

Development of a Four-Dimensional Variational Analysis System Using the Adjoint Method at GLA. Part 1: Dynamics

WINSTON C. CHAO

Goddard Laboratory for Atmospheres, NASA/GSFC, Greenbelt, Maryland

LANG-PING CHANG

General Sciences Corporation, Laurel, Maryland

(Manuscript received 12 August 1991, in final form 25 November 1991)

ABSTRACT

Recent developments in the field of data assimilation have pointed to variational analysis (essentially least-squares fitting of a model solution to observed data) using the adjoint method as a new direction that holds the potential of major improvements over the current optimal interpolation (OI) method. The shortcomings of the existing OI analysis method, such as the questionable basic assumptions underlying some of the statistical formulations and the linearity between the analysis and the observation, do not exist in variational analysis. Moreover, variational analysis, by fitting a model solution to data, has the potential of avoiding the long-standing spinup problem in numerical weather prediction. Finally, when the resulting analysis is used as the initial condition for forecasting, since initialization will be performed internally to the analysis procedure, no separate initialization procedure is required before forecast starts.

This paper describes the initial effort in the development of a four-dimensional variational analysis system. Although the development is based on the Goddard Laboratory for Atmospheres General Circulation Model (GLA GCM), the methods and procedures described in this paper can be applied to any model. The adjoint code that computes the gradients needed in the analysis can be written directly from the GCM code. An easy error-detection technique was devised in the construction of the adjoint model. Also, a method of determining the weights and the preconditioning scales for the cases where model-generated data, which are error free, are used as observation is proposed. Two test experiments show that the dynamics part of the system has been successfully completed. A limited comparison of two minimization codes was conducted. The procedures presented in this work are general and can be applied to various variational and sensitivity studies.

1. Introduction

In diagnostic studies of atmospheric general circulation and in determining initial conditions for numerical weather prediction, information on the state of the atmosphere on a global or on a limited domain basis and at regular intervals is required. Since observation, despite the use of satellites, does not have the required comprehensive coverage for all state parameters, the general circulation model solution has been introduced to help to fill the void. The task of blending the observed data and the model solution in a consistent manner in order to generate the "best" estimate of the four-dimensional atmospheric state is called *data assimilation*.

Currently, the method of data assimilation in use at major meteorological centers employs optimal inter-

polation (OI) (e.g., Lorenc 1981) as the objective analysis method. The OI is used sequentially in that as the model runs the available observations are inserted in a statistically optimal manner. Efforts have been devoted to making the model retain the inserted observational data—thus the word assimilation. One obvious property of the operational data assimilation is that the observed data can influence the outcome of the procedure or the analysis only at the current or future time. Despite its improvements over the previous methods, the current OI method still has limitations. First, being sequential, it is not a true four-dimensional assimilation method. Second, the vertical and horizontal separability assumption in the computation of correlation coefficients used in OI is only for mathematical convenience and does not have any physical basis. Many other assumptions, such as Gaussian error statistics (variational approach in using the least-squares distance function also makes this assumption), are made also for the purpose of convenience. Third, observations can affect the current analysis only in a linear manner—though current modifications of OI

Corresponding author address: Dr. Winston C. Chao, Goddard Laboratory for Atmospheres, NASA/GSFC Code 913, Greenbelt, MD 20771.

are addressing this problem—whereas a nonlinear influence is expected from a physical point of view. Nevertheless it should be pointed out that the weaknesses of the current operational method are more of an implementation nature than anything theoretical.

In view of these questionable features of the operational data assimilation, researchers, besides making further improvements (e.g., Lorenc 1988a,b; Parrish 1989), have devoted efforts to exploring other approaches. Among them a very promising one is variational analysis using an adjoint model. In this approach, the model solution is used to fit the observed data in a least-squares sense. The fit is measured by a distance function (also called cost, penalty, or objective function), which is a time integration of instantaneous distance of model state from the observation for the duration of the analysis. The initial and boundary conditions, and/or externally added forcing, can be adjusted in an iterative procedure to minimize the distance function, thus creating the best fit. The model solution in this best fit is the resulting analysis.

In the minimization procedure one uses a descent algorithm, which requires at each iteration the gradients of the distance function with respect to all the adjusting parameters (e.g., the initial and/or boundary conditions). A simple-minded way of obtaining these gradients is to perturb each adjusting parameter one at a time and to evaluate the variation in the distance function (Hoffman 1986). Obviously, since each evaluation of the distance function requires a model integration for the duration of the analysis period, and since for a GCM with ordinary resolution the adjusting parameters are numbered in the tens of thousands, this approach is impossible to implement. A feasible, yet still CPU time-demanding way of evaluating these gradients is through the adjoint method. Here, for each iteration in the descent algorithm, the forecast model is integrated forward once and the adjoint of the tangent linear equation is integrated backward to the initial time. The result of this second integration gives the gradients one seeks. A more detailed description of this approach will be given in the next section.

One important problem with variational analysis is that the model is not perfect. The model solution, which is used as the resulting analysis, despite its best fit to data, does not precisely describe the real atmosphere. This problem can be somewhat ameliorated by using additional three-dimensional forcing terms in the model equation as adjusting parameters (Derber 1989), which is a way of using the model as a weak constraint. However, there is a limit to what can be achieved by this remedy. The model error has a time spectrum. The additional forcing terms can correct only very few components. Thus, the importance of improving the model performance, with regard to variational analysis, can never be overemphasized.

The comparison of variational analysis with Kalman filter analysis has so far been only at a theoretical level.

For a linear model used as a weak constraint, the two approaches give the same results at the end of the analysis period (Thacker 1986). The comparison in cost will not be clear until more efforts have been devoted to both approaches.

One important property of variational analysis is that data at one time level can influence the analysis results of previous time levels. Thus, this approach is truly four-dimensional. Also, the relationship between data and analysis is nonlinear when the model or the forward interpolation is nonlinear or the distance function is not quadratic. The subsequent forecast starting from the ending time of the resulting analysis, which is a model solution with balance between dynamics and physics, has much diminished spinup problem (i.e., the first 6-h precipitation is too high or too low), though the analysis still suffers from the climate drift problem that is essentially a modeling problem. Also, since measures can be taken to suppress unrealistic gravity waves in variational analysis, a separate initialization procedure at the start of the forecast is not necessary. Furthermore, variational analysis is particularly suitable for assimilating synoptic data, such as satellite data. In view of these exciting promises it is not surprising that considerable effort has been devoted to variational analysis (Lewis and Derber 1985; Le Dimet and Talagrand 1986; Derber 1987, 1989; Talagrand and Courtier 1987; Courtier and Talagrand 1987, 1990; Lorenc 1988a,b; Navon et al. 1992).

Section 2 describes in more detail the fundamentals of variational analysis using an adjoint model. Section 3 describes our developmental effort using the Goddard Laboratory for Atmospheres General Circulation Model (GLA GCM); it also includes our contributions to error detection in the adjoint model and to weight setting when model-generated data are used as observations. Section 4 gives some test results, which indicate that the dynamics portion of our adjoint model has been successfully completed. Section 5 contains a brief discussion of the rate of convergence. Section 6 contains the summary and concluding remarks.

