Computational Optimization and Applications, 4, 241–262 (1995)

A Truncated Newton Optimization Algorithm in Meteorology Applications with Analytic Hessian/Vector Products

ZHI WANG
Center for Analysis and Prediction of Storms, University of Oklahoma, Norman, OK 73019

I.M. NAVON
Department of Mathematics and Supercomputer Computations, Research Institute, Florida State University, Tallahassee, FL 32306

X. ZOU
National Center for Atmospheric Research, Boulder, Colorado 80307

F.X. LE DIMET
Grenoble, Laboratoire de Modelisation et Calcul, B.P. 53X, 38041, Grenoble Cedex, France

Received March 9, 1994; Revised October 20, 1994

Abstract. A modified version of the truncated-Newton algorithm of Nash ([24], [25], [29]) is presented differing from it only in the use of an exact Hessian vector product for carrying out the large-scale unconstrained optimization required in variational data assimilation. The exact Hessian vector product is obtained by solving an optimal control problem of distributed parameters. (i.e. the system under study occupies a certain spatial and temporal domain and is modeled by partial differential equations) The algorithm is referred to as the adjoint truncated-Newton algorithm. The adjoint truncated-Newton algorithm is based on the first and the second order adjoint techniques allowing to obtain a better approximation to the Newton line search direction for the problem tested here. The adjoint truncated-Newton algorithm is applied here to a limited-area shallow water equations model with model generated data where the initial conditions serve as control variables. We compare the performance of the adjoint truncated-Newton algorithm with that of the original truncated-Newton method [29] and the LBFGS (Limited Memory BFGS) method of Liu and Nocedal [23]. Our numerical tests yield results which are twice as fast as these obtained by the truncated-Newton algorithm and are faster than the LBFGS method both in terms of number of iterations as well as in terms of CPU time.

1. Introduction

Most of numerical weather prediction models are distributed parameter systems [1, 51]. The term "distributed parameter system" implies that the response (e.g. cost function) of the system is governed by a set of partial differential equations and parameters embedded in the equations are spatially and time dependent [1, 38]. The issue of 4-dimensional variational data assimilation applied to the shallow water equations model has been the subject of several recent papers such as [4], [22], [48], [52], [53], [54] etc. The large-scale unconstrained minimization of a cost functional arising in the application of optimal control of distributed parameters assumes the following form

$$\min_U J(U) = \min_U \left\{ \frac{1}{2} \int_{t_0}^{t_f} [W(C\vec{X} - \vec{X}^0), (C\vec{X} - \vec{X}^0)]\, dt \right\}, \tag{1}$$

where $\vec{X}$ is the state variable vector in a Hilbert space $\mathcal{X}$, the inner product of the space of observations is denoted by $[\cdot, \cdot]$ using an Euclidean norm, $[t_0, t_f]$ denotes the assimilation

window where t_0 and t_f are the initial and final time and W is a weighting function usually taken to be the inverse of the covariance matrix. The objective function $J(U)$ is the weighted sum of the squares of the distance between the model solution and available observations distributed in space and time. $\vec{X}^0$ is the observation vector, the operator C represents an operator from the space of the model solution $\vec{X}$ to the space of observations [54]. The control variable U and the state vector $\vec{X}$ satisfy model equations

$$\frac{\partial \vec{X}}{\partial t} = F(\vec{X}), \tag{2}$$

$$\vec{X}(t_0) = U, \tag{3}$$

where t is the time, and $F \in C^2$ is a function of $\vec{X}$, where C^2 denotes a set of twice continuously differentiable functions. Therefore, for any initial condition given by Eq. (3), Eq. (2) has an unique solution, $\vec{X}$.

The control variables can be initial conditions or initial plus boundary conditions [49] or parameters [50] to be estimated. For simplicity, only initial conditions are taken as control variables.

Full-Newton descent methods (i.e., based on dense matrix factorizations) have never been used in a realistic large-scale optimal control of distributed parameter problems due to the fact that computation and storage of the Hessian are too costly to be practical [9]. Among the feasible methods for large-scale unconstrained minimization are (a) the limited memory conjugate gradient method ([34], [43]); (b) quasi-Newton type algorithms ([5], [9], [14], [37]); (c) limited-memory quasi-Newton methods such as LBFGS algorithm ([23], [36]) and (d) truncated-Newton algorithms ([26], [28], [29], [32], [41], [42]).

The truncated-Newton algorithm can be applied to many problems such as the above unconstrained minimization problem (Eq. (1)) and parallel minimization problems ([30], [31]). The main purpose of the present work is to propose a modified version of the truncated-Newton algorithm of Nash [29] which uses the second order adjoint technique to obtain an exact Hessian vector product required in calculating the Newton line search direction and to compare the numerical results obtained by the adjoint truncated-Newton algorithm with these obtained by the truncated-Newton of Nash [29] and by the LBFGS algorithm [23]. The adjoint truncated-Newton algorithm is only useful for optimal control problems where adjoint model codes exist or could be easily derived.

The truncated-Newton algorithms attempt to blend the rapid (quadratic) convergence rate of classic Newton method with feasible storage [29] and computational requirements [32] for large-scale unconstrained minimization problems. When these algorithms are used for the large-scale unconstrained minimization [53], the Hessian vector product is usually obtained by applying a finite-difference approximation technique to the gradient of the cost function with respect to the initial conditions, while the gradient is calculated by using the first order adjoint technique. Other alternatives for obtaining the Hessian/vector product such as automatic differentiation ([11], [13]), and higher order finite-differencing approximations exist. Many automatic differentiation variants are competitive in terms of complexity with analytic approaches. However, higher order finite-differencing approach is too costly to apply to optimal control problem of variational data assimilation.

We have recently learned that Santosa and Symes [39, 40, 45] also used the adjoint method to calculate the Hessian vector product in an application to inverse problems of reflection seismology. They have also applied the adjoint technique to calculate the Hessian vector

product required in a different version of the truncated Newton method [6]. Their results indicate that the Hessian vector product calculated by this technique yields considerable accuracy [39]. In our paper we derived the second order adjoint technique in a different manner than theirs and applied it to a different version of the truncated Newton method [29]. We also carried out an accuracy analysis of the Hessian vector product calculation.

We will briefly describe in Section 2 the shallow water equations model, then introduce in Section 3 the first order adjoint and the second order adjoint techniques used in the variational data assimilation to obtain an exact Hessian vector product. In Section 4 we present numerical results obtained by using the adjoint truncated-Newton algorithm, and compare them with the results obtained by using the limited memory quasi-Newton algorithm of Liu and Nocedal [23] and the truncated-Newton algorithm of Nash [29]. A detailed discussion concerning the accuracy of the finite-difference approximation of the Hessian vector product and the sources of error related to this approximation is provided. Summary and conclusions as well as topics for further research related to the adjoint truncated-Newton approach are presented in Section 5.

2. The shallow water equations model

In this section, we introduce the shallow water equations model which is used as the test model in this paper.

The shallow water equations model equations may be written as

$$\frac{\partial u}{\partial t} = -u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} + fv - \frac{\partial \phi}{\partial x}, \tag{4}$$

$$\frac{\partial v}{\partial t} = -u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} - fu - \frac{\partial \phi}{\partial y}, \tag{5}$$

$$\frac{\partial \phi}{\partial t} = -\frac{\partial (u\phi)}{\partial x} - \frac{\partial (v\phi)}{\partial y}. \tag{6}$$

where x, y, t, u, v, ϕ and f are the x and the y coordinates, the time, the two components of the horizontal velocity, the geopotential field and the Coriolis factor, respectively. The spatial domain is a 6000 km $\times$ 4400 km channel. The initial conditions are those of Grammeltvedt [18]. The southern and northern boundaries are rigid walls where the normal velocity components vanish, and it is assumed that the flow is periodic in the west-east direction with a wave length equal to the length of the channel. The time and space increments used in the model were

$$\Delta x = 300 \text{ km}, \ \Delta y = 220 \text{ km}, \ \Delta t = 600 \text{ s}, \tag{7}$$

which result in 21 $\times$ 21 grid point locations in the channel and a dimension of 1083 of the initial condition vector $U = (u, v, \phi)^t$ where superscript t denotes the transpose.

The shallow water equations model is non-dimensionalized by the following scaling

$$\begin{aligned} &\Delta x' = \Delta x/H, \quad \Delta y' = \Delta y/H, \quad \Delta t' = \Delta t U/H, \\ &u' = u/U, \quad v' = v/U, \quad \phi' = \phi/U^2, \\ &f' = fH/U, \quad g' = gH/U^2, \end{aligned} \tag{8}$$

where $H = 10^5$ and $U = 10^3$.

The cost function is defined in Eq. (1) where $W_\phi = 10^{-4}\ \mathrm{m}^{-4}\mathrm{s}^4$ and $W_u = W_v = 10^{-2}\ \mathrm{m}^{-2}\mathrm{s}^2$ are weighting factors which may in a more rigorous approach be taken as the inverse of estimates of the statistical root-mean-square observational errors on geopotential and wind components, respectively.

