Documentation of the Tangent Linear Model and Its Adjoint of the Adiabatic Version of the NASA GEOS-1 C-Grid GCM (Version 5.2)

Weiyu Yang*

and

I. Michael Navon†

* Supercomputer Computations Research Institute
Florida State University

†Department of Mathematics
and
Supercomputer Computations Research Institute
Florida State University

June 1, 1995

Abstract

A detailed description of the development of the tangent linear model (TLM) and its adjoint model of the adiabatic version of NASA GEOS-1 C-Grid GCM(Version 5.2) is presented. The derivations of and the methods for coding the TLM and its adjoint as well as the notation conventions used in these two models are described in detail. The flow charts of the NASA GEOS-1 GCM, its tangent linear model and adjoint model are provided. The procedures and their results of correctness verification of the TLM and the adjoint model are presented. Finally, tutorial examples of derivation of adjoint code from the tangent linear code are provided for the benefit of various users.

Contents

List of Figures

List of Tables

1 Introduction

The GEOS-1 C-Grid GCM was developed by Data Assimilation Office (DAO) at Goddard Laboratory for Atmosphere (GLA), NASA to be used in conjunction with an analysis scheme to produce a multi-year global atmospheric data set for climate research (Schubert et al., 1993). It has also been used to produce multiple 10-year climate simulations as part of the DAO's participation in the Atmospheric Model Intercomparison Project (AMIP) sponsored by the Program for Climate Model Diagnostics and Intercomparison (PCMDI) (see Gates, 1992).

The NASA GEOS-1 C-grid GCM has an advanced structure, i.e., a "plug-compatible" structure. It means that if "plug-compatible" rules are followed in coding different GCMs and parameterizations, codes can be "unplugged" from one model and "plugged" into another with little coding effort. Thus each part of GEOS-1 C-grid GCM can be used independently in another GCM. For instance, full physics package of GEOS-1 C-grid GCM has been used into NASA/GLA Semi-Lagrangian Semi-Implicit (SLSI) GCM. Having developed the tangent linear model(TLM) and adjoint model of the NASA GEOS-1 C-grid GCM contributes to various applications involving other 4-D variational data assimilation systems with different GCMs.

The earliest predecessor of the GEOS-1 C-Grid GCM was developed in 1989 based on "plug-compatible" concepts outlined in Kalnay et al. (1989), and subsequently improved in 1991 (Fox-Rabinovitz, et al., 1991; Helfand et al., 1991). The plug-compatibility of physical parameterizations together with plug-compatible concept of "Dynamical Core" introduced by Suarez and Takacs (1994) facilitated development and testing of new algorithms. Together DAO and Climate and Radiation Branch at GLA, NASA have produced a library of physical parameterizations and dynamical algorithms which may be utilized for various GCM applications.

In order to obtain a 4-D variational data assimilation system based on the NASA GEOS-1 C-grid GCM, a first prerequisite is to develop the tangent linear and its adjoint model, two key parts of any four dimensional variational assimilation system. With the tangent linear and its adjoint model, the 4-D variational data assimilation (VDA) system of the adiabatic version of the NASA GEOS-1 GCM was employed successfully to carry out a series of 4-D VDA experiments to research the Hessian precondition methods and to test a new proposed Hessian estimation algorithm (Yang, et al, 1995). This document describes the development of the tangent linear model and its adjoint model of the adiabatic version of NASA GEOS-1 C-Grid GCM.

In Section 2 we provide a condensed description of NASA GEOS-1 GCM, which includes the basic original atmospheric dynamical equations and their discrete forms, the discretization methods as well as the model structure. Then we present a detailed flow chart of its code, which should prove to be useful to first time users. Section 3 describes and documents in detail the derivation of and coding the tangent linear model (TLM) of the GEOS-1 GCM, its flow chart and its correctness verification against the full nonlinear forward code, as well as the notation conventions used in the TLM. Section 4 describes in detail the derivation of the adjoint model code (for the adiabatic version of the GEOS-1 GCM). A flow chart of the adjoint code is provided along with notation conventions and adjoint model correctness verification procedures. Finally,tutorial examples of derivation of adjoint code from the tangent linear code are provided for the benefit of various users.

2 Description of the NASA GEOS-1 GCM

2.1 Basic original atmospheric dynamical equations

In NASA GEOS-1 C-Grid GCM, a σ vertical coordinate is defined by

$$\sigma = \frac{p - p_T}{\pi} \tag{1}$$

where $\pi \equiv p_s - p_T$, p_s is the surface pressure and p_T is a constant prescribed pressure at the top of the model atmosphere. In the current NASA GEOS-1 GCM version, p_T=0.

The basic original atmospheric dynamics equations of the NASA GEOS-1 GCM are as follows(for obtaining the adiabatic version, just need to delete the terms related to diabatic processes). The continuity equation is

$$\frac{\partial \pi}{\partial t} = -\nabla_\sigma \cdot (\pi \mathbf{V}) - \frac{\partial(\pi \dot{\sigma})}{\partial \sigma} \tag{2}$$

where $\mathbf{V}$ is the horizontal velocity vector. The state equation is

$$\alpha = \frac{c_p \theta}{\sigma} \left(\frac{\partial P}{\partial \pi} \right)_\sigma \tag{3}$$

where α is the specific density, $\theta \equiv T/P$ is the potential temperature, T is the temperature, $P \equiv (p/p_0)^\kappa$, $\kappa = R/c_p$, R is the gas constant, c_p is the specific heat at constant pressure, and p_0 is a reference pressure which be taken as $p_0 = 1000\ hPa$.

The hydrostatic equation is

$$\frac{\partial \Phi}{\partial P} = -c_p \theta \tag{4}$$

where Φ is the geopotential.

The thermodynamic equation is written in flux form to facilitate the derivation of a θ-conserving differencing scheme:

$$\frac{\partial(\pi\theta)}{\partial t} = -\nabla_\sigma \cdot (\pi \mathbf{V} \theta) - \frac{\partial(\pi \dot{\sigma} \theta)}{\partial \sigma} + \frac{\pi \mathbf{Q}}{c_p P} \tag{5}$$

where $\mathbf{Q}$ is the diabatic heating per unit mass.

The equations of tendencies of an arbitrary number of atmospheric constituents, such as water vapor and ozone, are also written in flux form:

$$\frac{\partial\left(\pi q^{(k)}\right)}{\partial t} = -\nabla_\sigma \cdot \left(\pi \mathbf{V} q^{(k)}\right) - \frac{\partial\left(\pi \dot{\sigma} q^{(k)}\right)}{\partial \sigma} + \pi S^{(k)} \tag{6}$$

where $q^{(k)}$ is the specific mass of the kth constituent, and $S^{(k)}$ is its source per unit mass of air.

The momentum equation is written in "vector-invariant" form, as in Sadourny (1975) and Arakawa and Lamb (1981), to facilitate derivation of an energy- and enstrophy-conserving differencing scheme.

$$\frac{\partial \mathbf{V}}{\partial t} = -(f+\zeta)\mathbf{k} \times \mathbf{V} - \dot{\sigma}\frac{\partial \mathbf{V}}{\partial \sigma} - \nabla_\sigma(\Phi + K) - c_p \theta \nabla_\sigma P - \frac{g}{\pi}\frac{\partial \mathcal{T}}{\partial \sigma} \tag{7}$$

where f is the Coriolis parameter, $\mathbf{k}$ is the unit vector in the vertical, $\zeta \equiv \nabla_\sigma \times \mathbf{V}$ is the vertical component of the vorticity along σ surfaces, $K \equiv \frac{1}{2}(\mathbf{V} \cdot \mathbf{V})$ is the kinetic energy per unit mass, g is the acceleration of gravity, and $\mathcal{T}$ is the horizontal frictional stress.

2.2 Description of discretization methods and structure of the model

In GEOS-1 GCM, a Lorenz grid is used in the vertical, with both winds and temperatures defined at the same levels. The atmosphere between $\sigma = 0$ and $\sigma = 1$ is divided into LM layers. At these LM layers, the velocity, the potential temperature and the specific masses of all trace constituents are defined. The vertical velocity $\dot{\sigma}$ is defined at interfaces between layers and at top and bottom surfaces.

On horizontal grids, the prognostic variables are located on an Arakawa C grid. The temperature, pressure and all tracers are located at "p-points", which exclude the poles. The "u-points", at which zonal wind components are defined, are located between "p-points" and on the same latitude circles, while "v-points" are located between "p-points" and on the same meridians. The vorticity is defined at "ζ-points" on the same latitude circles as v and on the same meridians as u.

The discretization of the momentum equation is carried out with a second-order energy and potential enstrophy conserving scheme of Sadourny described by Burridge and Haseler (1977). A simple second-order finite difference scheme is used for discretizing the thermodynamic equation and the continuity equation.

A polar Fourier filter is applied to the tendencies of all the prognostic variables. The purpose of the polar filter is to avoid linear computational instability due to convergence of the meridians near the poles. The filter acts poleward of about 45^o latitude, and its strength is gradually increased towards the pole by increasing the number of affected zonal wavenumbers and the amount by which they are damped.

The time differencing scheme used in NASA GEOS-1 C-Grid GCM is the Brown-Campana scheme (1978). It is an explicit scheme used in conjunction with a leap-frog differencing scheme that relaxes somewhat the instability condition for gravity waves. The basic idea of the Brown-Campana scheme is to average the pressure gradient force over three time levels. In NASA GEOS-1 C-Grid GCM, a simple strategy is used which assumes the pressure gradient force is linearized. That is, to average the three time levels of the mass field and only to compute the pressure gradient once, while the averaged mass field is used only for the pressure gradient calculations. Asselin (1972) time filter and Shapiro filter are also used in the dynamic core of NASA GEOS-1 C-Grid GCM.

The NASA GEOS-1 C-Grid GCM has a resolution of $5^o \times 4^o$ longitude-latitude grid points in horizontal plane and 20 σ-coordinate levels. The time step used is five minutes.

2.3 Discrete dynamical equations

The discrete dynamical equations used in the adiabatic version of the NASA GEOS-1 C-Grid GCM are as follows (for all the definition of the used symbols and the detailed method of the Arakawa-Lamb C-grid discrete scheme please see Documentation of the ARIES/GEOS Dynamical Core (Suarez and Takacs, 1994)). The equations are presented here for ease of reference to corresponding equations in the tangent linear model.

The hydrostatic equations are

$$\Phi_{LM} = \Phi_s + c_p \theta_{LM} (\hat{P}_{LM+1} - P_{LM}) \tag{8}$$

and

$$\Phi_l = \Phi_{l+1} + c_p \hat{\theta}_{l+1} (P_{l+1} - P_l), \qquad \text{for } l = 1,\ LM - 1 \tag{9}$$

The continuity equations are

$$\frac{\partial \pi_{i,j}}{\partial t} = - \sum_{l=1}^{LM} \frac{1}{(\Delta_p^2)_{i,j}} [\delta_i u_l^* + \delta_j v_l^*]_{i,j} (\delta\sigma)_l \tag{10}$$

and

$$(\pi\dot{\sigma})_{i,j,l+1} = -\sigma_{l+1} \frac{\partial \pi_{i,j}}{\partial t} - \sum_{l=1}^{l} \frac{1}{(\Delta_p^2)_{i,j}} [\delta_i u_l^* + \delta_j v_l^*]_{i,j} (\delta\sigma)_l \quad \text{for } l = 1, LM - 1, \tag{11}$$

The thermodynamic equation is

$$\frac{\partial (\pi\theta_l)_{i,j}}{\partial t} = - \frac{1}{(\Delta_p^2)_{i,j}} [\delta_i (u^* \bar{\theta}^i)_l + \delta_j (v^* \bar{\theta}^j)_l]_{i,j} - \left(\frac{\delta(\pi\dot{\sigma}\hat{\theta})}{\delta_l \sigma} \right)_{i,j,l} \tag{12}$$

The component forms of the momentum equation are

$$
\begin{aligned}
\frac{\partial u_{i,j,l}}{\partial t} &= \frac{1}{(\Delta^u x)_{i,j}}[\alpha_{i,j} v^*_{i+1,j+1} + \beta_{i,j} v^*_{i,j+1} + \gamma_{i,j} v^*_{i,j} + \delta_{i,j} v^*_{i+1,j} + \\
&\quad + \nu_{i,j} u^*_{i,j-1} - \nu_{i,j+1} u^*_{i,j+1} - \epsilon_{i+1,j} u^*_{i+1,j} + \epsilon_{i,j} u^*_{i-1,j}]_l - \\
&\quad - \frac{1}{(\bar{\pi}^i)_{i,j}(\delta_l \sigma)_l} \frac{1}{2} \left[\overline{(\pi\dot{\sigma})}^i_l (u_l - u_{l-1}) + \overline{(\pi\dot{\sigma})}^i_{l+1} (u_{l+1} - u_l) \right]_{i,j} - \\
&\quad - \frac{1}{(\Delta^u x)_{i,j}} \left[\delta_i (\Phi_l + K_l) + c_p \overline{\theta_l \left(\frac{dP}{d\pi} \right)_l}^i \delta_i \pi \right]_{i,j} \qquad \text{for } j = 1, JM-1
\end{aligned} \tag{13}
$$

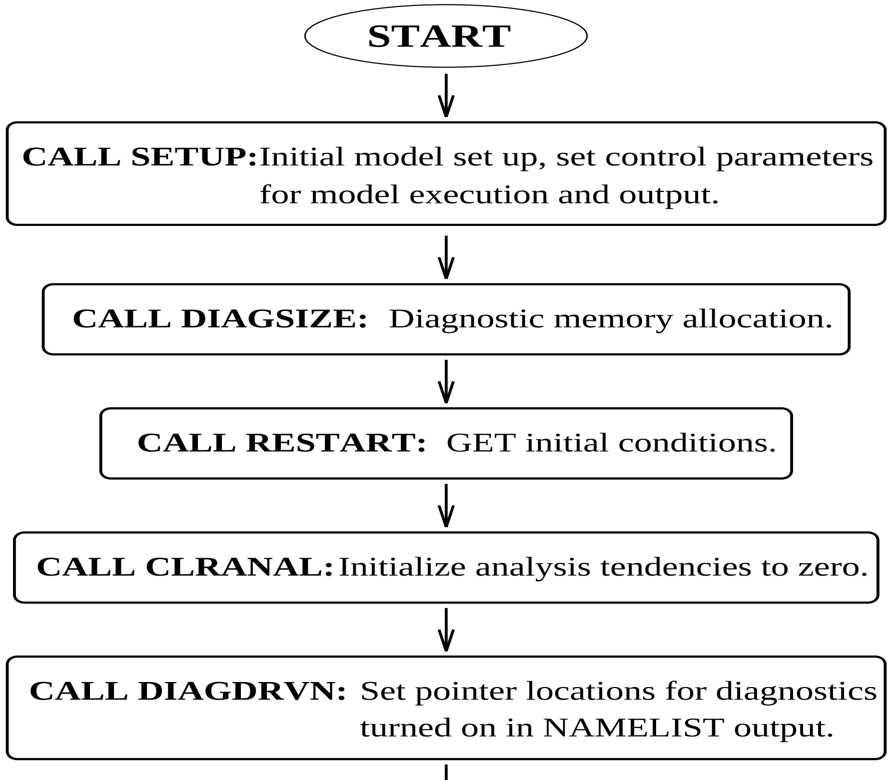
$$
\begin{aligned}
\frac{\partial v_{i,j,l}}{\partial t} &= -\frac{1}{(\Delta^v y)_j}[\alpha_{i-1,j-1} u^*_{i-1,j-1} + \beta_{i,j-1} u^*_{i,j-1} + \\
&\quad + \gamma_{i,j} u^*_{i,j} + \delta_{i-1,j} u^*_{i-1,j} + \\
&\quad + \mu_{i,j} v^*_{i+1,j} - \mu_{i-1,j} v^*_{i-1,j} - \varphi_{i,j} v^*_{i,j+1} + \varphi_{i,j-1} v^*_{i,j-1}]_l - \\
&\quad - \frac{1}{(\bar{\pi}^j)_{i,j}(\delta_l \sigma)_l} \frac{1}{2} \left[\overline{(\pi\dot{\sigma})}^j_l (v_l - v_{l-1}) + \overline{(\pi\dot{\sigma})}^j_{l+1} (v_{l+1} - v_l) \right]_{i,j} - \\
&\quad - \frac{1}{(\Delta^v y)_j} \left[\delta_j (\Phi_l + K_l) + c_p \overline{\theta_l \left(\frac{dP}{d\pi} \right)_l}^j \delta_j \pi \right]_{i,j} \qquad \text{for } j = 2, JM-1
\end{aligned} \tag{14}
$$

Here the α, β, γ, δ, ϵ, φ, ν and μ are linear combinations of neighboring potential vorticities.

