epsilon \to 0, \tag{12}$$

where p is the order of wavelets used and n is the dimensionality. In other words, the truncation error is related to the number of terms retained as

$$\|r^J(x)\|_2 \sim \mathcal{N}^{-p/n}, \quad \epsilon \to 0. \tag{13}$$

The accuracy of the differentiation procedure was examined by Vasilyev and Bowman [34] for the one-dimensional case and by Vasilyev [35] for the multi-dimensional case. The error in the adaptive wavelet approximation of derivatives is

$$\|D^m_{x_i} u(\mathbf{x}) - D^m_{x_i} u^j_\epsilon(\mathbf{x})\|_2 \sim \mathcal{N}^{-(p-m)/n}, \quad \epsilon \to 0, \tag{14}$$

where $D^m_{x_i}$ stands for derivative of order m in the x_i direction.

3. Proposed numerical method

3.1. Background and motivation

Let us consider the general parabolic initial value problem:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \mathscr{F}(u,t), \quad u \in \mathbb{R}^n, \ t \in (0,T), \tag{15}$$

$$u(0) = u_0, \tag{16}$$

where $\mathscr{F}$ represents any function (usually nonlinear). In the classical time-marching scheme, we divide the interval $[0, T]$ into N sub-intervals such that $t_n = n\Delta t$, where Δt is the width of each of the sub-intervals. We consider a single sub-interval $[t_n, t_{n+1}]$, for some n, and assume that u at $t = t_n$ is known. Using a suitable numerical method (e.g. forward difference in time), we can compute u at $t = t_{n+1}$ and repeat for the next sub-interval $[t_{n+1}, t_{n+2}]$. Thus, starting with the initial value, we march forward in time to compute the solution at each of the discrete temporal locations. In other words, we solve a sequence of algebraic problems, each of which is defined a sub-interval of $[0, T]$.

Two major difficulties arise in the time marching scheme. First, the truncation error accumulates in time. Let us assume that the local truncation error of the scheme is $O(\Delta t^{\alpha})$. Then the global error after N time steps is $O(\Delta t^{\alpha-1})$, where it is assumed that $N \sim 1/\Delta t$. However, it is easy to see that global error may become arbitrarily large if $N \gg 1/\Delta t$. Reducing the time step reduces the error locally, but provides no global error control [62]. For example, if we consider a scheme with $\alpha = 2$ and $\Delta t = 10^{-2}$, we expect the global error at the end of 10^2 time steps (i.e. $N \sim 1/\Delta t$) to be $O(10^{-2})$. This error will be $O(1)$ after 10^4 time–steps, if we continue marching. Clearly, reducing the time step will reduce the error only locally: the time integration error will continue to accumulate. Secondly, this process uses the same time step for all components of $u \in \mathbb{R}^n$ even though the problem may have multiple time scales. When the problem (15)–(16) is nonlinear, there is no simple way to adopt a non-uniform time stepping such that different components of u use different time steps. The conventional dynamically adjusted time stepping procedure only determines the maximum allowable time step at any particular time; it does not resolve the natural time scales of the governing dynamical system. Our goal in the following is to develop an AWCM-based method that addresses both these problems: global error control in time and local time stepping.

3.2. Wavelet-based adaptive integration

We now propose a wavelet-based technique to handle the difficulties associated with the classical time stepping schemes in the previous section. In contrast to time marching, where a sequence of discrete algebraic subproblems are solved, we propose to reduce the PDE to a single algebraic problem in the entire time domain $[0, T]$. Thus, to develop an AWCM integration technique for solving Eq. (15), we consider a pseudo boundary value problem in $[0, T]$ with a Dirichlet condition at $t = 0$ and a suitable terminal condition at $t = T$. Since the problem (15) is well-posed (i.e. its solution is uniquely determined from the available boundary data), adding a terminal condition makes the problem overdetermined. However, the numerical method needs information to correctly evolve the solution at the $t = T$ boundary from the initial data given on the $t = 0$ boundary. The correct procedure is to determine the value of the wavelet coefficients at the $t = T$ boundary using *nearest neighbours* such that the gradient of the solution is properly calculated at $t = T$. This ensures that the solution $u(x, T)$ is determined from $u(x, t < T)$. We call the constructed problem a *pseudo boundary value problem* because the proposed terminal condition

$$\frac{\mathrm{d}u}{\mathrm{d}t}(T) - \mathscr{F}(u, T) = 0 \tag{17}$$

is a dynamic condition at $t = T$, which we call an *evolution condition*. This boundary condition does not make the problem overdetermined and is not necessary for the existence or uniqueness of the solution. This is in contrast to the artificial or numerical boundary conditions that are sometimes used to determine the interior solution in a hyperbolic system [63].

Consider second generation bi-orthogonal wavelets on dyadic grids of $[0, T]$, i.e. $t_k^j = 2^{-j}k$ for all $j \geqslant 0$. Let $N^j + 1$ be a total number of grid points on level j. We now expand $u(t)$ and $\mathrm{d}u/\mathrm{d}t$ in multi-scale wavelet basis at scale 2^{-j} to get the following:

$$\begin{aligned}
&u(t_0^j) = u_0(t_0^j) \qquad \text{(left boundary)},\\
&\frac{\mathrm{d}u}{\mathrm{d}t}(t_k^j) - \mathscr{F}(u(t_k^j), t_k^j) = 0, \quad 1 \leqslant k \leqslant N^j - 1 \qquad \text{(internal points)},\\
&\frac{\mathrm{d}u}{\mathrm{d}t}(t_k^j) - \mathscr{F}(u(t_k^j), t_k^j) = 0, \quad k = N^j \qquad \text{(right boundary)}.
\end{aligned}$$

Since we are using a collocation-based weighted residual approximation of functions in wavelet basis, the derivatives of $u(t)$ at each of the collocation points t_k^j are computed from the approximation of $u(t)$ according to some discretization stencil (explained later). Using Eq. (9), the above system of difference equation thus reduces to the following algebraic problem:

$$L^j U^j = F^j, \tag{18}$$

where F^j and U^j are $(N^j+1)\times 1$ vectors and L^j is an $(N^j+1)\times(N^j+1)$ matrix. The entries of the matrix L^j depends on the function $\mathscr{F}$ such that

$$L^jU^j = \begin{cases} u(t_k^j) & \text{for } k=0, \\ \frac{\mathrm{d}u}{\mathrm{d}t}(t_k^j) - \mathscr{F}(u(t_k^j), t_k^j) & \text{for } 1 \leqslant k \leqslant N^j - 1, \\ \frac{\mathrm{d}u}{\mathrm{d}t}(t_k^j) - \mathscr{F}(u(t_k^j), t_k^j) & \text{for } k = N^j, \end{cases}$$

and the entries of the vector F^j are given by

$$F^j = \begin{cases} u_0(t_k^j) & \text{for } k=0, \\ 0 & \text{for } 1 \leqslant k \leqslant N^j - 1, \\ 0 & \text{for } k = N^j. \end{cases}$$

The solution of the nonlinear algebraic problem (18) gives the approximate solution $u(t_k^j)$ of the initial value problem (15) at each $t_k^j = 2^{-j}k$. The main advantages of this approach are the following:

- *Global error control.* Since $u(t)$ is approximated in the whole time interval $[0, T]$, the approximation error is bounded globally by the wavelet thresholding parameter according to Eq. (11).
- *Natural time stepping.* Time steps can be adapted easily to the natural time scale of each of the components of $u \in \mathbb{R}^n$ since the expansion coefficients d_k^j's are large if the solution is temporally intermittent. The implementation of natural time adaptation is a recursive process and is similar to that described by Vasilyev and Kevlahan [46] for the elliptic case.
- *Reduced computational complexity.* In second generation wavelet discretization, the entries of the matrix L^j do not need to be computed explicitly. The entries of the vector L^jU^j are computed directly in $\mathrm{O}(\mathscr{N})$ operations, where $\mathscr{N}$ is the number of the active degrees of freedom (independent of the dimensionality of the problem). Thus, the computational cost of obtaining the global algebraic problem does not exceed $\mathrm{O}(\mathscr{N})$. More details on this estimate are given in [46]. The multi-scale wavelet decomposition provides a natural frame work to construct the algebraic problem (18) on nested dyadic grids of $[0, T]$. Thus, a multi-grid strategy (discussed later) is a natural choice to solve the algebraic problem, which is optimal in solving algebraic problems.

Thus, in contrast to the classical approach, where a sequence of algebraic problems is solved, we construct a single algebraic problem that can be solved optimally using a multi-resolution (multi-grid) strategy.

3.3. Simultaneous space–time pseudo boundary value problem

Vasilyev and Kevlahan [46] developed a multi-level AWCM to solve elliptic partial differential equations. We have described the adaptive wavelet decomposition technique in Section 2, and presented a wavelet-based adaptive time integration technique in Section 3.2. In this subsection, we describe how to extend this wavelet-based adaptive integration technique to solve a nonlinear time-dependent partial differential equation. Since the algorithm is similar to the multi-level elliptic-AWCM described in [46], we only discuss the differences and mathematical aspects here. Interested readers who need a more details of the adaptive wavelet collocation method should refer to the work of Vasilyev and Bowman [34] and Vasilyev [35] [and the references therein].

To make the time dependence explicit in (1), we split the partial differential operator $\mathscr{L}$,

$$\mathscr{L}u \equiv \frac{\partial u}{\partial t} + \mathscr{H}u = f, \tag{19}$$

where we now assume that $u(x, t)$ is a function that depends on one spatial variable x, and $\mathscr{H}$ is an operator that consists of partial derivatives with respect to x only. For simplicity, we consider the case of one spatial dimension. Extension of this method to multiple spatial dimensions is straightforward in principle, but requires more care in implementation to ensure computational efficiency (e.g. improving the data structure and parallelizing the algorithm). In order to solve Eq. (19) in the space–time domain $[-1, 1]\times[0, T]$, we use the following boundary conditions:

$$\mathscr{B}_- u = g_- \quad \text{for } x = -1,\ t \in (0,T), \tag{20}$$

$$\mathscr{B}_+ u = g_+ \quad \text{for } x = 1,\ t \in (0,T), \tag{21}$$

$$u(x,0) = u_0(x) \quad \text{for } x \in [-1,1],\ t = 0, \tag{22}$$

where the operators $\mathscr{B}\pm$ can be specified to impose Dirichlet, Neumann, mixed, or periodic boundary conditions. While Eqs. (19)–(22) define an initial-boundary value problem that can be solved uniquely, we are interested here in constructing a (pseudo) boundary value problem in the space–time domain. For this purpose, as mentioned previously, we propose an evolution condition at $t = T$:

$$\frac{\partial u}{\partial t}(x,T) + \mathscr{H} u(x,T) = f(x,T). \tag{23}$$

If this boundary condition is added to the initial-boundary value problem defined by Eqs. (19)–(22), we can construct a boundary value problem in the space–time domain, without overdetermining the problem [64].

3.4. Discretization of a general boundary value problem

We have seen that the boundary value problem given by Eqs. (1) and (2) can be used to represent an initial value problem or an initial-boundary value problem with the proper choice of the operator $\mathscr{L}$, the domain Ω, and an evolution condition at the long time boundary $t = T$. The procedure for approximating any function and its derivatives described in Section 2 can now be applied to discretize the general boundary value problem (1). Let us construct a tensor product space–time dyadic grid $\mathbb{G}^j$ of $\Omega := \mathbb{I} \subset \mathbb{R} \times (0,T)$ at level j. An example of a space–time dyadic grid is shown in Fig. 1. The partial differential operator $\mathscr{L} := \frac{\partial}{\partial t} + \mathscr{H}(\partial_x, \partial_{xx}, \ldots)$ is approximated on each of the filled points in the space–time grid. This is supplemented by the approximation of the associated boundary conditions on each of the non-filled points.

Let $U^j = [u(x_k^j, t_n^j)]^{\mathrm{T}}$ be the one-dimensional vector containing the adaptive points of the nodal approximations of $u(x, t)$ at level j. In a collocation method, the derivative of $u(x, t)$ at $t = t_n^j, x = x_k^j$ is computed using the values of $u(x, t)$ at some neighboring grid points according to a discretization stencil. An example of a discretization stencil for a low-order approximation of the derivative, when $\mathscr{L} := \partial_t - \nu\partial_{xx}$, is presented in Fig. 2, which corresponds to a uniform mesh. In practice, we compute derivatives on simultaneous space–time adaptive grid with higher order accuracy [34]. It is easy to show that second generation wavelet collocation discretization of temporal and spatial derivatives in the space–time domain along with the discretized boundary conditions reduces the problem to a system of algebraic equations

$$L^j U^j = F^j, \tag{24}$$

where L^j is a $\mathscr{N} \times \mathscr{N}$ matrix, U^j is a $\mathscr{N} \times 1$ vector containing nodal values, F^j is a $\mathscr{N} \times 1$ vector that represents the source of the problem, and $\mathscr{N}$ is the total number of adaptive points on a space–time computational grid. The entries of $L^j U^j$ are computed in a similar way as in Eq. (18). The construction of this algebraic problem costs only $O(\mathscr{N})$ operations and the matrix L^j is never multiplied by the vector U^j explicitly. Therefore, any initial value problem may be reduced to a single (large) algebraic problem.

