View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by KU ScholarWorks

SIAM J. SCI. COMPUT.
Vol. 19, No. 4, pp. 1261–1289, July 1998

© 1998 Society for Industrial and Applied Mathematics

012

PSEUDOSPECTRAL SOLUTION OF NEAR-SINGULAR PROBLEMS USING NUMERICAL COORDINATE TRANSFORMATIONS BASED ON ADAPTIVITY*

L. S. MULHOLLAND†, W.-Z. HUANG‡, AND D. M. SLOAN†

Abstract. The work presented here describes a method of coordinate transformation that enables spectral methods to be applied efficiently to differential problems with steep solutions. The approach makes use of the adaptive finite difference method presented by Huang and Sloan [*SIAM J. Sci. Comput.*, 15 (1994), pp. 776–797]. This method is applied on a coarse grid to obtain a rough approximation of the solution and a suitable adapted mesh. The adaptive finite difference solution permits the construction of a smooth coordinate transformation that relates the computational space to the physical space. The map between the spaces is based on Chebyshev polynomial interpolation. Finally, the standard pseudospectral (PS) method is applied to the transformed differential problem to obtain highly accurate, nonoscillatory numerical solutions. Numerical results are presented for steady problems in one and two space dimensions.

Key words. adaptivity, equidistribution, pseudospectral, differential equations

AMS subject classifications. 65N35, 65N50, 76D30

PII. S1064827595291984

1. Introduction.

PS methods provide an attractive alternative to finite difference and finite element methods for numerical solution of differential equations. The PS method involves approximation by global basis functions—trigonometric or algebraic polynomials, for example—whereas finite difference and finite element methods involve local approximations. For problems with smooth solutions the convergence rate of PS methods is faster than algebraic as the number of grid points increases, and the significance of this so-called spectral convergence is that a specified accuracy can usually be achieved using fewer grid points than would be required by the algebraically convergent finite difference or finite element approaches. However, if a solution has a steep region such as a boundary layer or an interior layer the PS method will achieve high accuracy only if the number of grid points is sufficiently high to permit resolution of the localized phenomena. The required number can be prohibitive in practice, and if insufficient grid points are used an inaccurate numerical solution contaminated by global oscillations will result. In situations of this type a PS method will not be competitive with local approximation methods.

Various schemes have been proposed for improving spectral or PS approximations of steep or discontinuous solutions. The schemes might well be classified into two groups that treat dependent and independent variables, respectively.

Remedial action on dependent variables usually involves the addition of some form of artificial viscosity or the application of postprocessing such as filtering. Examples of filtering applied to spectral approximation of discontinuous solutions may be found in the papers by Majda, McDonough, and Osher [21]; Gottlieb, Lustman, and Orszag

*Received by the editors August 23, 1995; accepted for publication (in revised form) September 30, 1996; published electronically April 16, 1998.
http://www.siam.org/journals/sisc/19-4/29198.html

†Department of Mathematics, University of Strathclyde, Glasgow G1 1XH, Scotland (lmulholland@strath.ac.uk, d.sloan@strath.ac.uk). The first author was supported by the Engineering and Physical Sciences Research Council (EPSRC).

‡Department of Mathematics, University of Kansas, Lawrence, KS 66045 (whuang@math.ukans.edu).

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

[12]; and Cai, Gottlieb, and Shu [7], [8]. The vanishing viscosity method proposed by Tadmor for shock capturing is described in [23, 24, 25]. All of these methods produce spectral accuracy in smooth regions between discontinuities, with each discontinuity generally confined to one grid interval. Recently, Huang and Sloan [17] proposed an upwind PS method for singular perturbation problems without turning points. This upwind method is shown to be free of oscillations and spectrally accurate as the number of grid points tends to infinity, with the perturbation parameter held fixed.

Methods that deal with treatment of the independent variable are applicable to problems with steep, but smooth, solutions. Here, the usual approach is to apply a coordinate transformation that is designed to smooth out regions of high gradient. In the transformed coordinate a PS method can yield spectral accuracy using a reasonably small number of grid points. A transformation that adapts to the features of a solution—usually constructed as the solution is being generated numerically—is the basis of adaptive grid generators. This has been used to great effect with finite difference and finite element methods (for example, see [6], [10], [15], [18], [26], [27], [28]). A common theme in adaptive finite difference and finite element methods is the concept of equidistribution, which seeks to distribute some function evenly over the domain of the problem. The function is usually some measure of computational error or solution variation. The paper by Huang and Sloan [18] gives an interpretation of equidistribution in the context of adaptive grid generation for multidimensional problems. An equidistribution principle is developed in [18] and it is used to formulate a finite difference grid generation algorithm in two space dimensions.

Adaptive schemes have so far not been extended in a widely applicable formulation to spectral or PS approximation methods. In this paper we show how adaptivity based on equidistribution may be incorporated into PS discretization of problems in one space dimension. We also show how to improve the effectiveness of PS methods for steep solutions in one and two dimensions by adapting a coordinate transformation to a quickly computed finite difference solution. A highly accurate PS solution is then obtained in the transformed coordinate. Adaptively generated coordinate transformations have been used previously in PS computations. In [1] this approach is used to solve problems with steep, but continuous, solutions and a coordinate transformation is coupled with artificial viscosity to solve shock wave problems. An analogous technique is adopted in [2], [3], [4], [5], and [14] to solve the Burgers equation, reaction diffusion equations, and flame propagation problems. In each of these papers the coordinate transformation is represented as a parameterized function whose structure is known a priori. The parameter values are obtained by minimizing a functional that measures some error in the computed solution: for example, the functional might be related to the interpolation error. For a specified number of grid points the accuracy obtained in the transformed coordinate is much higher than that which would result from direct PS solution in the physical coordinate. Note that, in general, polynomial approximation in the transformed coordinate corresponds to approximation in the physical coordinate by means of basis functions that are not polynomials. It is the use of basis functions resulting from the mapping that gives rise to the high accuracy of the computed solution.

A parameterized transformation containing a small number of parameters—as used in [2], [3], [4], [5], and [14]—is only effective if prior knowledge of the solution is available. For example, the prior knowledge might be information on the locations of fronts. Here we remove the need for prior knowledge by using parameterized functions containing many parameters, with information on the features of the solution gener-

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

ated by means of a low-cost adaptive finite difference method. Since the knowledge of the solution is generated numerically the features of the solution are not needed a priori, and the method is thus widely applicable.

The objectives of this paper are to show that adaptivity may be coupled with PS discretization to produce highly accurate solutions to problems that have steep, smooth solutions. We show how a postprocessing PS discretization may be applied to adaptive finite difference solutions, computed quickly on coarse grids, to produce highly accurate results at little extra cost. The PS discretization is applied to a system that has the same degree of nonlinearity as the physical system. For example, if the physical PDE is linear, then the PS discrete equations are linear. Section 2 extends the finite difference adaptive method of Huang and Sloan [18] to deal with PS discretization in one space dimension. In section 3 we use the adaptive finite difference scheme in [18] to produce an approximation of the solution and an adapted mesh in the physical coordinate: here a coarse mesh suffices to give a solution that need not be highly accurate. Polynomial interpolation is then used to obtain approximations to the physical mesh at Chebyshev nodes, and a coordinate transformation is obtained by fitting a Chebyshev polynomial expansion to these nodal values. Finally, a standard PS method is employed to solve the transformed differential equation. The process is presented for problems in one space dimension in section 3 and for two space dimensions in section 4, and in each of these sections numerical results are described for steady problems with near-singular solutions. Section 5 contains conclusions and comments on our PS adaptive algorithms.

2. PS adaption based on equidistribution.

2.1. Formulation of algorithm. In [18] the concept of equidistribution is used to generate a one-to-one coordinate transformation from the computational domain D_c to the physical domain D_p in the form

$$\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{\xi}), \quad \boldsymbol{\xi} \in D_c \subset \boldsymbol{R}^m. \tag{2.1}$$

The determination of a mesh, or distribution of nodes, on D_p is equivalent to the construction of a (discrete) transformation (2.1). The presentation by Huang and Sloan shows how an equidistribution principle may be used to determine a distribution of nodes on D_p that corresponds—under the transformation (2.1)—to a given uniform mesh on D_c.

In this section we give the obvious PS extension of the algorithm in [18]: a (discrete) transformation (2.1) is constructed that determines a distribution of nodes on D_p corresponding to a set of Chebyshev–Gauss–Lobatto nodes on D_c. For simplicity we present the PS adaptive method for steady differential problems in one space dimension, in which case the map (2.1) takes the form

$$x = x(\xi), \quad \xi \in [-1, 1]. \tag{2.2}$$

Without loss of generality we assume that $x \in [-1, 1] \subset \boldsymbol{R}$ under (2.2), with

$$x(-1) = -1, \quad x(+1) = +1. \tag{2.3}$$

Consider the linear boundary value problem

$$\begin{cases} \epsilon \dfrac{d^2u}{dx^2} - p(x) \dfrac{du}{dx} - q(x)u = f(x), \quad x \in (-1, 1), \\ u(-1) = a, \quad u(+1) = b, \end{cases} \tag{2.4}$$

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

where $\epsilon > 0$ is a small constant ($\epsilon \ll 1$) and p, q, and f are smooth functions. (It will be obvious in the sequel that more general linear and nonlinear boundary value problems may be dealt with in an analogous manner.) If x is related to a coordinate $\xi \in D_c = [-1, 1]$ by the transformation (2.2) and (2.3), and if $v(\xi) = u(x(\xi))$, we may write equation (2.4) with independent variable ξ in the form

$$\text{(2.5)} \quad \begin{cases} \dfrac{\epsilon}{(x_\xi)^2}\dfrac{d^2v}{d\xi^2} - \left[\dfrac{p(x(\xi))}{x_\xi} + \dfrac{\epsilon x_{\xi\xi}}{(x_\xi)^3}\right]\dfrac{dv}{d\xi} - q(x(\xi))v = f(x(\xi)), \quad \xi \in (-1,1), \\ v(-1) = a, \quad v(+1) = b, \end{cases}$$

where $x_\xi \equiv \frac{dx}{d\xi}$. If $x = x(\xi)$ is chosen such that $v(\xi)$ is slowly varying, the PS approximation of (2.5) will exhibit spectral accuracy. To obtain an effective transformation (2.2) we need to make use of information concerning the features of the solution of (2.4). Here we input the essential information by ensuring that a monitor function based on scaled arc length be equally distributed throughout the physical domain D_p [18].

Following [18] we use the monitor function

$$\text{(2.6)} \qquad M(x,u) = 1 + \alpha^2\left(\frac{du}{dx}\right)^2,$$

where α is a real parameter. The condition that M be equally distributed between nodes in D_p that correspond one-to-one with Chebyshev–Gauss–Lobatto nodes in D_c is readily shown to be

$$\text{(2.7)} \qquad \sqrt{1-\xi^2}\, M^{\frac{1}{2}} x_\xi = \text{ constant}.$$

(2.7) is a reformulation of equation (16) in [18], with the nodes in the ξ coordinate now distributed at Chebyshev locations. The aim now is to obtain a PS solution of the coupled equations (2.5) and (2.7), making use of the boundary conditions (2.3) and the monitor function representation (2.6).

There are several possible discretizations of the mesh equation (2.7). The one that we eventually adopted is obtained by squaring (2.7) then differentiating with respect to ξ. This procedure gives

$$\text{(2.8)} \qquad x_\xi[(1-\xi^2)\, x_{\xi\xi} - \xi x_\xi] + \alpha^2 v_\xi[(1-\xi^2)\, v_{\xi\xi} - \xi v_\xi] = 0,$$

where we have used the transformed monitor function

$$\text{(2.9)} \qquad M = 1 + \alpha^2\Big(\frac{v_\xi}{x_\xi}\Big)^2.$$

Equations (2.3), (2.5), and (2.8) are solved by PS discretization to yield approximations $\{x_i\}_{i=0}^N$ and $\{v_i\}_{i=0}^N$ at the Chebyshev nodes

$$\text{(2.10)} \qquad \xi_i = -\cos\frac{\pi i}{N}, \quad i = 0, 1, \ldots, N,$$

where N is a positive integer and α is set to some preselected value. Approximations to derivatives at node ξ_i are computed as summations of the form

$$\text{(2.11)} \qquad (x_\xi)_i = \sum_{j=0}^{N} D_{ij}^{(1)}\, x_j, \quad (x_{\xi\xi})_i = \sum_{j=0}^{N} D_{ij}^{(2)}\, x_j,$$

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

where $D^{(1)}$ and $D^{(2)}$ are, respectively, the first-order and second-order PS differentiation matrices corresponding to the nodes (2.10). The reader is referred to references [9] and [11] for information on PS solutions.

