Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 198 (2004) 310–325

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Adjoint sensitivity analysis for time-dependent partial differential equations with adaptive mesh refinement ☆

Shengtai Li [a], Linda Petzold [b,*]

[a] Theoretical Division, Los Alamos National Laboratory, NM 87544, USA
[b] Department of Computer Science, University of California, Santa Barbara, Santa Barbara, CA 93106-5070, USA

Received 6 June 2002; received in revised form 13 January 2003; accepted 15 January 2003
Available online 18 February 2004

Abstract

A new adjoint sensitivity analysis approach is presented for time-dependent partial differential equations with adaptive mesh refinement. The new approach, called ADDA, combines the best features of both the adjoint of the discretization (AD) and discretization of the adjoint (DA) approaches. It removes the obstacles of applying AD to adaptive methods and, in contrast to DA, requires for its use only a minimal amount of knowledge about the formulation of adjoint PDEs and their boundary conditions. The effectiveness and efficiency of ADDA are demonstrated for several numerical examples.
© 2004 Elsevier Inc. All rights reserved.

Keywords: Adjoint method; Sensitivity analysis; Partial differential equations; Adjoint PDE; Discrete approach; Continuous approach; Adaptive mesh method

1. Introduction

In recent years there has been a growing interest in sensitivity analysis for large-scale systems governed by partial differential equations (PDEs). The results of sensitivity analysis have wide-ranging applications in science and engineering, including optimization, parameter estimation, model simplification, data assimilation, optimal control, uncertainty analysis and experimental design. Two important classical fields that make extensive use of sensitivity analysis are inverse heat-conduction problems [11] and shape design in aerodynamic optimization [15,16]. The mathematical theory for control systems governed by PDEs has created a framework for the formulation of inverse design and general aerodynamic problems at a reduced

☆ This work was supported by grants: DOE DE-FG03-00ER25430, NSF/ITR ACI-0086061, and NSF/KDI ATM-9873133.
* Corresponding author. Tel.: +1-805-893-5362; fax: +1-805-893-5435.
E-mail address: petzold@engineering.ucsb.edu (L. Petzold).

computational cost. Gradient-based methods are often used for solving optimization problems involving PDEs. The gradient can be calculated via sensitivity methods, which is more accurate than finite-difference methods. In the pioneering work of Jameson [15], the gradient is calculated indirectly by solving the adjoint equations.

Sensitivity methods for PDE problems have been studied by many authors (see [1,4,5,13–15]). Most of this work has focused on steady-state PDE control and design optimization problems. For the transient case, Bewley [4] has given an excellent review on the adjoint-based optimization approach for flow-control problems.

In this paper we focus on adjoint methods for general transient PDE systems. Although many of the results from the steady-state system can be readily extended to the time-dependent PDE system, the time-dependent system has some unique features that must be treated differently. For example, apart from the boundary conditions, we now have initial conditions that must be determined.

Given a parameter-dependent PDE system

$$F(t,u,u_t,u_x,u_{xx},p)=0, \tag{1}$$

and a vector of objective functions $G(x,u,p)$ that depend on u and p, the sensitivity problem usually takes the form: find $\mathrm{d}G/\mathrm{d}p$, where p is a vector of parameters. By the chain rule, the sensitivity $\mathrm{d}G/\mathrm{d}p$ is given by

$$\frac{\mathrm{d}G}{\mathrm{d}p}=\frac{\partial G}{\partial u}\frac{\partial u}{\partial p}+\frac{\partial G}{\partial p}. \tag{2}$$

If we treat (1) as a nonlinear system about u and p, say $H(u,p)=F(t,u,u_t,u_x,u_{xx},p)$, we have the following relationship:

$$\frac{\partial H}{\partial u}\frac{\partial u}{\partial p}+\frac{\partial H}{\partial p}=0.$$

Assuming that $\partial H/\partial u$ is boundedly invertible, the sensitivity $\mathrm{d}G/\mathrm{d}p$ is given by

$$\frac{\mathrm{d}G}{\mathrm{d}p}=-\frac{\partial G}{\partial u}\left(\frac{\partial H}{\partial u}\right)^{-1}\frac{\partial H}{\partial p}+\frac{\partial G}{\partial p}. \tag{3}$$

There are two basic methods to calculate $\mathrm{d}G/\mathrm{d}p$ in (3): forward and adjoint. Forward methods calculate $\partial u/\partial p=(\partial H/\partial u)^{-1}\partial H/\partial p$ first, which is the solution of the sensitivity PDE for each uncertain parameter. Adjoint methods compute $\partial G/\partial u(\partial H/\partial u)^{-1}$ first, which is the solution of the adjoint PDE. The sensitivity and adjoint PDEs will be defined later. Although both methods yield the same analytical sensitivities, the computational efficiency may be quite different, depending on the number of objective functions (dimension of G) and the number of sensitivity parameters (dimension of p). The forward method is attractive when there are relatively few parameters or a large number of objective functions, while the adjoint method is more efficient for problems involving a large number of sensitivity parameters and few objective functions. In this paper we focus on the adjoint method.

Two approaches can be taken for each method. In the first, called the *discrete* approach, we approximate the PDE by a discrete nonlinear system and then differentiate the discrete system with respect to the parameters. The discrete approach is easy to implement with the help of *automatic differentiation* tools, such as ADIFOR [6] and TAMC [12]. However, when the mesh is solution or parameter dependent (e.g., for an *adaptive* mesh or moving boundary), or a nonlinear discretization scheme (e.g., upwinding) is used, the discrete approach may not be computationally effective.

In the second, called the *continuous* approach, we differentiate the PDE with respect to the parameters first and then discretize the sensitivity or adjoint PDEs to compute the approximate sensitivities. The system resulting from the continuous approach is usually much simpler than that from the discrete approach.

It has been demonstrated [9] that a fully discrete approach is not very accurate and efficient for the sensitivity analysis of adaptive numerical integration of ordinary differential equations (ODEs). Therefore, in previous work for ODE and differential-algebraic equation (DAE) systems we have made use of the discrete approach only to generate the sensitivity and/or adjoint ODE (or DAE), which is then solved by adaptive numerical integration. In [7], we describe such an adjoint method for DAE systems of index up to 2.