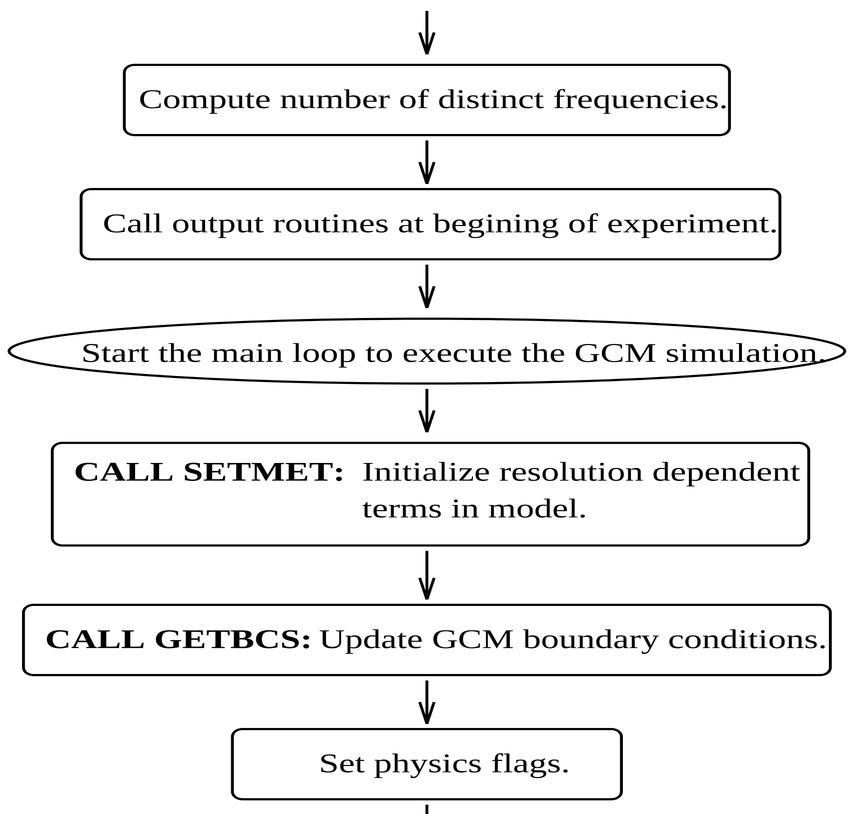
For further details concerning this GCM, we refer to Suarez et al., (1994) and Takacs et al. (1994). The detailed flow chart of the NASA GEOS-1 GCM with full physics package is in the next subsection (for getting the adiabatic version of it, one just needs to skip the computational processes related to the physics packages and the moisture processes).

2.4 Flow chart of the NASA GEOS-1 GCM

At the next six pages we will present the flow chart of the NASA GEOS-1 C-Grid GCM.



GEOS-1/ARIES GCM FLOW CHART, page 1.



GEOS-1/ARIES GCM FLOW CHART, page 2.

CALL CTOA: Convert ‘‘C’’ gridded data to ‘‘A’’ gridded data for physics packages use.

CALL MOISTIO: Compute the moist processes, Relaxed Arakawa-Schubert scheme and large-scale convection.

CALL SWRIO: Compute the short wave radiation processes.

CALL LWRIO: Compute the long wave radiation processes.

CALL TURBIO: Compute the turbulence parameterization processes.

GEOS-1/ARIES GCM FLOW CHART, page 3.

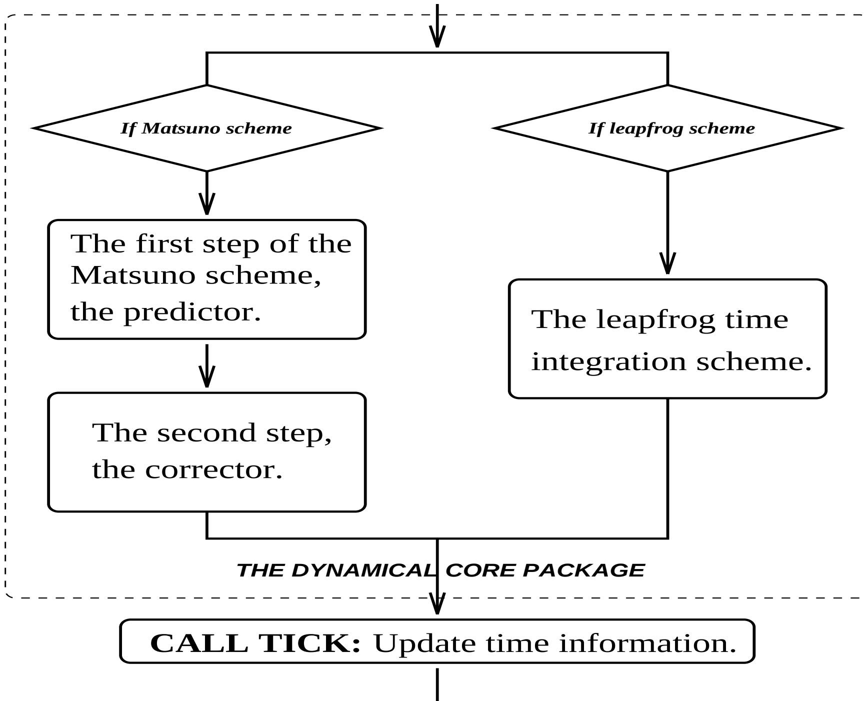
CALL ATOC: Convert ‘‘A’’ gridded data to ‘‘C’’ gridded data to execute the hydro dynamical processes.

Calculate the diagnostics, including

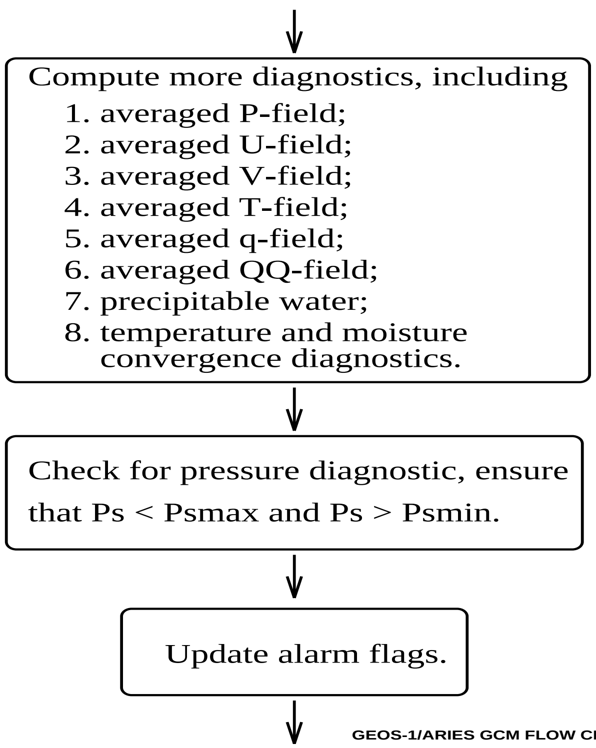
1. total diabatic U-tendency;
2. total diabatic V-tendency;
3. total diabatic T-tendency;
4. total diabatic q-tendency;
5. the analysis tendencies increment of U, V, T, q;
6. incident solar radiation;
7. net solar radiation at the ground;
8. solar radiation heating.

Starting the hydrodynamical processes, the time integration scheme is either the Matsuno two-step scheme or the leapfrog scheme.

GEOS-1/ARIES GCM FLOW CHART, page 4.



GEOS-1/ARIES GCM FLOW CHART, page 5.



GEOS-1/ARIES GCM FLOW CHART, page 6.

Call all output routines to output prognostic and diagnostic results.

CALL RESTART: write out restart and diagnostic results to disk.

End of the main loop.

CALL RESTART: write out current restart and diagnostic results to disk.

STOP

GEOS-1/ARIES GCM FLOW CHART, page 7.

THE FLOW CHART OF THE DYNAMICAL CORE PACKAGE DESCRIBING BOTH THE FIRST STEP (PREDICTOR) AND THE SECOND STEP (CORRECTOR) OF THE MATSUNO TIME INTEGRATION SCHEME AS WELL AS THE LEAPFROG TIME INTEGRATION SCHEME.

ENTER

Put total diabatic tendencies into the control variable tendency terms.

CALL SHAPIJ: apply the global Shapiro low-pass filter scheme on U, V, T, q fields to damp small-scale dispersive waves.

If integrating backward in time, invert the sign of the time tendency terms to ensure non-reversible effects are positive.

CALL SETGRID: define parameters for integrating at the staggered grid points.

GEOS-1/ARIES GCM FLOW CHART, page 8.

CALL SETDMP: calculate damping coefficients for high latitude filter.

↓

CALL PKAP: calculate Phillips ‘‘P**KAPPA’’ on C-grid.

↓

Compute perturbation geopotential height.

↓

Average mass to vorticity points.

↓

CAAL SUB1: compute kinetic energy, potential vorticity, ustar, vstar, which are used for Arakawa C-grid scheme.

↓

GEOS-1/ARIES GCM FLOW CHART, page 9.

↓

CALL SUB2: compute parameter alpha, beta, gamma, delta which are used for Arakawa C-grid scheme.

↓

CALL HADVECT: compute tendencies of height and wind due to the horizonal advection processes.

↓

CALL HADVCTT: compute temperature tendencies due to the horizonal advection processes.

↓

CALL HADVCTT: compute moisture tendencies due to the horizonal advection processes.

↓

Compute the adiabatic pressure and total pressure tendencies.

↓

GEOS-1/ARIES GCM FLOW CHART, page 10.

CALL GETOMEGA: compute omega diagnostic.

Compute PI*SIGMADOT

CALL VADVCT: calculate centered second-order vertical advection of U, V, T, q.

CALL FFTDDT: apply FFT scheme to filter the tendencies of U, V, T, q over the high-latitude region (polar filter).

GEOS-1/ARIES GCM FLOW CHART, page 11.

Add analysis increment to dynamical omega diagnostics.

CALL STEP: update prognostic fields one time-step, compute total tendency diagnostics, check global mean surface pressure and negative humidities, bump diagnostic counters, as well as **CALL TMFILT** for applying the Asselin time filter.

EXIT

GEOS-1/ARIES GCM FLOW CHART, page 12.

3 Tangent linear model of the adiabatic version of NASA GEOS-1 C-Grid GCM

3.1 Linearized discrete dynamical equations

The linearized discrete dynamical equations of Eqs. (8)- (14) used in the derivation of the tangent linear model of the adiabatic version of the NASA GEOS-1 GCM are as follows (we use { } to describe the basic state trajectory terms and $()'$ to denote the perturbation variables terms. For all other definitions of the symbols used please see the documentation of the ARIES/GEOS Dynamical Core (Suarez and Takacs, 1994)).

The linearized equations of the hydrostatic equations (8)- (9) are

$$(\Phi_{LM})' = (\Phi_s)' + c_p\{\theta_{LM}\}(\hat{P}_{LM+1} - P_{LM})' + c_p\{\hat{P}_{LM+1} - P_{LM}\}(\theta_{LM})' \tag{15}$$

and

$$(\Phi_l)' = (\Phi_{l+1})' + c_p\{\hat{\theta}_{l+1}\}(P_{l+1} - P_l)' + c_p\{P_{l+1} - P_l\}(\hat{\theta}_{l+1})', \qquad \text{for } l = 1,\ LM - 1 \tag{16}$$

The linearized continuity equations are

$$\frac{\partial(\pi_{i,j})'}{\partial t} = -\sum_{l=1}^{LM} \frac{1}{(\Delta_p^2)_{i,j}}[\delta_i(u_l^*)' + \delta_j(v_l^*)']_{i,j}(\delta\sigma)_l \tag{17}$$

and

$$\begin{aligned}\{\pi_{i,j,l+1}\}(\dot{\sigma}_{i,j,l+1})' \quad &+ \quad \{\dot{\sigma}_{i,j,l+1}\}(\pi_{i,j,l+1})' = -\sigma_{l+1}\frac{\partial(\pi_{i,j})'}{\partial t} - \\ &- \quad \sum_{l=1}^{l} \frac{1}{(\Delta_p^2)_{i,j}}[\delta_i(u_l^*)' + \delta_j(v_l^*)']_{i,j}(\delta\sigma)_l\end{aligned} \tag{18}$$

The linearized form of the thermodynamic equation (12) is

$$\begin{aligned}\frac{\partial(\{\pi\}(\theta_l)')_{i,j}}{\partial t} + \frac{\partial(\{\theta_l\}(\pi)')_{i,j}}{\partial t} \quad &= \quad -\frac{1}{(\Delta_p^2)_{i,j}}[\delta_i(\{u^*\}(\bar{\theta}^i)' + \{\bar{\theta}^i\}(u^*)')_l + \\ &+\delta_j(\{v^*\}(\bar{\theta}^j)' + \{\bar{\theta}^j\}(v^*)')_l]_{i,j} - \\ &- \left(\frac{\delta(\{\pi\dot{\sigma}\}(\hat{\theta})' + \{\pi\hat{\theta}\}(\dot{\sigma})' + \{\hat{\theta}\dot{\sigma}\}(\pi)')}{\delta_l\sigma}\right)_{i,j,l}\end{aligned} \tag{19}$$

