Pergamon

Bulletin of Mathematical Biology, Vol. 57, No. 4, pp. 593–617, 1995
Elsevier Science Ltd
© 1995 Society for Mathematical Biology
Printed in Great Britain. All rights reserved
0092–8240/95 $9.50 + 0.00

0092–8240(95)00011–9

A DATA ASSIMILATION TECHNIQUE APPLIED TO A PREDATOR–PREY MODEL

■ LINDA M. LAWSON
Department of Mathematics,
East Tennessee State University,
Johnson City, TN 37614-0002, U.S.A.
■ YVETTE H. SPITZ and EILEEN E. HOFMANN
Center for Coastal Physical Oceanography,
Old Dominion University,
Norfolk, VA 23529, U.S.A.
■ ROBERT BRYAN LONG
601 NE 2nd Place,
Dania, FL 33004, U.S.A.

A new approach for data assimilation, which is based on the adjoint method, but allows the computer code for the adjoint to be constructed directly from the model computer code, is described. This technique is straightforward and reduces the chance of introducing errors in the construction of the adjoint code. Implementation of the technique is illustrated by applying it to a simple predator–prey model in a model fitting mode. A series of identical twin numerical experiments are used to show that this data assimilation approach can successfully recover model parameters as well as initial conditions. However, the ease with which these values are recovered is dependent on the form of the model equations as well as on the type and amount of data that are available. Additional numerical experiments show that sufficient coefficient and parameter recoveries are possible even when the assimilated data contain significant random noise. Thus, for biological systems that can be described by ecosystem models, the adjoint method represents a powerful approach for estimating values for little-known biological parameters, such as initial conditions, growth rates, and mortality rates.

1. Introduction. During the past several years there has been an increase in the use of data assimilation techniques to improve forecasts obtained with numerical models. The most obvious application of data assimilation has been in meteorology, where data assimilation is now a routine component of numerical weather forecasting systems. More recently, improved data acquisition systems have resulted in the use of data assimilation techniques with numerical circulation models that have become more readily available in physical oceanography. Reviews of data assimilation methods as applied in meteorology and physical oceanography are found in Bengtsson *et al.* (1981), Lorenc (1986), Navon (1986), Haidvogel and Robinson (1989), and Ghil and Malanotte-Rizzoli (1991). Because of on-going advances in the technology associated with the acquisition, processing, and storage of large quantities of data, data assimilation is becoming an important topic in many areas where mathematical models are used to study physical or biological systems. One such area is the field of biological oceanography where satellite systems and other continuously-recording instruments are providing large quantities of data (Dickey, 1991).

Several methods, which include successive correction (Cressman, 1959; Bratseth, 1986), optimal interpolation (Gandin, 1963; Lorenc, 1981), Kalman filtering (Kalman, 1960; Kalman and Bucy, 1961; Ghil *et al.*, 1981), and the variational method (Lewis and Derber, 1985; Derber, 1985; Le Dimet and Talagrand, 1986; Lorenc, 1988a,b; Courtier and Talagrand, 1987; Talagrand and Courtier, 1987; Thacker, 1987; Thacker and Long, 1988; Long and Thacker, 1989a,b), have been developed for assimilation of data into numerical models. The latter procedure, often referred to as the adjoint method, differs from the other data assimilation procedures in that it can be applied to linear and non-linear models, can be implemented in a straightforward manner, and typically requires fewer calculations. This method minimizes a predetermined cost function which defines differences between model-derived quantities and measured quantities. A set of so-called "adjoint equations", derived from the model equations, are used to map the predefined cost function into the gradient of the cost function with respect to the adjustable input variables of the model. The gradient is then used in an iterative gradient–descent algorithm to progressively adjust the values of the input variables until they converge to values which maximize the agreement between model and observations (i.e. minimize the cost function). The result is an estimate of model parameters that give an optimal representation of the evolution of the state of the physical system being modeled.

The objectives of this paper are threefold. The first is to introduce a technique for constructing the adjoint computer code directly from the model computer code. This direct construction greatly reduces the errors that are potentially associated with adjoint code construction, and is straightforward to apply, thereby often making data assimilation more feasible. The second is to illustrate the use of this technique by applying it to a simple predator–prey model as a model-fitting scheme. The final objective is to show the usefulness of this approach for recovering not only model initial conditions, but also optimizing model parameters, especially growth and mortality rates. This aspect of the modeling is particularly important because growth and death rates that appear in biological models are often difficult to measure.

The focus of this study is on the use of the adjoint method with a biological model; not on the dynamics of the biological model. Hence, the predator–prey model used is simple. However, this model has the same basic structure as more complicated biological models, including marine ecosystem models. Moreover, this basic predator–prey model is well known in the biological community and therefore can serve as a prototype for illustrating data assimilation methods in the context of biological systems.

The following section presents a mathematical description of the adjoint method, a discussion of the technique for constructing the adjoint code, and a description of how the adjoint technique is combined with optimization techniques. In the third section we discuss the application of the technique, beginning with a description of the predator–prey model, and follow this by an example of the adjoint code construction. The results obtained from the application are discussed in the fourth section. The final section is a discussion and summary. The appendix contains details related to the mathematical description of the adjoint technique and a description of a method for verifying the correctness of the adjoint code.

2. The Adjoint Technique. The adjoint method of data assimilation consists of three components: the mathematical model with a cost function, the adjoint of the model, and an optimization technique. Values for the control variables are estimated and the model is implemented to provide a value of the cost function; the adjoint is then used to find the gradient of the cost function; the optimization technique makes use of the gradient to determine the direction and the optimal step size in that direction. Hence, the three components are used iteratively to adjust the initial estimates of the control variables (Fig. 1). These are discussed in the sections that follow.

2.1. *Description of the adjoint technique.* Adjoint techniques have been used in a variety of forms in applications to data assimilation problems. One of the more difficult aspects of applying adjoint techniques is the development of the computer code for the adjoint model. Some adjoint methods include development of the adjoint from the original model equations and coding the adjoint model, linearizing the computer code for the model (called the linear tangent model) and constructing the code for the adjoint from the computer code for the linearized form of the model (Talagrand, 1991), and computer construction of the adjoint code from the model code (Giering, 1992, 1995). An additional approach, which is described below, is a method of constructing the adjoint code that is based on a scheme that uses Lagrange multipliers to simplify the calculation of the partial derivatives needed to find the gradient of the cost function. The strength of this approach is that it makes it possible to construct the adjoint code directly from the model code.

Many mathematical models consist of a system of coupled equations, in which a set of control variables is required to determine a solution. The control variables may include those variables describing the initial state of the modeled system at the start of the assimilation interval, those representing parameters of the model, and those representing parameters of the forcing during the

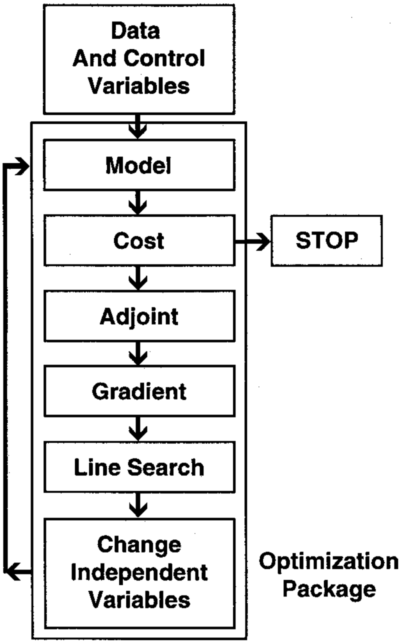

Figure 1. Schematic of the steps involved in the data assimilation scheme.

assimilation interval. The adjoint technique provides a method to find the best estimates of the control variables, which subsequently determine the evolution of the state of the physical system being modeled. These estimates are best with respect to the error statistics of the observations, and the model results using these estimates satisfying the dynamics of the model equations and the prescribed forcing.