2. Fundamentals of variational analysis using the adjoint method

a. Fitting model solution to data

In variational analysis certain parameters of a model are changed in order to fit the model solution to observed data and background. The fit is measured by a distance function

$$D = \frac{1}{2} \sum_{i=0}^m (\mathbf{X}_i - \hat{\mathbf{X}}_i)^T \mathbf{W} (\mathbf{X}_i - \hat{\mathbf{X}}_i) + (\mathbf{X}_0 - \mathbf{X}_b)^T \mathbf{B}^{-1} (\mathbf{X}_0 - \mathbf{X}_b),$$

where array \mathbf{X}_i is the model analog (i.e., model solution interpolated to the observation location) of observation

$\hat{\mathbf{X}}_i$ at time level i , which runs from 0 to m , and \mathbf{W} is the inverse of the observation error (including measurement error and representation error) covariance, which has different values for different variable types. The smaller D is the better the fit is. The method used for computing \mathbf{W} for our experiments will be discussed in a later section. Here \mathbf{X}_b is the analysis at the end of the preceding analysis, and \mathbf{B} is the analysis error covariance. If the analysis period is long enough, contribution from the second term is relatively small. Because of the limited scope of this initial paper, the second term will be dropped in any further discussion, and it will be included in future work. For a given model and a given set of observations, D is a function of the model parameters such as initial conditions \mathbf{X}_0 , boundary conditions, and/or extra terms in the model equations representing model errors. For simplicity of discussion, one can view D as a function of \mathbf{X}_0 only; that is, $D = D(\mathbf{X}_0)$. Thus, the task of variational analysis is, for a given set of observations in a certain period $\hat{\mathbf{X}}_i$ and background \mathbf{X}_b , to adjust \mathbf{X}_0 in order to minimize D . (How other model parameters should be handled will be discussed in section 3h.) The model solution corresponding to the \mathbf{X}_0 that gives the minimal D , obtained through a minimization algorithm, is the resulting analysis.

The analysis period is not arbitrary. If, given a perfect model, the analysis period exceeds the range of the atmospheric predictability, the fit will not be very close. When model error is taken into account, the analysis period cannot exceed the range within which the model can produce a good forecast. On the other hand, a very short analysis period (say 1 or 2 h) cannot realize the advantage of variational analysis of being truly four-dimensional. The optimal analysis period is yet to be determined experimentally.

b. Minimization algorithms

Of the different types of contending minimization algorithms (steepest descent method, Newton's method, quasi-Newton methods, and conjugate-gradient methods), the last one is the best compromise in terms of convergence rate and computer memory requirements for problems involving large dimensions. The determination of the gradients of D with respect to all of the adjusting parameters is required in many descent algorithms, and will be described in the next subsection.

To minimize $D(\mathbf{X}_0)$ an initial guess of \mathbf{X}_0 is taken at the beginning of an iterative procedure. At each iteration the previous estimate of \mathbf{X}_0^n is replaced by \mathbf{X}_0^{n+1} :

$$\mathbf{X}_0^{n+1} = \mathbf{X}_0^n + (\rho d)^n,$$

when d is the descent direction and ρ is the step size. In the steepest descent method d is simply the direction opposite to the gradient $\nabla_{\mathbf{X}_0} D$, or D' . The Newton's

method relates ρd to the gradient D' and the Hessian matrix \mathbf{D}'' , which contains the curvature information; that is, $(\rho d)^n = -D'/\mathbf{D}''$. Since Newton's method uses more information it provides a much faster convergence rate than steepest descent. However, this is accomplished at the expense of computing \mathbf{D}'' , which has an unacceptable memory core requirement when the dimension of \mathbf{X} is large, as in the meteorological variational analysis problem. Quasi-Newton methods attempt to approximate the curvature information from the previous iterations. However, they also suffer from the excessive core requirement. The conjugate-gradient methods also approximate \mathbf{D}'' from the D 's of the last few iterations but avoid the core requirement problem by not having to store a matrix. The memory requirement is on the order of $10n$, n being the dimension of \mathbf{X} , a very large but manageable requirement for the forthcoming supercomputers.

Various types of conjugate-gradient method are available. Navon and Legler (1987) have reviewed and evaluated many of them and have recommended the CONMIN method (Shanno 1978; Shanno and Phua 1980) for meteorological applications. Moreover, Navon et al. (1990) have vectorized the CONMIN routine for use on CYBER 205.

Gilbert and Lemarchal (1989) have tested a variable storage quasi-Newton algorithm by Nocedal (1980) and Liu and Nocedal (1989) that uses a limited BFGS update formula. Their code, MIQN3, will also be used in this study.

c. The adjoint method of computing $\nabla_{\mathbf{X}_0} D$

In the following description the adjusting parameters of the model are assumed to be the initial condition.

The adjoint of a linear operator \mathbf{A} , \mathbf{A}^* , is defined by $\langle u, \mathbf{A}v \rangle = \langle \mathbf{A}^*u, v \rangle$, where u and v are elements in a Hilbert space, with the inner product denoted by the angle bracket. In discretized problems, u and v are vectors and \mathbf{A} is a matrix, with \mathbf{A}^* being its transpose. Two simple properties of the adjoint operator, $\mathbf{I}^* = \mathbf{I}$ and $(\mathbf{AB})^* = \mathbf{B}^*\mathbf{A}^*$, are obvious. With these basics stated, one can consider the variation of the distance function D ,

$$\delta D = \sum_{i=0}^m \langle \mathbf{W}(\mathbf{X}_i - \hat{\mathbf{X}}_i), \delta \mathbf{X}_i \rangle,$$

where \mathbf{X}_i is solution to the model equation $\partial \mathbf{X} / \partial t = \mathbf{F}(\mathbf{X})$. Here, for simplicity, we have assumed that observation is available at all model grid points. Also, we have neglected the contribution to D from the distance to \mathbf{X}_b . The perturbation of the model state $\delta \mathbf{X}$, is governed by the tangent linear equation

$$\frac{\partial \delta \mathbf{X}}{\partial t} = \mathbf{F}'(t) \delta \mathbf{X},$$

where \mathbf{F}' is the derivative of \mathbf{F} with respect to \mathbf{X} , eval-

uated at \mathbf{X} . Using the Euler time scheme as an example, one can have

$$\begin{aligned}\delta\mathbf{X}_i &= (I + \Delta t F'_{i-1})\delta\mathbf{X}_{i-1} \\ &= (I + \Delta t F'_{i-1})(I + \Delta t F'_{i-2}) \cdots (I + \Delta t F'_0)\delta\mathbf{X}_0.\end{aligned}$$

Thus,

$$\delta D = \left\langle \sum_{i=1}^m (I + \Delta t F'_0^*)(I + \Delta t F'_1^*) \cdots (I + \Delta t F'_{i-1}^*) W(\mathbf{X}_i - \hat{\mathbf{X}}_i), \delta\mathbf{X}_0 \right\rangle.$$

The summation in this expression is the gradient $\nabla_{\mathbf{x}_0} D$. The i th term in this series is obtained by a backward finite-difference integration of the adjoint equation

$$-\frac{\partial \delta' \mathbf{X}}{\partial t} = F'^*(t) \delta' \mathbf{X}$$

from time step i to the initial step starting from

$$\delta' \mathbf{X} = W(\mathbf{X}_i - \hat{\mathbf{X}}_i).$$

Thus, the procedures to obtain $\nabla_{\mathbf{x}_0} D$ for a given \mathbf{X}_0 are as follows:

1) Integrate the model from time t_0 to t_m , starting from \mathbf{X}_0 , and store the solution at each time step

$$\mathbf{X}_i \quad (i = 0, 1, \dots, m).$$

2) Integrate the adjoint model

$$-\frac{\partial}{\partial t} \delta' \mathbf{X} = F'^*(t) \delta' \mathbf{X}$$

backward in time from t_m to t_0 , starting from $\delta' \mathbf{X} = W(\mathbf{X}_m - \hat{\mathbf{X}}_m)$, and at each time step t_i add to $\delta' \mathbf{X}$ the quantity $W(\mathbf{X}_i - \hat{\mathbf{X}}_i)$. The resulting $\delta' \mathbf{X}$ at t_0 is the gradient $\nabla_{\mathbf{x}_0} D$.

