International Journal of Computational Fluid Dynamics, August 2003 Vol. 17 (4), pp. 299–306

Taylor & Francis
Taylor & Francis Group

An Analysis of a Hybrid Optimization Method for Variational Data Assimilation

DACIAN N. DAESCU[a] and I.M. NAVON[b,*]

[a]*Institute for Mathematics and its Applications, University of Minnesota, Minneapolis, MN 55455, USA;* [b]*Department of Mathematics and School of Computational Science and Information Technology, Florida State University, Tallahassee, FL 32306-4120, USA*

(Received April 2003)

Dedicated to Professor Mutsuto Kawahara on the occasion of his 60th birthday

In four-dimensional variational data assimilation (4D-Var) an optimal estimate of the initial state of a dynamical system is obtained by solving a large-scale unconstrained minimization problem. The gradient of the cost functional may be efficiently computed using the adjoint modeling, at the expense equivalent to a few forward model integrations; for most practical applications, the evaluation of the Hessian matrix is not feasible due to the large dimension of the discrete state vector. Hybrid methods aim to provide an improved optimization algorithm by dynamically interlacing inexpensive L-BFGS iterations with fast convergent Hessian-free Newton (HFN) iterations. In this paper, a comparative analysis of the performance of a hybrid method vs. L-BFGS and HFN optimization methods is presented in the 4D-Var context. Numerical results presented for a two-dimensional shallow-water model show that the performance of the hybrid method is sensitive to the selection of the method parameters such as the length of the L-BFGS and HFN cycles and the number of inner conjugate gradient iterations during the HFN cycle. Superior performance may be obtained in the hybrid approach with a proper selection of the method parameters. The applicability of the new hybrid method in the framework of operational 4D-Var in terms of computational cost and performance is also discussed.

Keywords: Optimization; Hessian-free Newton; Limited memory L-BFGS; Data assimilation

INTRODUCTION

Four-dimensional variational data assimilation (4D-Var) aims to provide an optimal estimate of the initial state of a dynamical system through the minimization of a cost functional that measures the misfit between the modeled and observed state of the system over an analysis time interval (Kalnay, 2002). The implementation of the 4D-Var data assimilation is based on an iterative optimization procedure: each iteration requires (at least) a forward run of the model to obtain the value of the cost functional and a backward integration of the adjoint model to evaluate the gradient. Due to the high computational burden, a key element of the assimilation process is to select an efficient large-scale optimization algorithm. In many practical applications (e.g. oceanography, numerical weather prediction), the dimension of the discrete state vector may be as high as $10^6 - 10^7$ such that the explicit evaluation of the Hessian of the cost functional is not computationally feasible.

The experience gained thus far treating variational data assimilation problems, in particular large-scale unconstrained minimizations (Navon *et al.*, 1992; Zou *et al.*, 1993; Wang *et al.*, 1995; LeDimet *et al.*, 2002), is that in 2D both the truncated Newton (TN) method (Dembo *et al.*, 1982; Nash, 1984; Schlick and Fogelson, 1992) and the limited memory quasi-Newton method L-BFGS (Gilbert and LeMarechal, 1989; Liu and Nocedal, 1989) are powerful optimization algorithms which are more efficient than other techniques (Wang *et al.*, 1998).

The TN method is implemented as a Hessian-free Newton (HFN) method (see Nocedal and Wright, 1999) requiring only the products of the Hessian times a vector which may be approximated by finite differences or may be exactly evaluated using a second-order adjoint model (LeDimet *et al.*, 2002). The HFN method tends to blend the rapid (quadratic) convergence rate of the classical Newton method with feasible storage and computational requirements.

*Corresponding author. E-mail: navon@csit.fsu.edu

ISSN 1061-8562 print/ISSN 1029-0257 online © 2003 Taylor & Francis Ltd
DOI: 10.1080/1061856031000120510

The L-BFGS algorithm is simple to implement and uses a cost-effective formula that requires only first order derivative (gradient) information. Furthermore, the amount of storage required to build an approximation of the inverse Hessian matrix can be controlled by the user. It has been found that, in general, HFN performs better than L-BFGS for functions that are nearly quadratic, while for highly nonlinear functions L-BFGS outperforms TN (Nash and Nocedal, 1991). HFN requires less iterations than L-BFGS to reach the solution, while the computational cost of each iteration is high and the curvature information gathered in the process is lost after the iteration has been completed. L-BFGS, on the other hand, performs inexpensive iterations, with poorer curvature information—a process that may become slow on ill-conditioned problems.