The linearized component forms of the momentum equation (13)- (14) are

$$\frac{\partial(u_{i,j,l})'}{\partial t} \quad = \quad \frac{1}{(\Delta^u x)_{i,j}}[\{\alpha_{i,j}\}(v^*_{i+1,j+1})' + \{v^*_{i+1,j+1}\}(\alpha_{i,j})' + \tag{20}$$

$$
\begin{aligned}
& +\{\beta_{i,j}\}(v^*_{i,j+1})' + \{v^*_{i,j+1}\}(\beta_{i,j})' + \\
& +\{\gamma_{i,j}\}(v^*_{i,j})' + \{v^*_{i,j}\}(\gamma_{i,j})' + \{\delta_{i,j}\}(v^*_{i+1,j})' + \{v^*_{i+1,j}\}(\delta_{i,j})' + \\
& +\{\nu_{i,j}\}(u^*_{i,j-1})' + \{u^*_{i,j-1}\}(\nu_{i,j})' - \{\nu_{i,j+1}\}(u^*_{i,j+1})' - \{u^*_{i,j+1}\}(\nu_{i,j+1})' - \\
& -\{\epsilon_{i+1,j}\}(u^*_{i+1,j})' - \{u^*_{i+1,j}\}(\epsilon_{i+1,j})' + \{\epsilon_{i,j}\}(u^*_{i-1,j})' + \{u^*_{i-1,j}\}(\epsilon_{i,j})']_l + \\
& + \left\{ \frac{1}{(\bar{\pi}^i)^2_{i,j}(\delta_l\sigma)_l} \frac{1}{2} \left[\overline{(\pi\dot{\sigma})}^i_l (u_l - u_{l-1}) + \overline{(\pi\dot{\sigma})}^i_{l+1}(u_{l+1} - u_l) \right]_{i,j} \right\} (\bar{\pi}^i)'_{i,j} - \\
& - \frac{1}{\{(\bar{\pi}^i)_{i,j}\}(\delta_l\sigma)_l} \frac{1}{2} \left[\left\{ \overline{(\pi\dot{\sigma})}^i_l \right\} (u_l - u_{l-1})' + \{u_l - u_{l-1}\} \left(\overline{(\pi\dot{\sigma})}^i_l \right)' \right] - \\
& - \frac{1}{\{(\bar{\pi}^i)_{i,j}\}(\delta_l\sigma)_l} \frac{1}{2} \left[\left\{ \overline{(\pi\dot{\sigma})}^i_{l+1} \right\} (u_{l+1} - u_l)' + \{u_{l+1} - u_l\} \left(\overline{(\pi\dot{\sigma})}^i_{l+1} \right)' \right] - \\
& - \frac{1}{(\Delta^u x)_{i,j}} \left[\delta_i(\Phi_l + K_l)' + c_p \left\{ \overline{\theta_l \left(\frac{dP}{d\pi} \right)_l}^i \right\} \delta_i(\pi)' \right]_{i,j} - \\
& - \frac{1}{(\Delta^u x)_{i,j}} \left[c_p \delta_i \{\pi\} \left(\overline{\theta_l \left(\frac{dP}{d\pi} \right)_l}^i \right)' \right]_{i,j} \qquad \text{for } j = 1, JM-1
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial (v_{i,j,l})'}{\partial t} = & - \frac{1}{(\Delta^v y)_j} [\{\alpha_{i-1,j-1}\}(u^*_{i-1,j-1})' + \{u^*_{i-1,j-1}\}(\alpha_{i-1,j-1})' + \\
& +\{\beta_{i,j-1}\}(u^*_{i,j-1})' + \{u^*_{i,j-1}\}(\beta_{i,j-1})' + \\
& +\{\gamma_{i,j}\}(u^*_{i,j})' + \{u^*_{i,j}\}(\gamma_{i,j})' + \{\delta_{i-1,j}\}(u^*_{i-1,j})' + \{u^*_{i-1,j}\}(\delta_{i-1,j})' + \\
& +\{\mu_{i,j}\}(v^*_{i+1,j})' + \{v^*_{i+1,j}\}(\mu_{i,j})' - \{\mu_{i-1,j}\}(v^*_{i-1,j})' - \{v^*_{i-1,j}\}(\mu_{i-1,j})' - \\
& -\{\varphi_{i,j}\}(v^*_{i,j+1})' - \{v^*_{i,j+1}\}(\varphi_{i,j})' + \{\varphi_{i,j-1}\}(v^*_{i,j-1})' + \{v^*_{i,j-1}\}(\varphi_{i,j-1})' + \\
& + \left\{ \frac{1}{(\bar{\pi}^j)^2_{i,j}(\delta_l\sigma)_l} \frac{1}{2} \left[\overline{(\pi\dot{\sigma})}^j_l (v_l - v_{l-1}) + \overline{(\pi\dot{\sigma})}^j_{l+1}(v_{l+1} - v_l) \right]_{i,j} \right\} (\bar{\pi}^j)'_{i,j} - \\
& - \frac{1}{\{(\bar{\pi}^j)_{i,j}\}(\delta_l\sigma)_l} \frac{1}{2} \left[\left\{ \overline{(\pi\dot{\sigma})}^j_l \right\} (v_l - v_{l-1})' + \{v_l - v_{l-1}\} \left(\overline{(\pi\dot{\sigma})}^j_l \right)' \right]_{i,j} - \\
& - \frac{1}{\{(\bar{\pi}^j)_{i,j}\}(\delta_l\sigma)_l} \frac{1}{2} \left[\left\{ \overline{(\pi\dot{\sigma})}^j_{l+1} \right\} (v_{l+1} - v_l)' + \{v_{l+1} - v_l\} \left(\overline{(\pi\dot{\sigma})}^j_{l+1} \right)' \right]_{i,j} - \\
& - \frac{1}{(\Delta^v y)_j} \left[\delta_j(\Phi_l + K_l)' + c_p \left\{ \overline{\theta_l \left(\frac{dP}{d\pi} \right)_l}^j \right\} \delta_j(\pi)' \right]_{i,j} - \\
& - \frac{1}{(\Delta^v y)_j} \left[\delta_j \{\pi\} \left(\overline{\theta_l \left(\frac{dP}{d\pi} \right)_l}^j \right)' \right]_{i,j} \qquad \text{for } j = 2, JM-1
\end{aligned}
\tag{21}
$$

Here the α, β, γ, δ, ϵ, φ, ν and μ denote linear combinations of neighboring potential vorticities.

To obtain the perturbation temperature output fields $(T)'$, with the definition of θ,

$$\{\theta\} + (\theta)' = \frac{\{T\} + (T)'}{\{P\} + (P)'} \tag{22}$$

we have

$$\begin{aligned}
(T)' &= (\{\theta\} + (\theta)')(\{P\} + (P)') - \{\theta\}\{P\} \\
&= \left[(\{\theta\} + (\theta)')\left(\{p_s\}^\kappa + ((p_s)')^\kappa\right) - \{\theta\}\{p_s\}^\kappa\right]\mathcal{P}_l
\end{aligned} \tag{23}$$

where we take $p_T = 0$ and use Equation (27a) of Suarez and Takacs (1994)'s documentation of the dynamical core of NASA model,

$$P_l \equiv \pi^\kappa \mathcal{P}_l \tag{24}$$

where

$$\mathcal{P}_l = \frac{1}{p_0^\kappa(1+\kappa)}\left[\frac{\delta\sigma^{\kappa+1}}{\delta\sigma}\right]_l \tag{25}$$

We may design the TLM version of NASA GEOS-1 GCM from the above TLM discrete equations. However, to code the TLM conveniently and to avoid coding mistakes, we choose another way to code the tangent linear model, i.e., we linearize the original adiabatic version of NASA GEOS-1 GCM code segment by segment. The detailed method is presented in the next subsection.

3.2 Coding the tangent linear model

For coding the tangent linear model, we linearize the original nonlinear forward model code line by line, do loop by do loop and subroutine by subroutine. This amounts to obtain the exact same tangent linear model as by coding directly from the original linearized model dynamical equations.

The tangent linear model is the linearized nonlinear forward model in the vicinity of a basic state which is on a model trajectory. For any original code line, we may write it as

$$U = f(\mathbf{X}) \tag{26}$$

where

$$\mathbf{X} = (x_1,\ x_2,\ \cdots,\ x_m)^T \tag{27}$$

where U is a new derived variable related to the original control variables of the nonlinear forward model, i.e., it may be one of the original control variables or an intermediate variable which is a function of the original control variables. Here $x_1,\ x_2,\ \cdots,\ x_m$ (the components of the vector $\mathbf{X}$) are the required variables to derive U, which may consist of either the original model control variables or of the intermediate variables derived from the original control variables. m is the number of the required variables.

The corresponding tangent linear code will assume the form:

$$\begin{aligned}\delta U &= \delta x_1 \left(\frac{\partial f}{\partial x_1}\right)_{\mathbf{X}=\mathbf{X}_{basic\ state}} + \delta x_2 \left(\frac{\partial f}{\partial x_2}\right)_{\mathbf{X}=\mathbf{X}_{basic\ state}} + \cdots + \\ &\quad + \delta x_m \left(\frac{\partial f}{\partial x_m}\right)_{\mathbf{X}=\mathbf{X}_{basic\ state}}\end{aligned} \tag{28}$$

where $\mathbf{X} = \mathbf{X}_{basic\ state}$ means that in the expression $\frac{\partial f}{\partial x_i}$, $i = 1,\ 2,\ \cdots,\ m$, all the values of the required variables $x_1,\ x_2,\ \cdots,\ x_m$ are chosen to have the exact same values as those of the basic state trajectory values in the nonlinear forward model to ensure that the basic state of the integration of tangent linear model is exactly the basic state of the nonlinear model integrating trajectory. Here δU and $\delta x_1,\ \delta x_2,\ \cdots,\ \delta x_m$ are the corresponding perturbation variables of U and $x_1,\ x_2,\ \cdots,\ x_m$, respectively.

In order to obtain the necessary values of $\mathbf{X}_{basic\ state}$, the nonlinear model integrating trajectory, for the tangent linear model, we must apply a parallel method. This method consists of calculating in parallel the nonlinear model integration trajectory as the basic state $\mathbf{X}_{basic\ state}$ and carrying out the integration of perturbation variables(such as δU) in the tangent linear model.

The dynamical core part of the tangent linear model flow chart of the NASA GEOS-1 GCM is presented in Subsection 3.4 (for the adiabatic version one needs to omit some steps related to the computation of physical processes, moisture processes and some diagnostic calculations),

3.3 Notational convention for variables and subroutines used in the tangent linear model code

For convenience, the same original names used in the nonlinear forward model are employed for the corresponding perturbation variables in the tangent linear model code. For instance, we use "U" for "δU", "$PKHT$" for "$\delta(PKHT)$", "$USTR2$" for "$\delta(USTR2)$", etc.. This also means that the perturbation control variables in the TLM share the same common structure and same common block names as the GCM itself. So one needs to pay attention to this issue when running the TLM in conjunction with the original GCM.

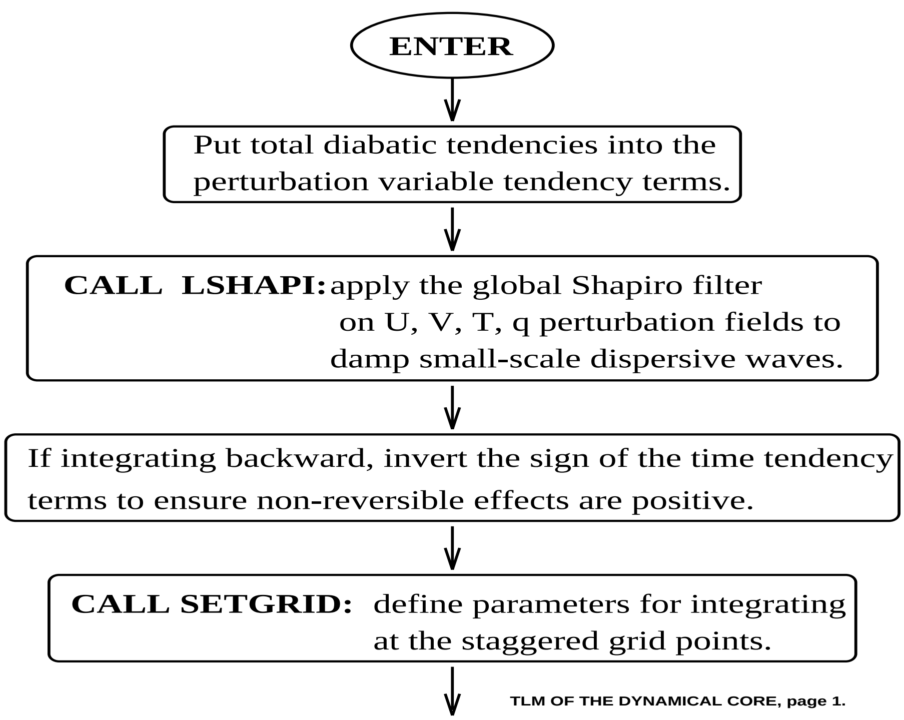
We just add a "0" at the end of a variable name in the original nonlinear forward model to represent the corresponding basic state variable, such as using "$U0$" for "$U_{basic\ state}$", "$PKHT0$" for "$(PKHT)_{basic\ state}$" , "$USTR20$" for "$(USTR2)_{basic\ state}$", etc..

For naming subroutines in the tangent linear model , we simply add a "L" at the beginning of the original names of subroutines of the nonlinear forward model. To conform to the general FORTRAN language rule, if the new name of a tangent linear subroutine exceeds six letters, we just retain its first six letters. For instance, for the subroutines of the original nonlinear model "$SUB1$", "$VADVCT$" and "$HADVCTT$", the corresponding names of the subroutines in the tangent linear model code are "$LSUB1$", "$LVADVC$" and "$LHADVC$", respectively.

3.4 Flow chart of the dynamical core of the tangent linear model

The following is the flow chart of the dynamical core of the tangent linear model of NASA GEOS-1 GCM.

THE FLOW CHART OF THE TANGENT LINEAR MODEL OF THE DYNAMICAL CORE PACKAGE DESCRIBING THE TWO STEPS OF THE MATSUNO TIME INTEGRATION SCHEME AND THE LEAPFROG TIME INTEGRATION SCHEME.



CALL SETDMP: calculate damping coefficients for high latitude filter.

CALL LPKAP: calculate perturbation ''P**KAPPA'' term.

Compute perturbation geopotential height.

Average perturbation mass to vort. points.

CAAL LSUB1: compute the perturbation terms, kinetic energy, potential vorticity, ustar, vstar , which are used for Arakawa C-grid scheme.

TLM OF THE DYNAMICAL CORE, page 2.

CALL SUB2: compute parameters alpha, beta, gamma, delta which are used for Arakawa C-grid scheme.

CALL LHADVE: compute tendencies of perturbations of height and wind due to the horizonal advection processes.

CALL LHADVC: compute perturbation temperature tendencies due to the horizonal advection processes.

CALL LHADVC: compute perturbation moisture tendencies due to the horizonal advection processes (for the adiabatic version, skip this step).

Compute the adiabatic perturbation pressure and total perturbation pressure tendencies.

TLM OF THE DYNAMICAL CORE, page 3.

Compute delta(PI*SIGMADOT)

CALL LVADVC: calculate centered second-order vertical advection of perturbations of U, V, T, q.

CALL FFTDDT: apply FFT scheme to filter the tendencies of the perturbation variables of U, V, T, q over the the high-latitude region.

TLM OF THE DYNAMICAL CORE, page 4.

Add analysis increment to dynamical omega diagnostics (for the adiabatic tangent linear model, it is not necessary to compute diagnostics. May skip this step).

CALL LSTEP: update perturbation prognostic fields one time-step, **CALL LTMFIL** for applying Asselin time filter. (The following computations are skipped in the adiabatic version of TLM, i.e., compute total perturbation tendency diagnostics, check global mean surface pressure and negative humidities, as well as bump diagnostic counters.)

EXIT

TLM OF THE DYNAMICAL CORE, page 5.

3.5 Verifying correctness of the tangent linear model

To verify the correctness of the tangent linear model, we compared the output of each subroutine of the tangent linear model with its counterpart in the original forward model. To verify the full tangent linear model, we employed a more quantitative method, described below.

The evolution of $\mathbf{X}$, the vector of control variables, is given by the integration of the model M between times t_0 and t_n as:

$$\mathbf{X}(t_n) = M(t_n, t_0)(\mathbf{X}(t_0)) = M(t_n, t_0)(\mathbf{X}_0(t_0) + \delta\mathbf{X}(t_0)) \tag{29}$$

whereas the first order evolution of the perturbation $\delta\mathbf{X}(t_n)$ is the result of the integration of the tangent linear model R :

$$\delta\mathbf{X}(t_n) = R(t_n, t_0)\,\delta\mathbf{X}(t_0) \tag{30}$$

We then compare the total perturbation

$$N(\delta\mathbf{X}(t_0)) = M(t_n, t_0)(\mathbf{X}_0(t_0) + \delta\mathbf{X}(t_0)) - M(t_n, t_0)(\mathbf{X}_0(t_0)) \tag{31}$$

with its linear component

$$L(\delta\mathbf{X}(t_0)) = R(t_n, t_0)\,\delta\mathbf{X}(t_0) \tag{32}$$

The difference between the two is denoted as

$$D(\delta\mathbf{X}(t_0)) = N(\delta\mathbf{X}(t_0)) - L(\delta\mathbf{X}(t_0)) \tag{33}$$

In order to quantify this comparison, we choose a norm whose square is defined by

$$\|\mathbf{X}\|^2 = \mathbf{X}^T W \mathbf{X} \tag{34}$$

in accordance with the norm used in the inner product of the cost function for the variational data assimilation problem. The relative difference between the tangent linear model and the nonlinear forward model is then defined as the ratio $\frac{\|D\|}{\|L\|}$. We first examine different components of $\frac{\|D\|}{\|L\|}$ and calculate correlation coefficients between nonlinear output fields N and linear output fields L according to the individual model variables contributions (u, v, T, Ps).