Let the vector $\mathbf{x}=(x_1, \ldots, x_m)$ denote the model control variables. The model can be thought of as a sequence of calculations, starting with arbitrary values assigned to the control variables, where each quantity calculated is a value of a model dependent variable. The model variables are given by the vector $\mathbf{z}=(z_1, \ldots, z_N)$ and the collection of observations, or data, are represented by the vector $\mathbf{d}=(d_1, \ldots, d_p)$. Similarly, let the output of the model form a vector consisting of quantities to be compared with the data, called model equivalents of the data, and let it be represented by the vector $\mathbf{a}=(a_1, \ldots, a_p)$. A cost function is defined to provide a measure of the misfit between the model equivalents of the data and the data. The values of the control variables that give the minimum of the cost function will define the model state that best fits the data. A typical form for the cost function is a weighted sum of products of the differences between the model equivalents of the data and the data and can be written as

$$J = \tfrac{1}{2}(\mathbf{d}-\mathbf{a})^T A^{-1}(\mathbf{d}-\mathbf{a}). \tag{1}$$

The matrix A^{-1} is ideally the inverse of the error covariance matrix for the observations. If A^{-1} is approximated by a diagonal matrix, J becomes a weighted sum of squares. Since the goal of the assimilation process is to minimize the cost function with respect to the control variables, this technique is best described as a least-squares method when a cost function of this form is used.

From the definition of the cost function, J depends on the vectors $\mathbf{x}$ and $\mathbf{z}$ through the output $\mathbf{a}$, $J = J(\mathbf{x}, \mathbf{z})$. For the construction of the adjoint model, it is helpful to express the cost function as the last of the variables to be calculated in the sequence of calculations that represent the model. The equation $z_{N+1} = J(\mathbf{x}, z_1, \ldots, z_N)$ states that the value in z_{N+1} is the cost J resulting from the model starting with the input $\mathbf{x}$. If f_i, for $i = 1, \ldots N+1$, represent functions in the space of all possible values for the xs and zs, then the sequence of model calculations can be expressed as

$$z_1 = f_1(\mathbf{x}), \quad z_n = f_n(\mathbf{x}, z_1, z_2, \ldots, z_{n-1}), \quad \text{for } 1 < n \leqslant N+1. \tag{2}$$

The adjoint technique uses the gradient of the cost function in the space of the control variables to determine a direction to change the values of the control variables in order to minimize the cost function. However, the cost function is typically not expressed explicitly in terms of the xs, and the gradient calculation would therefore require extensive application of the chain rule. In order to avoid this difficulty, Lagrange multipliers are introduced. Although the introduction of these variables increases the number of variables involved, the technique facilitates the calculation of the partial derivatives of the cost function with respect to the components of the input vector $\mathbf{x}$. Hence, the Lagrange function, L, is introduced as

$$L(\mathbf{x}, \mathbf{z}, \lambda) = z_{N+1} - \lambda_1(z_1 - f_1(\mathbf{x})) - \sum_{i=2}^{N+1} \lambda_i(z_i - f_i(\mathbf{x}, z_1, \ldots, z_{i-1})), \tag{3}$$

where $\lambda = (\lambda_1, \ldots, \lambda_{N+1})$ is the vector of Lagrange multipliers. As shown in appendix A.1, at a saddle point of the Lagrange function, that is, a point in $\mathbf{x}$, $\mathbf{z}$, λ space where the partial derivatives of L vanish simultaneously, one obtains the model equations (2), the adjoint equations of the form,

$$\lambda_{N+1} = 1 \quad \text{and} \quad \lambda_n = \sum_{i=n+1}^{N+1} \frac{\partial f_i}{\partial z_n} \lambda_i, \; n = N, \ldots, 1 \tag{4}$$

and expressions for the gradient of the cost function with respect to the initial inputs to the model

$$\frac{\partial L}{\partial x_k}=\sum_{i=1}^{N+1}\frac{\partial f_i}{\partial x_k}\lambda_i, \quad 1\leqslant k\leqslant m. \tag{5}$$

It is the form of the Lagrange function given in (3) on which the technique for construction of the adjoint code is based, as discussed in the following section. The adjoint equations (4) can be used to find all of the Lagrange multipliers, which are then used in (5) to find the components of the gradient of the cost function. It is this gradient that provides the information on the direction to modify the control variables for the next step in the iterative process.

It should be noted that there are no restrictions put on the forms of the model equations as given in (2). In this equation any model variable, z_i, can in theory depend on all of the previously calculated variables (as well as the control variables). However, in practice any model varible typically depends explicitly on only a few of the previously calculated variables, and hence the form of the equations is not as complicated as indicated here. Additionally, the functions f_i need to be differentiable in some neighborhood of the cost minimum and the cost function must be differentiable with respect to its explicit arguments. It should be noticed that in practice the implementation of the adjoint equations (4) is performed in the reverse order than the operations for the model equations.

2.2. *Constructing the adjoint from model code.* In most applications of the adjoint technique, construction of the computer code for the adjoint model is from the continuous or discrete form of the model equations or relies on model equations that have been linearized in terms of the control variables. The first approach often requires difficult calculations. The second, referred to as the linear tangent model technique, requires the development of the computer code for the linear tangent model. The technique introduced here is not based on either of these techniques. This technique does not require the linearization of the model code, nor does it use a computer program construction of the adjoint. This technique provides a straightforward method of constructing the code for the adjoint model and is particularly useful when the model is not very complicated. Here the adjoint code is constructed directly from the model code, where one line of model code leads to one or more lines of adjoint code. Since this technique consists of applying rules to each line of model code, it provides a straightforward method of adjoint code construction, thereby greatly reducing the chance of introducing errors.

For this technique it is useful to think of the model as a sequence of calculations and the computer code for the model as a description of those calculations. In this sense each line of computer code in the model can then be thought of a single calculation in this sequence. This means that there is an analogy between a model equation that can be written in the form

$$z_k = f_k(\mathbf{x}, z_1, z_2, \ldots, z_{k-1}), \tag{6}$$

and a line of computer code that can be described as

$$Y = G(X, \ldots).$$

With this notation, the variable Y is redefined by a statement on the right side of the equation that involves the variable X and other variables. Notice in (3) that terms of the Lagrange function consist of products of Lagrange multipliers multiplied by differences between the left and right sides of the model equations. Hence, by thinking of each line of computer code as a model equation, a Lagrange function can be constructed in exactly the same form as the function given in (3). To illustrate this technique, let X represent a control variable in the model equations, Y represent a dependent variable, and Z an intermediate variable that is used in the calculations. Two typical lines of model code can be represented by

$$\begin{aligned} Y &= G(X, \ldots) \\ Z &= F(X, Y, \ldots). \end{aligned} \tag{7}$$

For this example the Lagrange function would be of the form

$$L = \ldots - \lambda_Y(Y - G(X, \ldots)) - \lambda_Z(Z - F(X, Y, \ldots)) + \ldots. \tag{8}$$

Recall that in the description of the Lagrange function, (3), Lagrange multipliers were introduced for each model variable, z_k. Hence, in the context of computer code Lagrange multipliers will be introduced for each variable, appearing on the left side in any line of code. Notice that it is not necessary to ever construct the Lagrange function, and it is only given here to show the connection between the Lagrange function that would be constructed from lines of code and the function given in (3).