3. The adjoint Newton algorithm

3.1. Description of the first order adjoint and the second order adjoint model theories

Let us consider a perturbation U' on the initial condition U in Eq. (3). The perturbed solution $\vec{X} + \hat{X}$ satisfies the equation:

$$\begin{aligned}\frac{\partial(\vec{X}+\hat{X})}{\partial t} &= F(\vec{X}+\hat{X}),\\ &= F(\vec{X}) + \frac{\partial F}{\partial \vec{X}}\hat{X} + \frac{1}{2}\hat{X}^*\frac{\partial F^2}{\partial \vec{X}^2}\hat{X} + O(\|\hat{X}\|^3),\\ \vec{X}(t_0) + \hat{X}(t_0) &= U + U',\end{aligned}$$

where $\hat{X}^*$ denotes the complex conjugate transpose and $\|\hat{X}\|$ denotes the Euclidean norm of "$\hat{X}$". Using the model Eqs. (2) and (3) retaining the first order terms in $\hat{X}$, the above equation becomes:

$$\frac{\partial \hat{X}}{\partial t} = \frac{\partial F}{\partial \vec{X}}\hat{X}, \tag{9}$$

$$\hat{X}(t_0) = U'. \tag{10}$$

Equations (9) and (10) define the tangent linear model (TLM) of the nonlinear forward model given by Eqs. (2) and (3) in the vicinity of the particular solution $\vec{X}$ (see [46]) and the perturbation U' on the initial condition U in Eq. (3) has become the initial condition of the TLM. The TLM describes the temporal evolution of the perturbation $\hat{X}$, to the second order with respect to the initial perturbation U'. Appendix A provides a simple example which illustrates the accuracy of the TLM. For discussions related to the validity of TLM models, please see [21].

For a perturbation U' on the initial condition U in Eq. (3), the exact evolution of the perturbation due to this perturbation U' is given by the difference, $\vec{X}_2 - \vec{X}_1$, between two solutions of the model Eq. (2) with initial conditions $\vec{X}_1(t_0) = U$ and $\vec{X}_2(t_0) = U + U'$, respectively. The tangent linear variable defined by Eqs. (9)–(10) only gives a second order approximation, with respect to U', to the exact evolution of the perturbation due to the perturbation U'. The Gateaux derivative, $[\nabla_U J, U']$, of the cost function J is given by

$$\delta J = [\nabla_U J, U'] = \int_{t_0}^{t_f} [W(C\vec{X} - \vec{X}^0), C\hat{X}]\, dt, \tag{11}$$

To exhibit the linear dependence of δJ with respect to U' and consequently in order to compute the gradient of J, we introduce the adjoint variable P. Taking the inner product

of Eq. (9) with $-P$ and integrating between times t_0 and t_f gives

$$\int_{t_0}^{T} \left[-P, \frac{\partial \hat{X}}{\partial t} \right] dt = \int_{t_0}^{T} \left[-P, \frac{\partial F}{\partial \vec{X}} \hat{X} \right] dt, \tag{12}$$

Integrating by parts, one obtains

$$[-P(t_f), \hat{X}(t_f)] + [P(t_0), \hat{X}(t_0)] + \int_{t_0}^{t_f} \left[\hat{X}, \frac{\partial P}{\partial t} \right] = -\int_{t_0}^{t_f} \left[\left(\frac{\partial F}{\partial \vec{X}} \right)^* P, \hat{X} \right] dt, \tag{13}$$

or

$$[-P(t_f), \hat{X}(t_f)] + [P(t_0), \hat{X}(t_0)] = \int_{t_0}^{t_f} \left[\hat{X}, -\frac{\partial P}{\partial t} - \left(\frac{\partial F}{\partial \vec{X}} \right)^* P \right] dt, \tag{14}$$

Now let us define P as being the solution of

$$-\frac{\partial P}{\partial t} = \left(\frac{\partial F}{\partial \vec{X}} \right)^* P + C^* W (C\vec{X} - \vec{X}^0), \tag{15}$$

$$P(t_f) = 0, \tag{16}$$

Then from Eq. (11) one obtains

$$\delta J(U, U') = [\nabla_U J, U'] = [U', P(t_0)] \tag{17}$$

and

$$\nabla_U J = P(t_0). \tag{18}$$

Therefore the gradient of the cost function is obtained by a backwards integration of the adjoint system (15–16). The dependence of J on the initial condition U is implicit via X.

It is important to realize that the TLM defined by Eqs. (9)–(10) is only second order accurate except when the original model equations Eq. (2) are linear. Although the TLM is used to derive Eq. (18), Eq. (18) is exact. See Appendix A for an analytical example.

Let us denote the first order adjoint variable after a perturbation U' on the initial condition U by $P_{(U+U')}$. Expanding $\nabla_{U+U'} J$ around U in Taylor series, results in

$$\nabla_{U+U'} J = \nabla_U J + \nabla_U^2 J \cdot U' + \frac{1}{2} (U')^* \frac{\partial^3 J(\vec{\xi}_0)}{\partial U^3} U', \tag{19}$$

where $\nabla_U^2 J$ is the Hessian matrix of second derivatives of the cost function with respect to the initial condition and $\vec{\xi}_0$ is a point in the interval $[U, U + U']$. From Eq. (18), we know that

$$\nabla_{U+U'} J = P_{(U+U')}(t_0). \tag{20}$$

Therefore $P_{(U+U')}(t_0)$ may be written as

$$P_{(U+U')}(t_0) = P(t_0) + \hat{P}(t_0) + P_e(t_0), \tag{21}$$

where $P(t_0) = \nabla_U J$, $\hat{P}(t_0) = \nabla_U^2 J \cdot U'$ and $P_e(t_0) = 0.5(U')^* \partial^3 J(\vec{\xi}_0)/\partial U^3 U'$. Expanding $\partial F/\partial \vec{X}|_{(\vec{X}+\hat{X})}$ around $\vec{X}$, results in

$$\left.\frac{\partial F}{\partial \vec{X}}\right|_{(\vec{X}+\hat{X})} = \frac{\partial F}{\partial \vec{X}} + \frac{\partial^2 F}{\partial \vec{X}^2}\hat{X} + \frac{1}{2}\hat{X}^* \left.\frac{\partial^3 F}{\partial \vec{X}^3}\right|_{\vec{\xi}_1} \hat{X}. \tag{22}$$

where $\vec{\xi}_1$ is a point in the interval $[\vec{X}, \vec{X} + \hat{X}]$. Substituting Eqs. (21) and (22) into Eqs. (15)–(16), one obtains

$$-\frac{\partial(P + \hat{P} + P_e)}{\partial t} = \left\{\frac{\partial F}{\partial \vec{X}} + \frac{\partial^2 F}{\partial \vec{X}^2}\hat{X} + \frac{1}{2}\hat{X}^* \frac{\partial^3 F}{\partial \vec{X}^3}\hat{X}\right\}(P + \hat{P} + P_e) + C^* W(C(\vec{X} + \hat{X}) - \vec{X}^0), \tag{23}$$

i.e.

$$\begin{aligned} &-\frac{\partial P}{\partial t} - \frac{\partial \hat{P}}{\partial t} - \frac{\partial P_e}{\partial t} \\ &= \left(\frac{\partial F}{\partial \vec{X}}\right)^* P + \left(\frac{\partial F}{\partial \vec{X}}\right)^* \hat{P} + \left(\frac{\partial^2 F}{\partial \vec{X}^2}\hat{X}\right)^* P + \left(\frac{1}{2}\hat{X}^* \frac{\partial^3 F}{\partial \vec{X}^3}\hat{X}\right) P \\ &\quad + \left(\left(\frac{\partial^2 F}{\partial \vec{X}^2}\hat{X}\right)^* + \left(\frac{1}{2}\hat{X}_t^* \frac{\partial F^3}{\partial \vec{X}^3}\hat{X}_t\right)^*\right)\hat{P} + \left(\frac{\partial F}{\partial \vec{X}} + \frac{\partial^2 F}{\partial \vec{X}^2}\hat{X} + \frac{1}{2}\hat{X}^* \frac{\partial^3 F}{\partial \vec{X}^3}\hat{X}\right) P_e \\ &\quad + C^* W(C\vec{X} - \vec{X}^0) + C^* W C\hat{X}, \quad P(t_f) + \hat{P}(t_f) + P_e(t_f) = 0. \end{aligned} \tag{24}$$

If we let

$$\begin{aligned} -\frac{\partial P_e}{\partial t} &= \left(\frac{1}{2}\hat{X}^* \frac{\partial^3 F}{\partial \vec{X}^3}\hat{X}\right) P + \left(\left(\frac{\partial^2 F}{\partial \vec{X}^2}\hat{X}\right)^* + \left(\frac{1}{2}\hat{X}_t^* \frac{\partial F^3}{\partial \vec{X}^2}\hat{X}_t\right)^*\right)\hat{P} \\ &\quad + \left(\frac{\partial F}{\partial \vec{X}} + \frac{\partial^2 F}{\partial \vec{X}^2}\hat{X} + \frac{1}{2}\hat{X}^* \frac{\partial^3 F}{\partial \vec{X}^3}\hat{X}\right) P_e \end{aligned} \tag{25}$$

$$P_e(t_f) = 0. \tag{26}$$

Making use of Eqs. (25) and (15), one obtains the following equations from Eq. (24)

$$\frac{\partial \hat{P}}{\partial t} = \left(\frac{\partial F}{\partial \vec{X}}\right)^* \hat{P} + \left(\frac{\partial^2 F}{\partial \vec{X}^2}\hat{X}\right)^* P + C^* W C\hat{X}, \tag{27}$$

$$\hat{P}(t_f) = 0. \tag{28}$$

Equations (27) and (28) define the second order adjoint model of Eqs. (2) and (3). One backwards integration of the second order adjoint model yields an exact Hessian vector product:

$$\nabla_U^2 J \cdot U' = \hat{P}(t_0). \tag{29}$$

Santosa and Symes [39] showed that the exact Hessian can be computed at a modest cost for a problem similar to the one considered here. Their paper also showed the benefit of computing the Hessian in an optimization. The second order adjoint process is illustrated

in Appendix A using a simple nonlinear model with exact solutions for both the nonlinear model and the adjoint models.