Adaptive methods, including adaptive mesh refinement (AMR) [3,18] and other types of adaptive methods, have become more and more popular in the numerical simulation of transient PDE systems. Several AMR software packages (e.g., CHOMBO [8] from the Lawrence Berkerly National Laboratory, FLASH [10] from the flash center of the University of Chicago, and many others) have been developed and used in the large-scale simulation of physical systems. However, few of them have the sensitivity analysis capability. The adaptive regions and the number of state variables usually varies with time, which presents difficulties in sensitivity analysis and optimization. In previous work [21], we have proposed a framework for forward sensitivity analysis and optimal control for transient PDEs with AMR. In this paper we study how to apply the adjoint method to transient PDEs with AMR.

It is well known that the method of lines (MOL) can transform a PDE system into an ODE or DAE system by spatial discretization. Thus the sensitivity calculation methods in [7] can be used if the semi-discretized PDE is obtained. However, we have observed that the *adjoint of the discretization* (AD) may not be consistent with a PDE, and the adjoint variables are not smooth on an adaptive grid. Therefore, if the adaptive region is changing with time, the interpolation for the adjoint variables between different grids will introduce large errors.

A second possibility, called *discretization of the adjoint* (DA), is to form the adjoint PDE system along with its boundary conditions, and then discretize it. The DA approach is naturally consistent with the adjoint PDE system. Therefore, the adaptive grid method and interpolation can be used without difficulty. However, it is difficult to formulate proper boundary conditions for the adjoint of a general PDE system. To the best of our knowledge an algorithm for generating the boundary conditions does not exist for a general PDE system. Moreover, the adjoint system may become inadmissible for some objective functionals (see [1,2]), where the boundary conditions (or initial conditions) for the adjoint PDE system cannot be formulated properly.

Our objective is to automate the adjoint sensitivity analysis for a PDE system as much as possible, in particular for PDE problems where AMR is used. In this paper we propose an approach to combine the AD method and the DA method in an efficient manner so that it can be used with AMR. The new approach combines the best features of both the AD and DA methods. It removes the obstacles of applying AD to adaptive methods and requires only a minimal amount of knowledge about the formulation of adjoint PDEs and their boundary conditions. Both the AD and DA methods are used in this new approach which we call ADDA, but are applied in different regions. The new approach is able to use existing software, such as an AMR package for the mesh refinement, automatic differentiation tools for the forward solver to generate the derivative codes, and DASPKADJOINT [18] for the sensitivity calculation to help automate this process as much as possible.

The outline of this paper is as follows. In Section 2 we outline the calculation of the sensitivity by the adjoint method and derive the adjoint PDE system and its boundary and initial conditions. In Section 3 we describe the AD approach and illustrate its inconsistency and its difference from the DA approach through a simple example. In Section 4 we describe the strategy of combining AD and DA into one method ADDA and discuss its implementation in the context of AMR. Numerical examples are presented in Section 5 which demonstrate the effectiveness of the ADDA method.

2. Sensitivity calculation by the adjoint method

2.1. Sensitivity PDE

Given a PDE in the domain Ω, without loss of generality we will assume that the PDE involves only up to second order spatial derivatives

$$F(t, u, u_t, D_x(u), D_x^2(u), p) = 0.$$

The objective function will be defined as an integral in both the time and space domains by

$$G(x, u, p) = \int_0^T \int_\Omega g(t, x, u, p)\,\mathrm{d}x\,\mathrm{d}t. \tag{4}$$

Then the sensitivity of $G(x, u, p)$ with respect to p can be calculated via either the forward or the adjoint sensitivity method.

We first consider the forward sensitivity method. The sensitivity is given by

$$\frac{\mathrm{d}G}{\mathrm{d}p} = \int_0^T \int_\Omega (g_p + g_u s)\,\mathrm{d}x\,\mathrm{d}t,$$

where $s = u_p$ is obtained by solving the sensitivity PDE

$$\frac{\mathrm{d}F}{\mathrm{d}p} = F_p + F_u s + F_{u_t} s_t + F_{u_x} s_x + F_{u_{xx}} s_{xx} = 0.$$

If the dimension of p is large, then the number of sensitivity PDEs is large. Thus the method can become intractable for a large number of parameters.

2.2. Adjoint PDE and sensitivity evaluation

To derive the adjoint PDE and its boundary conditions, we introduce the adjoint variable v. Since $F(x, u, p) = 0$ on the domain Ω, the derived function satisfies

$$G(x, u, p) = G(x, u, p) - \int_0^T \int_\Omega vF\,\mathrm{d}x\,\mathrm{d}t$$

and the sensitivity $\mathrm{d}G/\mathrm{d}p$ is given by

$$\frac{\mathrm{d}G}{\mathrm{d}p} = \int_0^T \int_\Omega (g_u s + g_p) - \int_0^T \int_\Omega v \frac{\mathrm{d}F}{\mathrm{d}p}. \tag{5}$$

Using integration by parts, we obtain

$$\int_0^T \int_\Omega v \frac{\mathrm{d}F}{\mathrm{d}p} = \int_0^T \int_\Omega vF_p + \big(vF_u - (vF_{u_t})_t - (vF_{u_x})_x + (vF_{u_{xx}})_{xx}\big)s\,\mathrm{d}x\,\mathrm{d}t + \int_\Omega vF_{u_t} s\big|_0^T\,\mathrm{d}x$$
$$+ \int_0^T \int_{\partial\Omega} (vF_{u_x} s + vF_{u_{xx}} s_x - (vF_{u_{xx}})_x s)\,\mathrm{d}l.$$

Under certain circumstances, we can derive a set of boundary conditions for v so that s and s_x either disappear or are specified via Dirichlet boundary conditions. That is to say, the last term is easy to compute without solving for the transient values of s and s_x, if the boundary conditions are chosen properly. We denote the last term by BT. Defining v to satisfy

$$vF_u - (vF_{u_t})_t - (vF_{u_x})_x + (vF_{u_{xx}})_{xx} = g_u, \tag{6}$$

Eq. (5) becomes

$$\frac{dG}{dp} = \int_0^T \int_\Omega (g_p - vF_p) - \int_\Omega vF_{u_t}s|_0^T \, dx - BT. \tag{7}$$

Eq. (6) is called the adjoint PDE for the objective function G. With proper initial conditions at $t = T$, (6) can be solved backward for the value of v at initial time $t = 0$. In order to compute the sensitivity easily in (7) without solving the forward sensitivity PDE, the initial conditions for v at $t = T$ must be determined in such a way that the term $vF_{u_t}s(T)$ is either eliminated or replaced by another term that is easy to compute.