The data used to verify the tangent linear model is the January 1, 1985 00Z ECMWF data. As in Rabier and Courtier(1992), we chose zonal average fields as basic state initial condition, while departure of zonal average fields multiplied by a scaling factor α serves as the perturbation of initial conditions. Both the basic state initial condition fields (i.e., zonal average fields) and the perturbed initial condition fields (which is just the original ECMWF data) are good initial conditions for stable integration of the model and do not cause excitation of gravity waves. The amplitudes of different components of this perturbation are very large when $\alpha = 1$: at level $\sigma = 0.223$, the maximum zonal wind perturbation reaches 63 m/s; at level $\sigma = 0.352$, the maximum meridional wind perturbation reaches 65 m/s; the temperature perturbation at $\sigma = 0.029$ level is close to 18 K, while the maximum perturbation of the surface pressure is -424 hPa, due to the orography of the Tibetan Plateau. The tangent linear model check with these strongly perturbed initial conditions can shed light on possible coding errors. The period of integration, t_n, is taken to be 12 hours.

Table 1: Correlation Coefficients Between N Field and L Field:

α	u	v	T	p_s
1.0	0.8919275	0.9094758	0.8628350	0.8714384
10^{-1}	0.9981103	0.9986473	0.9971051	0.9981210
10^{-2}	0.9999804	0.9999861	0.9999694	0.9999809
10^{-3}	0.9999998	0.9999999	0.9999997	0.9999998
10^{-4}	1.0000000	1.0000000	1.0000000	1.0000000
10^{-5}	1.0000000	1.0000000	1.0000000	1.0000000

Table 2: Relative Error $\frac{||D||}{||L||}$ (%):

α	u	v	T	p_s
1.0	45.32	41.99	50.69	49.41
10^{-1}	6.18	5.20	7.62	6.14
10^{-2}	0.63	0.53	0.79	0.62
10^{-3}	6.33×10^{-2}	5.27×10^{-2}	7.88×10^{-2}	6.22×10^{-2}
10^{-4}	6.33×10^{-3}	5.27×10^{-3}	7.89×10^{-3}	6.22×10^{-3}
10^{-5}	6.33×10^{-4}	5.27×10^{-4}	7.89×10^{-4}	6.22×10^{-4}

Table 1 presents correlation between N field and L field for various values of parameter α, while Table 2 displays relative error $\frac{||D||}{||L||}$. From these tables we see that all correlation coefficients between N and L for each of the variable fields exceed 86% (when α=1.0), and reach values close to unity when α is less or equal to 0.1. As α decreases, the relative errors decrease to very small values in a linear manner. The correlation coefficients reach up to 10 digits of accuracy in vicinity of unity while relative error values attain an order of magnitude of 10^{-4} when α is equal to 10^{-5}. Comparing with similar relative errors analysis applied to adiabatic version of NASA/GLA Semi-Lagrangian Semi-Implicit (SLSI) GCM (Table 3) (Li et al., 1994), the tangent linear model of the adiabatic version of NASA

GEOS-1 C-Grid GCM seems to display better linearity property than the tangent linear model of NASA/GLA SLSI GCM. These analyses provide a reliable indication about the correctness of the tangent linear model code.

Table 3: Relative Error $\frac{\|D\|}{\|L\|}$ (%) of NASA/GLA 3-D SLSI GCM(from Li et al., 1994):

α	u	v	$\frac{T'}{T}$	$(\ln p_s)'$
1.0	34.85	42.09	44.12	28.77
10^{-1}	5.22	10.63	7.63	3.44
10^{-2}	6.05	4.81	26.40	3.44
10^{-3}	8.85×10^{-2}	0.14	0.11	5.41×10^{-2}
10^{-4}	2.77×10^{-2}	4.14×10^{-2}	2.67×10^{-2}	1.49×10^{-2}
10^{-5}	1.71×10^{-2}	2.16×10^{-2}	1.38×10^{-2}	1.34×10^{-2}

To assess impact of the length of integration period (or the length of data assimilation window) on the validity of tangent linear approximation, we carried out a check for different window lengths, up to 96 hours. For convenience, we averaged correlation coefficients of four model variables and used the norm (described in Eq. (3.6) in Li et al., 1994) for $\|D\|$ and $\|L\|$ to calculate relative error. The diagonal component values of weighting matrix used are $\mathbf{W}_u = \mathbf{W}_v = 10^{-3} \quad s^2 m^{-2}$, $\mathbf{W}_T = 10^{-1} \quad K^{-2}$, $\mathbf{W}_{Ps}$=$10^{-3} \quad hPa^{-2}$, respectively. We chose values of the parameter α=1.0, as representing a strong perturbation, α=0.1, as representing a normal perturbation, while α=0.01 represented a small perturbation.

Fig. 1 displays the correlation coefficients between the N and the L fields for three values of the parameter α with respect to different lengths of integration period. Fig. 2 displays the relative error $\frac{\|D\|}{\|L\|}$ curves. Considering the $\alpha = 0.1$ curve we deduce that the error of the tangent linear model is small when a normal perturbation is used corresponding to perturbations of wind, temperature and surface pressure of $1m/s$, $1K$ and $10hPa$, respectively. Even for integration periods of up to 96 hours, the correlation coefficient still exceeds 86% and the relative error is 57%. These numerical results confirm earlier results of Courtier and Talagrand (1987), Lacarra and Talagrand (1988), Rabier and Courtier (1992) that tangent linear model well approximates the nonlinear forward model for up to 4 days when initial perturbations are not too strong. For the weak perturbation case ($\alpha = 0.01$), the model displays a very good linear behavior with a correlation coefficient reaching 99.6% while the relative error remains below 10% after 4 days of numerical integration. For strongly perturbed initial conditions($\alpha = 1$), the validity of the tangent linear model decreases quickly as the length of integration period is increased. With these strong perturbations, the validity limit of the tangent linear model is less than a day.

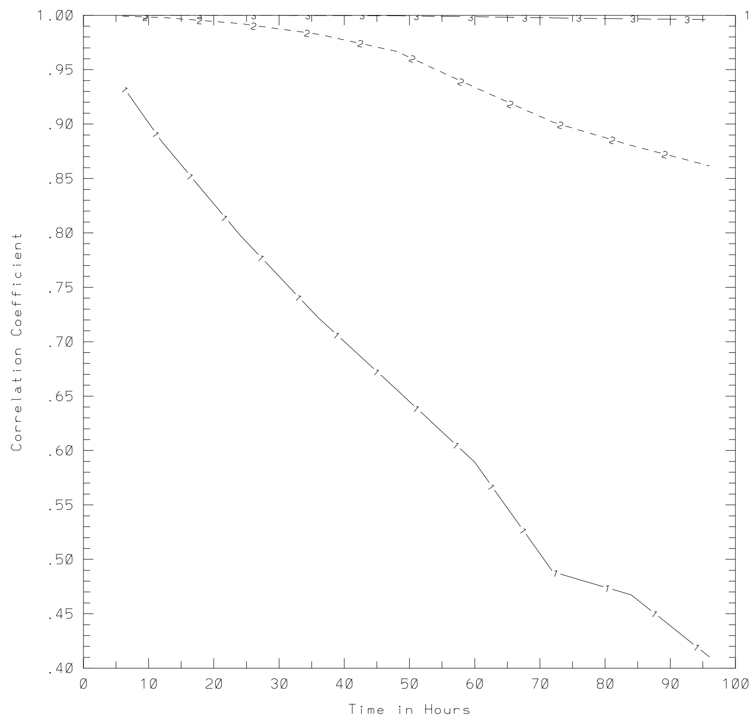


Figure 1: Correlation coefficient between the **N** fields and the **L** fields with respect to integration periods. Curve 1 corresponds to $\alpha = 1$, curve 2 corresponds to $\alpha = 0.1$, and curve 3 corresponds to $\alpha = 0.01$.

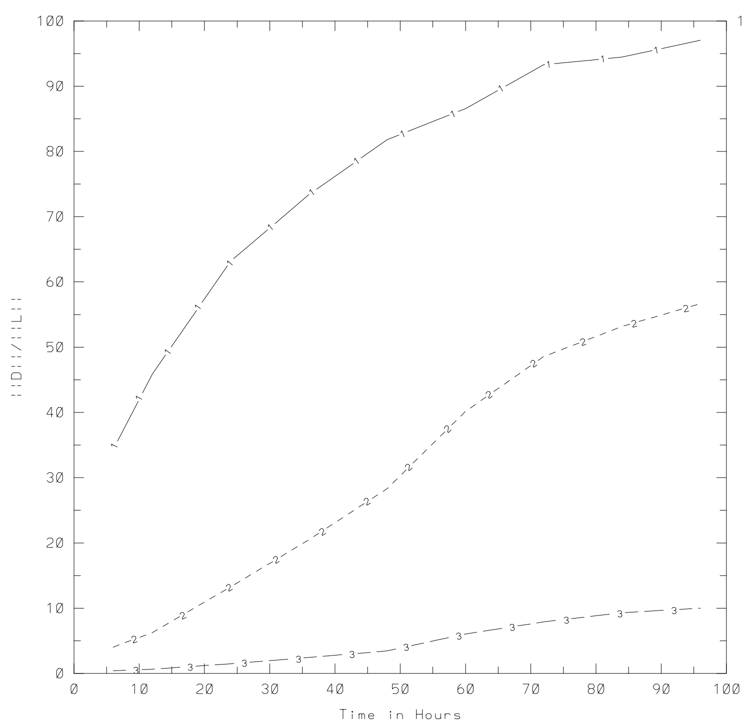


Figure 2: Relative error $\|\mathbf{D}\|/\|\mathbf{L}\|$ with respect to integration periods. Curve 1 corresponds to $\alpha = 1$, curve 2 corresponds to $\alpha = 0.1$, and curve 3 corresponds to $\alpha = 0.01$.

4 Adjoint model of the adiabatic version of the NASA GEOS-1 C-Grid GCM

4.1 Using adjoint method to calculate the gradient of the cost function

The practical determination of the adjoint model of the adiabatic version of the NASA GEOS-1 C-Grid GCM is the key computational method enabling us to calculate the gradient of the cost function with respect to the initial conditions (or other control variables) for carrying out the 4-D variational assimilation. In 4-D variational assimilation, the cost function, which measures the weighted difference between observations and forecasts in an adequate norm, is minimized by using a large-scale unconstrained minimization method iteratively which requires for its implementation the gradient of the cost function with respect to the control variables. Finally, the optimal state defines a trajectory which passes as close as possible in a least-squares sense to the observations while satisfying the system of coupled partial differential equations describing the numerical weather prediction model as strong constrains.

Assuming that the cost function consists of a weighted least square fit of the model forecast to the observations, it assumes the form :

$$J(\mathbf{X}(t_0)) = \frac{1}{2}\sum_{r=0}^{R}\left(\mathbf{X}(t_r) - \mathbf{X}^{obs}(t_r)\right)^T \mathbf{W}(t_r)\left(\mathbf{X}(t_r) - \mathbf{X}^{obs}(t_r)\right) \tag{35}$$

where $\mathbf{X}(t_r)$ is a model state vector of size $M(3K+1)$ containing the values of the zonal wind u, the meridional wind v, the temperature T and the surface pressure P_s. M is the number of grid points at each level; K is number of vertical levels. t_r is a given time in the assimilation window; $\mathbf{X}^{obs}(t_r)$ is a vector of observations defined over all grid points on all levels at time t_r; $\mathbf{W}(t_r)$ is an $N \times N$ diagonal weighting matrix. From Navon et al.(1992), we have the following expression

$$\left(\nabla J\left(\mathbf{X}(t_0)\right)\right)^T \mathbf{X}'(t_0) = \sum_{r=0}^{R}\left(\mathbf{W}(t_r)\left(\mathbf{X}(t_r) - \mathbf{X}^{obs}(t_r)\right)\right)^T \mathbf{X}'(t_r). \tag{36}$$

where $\mathbf{X}'(t_0)$ is the initial perturbation, $\mathbf{X}'(t_r)$ is the perturbation in the forecast resulting from the initial perturbation, $\nabla J\left(\mathbf{X}(t_0)\right)$ is the gradient of the cost function with respect to the initial conditions.

The tangent linear model of the nonlinear forward model can be symbolically expressed as

$$\mathbf{X}'(t_r) = \mathbf{P}_r \mathbf{X}'(t_0) \tag{37}$$

where $\mathbf{P}_r$ represents the result of applying all the operator matrices in the linear model to obtain $\mathbf{X}'(t_r)$ from $\mathbf{X}'(t_0)$.

We define the adjoint model as

$$\hat{\mathbf{X}}^r(t_0) = \mathbf{P}_r^T \hat{\mathbf{X}}(t_r), \quad r = 1, \cdots, R, \tag{38}$$

where $\hat{(\ )}$ represents an adjoint variable. After some algebra we obtain (see Navon et al., 1992) that the expression of the gradient of the cost function with respect to the initial conditions is

$$\nabla J\left(\mathbf{X}(t_0)\right) = \sum_{r=0}^{R} \mathbf{P}_r^T \mathbf{W}(t_r) \left(\mathbf{X}(t_r) - \mathbf{X}^{obs}(t_r)\right) \tag{39}$$

From this analysis, we note that the so called adjoint model operator is just the transpose of the tangent linear model operator.

The flow chart of the adjoint model (of the dynamical core of the adiabatic version of NASA GEOS-1 GCM is presented in Subsection 4.4.

4.2 Coding the adjoint model

Since the adjoint model equations consist of the transpose of the linearized version of the nonlinear forward model, if we view the tangent linear model as the result of the multiplication of a number of operator matrices:

$$\mathbf{P} = \mathbf{A}_1 \mathbf{A}_2 \cdots \mathbf{A}_N, \tag{40}$$

where each matrix $\mathbf{A}_i (i = 1, \cdots, N)$ represents either a subroutine or a single *DO* loop, then the adjoint model can be viewed as being a product of adjoint subproblems

$$\mathbf{P}^T = \mathbf{A}_N^T \mathbf{A}_{N-1}^T \cdots \mathbf{A}_1^T. \tag{41}$$

So the adjoint model is simply the complex conjugate of all the operations in the tangent linear model. Each of the DO loops and each of the subroutines in the tangent linear model have their adjoint image DO loop and subroutine. Therefore, we code the adjoint model directly from the discrete tangent linear model by rewriting the code of the tangent linear model sentence by sentence (i.e., on DO loop by one DO loop, subroutine by subroutine) in the opposite direction. This simplifies not only the complexity of constructing the adjoint model but also avoids the inconsistency generally arising from the derivation of the adjoint equations in analytic form followed by the discrete approximation.(Due to non-commutativity of discretization and adjoint operations).

4.3 Notational convention for variables and subroutines used in the adjoint model code

In a similar way as in the tangent linear model, we employed the same original variable names used in the nonlinear forward model for the corresponding adjoint variables in the adjoint model code. For instance, we use "U" for "$\hat{U}$", "$PKHT$" for "$(PK\hat{H}T)$", "$USTR2$" for "$(US\hat{T}R2)$", etc.. As in the TLM, this convention also means that the adjoint control variables in the adjoint model share the same common structure and same common block names as the GCM itself. So one needs to pay attention to it when carrying out four dimensional data assimilation experiments in which the adjoint model will be run in conjunction with the original GCM.

We also just add a "0" at the end of a variable name (in a similar way as done previously in the tangent linear model) to represent the corresponding basic state variable, such as using "$U0$" for "$U_{basic\ state}$", "$PKHT0$" for "$(PKHT)_{basic\ state}$", "$USTR20$" for "$(USTR2)_{basic\ state}$", etc..