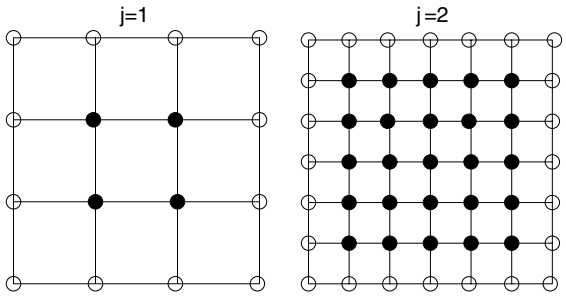

Fig. 1. An example of space–time dyadic grid at levels $j = 1$ and $j = 2$, where the horizontal and vertical axes corresponds to the space and time directions respectively, and filled or non-filled circles indicate if a point belongs to the interior or the boundary of the space–time domain respectively.

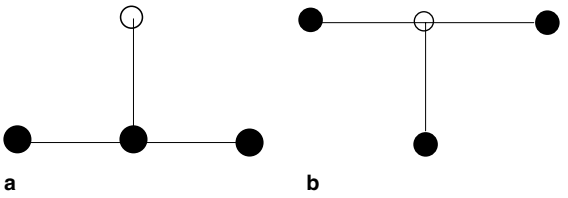

Fig. 2. (a) Explicit stencil. (b) Implicit stencil of discretization, where the horizontal direction corresponds to space and the vertical direction corresponds to time. The non-filled point corresponds to the location of wavelet where the equation is approximated, and the filled points are the nearest neighbors.

As noted earlier, since the time marching technique updates the solution for a fixed time, local truncation error increases at each time step. If the accumulated error is bounded above by some exponential function of time, then the numerical scheme is called stable [65]. However, if simultaneous space–time discretization is used, then the entire space–time mesh is solved at once and can be used by an error estimator to iteratively compute a new adapted space–time mesh. With this approach, the wavelet coefficients measure the local fluctuation of the solution simultaneously in space and time, and provide global control of the of both spatial and temporal error.

3.5. Implementation of the boundary conditions

In the collocation approach, implementation of general boundary conditions is straightforward. The given differential equation is approximated only for those collocation points that do not belong to the boundary, and the boundary conditions are approximated only at those collocation points that belong to the boundary. The algebraic problem (24) consists of the discretized PDE supplemented by the discrete approximation of the boundary conditions, which constitute an algebraic system, where the number of equations equals the number of unknowns.

3.6. Nonlinear multi-level adaptive solver

We have demonstrated a new procedure of numerically solving ordinary or partial differential equations using the AWCM. Our method constructs a single algebraic problem in the whole space–time domain. If the function $\mathscr{F}$ in Eq. (15), or the partial differential operator $\mathscr{H}$ in Eq. (19) is nonlinear, the algebraic problem (24) is also nonlinear. We now explain how to extend the AWCM multi-level solver developed in [46] to nonlinear problems.

One way of extending the AWCM multi-level solver to nonlinear problems is to use Newton's method. This produces a linear equation for the correction term at each iteration, one can then use the AWCM multi-level solver to solve this linear equation. However, this does not take full advantage of the multi-grid approach. The linear error equation is a local correction to the solution, and thus does not significantly influence the global rate of residual reduction. A more detailed discussion, with numerical examples of elliptic problem, can be found in [66]. The multi-level structure of the wavelet approximation gives us a natural frame-work to establish a wavelet-based full approximation scheme (WFAS), based on the concept of the full approximation scheme (FAS), which was originally developed for multi-grid methods by Brandt [8,67,68]. Multi-grid methods are very similar to AWCM in the sense that they both represent the solution on a nested sequence of grids. Previous research has shown how the FAS can be incorporated into multi-grid schemes [8,66,69–72]. Here the main difference is that the fast adaptive second generation lifted interpolating wavelet transform is used for both recursive prolongation and restriction on a grid that adapts to the solution after each V-cycle iteration.

Let us take Eq. (24) to be the fine grid problem at level J and $\tilde{U}^J$ to be an approximate solution of the fine grid problem. Let $\mathscr{R}^J$ be the wavelet restriction operator defined by

$$U^{J-1} = \mathscr{R}^{J-1} U^J,$$

and let

$$r^J := F^J - L^J \tilde{U}^J \tag{25}$$

be the residual of the approximate solution at level J. A two level approach aims to compute a correction V (usually called *coarse grid correction*) to the approximate solution $\tilde{U}^J$ by solving the problem (*coarse grid problem*)

$$L^{J-1}U^{J-1} = \underbrace{L^{J-1}(\mathscr{R}^{J-1}\tilde{U}^J) + \mathscr{R}^{J-1}r^J}_{F^{J-1}}, \tag{26}$$

which is defined on the coarse grid. The coarse grid correction V is obtained by prolonging the estimated error on the coarse grid as

$$V \leftarrow \mathscr{R}^J \underbrace{(\tilde{U}^{J-1} - \mathscr{R}^{J-1}\tilde{U}^J)}_{\text{coarse grid error}}. \tag{27}$$

The solution at level J is thus updated by

$$U^J \leftarrow \tilde{U}^J + \mathscr{R}^J(\tilde{U}^{J-1} - \mathscr{R}^{J-1}\tilde{U}^J). \tag{28}$$

Thus, we need to restrict both the residual and the solution from fine grid to coarse grid. This is in contrast to the linear case, where only the residual is restricted [69].

An iterative method for solving a system of algebraic equations is called a *relaxation* technique if the high frequency components of error are reduced rapidly in a few iterations [70]. The general multi-grid strategy relies on the idea that the solution is relaxed at the fine scales. Since the smooth components of error appear to be non-smooth on coarser scale, recursive restriction of smooth error to coarser and coarser scale is carried out to reach the coarsest scale where the error of approximation is computed. A recursive prolongation procedure propagates coarse grid error to finer and finer grids. Finally, the fine grid approximation is updated to get the next iterate. One cycle from finest grid to coarsest grid and from coarsest grid to finest grid is called a multi-grid V-cycle iteration [69]. At each step of V-cycle restriction and prolongation, a suitable relaxation scheme is used to smooth out the high frequency component of the error.

The main difference between the linear V-cycle and the nonlinear V-cycle is that both the fine grid residual and the fine grid approximation are restricted to coarser grid. We have implemented a Newton smoother for the nonlinear Burgers equation, which is computationally more efficient because the Jacobian matrix is neither computed explicitly nor multiplied by any solution vector explicitly. A locally linearized operator is discretized to solve for the correction term in the Newton iteration. Vasilyev and Kevlahan [46] introduced the AWCM multi-level solver for elliptic problems, and showed how it differs from the classical linear elliptic multi-grid solver. Note that the AWCM elliptic solver and the space–time WFAS solver both solve PDEs on a near optimal grid such that the global error is controlled by the a priori tolerance. Conventional linear multi-grid solver and nonlinear FAS solver invert elliptic operators with reduced computational cost such that the global error is reduced depending on how fine the finest grid is. These methods start with a given fine grid and perform most of the computational work in coarser grids [73]. In contrast, our method starts with a coarser grid and, given an error tolerance, seeks an accurate solution on a near optimal refined grid. A detailed discussion of different smoothing strategies is beyond the scope of the present paper, where we are mainly interested on global error control and multiple time stepping.

3.6.1. Smoothing in the simultaneous space–time domain

We now briefly outline how the approximate solution to a given equation is relaxed in a simultaneous space–time domain. Let us re-write Eq. (24) for fixed resolution 2^{-J} as

$$\mathbf{f}(\mathbf{u}) = 0, \tag{29}$$

where $\mathbf{u} = U^J$ is the solution vector for a given J and $\mathbf{f}$ is a $\mathscr{N} \times 1$ vector. Let $\mathbf{v}$ denotes the error at scale 2^{-J}. One can determine $\mathbf{v}$ by solving the following linear equation:

$$\mathscr{J}(\mathbf{u})\mathbf{v} = -\mathbf{f}(\mathbf{u}),$$

where $\mathscr{J}$ is the Jacobian of f. The correction $\mathbf{u} \leftarrow \mathbf{u} + \mathbf{v}$ constitutes one step of Newton's iteration and the smoothing strategy is described in Algorithm 1 in terms of Eq. (24). Note that the method converges quadratically if provided with a 'good' initial guess. However, the computational cost of the error equation is larger than the overall computational cost of the AWCM (which is $O(\mathscr{N})$). This is clearly not efficient.

To improve the efficiency of the method, we introduce an alternate approach that requires only $O(\mathscr{N})$ computations to obtain the error equation, where we linearize the original PDE about an approximate solution. As an example, consider u^k to be an approximate solution of Burgers equation,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - v\frac{\partial^2 u}{\partial x^2} = 0,$$

which can be updated as $u^k \leftarrow u^k + v$ by solving

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial u}{\partial x} - v\frac{\partial^2 v}{\partial x^2} = -f(u)$$

for v. We now discretize this equation to produce $\mathscr{J}(\mathbf{u})\mathbf{v} = -\mathbf{f}(\mathbf{u})$. Thus the computational cost for the Jacobian equation remains the same as that of the wavelet transform. To complete the smoothing stage at each level of resolution, we perform 3 sweeps of weighted Jacobi iteration to compute $\mathbf{v}$, which is than followed by another 3 sweeps of Newton iteration to update $\mathbf{u}$.

Algorithm 1. Adaptive WFAS V-cycle iteration

WHILE $\|F^J - L^J U^J\|_2 \geqslant \delta$
  $r^J := F^J - L^J U^J$
  for $j = J{:}1{:}-1$
    DO ν_1 steps of smoothing
    RESTRICT solution and residual
  end
  Solve $L^j U^j = F^j$ for $j = j_{\min}$
  for $j = 1{:}J{:}1$
    ESTIMATE coarse grid error
    PROLONG error to fine grid
    DO ν_2 steps of smoothing
  end
  UPDATE solution $U^J \leftarrow U^J + V$
end

Although this approach works with reasonable efficiency with the fast lifted interpolated wavelet transform, difficulties arise if the nonlinear term has the form of $g(u)\frac{\partial u}{\partial x}$, where the explicit form of $g(u)$ is not known. This is the case in solving the vorticity form of the Navier–Stokes equation. In order to mitigate this difficulty, and reduce the computational effort due to the inversion of the Jacobian, let us solve each of the equations in the system (29) for separate components of $\mathbf{u}$. In other words, we propose to solve ith nonlinear equation for ith component of $\mathbf{u}$ using Newton's method. A given u_i is now updated by a few sweeps of

$$u_i \leftarrow u_i - \frac{f_i([\ldots u_i \ldots])}{\partial f_i/\partial u_i}, \tag{30}$$

where the second term in Eq. (30) can be thought as a local correction to each of the components of $\mathbf{u}$. In order to synchronize this local correction, we can incorporate this in Algorithm 2 by replacing the Jacobian with the diagonal matrix,

$$\mathscr{J}(\mathbf{u}) := \operatorname{diag}\left[\frac{\partial f_i}{\partial u_i}\right].$$

We have examined this technique (i.e. replacing the smoother in the WFAS algorithm by the solution of a nonlinear elliptic equation), and noticed that only one Newton's sweep in (30) is sufficient for convergence. In fact, for the linear case the method reduces to Jacobi iteration.

The WFAS strategy for nonlinearities is outlined in Algorithm 1, while the smoothing technique is outlined in Algorithm 2.

Algorithm 2. Adaptive Newton smoother. $\mathscr{J}$ is the Jacobian.

Given an initial guess U^J
DO v_1 steps
  Solve $\mathscr{J}(U^J)V = -F(U^J) := F^J - L^J U^J$
  Correct $U^J \leftarrow U^J + V$
end

3.7. Construction of adaptive space–time grid

In the finite element approach, an adaptive grid is constructed by refining and coarsening according to a posteriori local error indicators [20]. Mesh refinement in wavelet-based techniques is somewhat different. The multi-level structure of wavelet decomposition provides a natural framework for mesh coarsening and refinement, and grid adaptation is automatic [34]. A wavelet coefficient d_k^j, by measuring the local fluctuation of a function in the neighborhood of a collocation point x_k^j, acts as a local error estimator. As described earlier, the error-balancing grid adaptation strategy aims to find the best representation of a function (e.g. lowest L^2 norm error) for a given number of grid points $\mathscr{N}$. This is called an optimal $\mathscr{N}$-term approximation to the function. In the proposed method, an optimal $\mathscr{N}$-term to a parabolic PDE is sought in the whole space–time domain such that the L^2 norm residual error is less than a specified tolerance.