If the PDAE has index (see [19,20]) of no more than one in time, we can choose v at $t = T$ so that $vF_{u_t}|_T = 0$ [7]. Then the sensitivity for the objective function is given by

$$\frac{dG}{dp} = \int_0^T \int_\Omega (g_p - vF_p) + \int_\Omega (vF_{u_t})|_{t=0}s_0 \, dx - BT.$$

For a standard PDE system, where F_{u_t} is an identity matrix or has full rank, the initial condition for the adjoint PDE (6) is $v = 0$. If the adjoint PDAE is index-1, the index-1 equation must be satisfied at the initial time. For an index-2 PDAE system, the condition $vF_{u_t}|_T = 0$ may conflict with the index-2 constraints. Hence it needs to be reconsidered carefully [7].

When the objective function is not of integral form in time, i.e., when the cost function is $\int_\Omega g(x, u, p) \, dx$, the above procedures result in an adjoint PDE system of homogeneous form

$$vF_u - (vF_{u_t})_t - (vF_{u_x})_x + (vF_{u_{xx}})_{xx} = 0, \quad v \in \Omega \tag{8}$$

with the same boundary conditions, and a different initial condition that satisfies

$$g_u - v(T)F_{u_t} = z(T), \tag{9}$$

where $z(T)$ may be zero if F_{u_t} has full rank (index-0 case). For a PDAE with time index greater than zero, $z(T)$ has a different form that may depend on the structure of the system. We should point out that, unlike the DAE case [7], the initial conditions $v(T)$ must be subject to the boundary conditions of the adjoint PDE. $z(T)$ may not be zero at boundaries if the boundary conditions are inconsistent with (9), even for the index-0 case. Finally, the sensitivity of $G = \int_\Omega g(x, u, p) \, dx$ with respect to p becomes

$$\frac{dG}{dp} = \int_\Omega g_p \, dx - \int_0^T \int_\Omega vF_p \, dx + \int_\Omega vF_{u_t}s(0) \, dx + \int_\Omega z(T)s(T) \, dx - BT, \tag{10}$$

where $z(T)s(T)$ can be calculated without evaluation of $s(T)$ [7] by exploiting the structure of the adjoint system (8) and the forward sensitivity PDE system.

If the proper boundary and initial conditions can be derived, the adjoint PDE system is discretized spatially and then solved backward in time. The sensitivity can be calculated via (10). This approach is called the discretization of the adjoint (DA) method.

2.3. Boundary conditions

The boundary conditions (BCs) should be derived in such a way that terms related to s and s_x disappear. It has been shown [13] that the boundary integral terms in the primal objective function lead to inhomogeneous BCs for the adjoint, while inhomogeneous BCs for the primal problem lead to boundary terms in the adjoint functional.

For a scalar PDE, the boundary conditions for the adjoint PDE can be obtained in a consistent manner. However, the boundary conditions for systems of PDEs are much more complicated (see [13]). Under certain restrictions, the boundary conditions exist and can be derived as described in [13]. We are aware of no approach for deriving boundary conditions for the adjoint PDE, in the general case.

2.4. Inadmissible objective functional

As pointed out by many authors (see [1,2]), some objective functionals may be “inadmissible” to an adjoint PDE problem. For partial differential algebraic equations (PDAEs) with index [19] in time of greater than zero, e.g. Navier–Stokes equations, the objective functional is more likely to be inadmissible.

For steady-state PDE problems, some inadmissible functionals can become admissible through the incorporation of auxiliary boundary equations (ABEs) [2]. We attempted to apply this idea to the time-dependent PDE problem. However, if $t = T$ is treated as one of the boundaries of the time domain, there are no boundary conditions specified at $t = T$. Hence it is difficult to apply the ABE method for the initial conditions. In many cases, an inadmissible problem can become admissible if the boundary conditions are also incorporated during the derivation of the adjoint PDE. However, this approach is highly problem-dependent and cannot be applied generally.

3. Adjoint of the discretization

In this section we study the AD approach. We assume that the PDE system has been transformed into an ODE/DAE system following semi-discretization in space. The adjoint ODE/DAE system is obtained by taking the adjoint of the discretized PDE system. The cost functional is also discretized spatially. The sensitivity of the functional can be calculated by the methods proposed for ODEs and DAEs in [7].

It is clear that no explicit boundary conditions are needed for the AD approach. The AD method also works for any cost functional; inadmissibility is not an issue. Moreover, the adjoint ODE system can be generated automatically via automatic differentiation tools (see TAMC [12] and its applications). It is ideal to a non-expert to calculate the sensitivity without knowing much details of the simulation codes.

3.1. Inconsistency of AD

As pointed out in [22], AD may not be consistent with the adjoint PDE if a nonlinear high-resolution discretization scheme, such as an upwinding scheme, is used. This type of inconsistency exists even for the forward sensitivity method. By the equivalence theorem, consistency (and stability) of the discrete scheme is equivalent to convergence. We have found that this type of inconsistency does not lead to incorrect results for the sensitivity of the objective functional. We think the reason is that although AD does not converge to the adjoint PDE pointwise, it converges in L^2, i.e., in a weak sense. It is the latter convergence that matters in calculating the gradient of the cost functional.

There is another type of inconsistency that can result from the spatial discretization near the boundaries, including the external boundaries and internal boundaries which are generated during the mesh adaptation. We illustrate this by a simple example. Consider the one-dimensional heat equation

$$u_t = u_{xx} \tag{11}$$

with boundary conditions

$$u_x(0) = 0, \quad u(1) = 1. \tag{12}$$

If central differencing is adopted on a uniform grid, the discretization yields

$$\begin{aligned} \dot{u}_1 &= \frac{2u_2 - 2u_1}{h^2}, \\ \dot{u}_i &= \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}, \quad i = 2, \ldots, N-1, \\ \dot{u}_N &= 0, \end{aligned}$$

where the first equation is obtained by adding a ghost boundary point u_0 and the boundary condition

$$u_x(0) = \frac{u_2 - u_0}{2h} = 0.$$

The equations for the adjoint of the discretization are

$$\begin{aligned} -\dot{v}_1 &= \frac{v_2 - 2v_1}{h^2}, \\ -\dot{v}_2 &= \frac{2v_1 - 2v_2 + v_3}{h^2}, \\ -\dot{v}_i &= \frac{v_{i+1} - 2v_i + v_{i-1}}{h^2}, \quad i = 3, \ldots, N-2, \\ -\dot{v}_{N-1} &= \frac{-2v_{N-1} + v_{N-2}}{h^2}, \\ -\dot{v}_N &= \frac{v_{N-1}}{h^2}. \end{aligned}$$