For the derivation of the second order adjoint model for the shallow water equations model, see [48].

3.2. *The adjoint truncated-Newton method*

We provide a brief description of the adjoint truncated-Newton algorithm which differs from the truncated-Newton algorithm ([29], [32]) only in the Hessian vector calculation required for solving the Newton equations at the k-th iteration

$$G_k \vec{d}_k = -\vec{g}_k, \tag{30}$$

where G_k is the Hessian of the cost function, $\vec{d}_k$ is the linear search direction and $\vec{g}_k = \nabla J(\vec{U}_k)$ is the gradient of the cost function with respect to the initial conditions. For a complete description of the truncated-Newton algorithm, see Nash [29].

The main steps of the adjoint truncated-Newton algorithm are as follows:

(1) Choose $\vec{U}_0$, an initial guess to the minimizer $\vec{U}^*$ and set the iteration counter to $k = 0$.
(2) Test $\vec{U}_k$ for convergence. If the following convergence criterion is satisfied

$$\|\vec{g}_k\| < 10^{-5} \cdot \|\vec{g}_0\|, \tag{31}$$

then stop. Otherwise continue.
(3) Solve approximately the Newton Eqs. (30) using a preconditioned modified-Lanczos algorithm where the Hessian vector product is obtained using a backwards integration of the second order adjoint model given by Eqs. (27) and (28).
(4) Set $k = k + 1$ and update

$$\vec{U}_{k+1} = \vec{U}_k + \alpha_k \vec{d}_k, \tag{32}$$

where α_k is the step-size obtained by conducting a line search using Davidon's cubic interpolation method [5]. Go to step 2.

The computational cost required to obtain the Hessian vector product is similar for both the finite-difference approach and the second order adjoint approach. The second order adjoint approach requires us to integrate the original nonlinear model and its tangent linear model forward in time once and integrate the first order adjoint model and the second order adjoint model backwards in time once. The finite-difference approach requires the integration of the original nonlinear model forward in time and the first order adjoint model backwards in time twice. The computational costs for each model integration are comparable.

4. Numerical results obtained using the adjoint truncated-Newton algorithm

In this section we display numerical results obtained by applying the adjoint truncated-Newton algorithm for the large-scale unconstrained minimization of the functional in the variational data assimilation and compare the results obtained by the adjoint truncated-Newton algorithm with these obtained by both the truncated-Newton and the LBFGS algorithms.

4.1. *Application of the adjoint truncated-Newton algorithm to variational data assimilation*

A simple experiment was conducted applying the adjoint truncated-Newton algorithm to minimize the cost functional J given by Eq. (1) in the variational data assimilation using the shallow water equations model. The experiment is devised as follows: the model generated values starting from the initial condition of Grammeltvedt [18] are used as observations, the initial guess is a randomly perturbed Grammeltvedt initial condition, and the length of the time assimilation window is 10 hours. We know the exact solution, i.e. that the value of the cost function at the minimum must be zero since we must retrieve the original initial conditions. The adjoint truncated-Newton algorithm described in Section 3.2 is used here for large-scale unconstrained minimization in the variational data assimilation experiment. The maximum number of conjugate gradient inner-iterations allowed for each adjoint truncated-Newton algorithm iteration is chosen as 50. The number of BFGS corrections that is to be stored in memory is denoted by M.

Computations were performed on the CRAY-YMP supercomputer at the Supercomputer Computations Research Institute in Florida State University. All the routines are coded in single precision FORTRAN. The runs were made on the CRAY-YMP supercomputer, for which the relative machine precision ϵ is approximately 10^{-14}. The variation of the objective function scaled by its initial value (J/J_0) as well as that of the norm of the gradient also scaled by its initial value ($\|\vec{g}\|/\|\vec{g}_0\|$) as a function of the number of iterations are displayed in Fig. 1, respectively. Figure 1 shows that after 16 minimization iterations the value of the cost function and the norm of the gradient were reduced by 10 and 6 orders of magnitude, respectively. At this stage the prescribed convergence criterion given by Eq. (31) is satisfied. The CPU times used by the adjoint truncated Newton algorithm and usual truncated Newton algorithm are 10.817 s and 24.491 s, respectively. The CPU times used by LBFGS of Liu and Nocedal [23] with number of LBFGS corrections of 5 is 15.585 s. The rms error, $\sqrt{\|\phi_r - \phi_u\|^2/N}$, between retrieved geopotential field ϕ_r after 16 iterations and the unperturbed geopotential field ϕ_u is 0.9069 m^2/s^2 which is 3 orders of magnitude smaller than that of the perturbations, where N is the number of components in ϕ_r. We conclude therefore that the adjoint truncated-Newton algorithm performs well both in terms of CPU time and the number of iterations.

4.2. *Numerical results*

In this section we compare the numerical behavior of the adjoint truncated-Newton unconstrained minimization algorithm with those of other robust large-scale unconstrained minimization methods. The methods tested are:

(1) Truncated-Newton—the truncated-Newton method of Nash ([27], [29]).
(2) LBFGS—the limited memory quasi-Newton method of Liu and Nocedal [23].

The test problem is the same as that described in Section 4.1. Computational efficiency and accuracy were used as leading criteria. For the adjoint truncated-Newton, truncated-Newton and LBFGS algorithms, the same convergence criterion as set by Eq. (31) is applied. The numerical results obtained are displayed in Table 2.

The first column displays the unconstrained minimization algorithms tested. The second column displays the parameters used for each algorithm. The third, fourth and eighth

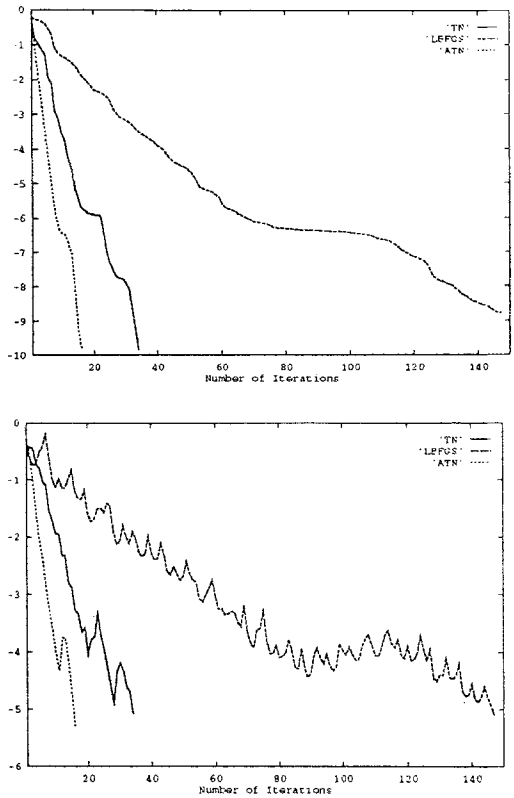

Figure 1. The variations of the log of the scaled cost function (J_k/J_0) (a) and the scaled gradient norm ($\|\vec{g}_k\|/\|\vec{g}_0\|$) (b) with the number of iterations using algorithms: adjoint truncated-Newton (dotted line), truncated-Newton (solid line) and LBFGS (dashed line), respectively.

columns record the number of iterations, the number of function calls, and the CPU time in seconds required to satisfy prescribed convergence criteria, respectively. The fifth column records the total number of conjugate gradient iterations (NCG) used to determine the Newton descent directions. The next two columns display the scaled cost function (J_k/J_0) and the scaled gradient norm ($\|\vec{g}_k\|/\|\vec{g}_0\|$) at the end of the assimilation process, respectively.

Both the adjoint truncated-Newton and truncated-Newton algorithms obtain the Newton descent direction by solving approximately the Newton equations using a truncated conjugate gradient algorithm. They only differ in the calculation of the Hessian vector product. However their relative performances turn out to be dramatically different. Table 2 indicates that for different maximum number of conjugate gradient inner-iterations allowed for each inner iteration, the adjoint truncated-Newton algorithm outperforms the truncated-Newton algorithm. When the maximum number of conjugate gradient inner-iterations is 5, the truncated-Newton algorithm stopped without satisfying the prescribed convergence criterion. When the maximum number of conjugate gradient inner-iterations is 4, the truncated-Newton algorithm required three times the CPU time used by the adjoint truncated-Newton algorithm to satisfy the same convergence criteria. In terms of CPU times the adjoint truncated-Newton, truncated-Newton and LBFGS algorithms yield optimal results when the maximum number of conjugate gradient inner-iterations is 4 and 3

Table 1. Errors between the true evolution given by Eq. (64) and the approximation given by Eq. (63) at $U = 0.9$ and $t = 0.5$ due to different perturbations on the initial condition U. $\vec{X}_2$ and $\vec{X}_1$ denote the model solutions starting from the initial conditions $U + U'$ and U, respectively.

