QC851
C47
no.22
ATSL

ISSN No. 0737-5352-22

DATA ASSIMILATION AND THE ADJOINT METHOD: An Overview

LIBRARIES
MAY 24 1991
COLORADO STATE UNIVERSITY

Gerald D. Taylor
Department of Mathematics
and
Center for Geosciences

Research Supported by U.S. Army Research Office
under Grant No. DAAL03-86-K-0175

CIRA Cooperative Institute for Research in the Atmosphere

Colorado State University

DATA ASSIMILATION AND THE ADJOINT METHOD:
An Overview

by

Gerald D. Taylor

Department of Mathematics
and
Center for Geosciences
Colorado State University
Fort Collins, Colorado 80523

This research was supported by ARO
under Grant No. DAAL03-86-K-0175 through the
U.S. Army Center for Geosciences and CIRA

May 1991

QC
851
.C47
no.22
ATSL

Abstract

In this note a general description of the application of the adjoint method for four dimensional data assimilation is given. The main focus of this discussion is that this method when applied to a discrete model is simply an application of the chair rule of multivariate calculus to compute the gradient of a real valued function of several variables. Several different time differencing schemes are considered, as well as, two examples of discrete models.

LIBRARIES
COLORADO STATE UNIVERSITY
Fort Collins, Colorado 80523

DATA ASSIMILATION AND THE ADJOINT METHOD

G. D. Taylor

1.General Mathematical Theory

Data assimilation, the effective integration of observations into predictive dynamical equations has been a major topic of investigation for some 30 years in meteorology. Using optimization theory, Le Dimet and Talagrand (1986) and Talagrand and Courtier (1987a, 1987b, 1990) have proposed a new approach for this problem called the adjoint method. Specifically, the following variational approach for the assimilation of observations is proposed: Define a real valued function measuring the "distance" between the model solution corresponding to a given initialization and the available observations. Then, using the adjoint method, the gradient of this distance function with respect to the initial condition can be calculated. Using a gradient based minimization algorithm (e.g. Buckley (1985)), that initialization for which the distance is a minimum is calculated via an iterative process. Thus, using this initialization the model will predict values that most closely fit the known observational data. The current wisdom is that this procedure is most effective when applied to the actual predictive equations used in the dynamical model. Since these codes are (usually) based upon some sort of discretization, the above minimization problem becomes the minimization of a (complicated) function of several variables and the adjoint method reduces to an application of the chain rule for calculating the derivative (gradient) of a composition of several functions.

As an extremely simple example of the setting in which one might wish to apply this approach consider the following continuous model: Given $u_0(z,0)$, find $u(z,t)$, for $0 \le z \le L$ and $t > 0$, such that

$$\begin{aligned} &\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial z} = 0, \quad 0 \le z \le L, \quad t > 0, \\ &u(0,t) = u(L,t), \quad t > 0, \\ &u(z,0) = u_0(z,0), \quad 0 \le z \le L. \end{aligned} \tag{1}$$

Discretizing in space, by setting $z_i = ih, h = L/N$, $u_i(t) = u(z_i,t)$ for $i = 0,\cdots,N$ and applying centered differences, gives the initial value system of N ordinary differential equations:

$$\begin{aligned} &\frac{du_i(t)}{dt} = -u_i(t)\left(\frac{u_{i+1}(t) - u_{i-1}(t)}{2h}\right), \quad i = 0,\cdots,N-1 \\ &u_i(0) = u_0(z_i,0), \quad i = 0,\cdots,N-1 \end{aligned}$$

where $u_N(t) = u_0(t)$ and $u_{-1}(t) = u_{N-1}(t)$ for all $t \ge 0$.

Writing

$$\mathbf{x}(t) = \left(u_0(t),\cdots,u_{N-1}(t)\right)^T \text{ and } \mathbf{F}(\mathbf{x}) = \left(f_0(\mathbf{x}),\cdots,f_{N-1}(\mathbf{x})\right)^T, \tag{2}$$

where

$$f_j(\mathbf{x}) = -u_j\left(\frac{u_{j+1} - u_{j-1}}{2h}\right), \quad j = 0,\cdots,N-1,$$

this system can be rewritten in vector form as

$$\begin{aligned} &\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}(\mathbf{x}(t)), \\ &\mathbf{x}(0) = \mathbf{x}_0 = \left(u_0(0),\cdots,u_{N-1}(0)\right)^T \end{aligned} \tag{3}$$

Now, suppose that observations exist for certain values of $u(z_i,t)$ at some distinct times t_j. Then, one seeks to determine an initialization,

$$\mathbf{x}_0 = \left(u(z_0,0),\cdots,u(z_{N-1})\right)^T,$$

for which the model solution of (1) corresponding to this initialization is the "closest" to these observations from the class of all possible model solutions (i.e., for all possible initializations).

In order to describe the adjoint method for accomplishing this we return to the general setting. Thus, let a dynamical model in the form of a system of ordinary equations be given:

$$\frac{d\mathbf{x}(t)}{dt} = \mathbf{F}(\mathbf{x}(t)), \tag{4}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ for $t \geq 0$ and $\mathbf{F} : \mathbb{R}^n \to \mathbb{R}^n$ with $\mathbf{F}(\mathbf{z}) = (f_1(\mathbf{z}), \cdots, f_n(\mathbf{z}))^T$, for $\mathbf{z} \in \mathbb{R}^n$ and each $f_j : \mathbb{R}^n \to \mathbb{R}$. Further assume that observational data at a set of distinct times $0 \leq t_1 < t_2 < \cdots < t_p$ have been converted to (some or all) grid point values (vector components of $\mathbf{x}(t_j)$) that we would like our model solution to assume at these times. Next, select a time advecting scheme that predicts model values $\mathbf{x}_j$ given an initialization $\mathbf{x}(0) = \mathbf{x}_0$ for $j = 0, \cdots, r$ such that $\mathbf{x}_{j_l} = \mathbf{x}(t_l)$ for $l = 1, \cdots, p$ with $j_p = r$. Note that this indexing allows for several types of advecting steps between the times at which the observations are available.

For a given initialization $\mathbf{x}_0$ define target values $\mathbf{x}_j^{obs}$ corresponding to the set of advected values $\mathbf{x}_j$ predicted by $\mathbf{x}_0$ by requiring that $\mathbf{x}_{j_l}^{obs}$ have the components at times t_l determined by the observational data and have all remaining components identical with the components of $\mathbf{x}_j$. Finally, define a real valued cost function $J(\mathbf{x}_0)$ by

$$J(\mathbf{x}_0) = \sum_{j=0}^{r} (\mathbf{x}_j - \mathbf{x}_j^{obs})^T (\mathbf{x}_j - \mathbf{x}_j^{obs}). \tag{5}$$

Because of the manner in which $\mathbf{x}_j^{obs}$ has been defined, $J(\mathbf{x}_0)$ is precisely the sum of the squares of the differences of the model values and observational values at all components where the observational data can be represented. (Note that a weighted cost function could have been used here; i.e., $J(\mathbf{x}_0) = \sum_{j=0}^{r} (\mathbf{x}_j - \mathbf{x}_j^{obs})^T W_j (\mathbf{x}_j - \mathbf{x}_j^{obs})$ where each W_j is a strictly positive definite symmetric $n \times n$ weighting matrix.) In this setting the problem of data assimilation *is taken* to mean

$$solve \quad min\{J(\mathbf{x}_0) : \mathbf{x}_0 \in \mathbb{R}^n\}. \tag{6}$$

This can be done using a black box gradient based (iterative) minimization routine when one is using the adjoint method since this method calculates $\nabla_{\mathbf{x}_0} J(\mathbf{x}_0)$, the gradient of $J(\mathbf{x}_0)$ at $\mathbf{x}_0$, which is required at each iteration step of the minimization scheme.