For naming subroutines, we simply change the letter "L" at the beginning of the names of the tangent linear model subroutines to "A" and used them as corresponding adjoint model subroutine names. We also retain the adjoint subroutine names which do not exceed six letters to conform to the general FORTRAN language rule. For instance, for the subroutines of the original nonlinear model "$SUB1$", "$VADVCT$" and "$HADVCTT$", the corresponding names of the subroutines in the tangent linear model code are "$LSUB1$", "$LVADVC$" and "$LHADVC$", respectively, and the corresponding adjoint subroutine names are "$ASUB1$", "$AVADVC$" and "$AHADVC$", respectively.

4.4 Flow chart of the dynamical core of the adjoint model

In this subsection, we present the flow chart of the dynamical core of the adjoint model of NASA GEOS-1 GCM.

THE FLOW CHART OF THE ADJOINT MODEL OF THE DYNAMICAL CORE PACKAGE DESCRIBING THE TWO STEPS OF THE MATSUNO TIME INTEGRATION SCHEME AND THE LEAPFROG TIME INTEGRATION SCHEME. All COMPUTATIONS RELATED TO RETRIEVE THE BASIC STATE TRAJECTORY AND INITIALIZATION OF VARIABLES AND PARAMETERS ARE NOT INCLUDED.

ENTER

↓

CALL ASTEP: update adjoint prognostic fields backward one time-step, and **CALL ATMFIL** for applying the adjoint computations of Asselin time filter.

↓

CALL AFTDDT: apply the adjoint computation of the FFT scheme to the tendencies of the adjoint variables U, V, T, q over the high-latitude region.

↓

ADJOINT OF THE DYNAMICAL CORE, page 1.

↓

CALL AVADVC: compute the adjoint operations of the centered second-order vertical advections of perturbations of U, V, T, q.

↓

Compute the adjoint of the delta(PI*SIGMADOT)

↓

Compute the adjoint of the total perturbation pressure tendencies and the adiabatic perturbation pressure tendencies.

↓

CALL AHADVC: compute the adjoint operations of the perturbation moisture tendencies due to the horizonal advection processes (for the adiabatic version, skip this step).

↓

ADJOINT OF THE DYNAMICAL CORE, page 2.

CALL AHADVC: compute the adjoint operations of the perturbation temperature tendencies due to the horizonal advection processes.

CALL AHADVE: compute the adjoint operations of the tendencies of perturbations of height and wind due to the horizonal advection processes.

CALL ASUB2: compute the adjoint operator related to parameters alpha, beta, gamma, delta which are used in the Arakawa C-grid scheme.

ADJOINT OF THE DYNAMICAL CORE, page 3.

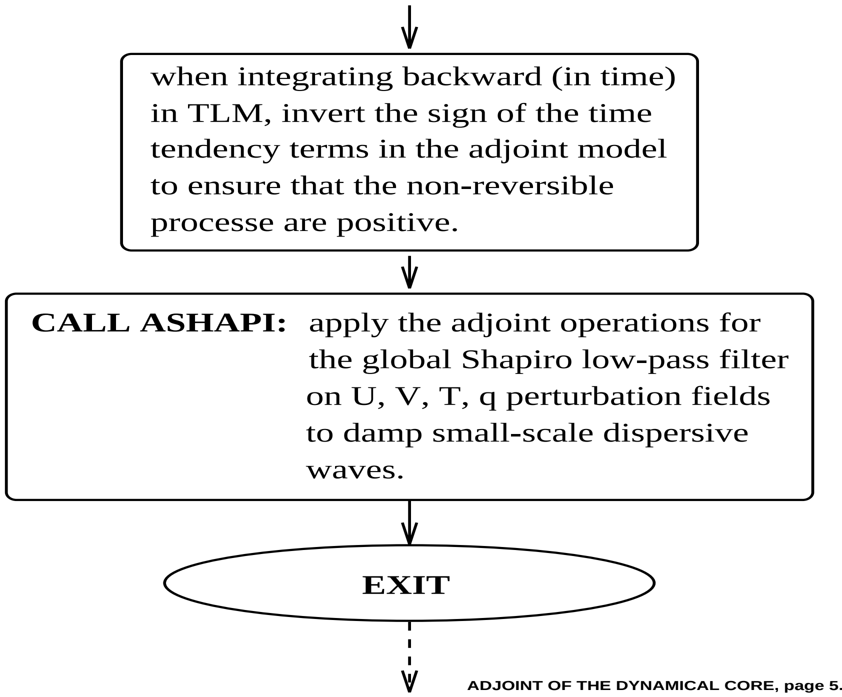
CAAL ASUB1: compute adjoint operator related to perturbation terms, kinetic energy, potential vorticity, ustar, vstar, which are used in the Arakawa C-grid scheme.

Adjoint computations of average perturbation mass to vorticity points.

Adjoint computations of the perturbation geopotential height.

CALL APKAP: apply the adjoint operations for calculating the perturbation ''P**KAPPA'' term.

ADJOINT OF THE DYNAMICAL CORE, page 4.



4.5 Verification of the correctness of the adjoint model

Integrating nonlinear model forward in time and its adjoint backwards in time, while forcing the r.h.s.of the adjoint model with difference between model and observations (see Eq. (39)), one can obtain value of gradient of cost function with respect to distributed control variables, which may consist of either the initial conditions or the initial conditions plus boundary conditions or model parameters. Since the adiabatic version of NASA GEOS-1 C-Grid GCM consists of thousands of lines of code, any minor coding error may cause the final gradient of cost function with respect to the control variables to be erroneous. Therefore, we need to verify the correctness of the linearization and adjoint coding segment by segment. Each segment may consist of either a subroutine or of several DO loops. For a detailed derivation of the adjoint model and verification of its correctness, see Navon et al.(1992).

The correctness of the adjoint of each operator was checked by applying the following identity (Navon et al., 1992)

$$(AQ)^{*T}(AQ) = Q^{*T}\left(A^{*T}(AQ)\right), \tag{42}$$

where Q represents the input of the original code, A represents either a single DO loop or a subroutine (See Navon et al. 1992). The left hand side involves only the tangent linear code, while the right hand side involves also adjoint code (A^{*T}). If Eq. (42) holds, the adjoint code is correct when compared with the TLM. In practice identity Eq. (42) holds only up to machine accuracy. In our verifications of the correctness of each segment of the adjoint model and the whole adjoint model, the LHS and the RHS of Eq.Eq. (42) attained 13 digits of accuracy which is near the machine accuracy limit of NASA Charney C-90 CRAY Computer which has intrinsic double precision.

In the subroutine "ASHAP", to ensure the correctness of the adjoint code check, we employ the "DOUBLE PRECISION" definition on some variables. In practice, to save computational cost, if do not need higher accuracy results of the adjoint model, one may omit use the "DOUBLE PRECISION" definition.

These results show that our adjoint code consists of absolutely the exact adjoint operators of the TLM of the adiabatic version of NASA GEOS-1 C-Grid GCM.

A gradient check (Fig. 3) was then performed to assess accuracy of the discrete adjoint model. This verification method is described below. First, we chose the cost function J as follows:

$$J(\mathbf{X}(t_0)) = \frac{1}{2}\sum_{r=0}^{R}\left(\mathbf{X}(t_r) - \mathbf{X}^{obs}(t_r)\right)^T \mathbf{W}(t_r)\left(\mathbf{X}(t_r) - \mathbf{X}^{obs}(t_r)\right) \tag{43}$$

where $\mathbf{X}(t_r)$ is an $N = (M(3K+1))$ component vector containing values of (u, v, T, Ps), with which NASA GEOS-1 GCM model is initialized, over all grid points and at all vertical levels at time t_r; M is the number of grid points at each vertical level; K is the number of vertical levels; R is the number of time levels for the analyzed fields in the assimilation window; t_r is a certain observation time in the assimilation window; $\mathbf{X}^{obs}(t_r)$ is the N-component vector of analyzed values of $\mathbf{X}$ over all grid points on all levels at time t_r; and $\mathbf{W}(t_r)$ is an $N \times N$ diagonal weighting matrix, where $\mathbf{W}_u$, $\mathbf{W}_v$, $\mathbf{W}_T$ and $\mathbf{W}_{Ps}$ are diagonal submatrices consisting of weighting factors for each variable, respectively. Their respective values (as used in gradient check calculation) were $\mathbf{W}_u = 10^{-3}\mathbf{I} \;\; s^2 m^{-2}$, $\mathbf{W}_v = 10^{-3}\mathbf{I} \;\; s^2 m^{-2}$, $\mathbf{W}_T = 10^{-1}\mathbf{I} \;\; K^{-2}$, $\mathbf{W}_{Ps}= 10^{-3}\mathbf{I} \;\; hPa^{-2}$. Then, let

$$J(\mathbf{X} + \alpha\mathbf{h}) = J(\mathbf{X}) + \alpha\mathbf{h}^T\nabla J(\mathbf{X}) + O(\alpha^2), \tag{44}$$

be a Taylor expansion of the cost function. Here α is a small scalar and $\mathbf{h}$ is a vector of unit length (such as $\mathbf{h} = \nabla J/\|\nabla J\|$). Rewriting the above formula we can define a function of α as

$$\Phi(\alpha) = \frac{J(\mathbf{X} + \alpha\mathbf{h}) - J(\mathbf{X})}{\alpha\mathbf{h}^T\nabla J(\mathbf{X})} = 1 + O(\alpha). \tag{45}$$

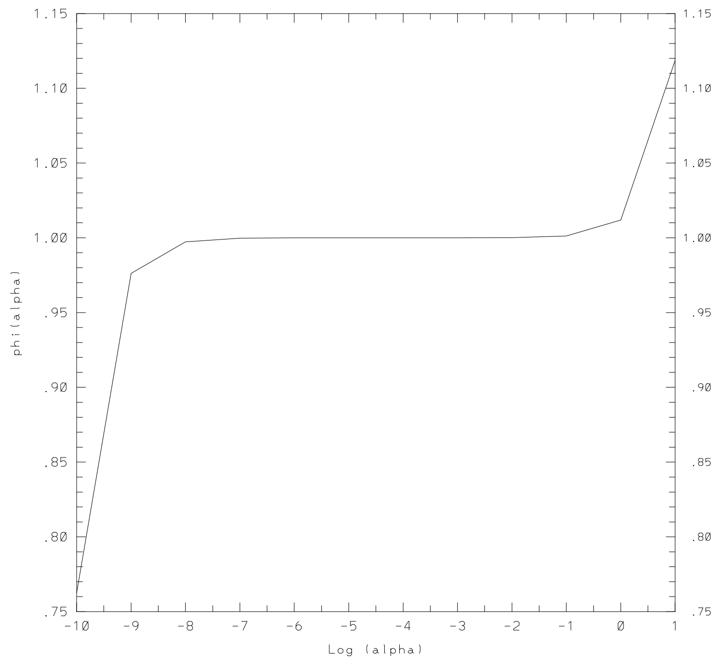


Figure 3: Variation of the $\phi(\alpha)$ with respect to $\log \alpha$ (gradient check of correctness of adjoint model). Integration period is 6 hours and $t = 6$ hours model generated observations were used. January 1, 1985 00Z ECMWF data was used as $t = 0$ observations. The first guess is the shifted 6-hour initial condition.

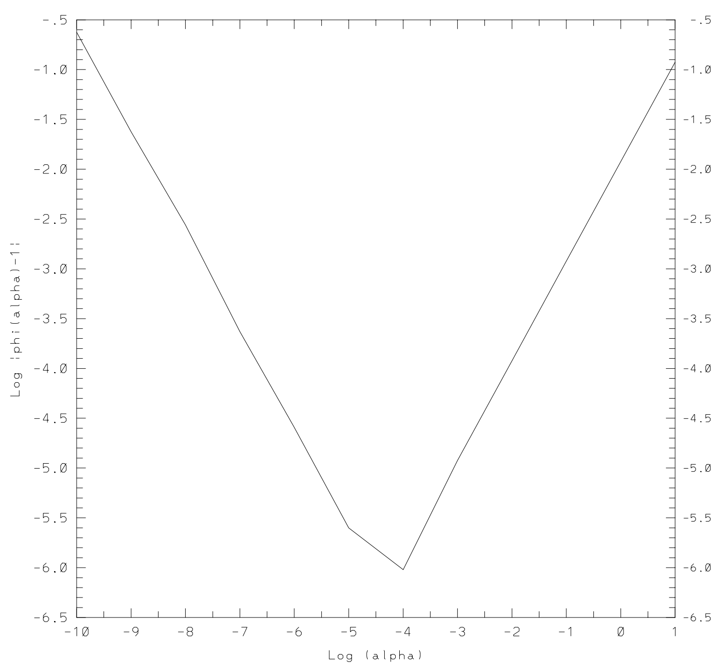


Figure 4: Variation of the $\log|\phi(\alpha) - 1|$ with respect to $\log \alpha$. Integration period is 6 hours and $t = 6$ hours model generated observations were used. January 1, 1985 00Z ECMWF data is used as $t = 0$ observations. The first guess is the shifted 6-hour initial condition.

For values of α which are small but not too close to the machine zero, one should expect to obtain a value for $\Phi(\alpha)$ which is close to unity. We obtained that the value of function $\phi(\alpha)$ equals unity to a high degree of accuracy when parameter α varied from 10^{-1} to 10^{-8}, and obeys a monotonically decreasing rule when α decreased over 12 orders of magnitude. From the residual of $\phi(\alpha)$(Fig. 4), we found that the residual tends linearly to zero . The gradient check verifies that adjoint model is correct and can be safely used to perform 4-D VDA experiments.

4.6 Some examples of coding the discrete adjoint model from the tangent linear model code

In the following we will provide some simple tutoring coding examples and technical methods which are very helpful for understanding the tangent linear model and the adjoint model coding techniques used in the adiabatic version of NASA GEOS-1 C-Grid GCM.

- **EXAMPLE 1:**

In the original nonlinear model

```
      DO 10 I=1, N-1
   10 X(I)=A*Y(I+1)
```

where A is a parameter, X and Y are N dimensional vectors. This DO loop is linear and do not need to be linearized. So the corresponding tangent linear code will remain identically the same as the original one,

```
      DO 10 I=1, N-1
   10 X(I)=A*Y(I+1)
```

The corresponding adjoint code form depends on whether the values of $Y(I)$ will be reused or not after this DO loop. If the values of $Y(I)$ will not be reused after this DO loop, the matrix form of this DO loop is

$$\begin{pmatrix} X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N-1) \end{pmatrix} = \begin{pmatrix} A & 0 & \cdots & 0 & 0 \\ 0 & A & \cdots & 0 & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & \cdots & 0 & A \end{pmatrix} \begin{pmatrix} Y(2) \\ Y(3) \\ \cdot \\ \cdot \\ \cdot \\ Y(N) \end{pmatrix} \tag{46}$$

The adjoint of the equation (46) can be written out directly as

$$\begin{pmatrix} Y(2) \\ Y(3) \\ \cdot \\ \cdot \\ \cdot \\ Y(N) \end{pmatrix} = \begin{pmatrix} A & 0 & 0 & \cdots & 0 & 0 \\ 0 & A & 0 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 0 & A \end{pmatrix} \begin{pmatrix} X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N) \end{pmatrix} \tag{47}$$

Equation (47) is equivalent to the following code

```
      DO 10 I=1, N-1
   10 Y(I+1)=A*X(I)
```

However, If the values of $Y(I)$ will be reused after this EXAMPLE 1 DO loop, the corresponding matrix form should modified to be

$$\begin{pmatrix} X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N-1) \\ Y(2) \\ Y(3) \\ \cdot \\ \cdot \\ \cdot \\ Y(N) \end{pmatrix} = \begin{pmatrix} A & 0 & 0 & \cdots & 0 & 0 \\ 0 & A & 0 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 0 & A \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} Y(2) \\ Y(3) \\ \cdot \\ \cdot \\ \cdot \\ Y(N) \end{pmatrix} \tag{48}$$

The adjoint of the equation (48) should be written out as

$$\begin{pmatrix} Y(2) \\ Y(3) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} = \begin{pmatrix} A & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & A & 0 & \cdots & 0 & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ . & . & . & \cdots & . & . & . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . & . & . & . & \cdots & . & . \\ . & . & . & \cdots & . & . & . & . & . & \cdots & . & . \\ 0 & 0 & 0 & \cdots & 0 & A & 0 & 0 & 0 & \cdots & 0 & 1 \end{pmatrix} \begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \\ Y(2) \\ Y(3) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} \tag{49}$$

Thus the adjoint code of Equation (49) will be

```
   DO 10 I=1, N-1
10 Y(I+1)=Y(I+1)+A*X(I)
```

Thus this is a different adjoint code compared to the code derived from Equation (/refeq47). The issue of identifying which variables on the right side of tangent linear code belong either to reused variables or non-reused variables is very important for adjoint code derivation.