The method starts with a suitable initial space–time grid $\mathbb{G}^j$ and an initial guess for the solution of Eq. (1), which incorporates the Dirichlet boundary values. When j is small and the solution is intermittent the residual $\|u - u^j\|_2$ is large, but it decreases as $j \to \infty$. We now aim to find the smallest possible $\mathbb{G}^{j+1} \supset \mathbb{G}^j$ such that $\|u - u^{j+1}\|_2 = \alpha\|u - u^j\|_2$, where $\alpha \in (0, 1)$. The properties of the wavelet coefficients d_k^j guarantee that there exists a J such that $\mathbb{G}^{J+1} = \mathbb{G}^J$ contains the minimal set of $\mathscr{N}$ points [19]. On this grid, we obtain the best $\mathscr{N}$-term approximation for the given tolerance. Since the wavelets are located simultaneously in both space and time, the number of degrees of freedom $\mathscr{N}$ is also minimized in space and time and the global error is estimated by the magnitude of the smallest wavelet coefficients retained. Thus, accuracy is controlled not by the range of the wavelet spectrum (i.e. range of scales j), but rather by the intensity of the spectrum. This is in contrast to the spectral method where the accuracy is increased by increasing the range of the resolved spectrum (i.e. adding larger wavenumbers). For more technical details of constructing the optimal adaptive wavelet grid, we refer the readers to [46, and the references therein].

3.8. Flip and solve method

When the actual computational time interval is very large, the size of the algebraic problem on the space–time domain can exceed the available computer memory. To overcome this technical limitation, we propose to decompose the space–time domain into a collection of sub-domains in the time direction. Let us partition the interval

$$(0, T) = \bigcup_{n=1}^{n=N-1} (T_n, T_{n+1}) \quad \text{with } T_1 = 0, T_N = T. \tag{31}$$

Thus, the space–time domain $\Omega = (-1, 1) \times (0, T)$ can be decomposed into a collection of sub-domains as

$$\Omega := \bigcup_{n=1}^{n=N-1} (-1, 1) \times (T_n, T_{n+1}). \tag{32}$$

Let $\Omega_n = (-1, 1) \times (T_n, T_{n+1})$. A schematic diagram of this sub-division is shown in Fig. 3. Thus, the problem in Ω can be solved as a sequence of sub-problems in each of the domains Ω_n, $n = 1 \ldots N-1$. In other words, we solve a problem P_n in domain Ω_n with boundary condition at $t = T_n$ as Dirichlet type and at $t = T_{n+1}$ as evolution type. For fixed n, to solve a problem P_n in Ω_n, Dirichlet boundary values at $t = t_n$ are known from the solution of problem P_{n-1}. Since the boundary $t = T_n$ (with a non-uniform adapted grid) is common to domains Ω_n and Ω_{n-1}, we can flip the domain Ω_n and the problem P_n over Ω_{n-1} and P_{n-1} respectively. Thus, solving the problem P_n in domain Ω_n is equivalent to solving a new *flipped* problem P'_{n-1} in Ω_{n-1}. To this end we call P_n as forward problem and P'_{n-1} as backward problem. By proceeding in the same way, we can flip and put all the sub-domains over the domain Ω_1. This allows us to construct a sequence of problems on Ω_1 corresponding to each of the sub-domains. Note that we still retain global error control over the entire domain Ω.

In this construction the PDE remains forward if the orientation of the corresponding domain is unchanged, and a problem changes to backward if the corresponding domain is flipped.

Let us now call a problem is backward if we change the direction of t, consider the given spatial boundary conditions and switch the boundary conditions at $t = T_1$ with that at $t = T_2$. Once the actual problem is solved in Ω_1, we flip the problem, take the computed solution at $t = T_2$ as initial condition and leave the boundary at $t = T_1$ as evolution type. The solution of the flipped problem in Ω_1 is now the solution of the actual problem in Ω_2. The process is continued until the desired maximum T is achieved.

The flip-and-solve technique resolves the difficulty associated with the available computer memory (when the total space–time degrees of freedom is too large) by solving a sequence of space–time problems. This is a technical implementation of the proposed space–time AWCM and does not affect the convergence of the algorithm. It is apparent that the terminal solution in the first space–time grid is used as the initial condition in the second space–time grid. This injects an error of $O(\epsilon)$ in the increasing time direction. However, since the error in the multi-level adaptive wavelet approximation is controlled by the same wavelet thresholding parameter ϵ [61], this injected error remains local and does not accumulate globally. This has been verified by numerical experiments.

4. Results and discussion

4.1. Model problems

The accuracy and efficiency of the proposed numerical method is now verified by solving nonlinear parabolic PDEs with highly intermittent solutions. Since our ultimate goal is the study of high Reynolds number turbulent flows, we will consider three simplified problems that capture the main features of turbulence

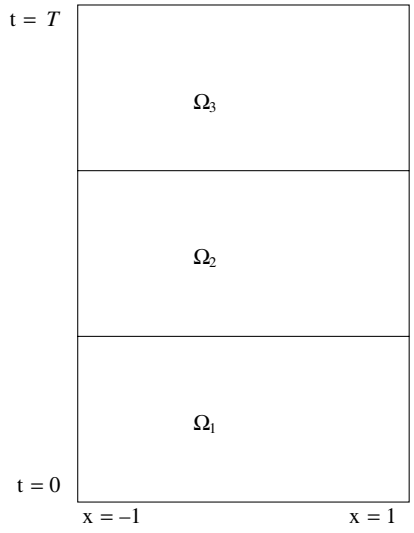

Fig. 3. Schematic diagram of the space–time domain and sub-domains.

dynamics. We first consider two related one-dimensional problems: the one-dimensional model of the Navier–Stokes equations known as Burgers equation [74] with fixed and moving shocks. These one-dimensional problems (solved on a $(1D + t)$ dimensional space–time domain) allow us to carefully evaluate the performance of the space–time algorithm. We are particularly interested in grid adaptation (e.g. local time step) and global error control in the space–time domain. We then analyze a much more realistic problem: the merger of two identical vortices at $Re = 1000$ calculated by solving the two-dimensional vorticity equations on a three-dimensional space–time domain. This problem shows that the space–time method performs well in higher dimensions on problems with more complex dynamics.

4.2. Problem formulation

4.2.1. Fixed shock

To examine the delicate balance between the nonlinear advection and the diffusion, we consider the boundary value problem for Burgers equation with periodic boundary condition in space (**x**-periodic), a smooth initial profile, and an evolution condition for the final time. The proposed forward and backward problems (notation explained in Section 3.8 are described below:

Forward problem. We construct a space–time forward boundary-value problem by considering long-time boundary as *evolution type*:

$$\begin{aligned} &-\nu\frac{\partial^2 u}{\partial x^2}(x,t)+u(x,t)\frac{\partial u}{\partial x}(x,t)+\frac{\partial u}{\partial t}(x,t)=0,\\ &u(-1,t)=u(1,t),\quad t\in(0,T),\\ &u(x,0)=\sin(\pi x),\quad x\in(-1,1)\\ &-\nu\frac{\partial^2 u}{\partial x^2}(x,T)+u(x,t)\frac{\partial u}{\partial x}(x,T)+\frac{\partial u}{\partial t}(x,T)=0. \end{aligned} \tag{33}$$

Let us call this solution $u_1(x, t)$ in the first space–time grid $[-1, 1]\times[0, T]$. The solution $u_2(x, t)$ in the second space–time grid $[-1, 1]\times[T, 2T]$ is obtained by solving the following backward problem.

Backward problem. We now construct a space–time backward boundary-value problem:

$$\begin{aligned} &-\nu\frac{\partial^2 u}{\partial x^2}(x,t)+u(x,t)\frac{\partial u}{\partial x}(x,t)-\frac{\partial u}{\partial t}(x,t)=0,\\ &u(-1,t)=u(1,t),\quad t\in(0,T),\\ &u(x,T)=u_1(x,T),\quad x\in(-1,1),\\ &-\nu\frac{\partial^2 u}{\partial x^2}(x,0)+u(x,t)\frac{\partial u}{\partial x}(x,0)-\frac{\partial u}{\partial t}(x,0)=0. \end{aligned}$$

The solution of the forward problem has a fixed shock that steepens as t increases with vanishing viscosity. Even with a relatively large viscosity considered here, $\nu = 10^{-3}/\pi$, the change from a uniformly smooth distribution to the shock structure is observed as early as $0 \leqslant t \leqslant 1/\pi$. This, in turn, results in the growth of the wavelet coefficients near $x = 0$ as t increases. Note that in order to resolve the shock structure we need to compute the solution up to scale J such that [26]

$$2^{-J} \leqslant [\partial u/\partial x]^{-1}. \tag{34}$$

In the above formulation, we consider the final time $T = 0.4$, which is sufficient for smooth initial condition to become highly intermittent.

The exact solution of the forward problem can be determined easily using the so-called *Cole–Hopf* transformation [75] and is given by

$$u_{\mathrm{ex}}(x,t) = -\frac{\int_{-\infty}^{+\infty}\sin(\pi(x-\eta))\exp\left(-\frac{\cos(\pi(x-\eta))}{2\pi\nu}\right)\exp\left(-\frac{\eta^2}{4\nu t}\right)\mathrm{d}\eta}{\int_{-\infty}^{+\infty}\exp\left(-\frac{\cos(\pi(x-\eta))}{2\pi\nu}\right)\exp\left(-\frac{\eta^2}{4\nu t}\right)d\eta}. \tag{35}$$

The exact solution $u_{ex}(x, t)$ plotted in Fig. 4 shows that Burgers equation develops a very intermittent solution in both space and time. The time scale is fastest where the gradient steepens.

4.2.2. Moving shock

We now study a problem where a localized structure (e.g. a shock) moves. By adding a constant speed to the inertial term of the Burgers equation with a shock type initial condition, we can construct a problem whose solution contains a shock that does not change in time, but moves in space according to the added constant speed. Although the mathematical construction of this problem is similar to the previous problem, additional numerical difficulties appear when an adaptive numerical method is used. The spatial grid should now refine and coarsen to follow the shock structure. In classical adaptive numerical methods, a moving mesh approach is the popular choice in this case [76]. In other words, a pre-calculated non-uniform grid is translated in order to follow the moving shock. In our case, the space–time grid adapts iteratively and automatically to resolve the fine scale structure in space and time.

Forward problem

$$\begin{aligned}
&-\nu\frac{\partial^2 u}{\partial x^2}+(v+u)\frac{\partial u}{\partial x}+\frac{\partial u}{\partial t}=0, \quad (x,t)\in(-1,1)\times(0,T),\\
&u(\pm 1,t)=\mp 1,\\
&u(x,0)=-\tanh\left(\frac{x-x_0}{2\nu}\right),\\
&-\nu\frac{\partial^2 u}{\partial x^2}+(v+u)\frac{\partial u}{\partial x}+\frac{\partial u}{\partial t}=0, \quad t=T,
\end{aligned} \tag{36}$$

Backward problem

$$\begin{aligned}
&-\nu\frac{\partial^2 u}{\partial x^2}+(v+u)\frac{\partial u}{\partial x}-\frac{\partial u}{\partial t}=0, \quad (x,t)\in(-1,1)\times(0,T),\\
&u(\pm 1,t)=\mp 1,\\
&-\nu\frac{\partial^2 u}{\partial x^2}+(v+u)\frac{\partial u}{\partial x}+\frac{\partial u}{\partial t}=0, \quad t=0,\\
&u(x,T)=u_1(x,T).
\end{aligned} \tag{37}$$

We solve this problem for $\nu=10^{-2}$, $x_0=0$, $v=1$, and $T=0.4$.

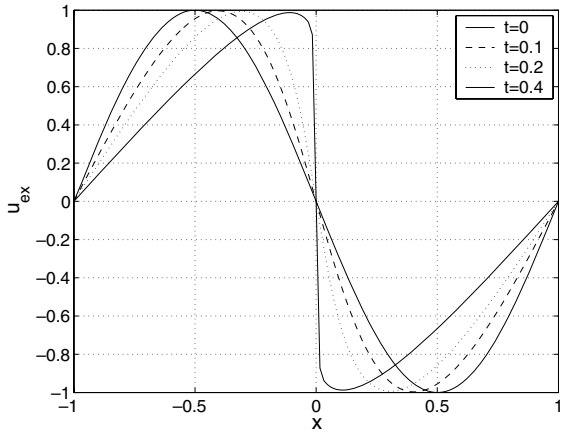

Fig. 4. Exact solution of the Burgers equation at different values of t.