$\lvert\vec{X}_2 - \vec{X}_1 - \hat{X}\rvert$	U'
$1.2195493765050 \times 10^{-1}$	1.
$1.5854141894547 \times 10^{-3}$	0.1
$1.6344476184732 \times 10^{-5}$	0.01
$1.6395183253561 \times 10^{-7}$	0.001
$1.6400272229676 \times 10^{-9}$	0.0001
$1.6399903435054 \times 10^{-11}$	0.00001
$1.6206145007512 \times 10^{-13}$	0.000001
$1.2021521456654 \times 10^{-15}$	0.0000001

Table 2. Numerical results of algorithms: adjoint truncated-Newton (ATN), truncated-Newton (TN) and LBFGS for the minimization of the cost function in the variational data assimilation problem when observations are model generated and control variables are the initial conditions. MCGI and M denote the maximum number of conjugate gradient inner-iterations and the number of BFGS corrections that is to be stored in memory, respectively. NCG, NFC and Iter. denote the total number of conjugate gradient iterations, the number of function calls and the number of iterations, respectively.

Algorithms		Iter.	NFC	NCG	J_k/J_0	$\lVert\vec{g}_k\rVert/\lVert\vec{g}_0\rVert$	CPU
ATN	MCGI=1	96	139	93	3.622×10^{-9}	8.552×10^{-6}	23.615 s
	MCGI=2	42	43	84	8.597×10^{-11}	6.922×10^{-6}	13.407 s
	MCGI=3	30	46	89	1.716×10^{-10}	9.347×10^{-6}	14.116 s
	MCGI=4	15	16	58	6.540×10^{-10}	8.467×10^{-6}	7.866 s
	MCGI=5	20	43	91	2.590×10^{-10}	8.467×10^{-6}	13.992 s
	MCGI=50	16	17	85	1.485×10^{-10}	4.822×10^{-6}	10.817 s
TN	MCGI=1	104	164	104	3.211×10^{-10}	6.455×10^{-6}	35.862 s
	MCGI=2	50	51	100	1.291×10^{-10}	8.577×10^{-6}	20.663 s
	MCGI=3	32	38	88	6.440×10^{-10}	8.785×10^{-6}	17.125 s
	MCGI=4	38	87	106	1.114×10^{-10}	5.929×10^{-6}	25.202 s
	MCGI=5	25	74	75	1.008×10^{-6}	1.113×10^{-3}	Failure
	MCGI=50	34	69	116	1.485×10^{-10}	8.156×10^{-6}	24.491 s
LBFGS	$M = 3$	163	167		7.892×10^{-9}	7.804×10^{-6}	16.511 s
	$M = 4$	157	167		2.211×10^{-9}	9.655×10^{-6}	16.724 s
	$M = 5$	147	153		1.658×10^{-9}	8.138×10^{-6}	15.585 s
	$M = 6$	153	159		1.585×10^{-9}	7.368×10^{-6}	16.290 s
	$M = 7$	148	158		2.080×10^{-9}	8.136×10^{-6}	16.333 s

and M is 5, respectively. In this case the CPU time required by adjoint truncated-Newton algorithm is about half of that required by either the truncated-Newton or the LBFGS algorithms.

If we relax the convergence criterion given by Eq. (31) by two orders of magnitude, then the adjoint truncated-Newton, truncated-Newton and LBFGS algorithms require 8, 16 and 55 iterations and take 5 s, 8 s and 6 s of CPU time to converge where we used a maximum number of conjugate gradient inner-iterations of 50 for the adjoint truncated-Newton and truncated-Newton and $M = 5$ for the LBFGS, respectively. Therefore even for the relaxed accuracy requirement the adjoint truncated-Newton algorithm performed slightly better than the LBFGS algorithm in terms of CPU time required to satisfy the convergence criteria.

Let us define the degree of nonlinearity of the cost function at $\vec{U}$ as

$$DN(\vec{U}) = (J(\vec{U}) - (J(\vec{U}^*) + p^t \nabla J(\vec{U}^*) + 0.5 p^t \nabla^2 J(\vec{U}^*) p))/\|p\|^3, \tag{33}$$

where $\vec{U}$ is a point between the starting point U_0 and the solution $\vec{U}^*$, $p = \vec{U} - \vec{U}^*$ and p^t denotes the transpose of p. $DN(\vec{U})$ gives a measure of the size of the third derivative or a deviation from quadratic behavior (see [32]).

For our test problem we noticed that $DN(\vec{U})$ increases from $1.7E-7$ to $4.7E+12$ as $\vec{U}$ approaches $\vec{U}^*$ from the starting point. Therefore we may classify our test problem as a highly nonlinear problem near the solution. Then it is not surprising that the LBFGS algorithm outperforms the truncated-Newton algorithm in terms of CPU time (Table 2) if we take into account Nash's observation that for most of highly nonlinear problems, the LBFGS algorithm performs better than the truncated-Newton algorithm [32]. Our point is that even in this case, if we use a more accurate line search direction in the truncated-Newton algorithm, the truncated-Newton algorithm can outperform the LBFGS as the adjoint truncated-Newton algorithm does for the particular optimal control problem tested here.

Therefore we conclude that the adjoint truncated-Newton algorithm turns out to be the most accurate and robust amongst the large-scale unconstrained minimization algorithms tested in terms of both CPU time and number of iterations for the specific problem tested here.

Table 2 also indicates that the LBFGS method is less sensitive to parameter choice (the number of LBFGS corrections that is stored in memory). The overall performance of the LBFGS method is comparable with that of adjoint truncated Newton method in terms of CPU time and accuracy. If analytic Hessian vector products are also used in a LBFGS version with preconditioning, its performance in terms of CPU time might be improved considerably.

4.3. *An accuracy analysis of the Hessian vector product*

The Hessian vector product $G_k U'$ for a given U' required by the inner conjugate algorithm of the adjoint truncated-Newton algorithm is obtained by the second order adjoint technique where the vector U' serves as the initial condition for the TLM model. The Hessian vector product $G_k U'$ of the truncated Newton algorithm is obtained by the following finite-difference approximation $G_k U'|_{FD}$

$$G_k U'|_{FD} = \frac{\hat{g}(\vec{U}_k + hU') - \hat{g}(\vec{U}_k)}{h}, \tag{34}$$

where G_k is the Hessian matrix at the k-th outer iteration, $h = \sqrt{\epsilon \times (1 + \|\vec{U}_k\|)}$ is the differencing parameter, where ϵ is taken to be the machine precision [29], and the computed gradients $\hat{g}(\vec{U}_k + hU')$ and $\hat{g}(\vec{U}_k)$ are obtained by using Eq. (18).

In order to consider the accuracy of the finite-difference approximation with respect to the differencing parameter h, we assume $h \in [h_{\min}, h_{\max}]$ where $h_{\min}$ and $h_{\max}$ are taken to be the machine accuracy ϵ and 10^3, respectively, such that the interval $[h_{\min}, h_{\max}]$ contains all reasonable sequences of differencing parameters.

Although a first order adjoint integration yields an exact gradient of the cost function with respect to the control variables, when a computer is used to calculate the gradient there are always computational errors involved. Let a positive quantity ϵ_0 denote an error bound

on the absolute error in the computed gradients of the cost function using Eq. (18) at $\vec{U}_k$ and $\vec{U}_k + hU'$. It will be assumed throughout this paper that the value of ϵ at the given point is available; an effective technique for computing ϵ is given by Hamming [20].

So one has

$$\hat{g}(\vec{U}_k) = \vec{g}(\vec{U}_k) + \theta_0\epsilon_0, \tag{35}$$

$$\hat{g}(\vec{U}_k + hU') = \vec{g}(\vec{U}_k + hU') + \theta_1\epsilon_0, \tag{36}$$

where $g(\vec{U}_k)$ denotes the true value of the gradient of the cost function with respect to the initial conditions, $|\theta_0| \leq 1$ and $|\theta_1| \leq 1$. Using a Taylor series expansion, one obtains

$$\begin{aligned}\vec{g}(\vec{U}_k + hU') &= \vec{g}(\vec{U}_k) + h\frac{\partial \vec{g}(\vec{U}_k)}{\partial \vec{U}_k}U' + \frac{h^2}{2}(U')^*\frac{\partial^2 \vec{g}(\vec{U}_k)}{\partial \vec{U}_k^2}U' + \cdots \\ &= \vec{g}(\vec{U}_k) + hG_kU'|_t + \frac{h^2}{2}(U')^*\frac{\partial^2 \vec{g}(\vec{\xi})}{\partial \vec{U}_k^2}U',\end{aligned} \tag{37}$$

where $G_kU'|_t = [\partial \vec{g}(\vec{U}_k)/\partial \vec{U}_k]U'$ denotes the true value of the Hessian vector product and $\vec{\xi}$ is a point in the interval $[\vec{U}_k, \vec{U}_k + hU']$.

