Computers and Mathematics with Applications 68 (2014) 568–577

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

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

A fast adaptive diffusion wavelet method for Burger's equation

Kavita Goyal, Mani Mehra *

Indian Institute of Technology Delhi, India

ARTICLE INFO

Article history:
Received 3 October 2013
Received in revised form 28 May 2014
Accepted 1 June 2014
Available online 3 July 2014

Keywords:
Multiresolution analysis
Burger's equation
Adaptive grid

ABSTRACT

A fast adaptive diffusion wavelet method is developed for solving the Burger's equation. The diffusion wavelet is developed in 2006 (Coifman and Maggioni, 2006) and its most important feature is that it can be constructed on any kind of manifold. Classes of operators which can be used for construction of the diffusion wavelet include second order finite difference differentiation matrices. The efficiency of the method is that the same operator is used for the construction of the diffusion wavelet as well as for the discretization of the differential operator involved in the Burger's equation. The diffusion wavelet is used for the construction of an adaptive grid as well as for the fast computation of the dyadic powers of the finite difference matrices involved in the numerical solution of Burger's equation. In this paper, we have considered one dimensional and two dimensional Burger's equation with Dirichlet and periodic boundary conditions. For each test problem the CPU time taken by fast adaptive diffusion wavelet method is compared with the CPU time taken by finite difference method and observed that the proposed method takes lesser CPU time. We have also verified the convergence of the given method.

1. Introduction

Burger's equation models the situations where both typical non-linearity and diffusion occur [1–4]. The equation is intensively used to test the numerical schemes. Various numerical methods have been developed for solving Burger's equation, for example finite element schemes in [5], finite difference schemes in [6], spectral methods in [7,8] and distributed functional approaches in [9,10].

While solving a partial differential equation (PDE) numerically, using an adaptive grid [11–13] has obvious advantages over using a static grid. Wavelets are widely used for numerical solutions of PDEs (and in particular the Burger's equation) on adaptive grids. In [14] one dimensional Burger's equation with periodic boundary conditions is solved on a static grid using Daubechies wavelet and in [15] it is solved using quasi wavelets [16]. An adaptive grid is generated using spline wavelets to solve one-dimensional Burger's equation with periodic boundary conditions in [17]. In [18] second generation wavelet is used to solve one dimensional Burger's equation on an adaptive grid. But the wavelet theory for numerical solution of PDEs on general manifold is a relatively new field, although there are many non wavelet numerical techniques available [19–22]. One of the work done in this direction is a dynamic adaptive numerical method for solving PDEs on the sphere using second

* Corresponding author. Tel.: +91 11 26591487.
E-mail addresses: kavita@maths.iitd.ac.in (K. Goyal), mmehra@maths.iitd.ac.in (M. Mehra).

http://dx.doi.org/10.1016/j.camwa.2014.06.007
0898-1221/

generation spherical wavelet [23]. Fast adaptive diffusion wavelet method (FADWM) developed in this paper can be seen as a first step in this direction.

The diffusion wavelet is introduced by Coifman and his collaborators in 2006 [24]. The most important feature of the diffusion wavelet is that it can be constructed on general manifolds. This wavelet has not been used for numerical solutions of PDEs and to best of our knowledge ours is the first attempt. In FADWM, diffusion wavelet is used for making an adaptive grid and for the fast computation of the dyadic powers of the finite difference matrices involved in the numerical solution of the Burger's equation.

The paper is organized as follows: Section 2 gives a brief description of the diffusion wavelet. FADWM is developed in Section 3. Section 4 contains the numerical results. Section 5 concludes the paper and gives a brief idea of the future work.

2. A brief description of diffusion wavelet

Diffusion wavelet [24] is constructed on any general manifold X. Multiresolution analysis (MRA) is built using a diffusion operator T on $\mathcal{L}_2(X)$ which is local, self adjoint and whose high powers have low numerical rank. For example $I-T$ with I as an identity operator on $\mathcal{L}_2(X)$ could be the Laplace–Beltrami operator. For any $f \in \mathcal{L}_2(X)$ we have $P_{\mathcal{V}^j} f(x) = \sum_{k\in X^j} c_k^j \phi_k^j(x)$. Computing the scaling function coefficients $\{c_k^j\}_{k\in X^j}$ using the values of $\{f(x_k)\}_{x_k\in X^j}$ is called diffusion scaling function transform (DST). Computing the function f from the coefficients $\{c_k^j\}_{k\in X^j}$ is called the inverse diffusion scaling function transform (IDST).