From Section 2, the adjoint PDE and its boundary conditions for an objective function $G = \int_0^1 g(u)\,\mathrm{d}x$ are given by

$$-v_t = v_{xx},$$

$$v_x(0) = 0, \quad v(1) = 0.$$

Central differencing applied to the adjoint PDE yields

$$\begin{aligned} -\dot{v}_1 &= \frac{2v_2 - 2v_1}{h^2}, \\ -\dot{v}_i &= \frac{v_{i+1} - 2v_i + v_{i-1}}{h^2}, \quad i = 2, \ldots, N-1, \\ -\dot{v}_N &= 0. \end{aligned}$$

It is clear that AD is *not* the same as DA. The difference between DA and AD occurs for the equations defined at x_1 and x_2, x_{N-1} and x_N. Further investigation shows that for this example, AD is not consistent with *any* continuum PDE, which is undesirable since by the equivalence theorem, consistency (and stability) of the discrete scheme is equivalent to convergence.

In many cases the inconsistency can be removed by some manipulations. For example, by variable substitution $w_1 = (1/2)v_1$, the first two equations at x_1 and x_2 become consistent with the adjoint PDE; by adding a new variable $w_N = 0$ and rewriting the last two equations as

$$-\dot{v}_{N-1} = \frac{w_N - 2v_{N-1} + v_{N-2}}{h^2},$$
$$-\dot{w}_N = 0,$$
$$-\dot{v}_N = \frac{v_{N-1}}{h^2},$$

the last two equations can be made to be consistent.

It is worthwhile to make several points about this type of manipulation. First, the sensitivity calculation method must be changed after the manipulation. The sensitivity evaluation method for the AD approach, described in [7], cannot be used directly. Neither can the evaluation method for the DA approach (10) be used without intervention. Second, the manipulation is highly problem-dependent and cannot be done automatically. It is error-prone and cumbersome to look into the AD-generated code and modify those parts related to the boundary discretizations for a large-scale application. Finally, it is not suitable for the artificial internal boundaries generated by the adaptive mesh method.

3.2. Difference between AD and DA

In this subsection we analyze the AD and DA methods, identify where the difference between AD and DA comes from, and how to eliminate or reduce it.

During the derivation of the adjoint PDE, we used the product $\langle v, F\rangle = \int_0^T \int_\Omega vF\,\mathrm{d}x\,\mathrm{d}t$. If we have discrete values of v and F at grid points, the product can be evaluated by a numerical integration method. If the trapezoidal rule is used on a one-dimensional grid, we obtain

$$\langle v, F\rangle = \int_\Omega vF\,\mathrm{d}x = V^{\mathrm{T}}MF + \mathrm{O}(h^3) = (V, F)_M + \mathrm{O}(h^3), \tag{13}$$

where $V = (V_1, \ldots, V_N)^{\mathrm{T}}$, $F = (F_1, \ldots, F_N)^{\mathrm{T}}$, and $M = \mathrm{diag}(0.5h, h, \ldots, h, 0.5h)$. An alternative way to evaluate the product is to first approximate v and F with basis functions at the grid points and then do the integration. If the first order chapeau (hat) function is used for a 1D grid, then

$$M = \frac{h}{6}\begin{pmatrix} 2 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 2 \end{pmatrix}.$$

Similarly, the objective function can be integrated in space by the same method

$$\int_\Omega g_u s\,\mathrm{d}x = g_U MS = (g_U, S)_M = (Mg_U, S),$$

where $(\cdot,\cdot)$ is the standard inner-product in R^N.

Assume A is a representation of the linear differential operator $\mathscr{D}$ in discrete space. If integration by parts is used in space, then the product is

$$\langle v, F\rangle = \langle v, \mathscr{D}s\rangle = \int_\Omega v\cdot\mathscr{D}s\,\mathrm{d}x = BT + \langle\mathscr{D}^* v, s\rangle,$$

which results in

$$(V, AS)_M = (BV, S)_M + BT + \mathrm{HOT}, \tag{14}$$

where B represents the discretization of the adjoint operator $\mathscr{D}^*$, and HOT represents the high order term $\mathrm{O}(h^q)$ from the discretization. For example (11) and boundary conditions (12), if the central-difference is used on a uniform grid then

$$B = A = \frac{1}{h^2}\begin{pmatrix} -2 & 2 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & & 0 \end{pmatrix}. \tag{15}$$

The discretization via the method of lines (MOL) of a forward linear problem yields a linear ODE

$$\dot{U} = AU$$

and objective function (assume the integration is over the whole time domain)

$$G(U) = \int_0^T Mg(U)\,\mathrm{d}t.$$

The conventional AD is with respect to the standard inner-product $(\cdot,\cdot)$ and generates the following adjoint ODE:

$$-\dot{V} = A^*V + Mg_U \tag{16}$$

with initial conditions $V = 0$ at $t = T$. However, the adjoint operation in DA is with respect to the inner product (13) and yields

$$-\dot{V} = BV + g_U \tag{17}$$

with initial condition $V = 0$. It is clear from (15) that B is different from A^*, and Mg_U is not g_U. Another difference between AD and DA comes from the sensitivity calculation. There is an extra term BT in (14), which does not arise in the derivation of AD. This boundary term BT, which comes from the integration by parts in space, is calculated as part of the sensitivity.

To sum up, AD and DA use two *different inner products* during the derivation, and DA also uses *integration by parts* and has an extra term BT in the sensitivity evaluation. In the following, we will show that even if AD uses the same inner product as DA, it still may not be consistent with the adjoint PDE.