Requiring that the derivatives of (8) vanish with respect to the control variables, dependent variables, and the Lagrange multipliers yields the gradient of the cost function, the adjoint equations, and the model equations, respectively. For example, the partial derivative of L with respect to the Lagrange multiplier for Y, λ_Y, vanishes when $\partial L/\partial \lambda_Y = 0$, giving the original line of model code $Y = G(X, \ldots)$. However, the adjoint equations are derived from the equations for the partial derivatives with respect to the control and dependent variables (see appendix A.1 for details). The requirement that the derivative of (8) vanishes with respect to Y gives the following equation,

$$\frac{\partial L}{\partial Y} = -\lambda_Y + \lambda_Z \frac{\partial F}{\partial Y} + \cdots = 0 \tag{9}$$

which is equivalent to

$$\lambda_Y = \lambda_Z \frac{\partial F}{\partial Y} + \cdots . \tag{10}$$

Thus, for the dependent variable Y, the line of model code, $Z = F(X, Y, \ldots)$, results in a contribution to the Lagrange multiplier of Y by the amount $\lambda_Z \partial F/\partial Y$, where Z is the variable that appears on the left side of the equation. Since these terms must be accumulated for every equation where Y appears on the right side, the equation that appears in the adjoint code is

$$\lambda_Y = \lambda_Y + \lambda_Z \frac{\partial F}{\partial Y}, \tag{11}$$

where care must be taken to initialize to zero the value stored in the variable names for the Lagrange multipliers.

The rule for construction of lines of adjoint code for a dependent variable will also apply when the variable is a control variable, denoted in these equations for lines of code by X. Recall that the component of the gradient of the cost function in the direction of X is the partial derivative of L with respect to X. Letting the component of the gradient be stored in the variable name of the Lagrange multiplier for X, λ_X, the line of model code $Z = F(X, Y, \ldots)$ leads to a line of code in the adjoint of the form

$$\lambda_X = \lambda_X + \lambda_Z \frac{\partial F}{\partial X}. \tag{12}$$

Notice that this is exactly the same rule that was used when the variable was a dependent variable, rather than a control variable. The difference here is that the value stored in the λ_X is, in fact, the component of the gradient of the cost function in the direction of the control variable X. This means that the repeated application of a single rule is all that is necessary to construct the adjoint code with the components of the gradient being equal to the Lagrange multipliers of the control variables. Again, it is important to realize that the computations in the adjoint equations are done in the reverse order of those in the model. Hence this line-by-line generation of the adjoint code must be done by reversing the order of the calculations in the model code. Details on how to check the adjoint computer code construction for errors are given in appendix A.2.

2.3. *Combining the model, adjoint, and optimization.* In the scheme described in the above sections, one output from the model is the value of the cost function, which gives a measure of the misfit between the model-derived values and the measured values. The adjoint equations then transform these misfits into the gradient of the cost function. The gradient is then used to find the direction to adjust the model control variables in order to decrease the difference between the model output and the data. Once the direction is found, the step size, that is the size of the change in that particular direction, must be determined. After the variables are adjusted by the calculated step size and direction, the model is again applied and the process repeated. Hence the iterative procedure consists of a model run, an adjoint run, and a step size calculation (Fig. 1). For linear models with a cost function of the form presented in (1), the number of iterations required to converge to the minimum of the cost is theoretically equal to the dimension of the vector of control variables, **x**. However, most models of physical phenomena are non-linear and require many more iterations.

The step size calculation, referred to as a line search, is usually done with standard optimization packages based on methods such as steepest descent and quasi-Newton. For the results presented here, a variable-storage quasi-Newton optimization procedure was used. This routine, N1QN3, is based on a technique described by Nocedal (1980), which has been further developed by Gilbert and Lemaréchal (1989). This procedure incorporates past history to determine the direction in which to modify the control variables in order to minimize the cost function. Other optimization packages were tested and these yielded no significantly better results. For comparison studies of various optimization packages see Gilbert and Lemaréchal (1989) and Zou *et al.* (1993).

3. Predator–Prey Model and Data Assimilation. The predator–prey model used in this study is given by

$$\frac{\mathrm{d}x}{\mathrm{d}t}=x(a_1+a_2x+a_3y) \tag{13}$$

$$\frac{\mathrm{d}y}{\mathrm{d}t}=y(a_4+a_5y+a_6x), \tag{14}$$

where x represents the concentration of the prey and y represents the concentration of the predator, given in number m^{-2}. The model coefficients are defined in Table 1. For this model, once the values for the coefficients and initial conditions are specified (Table 1), the solution is completely determined. This predator–prey model is standard and the properties of the solution to this model are well known. From an initial condition, the predator and prey concentrations converge to an equilibrium solution (Fig. 2a) after an initial adjustment (Fig. 2b). For the purpose of this study, (13) and (14) were solved numerically using an Euler scheme with a time step of 0.001 d.

Identical twin numerical experiments, that is, experiments where data are generated by the model, were used to test the data assimilation method. For the

Table 1. Definitions and values of the coefficients used in the predator–prey model. The solution obtained for this set of parameters is shown in Fig. 2

Variable	Definition	Units	Value
x	Prey concentration	Number m^{-2}	Calculated
y	Predator concentration	Number m^{-2}	Calculated
a_1	Prey specific growth rate	d^{-1}	4
a_2	Prey density dependence	(Number $\mathrm{m}^{-2})^{-1}$ d^{-1}	−2
a_3	Prey loss rate	(Number $\mathrm{m}^{-2})^{-1}$ d^{-1}	−4
a_4	Predator specific loss rate	d^{-1}	−6
a_5	Predator density dependence	(Number $\mathrm{m}^{-2})^{-1}$ d^{-1}	2
a_6	Predator growth rate	(Number $\mathrm{m}^{-2})^{-1}$ d^{-1}	4
x_1	Prey initial concentration	Number m^{-2}	1
y_1	Predator initial concentration	Number m^{-2}	1

form given in (13) and (14), the control variables consist of the coefficients a_i, $1 \leqslant i \leqslant 6$, and the initial conditions x_1 and y_1. In terms of the notation used in the discussion above, we are choosing the vector of control variables $\mathbf{x} = (a_1, a_2, \ldots, a_6, x_1, y_1)$. The dependent model variables and the model equivalents of the data are the same quantity, which is the concentration of the two species as a function of time. Parameter values and initial conditions for the model were specified (Table 1) and the time-dependent concentrations of the predator and prey were computed. These time-dependent distributions were then subsampled to obtain a data set. The model was then rerun with arbitrary values specified for model coefficients and/or initial conditions. The data for the predator and prey concentrations were then input into the data assimilation scheme in order to recover the original values of the model coefficients and/or initial conditions. Identical twin experiments provide a self-consistent data set for data assimilation and are used routinely to test data assimilation procedures.

The cost function that measures the differences between the model output and the data was defined to be

$$J = \frac{1}{2} \sum_i (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2, \tag{15}$$

where x_i and y_i are the model-derived estimates and $\hat{x}_i$ and $\hat{y}_i$ are the data values for the prey and predator, respectively. The summation is over indices i which correspond to times for which data exist. Since the cost function given in (15) is quadratic in form, the assimilation problem is essentially a least-squares optimization problem. This form of the cost function is equivalent to the definition of the cost function given in (1) where the covariance matrix is taken to be the identity matrix.