Solving for $G_kU'|_t$ from Eq. (37) and using Eqs. (35) and (36), one obtains

$$\begin{aligned}G_kU'|_t &= \frac{\vec{g}(\vec{U}_k + hU') - \vec{g}(\vec{U}_k)}{h} - \frac{h}{2}(U')^*\frac{\partial^2 \vec{g}(\vec{\xi})}{\partial \vec{U}_k^2}U' \\ &= \frac{\hat{g}(\vec{U}_k + hU') - \hat{g}(\vec{U}_k)}{h} + \frac{(\theta_1 - \theta_0)\epsilon_0}{h} - \frac{h}{2}(U')^*\frac{\partial^2 \vec{g}(\vec{\xi})}{\partial \vec{U}_k^2}U',\end{aligned} \tag{38}$$

where $-h(U')^*[\partial^2 \vec{g}(\vec{\xi})/\partial \vec{U}_k^2]U'/2$ is the truncation error resulting from the definition of the finite-differencing operator and $(\theta_1 - \theta_0)\epsilon_0/h$ is the condition error [14] resulting from computational error due to using computers. Combining Eqs. (34) and (38), one obtains

$$G_kU'|_t = G_kU'|_{FD} + \frac{(\theta_1 - \theta_0)\epsilon_0}{h} - \frac{h}{2}(U')^*\frac{\partial^2 \vec{g}(\vec{\xi})}{\partial \vec{U}_k^2}U'. \tag{39}$$

According to Eq. (29)

$$G_kU'|_t = G_kU'|_{SOA}, \tag{40}$$

where $G_kU'|_{SOA}$ denotes the Hessian vector product obtained by using the second order adjoint technique. If a computer is used to calculate $G_kU'|_{SOA}$, one has

$$G_kU'|_t = \hat{G}_kU'|_{SOA} + \theta_2\epsilon_0, \tag{41}$$

where $\hat{G}_kU'|_{SOA}$ is the computed Hessian vector product and $|\theta_2| \leq 1$. The error bound on the absolute error in the computed Hessian vector product is assumed to be the same as on that in the computed gradient. In order to investigate the truncation and condition errors one may look at the difference between the $\hat{G}_kU'|_{SOA}$ and $G_kU'|_{FD}$,

$$\hat{G}_kU'|_{SOA} - G_kU'|_{FD} = \frac{(\theta_1 - \theta_0)\epsilon_0}{h} - \frac{h}{2}(U')^*\frac{\partial^2 \vec{g}(\vec{\xi})}{\partial \vec{U}_k^2}U' - \theta_2\epsilon_0, \tag{42}$$

Table 3. The rms errors between the Hessian vector products obtained by using the second order adjoint and the finite-difference techniques for various differencing parameters at the end of 15 iterations of the adjoint truncated-Newton algorithm, respectively. The maximum number of conjugate gradient inner-iterations is 4 and h is the differencing parameter used in the original truncated-Newton algorithm.

Differencing parameters	rms errors
$h \times 10^{9}$	$7.3674493981314 \times 10^{-7}$
$h \times 10^{8}$	$6.9428037222996 \times 10^{-8}$
$h \times 10^{7}$	$6.8992950502148 \times 10^{-9}$
$h \times 10^{6}$	$6.8262063502701 \times 10^{-10}$
$h \times 10^{5}$	$6.2331779844824 \times 10^{-11}$
$h \times 10^{4}$	$1.1954425938165 \times 10^{-11}$
$h \times 10^{3}$	$1.4501562065627 \times 10^{-11}$
$h \times 10^{2}$	$2.9876262946020 \times 10^{-11}$
$h \times 10^{1}$	$2.6527501712652 \times 10^{-10}$
$h \times 10^{0}$	$2.6074180374738 \times 10^{-9}$
$h \times 10^{-1}$	$2.5350296945351 \times 10^{-8}$

where the condition error and $\theta_2 \epsilon_0$ are computational errors resulting from the fact that a computer is used to obtain the gradient and the Hessian vector product while the truncation error resulting from the finite-difference approximation changes with differencing parameter.

The rms errors between the Hessian vector products obtained by using the second order adjoint and the finite-difference techniques with various differencing parameters are displayed in Table 3. This Table indicates that (a) the Hessian vector products obtained by the finite-difference and the second order adjoint techniques follow a relationship given by Eq. (42) where the error term $O(h)$ dominates for large differencing parameter h, (b) for a small differencing parameter, the error term $O(1/h)$ in Eq. (42) dominates. The finite-difference technique involves subtractions of nearly equal numbers which result in cancellations of significant digits and which are the reason for the increase in the rms errors and (c) the differencing parameter h in the truncated Newton algorithm of Nash [29] may become too small near the end of the minimization process. Thus in practice, the Hessian vector product obtained by the second order adjoint approach is more accurate than that obtained by the finite-difference approach.

Figure 2 shows the first 50 components of the scaled difference between the Hessian vector products obtained by using the second order adjoint and the finite-difference techniques respectively after 15 iterations using the adjoint truncated-Newton algorithm. The differencing parameter is chosen as $h_1 = h$ (solid line), $h_2 = h \times 10^{-1}$ (dotted line) and $h_3 = h \times 10^{3}$ (dashed line) where h is the differencing parameter used in the original truncated Newton minimization algorithm, respectively. The results with $h_3 = h \times 10^{3}$ clearly illustrate that toward the end of the minimization the differencing parameter in the truncated Newton algorithm of Nash [29] is too small. This result agrees with the rms error evolution in Table 3. It can be seen that the finite-difference technique can yield an approximation of similar accuracy to that obtained by second order adjoint technique if the differencing parameter is properly chosen. However simply increasing or decreasing the differencing parameter h at every iteration will not improve the performance of the truncated Newton minimization algorithm since the differencing parameter depends on the vector U' at each iteration and the vector U' is not known prior to performing the minimization iteration.

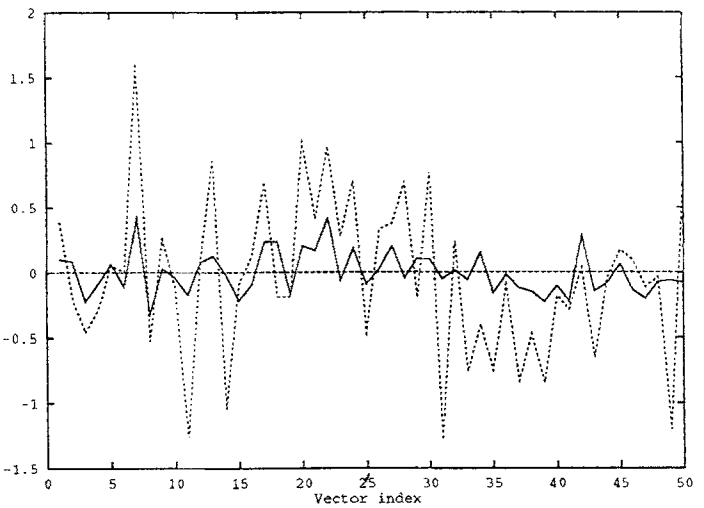

Figure 2. The first 50 components of the difference between the Hessian vector products obtained by using the second order adjoint and the finite-difference techniques scaled by a factor $1.E09$ after 15 iterations using the adjoint truncated-Newton algorithm where the differencing parameter is chosen as $h_1 = h$ (solid line), $h_2 = h \times 1.E-01$ (dotted line) and $h_3 = h \times 1.E+03$ (dashed line, which coincides with the x-axis) instead of h in the original truncated-Newton algorithm, respectively.

The differencing parameter h should be chosen such as (a) it balances the truncation error of the order h with the condition error ([14], [16], [17]) of order $1/h$, (b) h must be adjusted to the size of vector U' [41, 42] and (c) h should not become so small as to cause cancellations of significant digits [44]. It is difficult to choose an h which satisfies all these requirements. Some good choices of h in addition to that used in the truncated Newton minimization algorithm [29] are $h = 2(1 + \|\vec{U}_k\|)\sqrt{\epsilon}/\|U'\|$ [41, 42], $h = \sqrt{\epsilon}/\|U'\|$ [7] and $h = 2(1 + \|\vec{U}_k\|)\sqrt{\epsilon}/\|U'\|^2$ [37] etc. But results (not shown here) obtained using these choices of h are no better than those obtained when $h = \sqrt{\epsilon \times (1 + \|\vec{U}_k\|)}$ as used in the truncated Newton algorithm [29]. All these choices of h cause cancellations of significant digits at some stage of the minimization process.