The inner-product $(V,F)_M$ is equivalent to (MV,F). If we let $\tilde{V} = M^{-1}V$ in AD system (16), then

$$-\dot{\tilde{V}} = M^{-1}A^*M\tilde{V} + g_U, \tag{18}$$

which is more similar to DA system (17) than to (16). For an objective function that is defined only at the final time, AD and DA have different initial conditions. The transformation $\tilde{V} = M^{-1}V$ can transform the initial conditions of AD to the initial conditions of DA. Since

$$(M^{-1}A^*MV,S)_M = (V,AS)_M = (BV,S)_M + BT + \mathrm{HOT},$$

we obtain

$$\left((M^{-1}A^*M - B)V,S\right)_M = BT + \mathrm{HOT}. \tag{19}$$

Since M, A, and B are not unique, $M^{-1}A^*M$ may not be a good approximation to B even if $BT = 0$. For example, if A is a second order scheme and B is a fourth order scheme, $M^{-1}A^*M$ can be quite different from B. However, this does not mean that $M^{-1}A^*M$ is not consistent with the adjoint PDE. After studying several one-dimensional problems, we found that for a given discretization scheme A, there was an integration scheme M such that $M^{-1}A^*M$ was consistent with the adjoint PDE internally. General speaking, if every cell or node on a grid is treated in the same way during the discretization and integration, $M^{-1}A^*M$ is consistent with the adjoint PDE internally.

The inconsistency is most likely to occur at the boundaries, including external boundaries and artificial internal boundaries. For the external boundaries, we found that at those with Dirichlet BCs, $M^{-1}A^*M$ will never be consistent with a PDE. If the boundary conditions are of Neumann or mixed type, $M^{-1}A^*M$ is consistent with the adjoint PDE when there is no advection (or convection) term in the primal PDE. Otherwise, the inconsistency will occur at or near the boundaries. Artificial internal boundaries are usually generated from an adaptive grid.

With a fixed grid and a linear discretization method, a consistent approximation to the objective functional and a consistent approximation to the adjoint PDE can be produced outside the boundary layer. This is one of the bases for the ADDA method, which will be described in the next section. For a nonlinear discretization (e.g., the upwinding scheme), the consistency can be obtained in a weak sense.

There are three kinds of adaptive methods for time-dependent PDEs: r-refinement or moving grid method, p-refinement or adaptive order method and h-refinement or local grid refinement method. Internal boundaries can be generated from p-refinement and h-refinement. We have observed that AD for p-refinement and h-refinement is not consistent with the adjoint PDE at and near the interface between the low and high order methods (for p-refinement) and between the fine and coarse grids (for h-refinement). This is because the interface acts as an internal boundary and the extra terms generated from integration by parts there cannot be canceled numerically as they are analytically.

4. Adjoint sensitivity method for adaptive mesh refinement

There are two types of adaptive grids for time-dependent PDEs. The first is a fixed adaptive grid, which is adapted initially and then fixed during the time integration. The second type of adaptive grid also varies with time. When a new grid is generated, interpolation must be used to obtain the solution on the new grid. There is less interpolation error if the most interesting regions are covered by both the old and new grids. This is why the grid must be adapted frequently during the time integration. We focus on the second type of adaptive grid in the following. We also assume that the method of lines approach is used for solution of the forward PDE problem.

Since the adjoint variables must be interpreted differently in the AD and the DA methods, the two methods cannot easily be mixed, i.e., we cannot solve for the adjoint variables in AD and use them in the sensitivity evaluation of DA, and vice versa. This is true even when the transformation $\tilde{V} = M^{-1}V$ is applied to make them more similar.

The AD method is easy to implement. Given a spatial discretization for a PDE problem and sensitivity parameters, automatic differentiation tools can be used to generate a spatial discretization for the adjoint problem. However, the original AD method can be used only for a fixed grid. This is because the AD system may not be consistent with the PDE. Hence the interpolation in the adaptive grid method makes no sense for the AD method. The initial conditions of the original AD method also contain the grid-spacing information, which will be lost when the grid is refined later. Fig. 1 shows a plot of the adjoint variable (after the M transformation for AD) on the interface of the fine and coarse grids. It is obvious that the adjoint variable in the AD method exhibits a high frequency oscillation.

The DA method can be used for adaptive grids. Since the discretization is straightforward from a PDE, the DA system is usually simpler and requires less memory than the corresponding AD system. However, it is difficult to obtain the DA solver automatically. To our knowledge, no automatic tool can do that. This is partly because the boundary conditions and the boundary term BT in the DA method are problem dependent, and it is not possible to generate them automatically from a solver for the state variables. It would be troublesome for an inexperienced user to provide the boundary conditions and boundary term BT. Moreover, for inadmissible objective functionals, the DA method is problematic as discussed earlier.

To overcome the difficulties of the AD and DA methods, we propose a new method which combines the advantages of the two methods and can also be used in combination with AMR. We call this method ADDA, where AD takes care of the boundary discretization and DA takes care of the internal discretization and mesh adaptivity. We developed the ADDA method based on an observation that the discretization from the AD method is consistent with the adjoint PDE (hence it can be replaced with the discretization of the DA method) at the internal points if the mesh and the (linear) discretization are uniform everywhere except at the boundaries. The basic idea of the ADDA method is illustrated in Fig. 2.

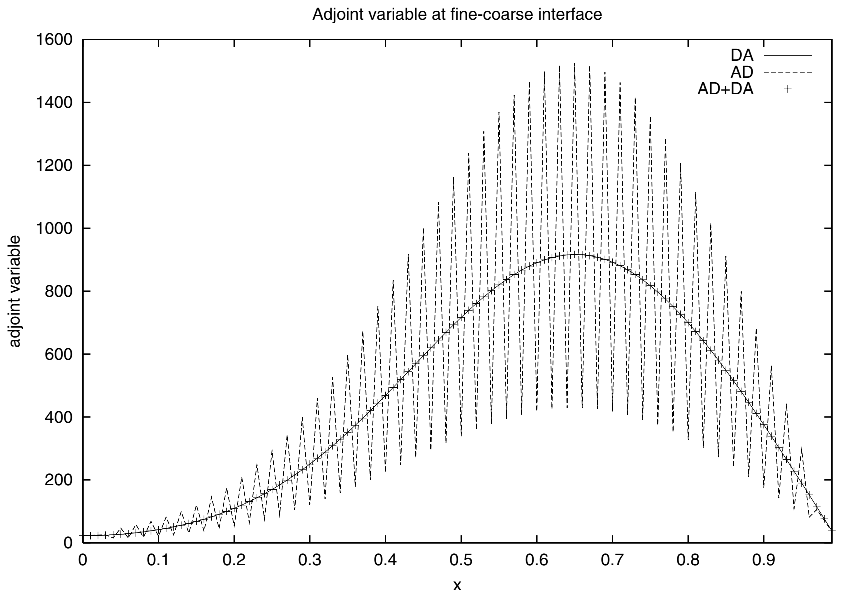

Fig. 1. Adjoint variable for different adjoint methods on an adaptive grid.