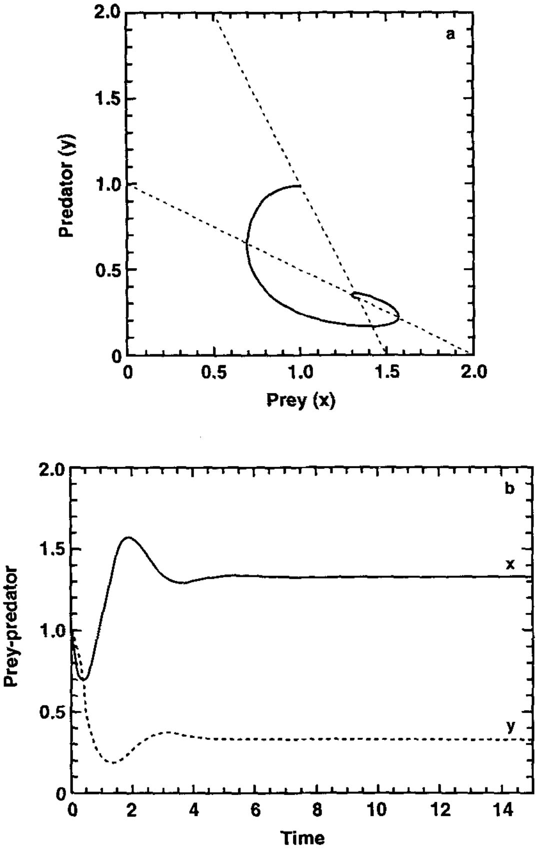

Figure 2. (a) Phase plane representation of the solution to the predator–prey model. Dashed lines are the predator and prey isoclines. Solid line is the trajectory of the solution. (b) Time domain representation of the solution to the predator–prey model. Time is in days.

3.1. *Construction of the adjoint code.* Given the code for the model, the method outlined above is easily applied to obtain the code for the adjoint. The following is an example of the application of this method in which a single line of model code is used to generate the lines that appear in the adjoint code.

In the development of this code $a(1), \ldots, a(6)$ are used for the model coefficients, $x(n)$, $y(n)$ are used for the x and y concentrations at time step n, and *nmax* denotes the maximum number of time steps. The computer code labels for the Lagrange multipliers are given by the model variable name preceded by an 'a'. For example, the Lagrange multiplier for $a(1)$ is $aa(1)$ and that for $x(n)$ is $ax(n)$. Recall that Lagrange multipliers will be used for each of the original coefficients, for the x and y values at each time step, and for the cost J.

The following lines of model code

```
    do 13 n=1, nmax-1
          x(n+1)=x(n)*(1.+dt*(a(1)+a(2)*x(n)+a(3)*y(n)))
          ⋮
 13 enddo
```

produce the following statements in the adjoint code:

```
    do 23 n=nmax-1,1,-1
          ⋮
          aa(3)=aa(3)+dt*y(n)*x(n)*ax(n+1)
          aa(2)=aa(2)+dt*x(n)*x(n)*ax(n+1)
          aa(1)=aa(1)+dt*x(n)*ax(n+1)
          ay(n)=ay(n)+dt*a(3)*x(n)*ax(n+1)
          ax(n)=ax(n)+dt*a(2)*x(n)*ax(n+1)
          ax(n)=ax(n)+(1.+dt*(a(1)+a(2)*x(n)+a(3)*y(n)))*ax(n+1)
 23 enddo
```

It is important that the variable names that store the Lagrange multipliers be initialized to zero upon entering the adjoint code. Also the construction of the adjoint code is done by following the model code in the reverse order of the calculations.

4. Results.

4.1. *Basic predator–prey model.* The most rigorous test of the data assimilation procedure was to attempt to recover values for all of the model coefficients and initial conditions. For this case, initial choices for the model coefficients and prey and predator initial conditions were made (Table 2) and data on the prey and predator concentrations were assimilated. The data set used for assimilation consisted of every other predator and prey concentration from the first 500 time intervals of the total data set (Fig. 2), which gave 250 data points. (The trajectory shown in Fig. 2 consists of 15,000 time intervals, that is, 15 days.) The recovery of all six model coefficients and initial prey and predator concentrations for these conditions is shown in Fig. 3.

The initial conditions were essentially recovered within the first 20 iterations (Fig. 3c). Once these values were found, the model coefficients were recovered with the recovery of the actual values of the prey coefficients (Fig. 3a) slightly

Table 2. Summary of the initial parameter values that were chosen for the data assimilation experiments. Also given are the modifications made to the basic predator–prey (PP) model given by (13) and (14) for the different cases and the prey (Pr) and predator (Pd) data that were assimilated. The results from the different cases are shown in the indicated figures

Case	a_1	a_2	a_3	a_4	a_5	a_6	x_1	y_1	Model form	Data input	Results
Case I	1	0	0	−1	0	0	2	2	PP	Pr and Pd Conc.	Figs 3 and 4
Case II	1	0	0	−1	0	0	2	2	PP scaled by a_1	Pr and Pd Conc.	Figs 5 and 6
Case III	1	0	0	−1	0	0	2	2	PP	Pr Conc. only	No convergence
Case IV	1	0	0	−1	0	0	2	2	PP with $a_3 = -a_6$	Pr and Pd Conc.	Fig. 7
Case V	1	0	0	−1	0	0	2	2	PP with $a_3 = -a_6$	Pr Conc. only	Fig. 8
Case VI	1	0	0	−1	0	0	0.1	0.1	PP	Pr and Pd Conc. with random noise	Fig. 9

preceding the recovery of the predator coefficients (Fig. 3b). In all cases, the exact original values (to four significant figures) were recovered. The value of the cost function rapidly decreased during the recovery of the initial conditions (Fig. 3d) and again later in the recovery process when the exact values of the coefficients were found. The pattern of initial adjustment of the initial conditions followed by the adjustment of the model coefficients was observed in all of the numerical twin experiments, suggesting that the initial values contribute more to the data misfits than the model parameters.

Additional numerical twin experiments (not shown) resulted in the recovery of all six model coefficients and initial values with as few as five data points of both predator and prey concentrations (1 point per 100 model time steps). However, the number of iterations required for convergence was much larger.

The ability to recover model coefficients and initial conditions and the iterations needed for the recovery are dependent on the structure of the cost function. For a nonlinear model, this structure can be quite complex and, for some cases, multiple minima are possible, which can result in nonunique solutions. In the present study, the cost surface is a function of all six model coefficients and two initial concentrations. A representation of the cost function would require an ability to plot in a nine-dimensional space. However, limited views of the cost surface are possible, such as the two-dimensional subspace representation of the logarithm of the cost function shown in Fig. 4. The prey growth and density dependent coefficients were varied and the other six parameters were fixed at their true values. This cost surface has a distinct minimum that corresponds to the true values of the coefficients a_1 and a_2.

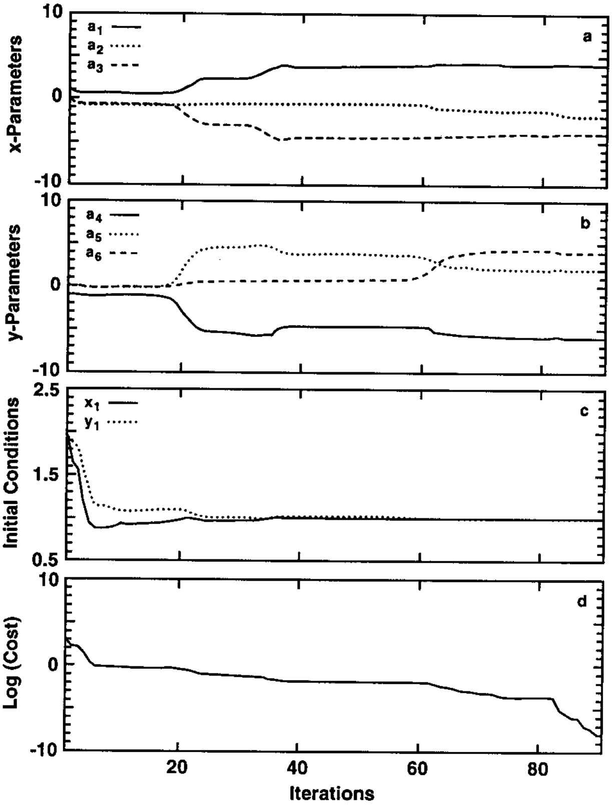

Figure 3. Recovery of the model coefficients for the (a) prey equation, (b) predator equation and the (c) predator and prey initial conditions for the basic predator–prey model, and the (d) logarithm of the cost function.