4.2.3. Vortex merging in 2D turbulence

The space–time AWCM is applied here to study vortex merging, which is known to be a fundamental nonlinear process in two-dimensional turbulence [43]. We solve the two-dimensional Navier–Stokes equation written in the velocity–vorticity form,

$$\frac{\partial \omega}{\partial t} + \mathbf{u} \cdot \nabla \omega - \frac{1}{Re} \nabla^2 \omega = 0 \quad \text{in } \Omega. \tag{38}$$

In two-dimensional flow, the vorticity is confined to the z-direction, and is related to the velocity field via

$$\nabla \times \mathbf{u} = \omega.$$

One can show that the velocity is a functional of vorticity, given by the Biot–Savart law,

$$\mathbf{u}(\mathbf{x}, t) = -\frac{1}{2\pi} \int \frac{\mathbf{x} - \mathbf{y}}{|\mathbf{x} - \mathbf{y}|^2} \times \omega(\mathbf{y}, t) \mathrm{d}\mathbf{y}, \quad t \in [0, T], \tag{39}$$

where the integral is taken over the entire spatial domain [77,78]. Greengard and Rokhlin [79,80] have developed a Fast Multiple Method (FMM) that we use here to evaluate the above integral such that the velocity field is computed from the vorticity field throughout the space–time domain.

The initial condition is two identical Gaussian vortices with vorticity given by

$$\omega(x, y, 0) = \frac{\Gamma}{\pi \sigma^2} \exp[-((x - r\cos\phi)^2 + (y - r\sin\phi)^2)/\sigma^2],$$

where the circulation $\Gamma = 1$, the vortex radius $\sigma = \sqrt{\nu\pi^2}$, the vortex separation $2r = 1$, and the vortex centres are located at angles $\phi = \pm\pi/4$. The Reynolds number is $Re = \Gamma/\nu = 1000$. This Reynolds number is high enough that the merging process generates intense intermittent vorticity in the form of filaments.

The space–time domain Ω is the product of a doubly periodic spatial domain $D = [-2.5, 2.5] \times [-2.5, 2.5]$ and the time interval [0, 40]. As for the other cases, we use an evolution condition at the final time boundary. We construct a space–time grid with subdomains of size $\Delta T = 0.4$, and solve the problem recursively using the flip-and-solve algorithm discussed earlier. In the absence of an exact solution, we verify the performance of the numerical method by comparing it to similar AWCM which uses a classical time marching scheme [47].

4.3. Results for Burgers equation

4.3.1. Progressive grid adaptation and reduction of error

The ability of the numerical method to adapt the computational grid progressively is tested by setting the threshold parameter ϵ to some nonzero value, and solving the problem on a suitable initial coarse grid. As described in [46], the initial grid is refined iteratively by analyzing the wavelet coefficients and retaining only those whose values are larger than ϵ, along with a 'security zone' of nearest neighbor wavelets in position and scale. Wavelet coefficients are added where the solution has strong gradients (i.e. small space scales and fast time scales). This iterative procedure produces a sequence of grids and a sequence of solution surfaces, where the number of grid points increases at each iteration. Since the wavelet coefficients d_k^j vanish as the scale decreases for fixed position, after enough iterations the number of grid points and maximum scale J no longer increase. This means that the sequence of grids, and associated solution, has converged.

The smallest resolved scale in our computation is controlled by the threshold parameter ϵ. More importantly, in an error control based adaptive method the tolerance and the maximum number of grid points are fixed [2]. In other words, prescribing the tolerance ϵ also prescribes the smallest resolved scale. Thus, progressive grid adaptation procedure continues adding smaller and smaller scales until it finally reaches a steady state where the level of multi-resolution and number of active points do not change. To show that the number of degrees of freedom in a near optimal grid is fixed for a given tolerance, the scale as a function of iteration and the corresponding number of active wavelets as a function of iteration are plotted in Fig. 5 for the developing shock and in Fig. 6 for the moving shock.

In order to visualize the progressive grid adaptation, we perform simulations with an error tolerance of $\epsilon = 10^{-5}$ without fixing the maximum allowable resolution. Due to the presence of localized structures,

an initial space–time grid that is relatively coarse compared to the smallest active scale is refined progressively and grid points accumulate near the singularity. We present simultaneous space–time adaptive grids at various scales j with $\epsilon = 10^{-5}$ in Fig. 7(b). The accumulation of grid points in the region of intermittency and the non-uniformity of the time step in space shows the efficiency of the method.

Due to the presence of localized structures, large scale computations have large errors while the error decreases with scale. To show the progressive decrease of error with scale, we plot in Fig. 7(a) the solution at $t = 0$ and at $t = 0.4$ in various scales respectively for the developing shock. The solution and grid at the finest scale are shown in Fig. 8 for moving shock.

4.3.2. Global error control

One can easily show that the classical time marching procedure accumulates error progressively as time increases. To reduce the globally accumulated error, one reduces the time steps. Since the reduction in the time step is limited by machine precision, there is no control on the global error accumulation for arbitrarily long times. Analytical results shows that the global error in the domain where the wavelet transform is performed is bounded by the threshold parameter ϵ [34,46]. Thus, a wavelet-based simultaneous space–time adaptive method does not allow error to increase without bound. To test this, we compute the evolution of the L^p norm of the error $E(t) := \{\int_x |u(x,t) - u_{\mathrm{ex}}(x,t)|^p \mathrm{d}x\}^{1/p}$, and present the result with $p = \infty$ (i.e. the maximum error). In the developing shock problem, the sharp transition at fixed position reaches a maximum at some time t_c, and is then smoothed by viscous forces. Thus, as pointed out by Vasilyev et al. [26], the error should be high in the neighborhood of maximum gradient as the solution is computed from the best $\mathcal{N}$ term approximation determined by the wavelet coefficients. This is seen in the temporal evolution of error in both of the problems (see Fig. 9(a) and (b)).

As mentioned earlier, to efficiently use the available computer memory we implement a 'flip-and-solve' technique when the time interval is very large (Section 3.8). Note that the global error retains the same bound if we use the 'flip-and-solve' technique or use the entire domain. To show this we solve the developing shock problem by dividing the space–time domain $[-1, 1]\times[0, 0.4]$ into two equal sub-domains. The forward problem is solved in sub-domain $[-1, 1]\times[0, 0.2]$, and the backward problem is solved in the sub-domain $[-1, 1]\times[0.2, 0.4]$. In Figs. 10 and 11, we present the evolution of error in each of the domain as well as the corresponding space–time grid. Clearly, the method retains the same accuracy and a similar grid. Thus, we can use this algorithm to solve problems that must be integrated for arbitrarily long times. The compression of the grid in the flip-and-solve technique is qualitatively the same if we compare Figs. 10(a) and 11(a) with the bottom of Fig. 7(b).

To further analyze spurious globally accumulated error, we now solve the developing shock problem using a standard pseudo-spectral approach with de-aliasing, where the time marching scheme is a hybridization of a third-order Runge–Kutta for the nonlinear term and a Crank–Nicholson scheme for the linear viscous term.

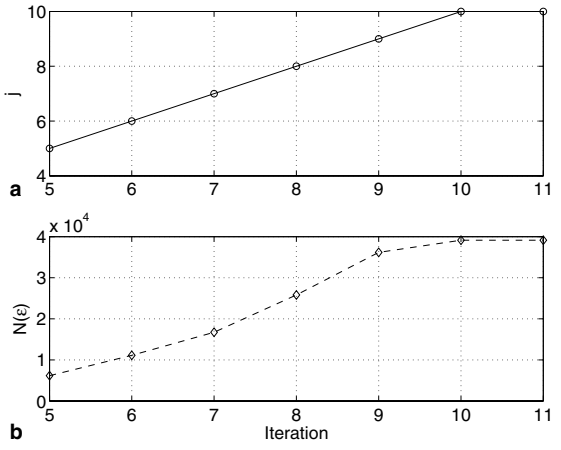

Fig. 5. For $\epsilon = 10^{-5}$, the algorithm reaches a natural scale $j = 10$, and the number of active wavelets $\mathcal{N}(\epsilon)$ adapts with the solution of the developing shock problem as the adaptive grid iteration proceeds.

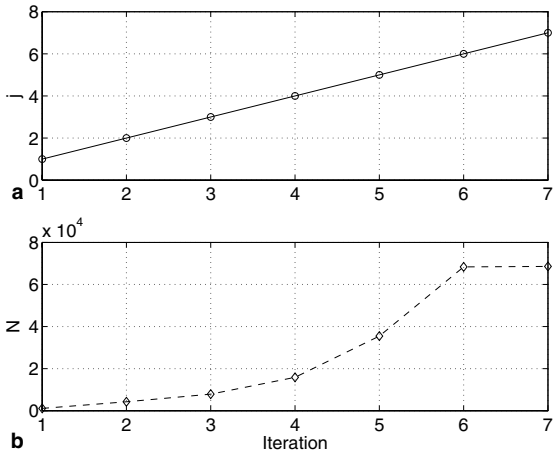

Fig. 6. For $\epsilon = 10^{-5}$, the algorithm reaches a natural scale $j = 7$ and the number of active wavelets $\mathcal{N}(\epsilon)$ adapts with the solution of the moving shock problem as the adaptive grid iteration proceeds.

Although, there is no a priori estimates for the globally accumulated error in a spectral time marching scheme, we expect that the leading order behavior of the global error is dominated by the temporal truncation error in the Crank–Nicholson scheme which is second order in time. Thus, if $\Delta x \sim \Delta t = O(10^{-3})$ then after marching 100 time steps, we expect that the error should accumulate at least to $O(10^{-4})$, or higher, since there are other sources of error. This accumulation of error is demonstrated in Fig. 12(a) where we compare the growth of L^∞ error in various time marching schemes (with fixed time step) to the space–time method. The error increases monotonically in time until about $t = 0.3$ in all the time marching schemes, while in the space–time method the error is controlled by wavelet thresholding parameter $\epsilon = 10^{-5}$. Fig. 12(b) shows that the error of the time marching schemes varies significantly with position, and is largest where the gradient of the solution is strongest (i.e. near $x = 0$). In contrast, the error of the space–time method is relatively homogeneous in space. The accumulation of error in the time marching methods could be slowed by using an adaptive time step, but it can never be eliminated entirely.

4.3.3. Accuracy and efficiency of the method

Eq. (13) predicts that increasing the number of grid points will decrease the global error by a factor of p/n, where n is the dimension of wavelet transform ($n = 2$ in this case) and p is the number of vanishing moment of the wavelet. Since n is fixed, increasing p decreases the error. On the other hand, decreasing ϵ increases the number of grid points (12), and thus reduces the error. Hence the best approximation of the solution is controlled by the parameters p, ϵ. To measure the accuracy of the proposed numerical method, we compute Eqs. (12) and (13) numerically for the fixed and moving shock problems for values of $p = 4, 6, 8$. As expected, increasing p improves the accuracy of the method. In Fig. 13, we present numerical results which agree with the theoretical predictions in Eqs. (12) and (13).

The asymptotic global error is given in terms of the threshold parameter ϵ by Eqs. (12) and (13). We present numerical results to verify the asymptotic error estimates [46] given by Eq. (13) in Fig. 13(a), where the pointwise L_∞ norm of the error in simulations with tolerance $\epsilon = 10^{-5}$ is plotted for $p = 4$, 6, 8. In Fig. 13(b), we present the effect of thresholding on the number of active collocation points as predicted by Eq. (12). This shows that the numerical method converges with sufficient accuracy, as predicted by the analytical error estimates.