Examples of deriving F'^* have been given by Talagrand and Courtier (1987) and Talagrand (1989). An alternative derivation of the adjoint equation is through the Lagrange multipliers method (Thacker and Long 1988). However, as will be shown in the next section, the adjoint-model code can be constructed directly from the forward-model code in a straightforward way, which is also conducive to easy error detection. Thus, the lengthy derivation of the adjoint model either in the differential form or in the discrete form is not necessary.

d. Suppression of gravity waves

In the variational analysis, since the model uses all of its degrees of freedom to fit data, unrealistically large gravity-wave modes can exist in the resulting analysis. Courtier and Talagrand (1990) suggested adding a penalty term proportional to $\|\partial G/\partial t\|$ to the distance

function as a solution where G denotes the collective amplitude of a set of gravity-wave modes. This combined with a nonlinear normal-mode initialization procedure on the initial condition at the beginning of each iteration gives a satisfactory resolution of the gravity-wave problem. We will defer the treatment of gravity waves to future work.

e. Adjoint of the physical processes

Writing the adjoint of heating and friction terms in the model equations can be very involved. These forcing terms, as functions of thermodynamic variables, can be discontinuous in themselves or in their first- and second-order derivatives. The difficulty such discontinuities can present and its resolution will be a subject of future work.

3. Details of the development

a. The forecast model

The forecast model used is the GLA fourth-order GCM (Kalnay et al. 1983), with substantial revisions by Suarez and Takacs of GLA. The dynamics part of the model contains polar filter, Shapiro filter, A-grid, fourth-order horizontal finite-differencing scheme, and the Arakawa-Suarez (1981) vertical finite-differencing scheme. Potential temperature, instead of temperature, is used as a variable. Time integration is done with an initial Matsuno time step followed by leapfrog time steps. The Asselin time filter is used to control time splitting. Matsuno scheme for all time steps is an option, but it is not used in this work because it requires more computer time. In order to reduce cost at the development stage, we use a low-resolution version of this model with three layers and a 12° latitude \times 10° longitude horizontal resolution. A time step of 7.5 min is used (though a larger time step is allowed). Also, we have conducted a linear test (Chao and Geller 1982) on the dynamics part of the model. The test rendered expected results and thus indicated that the linear terms in the model dynamics are correct. Nonlinear terms were checked by hand calculation. A brief description of the model physics is postponed for a future paper. Hereafter, the terms forecast model, GCM, and forward model are used interchangeably.

b. The adjoint model

The adjoint model can be constructed directly from the forward-model code. This is achieved by first writing the tangent linear model directly from the forward-model code. For example, a typical Fortran assignment statement in the forward-model code is

$$W = AW + BXY,$$

where A and B are constants and W , X , and Y are variables. The corresponding Fortran statement in the tangent linear code is

$$W = AW + B(FXY + XFY),$$

where FX and FY are the values of X and Y recorded during the forward run (Not all forward run quantities can be recorded, because of their volume; thus, recalculation of some of them, when needed, is necessary. In our code only the prognostic variables themselves are recorded at each time step. All other forward quantities are recalculated.), and X , Y , and W now represent the perturbation quantities. Once the tangent linear code is written, one can obtain the adjoint code by writing the adjoint of each statement of the tangent linear code starting from the last tangent linear statement and proceeding backward. The basic principle is to treat each statement in the tangent linear code as a linear operator, and the entire tangent linear code as a series of linear operators. A typical statement in the tangent linear code has the form of

$$W = AX + BY + CZ,$$

where X , Y , Z , and W are the perturbation variables, and A , B , and C are either constants or variables related to the forward run results. This statement can be considered as a linear operation of the form $\mathbf{O} = \mathbf{M}\mathbf{I}$, with $\mathbf{I} = (X, Y, Z)^T$, and $\mathbf{O} = (W, X, Y, Z)^T$, where T denotes transpose, that is,

$$\begin{pmatrix} W \\ X \\ Y \\ Z \end{pmatrix} = \begin{pmatrix} A & B & C \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix},$$

and the adjoint of this statement is

$$\begin{pmatrix} X' \\ Y' \\ Z' \end{pmatrix} = \begin{pmatrix} A & 1 & 0 & 0 \\ B & 0 & 1 & 0 \\ C & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} W' \\ X' \\ Y' \\ Z' \end{pmatrix},$$

where X' , Y' , Z' , and W' represent the gradients of D with respect to the forward model variables X , Y , Z , and W , respectively. This example leads to the general rule that the adjoint code corresponding to a tangent linear statement:

$$W = \sum_{i=1}^N A_i X_i$$

(assuming W does not appear on the right hand side) is the N statements (dropping the primes):

$$X_i = X_i + A_i W, \quad i = 1, N.$$

Following these N statements it is necessary to set W equal to zero, if W appears on the right-hand side of preceding statements in the tangent linear code. More-

over, if W appears on the right-hand side of the tangent linear statement, that is,

$$W = AW + \sum_{i=1}^N B_i X_i,$$

one can treat it as two statements:

$$Z = AW + \sum_{i=1}^N B_i X_i, \quad \text{and}$$

$$W = Z.$$

Following the rule just stated, one obtains the adjoint statements:

$$X_i = X_i + B_i W, \quad \text{for } i = 1, \dots, N, \quad \text{and}$$

$$W = AW.$$

The position of the last statement should not be changed. One can write the corresponding adjoint code directly from each forward-code statement with only a mental note of the corresponding tangent linear code and without actually writing down the entire tangent linear code. The simple rule of writing adjoint code obviously covers the writing of adjoint for any time scheme in a simple manner, since any time scheme when written in code is only a sequence of computations, and they can be handled just like any other group of Fortran statements.

Because of the reverse-order nature of the adjoint code a DO statement, such as DO 100 I = 1, 200 in the forward, is transformed in the adjoint code into DO 100 I = 200, 1, -1. If the order of the index I in the loop is not important in the forward code, it is also unimportant in the adjoint code.

The condition in IF THEN-ELSE statements determines which of the subsequent two groups of statements is to be used. In the case of no ELSE statement existing, the second group is a blank one. This condition should be recorded in the forward integration by recording or recalculating all variables appearing in the IF conditions in the forward integration. If the IF condition involves only indices or independent variables, no recording is required. In the adjoint code the same condition is used to choose, from the two corresponding groups of adjoint statements, the one corresponding to the group that is used in the forward code.

Although the IF THEN statements pose no difficulty as far as writing adjoint code and computing gradient are concerned, they can be a great impediment to the rate of convergence, as will be discussed in section 5. Incidentally, because of the mechanical and tedious nature of writing an adjoint code, developing an automatic adjoint generator is an active research area (Griewank and Corliss 1991).

Our forward code has been written in a flexible manner to allow any number of levels in the vertical direction. This feature has been preserved in the adjoint code.

c. An alternative derivation of rules for writing the adjoint code

The rules presented in the preceding subsection can be arrived at without invoking the concept of adjoint operator. Basic knowledge of calculus suffices. In a Fortran program a typical assignment statement has the form of

$$y = f(x_1, \dots, x_m),$$

where y is a single variable and f is a differentiable function. For simplicity, mathematical expressions have been used to represent the Fortran statement. In this subsection all equal signs have the meaning as used in a Fortran assignment statement. Chain rule gives

$$\partial y = \left(\frac{\partial f}{\partial x_1} \right) \partial x_1 + \dots + \left(\frac{\partial f}{\partial x_m} \right) \partial x_m, \quad (1)$$

(which is the tangent linear statement where the derivatives are calculated with results from the forward run) and

$$\nabla_{x_i} D = \left(\frac{\partial f}{\partial x_i} \right) \nabla_y D, \quad i = 1, \dots, m, \quad (2)$$

where $\nabla_y D$ denotes gradient of D with respect to y and D is a function of the final output of the Fortran code. With this reminder, one can consider two consecutive Fortran statements:

$$\begin{aligned} y &= f(x_1, \dots, x_m) \\ z &= g(y, x_1, \dots, x_m). \end{aligned} \quad (3)$$

Since we are concerned with the gradient of an output quantity with respect to input quantities, we will consider the second statement first. The gradient relationships are

$$\nabla_{x_i} D = \frac{\partial g}{\partial x_i} \nabla_z D, \quad i = 1, \dots, m, \quad (4)$$

$$\nabla_y D = \frac{\partial g}{\partial y} \nabla_z D, \quad (5)$$

and, from the first statement in (3),

$$\nabla_{x_i} D = \frac{\partial f}{\partial x_i} \nabla_y D + \nabla_{x_i} D, \quad i = 1, \dots, m. \quad (6)$$