For any function $f \in \mathcal{L}_2(X)$, $P_{\mathcal{V}^j} f = P_{\mathcal{V}^{j-1}} f + P_{\mathcal{W}^{j-1}} f$. So we can write $(P_{\mathcal{V}^j} f)(x) = \sum_{k\in X^{j-1}} c_k^{j-1}\phi_k^{j-1}(x) + \sum_{k\in Y^{j-1}} d_k^{j-1}\psi_k^{j-1}(x)$, where Y^{j-1} is the index set. Given the set $\mathbf{c}^j$, computing the sets $\mathbf{c}^{j-1} = \{c_k^{j-1}\}_{k\in X^{j-1}}$ and $\mathbf{d}^{j-1} = \{d_k^{j-1}\}_{k\in Y^{j-1}}$ is termed as partial diffusion wavelet transform (PDWT). Now for the coarsest level J_0 and the finest level J, we can decompose the space $\mathcal{V}^J$ as $\mathcal{V}^J = \mathcal{V}^{J_0} \bigoplus_{j=J_0}^{J-1} \mathcal{W}^j$. Therefore

$$(P_{\mathcal{V}^J} f)(x) = \sum_{k\in X^{J_0}} c_k^{J_0}\phi_k^{J_0}(x) + \sum_{j=J_0}^{J-1}\sum_{k\in Y^j} d_k^j \psi_k^j(x). \tag{1}$$

PDWT can be applied on $\mathbf{c}^j$ for $j = J, J-1, \ldots, J_0+1$ to obtain the full diffusion wavelet transform (FDWT) which will give all the coefficients in (1). Constructing the set $\mathbf{c}^j$ from the sets $\mathbf{c}^{j-1}$ and $\mathbf{d}^{j-1}$ is called inverse partial diffusion wavelet transform (IPDWT). Inverse full diffusion wavelet transform (IFDWT) is obtained by applying IPDWT recursively. For details one can see [24].

3. Fast adaptive diffusion wavelet method (FADWM)

3.1. Efficient computation of $\{T^{2^m}, m > 0\}$

Suppose that we are given a function $f \in \mathcal{L}_2(X)$ and we want to compute $T^{2^m} f \approx [T^{2^m}]_{\Phi^J}^{\Phi^J}\mathbf{f}$. Using $[T^{2^m}]_{\Phi^J}^{\Phi^{J-m}} = [T^{2^{m-1}}]_{\Phi^{J-(m-1)}}^{\Phi^{J-m}}[T^{2^{m-2}}]_{\Phi^{J-(m-2)}}^{\Phi^{J-(m-1)}}\cdots[T]_{\Phi^J}^{\Phi^{J-1}}[T]_{\Phi^J}^{\Phi^J}$, we can compute $[T^{2^m}]_{\Phi^J}^{\Phi^{J-m}}\mathbf{f}$ which is the vector of coordinates of $[T^{2^m}]_{\Phi^J}^{\Phi^J}\mathbf{f}$ in the basis Φ^{J-m} of $\mathcal{V}^{J-m}$, i.e., $\mathbf{c}^{J-m}$. From $\mathbf{c}^{J-m}$ we can compute $\mathbf{c}^J$ using IDST. $\mathbf{c}^J$ is nothing but the vector of coefficients of $[T^{2^m}]_{\Phi^J}^{\Phi^J}\mathbf{f}$ in the basis Φ^J which is $[T^{2^m}]_{\Phi^J}^{\Phi^J}\mathbf{f}$ itself. Algorithm to compute $[T^{2^m}]_{\Phi^J}^{\Phi^J}\mathbf{f}$ is:

$T^{2^m} f \approx \mathbf{c}^J =$ **ALGORITHM**(T, f, m)

1) $g = [T]_{\Phi^J}^{\Phi^J}\mathbf{f}$.

For $k = 0, 1, \ldots m-1$

2) $g = [T^{2^k}]_{\Phi^{J-k}}^{\Phi^{J-(k+1)}} g$ } This gives us $\mathbf{c}^{J-m}$.

end

3) $\mathbf{c}^{J-m} \xrightarrow{\text{IDST}} \mathbf{c}^J$.

4) $\mathbf{c}^J = [T^{2^m}]_{\Phi^J}^{\Phi^J}\mathbf{f}$.

We took $f(x) = x$ and computed $[T^{2^{12}}]_{\Phi^8}^{\Phi^8}\mathbf{f}$ (T is the diffusion operator constructed using the construction of [25]) both analytically (the matrix $[T]_{\Phi^8}^{\Phi^8}$ is multiplied 2^{12} times and finally with $\mathbf{f}$) and using the above algorithm. The errors in l_2 and l_∞ norms are 2.393×10^{-9} and 1.563×10^{-10} respectively. The CPU time taken for computing $[T^{2^{12}}]_{\Phi^8}^{\Phi^8}\mathbf{f}$ using the above algorithm is 2% of the CPU time taken for its analytic computation.