4.2. *Modified predator–prey model.* Modifying the structure of the predator–prey model from that given in (13) and (14) can alter the shape of the cost surfaces and thus the characteristics of the recoveries. To demonstrate this, the predator–prey model was written in the mathematically equivalent form

$$\frac{dx}{dt} = b_1 x(1 + b_2 x + b_3 y) \tag{16}$$

$$\frac{dy}{dt} = b_4 y(1 + b_5 y + b_6 x), \tag{17}$$

where $b_1 = a_1$, $b_1 b_2 = a_2$, etc. Using initial choices for the coefficients that correspond to the base case and the same data resolution (Table 2), it is still possible to recover all six model coefficients (Fig. 5a,b) and initial conditions

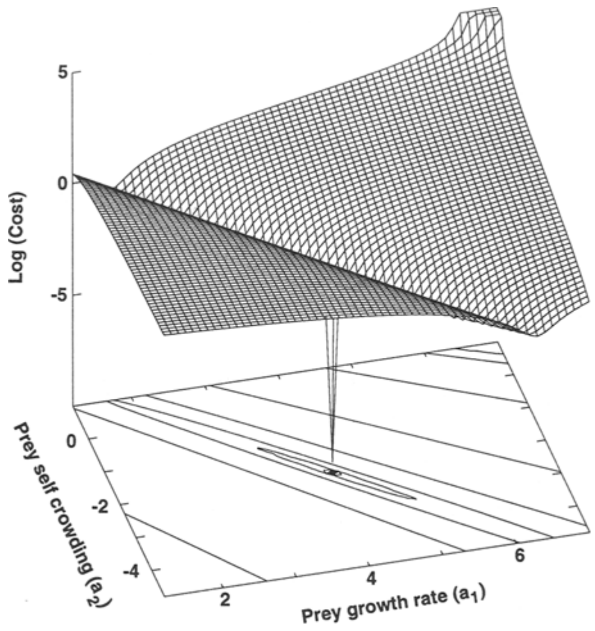

Figure 4. A representation of the structure of the logarithm of the cost function from the basic predator–prey model in the two-dimensional subspace (a_1, a_2).

(Fig. 5c). However, there is an increase in the number of iterations required for recovery.

This result can be explained by examining the structure of the logarithm of the cost surface in the subspace b_1, b_2 (Fig. 6). Unlike the surface from the basic predator–prey model, the axes of the contours in parameter space are no longer along straight lines. For a steepest descent technique (movement in the negative of the gradient) convergence is most rapid if the gradient lies on a line passing through the point at which the minimum cost is obtained. Although the optimization routine used in the present study was based on a quasi-Newton method, and the descent directions are not simply the negatives of the gradient, the curved nature of the axis of the trough in the cost surface contours results in a less rapid convergence.

4.3. *Data availability*. For some biological systems, it may not be possible to obtain simultaneous measurements of predators and prey. For example, concurrent measurements of zooplankton and phytoplankton in marine systems are difficult to obtain, especially for time scales of more than a few days. This would correspond to having observations for either x or y, but not both. In marine systems, observations for phytoplankton (prey) are usually available and this is also the quantity that can be measured from satellites. Hence, several

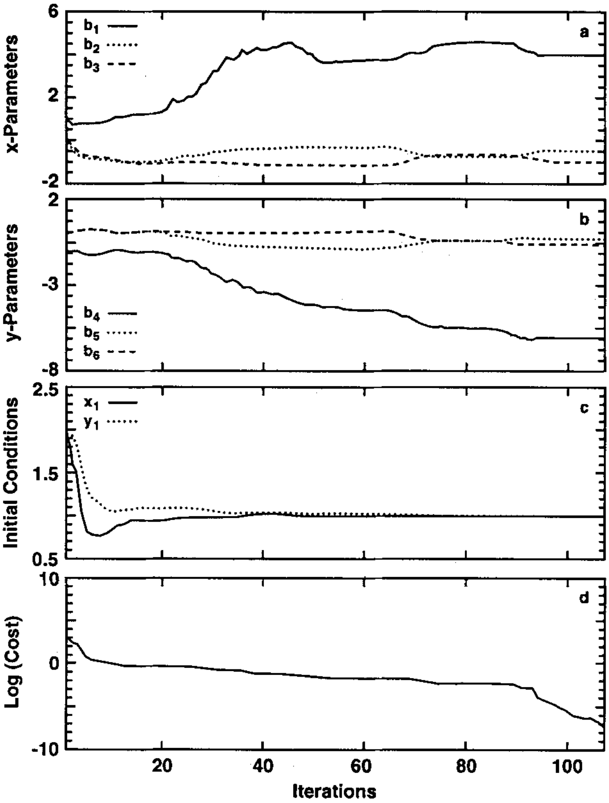

Figure 5. Recovery of the model coefficients for the (a) prey equation, (b) predator equation and the (c) predator and prey initial conditions for the modified predator–prey model, and the (d) logarithm of the cost function.

numerical experiments were done in which only prey observations were used for the data assimilation (Table 2). These data were input at different time resolutions. For all cases, it was not possible to consistently recover values for the six model coefficients and prey and predator initial conditions. This would seem to present a problem, then, for using satellite-derived measurements with predator–prey models.

However, if it is assumed that the prey and predator are linked by a common process, which is manifest in the mathematical model as a common coefficient, then parameter recovery is possible. For example, when $a_3 = -a_6$ the coefficients on the coupling terms of the model equations are such that there is strong feedback between the prey and predator. Using this convention, parameter recoveries were done where data were input for the prey and predator (Fig. 7) and the prey only (Fig. 8). Initial choices for the parameters are given in Table 2 and the predator and prey data sets were constructed as done for the basic model (Fig. 3).

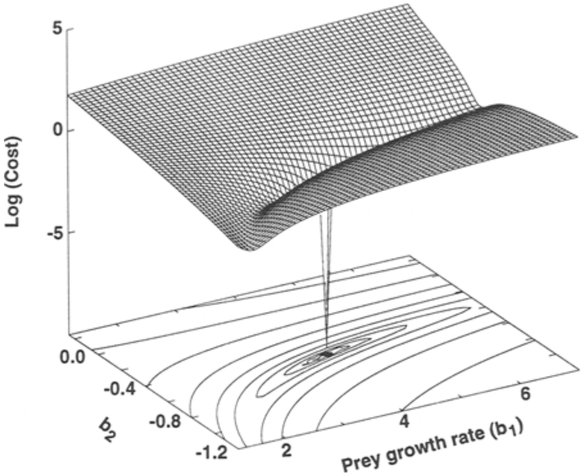

Figure 6. A representation of the structure of the logarithm of the cost function from the modified predator–prey model in the two-dimensional subspace (b_1, b_2).