4.4. Results for vortex merging

We now present numerical results for merger of two Gaussian vortices. The initial conditions and the computational domain are described in Section 4.2.3. The algorithm refines or coarsens the spatial mesh and the time step by the same factor, i.e. if Δx is changed to $\Delta x/2$ then Δt is changed to $\Delta t/2$. We therefore start with

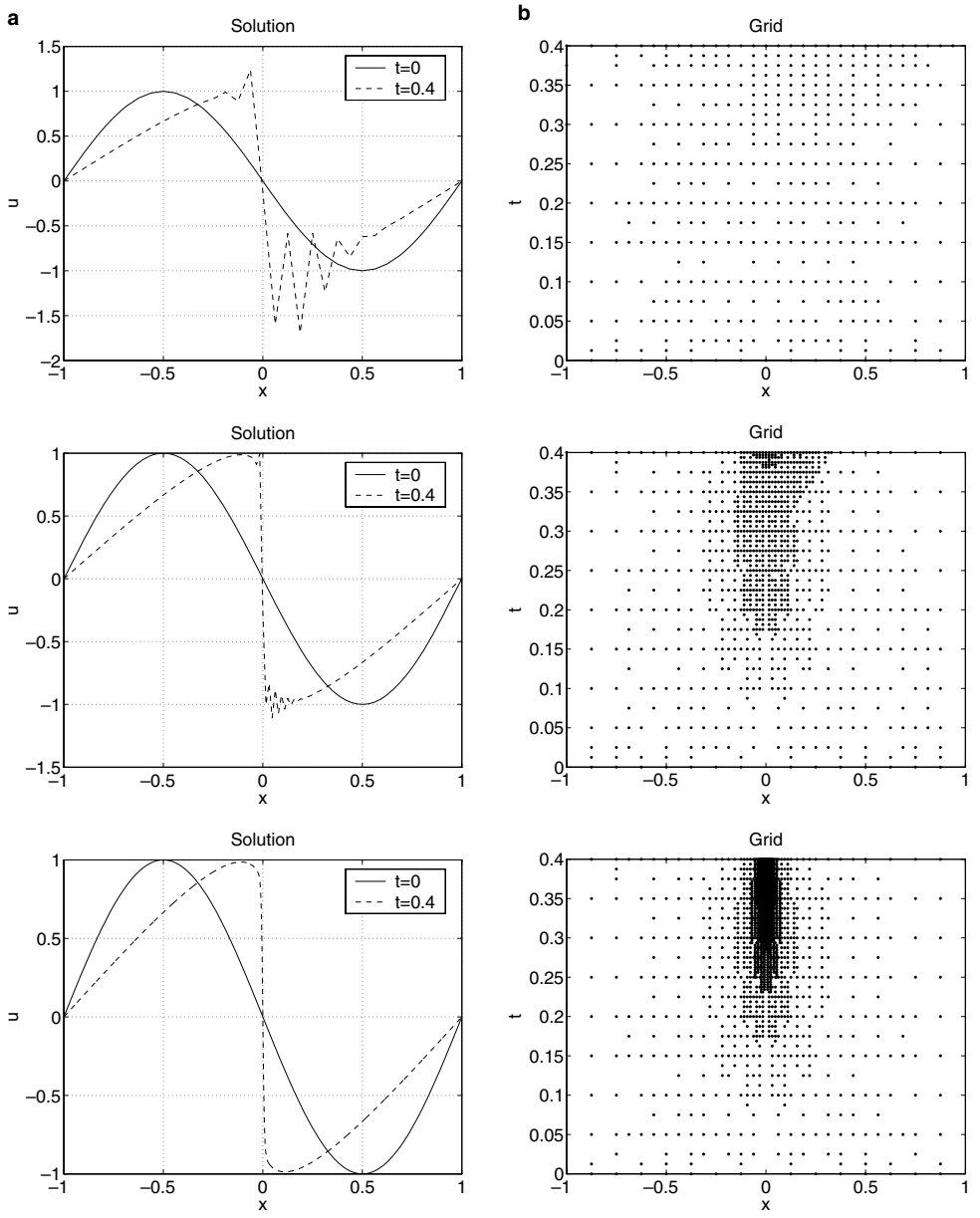

Fig. 7. Developing shock at scales $j = 3, 5, 9$ (top to bottom): (a) solution; (b) simultaneous space–time adaptive grid.

an initial coarse mesh with sufficient spatial resolution to resolve the initial condition and such that $\Delta t/\Delta x \sim \min(|\boldsymbol{u}|^{-1})$, where $\boldsymbol{u}$ is the velocity field. This CFL-type condition is not a strict requirement for convergence, and is only necessary to ensure that the grid resolution in space and time is optimal. One could, in principle, start with a $2 \times 2 \times 2$ mesh and refines the spatial domain such that an acceptable ratio of the refinement factor between $\boldsymbol{x}$ and t is obtained, which is the same as having a fixed ratio between Δx and Δt. Since no information is available for $t > 0$ at this moment, simply adapting the space–time mesh to the initial condition is not appropriate. Moreover, the CFL-type criterion is reasonable for the simulation of advection dominated flows.

In this simulation we use an initial space–time grid of size $32 \times 32 \times 2$ (see Fig. 16(a)). We allow three levels of grid refinement, which means the maximum resolution in any sub-domain of size $[-2.5, 2.5] \times [-2.5, 2.5] \times$

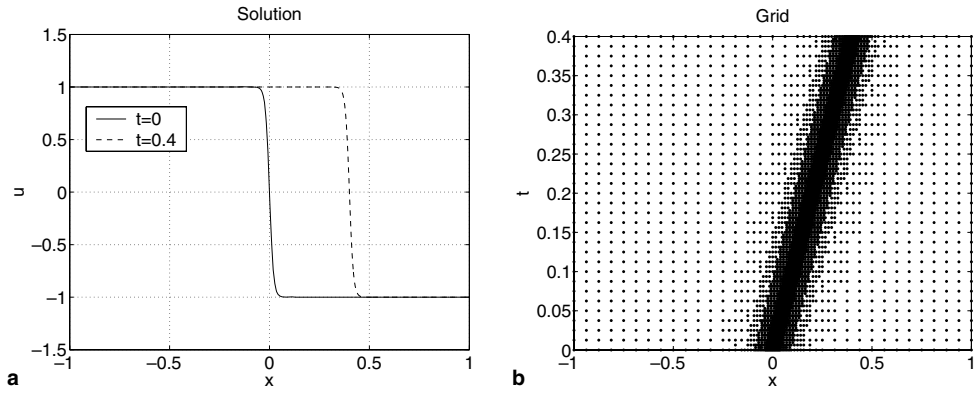

Fig. 8. Moving shock: (a) solution; (b) simultaneous space–time adaptive grid.

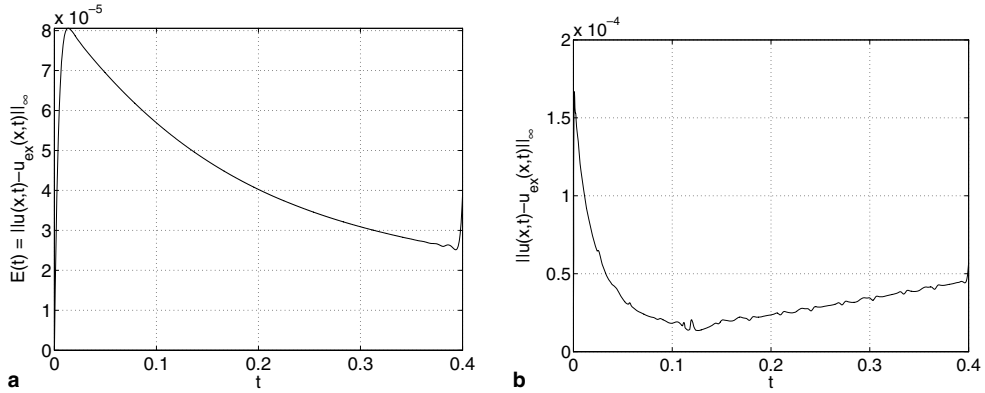

Fig. 9. Global error in time: (a) developing shock; (b) moving shock.

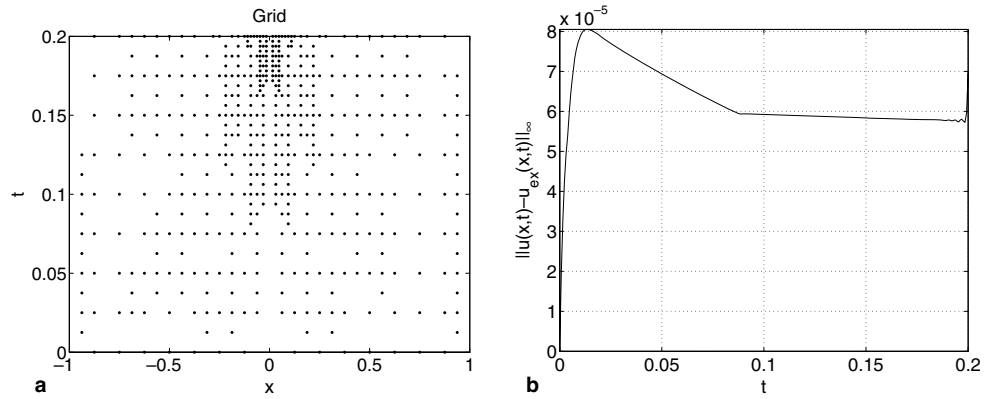

Fig. 10. Flip-and-solve results for first sub-domain: (a) simultaneous space–time adaptive grid of $[-1, 1]\times[0, 0.2]$; (b) global error in time.

$[0, 0.4]$ is $256 \times 256 \times 16$. The wavelet thresholding parameter $\epsilon = 10^{-5}$. Fig. 16(b) is the space–time grid of the first space–time sub-domain in the flip-and-solve algorithm, and Fig. 16(c) is the grid of the last space–time sub-domain. This result shows that the time steps are local and distributed according to the temporal intermittency of the solution. To the best of our knowledge, this is the first two-dimensional DNS of the vorticity or Navier–Stokes equations that adapts the time step to match the natural local time scale.

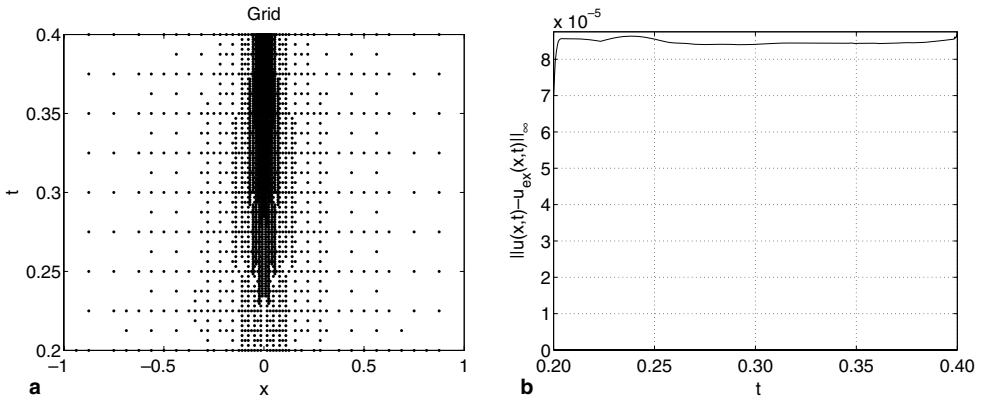

Fig. 11. Flip-and-solve results for second sub-domain: (a) simultaneous space–time adaptive grid of $[-1, 1]\times[0.2, 0.4]$; (b) global error in time.

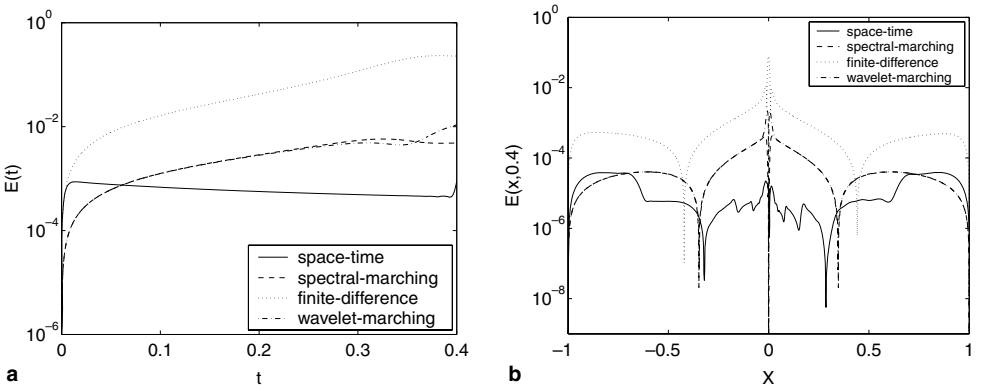

Fig. 12. (a) Global L^∞ error in time for the developing shock problem. Comparison of the space–time method with various time marching methods (with fixed time step). (b) Absolute difference between the numerical solutions and the exact solution at $t = 0.4$. (Note that errors for the spectral and wavelet time marching simulations largely overlap.)

Note that the vortex merging simulation is significantly different from the two previous problems because of the need to resolve both the velocity and vorticity field in the space–time domain. The evolution of the vorticity field is determined by solving the vorticity equation in simultaneous space–time domain and by performing the FMM calculation at each V-cycle Wavelet-based Full Approximation Scheme (WFAS) to find the velocity, which is then used for next V-cycle iteration. Once the WFAS calculations have converged, we use this solution to estimate the error and to adapt the space–time grid according to the algorithm described by Vasilyev and Kevlahan [46].