The nonlinear algebraic system in $2(N-1)$ unknowns arising from the PS discretization of (2.3), (2.5), and (2.8) is solved using a Newton–Raphson iteration with exact representation of the Jacobian, and with continuation in the parameters α and ϵ. The parameter ϵ is set to a value $\bar{\epsilon} \sim 1$ and a family of problems is solved on the continuation path $[0, \bar{\epsilon}] \rightarrow [\alpha, \bar{\epsilon}]$, with the arc-length scaling parameter increasing by small steps from 0 to α. A second family of problems is then solved on the continuation path $[\alpha, \bar{\epsilon}] \rightarrow [\alpha, \epsilon]$, with the diffusion parameter decreasing by small steps from $\bar{\epsilon}$ to ϵ. The discrete system with $\epsilon \ll 1$ may have a number of unstable solutions. By employing continuation in ϵ from the relatively large value—where there is only one (stable) solution for prescribed α—the path of the stable solution can be followed. The implementation of the continuation process involved a step length, initially set to solve in one step, reduced by a factor 4 on failure of the Newton–Raphson method and increased by a factor 1.1 on successful convergence. The Newton–Raphson method was deemed to have failed if the residual increased on successive iterations or when eight iterations were performed without convergence to a required tolerance.

If the equations are solved in the manner described above the mesh quality may be poor. In particular, for this one-dimensional situation, it is possible that node crossing will occur. This corresponds to a situation where the transformation (2.2) is not strictly monotonic increasing in some subregion of $[-1, 1]$. To avoid this type of difficulty it is necessary to incorporate some smoothing into the solution process. This has been discussed, for example, in [10] and [18] and an analysis is given in [16].

The smoothing introduced here is achieved by introducing filtering during the evaluation of PS derivative approximations as in equations (2.11). Derivative filtering is readily achieved by modifying the appropriate PS differentiation matrix, and for this purpose it is convenient to consider a factorized form of the matrix. Suppose a function v is approximated by the Nth degree polynomial expressed as

$$v(\xi) \sim v_N(\xi) = \sum_{n=0}^{N} a_n \, T_n(\xi), \tag{2.12}$$

where T_n is a Chebyshev polynomial of the first kind of degree n. If the coefficients $\{a_n\}_{n=0}^{N}$ are chosen such that $v(\xi_i) = v_N(\xi_i)$ for $i = 0, 1, \ldots, N$, then $\boldsymbol{a} = [a_0, a_1, \ldots, a_N]^T$ is related to $\boldsymbol{v}_N = [v_N(\xi_0), v_N(\xi_1), \ldots, v_N(\xi_N)]^T$ by the linear relation

$$\boldsymbol{v}_N = C^{-1} \, \boldsymbol{a} \quad \text{or} \quad \boldsymbol{a} = C \, \boldsymbol{v}_N, \tag{2.13}$$

where C is the discrete Chebyshev transform matrix. The first derivative of v is approximated by

$$\frac{dv_N}{d\xi} = \sum_{n=0}^{N} b_n^{(1)} \, T_n(\xi), \tag{2.14}$$

where the properties of Chebyshev polynomials enable us to relate $\boldsymbol{b}^{(1)} = [b_0^{(1)}, b_1^{(1)}, \ldots, b_N^{(1)}]^T$ to $\boldsymbol{a}$ as (see [13])

$$\boldsymbol{b}^{(1)} = E^{(1)} \, \boldsymbol{a}. \tag{2.15}$$

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

The vector that approximates the first-order derivative of v at the nodes $\{\xi_i\}_{i=0}^N$ is given by

$$\boldsymbol{v}_N^{(1)} = \Big[\frac{dv_N}{d\xi}(\xi_0),\ \frac{dv_N}{d\xi}(\xi_1), \ldots, \frac{dv_N}{d\xi}(\xi_N)\Big]^T,$$

and it is readily seen from (2.13)–(2.15) that this is

$$\boldsymbol{v}_N^{(1)} = C^{-1}\, E^{(1)}\, C\, \boldsymbol{v}_N. \tag{2.16}$$

The matrix $C^{-1}\, E^{(1)}\, C$ is the first-order PS differentiation matrix $D^{(1)}$ introduced in (2.11). Higher-order differentiation matrices have the structure

$$D^{(r)} = C^{-1}\, E^{(r)}\, C, \quad r = 2, 3, \ldots, \tag{2.17}$$

where $E^{(r)}$ is readily determined [13]. Alternative views of $D^{(r)}$ are described in the review by Fornberg and Sloan [11].

Filtering may be included in the differentiation process by modifying $D^{(r)}$ to

$$\tilde{D}^{(r)} = C^{-1}\, E^{(r)}\, BC, \tag{2.18}$$

where B is a diagonal matrix with elements $B_{ii} = \beta_i$, $i = 0, 1, \ldots, N$. In this section we have adopted an exponential filter of the form

$$\beta_i = \exp[-\delta(i/N)^\gamma] \ \text{ with } \ \delta = 32. \tag{2.19}$$

Note that $\beta_0 = 1,\ 0 < \beta_i < 1$ for $i = 1, 2, \ldots, N$, and $\beta_i \approx 0$ for i close to N. The effect of introducing B in (2.18) is to diminish the magnitudes of the high components in the expansion (2.12) prior to the differentiation process. Filtering may be included in the approximations to $x_\xi, x_{\xi\xi}, v_\xi, v_{\xi\xi}$ in (2.5), (2.8).

2.2. Illustrative examples. We now apply the above algorithm to four simple test problems: three linear and one nonlinear. In each case the same filtering strategy has been employed: the differentiation matrix $\tilde{D}(r)$, given by (2.18), is used for the evaluation of v_ξ and $v_{\xi\xi}$ in (2.8), and for the evaluation of x_ξ and $x_{\xi\xi}$ in the transformed differential equation to be solved.

Example 2.1. Consider the linear boundary value problem

$$\begin{cases} \epsilon \dfrac{d^2u}{dx^2} + 2x \dfrac{du}{dx} = 0, \quad x \in (-1, 1), \\ u(-1) = -1, \quad u(1) = 1. \end{cases} \tag{2.20}$$

Given that the exact solution of (2.20) will be very close to $u(x) = \mathrm{erf}(x/\sqrt{\epsilon})$, we note that for small ϵ this variable coefficient problem has a steep front at $x = 0$. Upon transformation, the term $\epsilon \frac{d^2u}{dx^2}$ becomes $\frac{\epsilon}{x_\xi}(\frac{v_\xi}{x_\xi})_\xi$, with the ith component in the discrete system given by $\frac{\epsilon}{(x_\xi)_i}\sum_{k=0}^N D_{ik}^{(1)}(\frac{v_\xi}{x_\xi})_k$. The same discretization procedure is adopted for the identical second derivative term in Examples 2.2–2.4.

We use problem (2.20) to first illustrate the necessity for some form of grid adaption when applying a PS method to approximate solutions with steep gradients. Figure 1a plots the standard PS method approximation for the solution of (2.20) with $\epsilon = 0.0002$, using $N = 32$. The corresponding adaptive PS approximation, with mesh adaption parameter $\alpha = 0.5$ and filter parameter $\gamma = 10$, is plotted in Figure 1b and

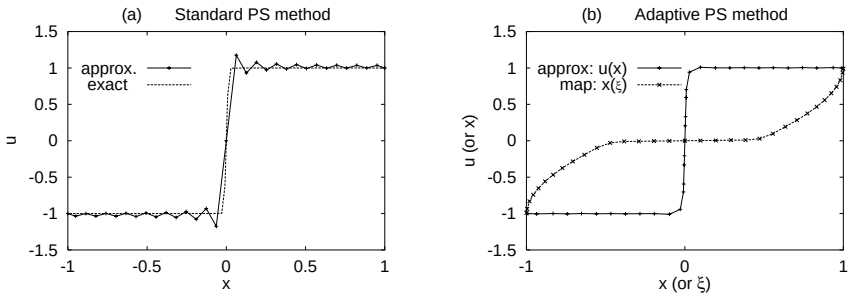

FIG. 1. *Standard and adaptive PS solutions of* (2.20) *with* $\epsilon = 0.0002$ *and* $N = 32$.

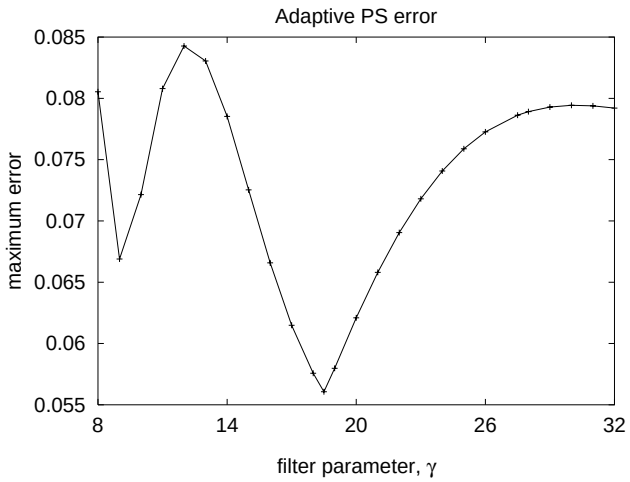

FIG. 2. *Maximum absolute errors versus filter parameter* γ *for adaptive PS solutions to* (2.20) *with* $\epsilon = 0.0002$, $\alpha = 0.5$, *and* $N = 32$.

shows that the grid adaption effectively removes the large oscillations by sufficiently resolving the steep front. Figure 1b also plots the mapping $x(\xi)$ used implicitly in the adaptive PS method; we note that the value of $x_\xi(\xi)$, though positive throughout, is very small in a small neighborhood of $x = 0$.

To show the effect of the filter parameter γ on the error in approximation of the adaptive PS method, Figure 2 plots the maximum absolute error against a range of values of γ. For smaller values of γ the filter becomes too strong for steep fronts to be sufficiently resolved: the continuation method halts before ϵ has been reduced as low as 0.0002. For larger values of γ the filter becomes too weak to dampen out small oscillations in $x(\xi)$ resulting in a loss of monotonicity: grid crossing is observed. The figure shows that within the range of permissible values of γ, the maximum error does not vary substantially.

Although the adaptive PS method appears to improve on the standard PS method for solving (2.20) given moderately small values of ϵ, it fails for values of ϵ much below 0.0002. A finite difference method, on the other hand, can provide (see section 3) results of a similar accuracy for much smaller values of ϵ.

Example 2.2. We now consider a problem obtained by modifying Example 2.1 so

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

that the solution has a nonzero gradient outside the front for small ϵ. This is

$$\begin{cases} \epsilon \dfrac{d^2u}{dx^2} + x \dfrac{du}{dx} = -\epsilon\pi^2 \cos(\pi x) - \pi x \sin(\pi x), & x \in (-1,1), \\ u(-1) = -2, \quad u(1) = 0. \end{cases} \tag{2.21}$$

For this problem, the adaptive PS method performs no better than the standard PS method for values of ϵ down to 0.001 and does not converge at all for smaller values of ϵ.

Example 2.3. The linear boundary value problem

$$\begin{cases} \epsilon \dfrac{d^2u}{dx^2} + 2x \dfrac{du}{dx} = -\dfrac{2}{\sqrt{\pi\epsilon}} \left[e^{-(x+0.5)^2/\epsilon} - e^{-(x-0.5)^2/\epsilon} \right], & x \in (-1,1), \\ u(-1) = -2, \quad u(1) = 2 \end{cases} \tag{2.22}$$

has steep fronts at $x = \pm 0.5$ for small ϵ.

The results for this problem are very similar to those of Example 2.2. The adaptive PS method performs no better than the standard PS method.