Comparison of the parameter recovery from the two cases indicates that the primary difference is the large increase in the number of required iterations when data are available only for the prey. An order of magnitude increase occurs in the number of iterations, which greatly increases computation time. However, in spite of this, it is encouraging that all of the model coefficients and initial conditions can be recovered. It is interesting to note that when no measurements are available for the predator, the data assimilation scheme just adjusts the initial value for x and then adjusts the value for y and the model coefficients simultaneously. In this regard, the value of y is treated as simply another model parameter.

4.4. *Data noise.* For the numerical experiments described in the previous sections, the data input for data assimilation were assumed to be perfect, i.e. random and measurement errors were not allowed. However, data sets acquired for biological systems are not perfect and have a level of uncertainty associated with them. To investigate the effect that non-perfect measurements can have, numerical experiments (Table 2) were done in which normally distributed random noise with varying amplitude was added to the input data sets. The percentage of noise that was added to any data value varied between 1 and 20% of the value. Hence, these experiments simulated uncertainty that is inherent in data.

In general, the addition of noise to the data results in convergence to model parameters and initial conditions that are different from those of the true values (Fig. 9). The amount by which the recovered parameters are perturbed from the true values is related to the amount of noise in the input data. However, for

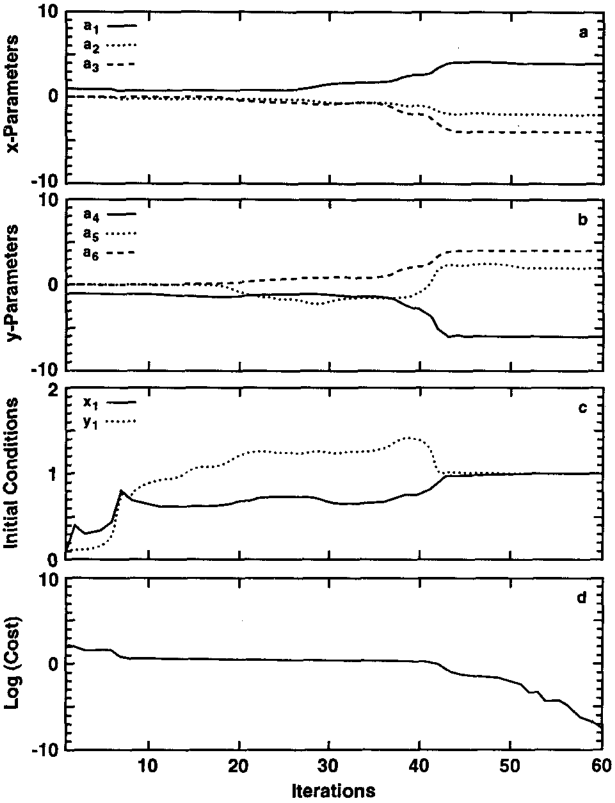

Figure 7. Recovery of the model coefficients for (a) prey equation, (b) predator equation and the (c) predator and prey initial conditions when the prey and predator are coupled through a common coefficient, and the (d) logarithm of the cost function.

the predator–prey model used in this study, the amount of displacement from the true values was small, even for a high percentage of noise. The density dependence coefficient in the prey and predator equations was the coefficient most affected by noise in the input data. Predator and prey initial conditions appeared to be equally affected by noise.

5. Discussion and Summary. A straightforward method of constructing the adjoint model code has been introduced that can be used to simplify the application of data assimilation schemes. The results obtained with this approach illustrate that this technique can be used with predator–prey type models to recover values for model coefficients and initial conditions. This is a powerful technique for biological systems where values for growth and mortality rates and initial conditions are often difficult or impossible to obtain.

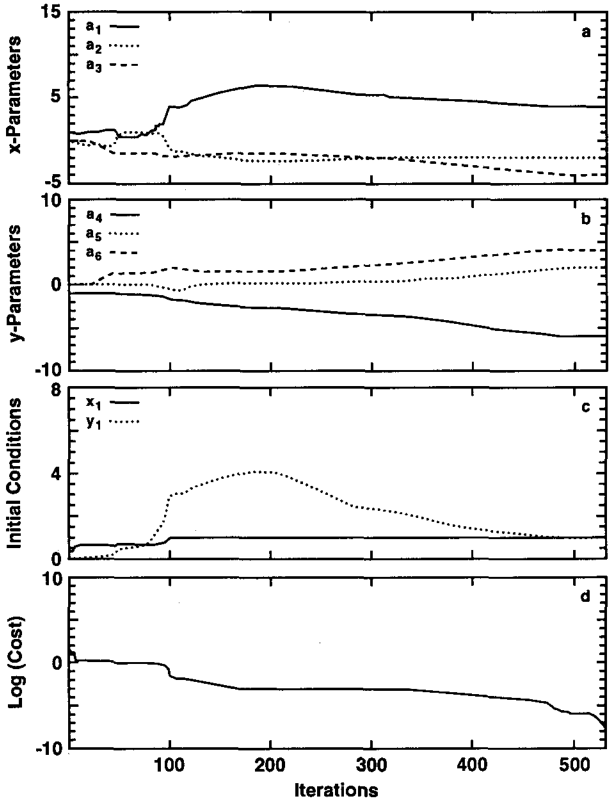

Figure 8. Recovery of the model coefficients for (a) prey equation, (b) predator equation and the (c) predator and prey initial conditions when data are input for the prey only, and the (d) logarithm of the cost function.

By having some information on species concentration or abundance over time, these rates can be obtained, assuming that the structure of the biological model is appropriate. For the cases presented, recovery of model coefficients was possible for all situations. Recovery of initial conditions was possible for all situations, except when the initial choices were in regions where one or both species increased without bound, i.e. the predator and prey systems were decoupled. However, for initial choices that were within the range of the true value as determined by biological and physical contraints, parameter recovery was always possible. This is not a severe limitation for most biological systems. Even the addition of noise to the input data sets did not significantly alter the ability to recover values from the data assimilation.

Perhaps the most encouraging result from this study is the ability to recover parameter values and initial conditions when data are available for only part of the biological system. However, when limited data are available the structure of the biological model needs to be carefully considered. In particular the

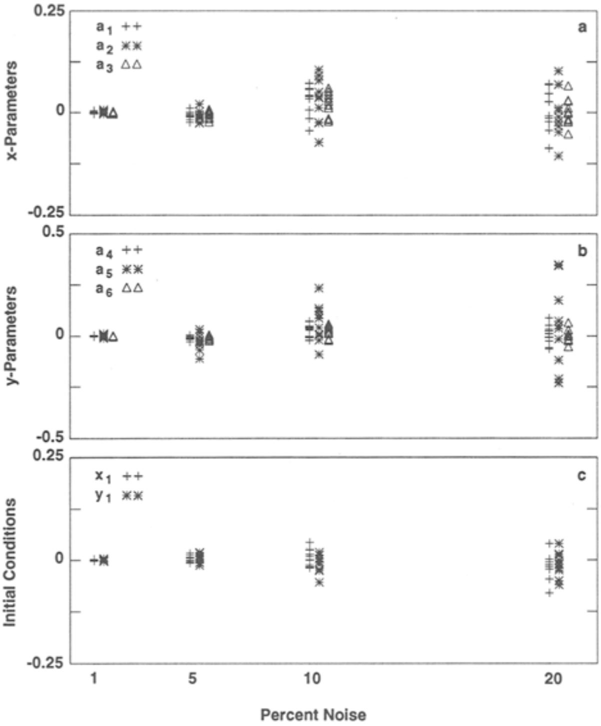

Figure 9. The effect of noise in the predator and prey data sets input for data assimilation on the values recovered for the (a) prey and (b) predator coefficients and (c) initial conditions. In each panel the differences between recovered and true values normalized by the true value are shown.

processes that couple ecosystem components must be well formulated because it is these forms that transfer information from the observations to the other model components. Thus, one use of the adjoint technique is to investigate the structure and dynamics of marine ecosystem models, as opposed to using data assimilation as a forecasting tool.