Equation (6), a Fortran assignment statement, is different from (2) in that an additional term, $\nabla_{x_i} D$, appears on the right-hand side. This additional term evaluated from (4) is necessary because a change in x_i can affect D through change in y in (3) and through change in x_i , which affects g directly. Thus, (2) corresponds to the special case of the consecutive statements:

$$\begin{aligned} y &= f(x_1, \dots, x_m), \quad \text{and} \\ x_i &= 0. \end{aligned}$$

Equation (6) is exactly the basic rule of writing adjoint code given in the preceding subsection when X_i is used to denote $\nabla_{x_i} D$. It is obvious that adjoint variables denote gradients of an output quantity with respect to the model variables.

d. Verification of the adjoint code

The rudimentary equation used for code verification is derived as follows. A model, or a subroutine of it, is an operator (very often nonlinear):

$$y = F(x), \quad \text{where } x \text{ is the input and}$$

y is the output, or in its component form

$$y_j = [F(x_1, x_2, \dots, x_n)]_j.$$

The gradients with respect to the input can be expressed by those with respect to the output:

$$\nabla_{x_i} D = \sum_j \left(\frac{\partial F}{\partial x_i} \right)_j \nabla_{y_j} D,$$

where D is any function of the output y . Since the corresponding adjoint code (or subroutine) is a linear operator linking the gradients with respect to the output array y to those with respect to the input array x , it can be expressed by

$$\nabla_x D = \mathbf{H}(\nabla_y D),$$

where \mathbf{H} is a matrix, or in its component form

$$\nabla_{x_i} D = \sum_j H_{i,j} \nabla_{y_j} D.$$

Therefore,

$$\left(\frac{\partial F}{\partial x_i} \right)_j = H_{i,j}.$$

The correctness of an adjoint subroutine can be verified by checking if the matrix \mathbf{H} and that corresponding to $(\partial F / \partial x_i)_j$ are the same. The procedure for computing \mathbf{H} is to call the adjoint subroutine with all arguments set at zero, except the i th one, which is a set at unity (All input arrays can be equivalenced to one larger array. Thus, we can assume there is only one input array; the same can be done for the output arrays.), and the output array gives the i th column of matrix \mathbf{H} . Matrix $(\partial F / \partial x_i)_j$ can be obtained by the perturbation method. In the perturbation method the i th element of the input array \mathbf{I} of the forward subroutine is perturbed by a small amount $\Delta \mathbf{I}$, and the resulting change in the output array \mathbf{O} , $\Delta \mathbf{O}$, divided by $\Delta \mathbf{I}$, should be close to the i th row of the matrix $(\partial F / \partial x_i)_j$. This array is to be compared with the corresponding row of the adjoint matrix \mathbf{H} . Any doubt as to whether the match is close enough can be removed by repeating the perturbation calculation in double precision, which allows a $\Delta \mathbf{I}$ several (say six) orders of magnitude

smaller. If the match improves greatly, success can be claimed. To reduce computational cost it is advantageous to use a minimal resolution version of the model for this testing purpose. Once the existence of an error has been established, a simple technique can be used to narrow down the area of search, that is, to comment out the first half of the forward subroutine and the corresponding second half of the adjoint code and see if the matrices match. Often by tracing the adjoint statements involved in computing the matrix element in question, one can also narrow down the area of search. After each individual adjoint subroutine has been tested, the entire adjoint code should be tested as a whole by the same procedure. Incidentally, for either the construction or the verification of the adjoint code, there is no need to write the tangent linear code.

e. Distance function

For the experiments described in this paper the distance function is defined as

$$D = \sum_{t=0}^m D_t$$

$$= \sum_{t=0}^m \left\{ \sum_{ijk} [W_u^k (u_{ijkt} - u_{ijkt}^o)^2 + W_v^k (v_{ijkt} - v_{ijkt}^o)^2 + W_T^k (T_{ijkt} - T_{ijkt}^o)^2] + \sum_{ij} W_{ps} (p_{sijt} - p_{sijt}^o)^2 \right\},$$

where \sum_{ijk} and \sum_{ij} are the summations over all observation points, \sum_t is the summation over time, W 's are the weights (we use the same weight for each variable at each level; for our three level model there are ten weights), and superscript o denotes observation. Other notations have their standard meanings. The weight W_x has the same units as those of x^{-2} . Thus, D is non-dimensional. Nonsuperscripted variables are those of the model. In the experiments discussed in this paper, data generated by the forecast model will be used as observation. Thus, observation is available at all grid points (The variables i and j in the definition are the model horizontal indices, and k is the vertical index.) and at all time steps. Interpolation from model grid to observation location is thus avoided. In the final implementation, other terms should be included in the distance function to represent penalty on gravity waves, distance from background, etc. However, they are not included in this initial paper.

f. Minimization procedure

The CONMIN procedure (Shanno 1978a; Shanno and Phua 1980), recommended by Navon and Legler (1987), and MIQN3 from INRIA (Gilbert and Lemairechal 1989) are used. A limited comparison of these two procedures will be presented in section 4.

g. Backward integration of the adjoint model

Since the leapfrog scheme used gives two sets of output at $t = m - 1$ and $t = m$ at the end of the forward run, the adjoint variables are initialized at $t = m$ and $t = m - 1$ with the gradients of $D_m + D_{m-1}$ with respect to model variables at $t = m$ and $t = m - 1$. In these initial conditions, $\nabla_{x_m} D_m$ and $\nabla_{x_{m-1}} D_{m-1}$ are computed directly from the definition of D_m and D_{m-1} , which involve observations and forward-model results, and $\nabla_{x_{m-1}} D_m$ and $\nabla_{x_m} D_{m-1}$ are set at zero. After a single time-step reverse leapfrog integration, one obtains the gradient of $(D_m + D_{m-1})$, with respect to model variables at $t = m - 1$ and $t = m - 2$. If there is observation at $t = m - 2$, its distance gradients with respect to model variables $\nabla_{x_{m-2}} D_{m-2}$ are added to the adjoint variables at $t = m - 2$ before the subsequent reverse leapfrog step is taken. This procedure repeats until one reaches time levels $t = 1$ and 0. Subsequently, an adjoint of the Matsuno step is used, followed by the addition of the gradient of D_0 , with respect to model variables at initial time. The end result is the gradients $\nabla_{x_0} D$ of the distance function with respect to the model initial conditions. The gradients thus computed should be verified against those computed by the perturbation method.

h. Variational analysis with respect to any model parameters

Thus far, the discussion has been restricted to the case that distance function is a function of initial conditions only. The procedure described in section 3b to compute gradients of distance function with respect to initial conditions can also be used to compute gradients with respect to any model parameters with minimal changes. The only change involved is to treat those parameters as variables when writing the adjoint codes. A single backward integration gives the gradients, with respect to both initial conditions, and the chosen model parameters. The minimization can be done in the phase space consisting of both initial conditions and the chosen model parameters. Alternatively, by ignoring the gradients with respect to initial conditions the adjoint model computes, one can do variational analysis with respect to the chosen model parameters only. If the chosen parameters are the additional systematic-error correction terms in the governing equations, one can repeat the work done by Derber (1989) with a GCM. Incidentally, it should be pointed out that the material in this subsection will not be used in the rest of this paper, but will be employed in future work.

i. Weights and preconditioning scales setting for model-generated "observation"