Example 2.4. The final illustration in this subsection is the nonlinear boundary value problem

$$\begin{cases} \epsilon \dfrac{d^2u}{dx^2} - \dfrac{d(u^2)}{dx} = 0, & x \in (-1,1), \\ u(-1) = 1, \quad u(1) = -1. \end{cases} \tag{2.23}$$

In this case the steep front, situated at $x = 0$, has width of order ϵ, whereas in the three previous examples the width was of order $\sqrt{\epsilon}$. Figure 3 plots the transformation $x(\xi)$, the monitor function (scaled to have maximum value 1) in computational space, and the computed solution, for $\epsilon = 0.01$, in physical and computational space, using $N = 32$, $\alpha = 1$, and $\gamma = 10$. The maximum absolute error in this case is 4.440×10^{-2} compared with an error of 0.678 for the standard PS method. Note that the effect of filtering u_ξ in the mesh equation is to allow contributions from higher derivatives $u_{\xi\xi}$, $u_{\xi\xi\xi}$, etc. to the computed monitor function [22]. This effect is desirable with respect to $u_{\xi\xi}$ for the better resolution of $x(\xi)$, where x_ξ is varying rapidly, but is less desirable with respect to higher derivatives. Unfortunately, the filtering process used here cannot be that selective.

The numerical experiments described above led us to conclude that simultaneous solution of discretized forms of (2.5) and (2.8) has its drawbacks. For example, in an extremely steep region the map $x = x(\xi)$ may have ripples that destroy monotonicity, and this gives rise to node crossing. To avoid this, extreme care is needed in introducing filtering. Furthermore, the effort expended in finding a very accurate solution of the mesh equation (2.8) reduces the computational efficiency of this approach, particularly if one wished to apply it to multidimensional problems. A high accuracy is not required for the node locations. In the next section we present a method that overcomes some of these problems.

3. Adaptive transformation method in one dimension.

3.1. Adaptive finite difference method. Here we give a brief description of the adaptive finite difference method that is used in the construction of the one-to-one coordinate transformation between the physical and computational domains. The method is a one-dimensional formulation of the equidistribution method described in

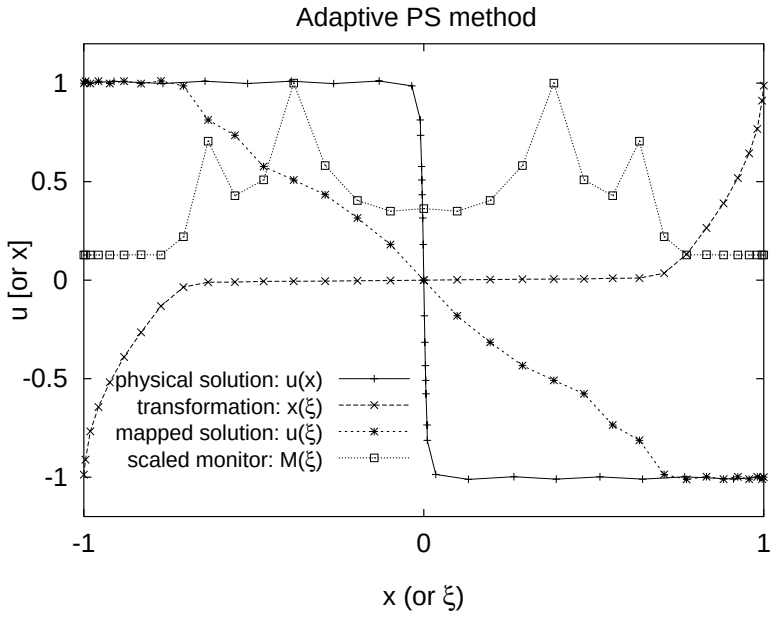

FIG. 3. *Map, monitor function and adaptive PS solution of* (2.23) *with* $\epsilon = 0.01$ *for* $N = 32$, $\alpha = 1$, *and* $\gamma = 10$.

[18]. Suppose the aim is to solve equation (2.4). We assume that a transformation $x = x(\eta)$ is required relating physical nodes $\{x_i\}_{i=0}^{n}$ in $D_p = [-1, 1]$ and evenly spaced nodes

$$\eta_i = -1 + \frac{2i}{n}, \quad i = 0, 1, \ldots, n \tag{3.1}$$

in $D_c = [-1, 1]$. The equidistribution principle is expressed as

$$\left[M\, x_\eta^2\right]^{\frac{1}{2}} = \text{ constant}, \tag{3.2}$$

where M is the scaled arc-length monitor function

$$M = 1 + \alpha^2 \Big(\frac{w_\eta}{x_\eta}\Big)^2 \tag{3.3}$$

introduced in (2.6), and $w(\eta) = u(x(\eta))$. Equation (3.2) is approximated at $\eta_{i+1/2}$ by second-order central finite differences to give

$$\left[M_{i+\frac{1}{2}} \Big(\frac{x_{i+1} - x_i}{\eta_{i+1} - \eta_i}\Big)^2\right]^{\frac{1}{2}} = \text{ constant}, \quad i = 0, 1, \ldots, n-1,$$

where

$$M_{i+\frac{1}{2}} = 1 + \alpha^2 \Big(\frac{w_{i+1} - w_i}{x_{i+1} - x_i}\Big)^2$$

and $x_i = x(\eta_i)$, $w_i = u(x_i)$. As in [18] we eliminate the constant and obtain the system

$$\left[M_{i-\frac{1}{2}} \Big(\frac{x_i - x_{i-1}}{\eta_i - \eta_{i-1}}\Big)^2\right]^{\frac{1}{2}} - \left[M_{i+\frac{1}{2}} \Big(\frac{x_{i+1} - x_i}{\eta_{i+1} - \eta_i}\Big)^2\right]^{\frac{1}{2}} = 0, \quad i = 1, 2, \ldots, n-1. \tag{3.4}$$

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

To obtain a smooth mesh we make use of smoothed matrices

$$\tilde{M}_{i+\frac{1}{2}} = \frac{\sum_{k=i-p}^{i+p} M_{k+\frac{1}{2}} \left(\frac{q}{q+1}\right)^{|k-i|}}{\sum_{k=i-p}^{i+p} \left(\frac{q}{q+1}\right)^{|k-i|}}, \tag{3.5}$$

where the positive real number q is the *smoothing parameter* and the nonnegative integer p is the *smoothing index* [18]. In (3.5) the summations contain only those elements that are well defined ($0 \le k \le n-1$). The discretized grid equation may now be written as

$$\begin{cases} \left[\tilde{M}_{i-\frac{1}{2}} \left(\frac{x_i - x_{i-1}}{\eta_i - \eta_{i-1}}\right)^2\right]^{\frac{1}{2}} - \left[\tilde{M}_{i+\frac{1}{2}} \left(\frac{x_{i+1} - x_i}{\eta_{i+1} - \eta_i}\right)^2\right]^{\frac{1}{2}} = 0, \\ \qquad\qquad \text{for} \quad i = 1, 2, \ldots, n-1, \\ \qquad\qquad \text{with} \quad x_0 = -1 \text{ and } x_n = +1. \end{cases} \tag{3.6}$$

This grid system is augmented by a finite difference approximation of the differential equation (2.4). This is obtained by first transforming to the computational coordinate—to give an equation akin to (2.5), with v and ξ replaced by w and η, respectively—and then discretizing on the uniform mesh by three-point upwind and central finite differences for the first and second derivatives, respectively. It should be noted that central differences may be used to approximate the first-order derivative within the adaptive algorithm. The resulting nonlinear algebraic system in $\{x_i\}_{i=1}^{n-1}$ and $\{u_i\}_{i=1}^{n-1}$ is solved by Newton iteration with exact Jacobian and continuation in α and ϵ as described immediately below equation (2.11).

Note the effect of the smoothing index p on the structure of the Jacobian in the Newton iteration. The linearization of (3.6) gives rise to a linear $(3+2p)$ block diagonal system, where each block is 2×2. As p increases the smoothness of the mesh increases, and mesh points tend to move out from steep regions to the adjacent regions of high curvature. Note, of course, that the matrices to be inverted in the Newton iteration become more dense as p is increased. To obtain a high quality mesh it is essential to find a balance between the tendency of increasing α^2 to pull mesh points into a steep region and the tendency of increasing p to move mesh points out from a steep region.

3.2. Construction of coordinate transformation. The previous section provides a discrete transformation that relates the physical mesh $\{x_i\}_{i=0}^{n}$ to an evenly spaced mesh $\{\eta_i\}_{i=0}^{n}$ on D_c. We now use this information to provide a smooth transformation like (2.2) that will take u in (2.4) to a suitably slowly varying v in (2.5). We seek a transformation $x = x(\xi)$ relating the computational and physical domains that satisfy some essential properties. To ensure that the map is differentiable and readily computable we shall approximate $x = x(\xi)$ by a real polynomial $P_m(\xi)$, where the degree m need not be set equal to n. To prevent node crossing in D_p we could impose the severe monotonicity condition $P'_m(\xi) > 0 \ \forall \xi \in [-1, 1]$. In principle, this can be achieved by a least squares approach, but it gives rise to a very unwieldy system of equations in the parameters that define P_m. The conditions imposed effectively ensure that P'_m has no real zeros in the interval $[-1, 1]$. There are approximation theoretic results on monotone approximation by polynomials that are based on Jackson-type theorems: the reader is referred to the paper by Leviatan [19] and references therein.

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Here we present a simple technique for the construction of a polynomial approximation to the map $x = x(\xi)$. It does not guarantee strict monotonicity for $\xi \in [-1, 1]$, but it gives consistently good results in practice. The transformation will be approximated by a Chebyshev polynomial expansion of degree m, and to evaluate coefficients in this expansion we require values of x at the Chebyshev nodes

$$\bar{\xi}_i = -\cos\frac{\pi i}{m}, \quad i = 0, 1, \ldots, m. \tag{3.7}$$

The values $\{\bar{x}_i\}_{i=0}^{m}$ that approximate x at these Chebyshev nodes are obtained by linear interpolation on the solution $\{x_j\}_{j=0}^{n}$ computed on the evenly spaced grid. The transformation (2.2) is approximated by

$$x(\xi) = \sum_{k=0}^{m} a_k T_k(\xi) \tag{3.8}$$

and the interpolatory conditions $x(\bar{\xi}_i) = \bar{x}_i,\ i = 0, 1, \ldots, m$, yield

$$a_k = \frac{2}{m}\,\frac{1}{\bar{c}_k}\sum_{j=0}^{m}\frac{\bar{x}_j}{\bar{c}_j}\,\cos\Big(\frac{\pi k j}{m}\Big), \tag{3.9}$$

where

$$\bar{c}_k = \begin{cases} 1, & k = 1, 2, \ldots, m-1, \\ 2, & k = 0, m. \end{cases} \tag{3.10}$$

To achieve spectral accuracy when solving the transformed equation (2.5) by means of a PS method it is absolutely essential that the transformation (2.2) (now approximated by (3.8)) be smooth. The polynomial approximation will not generally have positive gradient throughout $[-1, 1]$: there will be small amplitude, high frequency oscillations in regions of D_c that correspond to regions of large gradient in D_p. The effect of these oscillations can be diminished by using an appropriate filter in the expansion (3.8). The smoothed transformation is

$$x = P_m(\xi) = \sum_{k=0}^{m} \sigma_k\, a_k\, T_k(\xi), \tag{3.11}$$

where

$$0 \le \sigma_{k+1} \le \sigma_k \le 1 \quad \text{for} \quad k = 0, 1, \ldots, m-1.$$

Two filters used in the computations to be presented in this section are the raised cosine filter and the 2-parameter exponential filter. These may be written, respectively, as

$$\sigma_k = \frac{1}{2}\Big[1 + \cos\Big(\frac{\pi k}{m}\Big)\Big], \tag{3.12}$$

$$\sigma_k = \exp\big[-\delta(k/m)^{\gamma}\big], \tag{3.13}$$

where δ and γ are positive real numbers.

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Note that the introduction of filtering yields a transformation (3.11) that no longer satisfies the boundary conditions (2.3). If steep regions in the solution u of (2.4) are not close to the boundaries, then no real difficulties ensue, since $u(-1) \sim u(P_m(-1))$ and $u(+1) \sim u(P_m(+1))$. If the problem has a steep boundary layer, however, then the transformation must satisfy (2.3). This can be achieved if we proceed as follows. The procedure refers to real polynomials $\overline{Q}_{m-2}$ and Q_{m-2} of degree $m-2$ and to real polynomials $\overline{P}_m$ and P_m of degree m.

PROCEDURE.