We now turn to describing the adjoint method for this general setting. Since the form of the adjoint method depends on the (time) advection scheme used, we must give a more structured description of the advection scheme. Initially, we shall assume that the advection scheme is a one step scheme. That is, that the advected vector $\mathbf{x}_j$ is obtained from the previous advected vector $\mathbf{x}_{j-1}$. Thus, we shall assume that either

$$\mathbf{x}_j = \mathbf{F}_j(\mathbf{x}_{j-1}) \tag{7}$$

(an explicit step) or $\mathbf{x}_j$ is the result of one Newton iteration applied to

$$\mathbf{F}_j(\mathbf{x}_j) = \mathbf{G}_j(\mathbf{x}_{j-1}) \tag{8}$$

initialized with $\mathbf{x}_{j-1}$ (an implicit step). Here the functions $\mathbf{F}_j$ and $\mathbf{G}_j$ will depend upon the $\mathbf{F}$ of (4). For the case of an implicit step, $\mathbf{x}_j$ is actually the solution of the linear system

$$\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1})(\mathbf{x}_j - \mathbf{x}_{j-1}) = \mathbf{G}_j(\mathbf{x}_{j-1}) - \mathbf{F}_j(\mathbf{x}_{j-1}). \tag{9}$$

Note that the quantity $\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1})$ is simply a $n \times n$ matrix (the Jacobian of $\mathbf{F}_j$ evaluated at $\mathbf{x}_{j-1}$, see (A.1) in appendix A for details). Observe that, as described, an implicit step can be used to give a new estimate at the same time position as the previous step or an estimate at the next time step. This is one reason for the somewhat complicated indexing of the advection vectors relative to the observation times given earlier.

Returning to the cost function (5), from the derivative rule for the dot product $(\mathbf{F}(\mathbf{x}))^T(\mathbf{G}(\mathbf{x}))$ of two functions mapping $\mathbb{R}^n$ into $\mathbb{R}^n$ one has that

$$\mathbf{D}_{\mathbf{x}}\Big\{(\mathbf{F}(\mathbf{x}))^T(\mathbf{G}(\mathbf{x}))\Big\} = (\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}))^T(\mathbf{G}(\mathbf{x})) + (\mathbf{F}(\mathbf{x}))^T(\mathbf{D}_{\mathbf{x}}\mathbf{G}(\mathbf{x}))$$

so that

$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = 2\sum_{j=0}^{r}(\mathbf{x}_j - \mathbf{x}_j^{obs})^T \mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0) \tag{10}$$

where $\mathbf{D}_{\mathbf{x}_0} J(\mathbf{x}_0) = \nabla_{\mathbf{x}_0} J(\mathbf{x}_0)$ since $J(\mathbf{x}_0)$ is a real valued function of $\mathbf{x}_0$ and each $\mathbf{x}_j$ is a function of $\mathbf{x}_0$ via the advective iteration. Thus, it is necessary to be able to compute the $n \times n$ matrices $\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0)$ in order to be able to calculate the gradient of the cost function at a given initialization. Note that, by the chain rule

$$\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0) = [\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})][\mathbf{D}_{\mathbf{x}_{j-2}}\mathbf{x}_{j-1}(\mathbf{x}_{j-2})]\cdots[\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_1(\mathbf{x}_0)], \tag{11}$$

where we have used the fact that $\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_0(\mathbf{x}_0) = I$, the $n \times n$ identity matrix. Since for finite difference discretizations of the original continuous problem these matrices are usually banded with only a few non-zero diagonals, it should be preferable not to actually use matrix multiplies in any specific coding of this procedure. Rather, the product $\mathbf{y}^T[\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})]\cdots[\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_1(\mathbf{x}_0)]$ may be more effectively calculated by writing a special routine based on the structure of each $\mathbf{D}_{\mathbf{x}_{i-1}}\mathbf{x}_i(\mathbf{x}_{i-1})$ only storing the non-zero elements of each of these matrices and achieving the desired product of $\mathbf{y}^T$ and these matrices through a sequence of vector dot products (inner products). In the case of spectral discretizations the matrices will most likely be full matrices. However, in this case the overall dimension of the discrete problem should be considerably smaller due to the increased accuracy of spectral methods for problems having smooth solutions. Here matrix operations might be acceptable, although once again, in an actual operational application of this technique one should look for any special structure inherent to the problem that can be exploited such as using FFT's, for example.

Now, write

$$\begin{aligned}\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1}) &= \begin{pmatrix} \frac{\partial f_1^j}{\partial x_1^{j-1}}(\mathbf{x}_{j-1}) & \cdots & \frac{\partial f_1^j}{\partial x_n^{j-1}}(\mathbf{x}_{j-1}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n^j}{\partial x_1^{j-1}}(\mathbf{x}_{j-1}) & \cdots & \frac{\partial f_n^j}{\partial x_n^{j-1}}(\mathbf{x}_{j-1}) \end{pmatrix} \\ &= \begin{pmatrix} a_{11}^j(\mathbf{x}_{j-1}) & \cdots & a_{1n}^j(\mathbf{x}_{j-1}) \\ \vdots & \ddots & \vdots \\ a_{n1}^j(\mathbf{x}_{j-1}) & \cdots & a_{nn}^j(\mathbf{x}_{j-1}) \end{pmatrix}\end{aligned} \tag{12}$$

where $\mathbf{F}_j(\mathbf{x}) = (f_1^j(\mathbf{x}), \cdots, f_n^j(\mathbf{x}))^T$ with the first equality the definition of the derivative and the second equality a notational choice. Next, using (12) define the matrix,

$$\Gamma_j(\mathbf{x}_j, \mathbf{x}_{j-1}) = \begin{pmatrix} \left(d_1 \frac{\partial a_{11}^j}{\partial x_1^{j-1}}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{1n}^j}{\partial x_1^{j-1}}(\mathbf{z})\right) & \cdots & \left(d_1 \frac{\partial a_{11}^j}{\partial x_n^{j-1}}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{1n}^j}{\partial x_n^{j-1}}(\mathbf{z})\right) \\ \vdots & \ddots & \vdots \\ \left(d_1 \frac{\partial a_{n1}^j}{\partial x_1^{j-1}}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{nn}^j}{\partial x_1^{j-1}}(\mathbf{z})\right) & \cdots & \left(d_1 \frac{\partial a_{n1}^j}{\partial x_n^{j-1}}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{nn}^j}{\partial x_n^{j-1}}(\mathbf{z})\right) \end{pmatrix} \tag{13}$$

where $d_k = (x_k^j - x_k^{j-1})$ and $\mathbf{z} = \mathbf{x}_{j-1}$. Using this notation an iterative scheme for calculating $\nabla_{\mathbf{x}_0} J(\mathbf{x}_0)$ can be given. This scheme is essentially Horner's method for efficiently evaluating a polynomial (or as noted in the literature the adjoint method is equivalent to the backwards integration of the tangent equations corresponding to (4) and can be realized as the scheme given here). Thus, to calculate $\nabla_{\mathbf{x}_0} J(\mathbf{x}_0)$ one should proceed as follows:

1. Compute the vector

$$\mathbf{y}_r^T = (\mathbf{x}_r - \mathbf{x}_r^{obs})^T \mathbf{D}_{\mathbf{x}_{r-1}}\mathbf{x}_r(\mathbf{x}_{r-1}). \tag{14}$$

2. For $j = r-1, r-2, \cdots, 1$, compute the vectors

$$\mathbf{y}_j^T = \{(\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \mathbf{y}_{j+1}^T\}\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1}). \tag{15}$$

3. Set

$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = 2\{(\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \mathbf{y}_1^T\}. \tag{16}$$

Note that the adjoint method is calculated relative to a given $\mathbf{x}_0$ and the corresponding advected values $\mathbf{x}_j$ all must be stored for use in the adjoint sweep.