When real data are used, the weights should be the inverse of rms sum of measurement error and representation error. However, in the tests described in the

next section, the model-generated data are used as "observations." These observations have no error, and the normal method would render infinitely large weight, which implies infinite confidence, but can not be used. If the weights meant for real data are used and if no preconditioning scales (to be explained shortly) are used, in the first experiment we did (as described in the next section) the surface-pressure component of the distance function failed to diminish after large number of iterations and the convergence among the other variable types are not uniform. In the second experiment, convergence was achieved only after large number of iterations and when the rate of convergence was highly nonuniform among different variable types. The reason for such behavior lies in the fact that the constant distance surface is highly stretched in some dimensions. The upper panel of Fig. 1 gives such a sketch of stretched structure. In the first experiment the constant distance surfaces are so stretched in the surface-pressure direction that the gradient computed is practically perpendicular to surface pressure dimension. Thus, reduction in the surface-pressure component of the distance function cannot be achieved. The second experiment encountered the same problem,

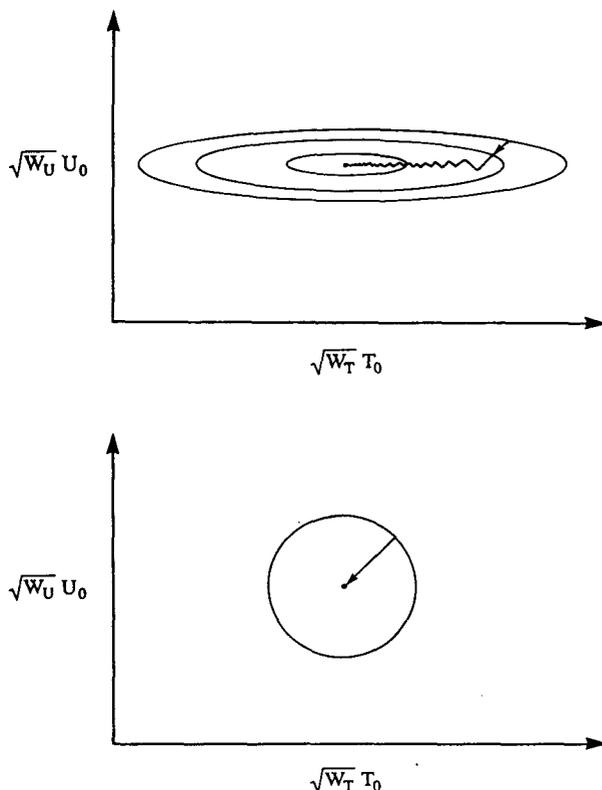


FIG. 1. (a) Schematic diagram showing the constant D surfaces in the nondimensional X_0 space when weights meant for the real data are used. (b) Schematic diagram showing the constant D surfaces in the nondimensional X_0 space when the optimal weights are used.

though to a lesser degree. Therefore, a different avenue must be sought.

Thus, we will take advantage of the freedom to choose the weights and the choice of setting of the preconditioning scales to enhance the rate of convergence and to ensure that the latter is uniform among different variables. The weights are set in a way that capitalizes a simple geometric concept. In the initial condition phase space the gradient G is of course perpendicular to the contour surface of D . The rate of convergence in the iterative procedure can be optimized if G points toward the observation point (Fig. 1). Our definition of D gives the constant D surface roughly an elliptic shape in the initial condition phase space. The convergence rate can be optimal if the elliptic shape can be transformed into a more spherical one. This can also result in a more uniform rate of convergence among the variable types. This concept forms the basis of our approach in determining the weights. This concept is well known and is linked to the condition number of the Hessian matrix, ratio of the largest eigenvalue of the Hessian matrix to the smallest eigenvalue (Gill et al. 1982; Thacker 1988). Instead of considering the dimensional initial condition space, we will consider the nondimensional initial condition space spanned by ten collective dimensions, that is, three variables types u , v , and T at three levels plus surface pressure. The word "collective" means summation over all horizontal grid points, and it will be more clear when the procedure is described in the Appendix. Figure 1 shows, as an example, a simpler initial-condition space of two collective dimensions, u and T . Points o , the center, and s , the starting point of the arrow, denote observation and model solution and G is the gradient of D with respect to u and T . The weights are determined such that line os is parallel to G in an iterative procedure given in the Appendix. The number of iterations required is an increasing function of the number of weights involved. In the experiments, ten weights were involved and eight iterations were sufficient. It is important to emphasize that our method requires that minimization be done in the nondimensional space, though the adjoint method of computing the gradients is still done in the dimensional space. The scales that are used to nondimensionalize the control parameters (in our present work, the initial conditions) are called the preconditioning scales. Our method clearly allows the computation of weights as functions of latitude and weights for more variables, such as mixing ratio.

In principle, the weights can be functions of grid points also. In this case there are as many weights as the number of initial conditions. Once the weights have been determined, convergence can be achieved in one minimization iteration, since the "surface" of equal distance function takes on a spherical shape in the initial condition space. However, little gain in total cost can be expected because the number of iterations involved in computing the weights will increase.

After a few (say, ten) minimization iterations, this procedure can be invoked again to recompute the weights. However, our limited experience indicates that the improvement from recomputing the weights is not always worth the extra cost of recomputing the weights. The final decision to use this restart technique should be based on the particular experiments at hand.

Finally, we like to reiterate that this method of setting weights and preconditioning scales is useful only in our test experiments in which model-generated data are used as observations to be described in the following section. When real data are used one should use weights inversely related to the observational error, and one has only the preconditioning scales to choose. How to set these scales will be discussed in future work.

4. Test experiments

Besides performing separate tests for the code of each individual adjoint subroutine in the model and their combination, the system (including the forecast model, the adjoint model, and the minimization procedure) should be tested as a whole. One simple test is to run the forecast model from any reasonable initial condition for a suitable period (for example one day) and use the resulting model-generated data as observation. Starting from another initial condition, the analysis system is expected to recover the original initial condition through the iterative procedure; that is, the distance function should reduce to zero. Such a test has the distinct advantage over tests using the real data. In the real-data case, reduction of the distance function to zero is not achieved leaving one in doubt as to whether the system is free of error. Incidentally, in this test the suppression of gravity waves is not a concern.

Two such tests were conducted. Analogous to the Rossby-Haurwitz wave test in Talagrand and Courtier (1987), the first test run generates the observation data from a run started from a normal-mode solution of wavenumber 1 of mixed Rossby-gravity mode with isothermal basic state (Chao and Geller 1982). This run was integrated for 6 h, and the results were stored at every time step and used as the observations. The results at hour 3 are used as the initial conditions for the analysis (or iteration) run. Thus, the analysis run started from an initial condition nearly identical to that of the observations run but with a westward phase shift of 38.4° . Figure 2 shows the distance function as a function of iterations. The results with M1QN3 (with $m = 10$, where m is a parameter related to the size of working array) give more than two orders of magnitude of reduction in the distance function in 14 iterations. Also, fairly uniform convergence was achieved among the variable types. The rms surface-pressure error reduced from 0.41 to 0.020 mb in 14 iterations, and that of the second-level u wind reduced from 0.16 to 0.004 m s^{-1} . This result indicates that our method of weights determination is nearly optimal. The same test using CONMIN gives a slower rate of convergence, it took two

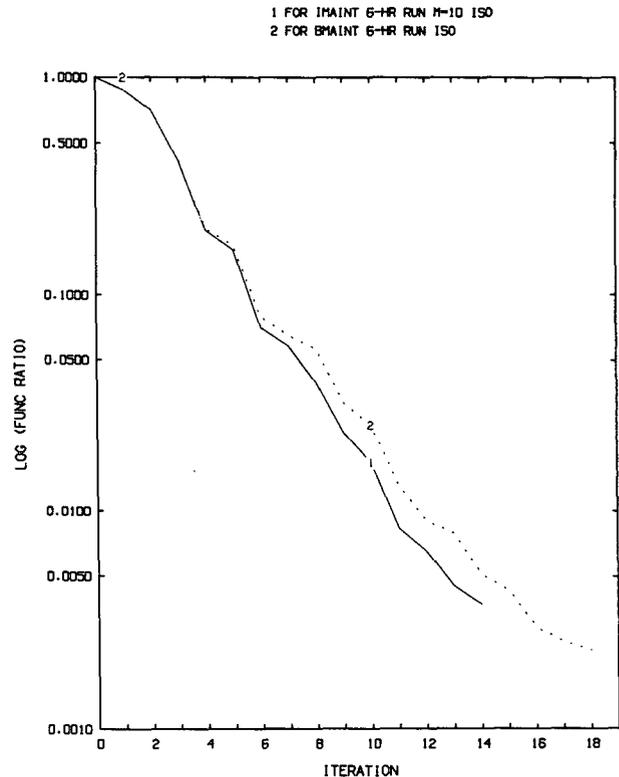


FIG. 2. Distance function as a function of iterations for the first test using M1QN3 with $m = 10$ (line 1) and CONMIN (line 2).