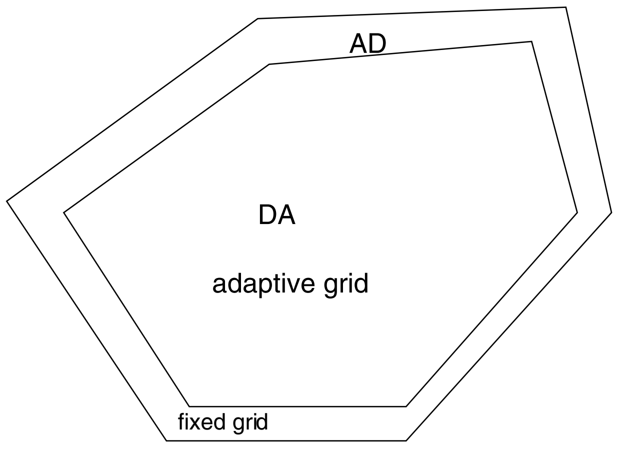

Fig. 2. Diagram of the ADDA method.

The results of the ADDA method should be equal or close to the results of the AD method on a nonadaptive fine grid. Given a reference nonadaptive fine grid, we first split the whole domain into two zones: boundary buffer zone and internal zone (see Fig. 2). The boundary buffer zone consists of the boundary points and points that use the boundary points in their discretization. The remainder of the points belong to the internal zone. Since the discretization of the AD method may not be consistent with the adjoint PDE at or near the boundaries, the buffer zone is fixed and never adapted during the entire time integration. In the internal zone, the discretization from the AD method can be replaced with the discretization from the DA method if we assume that the discretization from the AD method is consistent with the adjoint PDE. As discussed in Section 3.2, this assumption is not always true for a general discretization and grid. However, if the mesh and discretization of the forward problem are uniform in the internal zone, the adjoint of the discretization is indeed consistent with the adjoint PDE.

After the discretization in the internal zone has been replaced by that from the DA method, the mesh can be adapted to achieve efficiency without loss of accuracy. The adaptive mesh refinement in the internal zone is invisible to the AD method, which expects that the discretizations for the adjoint system are generated by the AD method on a nonadaptive fine grid. Instead the discretization is accomplished efficiently by the DA method on an adaptive grid. The internal zone looks like a black box to the AD method.

Since the sensitivity calculation is based on the AD method, the initial conditions for the adjoint system must be generated by the AD method. However, the initial conditions generated by the AD method may involve the grid spacing information due to the spatial discretization and integration in the objective functional evaluation. The variable transformation $\tilde{V} = M^{-1}V$ can eliminate the grid spacing information related to the integration scheme in the objective function evaluation. However, it cannot eliminate the grid spacing information related to the integrand function.

Strictly speaking, the values of the adjoint variables are different on different grids. That is why the sensitivity calculation by the AD method must be performed on a fixed mesh. The initial given mesh, which is the last mesh generated at $t = T$ in the forward adaptive method, may not be the same as the reference nonadaptive fine mesh we seek. Therefore, we must calculate the initial conditions for the ADDA method on the reference mesh first and then project them onto the initial given mesh by interpolation.

The overall algorithm of the ADDA method is as follows: First we obtain the initial conditions for the adjoint system by the AD method on a virtual nonadaptive fine grid. Then we transform and project them to the adaptive grid with a fixed boundary buffer zone. We assume that the discretization has been chosen so that AD is consistent with the adjoint PDE internally. Then the spatial discretization in the boundary buffer zone is generated by the AD method via automatic differentiation, and the discretization in the internal zone is defined by discretization of the adjoint PDE. Finally, an ODE or DAE time solver is used to advance the solution to the next time step. After the adjoint variables have been computed, the sensitivity evaluations of the AD method are used to calculate the sensitivities.

5. Numerical examples

In this section we present two examples to illustrate the difference between the AD, DA, and ADDA methods. For simplicity, we use only *h*-refinement and two refinement levels. We use the structured adaptive mesh refinement method (SAMR), which is described in [18]. Our SAMR algorithm in [18] also has a feature to calculate the sensitivity by the forward sensitivity method. In both examples, we first calculated the sensitivity by the forward method on a fine uniform grid, and then compared it with the results of the adjoint method. Central differencing is used in all of our spatial discretizations. Our ODE/DAE solver DASPKADJOINT [17] is used to do time integration and sensitivity calculation.

The first example is of reaction-diffusion type and is described in [23]. The PDE is given by

$$u_t = \Delta u + D(2-u)\exp(-d/u) \quad \text{on the domain } \Omega = (0,1)\times(0,1),$$
$$u|_{t=0} = 1, \quad \frac{\partial u}{\partial n} = 0 \quad \text{at } x=0,\ y=0, \text{ and } u=1 \quad \text{at } x=1,\ y=1, \tag{20}$$

where Δ is the Laplacian operator and $D = Re^d/d,\ R=5,\ d=20$. R is chosen as the sensitivity parameter, and the objective functional at the final time is defined by

$$\int_{0.8}^{1} \mathrm{d}x \int_{0.8}^{1} (u_x^2 + u_y^2)\,\mathrm{d}y.$$

This objective functional is inadmissible for the DA method, because the initial condition for the DA method

$$v_0 = \begin{cases} 2u_x\frac{\partial}{\partial x} + 2u_y\frac{\partial}{\partial y} & \text{if } (x,y) \in [0.8,1.0]\times[0.8,1.0], \\ 0 & \text{otherwise} \end{cases}$$

is not a regular function but a linear functional.

Table 1 shows the sensitivity calculated by the different methods. We can see that both the AD and DA methods are not very accurate on the adaptive grid. It is surprising that the DA method has a large error, which we believe is due to the inadmissible objective functional. However, if we use the initial conditions of ADDA, which are generated on a uniform fine grid, in the DA method, the ADDA and DA methods give almost the same results, which is expected because the boundary term $BT = 0$ and the boundary conditions generated by AD are the same as those given by DA.