The convergence rate of the adjoint truncated-Newton minimization algorithm is best understood intuitively by splitting the error into three terms [37]. Let $\vec{U}^*$ be the true solution to the problem, $G(\vec{U}_k)$ the exact Hessian at $\vec{U}_k$, G_k the approximate Hessian obtained by the finite-difference or by the second order adjoint technique, and H_k the approximate matrix to the inverse Hessian matrix that the inner conjugate gradient algorithm actually used, i.e. $\vec{p}_k = -H_k \vec{g}_k$. Then, if the step size chosen was 1 and if $G(\vec{U}_k)$ is positive definite,

$$\begin{aligned} \vec{U}_{k+1} - \vec{U}^* &= \vec{U}_k - H_k \vec{g}_k - \vec{U}^* \\ &= \left(\vec{U}_k - G(\vec{U}_k)^{-1}\vec{g}_k - \vec{U}^*\right) + \left(G(\vec{U}_k)^{-1} - G_k^{-1}\right)\vec{g}_k + \left(G_k^{-1} - H_k\right)\vec{g}_k \end{aligned} \quad (43)$$

Therefore

$$\|\vec{U}_{k+1} - \vec{U}^*\| \le \left\|\left(\vec{U}_k - G(\vec{U}_k)^{-1}\vec{g}_k - \vec{U}^*\right)\right\| + \left\|\left(G(\vec{U}_k)^{-1} - G_k^{-1}\right)\vec{g}_k\right\| + \left\|\left(G_k^{-1} - H_k\right)\vec{g}_k\right\| \quad (44)$$

The first error term is the Newton error at the $(k+1)$-th step. The second error term is due to the error in the second order adjoint technique or due to discrete differencing

and depends on the choice of h in the conjugate gradient algorithm. The third error term is an error due to round-off and early termination (truncation) in the conjugate gradient inner iteration. The second error induced by the second order adjoint technique is smaller than that caused by the finite-difference technique. Therefore the adjoint truncated-Newton minimization algorithm yields a speed-up for our test problem. It is also clear that both the adjoint truncated-Newton and truncated-Newton algorithms have the same convergence rate. We know that if cost function J is sufficiently smooth, G_k is a strongly consistent approximation to $G(\vec{U}_k)$, and $G(\vec{U}^*)$ is nonsingular, then local quadratic convergence can be obtained if the differencing parameter h decreases sufficiently rapidly ([3], [15]). Under the same conditions we expect the adjoint truncated-Newton algorithm obtains the same convergence. However in practice both the adjoint truncated-Newton and truncated-Newton algorithms have only a super linear convergence rate.

If we choose the vector U' as one of the coordinate directions $e_1, \ldots, e_n$ where e_i is the i-th unit vector and n is the number of the components in U', respectively, either the second order adjoint or the finite-difference technique will generate an approximation to the Hessian. Since, for smooth functions, the Hessian is symmetric, the approximate Hessian is often symmetrized by averaging corresponding elements in the upper and lower triangular parts of Hessian matrix. Again the second order adjoint technique will obtain an exact Hessian if we do not consider computational error while the finite-difference technique obtains only an approximation to the Hessian. Burger [2] pointed out that the second order adjoint technique requires less computing time than direct differentiation in obtaining the approximate Hessian and thus the results from the former method are more accurate than these from the latter method. Our results are consistent with those of Burger [2] in as far as the aspect of accuracy is concerned. Similar results to ours were obtained by Santosa and Symes [39, 40, 45] showing the advantage of using the second order adjoint technique to obtain the Hessian vector product in the truncated-Newton algorithm.

In summary, we conclude that the second order adjoint technique yields a more accurate value of the Hessian vector product compared to the finite-difference technique for optimal control problems tested here. When the differencing parameter h is too small or too big, cancellations of significant digits or truncation errors dominate the finite-difference approximation. It is hard to avoid the occurrence of these two types of error in the minimization process when the finite-difference technique is used [47]. Use of more accurate Hessian vector products in the inner conjugate gradient iteration of the truncated-Newton algorithm results in a better line search direction as measured by the amount of decrease in the cost function, but not in the residuals [33].

5. Summary and conclusions

In this paper, we proposed a modified version of the Nash truncated-Newton algorithm [33] by using the first order adjoint and the second order adjoint techniques, i.e. we proposed a new method to obtain a Hessian vector product to be used in the process of solving the Newton equations for the truncated-Newton minimization algorithm for an optimal control problem. The costs of the second order adjoint approach and the finite-difference approach are computationally comparable, each of them requiring four different model integrations when applied to the optimal control problem tested here. But the former approach yields an exact Hessian vector product, while the latter provides only an approximation to the

Hessian vector product if the finite-differencing parameter is taken to be the square root of the machine accuracy. The numerical results indicate that the new Hessian vector calculation strategy employed in the modified-Lanczos algorithm of Nash [33] allows the adjoint truncated-Newton algorithm to perform better than either the truncated-Newton or the LBFGS algorithms both in terms of CPU time as well as in term of number of iterations required to satisfy the prescribed convergence criterion in our test problems. This result may be very useful, since truncated-Newton type methods and conjugate-gradient methods have comparable storage requirements and constitute the only practical methods for solving many large-scale unconstrained minimization problems arising in 4-dimensional variational data assimilation.

In our test example, the eigenvalues of the Hessian at each iteration are positive [48], which implies the Hessians are positive definite and the cost function is strictly convex. Therefore the existence of a local minimum is assured. However the condition numbers at each iteration are large [48], which explains why all three algorithms require more than 16 iterations to satisfy the prescribed convergence criteria.

Theoretically, the truncated-Newton and the adjoint truncated-Newton minimization algorithms have the same convergence rate. However the adjoint truncated-Newton algorithm results in a speed-up due to the use of a more accurate Hessian vector product to obtain the line search direction.

The adjoint truncated-Newton algorithm, like its truncated-Newton counterpart, is a close approximation to the Newton method at reasonable storage and computational cost. We expect the adjoint truncated-Newton algorithm to yield a similar speed-up for other large-scale unconstrained minimization problems related to optimal control and variational data assimilation. The idea of obtaining the Hessian vector product using the second order adjoint technique can be applied in other settings requiring large-scale minimization, in cases where an adjoint model formulation is possible, or via automatic differentiation techniques [19]—pointing to a more general applicability of this idea. An application of the adjoint truncated-Newton minimization algorithm for minimizing a cost functional in optimal control of distributed parameters in a primitive equations 3-D spectral model will be reported separately, once the second order adjoint model of this model is derived.

Both the truncated Newton and the LBFGS methods are general purpose methods. The adjoint truncated-Newton algorithm is only useful for optimal control problems where the model equations serve as strong constraints. The application of the adjoint truncated-Newton algorithm requires one to develop the adjoint models corresponding to the forward model. One could extend the application of the second order adjoint to the LBFGS algorithm with the Hessian vector product being used as the right hand side in the LBFGS secant Eq. [10].

Acknowledgments

The authors would like to thank Dr. John Dennis for his insightful remarks, Dr. W.W. Symes for providing us his papers. The authors gratefully acknowledge the support of Dr. Pamela Stephens, Director of GARP Division of the Atmospheric Sciences at NSF. This research was funded by NSF Grant ATM-9102851. Additional support was provided by the Supercomputer Computations Research Institute at Florida State University, which is partially funded by the Department of Energy through contract No. DE-FC0583ER250000. The

insightful comments of two anonymous reviewers have greatly contributed toward sharpening the issues presented in the paper and are gratefully acknowledged.

A. A nonlinear model with exact solutions

This appendix will illustrate the adjoint process by using a quadratic model. It is emphasized that when the model equations are nonlinear, the corresponding TLM is a second order accurate approximation to the true evolution of the perturbations due to the perturbation on the initial condition of the model equations, the first order adjoint model yields an exact gradient of the cost function with respect to the control variables, and the second order adjoint model yields an exact Hessian vector product.

Let us now consider the following 1-dimensional nonlinear model equation

$$\frac{\partial X}{\partial t} = -X^2, \tag{45}$$

$$X(0) = U, \tag{46}$$

where time t changes from 0 to 1. The exact solution of the model is

$$X(t) = \frac{U}{(tU+1)}. \tag{47}$$

If the observations are model generated with the initial condition

$$X(0) = 1, \tag{48}$$

then from Eq. (47), the observations may be written as

$$X(t) = \frac{1}{(t+1)}. \tag{49}$$

Let us define the cost function as

$$J(U) = \frac{1}{2}\int_0^1 \langle (X - X^0), (X - X^0) \rangle dt, \tag{50}$$

i.e.

$$\begin{aligned} J(U) &= \frac{1}{2}\int_0^1 \left(\frac{U}{(tU+1)} - \frac{1}{(t+1)}\right)^2 dt \\ &= \frac{1}{2}\left[1 + U + \frac{2U}{(1-U)} \ln \frac{(U+1)}{2} - \frac{(1+3U)}{2(U+1)}\right]. \end{aligned} \tag{51}$$

The first order adjoint model of Eqs. (45) and (46) may be written as

$$-\frac{\partial P}{\partial t} = (-2X)^* P + (X - X^0), \tag{52}$$

i.e.

$$-\frac{\partial P}{\partial t} = -\frac{2U}{Ut+1} P + \frac{U}{(tU+1)} - \frac{1}{(t+1)}, \tag{53}$$

$$P(1) = 0, \tag{54}$$

where P is the adjoint variable. The gradient of the cost function with respect to the initial conditions is given by

$$\nabla_U J = P(0). \tag{55}$$

Equations (53–54) have an analytic solution of the following form

$$\begin{aligned} P(t) &= (tU+1)^2\left[\int\left(\frac{1}{(t+1)} - \frac{U}{(tU+1)}\right)\frac{1}{(tU+1)^2}dt + c\right] \\ &= (tU+1)^2\left[\int \frac{dt}{(t+1)(tU+1)^2} + \frac{1}{2(tU+1)^2} + c\right] \\ &= (tU+1)^2\left[\frac{1}{(1-U)(tU+1)} + \frac{1}{(1-U)^2}\ln\frac{t+1}{Ut+1}\right. \\ &\qquad \left. + \frac{1}{2(tU+1)^2} + c\right], \end{aligned} \tag{56}$$

where c is a constant to be determined by the final condition (54). Therefore

$$c = -\frac{1}{1-U^2} - \frac{1}{(1-U)^2}\ln\left(\frac{2}{U+1}\right) - \frac{1}{2(U+1)^2}, \tag{57}$$

and Eq. (59) yields

$$P(0) = \frac{1}{1-U} + \frac{1}{2} - \frac{1}{1-U^2} - \frac{1}{(1-U)^2}\ln\left(\frac{2}{U+1}\right) - \frac{1}{2(U+1)^2}. \tag{58}$$

Equation (58) is exactly the gradient of the cost function (51) with respect to the initial condition U.