We have compared the space–time results with the results of similar AWCMs which use adaptive wavelets for spatial discretization and an adaptive time marching scheme in the temporal direction [47]. We consider two time marching schemes: an explicit Krylov method [81] and an implicit Crank–Nicolson method. Although all methods use the same adaptivity in space, the global error of the space–time method is bounded by ϵ, while the time marching schemes control only the local time integration error. For consistency the tolerances of the time marching schemes are set such that the global time integration error remains $O(\epsilon)$ (i.e. we set the tolerance such that the total number of time steps times the local error is $O(\epsilon)$, with $\epsilon = 10^{-5}$). Note that the number of time steps increases with decreasing tolerance, and it is therefore not surprising that we must use extremely small tolerances to match the global error of the space–time method. We show the vorticity field computed at $t = 0.2$, $t = 9.6$, and $t = 25.2$ in Fig. 14. The spatial grids at corresponding time levels are shown in Fig. 15 for both methods. Note the similarity of the final spatial grid in both methods,

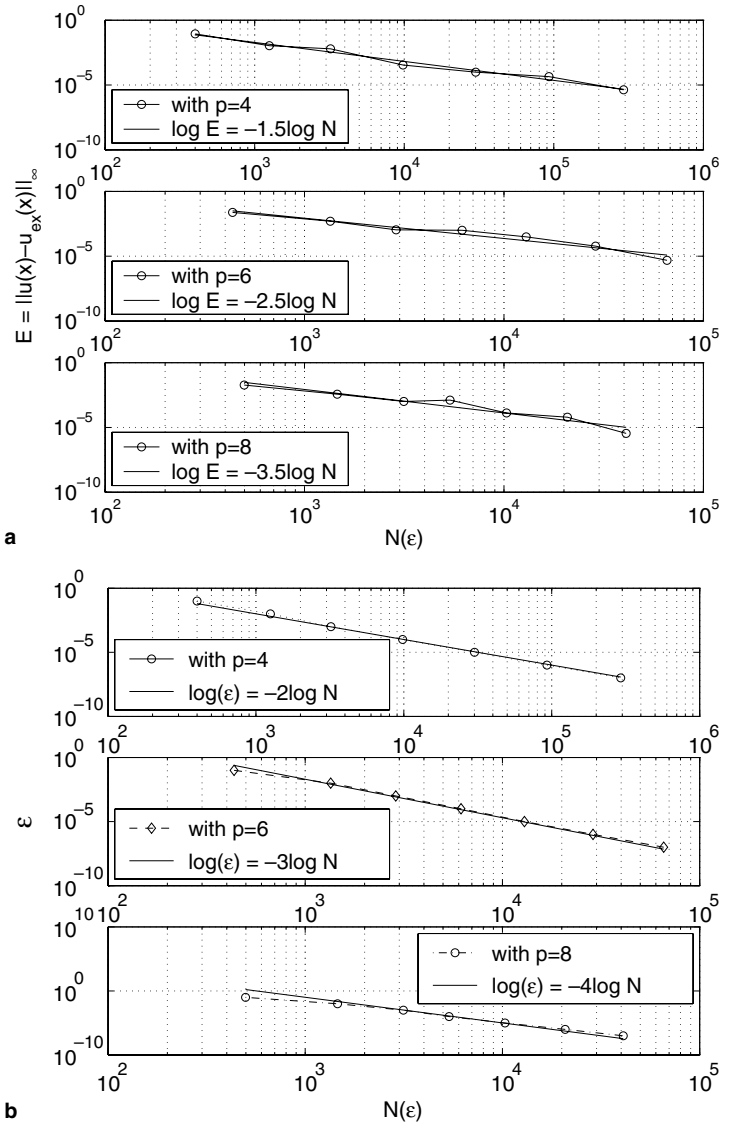

Fig. 13. (a) Point-wise L_∞ error in the solution of the fixed problem as a global function of active number of wavelets with $p = 4$, 6,8. (b) Threshold parameter ϵ as a function of active number of wavelets for $p = 4$, 6, 8.

despite the very different numerical algorithms used. These results provide a good qualitative assessment of the accuracy of the proposed numerical method compared with a standard adaptive time marching simulation.

4.5. Comparison of computational complexity and cost

One of the main contributions of the space–time method is to achieve a prescribed global space–time accuracy with reduced computational cost. Table 1 summarizes the total number of grid points, total CPU time, and the minimum and maximum Δt used in the entire simulation for each of the three methods. The number of active wavelets $\mathcal{N}$ (independent of the dimensionality) measures the computational complexity of a wavelet-based adaptive DNS technique. Table 1 shows that the space–time method uses about 7 times fewer space–time grid points than the Crank–Nicolson scheme, and about 18 times fewer than the Krylov method. The reduction in CPU time is not as large, although it is still significant: the space–time method is about twice as fast as the Crank–Nicolson method and about four times as fast as the Krylov method. There is clearly

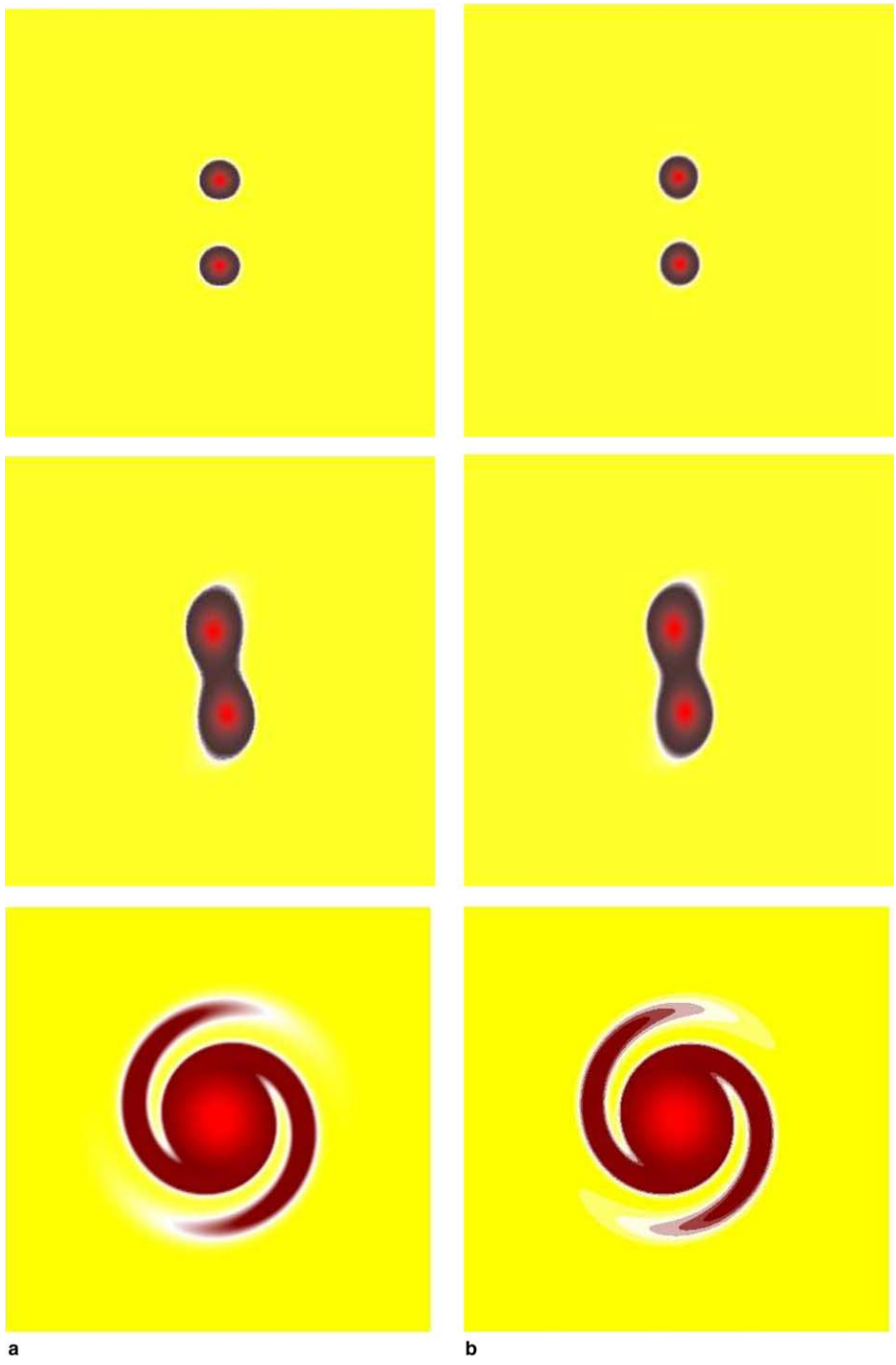

Fig. 14. Vortex merging at $Re = 1000$, vorticity field at times $t = 0.2, 9.6, 25.2$ (from top to bottom): (a) space–time AWCM solution; (b) Krylov time marching AWCM solution.

additional overhead associated with the iterative solution of the vorticity equation on the space–time domain. It is also interesting to compare the minimum time steps used in each method. Table 1 shows that the space–time method uses a minimum time scale roughly 2×10^3 times larger than the Crank–Nicolson method and 500 times larger than the Krylov method while achieving a similar accuracy. Note that the smallest time steps

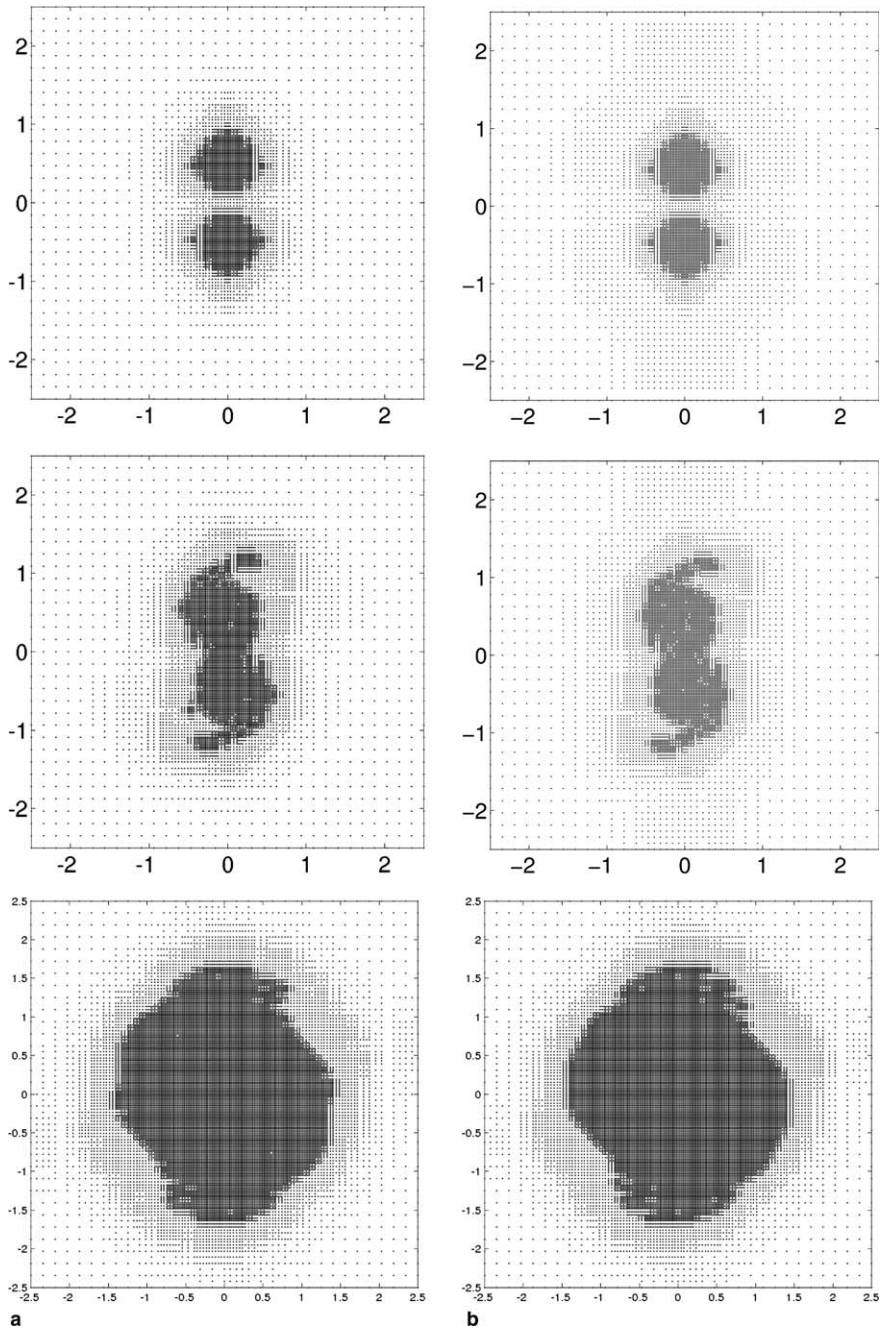

Fig. 15. Vortex merging at $Re = 1000$, grids at $t = 0.1, 9.6, 25.2$ (from top to bottom): (a) space–time AWCM; (b) Krylov time marching AWCM.

are applied only locally in the space–time method, while in the time marching methods they are applied homogeneously to all grid points.