With advances in technology, the array of data types and the frequency at which measurements are made will continue to increase. Within the oceanographic community, more emphasis is being placed on satellites, acoustic and optical instruments to provide measurements of marine food web components. The data streams from these instruments are enormous. Hence, the development of models with data assimilation capability are an essential part of the management and analysis of these data (GLOBEC, 1991; Abbott, 1992). In particular, the availability of ocean circulation models and satellite-derived distributions of biological quantities, such as the pigment distributions associated with marine phytoplankton distributions derived from the Coastal Zone Color Scanner (Yoder *et al.*, 1988), have resulted in an interest in improving coupled circulation–ecosystem models by the use of data assimilation.

The first attempt at using satellite-derived ocean color distributions with a coupled circulation–ecosystem model consisted of simply replacing the simulated phytoplankton fields with the ocean color fields whenever such data were available (Ishizaka, 1990; Ishizaka and Hofmann, 1993). This study showed clearly that insertion of data for even a part of the marine ecosystem significantly improved the reliability of the simulated distributions. This approach was also useful in narrowing the range of values for several of the model parameters, such as phytoplankton growth rates. However, simply overwriting the model predicted fields with existing data (i.e. data insertion) has several problems associated with it. The results are never more accurate than the last data inserted, it is not possible to average redundant or similar information to reduce random effects of errors, and the model can be dynamically unbalanced due to injection of information into evolving solutions. This latter problem is especially acute for marine ecosystem models where inserting observations on one ecosystem component requires adjusting the other ecosystem components for which there are usually no measurements (Ishizaka, 1990, 1993). Hence, general application of data insertion is limited by available measurements and by knowledge of population parameters such as growth rates. The adjoint technique described here provides an alternative approach for assimilation of data into marine ecosystem models. The results of this study are a first step in this process.

The authors thank J. C. Gilbert for supplying the code and documentation for the optimization package, N1QN3. This research was made possible by the National Science Foundation Visiting Professorship for Women program grant NSF-GER-9350097, which allowed LML to spend 1 year at the Center for Coastal Physical Oceanography of Old Dominion University. Support for EEH was provided by NASA contract NAGW-3550 and support for YHS was provided by the Center for Coastal Physical Oceanography. The computer and facilities support provided by the Center for Coastal Physical Oceanography is appreciated.

APPENDIX

A.1. Lagrange Function. The model equations (2), the adjoint equations (4), and the components for the gradient of the cost function (5) are obtained by finding a saddle point of the Lagrange function (3), that is, a point in **x**, **z**, λ space where the partial derivatives of L vanish simultaneously. At a saddle point

$$\frac{\partial L}{\partial x_k}=0 \quad \text{and} \quad \frac{\partial L}{\partial z_n}=\frac{\partial L}{\partial \lambda_n}=0,$$

where $k=1, \ldots, m$ and $n=1, \ldots, N+1$. The requirement that $\partial L/\partial\lambda_n=0$, for $n=1, \ldots, N+1$ simply returns the model equations (2). The condition that

$$\frac{\partial L}{\partial z_{N+1}}=0$$

gives

$$\lambda_{N+1}=1.$$

From (3) the partial derivatives of L with respect to the dependent variables give

$$\frac{\partial L}{\partial z_n}=-\lambda_n+\sum_{i=n+1}^{N+1}\frac{\partial f_i}{\partial z_n}\lambda_i=0 \text{ for } n=N, \ldots, 1$$

which implies that

$$\lambda_n=\sum_{i=n+1}^{N+1}\frac{\partial f_i}{\partial z_n}\lambda_i,\ n=N, \ldots, 1. \tag{A1}$$

Knowing $\lambda_{N+1}=1$ it is possible to use this equation to calculate the remaining Lagrange multipliers. Notice that the calculations are done in the reverse order of those in the model equations. Finally, the equations that result from the condition that the partial derivatives of L with respect to the control variables vanish give the equations

$$\frac{\partial L}{\partial x_k}=\sum_{i=1}^{N+1}\frac{\partial f_i}{\partial x_k}\lambda_i,\quad 1\leqslant k\leqslant m. \tag{A2}$$

That these give the components of the gradient of the cost can be seen as follows. Restricted to regions in space where the model equations are satisfied, a saddle point of L corresponds to a point where the gradient of the cost function J is zero, that is, $\nabla_{\mathbf{x}}J=0$. Using (3) it can be seen that for $\mathbf{x}, \mathbf{z}$ where the model equations are satisfied $L=z_{N+1}=J$, so that $\nabla_{\mathbf{x}}J=\nabla_{\mathbf{x}}L$. Since $L=L(\mathbf{x}, \mathbf{z}, \lambda)$ and $\mathbf{z}$ and λ depend on $\mathbf{x}$, by the chain rule

$$(\nabla_{\mathbf{x}}L)_k=\frac{\partial L}{\partial x_k}+\sum_{i=1}^{N+1}\left(\frac{\partial L}{\partial z_i}\frac{\partial z_i}{\partial x_k}+\frac{\partial L}{\partial \lambda_i}\frac{\partial \lambda_i}{\partial x_k}\right).$$

But at a saddle point for L we have $\partial L/\partial z_i=\partial L/\partial\lambda_i=0$ for $1\leqslant i\leqslant N+1$ so that

$$(\nabla_{\mathbf{x}}L)_k=\frac{\partial L}{\partial x_k}.$$

Hence $(\nabla_{\mathbf{x}}J)_k=\partial L/\partial x_k$ or $\nabla_{\mathbf{x}}J=\partial L/\partial\mathbf{x}$. The values of the Langrange multipliers are found by (A1) and then used in (A2) to calculate the components of the gradient of the cost function. Hence the adjoint equations map the cost function to the gradient of the cost function using the Lagrange multipliers as intermediate variables.

A.2. Verification of the Adjoint Model. Errors that may be introduced during construction of the adjoint code can be detected by perturbing the input vector, $\mathbf{x}$, by an amount $s\mathbf{u}$, where $\mathbf{u}$ denotes the direction along which $\mathbf{x}$ changes and s, a scalar, is the amount of the change. This change in the input will produce a change in the model output $\mathbf{a}$ to a value $\mathbf{a}'$ where $\delta\mathbf{a}=\mathbf{a}'-\mathbf{a}$. For a fixed direction $\mathbf{u}$, the cost is a function of the scalar s and changing s by $\delta s=\mathrm{d}s=s_0$, the change in the cost function is given by

$$\mathrm{d}J = \frac{\mathrm{d}J}{\mathrm{d}s}\,\mathrm{d}s = \frac{\mathrm{d}J}{\mathrm{d}s}\,s_0.$$

However, thinking of the cost as a function of **x**, the change in the cost can be approximated by

$$\mathrm{d}J \approx \nabla_{\mathbf{x}} J \cdot \mathrm{d}\mathbf{x} = \nabla_{\mathbf{x}} J \cdot (s_0\mathbf{u}) = s_0 \nabla_{\mathbf{x}} J \cdot \mathbf{u} = s_0 (\nabla_{\mathbf{x}} J)^T \mathbf{u}.$$