To complete the above scheme we must give formulas for $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$. Now, if $\mathbf{x}_j$ is the result of the explicit step, $\mathbf{x}_j = \mathbf{F}_j(\mathbf{x}_{j-1})$, with $\mathbf{F}_j(\mathbf{x}) = (f_1^j(\mathbf{x}), \cdots, f_n^j(\mathbf{x}))^T$, then

$$\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1}) = \mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1}) =$$

$$\begin{pmatrix} \frac{\partial f_1^j}{\partial x_1^{j-1}}(\mathbf{x}_{j-1}) & \cdots & \frac{\partial f_1^j}{\partial x_n^{j-1}}(\mathbf{x}_{j-1}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n^j}{\partial x_1^{j-1}}(\mathbf{x}_{j-1}) & \cdots & \frac{\partial f_n^j}{\partial x_n^{j-1}}(\mathbf{x}_{j-1}) \end{pmatrix}. \tag{17}$$

Whereas, if $\mathbf{x}_j$ is obtained by an implicit step, then $\mathbf{x}_j$ is the result of one Newton iteration applied to $\mathbf{F}_j(\mathbf{x}_j) = \mathbf{G}_j(\mathbf{x}_{j-1})$ initialized with $\mathbf{x}_{j-1}$, which is given in (9). Differentiating equation (9) with respect to $\mathbf{x}_{j-1}$, gives a linear equation for $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$. Indeed, using the notation introduced above and formula (A.5) of the appendix with $\mathbf{x} = \mathbf{x}_{j-1}$ and $\mathbf{F}(\mathbf{x}_{j-1}) = \mathbf{x}_j - \mathbf{x}_{j-1}$ we have that

$$[\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1})][\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})] = \mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{G}_j(\mathbf{x}_{j-1}) - \Gamma_j(\mathbf{x}_j, \mathbf{x}_{j-1}) \tag{18}$$

where we are using the Jacobian formula (12) (for both $\mathbf{F}_j$ and $\mathbf{G}_j$) and (13) which defines the matrix $\Gamma_j(\mathbf{x}_j, \mathbf{x}_{j-1})$. It must be noted that for an implicit step the matrix $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$ which in this case is no longer the same as $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1})$ will also most likely no longer be sparse and banded in the finite difference setting. However, when $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1})$ is sparse and banded then since $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$ is the solution to n linear systems having this matrix as the common matrix of each of the systems, it should be possible to efficiently solve them. Furthermore, this same matrix was used to solve a linear system in the forward advection step to determine $\mathbf{x}_j$ from $\mathbf{x}_{j-1}$ and hence a factorization of it from that step or some other efficient means of solving these systems may already be available. In addition, since what is desired is the product of $\mathbf{y}^T$ and $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$, one can accumulate the components of the vector $\mathbf{y}^T\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$ by taking the dot product of $\mathbf{y}^T$ and the columns of $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$ as they become available. In particular, if a factorization of $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1})$ is available then this could possibly be done very efficiently, especially if the matrices of the factorization are sparse and banded. Also, note that this column by column approach could be useful in creating a parallel code when such hardware becomes common.

It remains to generalize these results for the case that the advection scheme may be a multistep scheme at some or all steps. To do this we must only generalize the derivative matrices in the above formulas. Thus, suppose that $\mathbf{x}_j$ is the result of a two step explicit scheme, say

$$\begin{aligned} \mathbf{x}_1 &= \mathbf{F}_1(\mathbf{x}_0) \\ \mathbf{x}_j &= \mathbf{F}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}), \quad j \geq 2. \end{aligned} \tag{19}$$

In this case for $j \geq 2$,

$$\begin{aligned} \mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0) = &\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2})\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-1}(\mathbf{x}_0) \\ &+ \mathbf{D}_{\mathbf{x}_{j-2}}\mathbf{F}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2})\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-2}(\mathbf{x}_0) \end{aligned} \tag{20}$$

where we have applied a chain rule for $\mathbf{F}_j : \mathbb{R}^{2n} \to \mathbb{R}^n$, having vector form

$$\mathbf{F}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) = (f_1^j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}), \cdots, f_n^j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}))^T.$$

The derivative matrices in (20) are given by

$$\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1},\mathbf{x}_{j-2}) = \begin{pmatrix} \frac{\partial f_1^j}{\partial x_1^{j-1}}(\mathbf{z}) & \cdots & \frac{\partial f_1^j}{\partial x_n^{j-1}}(\mathbf{z}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n^j}{\partial x_1^{j-1}}(\mathbf{z}) & \cdots & \frac{\partial f_n^j}{\partial x_n^{j-1}}(\mathbf{z}) \end{pmatrix} \tag{21}$$

and

$$\mathbf{D}_{\mathbf{x}_{j-2}}\mathbf{F}_j(\mathbf{x}_{j-1},\mathbf{x}_{j-2}) = \begin{pmatrix} \frac{\partial f_1^j}{\partial x_1^{j-2}}(\mathbf{z}) & \cdots & \frac{\partial f_1^j}{\partial x_n^{j-2}}(\mathbf{z}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n^j}{\partial x_1^{j-2}}(\mathbf{z}) & \cdots & \frac{\partial f_n^j}{\partial x_n^{j-2}}(\mathbf{z}) \end{pmatrix} \tag{22}$$

where $\mathbf{z} = (\mathbf{x}_{j-1},\mathbf{x}_{j-2})$ and $\mathbf{x}_k = (x_1^k,\cdots,x_n^k)^T$. Now it can be shown in this case that $\nabla_{\mathbf{x}_0}J(\mathbf{x}_0)$ can be found iteratively as follows:

1. Set
$$\mathbf{v}_r = \mathbf{0} \quad \text{and} \quad \mathbf{w}_r = \mathbf{x}_r - \mathbf{x}_r^{obs}. \tag{23}$$

2. For $j = r-1, r-2, \cdots, 1$, compute the vectors
$$\begin{aligned} \mathbf{w}_j^T &= \mathbf{w}_{j+1}^T\mathbf{D}_{\mathbf{x}_j}\mathbf{F}_{j+1}(\mathbf{x}_j,\mathbf{x}_{j-1}) + (\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \mathbf{v}_{j+1}^T \\ \mathbf{v}_j &= \mathbf{w}_{j+1}^T\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_{j+1}(\mathbf{x}_j,\mathbf{x}_{j-1}) \end{aligned} \tag{24}$$

3. Set
$$\nabla_{\mathbf{x}_0}J(\mathbf{x}_0) = 2\{\mathbf{w}_1^T\mathbf{D}_{\mathbf{x}_0}\mathbf{F}_1(\mathbf{x}_0) + (\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \mathbf{v}_1^T\}. \tag{25}$$

A systematic procedure for constructing these iterative schemes can be developed by an approach similar to the method of Lagrange multipliers for calculating the minimum of a function subject to constraints. Indeed, if an auxiliary function $H(\mathbf{x}_0,\cdots,\mathbf{x}_r,\lambda_1,\cdots,\lambda_r)$, corresponding to the above example is defined by

$$H(\mathbf{x}_0,\cdots,\mathbf{x}_r,\lambda_1,\cdots,\lambda_r) = \sum_{j=0}^{r}(\mathbf{x}_j - \mathbf{x}_j^{obs})^T(\mathbf{x}_j - \mathbf{x}_j^{obs}) + \sum_{j=1}^{r}\lambda_j^T\left(\mathbf{F}_j(\mathbf{x}_{j-1},\mathbf{x}_{j-2}) - \mathbf{x}_j\right) \tag{26}$$

where we have set $\mathbf{x}_{-1} = 0$ so that $\mathbf{F}_1(\mathbf{x}_0,\mathbf{x}_{-1}) = \mathbf{F}_1(\mathbf{x}_0)$ as in (19). Now, considering all the arguments of H to be functions of $\mathbf{x}_0$ we have by the chain rule that

$$\begin{aligned} \nabla_{\mathbf{x}_0}H(\mathbf{x}_0,\cdots,\mathbf{x}_r,\lambda_1,\cdots,\lambda_r) = \sum_{j=0}^{r}\Big[&2(\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \lambda_{j+1}^T\mathbf{D}_{\mathbf{x}_j}\mathbf{F}_{j+1}(\mathbf{x}_j,\mathbf{x}_{j-1}) \\ &+ \lambda_{j+2}^T\mathbf{D}_{\mathbf{x}_j}\mathbf{F}_{j+2}(\mathbf{x}_{j+1},\mathbf{x}_j) - \lambda_j^T\Big]\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j \\ &+ \sum_{j=1}^{r}\left(\mathbf{F}_j(\mathbf{x}_{j-1},\mathbf{x}_{j-2}) - \mathbf{x}_j\right)^T\mathbf{D}_{\mathbf{x}_0}\lambda_j \end{aligned} \tag{27}$$

where $\lambda_{r+1} = \lambda_{r+2} = \mathbf{0}$. Next, note that if one requires that $\mathbf{x}_j = \mathbf{F}_j(\mathbf{x}_{j-1},\mathbf{x}_{j-2})$, $j \geq 1$ (so that $\mathbf{x}_1 = \mathbf{F}_1(\mathbf{x}_0)$ since we have notational set $\mathbf{x}_{-1} = \mathbf{0}$) then $H(\mathbf{x}_0,\cdots,\mathbf{x}_r,\lambda_1,\cdots,\lambda_r) = J(\mathbf{x}_0)$. In addition, the above formula (27) for $\nabla_{\mathbf{x}_0}H$ in this case reduces to