Let us define the ratios of grid points in a single space–time subdomain $R_{\mathrm{CN}}(t) = \mathscr{N}_{\mathrm{CN}}(t)/\mathscr{N}_{\mathrm{ST}}(t)$ and $R_{\mathrm{KRY}}(t) = \mathscr{N}_{\mathrm{KRY}}(t)/\mathscr{N}_{\mathrm{ST}}(t)$. The space–time subdomains are $[-2.5, 2.5]\times[-2.5, 2.5]\times[t^n, t^n + 0.4]$, $n = 0\ldots 100$, $t^0 = 0$. ST denotes the space–time method, CN denotes the Crank–Nicolson method, and KRY denotes the Krylov method. The plots of $R_{\mathrm{CN}}(t)$ and $R_{\mathrm{KRY}}(t)$ in Fig. 17 show how the computational complexity of the

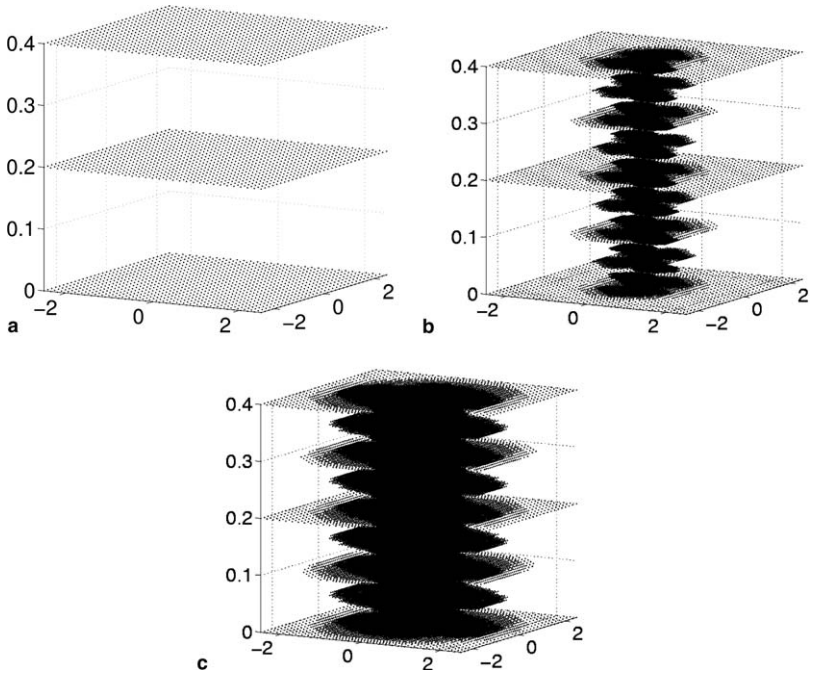

Fig. 16. Development of the space–time adaptive grid: (a) initial space–time grid; (b) first space–time grid; (c) 100th space–time grid.

Table 1
Comparison of the computational complexity and cost of the space–time and time marching DNS

	Space–time	Crank–Nicolson	Krylov
$\mathcal{N}$	25,041,353	174,823,834	455,960,480
Ratio of grid points	1	7	18
CPU time (s)	6.4×10^{4}	1.5×10^{5}	2.5×10^{5}
Ratio of CPU times	1	2.3	3.9
$\Delta t_{\min}$	2.35×10^{-2}	1.2×10^{-5}	4.8×10^{-5}
$\Delta t_{\max}$	2.0×10^{-1}	3.5×10^{-2}	6.3×10^{-3}
Number of time steps	$\leqslant$1600	14234	27115

Note that the equivalent number of time steps taken in the space–time method depends on the spatial location.

time marching schemes change with respect to the space–time method. It is interesting to note that the computational complexity of the Crank–Nicolson scheme decreases roughly monotonically, while that of the Krylov scheme reaches a peak at around $t = 20$ before finally decaying. This is likely due to the more accurate error control scheme of the Krylov method [81], since both schemes are unconditionally stable. The time step selected by the Crank–Nicolson scheme simply follows the average diffusion of vorticity (which decreases the average length scale), while the Krylov scheme decreases the time step when the dynamics are most rapid, i.e. during the intense vorticity filamentation that accompanies merger at about $t = 20$. Not surprisingly, the space–time approach is most advantageous when the dynamics are most intermittent (i.e. during filamentation). At long times the vorticity simply diffuses and the flow is no longer intermittent in time.

The time marching methods use more active grid points than the space–time method because they do not exploit temporal intermittency (i.e. the time step is the same for a locations) and they require much smaller time steps to achieve the specified global time integration error. Note that if the natural time scale were uniform over the entire spatial mesh (as is the case at long times), all methods should use roughly the same number of grid points at a given time. Fig. 17 therefore demonstrates that the proposed method is promising for highly intermittent problems. The reduction in the number of degrees of freedom is achieved using wavelet compression in the time direction, which adjusts the time step according to the natural local time scale of

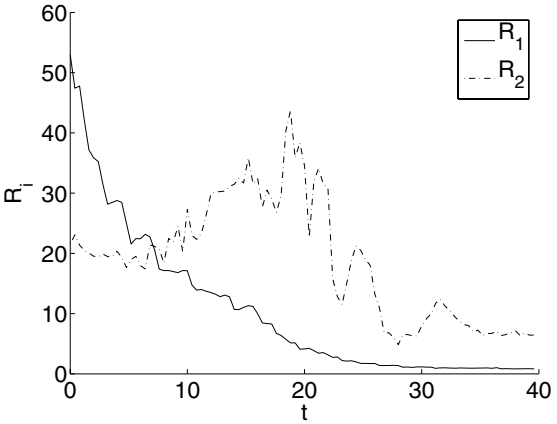

Fig. 17. The ratio of the number of grid points in each space–time sub-domain as a function of time for the vortex merging simulation. $R_{\mathrm{CN}}(t)$ is the ratio of Crank–Nicolson to space–time, and $R_{\mathrm{KRY}}(t)$ is the ratio of Krylov to space–time.

the flow. In adaptive time stepping, time steps are determined according to the smallest time scale at a fixed time. Thus, the slow time scale regions use an inappropriately small time step. In an efficient numerical method the total computational cost should be proportional to the actual number of degrees of freedom of the dynamical system. The space–time method attempts to achieve this.

5. Conclusions

The development of wavelet theory and its application to a wide variety of scientific problems has been an active research area for the last two decades [82]. Although it has been shown that adaptive wavelet compression gives an optimal $\mathcal{N}$-term approximation, and differential operators can be compressed using wavelets, no one has previously attempted to solve initial value problems using an adaptive wavelet basis in both space and time. Dynamical systems governed by nonlinear ordinary differential equations are often difficult to study numerically since conventional numerical schemes suffer not only from globally accumulated error, but are also not easy to adapt to multiple time stepping (i.e. different time steps for different scales or physical locations). A common example of the first problem is the numerical simulation of advection equations for arbitrarily long times [83,84]. Any Eulerian scheme is affected by spurious diffusive and dispersive error due to inaccurate spatial discretization [85]: this error becomes intolerable if the time interval is very long [83]. Multiple time scales are essential for the efficient simulation of equations which have highly intermittent structure, such as turbulent flows.

The main contribution of this paper is the development of a simultaneous space–time adaptive wavelet collocation method (AWCM) for nonlinear PDEs. The space–time AWCM provides an elegant solution to both global error control in time and multiple time stepping. It is based on the multi-level AWCM for elliptic equations developed by Vasilyev and Kevlahan [46] extended to nonlinear problems using the multi-grid full approximation scheme (FAS) [8]. The PDE is reduced to a single algebraic problem which is solved simultaneously on a space–time domain with appropriate boundary conditions. If necessary, the time domain can be split into sub-domains using the flip-and-solve method described in the paper. This is useful for problems where using a very large domain in the time direction is impractical due to memory constraints, or where the stopping time is not known a priori. The method has been illustrated here by using it to solve the $1D + t$-dimensional Burgers equation for fixed and moving shocks, and the $2D + t$-dimensional vorticity equation for merging vortices at $Re = 1000$. In the vortex merging example, we found that the space–time method uses roughly 18 times fewer space–time grid points and is roughly 4 times faster than a dynamically adaptive Krylov time marching method, while achieving similar global accuracy. The decrease in the number of grid points is due to two properties of the space–time method: local time-stepping (i.e. the size of time step adapts locally in space), and global control of the time integration error (i.e. fewer time steps are required since the time integration error is controlled globally and thus does not accumulate). The extension of the method to

three spatial dimensions is straightforward in principle, but requires an efficient four-dimensional data-structure and parallelization of the algorithm. We are currently working on this extension of the method to higher dimensions.

The examples presented here show that the efficiency and accuracy of the method is consistent with theoretical predictions. The algorithm finds the solution of nonlinear evolution problems on a near optimal grid to a prescribed tolerance controlled by the wavelet threshold parameter ϵ. In addition, the proposed numerical method also provides global error control in time, something which is impossible in conventional time-stepping schemes.

We have shown that one can apply a space–time wavelet adaptive method to compute the intermittent solution of nonlinear evolution problems on a near optimal grid in one and two spatial dimensions. Thus, this method should be well-suited to direct numerical simulation (DNS) of turbulent flows. A naive estimate based on the Kolmogorov micro-scale predicts that the number of active degrees of freedom in a DNS of turbulent flow scales like Re^3 in space–time domain, where Re is the Reynolds number and a uniform space–time grid is assumed. However, when the flow is fully turbulent (and hence highly intermittent) the actual number of degrees of freedom is much smaller than this naive estimate. We expect that a simultaneous space–time AWCM will approximate the actual number of active degrees of freedom in such flows much better than classical time-stepping methods. In fact, we intend to use the number of active wavelets in the space–time domain to estimate of how the number of degrees of freedom actually scales with Reynolds number. We would also like to apply the present method to a dynamical system that involve a wide range of time scales, such as a set of coupled reaction–diffusion equations involving chemical reactions with widely varying time scales.

Acknowledgements

JMA and NKRK would like to acknowledge support from NSERC and SHARCNET. Partial support for OVV was provided by the National Science Foundation (NSF) under grants no. EAR-0327269 and ACI-0242457 and National Aeronautics and Space Administration (NASA) under grant no. NAG-1-02116. We would like to thank Daniel E. Goldstein for helpful discussions and comments.

References

[1] P. Fletcher, S. Haswell, V. Paunov, Theoritical considerations of chemical reactions in micro-reactors operating under electro-osmotic and electrophoretic control, Analyst 124 (1999) 1273–1282.
[2] S. Muller, Adaptive Multiscale Schemes for Conversation LawsLecture Notes in Computational Science and Engineering, vol. 27, Springer, Berlin, 2003.
[3] D. Leslie, Developments in the Theory of Turbulence, Clarendon Press, Oxford, 1973.
[4] H. Tennekes, J.L. Lumley, A First Course in Turbulence, MIT Press, 1976.
[5] J. Mathieu, J. Scott, An Introduction to Turbulent Flow, Cambridge University Press, Cambridge, 2000.
[6] N.K.-R. Kevlahan, J.-M. Ghidaglia, Computation of turbulent flow past an array of cylinders using a spectral method with brinkman penalization, Eur. J. Mech. B – Fluids 20 (2001) 333–350.
[7] M. Griebel, F. Koster, Multiscale methods for the simulation of turbulent flows, in: E. Hirschel (Ed.), Numerical Flow Simulation III, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, vol. 82, Springer, Berlin, 2002, pp. 203–214.
[8] A. Brandt, Multi-level adaptive solutions to boundary-value problems, Math. Comput. 31 (1977) 333–390.
[9] I. Babuska, W.C. Rheinboldt, Error estimates for adaptive finite element computations, SIAM J. Numer. Anal. 15 (1978) 736–754.
[10] I. Babuska, W.C. Rheinboldt, A posteriori error estimates for finite element methods, Int. J. Numer. Math. Engrg. 12 (1978) 1597–1615.
[11] R.E. Bank, A multi-level iterative method for nonlinear elliptic equations, in: M.H. Schultz (Ed.), Elliptic Problem Solvers, Academic Press, New York, 1981, pp. 1–16.
[12] R.E. Bank, A. Weiser, Some a posteriori error estimators for elliptic partial differential equations, Math. Comput. 44 (1985) 283–301.
[13] R. Becker, C. Johnson, R. Rannacher, Adaptive error control for multigrid finite element methods, Computing 55 (1995) 271–288.
[14] F.A. Bornemann, B. Erdmann, R. Kornhuber, A posteriori error estimates for elliptic problems in two and three space dimensions, SIAM J. Numer. Anal. 33 (1996) 1188–1204.
[15] R. Verfurth, A posteriori error estimation and adaptive mesh-refinement techniques, J. Comput. Appl. Math. 50 (1994) 67–83.
[16] R. Glowinski, A. Rieder, R.O. Wells Jr., X. Zhou, A preconditioned CG-method for wavelet-Galerkin discretizations of elliptic problems, Z. Angew. Math. Mech. 75 (1995) 683–684. Available from: <http://citeseer.ist.psu.edu/glowinski94preconditioned.html>.
[17] A. Cohen, R. Masson, Wavelet methods for second order elliptic problems – preconditioning and adaptivity, Preprint R 97036, Univ. P. et M. Curie, Paris, 1997. Available from: <http://citeseer.ist.psu.edu/article/cohen97wavelet.html>.