Hence

$$(\nabla_{\mathbf{x}} J)^T \mathbf{u} = \frac{\mathrm{d}J}{\mathrm{d}s}.$$

Using the form of J given in (1), the change in the cost function can be written as

$$\frac{\mathrm{d}J}{\mathrm{d}s} = (\mathbf{d}-\mathbf{a})^T A^{-1}\left(-\frac{\mathrm{d}\mathbf{a}}{\mathrm{d}s}\right).$$

However, for very small step sizes s_0, we have $\mathrm{d}\mathbf{a}/\mathrm{d}s \approx \delta\mathbf{a}/\delta s = \delta\mathbf{a}/s_0$. Substituting into the equation above gives

$$\frac{\mathrm{d}J}{\mathrm{d}s} = -(\mathbf{d}-\mathbf{a})^T A^{-1}\frac{\delta\mathbf{a}}{s_0} = (\nabla_{\mathbf{x}} J)^T \mathbf{u}.$$

Hence in the limit as s_0 goes to 0 we have

$$\lim_{s_0 \to 0} \frac{-(\mathbf{d}-\mathbf{a})^T A^{-1}\delta\mathbf{a}}{s_0 (\nabla_{\mathbf{x}} J)^T \mathbf{u}} = 1.$$

The numerator of this equation is determined from the model while the denominator is found using the adjoint. If there are no errors in the adjoint code this ratio should approach 1. It is possible to use for the vector **u** the gradient of the cost function. Although there are no restrictions to the choice of the vector **u** this scheme will not detect errors in directions of variables that are orthogonal to the vector **u**.

REFERENCES

Abbott, M. R. 1992. Report of the U.S. JGOFS workshop on modeling and data assimilation. U.S. JGOFS Planning Report Number 14, 28 pp.

Bengtsson, L., M. Ghil and E. Källén (Eds). 1981. *Dynamic Meteorology: Data Assimilation Methods*. New York: Springer.

Bratseth, A. M. 1986. Statistical interpolation by means of successive corrections. *Tellus* **38A**, 439–447.

Courtier, P. and O. Talagrand. 1987. Variational assimilation of meteorological observations with the adjoint vorticity equation, II. Numerical results. *Q.J.R. met. Soc.* **113**, 1329–1347.

Cressman, G. P. 1959. An operational objective analysis system. *Mon. Wea. Rev.* **87**, 367–374.

Derber, J. C. 1985. The variational four-dimensional assimilation of analyses using filtered models as constraints. Ph.D. dissertation, 142 pp. Univ. of Wisconsin, Madison, Wisconsin.

Dickey, T. D. 1991. The emergence of concurrent high-resolution physical and bio-optical measurements in the upper ocean and their applications. *Rev. Geophys.* **29**, 383–413.

Gandin, L. S. 1963. *Objective Analysis of Meteorological Fields*. (From the Russian) Israel program for Scientific Translations, 1965, Jerusalem, 242 pp.

Ghil, M. and P. Malanotte-Rizzoli. 1991. Data assimilation in meteorology and oceanography. *Adv. Geophys.* **33**, 141–266.

Ghil, P. E., W. Murray and M. H. Wright. 1981. *Practical Optimization.* Orlando: Academic Press.

Giering, R. 1992. *Adjoint Model Compiler.* Max-Planck-Institute für Meterologie.

Giering, R. 1995. Adjoint code generation. Max-Planck-Institute für Meterologie, in preparation.

Gilbert, J. C. and C. Lemaréchal. 1989. Some numerical experiments with variable-storage quasi-Newton algorithms. *Math. Prog.* **45**, 405–435.

GLOBEC. 1991. Theory and modeling in GLOBEC: a first step. U.S. GLOBEC Report Number 1, 9 pp.

Haidvogel, D. B. and A. R. Robinson (Eds). 1989. Special issue: data assimilation. *Dyn. Atmos. Oceans* **13**, 171–513.

Ishizaka, J. 1990. Coupling of Coastal Zone Color Scanner data to a physical–biological model of the southeastern U.S. continental shelf ecosystem—3. Nutrient and phytoplankton fluxes and CZCS data assimilation. *J. geophys. Res.* **95**, 20,167–20,181.

Ishizaka, J. 1993. Data assimilation for biogeochemical models. In *Towards a Model of Ocean Biogeochemical Processes.* G. T. Evans and M. J. R. Fasham (Eds), pp. 295–316. New York: Springer.

Ishizaka, J. and E. E. Hofmann. 1993. Coupling of ocean color data to physical-biological models. In *Ocean Colour: Theory and Applications in a Decade of CZCS Experience.* V. Barale and P. M. Schlittenhardt (Eds), pp. 271–288. ECSC, EEC, EAEC, The Netherlands.

Kalman, R. E. 1960. A new approach to linear filtering and prediction problems. *J. Basic Engng* (*Transactions of the ASME*) **82D**, 35–45.

Kalman, R. E. and R. S. Bucy. 1961. New results in linear filtering and prediction theory. *J. Basic Engng* (*Transactions of the ASME*) **83D**, 95–108.

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

Lewis, J. M. and J. C. Derber. 1985. The use of adjoint equations to solve a variational adjustment problem with adjective constraints. *Tellus, Ser. A* **37**, 309–322.

Long, R. B. and W. C. Thacker. 1989a. Data assimilation into a numerical equatorial ocean model, Part I: the model and the assimilation algorithm. *Dyn. Atmos. Oceans* **13**, 379–412.

Long, R. B. and W. C. Thacker. 1989b. Data assimilation into a numerical equatorial ocean model, Part II: assimilation experiments. *Dyn. Atmos. Oceans* **13**, 413–440.

Lorenc, A. C. 1981. A global three-dimensional multivariate statistical interpolation scheme. *Mon. Wea. Rev.* **109**, 701–721.

Lorenc, A. C. 1986. Analysis methods for numerical weather prediction. *Q. J. R. met. Soc.* **112**, 1177–1194.

Lorenc, A. C. 1988a. Optimal non-linear objective analysis. *Q. J. R. met. Soc.* **114**, 205–240.

Lorenc, A. C. 1988b. A practical approximation to optimal 4-dimensional objective analysis. *Mon. Wea. Rev.* **116**, 730–745.

Navon, I. M. 1986. A review of variational and optimization methods in meteorology. In *Variational Methods in Geosciences*, Y. Sasaki (Ed.), pp. 29–34. New York: Elsevier.

Nocedal, J. 1980. Updating quasi-Newton matrices with limited storage. *Math. Comput.* **35/151**, 773–782.

Talagrand, O. 1991. The use of adjoint equations in numerical modeling of the atmospheric circulation. In *Automatic Differentiation of Algorithms: Theory, Implementation and Application*, A. Griewank and G. Corliess (Eds), pp. 169–180. SIAM.

Talagrand, O. and P. Courtier. 1987. Variational assimilation of meteorological observations with the adjoint vorticity equation, I, theory. *Q. J. R. met. Soc.* **113**, 1311–1328.

Thacker, W. C. 1987. Three lectures on fitting numerical models to observations. External Rep., GKSS 87/E/65, GKSS Forschungszentrum Geesthacht, Geesthacht.

Thacker, W. C. and R. B. Long. 1988. Fitting dynamics to data. *J. geophys. Res.* **93**, 1227–1240.

Yoder, J. A., W. E. Esaias, G. C. Feldman and C. R. McClain. 1988. Satellite ocean color-status report. *Oceanography* **1**, 18–20.

Zou, X., I. M. Navon, M. Berger, K. H. Phua, T. Schlick and F. X. Le Dimet. 1993. Numerical experience with limited-memory quasi-Newton and truncated Newton methods. *SIAM J. Optim.* **3**, 582–608.

Received 1 May 1994
Revised version accepted 10 January 1995