$$\begin{aligned} \nabla_{\mathbf{x}_0}H(\mathbf{x}_0,\cdots,\mathbf{x}_r,\lambda_1,\cdots,\lambda_r) = &\left[2(\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \lambda_1^T\mathbf{D}_{\mathbf{x}_0}\mathbf{F}_1(\mathbf{x}_0) + \lambda_2^T\mathbf{D}_{\mathbf{x}_0}\mathbf{F}_2(\mathbf{x}_1,\mathbf{x}_0)\right] \\ &+ \sum_{j=1}^{r}\Big[2(\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \lambda_{j+1}^T\mathbf{D}_{\mathbf{x}_j}\mathbf{F}_{j+1}(\mathbf{x}_j,\mathbf{x}_{j-1}) \\ &+ \lambda_{j+2}^T\mathbf{D}_{\mathbf{x}_j}\mathbf{F}_{j+2}(\mathbf{x}_{j+1},\mathbf{x}_j) - \lambda_j^T\Big]\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j \end{aligned} \tag{28}$$

where the parameters, $\lambda_1, \cdots, \lambda_r$, are still free to be chosen in any manner we please. If one now selects them to force all but the first term of the remaining sum in (28) to be zero, then one obtains the recusive formulas

$$\lambda_j^T = 2(\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \lambda_{j+1}^T \mathbf{D}_{\mathbf{x}_j} \mathbf{F}_{j+1}(\mathbf{x}_j, \mathbf{x}_{j-1}) + \lambda_{j+2}^T \mathbf{D}_{\mathbf{x}_j} \mathbf{F}_{j+2}(\mathbf{x}_{j+1}, \mathbf{x}_j), \quad j = r, r-1, \cdots, 1, \tag{29}$$

and

$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = 2(\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \lambda_1^T \mathbf{D}_{\mathbf{x}_0} \mathbf{F}_1(\mathbf{x}_0) + \lambda_2^T \mathbf{D}_{\mathbf{x}_0} \mathbf{F}_2(\mathbf{x}_1, \mathbf{x}_0),$$

where $\lambda_{r+1} = \lambda_{r+2} = \mathbf{0}$. Note that each of these formulas are precisely determined by the condition that $\mathbf{D}_{\mathbf{x}_j} H = \mathbf{0}$ which is one of the determining equations in the method of Lagrange multipliers.

As a last general example, consider a 2-step advection scheme where $\mathbf{x}_j$ for $j \geq 2$ can be either an explicit 2-step as in (19) or the result of one Newton iteration applied to the equation

$$\mathbf{F}_j(\mathbf{x}_j) = \mathbf{G}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) \tag{30}$$

initialized with $\mathbf{x}_{j-1}$. That is, $\mathbf{x}_j$ is the solution of the linear $n \times n$ system

$$\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1})(\mathbf{x}_j - \mathbf{x}_{j-1}) = \mathbf{G}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) - \mathbf{F}_j(\mathbf{x}_{j-1}). \tag{31}$$

Now, the gradient of the cost function $J(\mathbf{x}_0)$ is defined by (10),

$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = \sum_{j=0}^{r} (\mathbf{x}_j - \mathbf{x}_j^{obs})^T \mathbf{D}_{\mathbf{x}_0} \mathbf{x}_j(\mathbf{x}_0),$$

regardless of the advection scheme. Hence, the task of describing the adjoint method for a given advection scheme really reduces to defining recursive formulas for the derivatives $\mathbf{D}_{\mathbf{x}_0} \mathbf{x}_j(\mathbf{x}_0)$ in terms of the derivatives with respect to $\mathbf{x}_0$ of those $\mathbf{x}_k$ for $j \geq k$ on which $\mathbf{x}_j$ directly depends. Now, if $\mathbf{x}_j$ is the result of an explicit step as described in (19) then $\mathbf{D}_{\mathbf{x}_0} \mathbf{x}_j(\mathbf{x}_0)$ is given by (20)-(22). On the other hand, if $\mathbf{x}_j$ is determined by the linear system (31) then $\mathbf{D}_{\mathbf{x}_0}$ can be found by applying the chain rule to this system. Since

$$\mathbf{D}_{\mathbf{x}_0}[\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1})(\mathbf{x}_j - \mathbf{x}_{j-1})](\mathbf{x}_0) = [\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1})(\mathbf{x}_j - \mathbf{x}_{j-1})] \mathbf{D}_{\mathbf{x}_0} \mathbf{x}_{j-1}(\mathbf{x}_0)$$

by the chain rule, and the first derivative on the right hand side of this expression is given by (A.5), it follows that $\mathbf{D}_{\mathbf{x}_0} \mathbf{x}_j(\mathbf{x}_0)$ satisfies

$$\begin{aligned} \mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1}) \mathbf{D}_{\mathbf{x}_0} \mathbf{x}_j(\mathbf{x}_0) = {} & \mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{G}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) \mathbf{D}_{\mathbf{x}_0} \mathbf{x}_{j-1}(\mathbf{x}_0) \\ & + \mathbf{D}_{\mathbf{x}_{j-2}} \mathbf{G}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) \mathbf{D}_{\mathbf{x}_0} \mathbf{x}_{j-2}(\mathbf{x}_0) - \Gamma_j(\mathbf{x}_j, \mathbf{x}_{j-1}) \mathbf{D}_{\mathbf{x}_0} \mathbf{x}_{j-1}(\mathbf{x}_0), \end{aligned} \tag{32}$$

where $\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{G}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2})$ and$\mathbf{D}_{\mathbf{x}_{j-2}} \mathbf{G}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2})$ are defined in(21) and (22). To continue the development of this example one would now use these formulas to attempt to build recursive formulas for the gradient as done in the explicit 2-step discussion. However, at this juncture we wish to actually construct some adjoint methods for some specific advection schemes and also illustrate the precise form of the Jacobian matrices for two finite difference settings.

2. Some Specific Examples

In this section we give explicit formulas for the adjoint method applied to a system of ordinary differential equations corresponding to some standard advection schemes. We will also illustrate the form of some Jacobian matrices in two settings. Thus, we assume that the continuous model has been discretized in space resulting in a time dependent problem of the form of (4):

$$\frac{d\mathbf{x}}{dt} = \mathbf{F}(\mathbf{x}), \quad \mathbf{x}(0) = \mathbf{x}_0$$

with $\mathbf{x}$ an n vector of time dependent model values. We further assume that this system is to be solved at discrete times via a specific time advection scheme.

For our first example we shall assume that the time advection scheme is the following Runge-Kutta method of order 2:

$$\mathbf{x}_j = \mathbf{x}_{j-1} + \frac{h}{2}\big[\mathbf{F}(\mathbf{x}_{j-1}) + \mathbf{F}\big(\mathbf{x}_{j-1} + h\mathbf{F}(\mathbf{x}_{j-1})\big)\big], \tag{33}$$

for $j = 1, 2, \cdots$, where h denotes the time step increment and $\mathbf{x}_0$ is a given initialization. Note that we have tacitly assumed that $\mathbf{F}$ of (4) does depend explicitly on t and that equal time steps are being used. Here $\mathbf{x}_j$is an approximation for $\mathbf{x}(t_j), t_j = jh$. In a more general problem where $\mathbf{F}$ is of the form $\mathbf{F}(t, \mathbf{x}(t))$ with $\mathbf{F} : \mathbb{R}^{n+1} \to \mathbb{R}^n$ and unequal time steps were taken the equations (33) would be replaced by

$$\mathbf{x}_j = \mathbf{x}_{j-1} + \frac{h_j}{2}\Big[\mathbf{F}(t_{j-1}, \mathbf{x}_{j-1}) + \mathbf{F}\big(t_j, \mathbf{x}_{j-1} + h_j + \mathbf{F}(t_{j-1}, \mathbf{x}_{j-1})\big)\Big]$$

where $t_0 = 0$ and $t_j = t_{j-1} + h_j$ for $j \geq 1$. (Of course, the use an unequal time step Runge-Kutta would be highly unusual.) Returning to (33), we must calculate $\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0)$ in terms of $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_{j-1})$ and $\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-1}(\mathbf{x}_0)$. This is easily done using the chain rule. Indeed, writing $\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{y})$ for the Jacobian of $\mathbf{F}$ evaluated at $\mathbf{y} \in \mathbb{R}^n$ we have that

$$\begin{aligned}\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0) &= \mathbf{D}_{\mathbf{x}_{j-1}}\Big\{\mathbf{x}_{j-1} + \frac{h}{2}\big[\mathbf{F}(\mathbf{x}_{j-1}) + \mathbf{F}\big(\mathbf{x}_{j-1} + h\mathbf{F}(\mathbf{x}_{j-1})\big)\big]\Big\}\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-1}(\mathbf{x}_0) \\ &= \Big[I + \frac{h}{2}\big[\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1}) + \mathbf{D}_{\mathbf{x}}\mathbf{F}\big(\mathbf{x}_{j-1} + h\mathbf{F}(\mathbf{x}_{j-1})\big)\big\{I + \mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1})\big\}\big]\Big]\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-1}(\mathbf{x}_0).\end{aligned} \tag{34}$$