(i) Define $\overline{Q}_{m-2}$ by $\overline{P}_m(\xi) = (1-\xi^2)\,\overline{Q}_{m-2}(\xi) + \xi = \sum_{k=0}^{m} a_k T_k(\xi)$;

(ii) Write $\overline{Q}_{m-2}$ in the form

$$\overline{Q}_{m-2}(\xi) = \sum_{k=0}^{m-2} b_k\, T_k(\xi)$$

and obtain $\{b_k\}_{k=0}^{m-2}$ using the recurrence relation

$$b_j := -4a_{j+2} + 2b_{j+2} - b_{j+4} \qquad \text{for } j = m-2, m-3, \ldots, 0$$

($b_j \equiv 0$ for $j > m-2$) followed by the correction $b_0 := 0.5b_0$;

(iii) Replace $\overline{Q}_{m-2}(\xi)$ by the filtered expansion

$$Q_{m-2}(\xi) = \sum_{k=0}^{m-2} \sigma_k\, b_k\, T_k(\xi);$$

(iv) Define P_m by

$$P_m(\xi) = (1-\xi^2)\, Q_{m-2}(\xi) + \xi. \tag{3.14}$$

Note that (3.14) satisfies the boundary conditions at $\xi = \pm 1$.

This completes the construction of the approximation to the map (2.2) and (2.3). The differential problem (2.4) is transformed to the form (2.5), and this problem is now solved using a standard PS method based on nodes

$$\xi_i = -\cos\frac{\pi i}{N}, \quad i = 0, 1, \ldots, N. \tag{3.15}$$

The complete computation uses integers n, m, and N that are completely independent of each other. Typical values in the computations described below are $n = 40$, $m = 64$, and $N = 128$.

3.3. Illustrative examples. Here we repeat, for the adaptive transformation method outlined above, the examples given in section 2.2, that is, the interior layer problems (2.20)–(2.23).

Example 3.1. To illustrate how well the adaptive transformation method can cope with very steep fronts, we apply the method to problem (2.20) with $\epsilon = 10^{-10}$. In all calculations involving the smoothing process (3.5) we set the parameter q to the value 2. Filtering of the Chebyshev expansion (3.11) was effected using the exponential filter (3.13) with δ set to the value 32. Figure 4a plots the approximate solution on physical and computational meshes and the smoothed and unsmoothed maps for the parameter set $\{n = 64,\ \alpha = 8,\ p = 18,\ m = 164,\ \gamma = 7,\ N = 128\}$. Figure 4b plots the corresponding monitor function, in physical and computational space, for

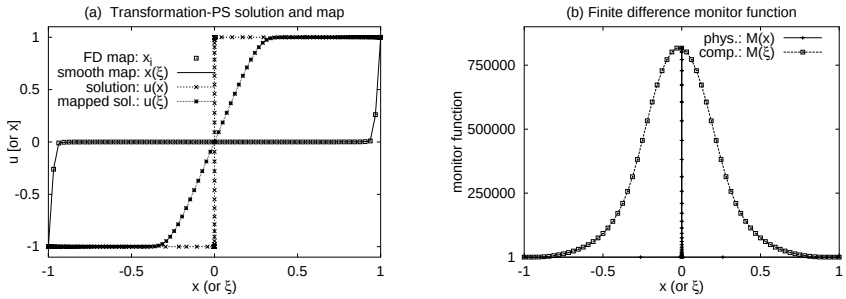

FIG. 4. *Adaptive transformation solution, maps, and monitor function for* (2.20) *with* $\epsilon = 10^{-10}$, $n = 64$, $m = 164$, $N = 128$.

TABLE 1

Maximal errors of adaptive transformation method, using $n = 32$, $\alpha = 4$, $p = 8$, $m = 64$, *and* $\gamma = 4$ *for* (2.20) *with* $\epsilon = 10^{-6}$.

N	max. error	N	max. error
16	8.913×10^{-3}	20	4.472×10^{-3}
32	1.122×10^{-4}	40	9.967×10^{-6}
64	4.309×10^{-8}	80	7.920×10^{-9}
128	1.830×10^{-12}	160	1.781×10^{-13}

the adaptive finite difference method. For this steep problem (front–width$\approx 10^{-5}$) the finite difference error is 3.304×10^{-3} and the transformation-PS error is 2.317×10^{-7}.

In Table 1 we summarize the maximal errors of computed solutions to (2.20), for different values of N, of the adaptive transformation method for the case $\epsilon = 10^{-6}$ using $m = 64$ and filter (3.13) with $\gamma = 4$; these are based on an adaptive finite difference computed solution, using $n = 32$, $\alpha = 4$, and $p = 8$, whose maximum absolute error was 9.478×10^{-3}. The increase in accuracy as N increases clearly exhibits spectral convergence. For the finite difference method alone to achieve an accuracy of 1.830×10^{-12} it would have to use in the order of 2.3 million mesh points.

Thus we have demonstrated that PS postprocessing can produce very accurate solutions, but we have yet to demonstrate its computational efficiency. An interesting point here is, what do we compare the cost of the PS method against? We have two clear choices:

(i) a purely adaptive finite difference strategy with increasing number of grid points,

(ii) low-order finite difference postprocessing replacing PS postprocessing.

The first case, where we simply apply the first stage finite difference adaptive grid method with more grid points to achieve greater accuracy, will not be competitive as our adaptive algorithm stands. This is because the first stage algorithm has been designed for robustness (able to solve very stiff problems) rather than computational efficiency. More specifically, for the same number of grid points n the second-order adaptive grid method repeatedly solves a nonlinear system of $2(n-1)$ equations using a nearly full Jacobian, while the PS method solves a full system of $(n-1)$ equations which will be linear if the underlying PDE is linear. The second method provides for a much more realistic (and much studied) comparison, that is, between PS and finite difference methods for solving problems with smooth solutions. As an example, we apply centered second-order finite difference postprocessing to the above

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

problem. Figure 5 graphs the computational cost of achieving a range of accuracies for both PS and second-order finite difference postprocessing normalized to the cost for the adaptive grid method to obtain an adapted grid; all parameter values are those listed in the caption for Table 1. Figure 5 clearly shows that the PS method is more efficient than the finite difference method when very high accuracy of solution is sought. Projected finite difference cost is an estimate for higher accuracies based on an $O(n)$ operation count and no round-off error. Further work is required to give a detailed comparison of postprocessing strategies over a number of test problems in one and two dimensions.

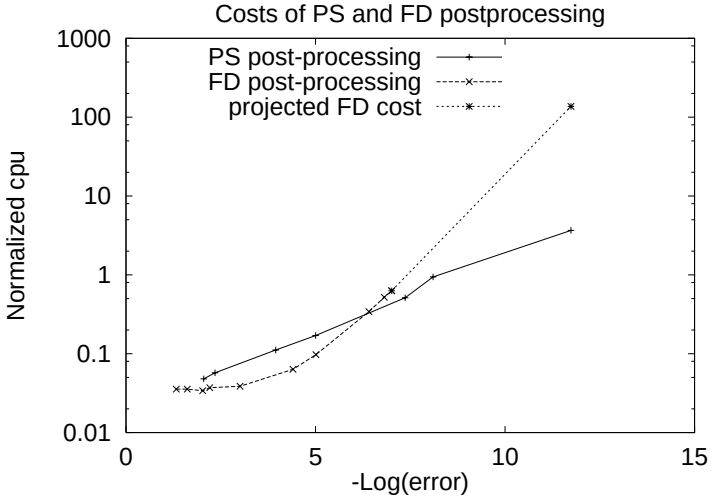

FIG. 5. *Comparison of computational costs for PS and second-order finite difference postprocessing to achieve a range of accuracies to the solution of* (2.20).

For the remaining examples in this section we simply provide figures showing the computed solutions and their associated transformations for very small values of ϵ. The general trends with respect to the various parameters of the overall method are similar in all cases.

Example 3.2. Figure 6 shows the adaptive transformation solution and associated transformation for problem (2.21) with $\epsilon = 10^{-10}$ and using the parameter values listed in the caption. The maximum pointwise error of the method in this case is 7.549×10^{-6} compared with 8.575×10^{-2} for the adaptive finite difference method.

Example 3.3. Figure 7 shows the adaptive transformation solution and associated transformation for problem (2.22) with $\epsilon = 10^{-6}$ and using the parameter values listed in the caption. The maximum pointwise error of the method in this case is 3.134×10^{-5} compared with 2.372×10^{-2} for the adaptive finite difference method.

Example 3.4. Figure 8 shows the adaptive transformation solution and associated transformation for the nonlinear problem (2.23) with $\epsilon = 10^{-5}$ and using the parameter values listed in the caption. The maximum pointwise error of the method in this case is 6.311×10^{-7} compared with 2.192×10^{-3} for the adaptive finite difference method.

Parameter sensitivity. In the overall method there are many free parameters for which values must be set. For some of these parameters the values can be fixed

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

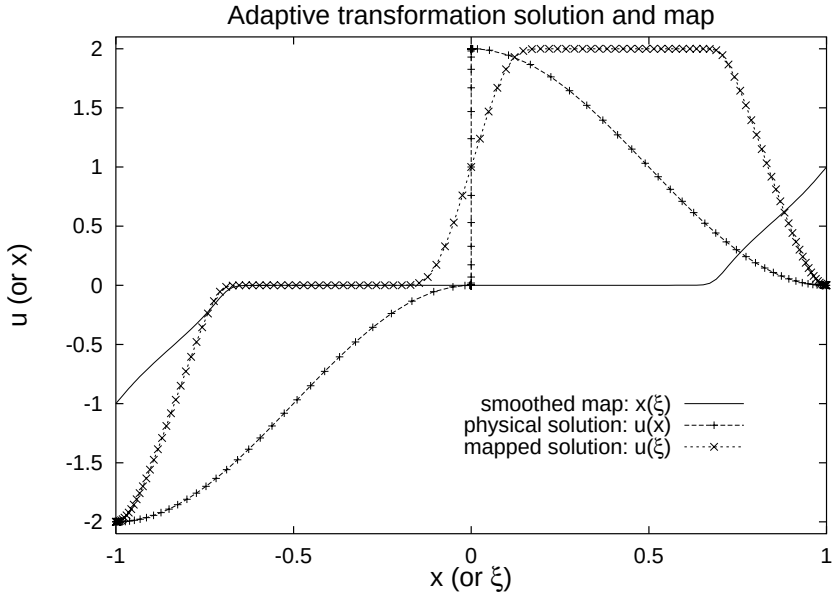

FIG. 6. *Adaptive transformation solution and transformation for* (2.21) *with* $\epsilon = 10^{-10}$ *and using* $n = 64$, $\alpha = 2$, $p = 16$, $m = 128$, $N = 128$, *and* $\gamma = 6$.

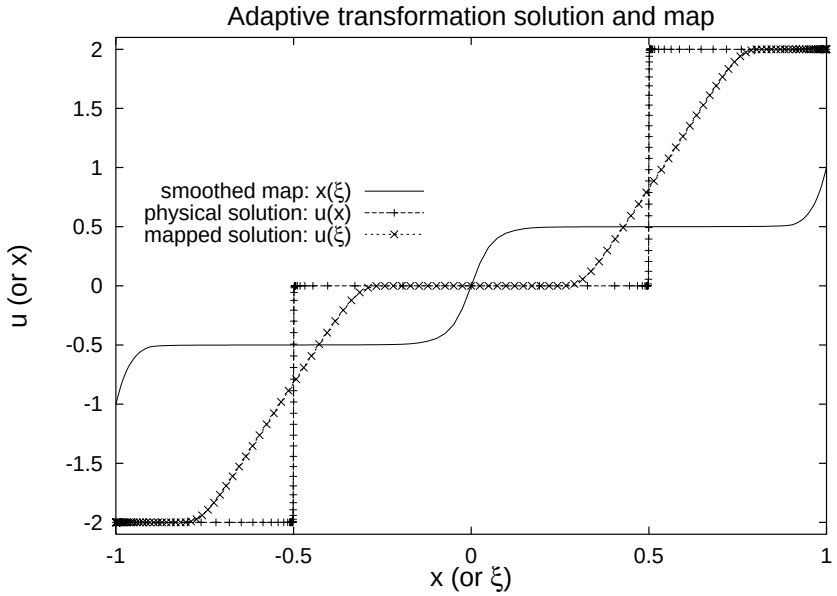

FIG. 7. *Adaptive transformation solution and transformation for* (2.22) *with* $\epsilon = 10^{-6}$ *and using* $n = 128$, $\alpha = 6$, $p = 8$, $m = 164$, $N = 128$, *and* $\gamma = 6$.