- **EXAMPLE 2:**

In the original nonlinear model code and the tangent linear model code, a DO loop assumes the following form

```
   DO 10 I=1, N-1
10 X(I)=X(I)+A*Y(I+1)
```

The matrix form of the above DO loop (if the values of $Y(I)$ will not be reused) for the tangent linear model is

$$\begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 & A & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & A & \cdots & 0 & 0 \\ . & . & \cdots & . & . & . & . & \cdots & . & . \\ . & . & \cdots & . & . & . & . & \cdots & . & . \\ . & . & \cdots & . & . & . & . & \cdots & . & . \\ 0 & 0 & \cdots & 0 & 1 & 0 & 0 & \cdots & 0 & A \end{pmatrix} \begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \\ Y(2) \\ Y(3) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} \tag{50}$$

The adjoint operation matrix equation will be (by transposition, i.e., for tangent linear code $\mathbf{X} = L\tilde{\mathbf{X}}$, the adjoint code is $\tilde{\mathbf{X}} = L^T\mathbf{X}$.)

$$\begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \\ Y(2) \\ Y(3) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ . & . & \cdots & . & . \\ . & . & \cdots & . & . \\ . & . & \cdots & . & . \\ 0 & 0 & \cdots & 0 & 1 \\ A & 0 & \cdots & 0 & 0 \\ 0 & A & \cdots & 0 & 0 \\ . & . & \cdots & . & . \\ . & . & \cdots & . & . \\ . & . & \cdots & . & . \\ 0 & 0 & \cdots & 0 & A \end{pmatrix} \begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \end{pmatrix} \tag{51}$$

So the corresponding adjoint code is

```
      DO 10 I=1, N-1
   10 Y(I+1)=A*X(I)
```

Similarly, we may derive easily the adjoint code for this example when $Y(I)$ will be reused after this DO loop in the tangent linear model as

```
      DO 10 I=1, N-1
   10 Y(I+1)=Y(I+1)+A*X(I)
```

• **EXAMPLE 3:**

In the nonlinear model, a DO loop is

```
      DO 10 I=1, N
   10 Z(I)=X(I)*Y(I)
```

where X, Y and Z are N dimensional vector variables. The corresponding tangent linear model code is

```
      DO 10 I=1, N
   10 Z(I)=Y0(I)*X(I)+X0(I)*Y(I)
```

Recall from Subsections 3.3 and 4.3 that $Y0$ and $X0$ are basic state variables. If $Y(I)$ and $X(I)$ will not be reused anymore after this DO loop, the matrix form of this DO loop is

$$
\begin{pmatrix} Z(1) \\ Z(2) \\ \cdot \\ \cdot \\ \cdot \\ Z(N) \end{pmatrix} = \begin{pmatrix} Y0(1) & 0 & \cdots & 0 & X0(1) & 0 & \cdots & 0 \\ 0 & Y0(2) & \cdots & 0 & 0 & X0(2) & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & Y0(N) & 0 & 0 & \cdots & X0(N) \end{pmatrix} *
$$

$$
* \begin{pmatrix} X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N) \\ Y(1) \\ Y(2) \\ \cdot \\ \cdot \\ \cdot \\ Y(N) \end{pmatrix} \tag{52}
$$

The adjoint of the equation (52) can be written out directly as

$$
\begin{pmatrix} X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N) \\ Y(1) \\ Y(2) \\ \cdot \\ \cdot \\ \cdot \\ Y(N) \end{pmatrix} = \begin{pmatrix} Y0(1) & 0 & \cdots & 0 \\ 0 & Y0(2) & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & Y0(N) \\ X0(1) & 0 & \cdots & 0 \\ 0 & X0(2) & \cdots & 0 \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ \cdot & \cdot & \cdots & \cdot \\ 0 & 0 & \cdots & X0(N) \end{pmatrix} \begin{pmatrix} Z(1) \\ Z(2) \\ \cdot \\ \cdot \\ \cdot \\ Z(N) \end{pmatrix} \tag{53}
$$

The adjoint DO loop code is

```
      DO 10 I=1, N
      X(I)=Y0(I)*Z(I)
   10 Y(I)=X0(I)*Z(I)
```

If $Y(I)$ and $X(I)$ will be reused in future, the matrix form of the linearized code of EXAMPLE 3 is

$$\begin{pmatrix} Z(1) \\ Z(2) \\ . \\ . \\ . \\ Z(N) \\ X(1) \\ X(2) \\ . \\ . \\ . \\ X(N) \\ Y(1) \\ Y(2) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} = \begin{pmatrix} Y0(1) & 0 & \cdots & 0 & X0(1) & 0 & \cdots & 0 \\ 0 & Y0(2) & \cdots & 0 & 0 & X0(2) & \cdots & 0 \\ . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . \\ 0 & 0 & \cdots & Y0(N) & 0 & 0 & \cdots & X0(N) \\ 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} *$$

$$* \begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N) \\ Y(1) \\ Y(2) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} \qquad (54)$$

Then the adjoint matrix form is

$$\begin{pmatrix} X(1) \\ X(2) \\ . \\ . \\ . \\ X(N) \\ Y(1) \\ Y(2) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} = \begin{pmatrix} Y0(1) & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & Y0(2) & \cdots & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ . & . & \cdots & . & . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . & . & . & \cdots & . \\ 0 & 0 & \cdots & Y0(N) & 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ X0(1) & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & 0 \\ 0 & X0(2) & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 1 & \cdots & 0 \\ . & . & \cdots & . & . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . & . & . & \cdots & . \\ . & . & \cdots & . & . & . & \cdots & . & . & . & \cdots & . \\ 0 & 0 & \cdots & X0(N) & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 1 \end{pmatrix} * \\ * \begin{pmatrix} Z(1) \\ Z(2) \\ . \\ . \\ . \\ Z(N) \\ X(1) \\ X(2) \\ . \\ . \\ . \\ X(N) \\ Y(1) \\ Y(2) \\ . \\ . \\ . \\ Y(N) \end{pmatrix} \tag{55}$$

Thus the adjoint code can be obtained from above equation as

```
      DO 10 I=1, N
      X(I)=X(I)+YO(I)*Z(I)
   10 Y(I)=Y(I)+XO(I)*Z(I)
```

• **EXAMPLE 4:**

In last three examples, all the calculations do not have recursion relation. Here we present a simple example which exhibits a recursion relation. The DO loop code of the original nonlinear model and its tangent linear model is

```
   DO 10  I=1, N-1
10 Z(I+1)=Z(I)+YO(I)*X(I)+XO(I)*YY
```

where $Z(1)$, $X(I)$ and YY are control variables and will be reused in future, $Y0$ and $X0$ are two parameter vectors. This recursion loop can be expressed in matrix form as

$$\begin{pmatrix} Z(1) \\ Z(2) \\ Z(3) \\ \cdot \\ \cdot \\ \cdot \\ Z(N-1) \\ Z(N) \\ X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N-1) \\ YY \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 1 & 0 & 0 & \cdots & 0 & Y0(N-1) & X0(N-1) \\ 0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix} *$$

$$
* \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
0 & 0 & 0 & \cdots & 1 & 0 & 0 & 0 & \cdots & Y0(N-2) & 0 & X0(N-2) \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1
\end{pmatrix} * \cdots *
$$

$$
* \cdots * \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
1 & 0 & 0 & \cdots & 0 & 0 & Y0(1) & 0 & \cdots & 0 & 0 & X0(1) \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
Z(1) \\ Z(2) \\ \cdot \\ \cdot \\ \cdot \\ Z(N-2) \\ Z(N-1) \\ X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N-1) \\ YY
\end{pmatrix} \quad (56)
$$

The corresponding adjoint matrix equation is

$$
\begin{pmatrix} Z(1) \\ Z(2) \\ . \\ . \\ . \\ Z(N-2) \\ Z(N-1) \\ X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \\ YY \end{pmatrix}
=
\begin{pmatrix}
1 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & Y0(1) & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\
0 & X0(1) & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 1
\end{pmatrix} *
$$

$$
* \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 1 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\
0 & 0 & Y0(2) & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 1 & 0 \\
0 & 0 & X0(2) & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 1
\end{pmatrix} * \cdots *
$$

$$
\cdots \begin{pmatrix}
1 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 1 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 1 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
. & . & . & \cdots & . & . & . & . & \cdots & . & . \\
0 & 0 & 0 & \cdots & 0 & 0 & 0 & 0 & \cdots & 0 & 0 \\
0 & 0 & 0 & \cdots & 0 & Y0(N-1) & 0 & 0 & \cdots & 1 & 0 \\
0 & 0 & 0 & \cdots & 0 & X0(N-1) & 0 & 0 & \cdots & 0 & 1
\end{pmatrix}
\begin{pmatrix}
Z(1) \\ Z(2) \\ Z(3) \\ . \\ . \\ . \\ Z(N-1) \\ Z(N) \\ X(1) \\ X(2) \\ . \\ . \\ . \\ X(N-1) \\ YY
\end{pmatrix} \qquad (57)
$$

Finally, the adjoint code for this recursion DO loop is

```
      DO 10  I=N-1, 1, -1
      Z(I)=Z(I)+Z(I+1)
      X(I)=X(I)+Y0(I)*Z(I+1)
      YY=YY+X0(I)*Z(I+1)
   10 Z(I+1)=0.0
```

For this example, the DO loop variable I must evolve in the opposite direction of the variation of the DO loop variable in the tangent linear model code. As a matter of fact, the adjoint model is always integrated in the opposite direction of the tangent linear model. Thus, to avoid unexpected mistakes in the adjoint code and to conveniently identify and code the adjoint model, we follow the convention that the DO loop variables in all DO loops of adjoint model code should evolve in the opposite direction of the corresponding tangent linear model DO loops.

- **EXAMPLE 5:**

Here we present a simple example for a subroutine, the original code of which is

```
      SUBROUTINE SIM(X, Y, N)
      DIMENSION X(N), Y(N), Z(N*2)

      Y(N)=0.0
      DO 10 I=1, N-1
   10 Y(I)=X(I+1)**2
      DO 20 I=1, N
   20 Z(I)=X(I)**3
      DO 30 I=1, N
   30 Z(I+N)=X(I)**4
      DO 40 I=1, N
   40 Y(I)=Y(I)+10.0*Z(I)+8.0*Z(I+N)

      CALL OTHER(Y, N)

      RETURN
      END
```

In this subroutine, the input variable is "X", and the output variables are "X" and "Y". Its tangent linear code is

```
      SUBROUTINE LSIM(X, Y, N, X0, Y0)
      DIMENSION X(N), Y(N), Z(N*2), X0(N), Y0(N), Z0(N*2)

      Y(N)=0.0
      Y0(N)=0.0
      DO 10 I=1, N-1
      Y(I)=2.0*X0(I+1)*X(I+1)
   10 Y0(I)=X0(I+1)**2
      DO 20 I=1, N
      Z(I)=3.0*X0(I)**2*X(I)
   20 Z0(I)=X0(I)**3
      DO 30 I=1, N
      Z(I+N)=4.0*X0(I)**3*X(I)
```

```
   30 ZO(I+N)=XO(I)**4
      DO 40 I=1, N
      Y(I)=Y(I)+10.0*Z(I)+8.0*Z(I+N)
   40 YO(I)=YO(I)+10.0*ZO(I)+8.0*ZO(I+N)

      CALL LOTHER(Y, N, YO)

      RETURN
      END
```

The corresponding adjoint subroutine code(assume $X(I)$ will be reused in future) is

```
      SUBROUTINE ASIM(X, Y, N, XO, YO)
      DIMENSION X(N), Y(N), Z(N*2), XO(N), YO(N), ZO(N*2)

      CALL AOTHER(Y, N, YO)

      DO 40 I=N, 1, -1
      Z(I)=10.0*Y(I)
   40 Z(I+N)=8.0*Y(I)
      DO 30 I=N, 1, -1
   30 X(I)=X(I)+4.0*XO(I)**3*Z(I+N)
      DO 20 I=N, 1, -1
   20 X(I)=X(I)+3.0*XO(I)**2*Z(I)
      DO 10 I=N-1, 1, -1
   10 X(I+1)=X(I+1)+2.0*XO(I+1)*Y(I)

      RETURN
      END
```

- **EXAMPLE 6:**

Finally, we provide here a real subroutine in the adiabatic version of the NASA GEOS-1 GCM. This subroutine relates the horizontal advection calculations of momentum equations using the Arakawa-Lamb c-grid energy-conserving form.