Hence

$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = 2\sum_{j=0}^{r}(\mathbf{x}_j - \mathbf{x}_j^{obs})^T \mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0)$$

can be obtained iteratively as follows:

1. Set
$$\begin{aligned}\mathbf{w}_r &= (\mathbf{x}_r - \mathbf{x}_r^{obs}), \\ \mathbf{y}_r^T &= \mathbf{w}^T\Big[I + \frac{h}{2}\big[\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{r-1}) + \mathbf{D}_{\mathbf{x}}\mathbf{F}\big(\mathbf{x}_{r-1} + h\mathbf{F}(\mathbf{x}_{r-1})\big)\big\{I + \mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{r-1})\big\}\big]\Big].\end{aligned}$$
2. For $j = r-1, r-2, \cdots, 1$, set
$$\begin{aligned}\mathbf{w}_j &= (\mathbf{x}_j - \mathbf{x}_j^{obs}) + \mathbf{y}_{j+1}, \\ \mathbf{y}_j^T &= \mathbf{w}^T\Big[I + \frac{h}{2}\big[\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1}) + \mathbf{D}_{\mathbf{x}}\mathbf{F}\big(\mathbf{x}_{j-1} + h\mathbf{F}(\mathbf{x}_{j-1})\big)\big\{I + \mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1})\big\}\big]\Big].\end{aligned}$$
3. Finally, set
$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = 2\big((\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \mathbf{y}_1^T\big).$$

Note that this is precisely the scheme given in the general explicit one step discussion with $\mathbf{F}_j(\mathbf{x}_{j-1})$ given by (33) and

$$\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{F}_j(\mathbf{x}_{j-1}) = \Big[I + \frac{h}{2}\big[\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1}) + \mathbf{D}_{\mathbf{x}}\mathbf{F}\big(\mathbf{x}_{j-1} + h\mathbf{F}(\mathbf{x}_{j-1})\big)\big\{I + \mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1})\big\}\big]\Big].$$

Now, if $\mathbf{F}(\mathbf{x})$ is given by (2) so that

$$\mathbf{F}(\mathbf{x}) = (f_0(\mathbf{x}), \cdots, f_n(\mathbf{x}))^T$$

where we have written n for $N-1$ and each

$$f_i(\mathbf{x}) = u_i\big(\frac{u_{i+1} - u_{i-1}}{2h}\big), \quad i = 0, \cdots, n$$

with $\mathbf{x} = (u_0, \cdots, u_n)^T$, $u_{-1} = u_n$ and $u_{n+1} = u_0$. In this case, we have that $\mathbf{D_x F(x)}$ is a sparse $(n+1)\times(n+1)$ matrix. Indeed, using the notation

$$T(c_i, a_i, b_i; n) = \begin{pmatrix} a_0 & b_0 & 0 & \dots & 0 & c_0 \\ c_1 & a_1 & b_1 & \dots & 0 & 0 \\ 0 & c_2 & a_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ b_n & 0 & 0 & \dots & c_n & a_n \end{pmatrix}, \tag{35}$$

we have for this particular $\mathbf{F}$ that

$$\mathbf{D_x F(x)} = \frac{1}{2h} T(c_i, a_i, b_i; n) \tag{36}$$

with

$$\begin{aligned} c_i &= -x_i, \quad i = 0, \cdots, n \\ a_i &= x_{i+1} - x_{i-1}, \quad i = 0, \cdots, n \\ b_i &= x_i, \quad i = 0, \cdots, n, \end{aligned} \tag{37}$$

where $u_{-1} = u_n$ and $u_{n+1} = u_0$. Hence, for the formula given in step 1 above writing $\mathbf{x}_j = (x_0^j, \cdots, x_n^j)^T$, the matrix $\mathbf{D_x F}(\mathbf{x}_{r-1} + h\mathbf{F}(\mathbf{x}_{r-1}))$ is given by (36) with

$$\begin{aligned} c_i &= -(x_i^{r-1} + h f_i^{r-1}(\mathbf{x}_{r-1})) \\ a_i &= x_{i+1}^{r-1} + h f_{i+1}^{r-1}(\mathbf{x}_{r-1}) - x_{i-1}^{r-1} - h f_{i-1}^{r-1}(\mathbf{x}_{r-1}) \\ b_i &= x_i^{r-1} + h f_i^{r-1}(\mathbf{x}_{r-1}), \end{aligned} \tag{38}$$

where, as above, $x_{-1}^{r-1} = x_n^{r-1}$ and $x_{n+1}^{r-1} = x_0^{r-1}$. Thus, in this step and suceeding steps of this method the evaluation of $\mathbf{y}^T$ can be done efficiently by using recursive formulas that compute products of the form $\mathbf{z}^T A$, for $\mathbf{z} \in \mathbb{R}^n$ and A an $(n+1) \times (n+1)$ tridiagonal matrix, which avoid matrix operations.

For our second example we consider the two step explicit method given by the Adams-Bashforth method of order 2. Thus, the formulas of (19) become

$$\begin{aligned} \mathbf{x}_1 &= \mathbf{x}_0 + h\mathbf{F}(\mathbf{x}_0) \\ \mathbf{x}_j &= \mathbf{x}_{j-1} + \frac{h}{2}[3\mathbf{F}(\mathbf{x}_{j-1}) - \mathbf{F}(\mathbf{x}_{j-2})] \quad j \geq 2, \end{aligned} \tag{39}$$

where $\mathbf{F}$ is the function of (4). Once again using the notation that $\mathbf{D_x F(y)}$ denotes the Jacobian of $\mathbf{F}$ evaluated at $\mathbf{y}$ we have that the formulas (21) and (22) for the Jacobians of a general explicit 2-step scheme reduce to

$$\mathbf{D}_{\mathbf{x}_{j-1}} \mathbf{F}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) = I + \frac{3h}{2} \mathbf{D_x F}(\mathbf{x}_{j-1}) \tag{40}$$

and

$$\mathbf{D}_{\mathbf{x}_{j-2}} \mathbf{F}_j(\mathbf{x}_{j-1}, \mathbf{x}_{j-2}) = -\frac{h}{2} \mathbf{D_x F}(\mathbf{x}_{j-2}). \tag{41}$$

With these formulas, the iterative scheme (23)-(25) can be utilized to give the adjoint method for a scheme using the Adams-Bashforth method of order 2 as it's advection scheme. If the system being solved is that given by (2) and (3) then the matrices in (40) and (41) are the tridiagonal matrix given in (36) and (37) evaluated at the appropriate $\mathbf{x}$ value. Before leaving this example, we shall illustrate the Lagrange multiplier approach for generating the associated adjoint method. Let $\mathbf{F}$ denote the function of (4) and let $\mathbf{D_x F(y)}$ denotes the Jacobian of $\mathbf{F}$ evaluated at $\mathbf{y}$. Writing

$$H(\mathbf{x}_0, \cdots, \mathbf{x}_r, \lambda_1, \cdots, \lambda_r) = \sum_{j=0}^{r} (\mathbf{x}_j - \mathbf{x}_j^{obs})^T (\mathbf{x}_j - \mathbf{x}_j^{obs}) + \sum_{j=1}^{r} \lambda_j^T \left[\mathbf{x}_{j-1} + \frac{h}{2}(3\mathbf{F}(\mathbf{x}_{j-1}) - \mathbf{F}(\mathbf{x}_{j-2})) - \mathbf{x}_j \right] \tag{42}$$

where we have set $\mathbf{F}(\mathbf{x}_{-1}) = \mathbf{F}(\mathbf{x}_0)$ so that $\mathbf{x}_0 + \frac{h}{2}(3\mathbf{F}(\mathbf{x}_0) - \mathbf{F}(\mathbf{x}_{-1})) = \mathbf{x}_0 + h\mathbf{F}(\mathbf{x}_0)$ as defined in (19). Now, considering all the arguments of H to be functions of $\mathbf{x}_0$ we have by the chain rule that