Let us now consider a perturbation, U', on the initial condition for X, U. The resulting perturbations for the variables X and P, $\hat{X}$ and $\hat{P}$, are obtained from Eqs. (45), (46) and (53–54) as

$$\frac{\partial \hat{X}}{\partial t} = -2X\hat{X}, \tag{59}$$

$$\hat{X}(0) = U', \tag{60}$$

$$-\frac{\partial \hat{P}}{\partial t} = (-2X)^*\hat{P} + (-2\hat{X})^*P + \hat{X}, \tag{61}$$

$$\hat{P}(1) = 0, \tag{62}$$

Equations (59–60) and (61)–(62) are the TLM and second order adjoint model of the model Eqs. (45) and (46), respectively. Substituting $X = U/(tU+1)$ into Eq. (59) one obtains the following exact solution of the TLM (59)

$$\hat{X} = \frac{U'}{(tU+1)^2}, \tag{63}$$

which is a quadratic approximation to

$$\frac{U+U'}{t(U+U')+1}-\frac{U}{t(U)+1}, \tag{64}$$

the exact evolution of the perturbation due to the perturbation U' on the initial condition U. Table 1 summarizes the errors between the true evolution given by Eq. (64) and the approximation given by Eq. (63) at $U = 0.9$ and $t = 0.5$ due to different perturbations on the initial condition U. Table 1 clearly indicates that

$$\frac{U+U'}{t(U+U')+1}-\frac{U}{t(U)+1}=\frac{U'}{(tU+1)^2}+O(\|U'\|^2). \tag{65}$$

Substituting $X = U/(tU+1)$, the first order adjoint solution (56) and the TLM solution (63) into second order adjoint Eq. (61), one obtains

$$-\frac{\partial \hat{P}}{\partial t}=\frac{-2U}{(tU+1)}\hat{P}-\frac{2U'}{(1-U)(tU+1)}-\frac{2U'}{(1-U)^2}\ln\frac{t+1}{tU+1}-2U'c, \tag{66}$$

where c is given by (57). Equations (66) and (62) have an analytical solution of the following form

$$\begin{aligned}\hat{P}(t) &= (tU+1)^2\left[\int\left[\frac{2U'}{(1-U)(tU+1)}+\frac{2U'}{(1-U)^2}\ln\frac{t+1}{tU+1}+2U'c\right]\right.\\ &\qquad\left.\times\frac{1}{(tU+1)^2}dt+c_1\right]\\ &= (tU+1)^2\left\{-\frac{U'}{U(1-U)(tU+1)^2}+\frac{2U'}{(1-U)^2U}\right.\\ &\qquad\left.\times\left[\frac{2U}{U^2-1}\ln\frac{tU+1}{t+1}+\frac{1}{Ut+1}\right]-\frac{2U'c}{U(tU+1)}+c_1\right\},\end{aligned} \tag{67}$$

where

$$c_1=U'\frac{\frac{1}{(1-U^2)}+2c}{U(1+U)}-2U'\left\{\frac{2U}{(U^2-1)}\ln\frac{U+1}{2}+\frac{1}{U+1}\right\}\Big/[(1-U)^2U], \tag{68}$$

is a constant determined by the final condition (62). Therefore Eq. (67) yields

$$\begin{aligned}\hat{P}(0) &= U'\left[-\frac{1}{U(1-U)}+\frac{2}{(1-U)^2U}-\frac{2c}{U}\right]+c_1\\ &= U'\left[\frac{U^2+U+2}{(1-U^2)^2}+\frac{1}{(1+U)^3}\right],\end{aligned} \tag{69}$$

which is exactly the Hessian vector product $[\partial^2 J/\partial U^2]U'$ directly obtained from Eq. (51)

$$\begin{aligned}\frac{\partial^2 J}{\partial U^2}U' = &\left\{\frac{1}{(1-U)^2}-\frac{2U}{(1-U^2)^2}-\frac{2}{(1-U)^3}\ln\frac{2}{(U+1)}\right.\\ &\left.+\frac{1}{(1-U)^2(1+U)}+\frac{1}{(1+U)^3}\right\}U',\end{aligned} \tag{70}$$

i.e.

$$\hat{P}(0) = \frac{\partial^2 J}{\partial U^2} U'. \tag{71}$$

B. A numerical accuracy analysis of the Hessian vector product

This Appendix provides a numerical accuracy analysis of the Hessian vector product from another point of view.

Let C_1, C_2 and C_3 denote the averages of the components of the vectors $(\theta_3 - \theta_0)\epsilon_0$, $-(U')^*[\partial^2 \vec{g}_t(\vec{\xi})/\partial \vec{U}_k^2]U'$ and $(\theta_0 - \theta_1)\epsilon_0 + 0.5(U')^*[\partial^3 J(\vec{\xi}_0)/\partial U^3]U'$, respectively, and assume that they are constant in the vicinity of some $h \in [h_{\min}, h_{\max}]$. The rms error d between the the Hessian vector products obtained by using the second order adjoint and the finite-difference techniques approximately satisfies the following equation

$$\frac{C_1}{h} + C_2 h + C_3 = d, \tag{72}$$

If we choose h and d as the three differencing parameters and the three rms errors in Table 3 from row 2 to row 4, from row 5 to row 7, and from row 8 to row 10, respectively, then we obtain three systems of linear equations in C_1, C_2 and C_3. The solutions of the systems of linear equations are

$$C_1 = 3.3 \times 10^{-10}, \quad C_2 = 8.5 \times 10^{-11}, \quad C_3 = -5.3 \times 10^{-11}, \tag{73}$$

$$C_1 = 7.0 \times 10^{-14}, \quad C_2 = 6.9 \times 10^{-11}, \quad C_3 = 5.4 \times 10^{-12}, \tag{74}$$

and

$$C_1 = 2.1 \times 10^{-14}, \quad C_2 = -1.6 \times 10^{-9}, \quad C_3 = 5.1 \times 10^{-12}, \tag{75}$$

respectively. In the three cases, if h is chosen as the differencing parameter from the second row, from the sixth row and from the tenth row of Table 3, respectively, then the three terms in Eq. (72) are

$$\frac{C_1}{h} = 4.0 \times 10^{-13}, \quad C_2 h = 6.9 \times 10^{-8}, \quad C_3 = -5.3 \times 10^{-11}, \tag{76}$$

$$\frac{C_1}{h} = 8.5 \times 10^{-13}, \quad C_2 h = 5.7 \times 10^{-12}, \quad C_3 = 5.4 \times 10^{-12}, \tag{77}$$

and

$$\frac{C_1}{h} = 2.6 \times 10^{-9}, \quad C_2 h = -1.3 \times 10^{-14}, \quad C_3 = 5.1 \times 10^{-12}, \tag{78}$$

respectively. Clearly the truncation error $C_2 h$ and the condition error C_1/h dominate the first and third cases, respectively. In the second case, the magnitudes of the errors from the finite-difference approach and the second order adjoint approach are similar. These results are consistent with the theoretical analysis presented subsection 4.3.