———The original nonlinear forward model subroutine code is———

```
      SUBROUTINE HADVECT ( DUDT, DVDT, CONV,
     .                     FACTU, FACTV, FACTH, A,B,C,D,
     .                     ZKE, Q, PHI, PKZ,
     .                     USTAR,VSTAR,TBARU,TBARV, IM,JM,
     .                     UDOT,VDOT )

      DIMENSION  DUDT(IM,JM),  DVDT(IM,JM), CONV(IM,JM)
      DIMENSION     A(IM,JM),     B(IM,JM),    C(IM,JM),   D(IM,JM)
      DIMENSION   ZKE(IM,JM),     Q(IM,JM),  PHI(IM,JM), PKZ(IM,JM)
      DIMENSION USTAR(IM,JM), VSTAR(IM,JM)
      DIMENSION TBARU(IM,JM), TBARV(IM,JM)
      DIMENSION  UDOT(IM,JM),  VDOT(IM,JM)
      DIMENSION FACTU(IM,JM)
      DIMENSION FACTV(IM,JM)
      DIMENSION FACTH(IM,JM)

      PARAMETER ( ZERO   = 0.00 )
      PARAMETER ( AHALF  = 0.50 )
      PARAMETER ( ONE    = 1.00 )
      PARAMETER ( TWO    = 2.00 )
      PARAMETER ( THREE  = 3.00 )
      PARAMETER ( FOUR   = 4.00 )

      IMJM   = IM* JM
      IMJMM1 = IM*(JM-1)
      IMJMM2 = IM*(JM-2)
      IMJMM3 = IM*(JM-3)
      IMJMM4 = IM*(JM-4)

      CP = GETCON('CP')

C ***********************************************************************
C ****                COMPUTE HEIGHT TENDENCIES                     ****
C ***********************************************************************

      J = 2
      DO I=1,IMJMM2
      CONV(I,J) = - ( USTAR(I,J) - USTAR(I-1,J)
     .              + VSTAR(I,J) - VSTAR(I,J-1) )
```

```
      ENDDO

C FIX LONGITUDINAL BOUNDARIES
C ---------------------------
      IM1 = IM
      I   = 1
      DO J=2,JM-1
      CONV(I,J) = - ( USTAR(I,J) - USTAR(IM1,J)
     .              + VSTAR(I,J) - VSTAR(I,J-1) )
      ENDDO

C ***********************************************************************
C ****                COMPUTE U-WIND TENDENCIES                     ****
C ***********************************************************************

      J = 2
      DO  I=1,IMJMM2
            UDOT(I,J)     =  A (I   ,J  ) * VSTAR (I+1,J  )
     .                     + B (I   ,J  ) * VSTAR (I   ,J  )
     .                     + C (I   ,J  ) * VSTAR (I   ,J-1)
     .                     + D (I   ,J  ) * VSTAR (I+1,J-1)
     .                    - (ZKE(I+1,J  ) + PHI   (I+1,J  ) )
     .                    + (ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .     -CP*TBARU(I,J)* (PKZ(I+1,J  ) - PKZ   (I   ,J  ) )
      ENDDO

C FIX LONGITUDINAL BOUNDARIES
C ---------------------------
      DO J=2,JM-1
      IM1 = IM-1
      I   = IM
      IP1 = 1
            UDOT(I,J)     =  A (I   ,J  ) * VSTAR (IP1,J  )
     .                     + B (I   ,J  ) * VSTAR (I   ,J  )
     .                     + C (I   ,J  ) * VSTAR (I   ,J-1)
     .                     + D (I   ,J  ) * VSTAR (IP1,J-1)
     .                    - (ZKE(IP1,J  ) + PHI   (IP1,J  ) )
     .                    + (ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .     -CP*TBARU(I,J)* (PKZ(IP1,J  ) - PKZ   (I   ,J  ) )

      IM1 = IM
      I   = 1
      IP1 = 2
```

```
      UDOT(I,J)     =  A (I   ,J  ) * VSTAR (IP1,J  )
     .                 + B (I   ,J  ) * VSTAR (I   ,J  )
     .                 + C (I   ,J  ) * VSTAR (I   ,J-1)
     .                 + D (I   ,J  ) * VSTAR (IP1,J-1)
     .               - (ZKE(IP1,J  ) + PHI   (IP1,J  ) )
     .               + (ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .     -CP*TBARU(I,J)* (PKZ(IP1,J  ) - PKZ   (I   ,J  ) )
      ENDDO

C ***********************************************************************
C ****                COMPUTE V-WIND TENDENCIES                     ****
C ***********************************************************************

      J=JM-1
      DO I=1, IM
      VDOT(I, J)=ZERO
      END DO

        J = 2
        DO  I=1,IMJMM3
            VDOT(I,J)     = - C (I   ,J+1) * USTAR (I   ,J+1)
     .                      - D (I-1,J+1) * USTAR (I-1,J+1)
     .                      - A (I-1,J  ) * USTAR (I-1,J  )
     .                      - B (I   ,J  ) * USTAR (I   ,J  )
     .                 - ( ZKE(I   ,J+1) + PHI   (I   ,J+1) )
     .                 + ( ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .      -CP*TBARV(I,J)*( PKZ(I   ,J+1) - PKZ   (I   ,J  ) )
      ENDDO

      DO  J=2,JM-2
      IM1 = IM
      I   = 1
      IP1 = 2
            VDOT(I,J)     = - C (I   ,J+1) * USTAR (I   ,J+1)
     .                      - D (IM1,J+1) * USTAR (IM1,J+1)
     .                      - A (IM1,J  ) * USTAR (IM1,J  )
     .                      - B (I   ,J  ) * USTAR (I   ,J  )
     .                 - ( ZKE(I   ,J+1) + PHI   (I   ,J+1) )
     .                 + ( ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .      -CP*TBARV(I,J)*( PKZ(I   ,J+1) - PKZ   (I   ,J  ) )

      IM1 = IM-1
      I   = IM
```

```
      IP1 = 1
            VDOT(I,J)      = - C (I   ,J+1) * USTAR (I   ,J+1)
     .                       - D (IM1,J+1) * USTAR (IM1,J+1)
     .                       - A (IM1,J  ) * USTAR (IM1,J  )
     .                       - B (I   ,J  ) * USTAR (I   ,J  )
     .                    - ( ZKE(I   ,J+1) + PHI   (I   ,J+1) )
     .                    + ( ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .      -CP*TBARV(I,J)*( PKZ(I   ,J+1) - PKZ   (I   ,J  ) )
      ENDDO

C ***********************************************************************
C ****                     APPLY SCALE FACTORS                       ****
C ***********************************************************************

      DO I=1,IMJMM2
      CONV(I,2) =             CONV(I,2) * FACTH(I,2)
      DUDT(I,2) = DUDT(I,2) + UDOT(I,2) * FACTU(I,2)
      DVDT(I,2) = DVDT(I,2) + VDOT(I,2) * FACTV(I,2)
      ENDDO

      DO I=1, IMJM
      UDOT(I, 1)=ZERO
      VDOT(I, 1)=ZERO
      END DO

      RETURN
      END
```

———Its corresponding TLM subroutine code is accordingly———

```
      SUBROUTINE LHADVE ( DUDT, DVDT, CONV,
     .                     FACTU, FACTV, FACTH, A,B,C,D,
     .                     ZKE, PHI, PKZ,
     .                     USTAR,VSTAR,TBARU,TBARV, IM,JM,
     .                     UDOT,VDOT,
     .                    DUDTO, DVDTO, CONVO,
     .                     AO,BO,CO,DO,
     .                     ZKEO, PHIO, PKZO,
     .                     USTARO,VSTARO,TBARUO,TBARVO,
     .                     UDOTO,VDOTO)

      DIMENSION  DUDT(IM,JM),  DVDT(IM,JM), CONV(IM,JM)
      DIMENSION  DUDTO(IM,JM),  DVDTO(IM,JM), CONVO(IM,JM)
      DIMENSION     A(IM,JM),     B(IM,JM),    C(IM,JM),   D(IM,JM)
      DIMENSION    AO(IM,JM),    BO(IM,JM),   CO(IM,JM),  DO(IM,JM)
      DIMENSION   ZKE(IM,JM),   PHI(IM,JM), PKZ(IM,JM)
      DIMENSION  ZKEO(IM,JM),  PHIO(IM,JM),PKZO(IM,JM)
      DIMENSION USTAR(IM,JM), VSTAR(IM,JM)
      DIMENSION USTARO(IM,JM), VSTARO(IM,JM)
      DIMENSION TBARU(IM,JM), TBARV(IM,JM)
      DIMENSION TBARUO(IM,JM), TBARVO(IM,JM)
      DIMENSION  UDOT(IM,JM),  VDOT(IM,JM)
      DIMENSION  UDOTO(IM,JM),  VDOTO(IM,JM)
      DIMENSION FACTU(IM,JM)
      DIMENSION FACTV(IM,JM)
      DIMENSION FACTH(IM,JM)

      PARAMETER ( ZERO   = 0.00 )
      PARAMETER ( AHALF  = 0.50 )
      PARAMETER ( ONE    = 1.00 )
      PARAMETER ( TWO    = 2.00 )
      PARAMETER ( THREE  = 3.00 )
      PARAMETER ( FOUR   = 4.00 )

      IMJM   = IM* JM
      IMJMM2 = IM*(JM-2)

      CP = GETCON('CP')
      DO I=1, IMJM
```

```
      UDOT(I, 1)=ZERO
      VDOT(I, 1)=ZERO
      END DO

C **********************************************************************
C ****                  COMPUTE HEIGHT TENDENCIES                   ****
C **********************************************************************

      DO J = 2, JM-1
      DO I=2,IM
      CONV(I,J) = - ( USTAR(I,J) - USTAR(I-1,J)
     .              + VSTAR(I,J) - VSTAR(I,J-1) )
      CONVO(I,J) = - ( USTARO(I,J) - USTARO(I-1,J)
     .              + VSTARO(I,J) - VSTARO(I,J-1) )
      ENDDO
      ENDDO

      J=JM-1
      DO I=1, IM
      VDOT(I, J)=ZERO
      END DO

C FIX LONGITUDINAL BOUNDARIES
C ---------------------------
      IM1 = IM
      I   = 1
      DO J=2,JM-1
      CONV(I,J) = - ( USTAR(I,J) - USTAR(IM1,J)
     .              + VSTAR(I,J) - VSTAR(I,J-1) )
      CONVO(I,J) = - ( USTARO(I,J) - USTARO(IM1,J)
     .              + VSTARO(I,J) - VSTARO(I,J-1) )
      ENDDO

C **********************************************************************
C ****                  COMPUTE U-WIND TENDENCIES                   ****
C **********************************************************************

      DO J=2,JM-1
      DO  I=1,IM-1
            UDOT(I,J)     =
     .                        A (I   ,J   ) * VSTARO(I+1,J   )
     .                      + AO(I   ,J   ) * VSTAR (I+1,J   )
     .                      + B (I   ,J   ) * VSTARO(I   ,J   )
```

```
     .                     + B0(I   ,J   ) * VSTAR (I   ,J   )
     .                     + C (I   ,J   ) * VSTAR0(I   ,J-1)
     .                     + C0(I   ,J   ) * VSTAR (I   ,J-1)
     .                     + D (I   ,J   ) * VSTAR0(I+1,J-1)
     .                     + D0(I   ,J   ) * VSTAR (I+1,J-1)
     .                   - (ZKE(I+1,J   ) + PHI    (I+1,J   ) )
     .                   + (ZKE(I   ,J   ) + PHI    (I   ,J   ) )
     .      -CP*(TBARU(I,J)* (PKZ0(I+1,J   ) - PKZ0   (I   ,J   ) )
     .      +TBARU0(I,J)* (PKZ(I+1,J   ) - PKZ    (I   ,J   ) ))
             UDOT0(I,J)      =   A0(I   ,J   ) * VSTAR0(I+1,J   )
     .                     + B0(I   ,J   ) * VSTAR0(I   ,J   )
     .                     + C0(I   ,J   ) * VSTAR0(I   ,J-1)
     .                     + D0(I   ,J   ) * VSTAR0(I+1,J-1)
     .                   - (ZKE0(I+1,J   ) + PHI0   (I+1,J   ) )
     .                   + (ZKE0(I   ,J   ) + PHI0   (I   ,J   ) )
     .      -CP*TBARU0(I,J)* (PKZ0(I+1,J   ) - PKZ0   (I   ,J   ) )
      ENDDO
      ENDDO

C FIX LONGITUDINAL BOUNDARIES
C ---------------------------
      I   = IM
      IP1 = 1
      DO J=2,JM-1
            UDOT(I,J)      =
     .                       A (I   ,J   ) * VSTAR0(IP1,J   )
     .                     + A0(I   ,J   ) * VSTAR (IP1,J   )
     .                     + B (I   ,J   ) * VSTAR0(I   ,J   )
     .                     + B0(I   ,J   ) * VSTAR (I   ,J   )
     .                     + C (I   ,J   ) * VSTAR0(I   ,J-1)
     .                     + C0(I   ,J   ) * VSTAR (I   ,J-1)
     .                     + D (I   ,J   ) * VSTAR0(IP1,J-1)
     .                     + D0(I   ,J   ) * VSTAR (IP1,J-1)
     .                   - (ZKE(IP1,J   ) + PHI    (IP1,J   ) )
     .                   + (ZKE(I   ,J   ) + PHI    (I   ,J   ) )
     .      -CP*(TBARU(I,J)* (PKZ0(IP1,J   ) - PKZ0   (I   ,J   ) )
     .      +TBARU0(I,J)* (PKZ(IP1,J   ) - PKZ    (I   ,J   ) ))
            UDOT0(I,J)      =  A0(I   ,J   ) * VSTAR0(IP1,J   )
     .                     + B0(I   ,J   ) * VSTAR0(I   ,J   )
     .                     + C0(I   ,J   ) * VSTAR0(I ,J-1)
     .                     + D0(I   ,J   ) * VSTAR0(IP1,J-1)
     .                   - (ZKE0(IP1,J   ) + PHI0(IP1,J   ) )
     .                   + (ZKE0(I   ,J   ) + PHI0   (I   ,J   ) )
```

```
     .     -CP*TBARUO(I,J)* (PKZO(IP1,J  ) - PKZO  (I   ,J) )
      ENDDO

C *********************************************************************
C ****                COMPUTE V-WIND TENDENCIES                     ****
C *********************************************************************

        DO J=2, JM-2
        DO  I=2,IM
            VDOT(I,J)     =
     .                      - CO(I   ,J+1) * USTAR (I   ,J+1)
     .                      - C (I   ,J+1) * USTARO(I   ,J+1)
     .                      - DO(I-1,J+1) * USTAR (I-1,J+1)
     .                      - D (I-1,J+1) * USTARO(I-1,J+1)
     .                      - AO(I-1,J  ) * USTAR (I-1,J  )
     .                      - A (I-1,J  ) * USTARO(I-1,J  )
     .                      - BO(I   ,J  ) * USTAR (I   ,J  )
     .                      - B (I   ,J  ) * USTARO(I   ,J  )
     .                   - ( ZKE(I   ,J+1) + PHI   (I   ,J+1) )
     .                   + ( ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .      -CP*(TBARVO(I,J)*( PKZ(I   ,J+1) - PKZ   (I   ,J  ) )
     .      +TBARV(I,J)*( PKZO(I   ,J+1) - PKZO  (I   ,J  ) ))
            VDOTO(I,J)     = - CO(I   ,J+1) * USTARO(I   ,J+1)
     .                      - DO(I-1,J+1) * USTARO(I-1,J+1)
     .                      - AO(I-1,J  ) * USTARO(I-1,J  )
     .                      - BO(I   ,J  ) * USTARO(I   ,J  )
     .                   - ( ZKEO(I   ,J+1) + PHIO  (I   ,J+1) )
     .                   + ( ZKEO(I   ,J  ) + PHIO  (I   ,J  ) )
     .      -CP*TBARVO(I,J)*( PKZO(I   ,J+1) - PKZO  (I   ,J  ) )
      ENDDO
      ENDDO

      IM1 = IM
      I   = 1
      DO  J=2,JM-2
            VDOT(I,J)     =
     .                      - CO(I   ,J+1) * USTAR (I   ,J+1)
     .                      - C (I   ,J+1) * USTARO(I   ,J+1)
     .                      - DO(IM1,J+1) * USTAR (IM1,J+1)
     .                      - D (IM1,J+1) * USTARO(IM1,J+1)
     .                      - AO(IM1,J  ) * USTAR (IM1,J  )
     .                      - A (IM1,J  ) * USTARO(IM1,J  )
     .                      - BO(I   ,J  ) * USTAR (I   ,J  )
```

```
     .                    - B (I   ,J  ) * USTARO(I   ,J  )
     .                 - ( ZKE(I   ,J+1) + PHI   (I   ,J+1) )
     .                 + ( ZKE(I   ,J  ) + PHI   (I   ,J  ) )
     .      -CP*(TBARVO(I,J)*( PKZ(I   ,J+1) - PKZ   (I   ,J  ) )
     .      +TBARV(I,J)*( PKZO(I   ,J+1) - PKZO  (I   ,J  ) ))
            VDOTO(I,J)     = - CO(I   ,J+1) * USTARO(I   ,J+1)
     .                   - DO(IM1,J+1) * USTARO(IM1,J+1)
     .                   - AO(IM1,J  ) * USTARO(IM1,J  )
     .                   - BO(I   ,J  ) * USTARO(I   ,J  )
     .                 - ( ZKEO(I   ,J+1) + PHIO  (I   ,J+1) )
     .                 + ( ZKEO(I   ,J  ) + PHIO  (I   ,J  ) )
     .      -CP*TBARVO(I,J)*( PKZO(I   ,J+1) - PKZO  (I   ,J  ) )
      ENDDO

C ***********************************************************************
C ****                    APPLY SCALE FACTORS                        ****
C ***********************************************************************

      DO I=1,IMJMM2
      CONV(I,2) =             CONV(I,2) * FACTH(I,2)
      CONVO(I,2) =             CONVO(I,2) * FACTH(I,2)
      DUDT(I,2) = DUDT(I,2) + UDOT(I,2) * FACTU(I,2)
      DUDTO(I,2) = DUDTO(I,2) + UDOTO(I,2) * FACTU(I,2)
      DVDT(I,2) = DVDT(I,2) + VDOT(I,2) * FACTV(I,2)
      DVDTO(I,2) = DVDTO(I,2) + VDOTO(I,2) * FACTV(I,2)
      ENDDO

      DO I=1, IMJM
      UDOT(I, 1)=ZERO
      VDOT(I, 1)=ZERO
      END DO

      RETURN
      END
```