The second example is the 2D Burgers' equation

$$u_t = a\nabla^2 u - uu_x - uu_y, \quad 0.6 \leqslant t \leqslant 1.3.$$

The initial and boundary conditions are chosen such that the exact solution is

$$u(x,y,t) = \frac{1}{1 + e^{(x+y-t)/(2a)}}.$$

The sensitivity parameter is chosen to be a, with initial value $a = 0.008$, and the objective functional at $t = 1.3$ is

$$g(x,y) = \int_0^1 \int_0^1 u(x,y,1.3)^2\,\mathrm{d}x\,\mathrm{d}y.$$

In order to test the adjoint method for different boundary conditions, we set Dirichlet boundary conditions on the right and top boundaries and Neumann boundary conditions on the left and bottom boundaries. We first solved it by the forward sensitivity method, and then compared the results with those of the adjoint methods. The sensitivity equation is

Table 1
Comparison of different sensitivity methods

Method	Fixed		Adapted with two refinement levels			
	Forward	AD	AD	DA	DA2[a]	ADDA
Base grid points	100×100	100×100	50×50	50×50	50×50	50×50
Sensitivity	0.01478	0.01476	0.01440	0.01693	0.01470	0.01470

[a] DA2 is DA with the initial conditions of ADDA.

$$s_t = a\nabla^2 s - (us)_x - (us)_y.$$

The adjoint equation is

$$-v_t = a\nabla^2 v + uv_x + uv_y,$$

where u is the state variable. The AD method is generated automatically by TAMC [12] from the forward solver. For the DA method, the Dirichlet boundary conditions become homogeneous $v = 0$, and the Neumann boundary conditions become

$$v_{\vec{n}} + uv = 0.$$

For this problem, the boundary term BT is

$$BT = \int_{0.6}^{1.3} \mathrm{d}t \left(\int_0^1 v_y u_p|_{y=1}\, \mathrm{d}x + \int_0^1 v_x u_p|_{x=1}\, \mathrm{d}y + \int_0^1 v u_{yp}|_{y=0}\, \mathrm{d}x + \int_0^1 v u_{xp}|_{x=0}\, \mathrm{d}y \right),$$

which is no longer zero. Fig. 3 shows the refinement at the beginning and final times. Table 2 shows the sensitivities calculated by different methods.

The sensitivity evaluation of the DA method includes the BT term, $BT = 0.2032$, which is not present for the sensitivity evaluation of the AD and ADDA methods. Fig. 4 shows the values of the adjoint variable at the line $y = 0.04$. We can see that the results of ADDA are quite different from that of DA due to the different boundary conditions, although they use the same discretization internally.

We should point out that we chose these two examples in part because the results could be compared to those of the forward sensitivity methods. In practice, the adjoint method is advantageous if there are many

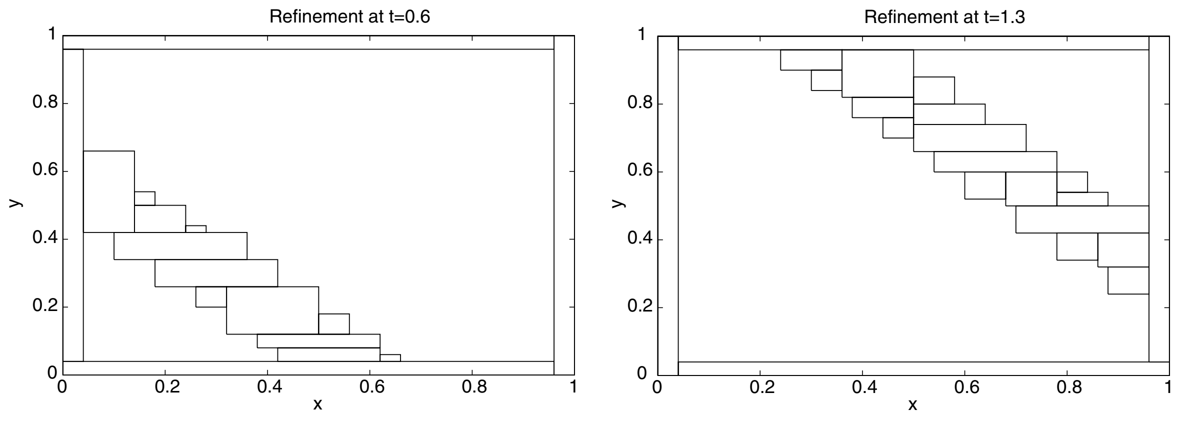

Fig. 3. Grid refinement at different times for Burgers' equations.

Table 2
Comparison of different sensitivity methods

Method	Fixed		Adapted with two refinement levels		
	Forward	AD (fixed)	AD (adaptive)	DA	ADDA
Grid points	100×100	100×100	50×50	50×50	50×50
Sensitivity	1.6036	1.5984	1.4913	1.5797	1.5952

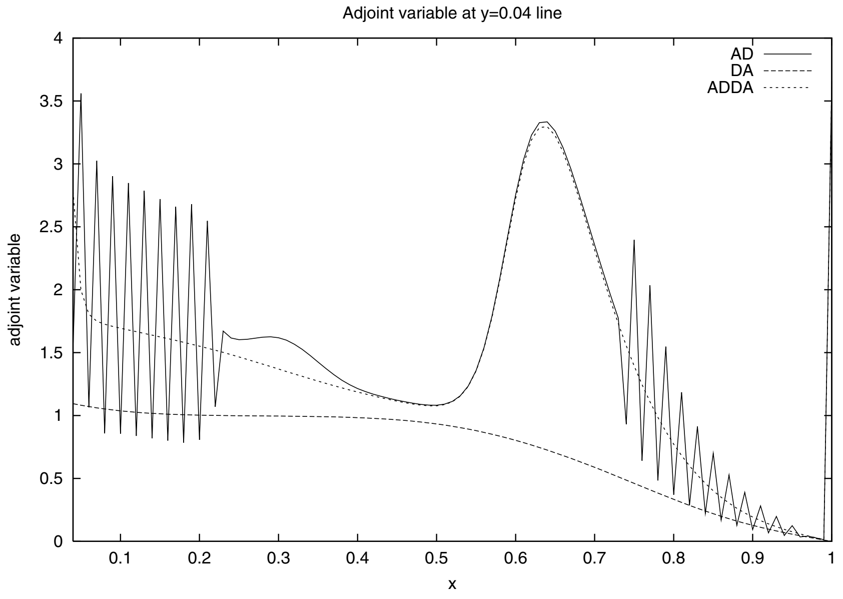

Fig. 4. Adjoint variable for different adjoint methods on an adaptive grid.

parameters and few objective functions. The ADDA approach is insensitive to the thickness of the AD subdomain as long as it covers the stencil of the numerical scheme. For example, for central-differencing on a domain with ghost boundaries, a thickness of only two cells is sufficient.