References

1. H.T. Banks and K. Kunisch, "Estimation techniques for distributed parameter systems," Birkhauser: Boston (Systems & Control: Formulations & Applications), Vol. 11, p. 315, 1989.
2. J. Burger, J.L. Brizaut, and M. Pogu, "Comparison of two methods for the calculation of the gradient and of the Hessian of the cost functions associated with differential systems," Mathematics and Computers in Simulation, Vol. 34, pp. 551–562, 1992.
3. J.C.P. Bus, "Convergence of Newton-like methods for solving systems of nonlinear equations," Numerische Mathematik, Vol. 27, pp. 271–281, 1977.
4. P. Courtier and O. Talagrand, "Variational assimilation of meteorological observations with the adjoint equations Part 2. Numerical results," Q.J.R. Meteorol. Soc., Vol. 113, pp. 1329–1347, 1987.
5. W.C. Davidon, "Variable metric method for minimization," A.E.C. Research and Development Report, ANL-5990 (Rev.), 1959.
6. R.S. Dembo, S.C. Eisenstat, and T. Steihaug, "Inexact Newton methods," SIAM Journal of Numerical Analysis, Vol. 19, pp. 400–408, 1982.
7. R.S. Dembo and T. Steihaug, "Truncated-Newton algorithms for large-scale unconstrained optimization," Math. Prog., Vol. 26, pp. 190, 212, 1983.
8. J. Dennis and Robert B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Englewood Cliffs, N.J.: Prentice-Hall, p. 378, 1983.
9. J. Dennis and B. Schnabel, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Englewood Cliffs, N.J.: Prentice-Hall, 1983.
10. J. Dennis, Personal communication, Dept. of Math., Rice University, Houston, TX, 1994.
11. L.C.W. Dixon, "Use of automatic differentiation for calculating Hessians and Newton steps," in Automatic Differentiation of Algorithms: Theory, Implementation, and Application, Andreas Griewank and George F. Corliss (eds.), *SIAM*, Philadelphia, pp. 115–125, 1991.
12. Ronald M. Errico, Tomislava Vukićević, and Kevin Raeder, "Examination of the accuracy of a tangent linear model," *Tellus*, Vol. 45A, pp. 462–477, 1993.
13. J.C. Gilbert, "Automatic differentiation and iterative processes," Optimization Methods and Software, Vol. 1, pp. 13–21, 1992.
14. P.E. Gill and W. Murray, "Quasi-Newton methods for unconstrained optimization," J. Inst. Maths Applics, Vol. 9, pp. 91–108, 1972.
15. P.E. Gill and W. Murray, Numerical Methods for Unconstrained Optimization, Academic Press, London and New-York, p. 283, 1974.
16. P.E. Gill and W. Murray, "Newton-type methods for unconstrained and linearly constrained optimization," Math. Prog., Vol. 28, pp. 311–350, 1979.
17. P.E. Gill, W. Murray, M.C. Saunders, and M.H. Wright, "Computing forward-difference intervals for numerical optimization," SIAM J. Sci. Stat. Comput., Vol. 4, pp. 310–321, 1983.
18. A. Grammeltvedt, "A survey of finite-difference schemes for the primitive equations for a barotropic fluid," Mon. Wea. Rev., Vol. 97, pp. 387–404, 1969.
19. A. Griewank and George F. Corliss, Automatic Differentiation of Algorithms: Theory, Implementation, and Application, SIAM, Philadelphia, p. 353, 1991.
20. R.W. Hamming, Numerical Methods for Scientists and Engineers, 2nd edition, New York: McGraw-Hill, 1973.
21. J.F. Lacarra and O. Talagrand, "Short-range evolution of small perturbations in a barotropic model," *Tellus*, Vol. 40A, pp. 81–95, 1988.
22. F.X. Le Dimet and O. Talagrand, "Variational algorithms for analysis and assimilation of meteorological observations: Theoretical aspects," Tellus, Vol. 38A, pp. 97–110, 1986.
23. D.C. Liu, and Jorge Nocedal, "On the limited memory BFGS method for large scale minimization," Math. Prog., Vol. 45, pp. 503–528, 1989.
24. S.G. Nash, "Truncated-Newton methods," Ph.D. Thesis, Computer Science Department, Stanford University, CA, 1982.
25. S.G. Nash, "Newton-type minimization via the Lanczos method," SIAM J. Numer. Anal., Vol. 21, No. 4, pp. 770–788, 1984.
26. S.G. Nash, "Truncated-Newton methods for large-scale function minimization," Applications of Nonlinear Programming to Optimization and Control, H.E. Rauch (ed.), Pergamon Press, Oxford, pp. 91–100, 1984.
27. S.G. Nash, "User's guide for TN/TNBC: Fortran routines for nonlinear optimization," Tech. Rep. No., Vol. 397, p. 17, 1984.

28. S.G. Nash, "Solving nonlinear programming problems using truncated-Newton techniques," Numerical optimization, P.T. Boggs, R.H. Byrd, and R.B. Schnabel (eds.), Philadelphia: SIAM, pp. 119–136, 1984.
29. S.G. Nash, "Preconditioning of truncated-Newton methods," SIAM J. Sci. Stat. Comput., Vol. 6, No. 3, pp. 599–616, 1985.
30. S.G. Nash and A. Sofer, "Block truncated-Newton methods for parallel optimization," Math. Prog., Vol. 45, pp. 529–546, 1989.
31. S.G. Nash and A. Sofer, "A parallel line search for Newton type methods in computer science and statistics," in Proceeding 21-st Symposium on the Interface, K. Berk and L. Malone (eds.), ASA, pp. 134–137, 1989.
32. S.G. Nash and Jorge Nocedal, "A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization," Tech. Rep. NAM, Vol. 2, Dept. of Electrical Engineering and Computer Science, Northwestern University, p. 19, Dec. 1989.
33. S.G. Nash and A. Sofer, "Assessing a search direction within a truncated-Newton method," Operations Research Letters, Vol. 9, No. 4, pp. 219–221, 1990.
34. I.M. Navon and D.M. Legler, "Conjugate gradient methods for large scale minimization in meteorology," Mon. Wea. Rev., Vol. 115, pp. 1479–1502, 1987.
35. I.M. Navon, X.L. Zou, J. Derber, and J. Sela, "Variational data assimilation with an adiabatic version of the NMC Spectral Model," Mon. Wea. Rev., Vol. 122, pp. 1433–1446, 1992.
36. J. Nocedal, "Updating quasi-Newton matrices with limited storage," Mathematics of Computation, Vol. 35, pp. 773–782, 1980.
37. D.P. O'Leary, "A discrete Newton algorithm for minimizing a function of many variables," Math. Prog., Vol. 23, pp. 20–23, 1983.
38. S. Omatu and John H. Seinfeld, "Distributed Parameter Systems, Theory and Applications," Oxford Science Publications, Oxford Mathematical Monographs, Claredon Press, Oxford, p. 430, 1989.
39. Santosa, Fadil, and William W. Symes, "Computation of the Hessian for least-squares solutions of inverse problems of reflection seismology," Inverse Problems, Vol. 4, pp. 211–233, 1988.
40. Santosa, Fadil, and William W. Symes, "An analysis of least squares velocity inversion," Society of Exploration Geophysicists, Geophysical monograph #4, Tulsa, 1989.
41. T. Schlick and A. Fogelson, "TNPACK—A truncated Newton minimization Package for large-scale problems: I. Algorithm and usage," ACMTOMS, Vol. 18, No. 1, pp. 46–70, 1992.
42. T. Schlick and A. Fogelson, "TNPACK—A Truncated Newton minimization package for large-scale problems: II. Implementation examples," ACMTOMS, Vol. 18, No. 1, pp. 71–111, 1992.
43. D.F. Shanno and K.H. Phua, "Remark on algorithm 500—a variable method subroutine for unconstrained nonlinear minimization," ACM Trans. on Mathematical Software, Vol. 6, pp. 618–622, 1980.
44. G.W. Stewart III, "A modification of Davidon's minimization method to accept difference approximations of derivatives," Journal of the Association for Computing Machinery, Vol. 14, No. 1, pp. 72–83, 1967.
45. William W. Symes, "A differential semblance algorithm for the inverse problem of reflection seismology," Computers Math. Applic., Vol. 22, No. 4/5, pp. 147–178, 1991.
46. O. Talagrand and P. Courtier, "Variational assimilation of meteorological observations with the adjoint vorticity equation—Part 1, Theory," Q.J.R. Meteorol. Soc., Vol. 113, pp. 1311–1328, 1987.
47. Thomas, R. Cuthbert, Jr., Optimization using personal computers, John Wiley & Sons, New York, p. 474, 1987.
48. Z. Wang, I.M. Navon, F.X. Le Dimet, and X. Zou, "The Second Order Adjoint Analysis: Theory and Application," Meteorology and Atmospheric Physics, Vol. 50, pp. 3–20, 1992.
49. Z. Wang, I.M. Navon, and X. Zou, "The adjoint truncated Newton algorithm for large-scale unconstrained optimization," Tech. Rep. FSU-SCRI-92-170, Florida State University, Tallahassee, Florida, p. 44, 1993.
50. Zhi, Wang, "Variational data assimilation with 2-D shallow water equations and 3-D FSU global spectral models," Tech. Rep. FSU-SCRI-93T-149, Florida State University, Tallahassee, Florida, p. 235, 1993.
51. William W.-G. Yeh, "Review of parameter identification procedures in groundwater hydrology: The inverse problem," Water Resources Research, Vol. 22, No. 2, pp. 95–108, 1986.
52. X. Zou, I.M. Navon, F.X. Le Dimet, A. Nouailler, and T. Schlick, "A comparison of efficient large-scale minimization algorithms for optimal control applications in meteorology," Tech. Rep. FSU-SCRI-90-167, Florida State University, Tallahassee, Florida, p. 44, 1990.
53. X. Zou, I.M. Navon, M. Berger, Paul K.H. Phua, T. Schlick, and F.X. Le Dimet, "Numerical experience with limited-Memory Quasi-Newton methods and Truncated Newton methods," SIAM Jour. on Numerical Optimization, Vol. 3, pp. 582–608, 1993.
54. X. Zou, I.M. Navon, and F.X. Le Dimet, "Incomplete observations and control of gravity waves in variational data assimilation," Tellus, Vol. 44A, pp. 273–296, 1992.