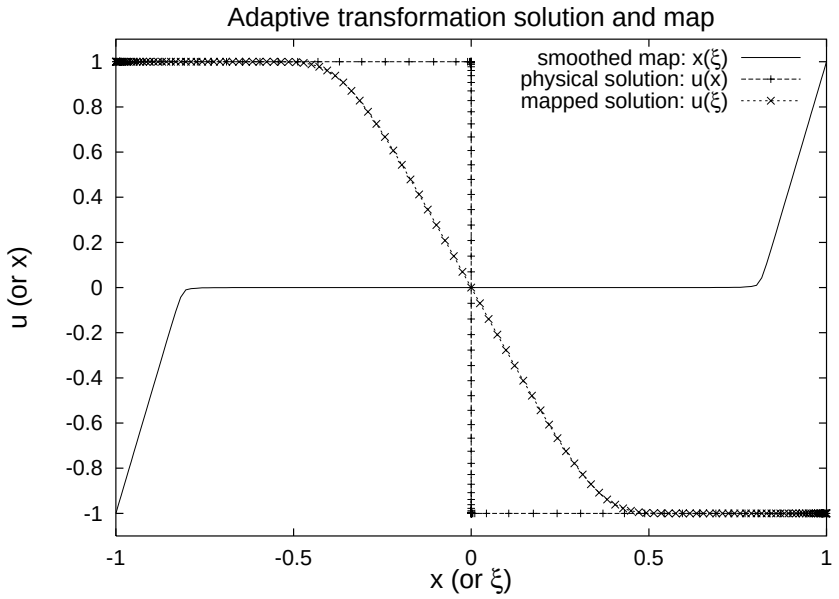

FIG. 8. *Adaptive transformation solution and transformation for* (2.23) *with* $\epsilon = 10^{-5}$ *and using* $n = 64$, $\alpha = 2$, $p = 9$, $m = 164$, $N = 128$, *and* $\gamma = 6$.

and never altered: e.g., $q = 2$ in the smoothing process (3.5) and $\delta = 32$ in the filter (3.13). For the degree of polynomial expansion m in the transformation procedure, a value of 164 was found to be large enough in all test cases tried in this paper. The really critical parameters with respect to PS solution sensitivity are the smoothing index p and the filter parameter γ. we now consider these parameters in more detail.

Solution sensitivity with respect to γ. We first address the question of how sensitive is the adaptive transformation method to the choice of filter parameter γ. As an example, Figure 9 plots the maximum pointwise error of the method applied to (2.20) with $\epsilon = 10^{-6}$ and using $n = 64$, $\alpha = 4$, $p = 8$, $m = 164$, $N = 128$ for a range of values for the filter parameter in (3.13). In this case, the optimal choice of $\gamma = 2.3$ is very sharp; however, any value between 2.1 and 8 would yield a satisfactory error and the rapid convergence achieved by the PS method will reduce the error very quickly with increasing N. For the other test problems the optimal choice for γ is less sharp, but there is a persisting pattern: γ has a minimum value below which the mapping is no longer monotone, γ should be small (i.e., close to but not less than the minimum value), and the best choice for γ is relatively invariant with respect to n and α for a particular test problem.

Solution sensitivity with respect to *p*. With regard to the choice of value for the smoothing index p we note that the best choice with respect to minimizing error in the adaptive finite difference computed solution is not necessarily the ideal choice with respect to the construction of a smooth coordinate transformation. It is our experience that the latter is usually larger than the former to allow for better resolution of $x(\xi)$ wherever $|x_{\xi\xi}|$ is large. Figure 10 indicates, for Example 3.4, how the best choice for p might vary with different values for n and α. We observe that choosing $8 \leq p \leq 10$ is close to optimal in all cases. It is our experience that the best

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

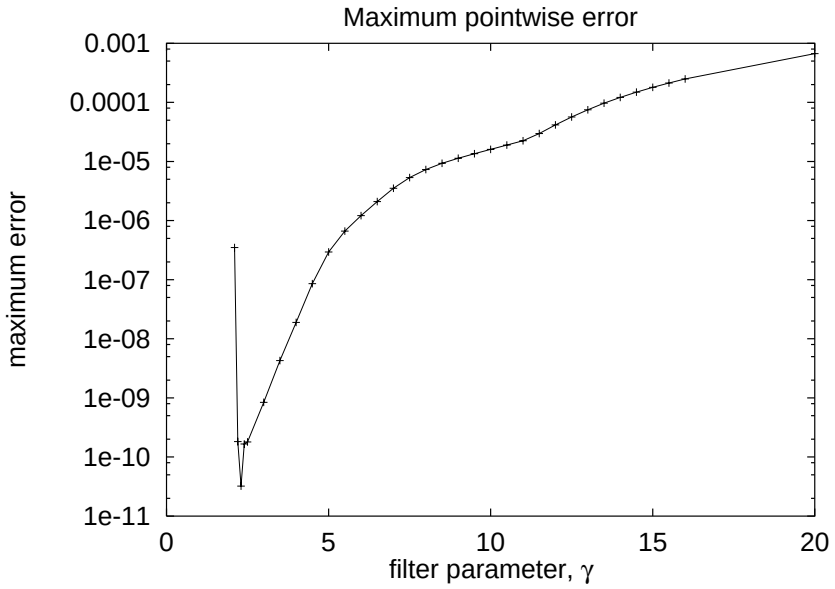

FIG. 9. *Maximum pointwise error versus filter parameter γ for the adaptive transformation method applied to* (2.20)*, with $\epsilon = 10^{-6}$ and using $n = 64$, $\alpha = 4$, $p = 8$, $m = 164$, $N = 128$.*

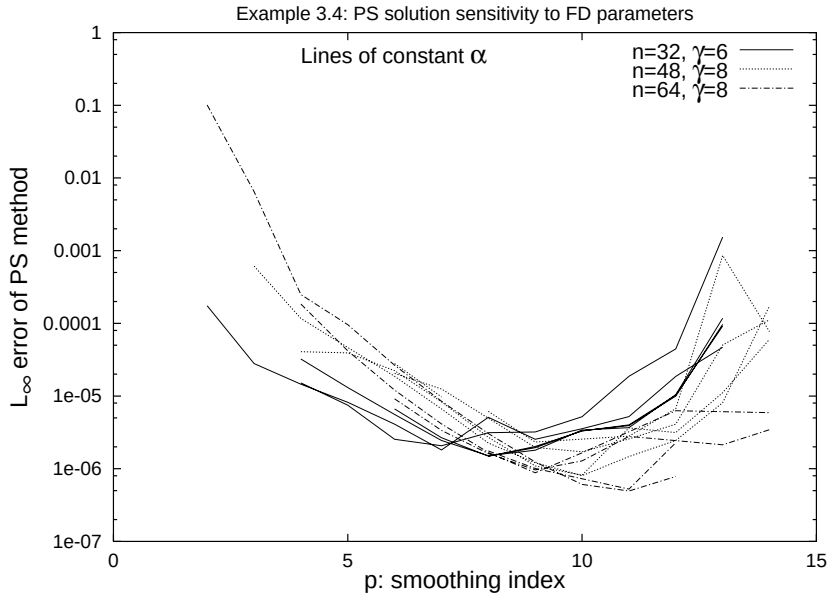

FIG. 10. *Maximum pointwise error versus smoothing index p for the adaptive transformation method applied to* (2.23)*, with $\epsilon = 10^{-3}$ and using $\gamma = 6$ or 8, $m = 164$ and $N = 128$, and various values for n, α, and γ*

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

choice for p varies from problem to problem but does not vary significantly as we vary n and α when solving a particular problem.

From our gathered numerical evidence it would seem that the choice of p is far less critical in obtaining good PS approximations than is the choice of γ. This conclusion is reached by examining contour plots of maximum errors against p and γ (for brevity these are not included here) which show steep gradients in the γ direction and shallow gradients in the p direction.

4. Adaptive transformation method in two dimensions.

4.1. Construction of the coordinate transformation. Here we consider a second-order boundary value problem represented as

$$L(u, u_x, u_y, u_{xx}, u_{xy}, u_{yy}; \epsilon) = 0, \tag{4.1}$$

where, as before, ϵ is a small positive parameter and L is a linear or nonlinear operator. Although the method presented here can generally be applied to problems defined on simply connected domains in $\boldsymbol{R}^2$ we select the rectangular domain $[-1,1]\times[-1,1]$ for simplicity. Suppose that u is subject to Dirichlet boundary conditions

$$\begin{cases} u(-1,y) = u_L(y), \quad u(+1,y) = u_R(y) \text{ for } y \in [-1,1], \\ u(x,-1) = u_B(x), \quad u(x,+1) = u_T(x) \text{ for } x \in [-1,1], \end{cases} \tag{4.2}$$

and that u is continuously differentiable, albeit steep, on $[-1,1]\times[-1,1]$. In this section we assume that we have at our disposal a one-to-one coordinate transformation

$$x = X(\xi,\eta) \text{ and } y = Y(\xi,\eta), \quad \forall (\xi,\eta) \in [-1,1]\times[-1,1], \tag{4.3}$$

where ξ and η denote the spatial coordinates on the computational domain D_c. Furthermore, suppose that the transformations (4.3) satisfy end conditions

$$\begin{cases} X(-1,\eta) = -1, \quad X(+1,\eta) = +1 \quad \forall\, \eta \in [-1,1], \\ Y(\xi,-1) = -1, \quad Y(\xi,+1) = +1 \quad \forall\, \xi \in [-1,1]. \end{cases} \tag{4.4}$$

Under the transformation (4.3) we may write the differential problem (4.1) and (4.2) in terms of the dependent variable $v(\xi,\eta) = u(X(\xi,\eta), Y(\xi,\eta))$ and independent variables ξ and η. As in the analogous transformation of (2.4) to (2.5) for one-dimensional problems we assume that $v(\xi,\eta)$ is sufficiently smooth for PS approximation to yield spectral accuracy.

The method used to generate X and Y is an extension of the one-dimensional method described in section 3. The first stage of the construction involves the solution of problem (4.1) by means of the two-dimensional adaptive finite difference method of Huang and Sloan [18]. Our approach differed from that described in [18] only in the technique adopted for the solution of the coupled nonlinear algebraic system in the grid locations $(x_{i,j}, y_{i,j})$ and the approximations $u_{i,j}$ at these points for $i = 0,1,\ldots,n$; $j = 0,1,\ldots,m$. Here we use a fully coupled system and a Newton iteration with exact Jacobian and continuation. The nonlinear solver is the obvious extension of that outlined for one-dimensional problems under equation (3.6). An alternative approach to the computation of the coarse grid is offered by the adaptive finite volume method proposed by Mackenzie [20].

As in the one-dimensional case, we aim to construct transformations that prevent mesh crossing. The monotonicity condition which we hope to achieve is given in the following definition.

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

DEFINITION 4.1. *The two-dimensional coordinate transformation* (4.3) *is monotonic in* ξ *and* η *in* $[-1,1]\times[-1,1]$ *if, given*

$$-1 \le \xi_1 < \xi_2 \le 1 \quad \textit{and} \quad -1 \le \eta_1 < \eta_2 \le 1,$$

then

$$\begin{aligned} X(\xi_1,\eta) &< X(\xi_2,\eta) \qquad \forall\ \eta \in [-1,1] \ \textit{and} \\ Y(\xi,\eta_1) &< Y(\xi,\eta_2) \qquad \forall\ \xi \in [-1,1]. \end{aligned}$$

Monotonicity in ξ and η, as defined above, will not guarantee the reversibility of the coordinate transformation (4.3): the necessary and sufficient condition for reversibility is that the Jacobian of the transformation be nonzero on the whole computational domain, including the boundary. Here we use the obvious extension of the one-dimensional approach that was described in section 3. Again, it does not guarantee strict monotonicity, but it gives good results for some extremely severe test problems in two space dimensions.