$$\begin{aligned}\nabla_{\mathbf{x}_0} H(\mathbf{x}_0, \cdots, \mathbf{x}_r, \lambda_1, \cdots, \lambda_r) = &\Big[2(\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \lambda_1^T + h(\lambda_1^T - \frac{1}{2}\lambda_2^T)\mathbf{D}_{\mathbf{x}_0}\mathbf{F}(\mathbf{x}_0)\Big] \\ &+ \sum_{j=1}^{r}\Big[2(\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \lambda_{j+1}^T + \frac{h}{2}(3\lambda_{j+1}^T - \lambda_{j+2}^T)\mathbf{D}_{\mathbf{x}_j}\mathbf{F}(\mathbf{x}_j) - \lambda_j^T\Big]\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j \\ &+ \sum_{j=1}^{r}\Big[\mathbf{x}_{j-1} + \frac{h}{2}(3\mathbf{F}(\mathbf{x}_{j-1}) - \mathbf{F}(\mathbf{x}_{j-2})) - \mathbf{x}_j\Big]^T \mathbf{D}_{\mathbf{x}_0}\lambda_j\end{aligned} \tag{43}$$

Since we are assuming that we are using the Adams-Bashforth method as described in (39) as our time advection scheme, the last summation of (43) is zero and in this case the functions H and J are identical. Now using the fact that (43) is true for any choice of the parameters $\lambda_1, \cdots, \lambda_r$, we recursively select them to give a recursive procedure for calculating $\nabla_{\mathbf{x}_0} J(\mathbf{x}_0)$. Setting $\lambda_{r+1} = \lambda_{r+2} = \mathbf{0}$, we define these parameters recursively by requiring that

$$\lambda_j^T = 2(\mathbf{x}_j - \mathbf{x}_j^{obs})^T + \lambda_{j+1}^T + \frac{h}{2}(3\lambda_{j+1}^T - \lambda_{j+2}^T)\mathbf{D}_{\mathbf{x}_j}\mathbf{F}(\mathbf{x}_j) \tag{44}$$

for $j = r, r-1, \cdots, 1$. Finally, using the vectors λ_1^T and λ_2^T determined in (44), we have that

$$\nabla_{\mathbf{x}_0} J(\mathbf{x}_0) = 2(\mathbf{x}_0 - \mathbf{x}_0^{obs})^T + \lambda_1^T + h(\lambda_1^T - \frac{1}{2}\lambda_2^T)\mathbf{D}_{\mathbf{x}_0}\mathbf{F}(\mathbf{x}_0). \tag{45}$$

As a final note, we wish to observe that the above development could be done completely componentwise using summations rather than matrices and vectors. Although this approach would probably be more tedious, it would actually exhibit the precise dependence of all the variables and probably lead to correct Jacobians in a somewhat more straight forward manner. It would seem that this approach should be most effective when coupled with a symbolic algebra package to compute the needed partial derivatives. If one could couple with this a symbolic package that could also determine the structure of all matrices involved that would take effective advantage of sparseness and other special structure then this would be real close to giving an effective automated procedure for the adjoint method.

Next, we consider the case that the advection scheme is given by combining the above Adams-Bashforth explicit scheme with a second order Adams-Moulton correction to get a combined explicit and implicit scheme. Thus, if $\mathbf{F}$ denotes the function of (4) and $\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{y})$ denotes the Jacobian of $\mathbf{F}$ evaluated at $\mathbf{y}$ then the advection scheme becomes

$$\begin{aligned}\mathbf{x}_1 &= \mathbf{x}_0 + h\mathbf{F}(\mathbf{x}_0) \\ \mathbf{x}_j &= \mathbf{x}_{j-1} + \frac{h}{2}[3\mathbf{F}(\mathbf{x}_{j-1}) - \mathbf{F}(\mathbf{x}_{j-2})], \quad j \geq 2 \ \textit{and even}, \\ [I - \frac{h}{2}\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1})](\mathbf{x}_j - \mathbf{x}_{j-1}) &= h\mathbf{F}(\mathbf{x}_{j-1}), \quad j \geq 3 \ \textit{and odd},\end{aligned} \tag{46}$$

where this last equation is one Newton iteration applied to

$$\mathbf{x}_j - \frac{h}{2}\mathbf{F}(\mathbf{x}_j) = \mathbf{x}_{j-1} + \frac{h}{2}\mathbf{F}(\mathbf{x}_{j-1}), \tag{47}$$

initialized with $\mathbf{x}_{j-1}$. Note that the third equation of (42) is a linear system that determines $\mathbf{x}_j$ and must be solved at each odd numbered step of the advection scheme. When the matrix $I - \frac{h}{2}\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1})$ is banded then this may be done efficiently. Also, since the same system with different right hand sides will have to be solved to find the matrix $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_0)$, the same linear systems solver should probably be used in this case. Now for these odd steps, the matrices $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_0)$ are the solution to the linear equations

$$[I - \frac{h}{2}\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1})]\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_0) = I + \frac{h}{2}\mathbf{D}_{\mathbf{x}}\mathbf{F}(\mathbf{x}_{j-1}) + \Gamma(\mathbf{x}_{j-1}, \mathbf{x}_{j-2})$$

where

$$\mathbf{D_x F}(\mathbf{x}_{j-1}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}_{j-1}) & \cdots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}_{j-1}) \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}_{j-1}) & \cdots & \frac{\partial f_n}{\partial x_n}(\mathbf{x}_{j-1}) \end{pmatrix} \tag{48}$$

$$= \begin{pmatrix} a_{11}(\mathbf{x}_{j-1}) & \cdots & a_{1n}(\mathbf{x}_{j-1}) \\ \vdots & \ddots & \vdots \\ a_{n1}(\mathbf{x}_{j-1}) & \cdots & a_{nn}(\mathbf{x}_{j-1}) \end{pmatrix}$$

with $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_n(\mathbf{x}))^T$ and

$$\Gamma(\mathbf{x}_j, \mathbf{x}_{j-1}) = \begin{pmatrix} \left(d_1 \frac{\partial a_{11}}{\partial x_1}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{1n}}{\partial x_1}(\mathbf{z})\right) & \cdots & \left(d_1 \frac{\partial a_{11}}{\partial x_n}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{1n}}{\partial x_n}(\mathbf{z})\right) \\ \vdots & \ddots & \vdots \\ \left(d_1 \frac{\partial a_{n1}}{\partial x_1}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{nn}}{\partial x_1}(\mathbf{z})\right) & \cdots & \left(d_1 \frac{\partial a_{n1}}{\partial x_n}(\mathbf{z}) + \cdots + d_n \frac{\partial a_{nn}}{\partial x_n}(\mathbf{z})\right) \end{pmatrix} \tag{49}$$

with $d_k = (x_k^j - x_k^{j-1})$ and $\mathbf{z} = \mathbf{x}_{j-1}$. Now, in the adjoint steps one does not need the matrix $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_0)$ but rather the product of $\mathbf{w}^T\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_0)$. Thus, if a banded LU factorization for $I - \frac{h}{2}\mathbf{D_x F}(\mathbf{x}_{j-1})$ was available from the forward implicit step, say, then this same factorization could be used to find the columns of $\mathbf{D}_{\mathbf{x}_{j-1}}\mathbf{x}_j(\mathbf{x}_0)$, where the inner product of these columns and $\mathbf{w}$ could be calculated as they become available, avoiding matrix operations.