more iterations to achieve the same amount of reduction in the distance function. M1QN3 uses more terms in the approximation to \mathbf{D}'' than the conjugate-gradient method in CONMIN.¹

In the second test, which is closer to the real situation, the model was run for 6 h starting from hour 0 of 4 January 1979 of the European Centre for Medium-Range Weather Forecasts (ECMWF) analysis, and the results were stored as observation. Iteration run started from hour 0 of 24 January 1979. Figure 3 shows the distance function as a function of iteration for the two minimization codes. Both achieved a similar amount of reduction in distance function. The rms error of surface pressure reduced from 27 to 1.4 mb, and that of the second level u reduced from 13.7 to 0.64 m s^{-1} after 30 iterations. CONMIN took almost twice as many gradient computations (each involves the integration of both forward and adjoint models) as M1QN3 did, making the latter a much better choice. However, M1QN3 uses more memory. Table 1 gives a summary of the comparison of the performance of the two minimization codes. Further iterations with CONMIN, Fig. 4, reduced the surface-pressure rms difference to less

¹ The optional quasi-Newton method in CONMIN can not be used because of the excessive memory requirement.

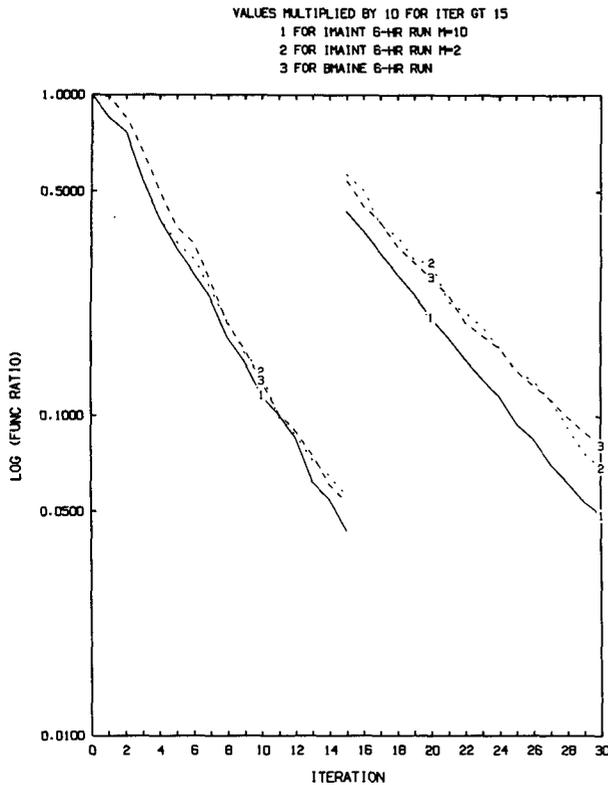


FIG. 3. Distance function normalized by its initial value as a function of iterations for the second test using MIQN3 with $m = 10$ (line 1) and $m = 2$ (line 2) and CONMIN (line 3).

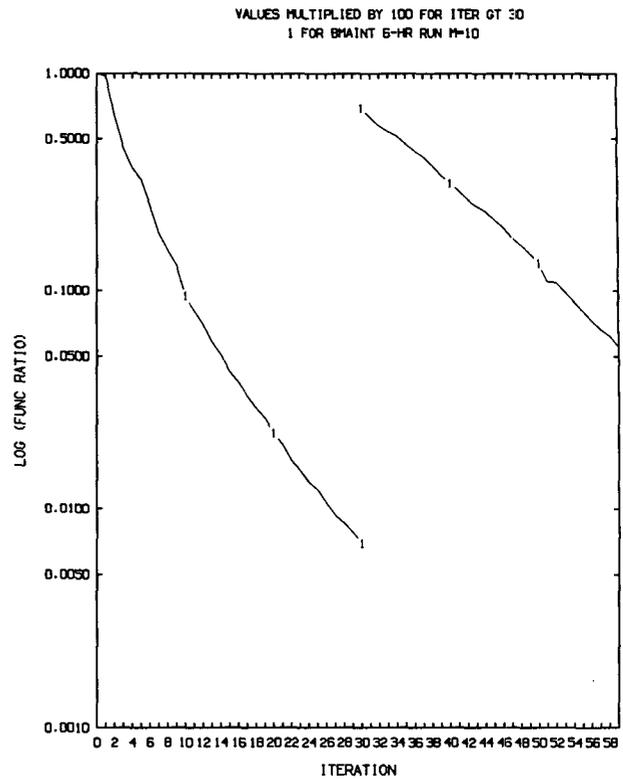


FIG. 4. Same as Fig. 3 with CONMIN only for 60 iterations.

than 0.5 mb after 60 iterations. These results indicated that our development effort, thus far, has been successful. We have also conducted the same experiments using weights as a function of variable types but not of levels. As expected, the results were much inferior in terms of rate of convergence.

5. Rate of convergence and further experiments

Our experiments show that the rate of convergence for a 6-h analysis period is much slower than that for a 2-h period (not shown). The reason for this (c.f., Derber 1987) lies in the shape of constant D surfaces in the X_0 space, as mentioned in section 3i. The more

spherical these surfaces are the faster the rate of convergence is. In the extreme case of a zero analysis period (i.e., $m = 0$ in section 3e), surfaces of constant D have the shape of a smooth hyperellipsoid, and through adjustment of weights or control parameters they can be made spherical. As the analysis period increases, the surfaces of constant D can stretch in many dimensions, resulting in a slower rate of convergence. This concept also explains why the rate of convergence in our first experiment is faster than that in the second experiment (Figs. 2 and 3). The first experiment has relatively slower evolution.

We have repeated the experiment shown in Fig. 3 after changing the Shapiro filter used in the model from sixteenth order to eighth order. The results showed little change in the rate of convergence. This merely indicates

TABLE 1. Results after 30 iterations.

	CONMIN	MIQN3, $m = 2$	MIQN3, $m = 10$	Initial value
Ratio of final distance function to initial distance function	0.0071	0.0069	0.0046	1
Root-mean-square error in surface pressure (mb)	1.98	1.84	1.42	27
Root-mean-square error in second-level u wind ($m s^{-1}$)	0.77	0.77	0.64	13.7
Root-mean-square error in second-level θ (K)	0.18	0.16	0.13	3.9
Number of function calls*	62	33	33	

* Each function call involves a forward run with the GCM and a backward run with the adjoint model.

that the difference between the eighth- and the sixteenth-order Shapiro filter is negligible. However, our experiments revealed a great sensitivity of the rate of convergence with respect to observation interval. Figure 5 shows the rate of convergence for observation intervals of 7.5 min, 1 h, 3 h, and 6 h in experiments otherwise identical to that associated with Fig. 3 (using CONMIN), with weights and scales kept the same. The rate of convergence is higher for lower frequency of observation used and is the best for 6-h interval, that is, when only observations at hour 0 and hour 6 are used. Another interesting feature in Fig. 5 is that such sensitivity to the observational interval is not very prominent during the initial iterations.

These findings can be again explained by considering the shape of the distance function in the scaled initial-condition space. But we will first consider the quantity D_t , as defined in section 3e, in the scaled initial-condition space. Contours of D_t are nearly spherical for $t = 0$ and depart from being spherical as t increases mainly through the linear process of stretching in certain dimensions in the initial two days (Lacarra and Talagrand 1988). As illustrated by the solid curve in Fig. 6, the degree of departure (DD) of D_t contour surfaces from a perfect sphere, increases as t increases, and saturates at a certain value. Mathematically, DD can be equated to the condition number of the Hessian associated with D_t in the scaled initial-condition space.