The finite difference adaptive method [18] on the coarse, evenly spaced computational grid

$$\{\xi_i\}_{i=0}^{n}, \quad \{\eta_j\}_{j=0}^{m}$$

produces the physical mesh $(x_{i,j},\, y_{i,j})$ for $i = 0,1,\ldots,n$ and $j = 0,1,\ldots,m$. We then use bilinear interpolation to obtain the values $\overline{x}_{k,\ell}$ and $\overline{y}_{k,\ell}$ for $k = 0,1,\ldots,n_x$ and $\ell = 0,1,\ldots,m_y$, where these values approximate x and y, respectively, at the Chebyshev nodes

$$\overline{\xi}_k = -\cos\frac{\pi k}{n_x}, \quad \overline{\eta}_\ell = -\cos\frac{\pi \ell}{m_y}. \tag{4.5}$$

The transformation for x in (4.3) is approximated by

$$\overline{P}_x(\xi,\eta) = \sum_{i=0}^{n_x}\sum_{j=0}^{m_y} a_{i,j}^{x}\, T_i(\xi)\, T_j(\eta), \tag{4.6}$$

and the interpolatory conditions

$$\left\{\begin{array}{ll} & \overline{P}_x(\overline{\xi}_k,\overline{\eta}_\ell) = \overline{x}_{k,\ell} \\ \text{for} & k = 0,1,\ldots,n_x, \\ & \ell = 0,1,\ldots,m_y, \end{array}\right. \tag{4.7}$$

yield the coefficients $a_{k,\ell}^{x}$. Similarly, the transformation for y in (4.3) is approximated by

$$\overline{P}_y(\xi,\eta) = \sum_{i=0}^{n_x}\sum_{j=0}^{m_y} a_{i,j}^{y}\, T_i(\xi)\, T_j(\eta), \tag{4.8}$$

with coefficients selected such that

$$\left\{\begin{array}{ll} & \overline{P}_y(\overline{\xi}_k,\overline{\eta}_\ell) = \overline{y}_{k,\ell} \\ \text{for} & k = 0,1,\ldots,n_x, \\ & \ell = 0,1,\ldots,m_y. \end{array}\right. \tag{4.9}$$

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Smoothing is introduced to $\overline{P}_x$ and $\overline{P}_y$ by means of a boundary-preserving filter which is now illustrated for the transformation $\overline{P}_x$.

PROCEDURE

(i) Define a bivariate polynomial $\overline{Q}_x$ by $\overline{P}_x(\xi,\eta) = (1-\xi^2)\,\overline{Q}_x(\xi,\eta) + \xi$;

(ii) Write $\overline{Q}_x$ as a double Chebyshev series in the form

$$\overline{Q}_x(\xi,\eta) = \sum_{i=0}^{n_x-2} \sum_{j=0}^{m_y} b^x_{i,j}\, T_i(\xi)\, T_j(\eta).$$

The $b^x_{i,j}$ are obtained using, for each $j = 0, 1, \ldots, m_y$, the recurrence relation

$$b^x_{i,j} := -4a^x_{i+2,j} + 2b^x_{i+2,j} - b^x_{i+4,j} \qquad (b^x_{i,j} \equiv 0 \text{ for } i > n_x - 2)$$

for $i = n_x - 2, n_x - 3, \ldots, 0$, followed by $b^x_{0,j} := 0.5 b^x_{0,j}$;

(iii) Replace $\overline{Q}_x(\xi,\eta)$ by the filtered expansion

$$Q_x(\xi,\eta) = \sum_{i=0}^{n_x-2} \sum_{j=0}^{m_y} \sigma_{i,j}\; b^x_{i,j}\, T_i(\xi)\, T_j(\eta);$$

(iv) Define P_x by

$$P_x(\xi,\eta) = (1-\xi^2)\, Q_x(\xi,\eta) + \xi. \tag{4.10}$$

In (iii) above the filter is conveniently defined by

$$\sigma_{i,j} = \exp\left[-\delta \left(\frac{\sqrt{(i^2+j^2)}}{\sqrt{(n_x^2+m_y^2)}} \right)^{\gamma} \right], \tag{4.11}$$

where δ and γ are positive real numbers. As in the one-dimensional case the parameter δ was set to the value 32 in all numerical computations. The transformation P_y is defined by an analogous procedure, and the transformations (4.3) are given by the identification

$$\begin{cases} x = X(\xi,\eta) := P_x(\xi,\eta), \\ y = Y(\xi,\eta) := P_y(\xi,\eta). \end{cases} \tag{4.12}$$

This completes the construction of the approximation to the map (4.3) and (4.4). The availability of the differentiable map permits the transformation of (4.1) to a problem with dependent variable v and independent variables ξ and η, where $v(\xi,\eta) = u(X(\xi,\eta), Y(\xi,\eta))$. The transformation makes use of the identities

$$\xi_x = \frac{1}{g}\, y_\eta, \quad \xi_y = -\frac{1}{g}\, x_\eta, \quad \eta_x = -\frac{1}{g}\, y_\xi, \quad \eta_y = \frac{1}{g}\, x_\xi, \tag{4.13}$$

where g is the Jacobian of the transformation, $\frac{\partial(X,Y)}{\partial(\xi,\eta)}$. For example, the first derivative terms become

$$u_x = \xi_x\, v_\xi + \eta_x\, v_\eta = (y_\eta\, v_\xi - y_\xi\, v_\eta)/g,$$
$$u_y = \xi_y\, v_\xi + \eta_y\, v_\eta = (-x_\eta\, v_\xi + x_\xi\, v_\eta)/g.$$

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

The second derivative term u_{xx} transforms to

$$u_{xx} = \xi_{xx}\, v_\xi + \eta_{xx}\, v_\eta + (\xi_x)^2 v_{\xi\xi} + 2\xi_x\, \eta_x\, v_{\xi\eta} + (\eta_x)^2 v_{\eta\eta},$$

where differentiation of the appropriate terms in (4.13) yields

$$\begin{cases} \xi_{xx} &= \dfrac{1}{g^3}\big[-(y_\eta)^2 g_\xi + y_\xi y_\eta g_\eta + y_\eta y_{\xi\eta} g - y_\xi y_{\eta\eta} g\big], \\ \eta_{xx} &= \dfrac{1}{g^3}\big[y_\xi y_\eta g_\xi - (y_\xi)^2 g_\eta + y_\eta y_{\xi\xi} g - y_\xi y_{\xi\eta} g\big]. \end{cases} \tag{4.14}$$

u_{xx} is expressed in terms of ξ and η derivatives by means of (4.13) and (4.14). If all second derivatives are transformed in a similar manner the differential problem (4.1) and (4.2) may be expressed in the form

$$\mathcal{L}(v, v_\xi, v_\eta, v_{\xi\xi}, v_{\xi\eta}, v_{\eta\eta}; \epsilon) = 0, \quad (\xi, \eta) \in [-1, 1] \times [-1, 1], \tag{4.15}$$

with boundary conditions

$$\begin{cases} v(-1, Y(-1, \eta)) &= u_L(Y(-1, \eta)) \text{ for } \eta \in [-1, 1], \\ v(+1, Y(+1, \eta)) &= u_R(Y(+1, \eta)) \text{ for } \eta \in [-1, 1], \\ v(X(\xi, -1), -1) &= u_B(X(\xi, -1)) \text{ for } \xi \in [-1, 1], \\ v(X(\xi, +1), +1) &= u_T(X(\xi, +1)) \text{ for } \xi \in [-1, 1]. \end{cases} \tag{4.16}$$

Equations (4.15) and (4.16) are now solved using a standard PS method based on nodes

$$\begin{cases} \xi_i &= -\cos\dfrac{\pi i}{N_x}, \quad i = 0, 1, \ldots, N_x, \\ \eta_j &= -\cos\dfrac{\pi j}{M_y}, \quad j = 0, 1, \ldots, M_y. \end{cases} \tag{4.17}$$

The complete computation uses integers n, m, n_x, m_y, N_x, and M_y that are independent of each other. The main programming effort for the overall method in two dimensions is in the coding of the exact Jacobian of the adaptive finite difference discrete system for the mesh equation incorporating variable adaptivity (α) and smoothing (p). However, this effort need only be expended once, provided the same form of monitor function is used; only a little extra effort is involved in recoding the discretizations of different differential equations to be solved.

4.2. Illustrative examples.

Example 4.1. Consider the convection-diffusion equation

$$\frac{1}{\epsilon}\frac{\partial u}{\partial x} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \omega^2\big[1 - e^{(x-1)/\epsilon}\big]\sin(\omega y), \ 0 < x, y < 1, \tag{4.18}$$

with boundary conditions on the unit square given by the exact solution

$$u(x, y) = \big[1 - e^{(x-1)/\epsilon}\big]\sin(\omega y). \tag{4.19}$$

Here ω and ϵ are prescribed positive constants, with, generally, $\epsilon \ll 1$. The exact solution has a steep front along the line $x = 1$ whose gradient is independent of y. We would therefore expect that mesh adaption will be mostly in the x direction and that

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

the grid will remain orthogonal. Figure 11 shows the adaptive finite difference and transformed PS meshes generated for problem (4.18) with $\omega = \pi$ and $\epsilon = 0.001$ and using the parameter values listed in the caption. Figure 12 shows the corresponding adaptive transformation solution in both physical and computational coordinates; this solution has a maximum pointwise error of 1.474×10^{-4}, whereas the adaptive finite difference method has an error of 4.060×10^{-2}. A cross section of the associated transformation $X(\xi, \eta)$ for $\eta = -1$ is plotted in Figure 13—the cross section is very similar across all values of η; as expected the transformation $Y(\xi, \eta) \approx \eta \ \forall \xi$.

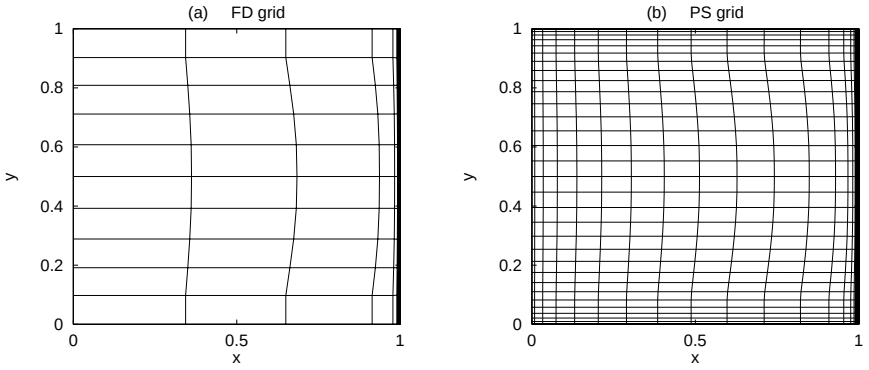

FIG. 11. *Adaptive finite difference and transformed PS meshes for* (4.18) *with* $\omega = \pi$ *and* $\epsilon = 0.001$ *and using* $n = m = 10$, $\alpha = 1$, $p = 4$, $n_x = n_y = 64$, $\gamma = 2$, $N_x = N_y = 32$.

If we now set $\omega = 1.5\pi$ in (4.18) then we would expect a small amount of grid adaption in the y direction and some variation in cross section of $X(\xi, \eta)$ for different values of η. Figure 14 shows the adaptive finite difference and transformed PS meshes generated for problem (4.18) with $\omega = 1.5\pi$ and all other parameter values unaltered. Figure 15 shows the corresponding adaptive transformation solution in both physical and computational coordinates; this solution has a maximum pointwise error of 1.105×10^{-3}, whereas the adaptive finite difference method has an error of 5.839×10^{-2}. Cross sections of the associated transformation $X(\xi, \eta)$ for $\eta = -1$ and $\eta = 0.773$ are plotted in Figure 16—the cross sections do vary slightly with η.