For our last example we begin the development of the adjoint method for a FORTRAN code given in appendix F of the book "An Introduction to Three Dimensional climate Modeling" by Washington and Parkinson. This code is a benchmark weather prediction program for comparing the performance of supercomputers that was coded by P. Swarztrauber in 1984 at NCAR. It is based on a paper by Sadourny, (1975) studying the dynamics of finite difference models of the shallow water equations. The continuous model of the shallow water equations used here is one where the Coriolis terms have been dropped and the potential vorticity is employed in the advection terms. The finite difference scheme is a staggered grid scheme that conserves potential enstrophy. The time differencing method is a centered difference scheme with a small amount of smoothing between successive time steps to damp out high oscillations. This scheme can be realized formally as a 3-step explicit method. To do this, we first introduce the predicted grid values $\mathbf{u}, \mathbf{v}$, and $\mathbf{p}$, the east-west and north-south velocities and the pressure/density, respectively. Using a grid spacing of Δx with M grid points in the x direction and a grid spacing of Δy with N grid points in the y direction and assuming all dependent values are periodic in both directions, these vectors are given by $\mathbf{u} = \{u_{ij}\}_{i=2,j=1}^{N+1,M}$, $\mathbf{v} = \{v_{ij}\}_{i=1,j=2}^{N,M+1}$, $\mathbf{p} = \{p_{ij}\}_{i=1,j=1}^{N,M}$, where each vector is ordered lexicographically (i.e., $\mathbf{u} = (u_{21}, u_{22}, \cdots, u_{2M}, u_{31}, \cdots, u_{N+1,1}, \cdots, u_{N+1,M})^T$, etc.). Also, the periodicity assumption requires that $u_{1j} = u_{N+1,j}$, $u_{11} = u_{N+1,M+1}$, etc. Write $\mathbf{x} = (\mathbf{u}, \mathbf{v}, \mathbf{p})^T$ and define $\mathbf{F} : \mathbb{R}^{3NM} \to \mathbb{R}^{3NM}$ by $\mathbf{F}(\mathbf{x}) = (\mathbf{f}_1(\mathbf{x}), \mathbf{f}_2(\mathbf{x}), \mathbf{f}_3(\mathbf{x}))^T$ where each $\mathbf{f}_k : \mathbb{R}^{3NM} \to \mathbb{R}^{NM}$ and corresponds to the steps of the code giving UNEW, VNEW and PNEW, respectively. For example,

$$(\mathbf{f}_1(\mathbf{x}) = (f_{21}^1(\mathbf{x}), \cdots, f_{2M}^1(\mathbf{x}), f_{31}^1(\mathbf{x}), \cdots, f_{N+1,1}^1(\mathbf{x}), \cdots, f_{N+1,M}^1(\mathbf{x}))^T$$

where

$$\begin{aligned} f_{ij}^1(\mathbf{x}) = & \left[\frac{\frac{4}{\Delta x}(v_{i,j+1} - v_{i-1,j-1}) - \frac{4}{\Delta y}(u_{i,j+1} - u_{ij})}{p_{i-1,j} + p_{ij} + p_{i,j+1} + p_{i-1,j+1}} + \frac{\frac{4}{\Delta x}(v_{ij} - v_{i-1,j}) - \frac{4}{\Delta y}(u_{ij} - u_{i,j-1})}{p_{i-1,j-1} + p_{i,j-1} + p_{i,j} + p_{i-1,j}}\right] \\ & \times \frac{1}{2}\left[(p_{i,j+1} + p_{ij}) \times v_{i,j+1} + (p_{i-1,j+1} + p_{i-1,j}) \times v_{i-1,j}\right. \\ & \left. + (p_{i-1,j} + p_{i-1,j-1}) \times v_{i-1,j} + (p_{ij} + p_{i,j-1}) \times v_{ij}\right] \\ & - \frac{1}{\Delta x}\left[p_{ij} + \frac{1}{4}(u_{i+1,j}^2 + u_{ij}^2 + v_{i,j+1}^2 + v_{ij}^2) - p_{i-1,j} - \frac{1}{4}(u_{ij}^2 + u_{i-1,j}^2 + v_{i-1,j+1}^2 + v_{i-1,j}^2)\right]. \end{aligned} \tag{50}$$

In actually setting these equations up one must also use the periodicity of these vectors whenever possible in order that the vector components that occur in these equations are in the primary range as listed above

(i.e., the first index of u must be in the range 2 to $N+1$ and the second index in the range 1 to M). The other equations for the two remaining component vector functions of $\mathbf{F}$ can be likewise written from the code. Given this notation, the advection scheme of this code for given initial fields $\mathbf{x}_0 = (\mathbf{u}_0, \mathbf{v}_0, \mathbf{p}_0)^T$ can be described as:

$$\begin{aligned}
\mathbf{x}_1 &= \mathbf{x}_0 + \frac{h}{2}\mathbf{F}(\mathbf{x}_0) \\
\mathbf{x}_2 &= \mathbf{x}_1 + h\mathbf{F}(\mathbf{x}_1) \\
\mathbf{x}_j &= \mathbf{x}_{j-2} + \alpha(\mathbf{x}_{j-1} + 2\mathbf{x}_{j-2} + \mathbf{x}_{j-3}) + h\mathbf{F}(\mathbf{x}_{j-1})
\end{aligned}$$

for $j \geq 2$ where α is the smoothing parameter and is given the value of .001 in the code as presented. Note that the same vector valued function $\mathbf{F}$ occurs in every step of this iterative scheme. Furthermore, the Jacobian of this function can be written in terms of the Jacobian functions of its three component functions. Indeed,

$$\mathbf{D_x F(y)} = \begin{pmatrix} \mathbf{D_x f}_1(\mathbf{y}) \\ \mathbf{D_x f}_2(\mathbf{y}) \\ \mathbf{D_x f}_3(\mathbf{y}) \end{pmatrix}$$

where the rows of $\mathbf{D_x f}_1(\mathbf{y})$, say, are found by taking the partials of the component functions of $\mathbf{f}_1$ having the same row position as the row sought in $\mathbf{D_x f}_1(\mathbf{y})$. Note that $\mathbf{D_x f}_j(\mathbf{y})$, for $j = 1, 2, 3$ can depend on any of the components of $\mathbf{x}$, although, the actual number of components of $\mathbf{x}$ on which a given component function depends is quite small. Thus, the following recursive formulas hold for the adjoint method in this problem:

$$\begin{aligned}
\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_1(\mathbf{x}_0) &= I + \frac{h}{2}\mathbf{D_x F}(\mathbf{x}_0) \\
\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_2(\mathbf{x}_0) &= [I + h\mathbf{D_x F}(\mathbf{x}_1)]\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_1(\mathbf{x}_0) \\
\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_j(\mathbf{x}_0) &= \mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-2}(\mathbf{x}_0) + \alpha\big(\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-1}(\mathbf{x}_0) + 2\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-2}(\mathbf{x}_0) + \mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-3}(\mathbf{x}_0)\big) \\
&\quad + h\mathbf{D_x F}(\mathbf{x}_{j-1})\mathbf{D}_{\mathbf{x}_0}\mathbf{x}_{j-1}(\mathbf{x}_0)
\end{aligned}$$

for $j \geq 3$ where the matrices on the right handed side are now defined recursively.

References

[1975] Sadourny, R., The dynamics of finite difference models of the shallow water equations, *J. Atm. Sci.*, **32**, 680 - 689.

[1986] Washington, W. M. and Parkinson, C. L., An Introduction to Three Dimensional Climate Modeling, *Oxford Press*, New York.

[1985] Buckley, A., ALGORITHM 630: BBVSCG - A variable-storage algorithm for function minimization, *ACM Trans. on Math. Soft.*, **11**, 103 - 119.

[1986] Le Dimet, F. X. and Talagrand, O., Variational algorithms for analysis and assimilation of meteorological observations: theoretical aspects, *Tellus*, **38A**, 97 - 110.

[1987a] Talagrand, O. and Courtier, P., Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory, *Q.J.R. Meteorol. Soc.*, **113**, 1311 - 1328.

[1987b] Talagrand, O. and Courtier, P., Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Numerical results, *Q.J.R. Meteorol. Soc.*, **113**, 1329-1347.

[1990] Courtier, P. and Talagrand, O., Variational assimilation of meteorological observations with the direct and adjoint shallow-equation, *Tellus*, **42A**, 531-549.

Appendix A

In this appendix we summarize some multidimensional derivative facts. However, at first let us observe that with the symbolic algebraic packages currently available (i.e., Maple, Mathematica, etc.) the task of calculating Jacobians could either be relegated to these codes or double checked by these codes. That is, the task of finding the derivatives described below could be completely automated; although, questions concerning effective computation with the resultant expressions would seem to still need individual attention.