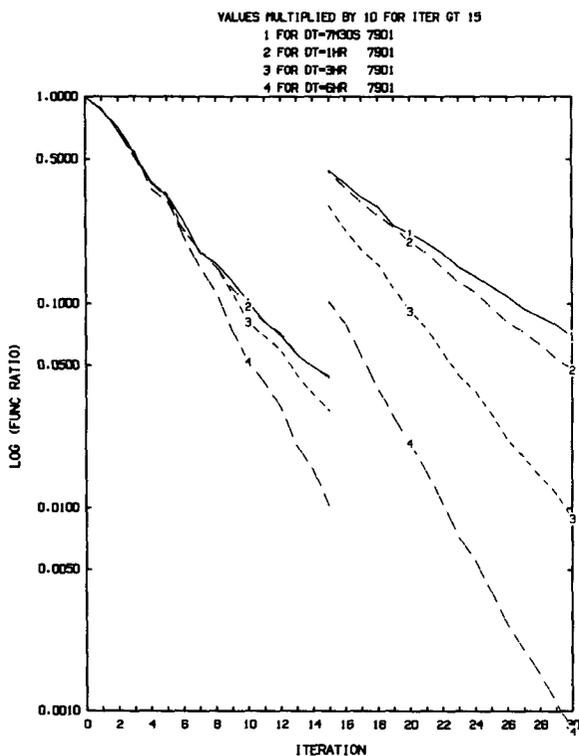


FIG. 5. Same as Fig. 3 (using CONMIN), but with observation interval set at $\Delta t = 7.5$ min, 1 h, 3 h, and 6 h.

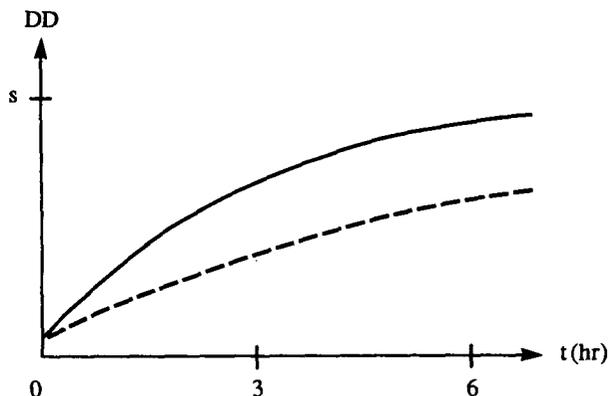


FIG. 6. Schematic diagram showing the departure from spherical shape of the contour surfaces of D_t , as defined in section 3e, in the scaled initial-condition space as a function of t . The solid curve is for the later iterations, and the dash curve is for the initial iterations.

The DD of the distance function in the scaled initial-condition space, the deciding factor of the rate of convergence, is the average of DD of the D_t 's used. Since $DD_{\text{hour } 3}$ is greater than $(DD_{\text{hour } 0} + DD_{\text{hour } 6})/2$, DD of the distance function for $\Delta t = 3$ h is greater than that for $\Delta t = 6$ h [i.e., $(DD_{\text{hour } 0} + DD_{\text{hour } 3} + DD_{\text{hour } 6})/3 > (DD_{\text{hour } 0} + DD_{\text{hour } 6})/2$]. Therefore, it is obvious that the rate of convergence is less for $\Delta t = 3$ h than for $\Delta t = 6$ h. Further halving Δt has a smaller impact on the rate of convergence. Thus, the sensitivity of the rate of convergence to observation interval hinges on the saturation of DD in Fig. 6. The reason that this sensitivity is not very prominent in the initial iterations (Fig. 5) lies in the fact that for the initial iterations the initial condition is far away from hour 0 observation and DD takes a longer time to saturate (the dashed curve in Fig. 6). Here $DD_{\text{hour } 3}$ is still greater than $(DD_{\text{hour } 0} + DD_{\text{hour } 6})/2$, but not by much.

6. Summary and future work

In summary, we have completed writing the dynamics part of a variational analysis system based on the GLA GCM using the adjoint method. We devised a simple error-detection procedure, which greatly facilitated the construction of the adjoint code. A method of determining weights in the distance function and preconditioning scales when model-generated data are used is presented, which gives a better rate of convergence and a more uniform convergence among different variable types. Two test runs, one with a normal-mode solution and the other with realistic initial conditions, have shown that our work, thus far, has been successful. Our experiments showed a greater rate of convergence for shorter analysis period and longer observation interval. These results have been explained by considering the shape of distance function contour surfaces in the scaled initial-condition space.

The remaining work to complete the development

is sizeable. The immediate future work involves the handling of the real data, the setting of preconditioning scales for the real data, incorporation of a quality control procedure, suppression of gravity waves, and incorporation of the background term in the distance function. Further work with the full model is in progress. There are two directions under consideration for the purpose of reducing the computational requirement of variational analysis. One is to use multiple resolution systems; the first few iterations that fit mostly the larger-scale features can be done with a coarser resolution. The second direction is the use of a semi-Lagrangian scheme that allows a larger time step. For the purpose of reducing the storage requirement, periodic storing of the forward run results, instead of at every time step, will be studied. The comparison of two minimization codes (M1QN3 and CONMIN with the former showing better performance) in this paper is very limited. Further comparison will be conducted with a full model and with higher spatial resolution.

Besides stretching, which is a linear phenomenon, constant D surfaces can become irregular through nonlinear evolution of the model (Fig. 7), resulting in slowing down of the rate of convergence. This is a particular concern when physical processes are included that have high nonlinearity. In more extreme cases, some IF statements may allow discontinuous results across the IF condition thresholds resulting in discontinuous surfaces of constant distance function. The model should be revised to remove these discontinuities

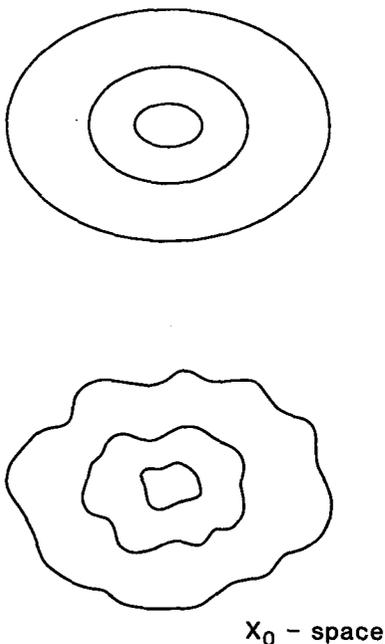


FIG. 7. Schematic diagram showing (a) the smoothness of the constant D surfaces in the X_0 space when the analysis period is short and (b) the unevenness after a period of nonlinear evolution.

as much as possible. Discontinuities in first and second derivatives across the IF condition threshold should also be removed as much as possible to improve the rate of convergence. Such a task naturally requires a thorough knowledge of the model design, and will be an important part of our future work.

Acknowledgments. The first author has benefited from discussions with O. Talagrand of LMD and J. Derber of NMC with regard to the adjoint method. I. M. Navon of FSU is acknowledged for providing minimization codes CONMIN. J. Charles Gilbert of INRIA is acknowledged for providing minimization code M1QN3 of the Modulopt Library. The new version of the GLA GCM was provided by M. Suarez and L. Takacs of GLA. Roger Daley made helpful comments on the manuscript. Financial support organized by J. R. Bates made this work possible.

APPENDIX

Procedure for Computing Weights and Preconditioning Scales When Model-generated Data Are Used

The procedure in the following is only for the cases where model-generated data are used as observations. When real data are used, weights should be related to observation error, and one has only the preconditioning scales to choose, which will be discussed in future work.