Example 4.2. Fit a PS grid to

$$u = \tanh\left[\frac{1}{\epsilon}\left(\frac{1}{16} - \left(x - \frac{1}{2}\right)^2 - \left(y - \frac{1}{2}\right)^2\right)\right]. \tag{4.20}$$

For small ϵ this function resembles a top hat with front located on the circle $(x-\frac{1}{2})^2+(y-\frac{1}{2})^2 = \frac{1}{16}$. We use the adaptive finite difference method with $n = m = 20$, $\alpha = 4$, and $p = 2$ to generate a finite difference grid for the function with $\epsilon = 0.01$. The transformation method is then used, with settings $n_x = m_y = 64$ and $\gamma = 2$, to generate mappings $X(\xi, \eta)$ and $Y(\xi, \eta)$ which allow u to be transformed into a function of ξ and η. This done, an $N_x = M_y = 32$ PS grid is laid in (ξ, η) space. Figure 17 shows the finite difference and PS grids adapted to the function (4.20). Note that the finite difference grid actually crosses due to the high level of adaption ($\alpha = 4$) and the low level of smoothing ($p = 2$), while the PS grid does not cross. This is a clear case of wrinkles in the initial data being smoothed out by the filtering process used in obtaining $X(\xi, \eta)$ and $Y(\xi, \eta)$. Figure 18 shows how the coordinate

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

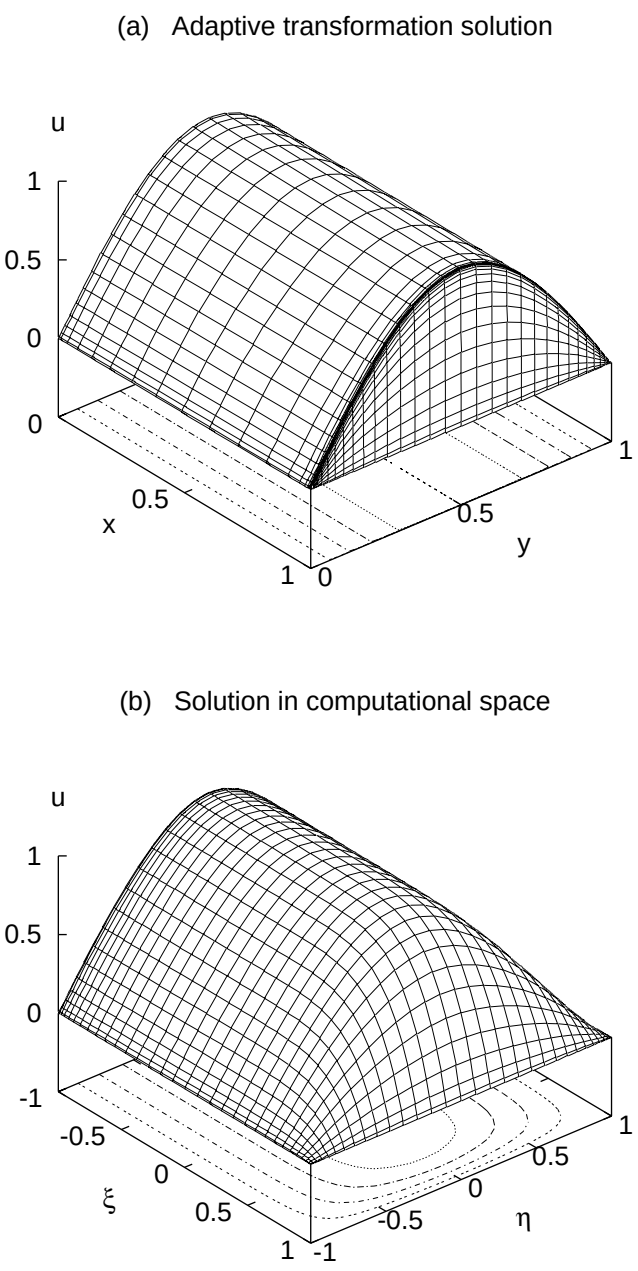

FIG. 12. *Adaptive transformation solution, in physical and computational coordinates of* (4.18) *with* $\omega = \pi$ *and* $\epsilon = 0.001$ *and using* $n = m = 10$, $\alpha = 1$, $p = 4$, $n_x = n_y = 64$, $\gamma = 2$, $N_x = N_y = 32$.

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

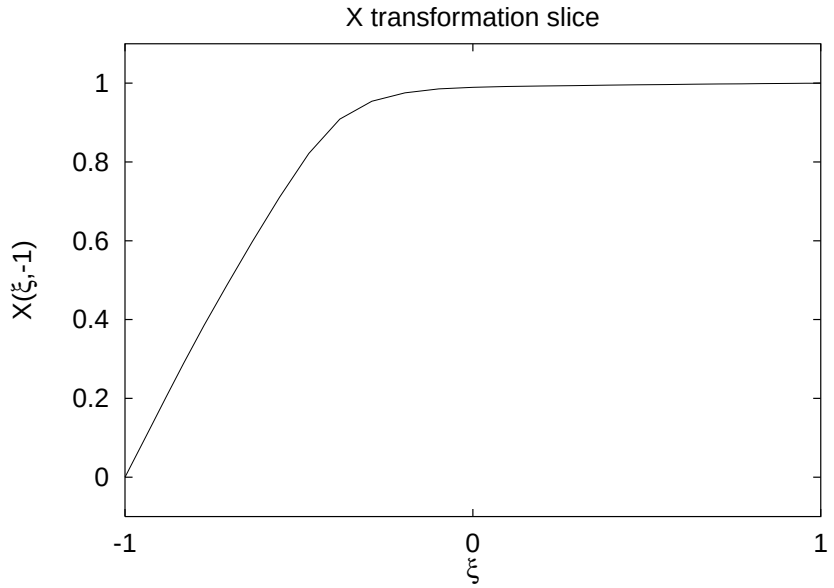

FIG. 13. *Coordinate transformation slice* $X(\xi, -1)$ *for* (4.18) *with* $\omega = \pi$ *and* $\epsilon = 0.001$ *and using* $n_x = n_y = 64$ *and* $\gamma = 2$.

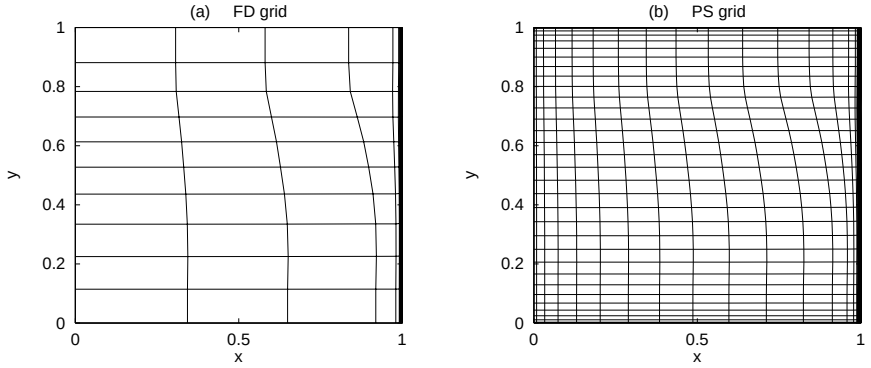

FIG. 14. *Adaptive finite difference and transformed pseudospectral meshes for* (4.18) *with* $\omega = 1.5\pi$ *and* $\epsilon = 0.001$ *and using* $n = m = 10$, $\alpha = 1$, $p = 4$, $n_x = n_y = 64$, $\gamma = 2$, $N_x = N_y = 32$.

transformation has mapped the sharp circular front onto a slope of much shallower gradient. The grid adaption for this function is intrinsically two dimensional, so we would expect considerable variation in the profile of the transformation $X(\xi, \eta)$ as η changes value. Figure 19 plots these profiles for all the discrete values of η, $\eta_i = -\cos\frac{\pi i}{32}$ for $i = 0, 1, \ldots, 32$. The greatest amount of adaption (small gradients at $x = 0.25$ and $x = 0.75$) occurs when $\eta = 0.0$ while the least amount of adaption occurs at $\eta = \pm 1$.

We also constructed a Poisson problem with right-hand side chosen such that (4.20) is the known solution. The previously generated mappings $X(\xi, \eta)$ and $Y(\xi, \eta)$ are now used to transform the Poisson equation which is then solved by the standard PS method. Table 2 lists the maximum pointwise errors of the PS approximations

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

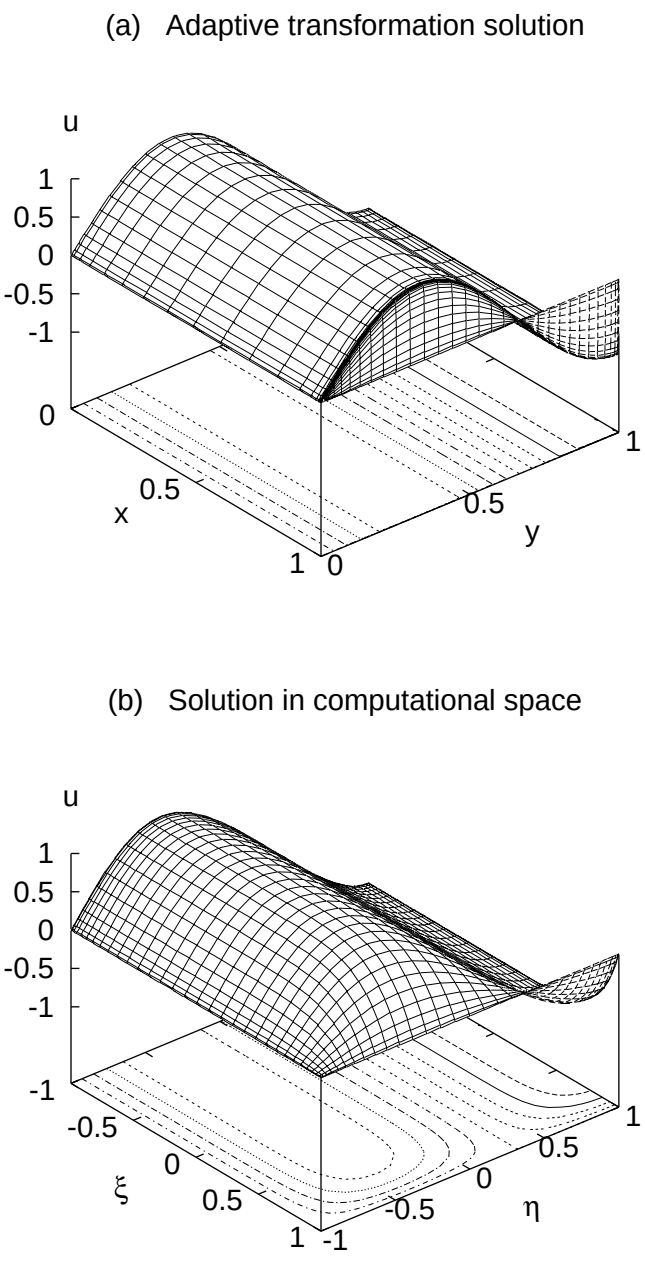

FIG. 15. *Adaptive transformation solution, in physical and computational coordinates of* (4.18) *with* $\omega = 1.5\pi$ *and* $\epsilon = 0.001$ *and using* $n = m = 10$, $\alpha = 1$, $p = 4$, $n_x = n_y = 64$, $\gamma = 2$, $N_x = N_y = 32$.

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

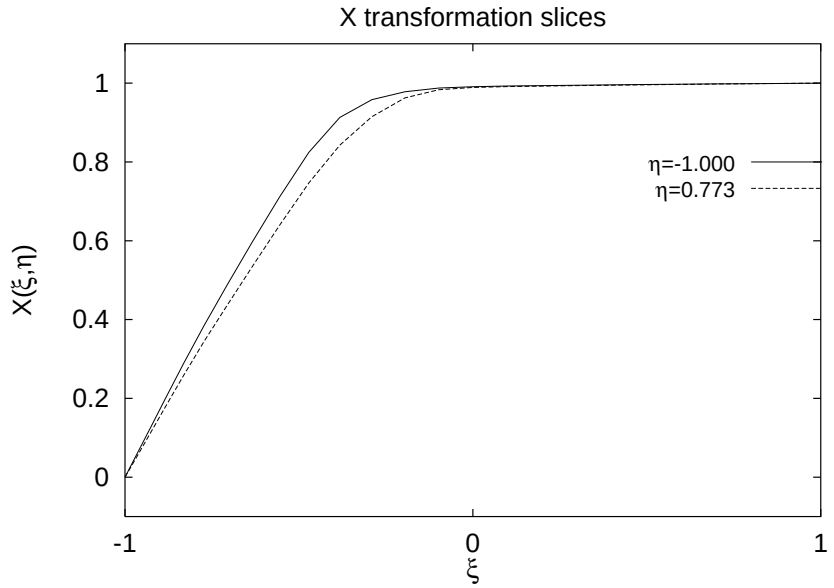

FIG. 16. *Coordinate transformation slices* $X(\xi,-1)$ *and* $X(\xi,0.773)$ *for* (4.18) *with* $\omega = 1.5\pi$ *and* $\epsilon = 0.001$ *and using* $n_x = n_y = 64$ *and* $\gamma = 2$.