Thus, let $\mathbf{F} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ with $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \cdots, f_m(\mathbf{x}))^T$, for $\mathbf{x} \in \mathbb{R}^n$ and each $f_j : \mathbb{R}^n \rightarrow \mathbb{R}$, $j = 1, \cdots, m$. Then

$$\mathbf{D_x F(x)} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \frac{\partial f_1}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \frac{\partial f_2}{\partial x_1}(\mathbf{x}) & \frac{\partial f_2}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_2}{\partial x_n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1}(\mathbf{x}) & \frac{\partial f_m}{\partial x_2}(\mathbf{x}) & \cdots & \frac{\partial f_m}{\partial x_n}(\mathbf{x}) \end{pmatrix} \tag{A.1}$$

where $\mathbf{x} = (x_1, \cdots, x_n)^T$, so that $\mathbf{D_x F(x)}$ is a $m \times n$ matrix. For the special case that $m = 1$ so that $\mathbf{F} = f : \mathbb{R} \rightarrow \mathbb{R}^n$ one has that

$$\begin{aligned} \mathbf{D_x} f(\mathbf{y}) &= \nabla_{\mathbf{x}} f(\mathbf{y}) \\ &= (\frac{\partial f}{\partial x_1}(\mathbf{y}), \cdots, \frac{\partial f}{\partial x_n}(\mathbf{y})) \end{aligned} \tag{A.2}$$

which is simply the gradient of f evaluated at $\mathbf{y}$.

Next, let $A : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$ be defined by

$$A(\mathbf{x}) = \begin{pmatrix} a_{11}(\mathbf{x}) & a_{12}(\mathbf{x}) & \cdots & a_{1n}(\mathbf{x}) \\ a_{21}(\mathbf{x}) & a_{22}(\mathbf{x}) & \cdots & a_{2n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}(\mathbf{x}) & a_{n2}(\mathbf{x}) & \cdots & a_{nn}(\mathbf{x}) \end{pmatrix} \tag{A.3}$$

where each $a_{ij} : \mathbb{R}^n \rightarrow \mathbb{R}$. Here $\mathbf{D_x} A(\mathbf{y})$ is a linear mapping of $A : \mathbb{R}^n \rightarrow \mathbb{R}^{n \times n}$ and using the definition of $\mathbf{D_x} A(\mathbf{y})$ it can be shown for $\mathbf{z} \in \mathbb{R}^n$ that

$$[\mathbf{D_x} A(\mathbf{y})](\mathbf{z}) = \begin{pmatrix} \nabla_{\mathbf{x}} a_{11}(\mathbf{y}) \cdot \mathbf{z} & \nabla_{\mathbf{x}} a_{12}(\mathbf{y}) \cdot \mathbf{z} & \cdots & \nabla_{\mathbf{x}} a_{1n}(\mathbf{y}) \cdot \mathbf{z} \\ \nabla_{\mathbf{x}} a_{21}(\mathbf{y}) \cdot \mathbf{z} & \nabla_{\mathbf{x}} a_{22}(\mathbf{y}) \cdot \mathbf{z} & \cdots & \nabla_{\mathbf{x}} a_{2n}(\mathbf{y}) \cdot \mathbf{z} \\ \vdots & \vdots & \ddots & \vdots \\ \nabla_{\mathbf{x}} a_{n1}(\mathbf{y}) \cdot \mathbf{z} & \nabla_{\mathbf{x}} a_{n2}(\mathbf{y}) \cdot \mathbf{z} & \cdots & \nabla_{\mathbf{x}} a_{nn}(\mathbf{y}) \cdot \mathbf{z} \end{pmatrix} \tag{A.4}$$

where $\nabla_{\mathbf{x}} a_{ij}(\mathbf{y}) \cdot \mathbf{z}$ represents the dot product on the two n-vectors $\nabla_{\mathbf{x}} a_{ij}(\mathbf{y})$ (the gradient of the function $a_{ij}(\mathbf{x})$ with respect to $\mathbf{x}$ evaluated at $\mathbf{y}$) and $\mathbf{z}$.

Finally, consider $\mathbf{H} : \mathbb{R}^n \rightarrow \mathbb{R}^n$ where $\mathbf{H}(\mathbf{x}) = A(\mathbf{x})\ \mathbf{F}(\mathbf{x})$ with $A(\mathbf{x})$ and $\mathbf{F}(\mathbf{x})$ (for $m = n$ here) are defined in the above paragraph. Then $\mathbf{D_x H}(\mathbf{x})$ is a linear mapping of $\mathbb{R}^n \rightarrow \mathbb{R}^n$. Thus, for $\mathbf{z} \in \mathbb{R}^n$, using the "product" rule we have that

$$\begin{aligned} [\mathbf{D_x H}(\mathbf{x})]\mathbf{z} &= [A(\mathbf{x})]\{[\mathbf{D_x F}(\mathbf{x})]\mathbf{z}\} + \{[\mathbf{D_x} A(\mathbf{x})]\mathbf{z}\}\mathbf{D_x F}(\mathbf{x}) \\ &= [A(\mathbf{x})]\{[\mathbf{D_x F}(\mathbf{x})]\mathbf{z}\} + \begin{pmatrix} \nabla_{\mathbf{x}} a_{11}(\mathbf{x}) \cdot \mathbf{z} & \cdots & \nabla_{\mathbf{x}} a_{1n}(\mathbf{x}) \cdot \mathbf{z} \\ \nabla_{\mathbf{x}} a_{21}(\mathbf{x}) \cdot \mathbf{z} & \cdots & \nabla_{\mathbf{x}} a_{2n}(\mathbf{x}) \cdot \mathbf{z} \\ \vdots & \ddots & \vdots \\ \nabla_{\mathbf{x}} a_{n1}(\mathbf{x}) \cdot \mathbf{z} & \cdots & \nabla_{\mathbf{x}} a_{nn}(\mathbf{x}) \cdot \mathbf{z} \end{pmatrix} \mathbf{F}(\mathbf{x}) \\ &= [A(\mathbf{x})]\{[\mathbf{D_x F}(\mathbf{x})]\mathbf{z}\} + \begin{pmatrix} (f_1(\mathbf{x})\nabla_{\mathbf{x}} a_{11}(\mathbf{x}) + \cdots + f_n(\mathbf{x})\nabla_{\mathbf{x}} a_{1n}(\mathbf{x})) \cdot \mathbf{z} \\ \vdots \\ (f_1(\mathbf{x})\nabla_{\mathbf{x}} a_{n1}(\mathbf{x}) + \cdots + f_n(\mathbf{x})\nabla_{\mathbf{x}} a_{nn}(\mathbf{x})) \cdot \mathbf{z} \end{pmatrix} \end{aligned}$$

Hence,

$$[\mathbf{D_x H(x)}] = \begin{pmatrix} a_{11}(\mathbf{x}) & a_{12}(\mathbf{x}) & \dots & a_{1n}(\mathbf{x}) \\ a_{21}(\mathbf{x}) & a_{22}(\mathbf{x}) & \dots & a_{2n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1}(\mathbf{x}) & a_{n2}(\mathbf{x}) & \dots & a_{nn}(\mathbf{x}) \end{pmatrix} \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \frac{\partial f_1}{\partial x_2}(\mathbf{x}) & \dots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \frac{\partial f_2}{\partial x_1}(\mathbf{x}) & \frac{\partial f_2}{\partial x_2}(\mathbf{x}) & \dots & \frac{\partial f_2}{\partial x_n}(\mathbf{x}) \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}) & \frac{\partial f_n}{\partial x_2}(\mathbf{x}) & \dots & \frac{\partial f_n}{\partial x_n}(\mathbf{x}) \end{pmatrix} \tag{A.5}$$

$$+ \begin{pmatrix} \left(f_1(\mathbf{x})\frac{\partial a_{11}}{\partial x_1}(\mathbf{x}) + \cdots + f_n(\mathbf{x})\frac{\partial a_{1n}}{\partial x_1}(\mathbf{x})\right) & \dots & \left(f_1(\mathbf{x})\frac{\partial a_{11}}{\partial x_n}(\mathbf{x}) + \cdots + f_n(\mathbf{x})\frac{\partial a_{1n}}{\partial x_n}(\mathbf{x})\right) \\ \vdots & \ddots & \vdots \\ \left(f_1(\mathbf{x})\frac{\partial a_{n1}}{\partial x_1}(\mathbf{x}) + \cdots + f_n(\mathbf{x})\frac{\partial a_{nn}}{\partial x_1}(\mathbf{x})\right) & \dots & \left(f_1(\mathbf{x})\frac{\partial a_{n1}}{\partial x_n}(\mathbf{x}) + \cdots + f_n(\mathbf{x})\frac{\partial a_{nn}}{\partial x_n}(\mathbf{x})\right) \end{pmatrix}$$