———The corresponding adjoint subroutine code is accordingly———

```
      SUBROUTINE AHADVE ( DUDT, DVDT, CONV,
     .                    FACTU, FACTV, FACTH, A,B,C,D,
     .                    ZKE, PHI, PKZ,
     .                    USTAR,VSTAR,TBARU,TBARV, IM,JM,
     .                    UDOT,VDOT,
     .                    AO,BO,CO,DO,
     .                    PKZO,
     .                    USTARO,VSTARO,TBARUO,TBARVO)

      DIMENSION  DUDT(IM,JM),  DVDT(IM,JM), CONV(IM,JM)
      DIMENSION     A(IM,JM),     B(IM,JM),    C(IM,JM),   D(IM,JM)
      DIMENSION    AO(IM,JM),    BO(IM,JM),   CO(IM,JM),  DO(IM,JM)
      DIMENSION   ZKE(IM,JM),   PHI(IM,JM), PKZ(IM,JM)
      DIMENSION  PKZO(IM,JM)
      DIMENSION USTAR(IM,JM), VSTAR(IM,JM)
      DIMENSION USTARO(IM,JM), VSTARO(IM,JM)
      DIMENSION TBARU(IM,JM), TBARV(IM,JM)
      DIMENSION TBARUO(IM,JM), TBARVO(IM,JM)
      DIMENSION  UDOT(IM,JM),  VDOT(IM,JM)
      DIMENSION FACTU(IM,JM)
      DIMENSION FACTV(IM,JM)
      DIMENSION FACTH(IM,JM)

      PARAMETER ( ZERO   = 0.00 )
      PARAMETER ( AHALF  = 0.50 )
      PARAMETER ( ONE    = 1.00 )
      PARAMETER ( TWO    = 2.00 )
      PARAMETER ( THREE  = 3.00 )
      PARAMETER ( FOUR   = 4.00 )

      IMJM   = IM* JM
      IMJMM2 = IM*(JM-2)

      CP = GETCON('CP')

C *********************************************************************
C ****                    APPLY SCALE FACTORS                      ****
C *********************************************************************
```

```
      DO I=1, IMJM
      UDOT(I, 1)=ZERO
      VDOT(I, 1)=ZERO
      END DO

      DO I=IMJMM2, 1, -1
      VDOT(I,2)=DVDT(I,2) * FACTV(I,2)

      UDOT(I,2)=DUDT(I,2) * FACTU(I,2)

      CONV(I,2)=CONV(I,2) * FACTH(I,2)
      END DO

C ***********************************************************************
C ****                   COMPUTE V-WIND TENDENCIES                  ****
C ***********************************************************************

      DO I=1, IMJM
      ZKE(I, 1)=ZERO
      PHI(I, 1)=ZERO
      A(I, 1)=ZERO
      B(I, 1)=ZERO
      C(I, 1)=ZERO
      D(I, 1)=ZERO
      END DO

      IM1 = IM
      I   = 1
      DO  J=JM-2, 2, -1
      AM10=-CP*VDOT(I,J)
      AM20=AM10*TBARV0(I,J)
      TBARV(I,J)=AM10*( PKZ0(I  ,J+1) - PKZ0  (I  ,J  ) )
      PKZ(I  ,J+1)=PKZ(I  ,J+1)+AM20
      PKZ   (I  ,J  )=PKZ   (I  ,J  )-AM20
      ZKE(I, J)=ZKE(I, J)+VDOT(I,J)
      ZKE(I  ,J+1)=ZKE(I  ,J+1)-VDOT(I,J)
      PHI   (I  ,J  )=PHI   (I  ,J  )+VDOT(I,J)
      PHI   (I  ,J+1)=PHI   (I  ,J+1)-VDOT(I,J)
      B (I  ,J  )=B (I  ,J  )-VDOT(I,J) * USTAR0(I  ,J  )
      USTAR (I  ,J  )=USTAR (I  ,J  )-B0(I  ,J  ) * VDOT(I,J)
      A (IM1,J  )=A (IM1,J  )-VDOT(I,J) * USTAR0(IM1,J  )
      USTAR (IM1,J  )=USTAR (IM1,J  )-A0(IM1,J  ) * VDOT(I,J)
      D (IM1,J+1)=D (IM1,J+1)-VDOT(I,J) * USTAR0(IM1,J+1)
```

```
      USTAR (IM1,J+1)=USTAR (IM1,J+1)-D0(IM1,J+1) * VDOT(I,J)
      C (I  ,J+1)=C (I  ,J+1)-VDOT(I,J) * USTAR0(I  ,J+1)
      USTAR (I  ,J+1)=USTAR (I  ,J+1)-C0(I  ,J+1) * VDOT(I,J)
      END DO

      DO J=JM-2, 2, -1
      DO  I=IM, 2, -1

      AM10=-CP*VDOT(I,J)
      AM20=AM10*TBARV0(I,J)
      TBARV(I,J)=AM10*( PKZ0(I  ,J+1) - PKZ0  (I  ,J  ) )
      PKZ(I  ,J+1)=PKZ(I  ,J+1)+AM20
      PKZ   (I  ,J  )=PKZ   (I  ,J  )-AM20
      ZKE(I, J)=ZKE(I, J)+VDOT(I,J)
      ZKE(I  ,J+1)=ZKE(I  ,J+1)-VDOT(I,J)
      PHI   (I  ,J  )=PHI   (I  ,J  )+VDOT(I,J)
      PHI   (I  ,J+1)=PHI   (I  ,J+1)-VDOT(I,J)
      B (I  ,J  )=B (I  ,J  )-VDOT(I,J) * USTAR0(I  ,J  )
      USTAR (I  ,J  )=USTAR (I  ,J  )-B0(I  ,J  ) * VDOT(I,J)
      A (I-1,J  )=A (I-1,J  )-VDOT(I,J) * USTAR0(I-1,J  )
      USTAR (I-1,J  )=USTAR (I-1,J  )-A0(I-1,J  ) * VDOT(I,J)
      D (I-1,J+1)=D (I-1,J+1)-VDOT(I,J) * USTAR0(I-1,J+1)
      USTAR (I-1,J+1)=USTAR (I-1,J+1)-D0(I-1,J+1) * VDOT(I,J)
      C (I  ,J+1)=C (I  ,J+1)-VDOT(I,J) * USTAR0(I  ,J+1)
      USTAR (I  ,J+1)=USTAR (I  ,J+1)-C0(I  ,J+1) * VDOT(I,J)
      END DO
      END DO

C FIX LONGITUDINAL BOUNDARIES
C ---------------------------

      I   = IM
      IP1 = 1
      DO J=JM-1, 2, -1
      AM10=-CP*UDOT(I,J)
      AM20=AM10*TBARU0(I,J)
      PKZ(IP1,J  )=PKZ(IP1,J  )+AM20
      PKZ   (I  ,J  )=PKZ   (I  ,J  )-AM20
      TBARU(I,J)=AM10*(PKZ0(IP1,J  ) - PKZ0  (I  ,J  ) )
      ZKE(I  ,J  )=ZKE(I  ,J  )+UDOT(I,J)
      PHI   (I  ,J  )=PHI   (I  ,J  )+UDOT(I,J)
      ZKE(IP1,J  )=ZKE(IP1,J  )-UDOT(I,J)
      PHI   (IP1,J  )=PHI   (IP1,J  )-UDOT(I,J)
```

```
      VSTAR (IP1,J-1)=VSTAR (IP1,J-1)+D0(I   ,J   ) * UDOT(I,J)
      D (I   ,J   )=D (I   ,J   )+VSTAR0(IP1,J-1) * UDOT(I,J)
      VSTAR (I   ,J-1)=VSTAR (I   ,J-1)+C0(I   ,J   ) * UDOT(I,J)
      C (I   ,J   )=C (I   ,J   )+VSTAR0(I   ,J-1) * UDOT(I,J)
      VSTAR (I   ,J   )=VSTAR (I   ,J   )+B0(I   ,J   ) * UDOT(I,J)
      B (I   ,J   )=B (I   ,J   )+VSTAR0(I   ,J   ) * UDOT(I,J)
      VSTAR (IP1,J   )=VSTAR (IP1,J   )+A0(I   ,J   ) * UDOT(I,J)
      A (I   ,J   )=A (I   ,J   )+VSTAR0(IP1,J   ) * UDOT(I,J)
      END DO

C ***********************************************************************
C ****                COMPUTE U-WIND TENDENCIES                     ****
C ***********************************************************************
      DO J=JM-1, 2, -1
      DO  I=IM-1, 1, -1
      AM10=-CP*UDOT(I,J)
      AM20=AM10*TBARU0(I,J)
      PKZ(I+1,J   )=PKZ(I+1,J   )+AM20
      PKZ   (I   ,J   )=PKZ   (I   ,J   )-AM20
      TBARU(I,J)=AM10*(PKZ0(I+1,J   ) - PKZ0   (I   ,J   ) )
      ZKE(I   ,J   )=ZKE(I   ,J   )+UDOT(I,J)
      PHI   (I   ,J   )=PHI   (I   ,J   )+UDOT(I,J)
      ZKE(I+1,J   )=ZKE(I+1,J   )-UDOT(I,J)
      PHI   (I+1,J   )=PHI   (I+1,J   )-UDOT(I,J)
      VSTAR (I+1,J-1)=VSTAR (I+1,J-1)+D0(I   ,J   ) * UDOT(I,J)
      D (I   ,J   )=D (I   ,J   )+VSTAR0(I+1,J-1) * UDOT(I,J)
      VSTAR (I   ,J-1)=VSTAR (I   ,J-1)+C0(I   ,J   ) * UDOT(I,J)
      C (I   ,J   )=C (I   ,J   )+VSTAR0(I   ,J-1) * UDOT(I,J)
      VSTAR (I   ,J   )=VSTAR (I   ,J   )+B0(I   ,J   ) * UDOT(I,J)
      B (I   ,J   )=B (I   ,J   )+VSTAR0(I   ,J   ) * UDOT(I,J)
      VSTAR (I+1,J   )=VSTAR (I+1,J   )+A0(I   ,J   ) * UDOT(I,J)
      A (I   ,J   )=A (I   ,J   )+VSTAR0(I+1,J   ) * UDOT(I,J)
      END DO
      END DO

C FIX LONGITUDINAL BOUNDARIES
C ---------------------------
      IM1 = IM
      I   = 1
      DO J=JM-1, 2, -1
      USTAR(I,J)=USTAR(I,J)-CONV(I,J)
      USTAR(IM1,J)=USTAR(IM1,J)+CONV(I,J)
      VSTAR(I,J)=VSTAR(I,J)-CONV(I,J)
```

```
      VSTAR(I,J-1)=VSTAR(I,J-1)+CONV(I,J)
      END DO
C ************************************************************************
C ****                COMPUTE HEIGHT TENDENCIES                       ****
C ************************************************************************
      DO J=JM-1, 2, -1
      DO I=IM, 2, -1
      USTAR(I,J)=USTAR(I,J)-CONV(I,J)
      USTAR(I-1,J)=USTAR(I-1,J)+CONV(I,J)
      VSTAR(I,J)=VSTAR(I,J)-CONV(I,J)
      VSTAR(I,J-1)=VSTAR(I,J-1)+CONV(I,J)
      END DO
      END DO

      DO I=1, IMJM
      UDOT(I, 1)=ZERO
      VDOT(I, 1)=ZERO
      END DO

      RETURN
      END
```

References

Arakawa, A., and V. R. Lamb, 1981: A potential enstrophy and energy conserving scheme for the shallow water equations, *Mon. Wea. Rev.*, **109**, 18-36.

Asselin, R., 1972: Frequency filter for time integrations, *Mon. Wea. Rev.*, **100**, 487-490.

Brown, J. A., and K. Campana, 1978: An economical time-differencing system for numerical weather prediction, *Mon. Wea. Rev.*, **106**, 1125-1136.

Burridge, D. M., and J. Haseler, 1977: A model for medium range weather forecasting - adiabatic formulation, *Tech. Report, No. 4, European Center for Medium Range Weather Forecasts*, Bracknell, Berkshire, UK.

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

Fox-Rabinovitz, M., H. M. Helfand, A. Hou, L. L. Takacs, and A. Molod, 1991: Numerical experiments on forecasting, climate simulation and data assimilation with the new 17 layer GLA GCM, *Proceedings of the AMS Ninth Conference on Numerical Weather Prediction*, 21-25 October 1991, Denver, CO, 506-509.

Gates, W. L., 1992: AMIP: The Atmospheric Model Intercomparison Project, *Bull. Ameri. Meteor. Soc.*, **73**, 1962-1970.

Helfand, H. M., M. Fox-Rabinovitz, L. Takacs, and A. Molod, 1991: Simulation of the planetary boundary layer and turbulence in the GLA GCM, *Proceedings of the AMS Ninth Conference on Numerical Weather Prediction*, 21-25 October 1991, Denver, CO, 514-517.

Kalnay, E. M. Kanamitsu, J. Pfaendtner, J. Sela, M. Suarez, J. Stackpole, J. Tuccillo, L. Umscheid, and D. Williamson, 1989: Rules for the interchange of physical parameterizations, *Bull. Ameri. Meteor. Soc.*, **70**, 620-622.

Lacarra, J.-F., and O. Talagrand, 1988: Short-range evolution of small perturbations in a barotropic model, *Tellus*, **40A**, 81-95.

Li, Y., I. M. Navon, J. R. Bates, S. Moorthi, R. W. Higgins, X. Zou and W. Yang, 1993: Development of the Adjoint model of the Adiabatic Version of the 3-D NASA/GLA Semi-Lagrangian Semi-implicit GCM, *Technical Report*, **FSU-SCRI-93-49**, April 1993.

Li, Y., I. M. Navon, W. Yang, X. Zou, J. R. Bates, S. Moorthi, and R. W. Higgins, 1994: Four-Dimensional Variational Data Assimilation Experiments with a Multilevel semi-Lagrangian semi-Implicit GCM. *Mon. Wea. Rev.*, **122**, 5, 966-983.

Navon, I. M., 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.

Rabier, F., and P. Courtier, 1992: Four-Dimensional assimilation in the presence of baroclinic instability, *Q. J. R. Meteorol. Soc.*, **118**, 649-672.

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

Schubert, S. D., J. Pfaendtner, and R. Rood, 1993: An assimilated data set for Earth Science applications, *Bull. Ameri. Meteor. Soc.*, **74**, 2331-2342.

Schubert, S., M. Suarez, C. K. Park, and S. Moorthi, 1993: GCM simulations of interseasonal variability in the Pacific/North American region, *J. Atmos. Sci.*, **50**, 1991-2007.

Suarez, M. J., and L. L. Takacs, 1994: Documentation of the ARIES/GEOS Dynamical Core, Version 2, May 19, 1994.

Takacs, L. L., A. Molod, and T. Wang, 1994: Goddard Earth Observing System (GEOS) General Circulation Model (GCM), Version 1, May, 17, 1994.

Yang, Weiyu, I. Michael Navon, 1995: A New Hessian Preconditioning Method Applied to Variational Data Assimilation Experiments using Adiabatic Version of NASA/GEOS-1 GCM, *Technical Report*, **FSU-SCRI-95-03**, Supercomputer Computations Research Institute, Florida State University, 43pp.

Yang, Weiyu, I. Michael Navon, 1995: Extending a Hessian Preconditioning Method to 4-D Variational Data Assimilation Experiments using the Adiabatic Version of the NASA/GEOS-1 C-Grid GCM, *WMO International Symposium on Assimilation of Observations in Meteorology and Oceanography*, Tokyo, Japan, March 13-17, 1995, also in the *WMO International Workshop on Imbalances of Slowly Varying Components of Predictable Atmospheric Motions*, Beijing, China, March 7-10, 1995.

Yang, Weiyu, I. Michael Navon, and Philippe Courtier, 1995: A New Hessian Preconditioning Method Applied to Variational Data Assimilation Experiments using the Adiabatic Version of NASA/GEOS-1 GCM, Submitted to *Mon. Wea. Rev.*.