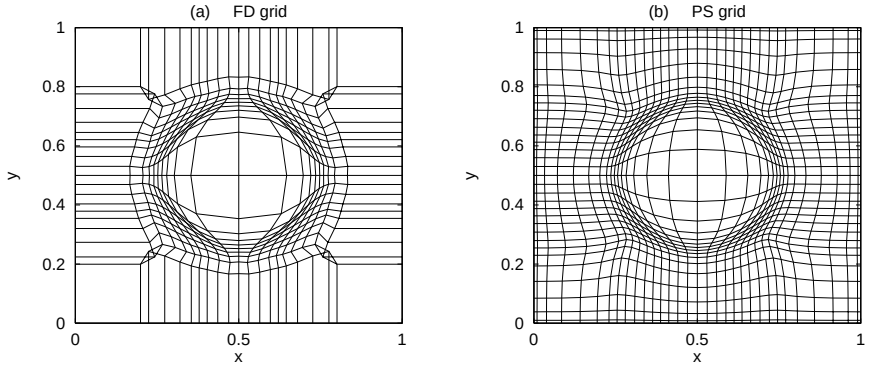

FIG. 17. *Adaptive finite difference and transformed PS meshes for function* (4.20) *with* $\epsilon = 0.01$ *and using* $n = m = 20$, $\alpha = 4$, $p = 2$, $n_x = n_y = 64$, $\gamma = 2$, $N_x = N_y = 32$.

based on $N_x = N_y = N$ PS grids for small values of N. Again the rapid convergence of the PS method is demonstrated.

5. Conclusions and comments. An adaptive coordinate transformation method has been described that enables differential problems with near-singular solutions to be solved by PS methods. A key advantage of our approach is that it describes a mechanism that allows PS discretization to be coupled with other finite difference or finite element methods to give highly accurate computed solutions at little extra cost. The local adaptive methods may be computed at low computational cost on coarse grids to produce input for a PS postprocessing. If the physical PDE is linear then the equation to be solved by the PS method is also linear, and if the physical PDE

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

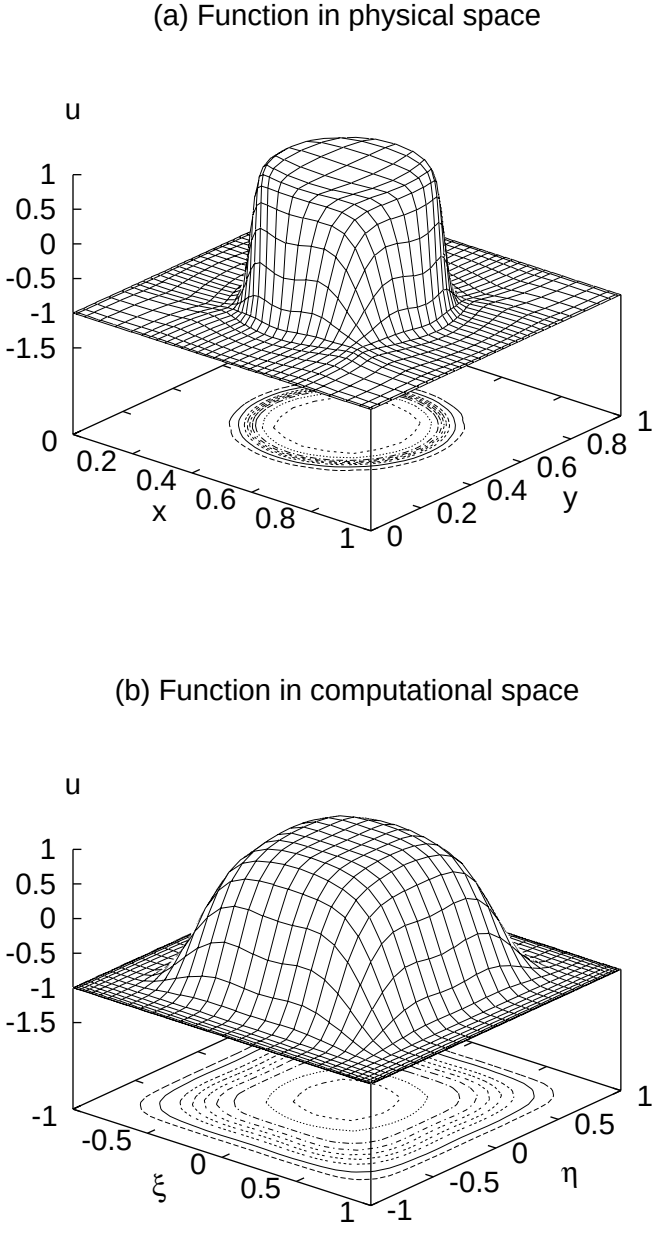

FIG. 18. *Function* (4.20) *with* $\epsilon = 0.01$ *fitted to PS mesh in both physical and computational coordinates using* $n = m = 20$, $\alpha = 4$, $p = 2$, $n_x = n_y = 64$, $\gamma = 2$, $N_x = N_y = 32$.

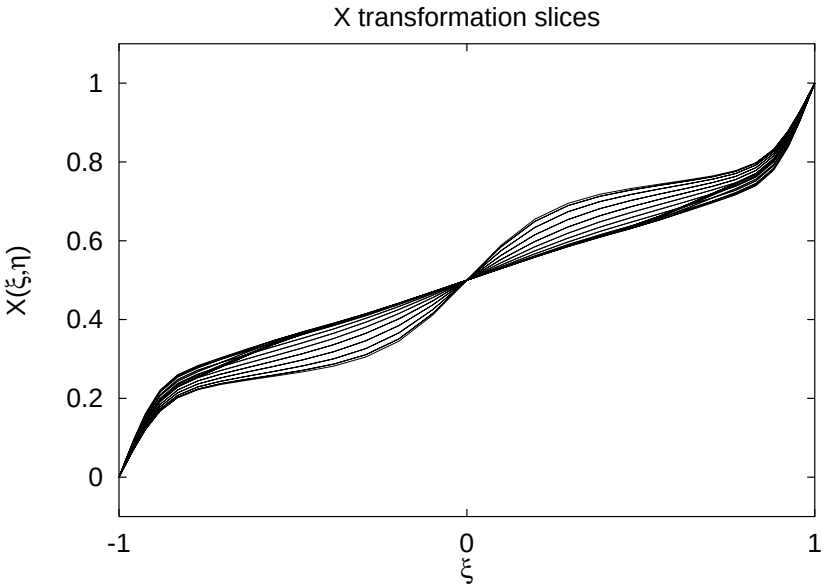

FIG. 19. *Coordinate transformation slices, $X(\xi, \eta)$ over many values of η, for the function* (4.20) *with* $\epsilon = 0.01$ *and using* $n_x = n_y = 64$ *and* $\gamma = 2$.

TABLE 2

Maximum pointwise errors of PS solution to transformed Poisson equation with (4.20) *as exact solution for several small values of* $N_x = N_y = N$.

N	L_∞
16	1.990×10^{-1}
24	1.683×10^{-2}
32	7.635×10^{-4}

is nonlinear the local adaptive solution provides an initial estimate for the iterative solution of the nonlinear PS equations. Numerical results for steady problems show that the method is robust and extremely accurate, even in cases of extreme stiffness. The approach presented here may be regarded as an extension of the adaptive finite difference method described in [18] (other adaptive finite difference or finite element methods might replace that described in [18]): greatly improved accuracy is achieved at little extra computational cost.

The method of Huang and Sloan [18] uses the idea of equidistribution, based on the monitor function (3.3). Ideally, one seeks a method that equidistributes the local error over the domain of the problem, and it would be of interest to examine the error distribution properties of the method presented here. A good coordinate transformation is probably one that is driven by a sharp local error estimate.

Clearly, this method needs further investigation, and it should be applied to more complex differential systems. In this way the strengths and weaknesses may be identified. Currently work is under way on extending the PS coordinate transformation method to time-dependent problems.

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

Downloaded 02/25/15 to 129.237.45.148. Redistribution subject to SIAM license or copyright; see http://www.siam.org/journals/ojsa.php

REFERENCES

[1] J. M. Augenbaum, *An adaptive pseudospectral method for discontinuous problems*, Appl. Numer. Math., 5 (1989), pp. 459–480.

[2] C. Basdevant, M. Deville, P. Haldenwang, J. M. Lacroix, D. Orlandi, A. Patera, R. Peyret, and J. Quazzani, *Spectral and finite difference solutions of Burgers' equation*, Comput. Fluids, 14 (1986), pp. 23–41.

[3] A. Bayliss, D. Gottlieb, B. Matkowsky, and M. Minkoff, *An adaptive pseudospectral method for reaction diffusion problems*, J. Comput. Phys., 81 (1989), pp. 421–443.

[4] A. Bayliss and B. Matkowsky, *Fronts, relaxation oscillations and period doubling in solid fuel combustion*, J. Comput. Phys., 71 (1987), pp. 147–188.

[5] A. Bayliss and E. Turkel, *Mappings and accuracy for Chebyshev pseudospectral approximations*, J. Comput. Phys., 101 (1992), pp. 349–359.

[6] J. U. Brackbill and J. S. Saltzman, *Adaptive zoning for singular problems in two dimensions*, J. Comput. Phys., 46 (1982), pp. 342–368.

[7] W. Cai, D. Gottlieb, and C.-W. Shu, *Essentially nonoscillatory spectral Fourier methods for shock wave calculation*, Math. Comput., 52 (1989), pp. 389–410.

[8] W. Cai, D. Gottlieb, and C.-W. Shu, *On one-sided filters for spectral Fourier approximation of discontinuous functions*, SIAM J. Numer. Anal., 29 (1992), pp. 905–916.

[9] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral Methods in Fluid Dynamics*, Springer-Verlag, Berlin, Heidelberg, New York, 1988.

[10] E. A. Dorfi and L. O'C. Drury, *Simple adaptive grids for* 1*-D initial value problems*, J. Comput. Phys., 69 (1987), pp. 175–195.

[11] B. Fornberg and D. M. Sloan, *A review of PS methods for solving PDEs*, Acta Numerica, (1994), pp. 203–267.

[12] D. Gottlieb, L. Lustman, and S. A. Orszag, *Spectral calculations in one-dimensional inviscid compressible flow*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 296–310.

[13] D. Gottlieb and S. A. Orszag, *Numerical Analysis of Spectral Methods*, SIAM, Philadelphia, 1977.

[14] H. Guillard and R. Peyret, *On the use of spectral methods for the numerical solution of stiff problems*, Comput. Meth. Appl. Mech. Engrg., 66 (1988), pp. 17–43.

[15] D. F. Hawken, J. J. Gottlieb, and J. S. Hansen, *Review of some adaptive node-movement techniques in finite element and finite difference solutions of PDEs*, J. Comput. Phys., 95 (1991), pp. 254–302.

[16] W.-Z. Huang and R. D. Russell, *Analysis of Moving Mesh PDEs with Spatial Smoothing*, Simon Fraser Mathematics Research Report, Simon Fraser University, Burnaby, BC, 1993, pp. 93–17.

[17] W.-Z. Huang and D. M. Sloan, *A new pseudospectral method with upwind features*, IMA J. Numer. Anal., 13 (1993), pp. 413–430.

[18] W.-Z. Huang and D. M. Sloan, *A simple adaptive grid method in two dimensions*, SIAM J. Sci. Comput., 15 (1994), pp. 776–797.

[19] D. Leviatan, *Monotone and comonotone polynomial approximation revisited*, J. Approx. Theory, 53 (1988), pp. 1–16.

[20] J. A. Mackenzie, *The efficient generation of simple two-dimensional adaptive grids*, SIAM J. Sci. Comput., (1998), to appear.

[21] A. Majda, J. McDonough, and S. Osher, *The Fourier method for non-smooth initial data*, Math. Comput., 32 (1978), pp. 1041–1081.

[22] L. S. Mulholland and D. M. Sloan, *The effect of filtering on the pseudospectral solution of evolutionary partial differential equations*, J. Comput. Phys., 96 (1991), pp. 369–390.

[23] E. Tadmor, *Convergence of spectral methods for nonlinear conservation laws*, SIAM J. Numer. Anal., 26 (1989), pp. 30–44.

[24] E. Tadmor, *Shock capturing by the spectral viscosity method*, Comput. Meth. Appl. Mech. Engrg., 80 (1990), pp. 197–208.

[25] E. Tadmor, *Local error estimates for discontinuous solutions of nonlinear hyperbolic equations*, SIAM J. Numer. Anal., 28 (1991), pp. 891–906.

[26] T. Tang and M. R. Trummer, *Boundary layer resolving pseudospectral methods for singular perturbation problems*, SIAM J. Sci. Comput., 17 (1996), pp. 430–438.

[27] J. F. Thompson, Z. U. A. Warsi, and C. W. Mastin, *Numerical Grid Generation*, North–Holland, New York, 1985.

[28] A. B. White, *On selection of equidistribution meshes for two-point boundary value problems*, SIAM J. Numer. Anal., 16 (1979), pp. 472–502.