Our ADDA approach is not restricted to the method of lines with finite-difference method. It can be applied to any PDE discretization (e.g., finite element and full discretization). The only difference for different approaches is in the sensitivity calculation.

6. Conclusion

We have proposed a new adjoint sensitivity analysis approach (ADDA) that combines the discrete approach (AD), where the adjoint system is derived by automatic differentiation of the discretized PDE, with the continuous approach (DA), where the adjoint system is derived from the discretization of the adjoint PDE, for time-dependent PDEs. The ADDA method removes the obstacles of applying AD in the context of AMR and, in contrast to DA, requires for its use only a minimal amount of expertise on the formulation of the adjoint PDE and its boundary conditions.

The DA approach can be used with AMR. Although experts have often been successful in deriving boundary conditions for the adjoint of a given PDE system, there is no systematic method for deriving the boundary conditions for the adjoint of a general PDE system. Moreover, the use of the DA method is complicated by the inadmissibility problem for some objective functionals. In contrast, generation of boundary conditions for the adjoint problem is not an issue for the AD approach, and there is no problem with inadmissibility. On the other hand, the AD approach cannot be used with AMR because it is usually inconsistent with the adjoint PDE (or with any PDE) at the internal grid boundaries.

In the ADDA method, the discretization on (or near) the boundary is generated by AD, which circumvents the difficulties of deriving the boundary conditions for the adjoint PDE system. The discretization at the internal points comes from a DA approach, which is naturally consistent with the adjoint PDE.

During the time integration, the adaptive method is applied only at the internal points. The interface between the AD and DA approaches becomes seamless if the discretization in the AD approach is consistent with the adjoint PDE internally on a nonadaptive grid.

References

[1] W.K. Anderson, V. Venkatakrishnan, Aerodynamic design optimization on unstructured grid with a continuous adjoint formulation, in: 35'th Aerospace Science Meeting & Exhibit, 1997, AIAA 97-0643.

[2] E. Arian, M. Salas, Admitting the inadmissible: adjoint formulation for incomplete cost functionals in aerodynamic optimization, ICASE Report No. 97-69, 1997.

[3] M.J. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comput. Phys. 53 (1984) 484–512.

[4] T.R. Bewley, Flow control: new challenges for a new renaissance, Prog. Aerospace Sci. 37 (2001) 21–58.

[5] J. Borggaard, J. Burns, A PDE sensitivity equation method for optimal aerodynamic design, J. Comput. Phys. 136 (1997) 366–384.

[6] C. Bischof, A. Carle, G. Corliss, A. Griewank, P. Hovland, ADIFOR – generating derivative codes from Fortran programs, Sci. Program. 1 (1992) 11–29.

[7] Y. Cao, S. Li, L. Petzold, R. Serban, Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution, SIAM J. Sci. Comput. 24 (2003) 1076–1089.

[8] P. Colella, Dan Martin, and other staffs in applied numerical analysis group, Chombo1.2, released in 2002, Lawrence Berkley National Laboratory. Available from <http://seesar.lbl.gov/anag/chombo/index.html>.

[9] P. Eberhard, C. Bischof, Automatic differentiation of numerical integration algorithms, Argonne National Lab., ANL/MCS-P621-1196, 1996 (preprint).

[10] B. Fryxell, K. Olson, P. Ricker, F.X. Timmes, M. Zingale, D.Q. Lamb, P. MacNeice, R. Rosner, J.W. Truran, H. Tufo, Flash: an adaptive mesh hydrodynamics code fro modeling astrophysical thermonuclear flashes, ApJS 131 (2000) 273.

[11] Y. Jarny, M.N. Ozisik, J.P. Bardon, A general optimization method using adjoint equation for solving multidimensional inverse heat conduction, Int. J. Heat Mass Transfer 34 (1991) 2911–2919.

[12] R. Giering, T. Kaminski, Recipes for adjoint code construction, ACM Trans. Math. Software 24 (1998) 437–474.

[13] M.B. Giles, N.A. Pierce, Adjoint equations in CFD: duality, boundary conditions and solution behaviour, AIAA Paper 97-1850, 1997.

[14] P. Ghattas, J.-H. Bark, Optimal control of two and three-dimensional incompressible Navier–Stokes flows, J. Comput. Phys. 136 (1997) 231–244.

[15] A. Jameson, Aerodynamic design via control theory, J. Sci. Computing 3 (1988) 233–260.

[16] S.K. Nadarajah, A. Jameson, A comparison of the continuous and discrete adjoint approach to automatic aerodynamic optimization, AIAA paper 00-0067, 2000.

[17] S. Li, L. Petzold, User's guide for the DASPK adjoint solver, Technical Report 01-39, University of California, Santa Barbara, 2001.

[18] S. Li, L. Petzold, J. Hyman, Solution adapted nested grid refinement and sensitivity analysis for parabolic partial differential equations, in: ecture Notes in Computational Science and Engineering, Springer, Berlin, 2001.

[19] W.S. Martinson, P.I. Barton, A differentiation index for partial differential-algebraic equations, SIAM J. Sci. Comput. 21 (2000) 2295–2315.

[20] S.L. Campbell, C.W. Gear, ODE/DAE integrators and MOL problems, Z. Angew. Math. Mech. 76S (1996) 251–254.

[21] R. Serban, S. Li, L. Petzold, Adaptive algorithms for optimal control of time-dependent partial differential-algebraic equation systems, Int. J. Numer. Meth. Eng. 57 (2003) 1457–1469.

[22] A. Sei, W.W. Symes, A note on consistency and adjointness for numerical schemes, Technical Report TR95-04, Department of Computational and Applied Mathematics, Rice University, 1995.

[23] P.A. Zegeling, Moving finite-element solution of time-dependent partial differential equations in two space dimensions, Department of Numerical Mathematics, CWI, Amsterdam, Report NM-R9206, 1992.