[18] S. Dahlke, W. Dahmen, R. Hochmuth, R. Schneider, Stable multiscale bases and local error estimation for elliptic problems, Applied Numerical Mathematics: Transactions of IMACS 23 (1) (1997) 21–47. <http://citeseer.ist.psu.edu/dahlke96stable.html> .
[19] A. Cohen, W. Dahmen, R. DeVore, Adaptive wavelet methods for elliptic operator equations: convergence rates, Math. Comput. 70 (233) (2001) 27–75. Available from: <http://citeseer.ist.psu.edu/article/cohen98adaptive.html>.
[20] A. Barinka, T. Barsch, P. Charton, A. Cohen, S. Dahlke, W. Dahmen, K. Urban, Adaptive wavelet schemes for elliptic problems implementation and numerical experiments, SIAM J. Sci. Comput. 23 (3) (2001) 910–939. Available from: <http://www.siam.org/journals/sisc/23-3/36550.html>.
[21] J. Liandrat, P. Tchamitchian, Resolution of the 1d regularized burgers equation using a spatial wavelet approximation, Technical Report NASA Contractor Report 187480, NASA Langley Research Center, Hampton VA, 1990.
[22] E. Bacry, S. Mallat, G. Papanicolaou, A wavelet based space–time adaptive numerical method for partial differential equations, Math. Modell. Numer. Anal. 26 (1992) 793–834. Available from: <http://citeseer.ist.psu.edu/100641.html>.
[23] A. Harten, Adaptive multiresolution schemes for shock computations, J. Comput. Phys. 115 (1994) 319–338.
[24] A. Harten, Multiresolution algorithms for the numerical solution of hyperbolic conservation laws, Comm. Pure Appl. Math. 48 (1995) 1305–1342.
[25] O.V. Vasilyev, S. Paolucci, M. Sen, A multilevel wavelet collocation method for solving partial differential equations in a finite domain, J. Comput. Phys. 120 (1995) 33–47.
[26] O.V. Vasilyev, S. Paolucci, A dynamically adaptive multilevel wavelet collocation method for solving partial differential equations in a finite domain, J. Comput. Phys. 125 (1996) 498–512.
[27] W. Cai, J.Z. Wang, Adaptive multiresolution collocation methods for initial boundary value problems of nonlinear PDEs, SIAM J. Numer. Anal. 33 (1996) 937–970.
[28] O.V. Vasilyev, S. Paolucci, A fast adaptive wavelet collocation algorithm for multidimensional PDEs, J. Comput. Phys. 138 (1997) 16–56.
[29] G. Beylkin, J.M. Keiser, On the adaptive numerical solution of nonlinear partial differential equations in wavelet bases, JCP 132 (CP965562) (1997) 233–259.
[30] J. Frhlich, K. Schneider, An adaptive wavelet-Vaguelette algorithm for the solution of PDEs, J. Comput. Phys. 130 (2) (1997) 174–190.
[31] L. Jameson, A wavelet-optimized, very high order adaptive grid and order numerical method, SIAM J. Sci. Comput. 19 (6) (1998) 1980–2013.
[32] M. Holmstrom, J. Walden, Adaptive wavelet methods for hyperbolic PDEs, J. Sci. Comput. 13 (1998) 19–49.
[33] M. Holmstrom, Solving hyperbolic PDEs using interpolating wavelets, SIAM J. Sci. Comput. 21 (2) (1999) 405–420.
[34] O.V. Vasilyev, C. Bowman, Second-generation wavelet collocation method for the solution of partial differential equations., J. Comput. Phys. 165 (2000) 660–693.
[35] O.V. Vasilyev, Solving multi-dimensional evolution problems with localized structures using second generation wavelets, in: Int. J. Comput. Fluid Dyn., vol. 17 of Special issue on High-resolution methods in Computational Fluid Dynamics, 2003, pp. 151–168.
[36] W. Sweldens, The construction and application of wavelets in numerical analysis, Ph.D. Thesis, University of Belgium, 1995.
[37] R. Glowinski, W. Lawton, M. Ravachol, E. Tenenbaum, Wavelet solution of linear and nonlinear elliptic, parabolic and hyperbolic problems in one dimension, in: R. Glowinski, A. Lichnewski (Eds.), Proceedings of the Ninth International Conference on Computing Methods in Applied Sciences and Engineering, SIAM, Philadelphia, 1990, pp. 55–120.
[38] W. Lawton, W. Morrell, E. Tenenbaum, J. Weiss, The wavelet-Galerkin method for partial differential equations, Technical Report, Aware, Inc., 1990.
[39] S. Mallat, W.L. Hwang, Singularity detection and processing with wavelets, Technical Report 549, Courant Institute of Mathematical Sciences, New York University, March 1991.
[40] K. Amaratunga, J.R. Williams, S. Qian, J. Weiss, Wavelet-Galerkin solutions for one-dimensional partial differential equations, Int. J. Numer. Methods Engrg. 27 (1994) 2703–2716. Available from: <http://citeseer.ist.psu.edu/amaratunga92waveletgalerkin.html>.
[41] R. Glowinski, A. Rieder, R.O. Wells Jr., X. Zhou, A wavelet multigrid preconditioner for dirichlet boundary value problems in general domains, Modelisation Mathematique et Analyse Numerique. Available from: <http://citeseer.ist.psu.edu/glowinski96wavelet.html>.
[42] R.O. Wells Jr., X. Zhou, Wavelet solutions for the Dirichlet problem, Numer. Math. 70 (3) (1995) 379–396. Available from: <http://citeseer.ist.psu.edu/jr95wavelet.html>.
[43] K. Schneider, N.K.-R. Kevlahan, M. Farge, Comparison of an adaptive wavelet method and nonlinearly filtered pseudo-spectral methods for two dimensiional turbulence, Theoret. Comput. Fluid Dyn. 9 (1997) 191–206.
[44] A. Debussche, T. Dubois, R. Temam, The nonlinear Galerkin method: a multi-scale method applied to the simulation of homogeneous turbulent flows, ICASE Rechnical Report(93–93).
[45] W. Sweldens, The lifting scheme: a construction of second generation wavelets, SIAM J. Math. Anal. 29 (2) (1997) 511–546.
[46] O.V. Vasilyev, N.-R. Kevlahan, An adaptive multilevel wavelet collocation method for elliptic problems., J. Comput. Phys. 206 (2005) 412–431.
[47] O.V. Vasilyev, N.K.-R. Kevlahan, Hybrid wavelet collocation-Brinkman penalization method for complex geometry flows. Int. J. Numer. Math. Fluids 40.
[48] N. Kevlahan, O.V. Vasilyev, D. Goldstein, A. Jay, A three-dimensional adaptive wavelet method for fluid–structure interaction, in: B.J. Geurts, R. Friedrich, O. Métais (Eds.), Direct and Large-Eddy Simulation V, 2004, pp. 147–154.
[49] N.K.-R. Kevlahan, O.V. Vasilyev, An adaptive wavelet collocation method for fluid–structure interaction at high Reynolds numbers, SIAM J. Sci. Comput. 26 (6) (2005) 1894–1915.

[50] D.A. Goldstein, O.V. Vasilyev, Stochastic coherent adaptive large eddy simulation method, Phys. Fluids 16 (7) (2004) 2497–2513.
[51] J.M. Wendlandt, J.E. Marsden, Mechanical integrators derived from a discrete variational principle, Physica D 106 (1997) 223–246.
[52] C. Kane, J.E. Marsden, M. Ortiz, Symplectic-energy-momentum preserving variational integrator, J. Math. Phys. 40.
[53] C. Kane, J.E. Marsden, M. Ortiz, M. West, Variational integrators and the newmark algorithm for conservative and dissipative mechanical system, Int. J. Numer. Math. Engrg. 49 (1999) 1295–1325.
[54] J.E. Marsden, M. West, Discrete mechanics and variational integrators, Acta Numerica (2001) 357–514.
[55] J.E. Marsden, S. Shkoller, The anisotropic Lagrangian averaged Euler and Navier–Stokes equations, Arch. Rat. Mech. Anal. 166 (2003) 27–46.
[56] A. Lew, J.E. Marsden, M. Ortiz, M. West, Asynchronous variational integrators, Arch. Rat. Mech. Anal. 167 (2003) 85–146.
[57] P. Tremblay, Y. Bourgault, S. Tavoularis, Control of discretization error for time-continuous space–time fem through mesh movement, Not known.
[58] B. Jawerth, W. Sweldens, An overview of wavelet based multiresolution analyses, SIAM Rev. 36 (1994) 377–412.
[59] W. Sweldens, The lifting scheme: a new philosophy in biorthogonal wavelet constructions, in: A.F. Laine, M. Unser (Eds.), Wavelet Applications in Signal and Image Processing III, Proc. SPIE 2569, 1995, pp. 68–79.
[60] A. Cohen, Numerical Analysis of Wavelet Method, Elsevier, Amsterdam, 2003.
[61] D.L. Donoho, Interplating wavelet transforms, Technical Report 408, Department of Statistics, Stanford University, 1992.
[62] T. Humphries, D. Higham, Phase space error control for dynamical systems, SIAM J. Sci. Comput. 21 (2000) 2275–2294.
[63] T. Colonius, Modeling artifical boundary conditions for compressible flow, Annu. Rev. Fluid Mech. 36 (2004) 315–345.
[64] F. John, Partial Differential Equations, Springer, Berlin, 1981.
[65] J.W. Thomas, Numerical Partial Differential Equations: Finite Difference Methods, Springer, Berlin, 1995.
[66] W.I. Briggs, V.E. Henson, S.F. McCormick, A Multigrid Tutorial, second ed., SIAM, Philadelphia, 1999.
[67] A. Brandt, Multi-level adaptive techniques, Technical Report RC6026, IBM Research Report, 1976.
[68] A. Brandt, Multi-level adaptive techniques (MLAT) for partial differential equations: ideas and software, Mathematical Software III (1977) 273–314 (ICASE report 77-20).
[69] P. Wesseling, An Introduction to Multigrid Methods, John Wiley, Chichester, 1992.
[70] W. Hackbusch, Multigrid Methods and Applications, Springer, New York, 1985.
[71] D. Sidilkover, U.M. Ascher, A multigrid solver for the steady-state Navier–Stokes equations using the pressure-Poisson formulation, Comput. Appl. Math. 14 (1995) 21–35.
[72] J.C. Bowman, A. Zeiler, D. Biskamp, A multigrid algorithm for nonlocal collisional electrostatic drift-wave turbulence, J. Comput. Phys. 158 (2) (2000) 239–261.
[73] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C: The Art of Scientific Computing, Cambridge University Press, Cambridge, 1997.
[74] J.M. Burgers, A mathematical model illustrating the theory of turbulence, Adv. Appl. Mech. 1 (1948) 171–1948.
[75] C. Basdevant, M. Deville, P. Haldenwang, J.M. Lacroix, J. Ouazzani, R. Peyret, P. Orlandi, A.T. Patera, Spectral and finite difference solutions of the Burgers equation, Computers & Fluids 14 (1986) 23–41.
[76] J.M. Stockie, J.A. Mackenzie, R.D. Russell, A moving mesh method for one-dimensional hyperbolic conservation laws, SIAM J. Sci. Comput. 22 (5) (2001) 1791–1813.
[77] G.-H. Cottet, P.D. Koumoutsakos, Vortex Methods: Theory and Practice, Cambridge University press, Cambridge, 2000.
[78] N.K.-R. Kevlahan, Stochastic differential equation models of vortex merging and reconnection, Phys. Fluids 17 (2005) 064107.
[79] L. Greengard, V. Rokhlin, A fast algorithm for particle simulations, J. Comput. Phys. 135 (1997) 280–292.
[80] H. Cheng, L. Greengard, V. Rokhlin, A fast adaptive multipole algorithm in three dimensions, J. Comput. Phys. 155 (1999) 468–498.
[81] W.S. Edwards, L.S. Tuckerman, R.A. Friesner, D.C. Sorensen, Krylov methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 110 (1994) 82–102.
[82] J.S. Walker, A Primer on Wavelets and their Scientific Applications, Chapman and Hall/CRC, 1999.
[83] M.J. Alam, A fully Lagrangian advection scheme for electro-osmotic flow, Master's Thesis, University of Alberta, Canada, 2000.
[84] M.J. Alam, J.C. Bowman, Energy conserving simulation of incompressible electro-osmotic and pressure driven flow, Theoret. Comput. Fluid Dyn. 16 (2002) 133–150.
[85] J.C. Tannehill, D.A. Anderson, R.H. Pletcher, Comput. Fluid Mech. Heat Transfer, Taylor and Francis, 1997.