Starting from arbitrary weights W_u , W_T , and W_{p_s} , the steps are as follows:

1) Compute the ratio of contributions to distance function by integrating the forward model:

$$W_T \sum (T - T^o)^2 : W_u \sum (u - u^o)^2 :$$

$$W_{p_s} \sum (p_s - p_s^o)^2 \equiv \alpha^2 : \beta^2 : \gamma^2,$$

where the summation is done over three-dimensional grid points (two-dimensional for p_s) and over all time levels.

2) Compute the gradients of D with respect to initial conditions by integrating the adjoint model and computing the magnitude of collective components:

$$|\nabla_T D|, |\nabla_u D|, |\nabla_{p_s} D|.$$

3) Multiply ∇D 's by $(W_T)^{-1/2}$, $(W_u)^{-1/2}$, $(W_{p_s})^{-1/2}$ and compute ratios

$$|\nabla_{W_T^{1/2} T} D| : |\nabla_{W_u^{1/2} U} D| : |\nabla_{W_{p_s}^{1/2} P} D| \equiv A : B : C.$$

4) Multiply W_T , W_u , and W_{p_s} by A/α , B/β , and C/γ to get new W 's.

5) Keep repeating steps 1–4 until the change in W 's is no longer significant.

The purpose of step 4 is to make the gradient vector in the collective nondimensional space point toward

the observation; that is, $\alpha:\beta:\gamma = A:B:C$. Since changing weights changes the definition of D , the procedure should be iterated.

Extension to the case where W is a function of vertical level is straightforward. With weights thus determined, the minimization procedure should be done in the nondimensional space. In other words, we scale the variables with scales $W_u^{-1/2}$, $W_T^{-1/2}$, and $W_{p_s}^{-1/2}$ for a preconditioning purpose. However, since the forecast and adjoint models are done in the dimensional space, in the minimization code the variables should be converted to dimensional quantities before computing the gradient $\nabla_x D$. After the gradient computation, X and $\nabla_x D$ are both converted to nondimensional variables by dividing and multiplying, respectively, by $W_x^{1/2}$. Although three weights are used in this description, extension to allow weights as functions of vertical level and weights for more variables is straightforward. In the experiments reported in section 4, ten weights are used, one for each of the variables u , v , and T , at each of the three levels and one for p_s .

REFERENCES

- Arakawa, A., and M. J. Suarez, 1983: Vertical differencing of the primitive equations in sigma coordinates. *Mon. Wea. Rev.*, **111**, 34–45.
- Chao, W. C., and M. A. Geller, 1982: Utilization of normal mode initial conditions for detecting errors in the dynamics part of primitive equation global models. *Mon. Wea. Rev.*, **110**, 304–306. (corrigendum, **110**, 1324.)
- Courtier, P., and O. Talagrand, 1987: Variational assimilation of meteorological observations with the adjoint vorticity equation. II: Numerical results. *Quart. J. Roy. Meteor. Soc.*, **113**, 1329–1347.
- , and —, 1990: Variational assimilation of meteorological observations with the direct and adjoint shallow water equations. *Tellus*, **42A**, 531–549.
- Derber, J. C., 1987: Variational four-dimensional analysis using quasi-geostrophic constraints. *Mon. Wea. Rev.*, **115**, 998–1008.
- , 1989: A variational continuous assimilation technique. *Mon. Wea. Rev.*, **117**, 2437–2446.
- Gilbert, J. C., and C. Lemarechal, 1989: Some numerical experiments with variable-storage quasi-Newton algorithms. *Math. Prog.*, **45**, 407–435.
- Gill, P. E., W. Murray, and M. H. Wright, 1982: *Practical Optimization*. Academic Press, 401 pp.
- Griewank, A., and G. Corliss, 1991: Automatic differentiation of algorithms: Theory, implementation, and application. *SIAM Proc. Appl. Math.*, **53**, 352 pp.
- Hoffman, R. N., 1986: A four-dimensional analysis exactly satisfying equations of motion. *Mon. Wea. Rev.*, **114**, 388–397.
- Kalnay, E., R. Balgobind, W. Chao, D. Edelmann, J. Pfaendner, L. Takacs, and K. Takano, 1983: Documentation of the GLAS fourth-order general circulation model. Vol. 1: Model Documentation. NASA Tech. Memo 86064. [Available from NASA/GSFC, Greenbelt, MD 20771.]
- Lacarra, J.-F., and O. Talagrand, 1988: Short-range evolution of small perturbations in a barotropic model. *Tellus*, **40A**, 81–95.
- Le Dimet, F. X., and O. Talagrand, 1986: Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects. *Tellus*, **38A**, 97–110.
- Legler, D. M., I. M. Navon, and J. O'Brien, 1989: Objective analysis of pseudostress over the Indian Ocean using a direct minimization approach. *Mon. Wea. Rev.*, **117**, 709–720.
- Lewis, J. M., and J. C. Derber, 1985: The use of adjoint equations to solve a variational adjustment problem with advective constraints. *Tellus*, **37A**, 309–322.
- Liu, D. C., and J. Nocedal, 1989: On the limited memory BFGS method for large scale optimization. *Math. Prog.*, **45**, 503–528.
- Lorenc, A. C., 1981: A global three-dimensional multivariate statistical interpolation scheme. *Mon. Wea. Rev.*, **109**, 701–721.
- , 1988a: Optimal nonlinear objective analysis. *Quart. J. Roy. Meteor. Soc.*, **114**, 205–240.
- , 1988b: A practical approximation to optimal four dimensional objection analysis. *Mon. Wea. Rev.*, **116**, 730–745.
- Navon, I. M., and D. M. Legler, 1987: Conjugate-gradient methods for large-scale minimization in meteorology. *Mon. Wea. Rev.*, **115**, 1479–1502.
- , P. K. H. Phua, and M. Ramamurthy, 1990: Vectorization of the conjugate-gradient method for large scale minimization in meteorology. *J. Optimization Theory Appl.*, **66**, 71–93.
- , X. Zou, J. Derber, and J. Sela, 1992: Variational data assimilation with an adiabatic version of the NMC spectral model. *Mon. Wea. Rev.*, **120**, 1433–1446.
- Nocedal, J., 1980: Updating quasi-Newton matrices with Limited storage. *Math. Comput.*, **35/15**, 773–782.
- Parrish, D., 1988: The introduction of Hough functions into optimal interpolation. *Eighth Conf. on NWP*, Baltimore, MD, Amer. Meteor. Soc., 191–196.
- Shanno, D. F., 1978: Conjugate-gradient methods with inexact searches. *Math. Oper. Res.*, **3**, 244–256.
- , and P. K. H. Phua, 1980: Remark on algorithm 500: A variable method subroutine for unconstrained nonlinear minimization. *AGM Trans. Math. Software*, **6**, 618–622.
- Talagrand, O., 1989: Four-dimensional variational assimilation. *Proc., Seminar on Data Assimilation and the Use of Satellite Data*. ECMWF 1988, Vol. 2, 1–30. [Available from ECMWF, Shinfield Park, Reading/Berk. RG2 9AX, U.K.]
- , and P. Courtier, 1987: Variational assimilation of meteorological observations with the adjoint vorticity equation. I: Theory. *Quart. J. Roy. Meteor. Soc.*, **113**, 1311–1328.
- Temperton, C., and D. L. Williamson, 1981: Normal mode initialization for a multilevel grid-point model. Part I: Linear aspects. *Mon. Wea. Rev.*, **109**, 729–743.
- Thacker, W. C., 1986: Relationship between statistical and deterministic methods of data assimilation. *Variational Methods in Geosciences*, Y. K. Sasaki, Ed., Elsevier, 173–179.
- , 1988: Three lectures on fitting numerical models to observations. Report No. GKSS-Forschungszentrum Geesthacht GmbH Heethacht. [Available from Atlantic Oceanographic and Meteorology Laboratory, 4301 Rickenbacker Causeway, Miami, FL 33149.]
- , and R. B. Long, 1988: Fitting dynamics to data. *J. Geophys. Res.*, **93**, 1227–1240.