Recently, Morales and Nocedal (2002) have developed a hybrid method that consists of interlacing in a dynamical way L-BFGS and HFN iterations. The hybrid method aims to alleviate the shortcomings of both L-BFGS and HFN and for this reason Morales and Nocedal (2002) called it an "enriched method". A powerful preconditioning method is used for the conjugate gradient algorithm in the inner iteration of the HFN (Morales and Nocedal, 2000). The limited memory matrix constructed in the L-BFGS iterations is updated during the cycle of HFN iterations thus playing a dual role in the optimization process: to precondition the inner conjugate gradient iteration in the first iteration of the HFN cycle as well as to provide the initial approximation of the inverse of the Hessian matrix for the next L-BFGS cycle. In this way information gathered by each method improves the performance of the other without increasing the computational cost.

The analysis presented by Morales and Nocedal (2002) revealed that on a large number of test problems the hybrid method outperformed the HFN method, but not the L-BFGS method. In this paper, we investigate the potential benefits that may be achieved by using the hybrid method vs. L-BFGS and HFN methods in variational data assimilation. In the second section, we outline the L-BFGS and HFN methods as well as the hybrid algorithm. In the third section, we present a two-dimensional shallow-water model and consider the variational data assimilation problem in an idealized (twin experiments) framework. Properties of the cost functional with high impact on the optimization process such as the deviation from quadratic and the condition number of the associated Hessian matrix are analyzed. Numerical results comparing L-BFGS, HFN, and the hybrid method are presented in the fourth section for various hybrid method controlling parameters. The performance of each method is analyzed in terms of the CPU time and number of function/gradient evaluations required during the minimization. A summary and a brief discussion of their adequacy in terms of computational effort for minimizing cost functionals arising in 4D variational data assimilation are presented in the fifth section.

LARGE-SCALE MINIMIZATION METHODS

In this section, we outline the algorithms used in this study to solve the large-scale unconstrained optimization problem

$$\min_{\mathbf{x}\in R^n} f(\mathbf{x}).$$

The cost functional f is assumed to be smooth, and we will further assume that the gradient $g(\mathbf{x}) = \nabla f(\mathbf{x})$ may be efficiently provided, whereas explicit evaluation of the Hessian matrix $\mathbf{G}(\mathbf{x}) = \nabla^2 f(\mathbf{x})$ is prohibitive due to the large number n of variables.

The Limited Memory BFGS Algorithm

The L-BFGS method (Liu and Nocedal, 1989) is an adaptation of the BFGS method to large problems, achieved by changing the Hessian update of the latter (Nocedal, 1980; Byrd *et al.*, 1994). Thus, in BFGS we use an approximation $\tilde{\mathbf{H}}_k$ to the inverse of the Hessian matrix $\nabla^2 f(\mathbf{x}_k)$ which is updated by

$$\tilde{\mathbf{H}}_{k+1} = \mathbf{V}_k^T \tilde{\mathbf{H}}_k \mathbf{V}_k + \rho_k \mathbf{s}_k \mathbf{s}_k^T \tag{1}$$

where $\mathbf{V}_k = \mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T$, $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, $\rho_k = 1/(\mathbf{y}_k^T \mathbf{s}_k)$, and $\mathbf{I}$ is the identity matrix. The search direction is given by

$$\mathbf{p}_{k+1} = -\tilde{\mathbf{H}}_{k+1} \mathbf{g}_{k+1}.$$

In L-BFGS, instead of forming the matrices $\tilde{\mathbf{H}}_k$ explicitly (which would require a large memory for a large problem) one only stores the vectors $\mathbf{s}_k$ and $\mathbf{y}_k$ obtained in the last m iterations which define $\tilde{\mathbf{H}}_k$ implicitly; a cyclical procedure is used to retain the latest vectors and discard the oldest ones. Thus, after the first m iterations, Eq. (1) becomes

$$\begin{aligned}\tilde{\mathbf{H}}_{k+1} = {} & (\mathbf{V}_k^T \ldots \mathbf{V}_{k-m}^T) \tilde{\mathbf{H}}_{k+1}^0 (\mathbf{V}_{k-m} \ldots \mathbf{V}_k) \\ & + \rho_{k-m} (\mathbf{V}_k^T \ldots \mathbf{V}_{k-m+1}^T) \mathbf{s}_{k-m} \mathbf{s}_{k-m}^T (\mathbf{V}_{k-m-1}^T \ldots \mathbf{V}_k) \\ & + \rho_{k-m-1} (\mathbf{V}_k^T \ldots \mathbf{V}_{k-m+2}^T) \mathbf{s}_{k-m+1} \mathbf{s}_{k-m+1}^T (\mathbf{V}_{k-m+2}^T \ldots \mathbf{V}_k) \\ & + \cdots + \rho_k \mathbf{s}_k \mathbf{s}_k^T\end{aligned}$$

with the initial guess $\tilde{\mathbf{H}}_{k+1}^0$ which is the sparse matrix

$$\tilde{\mathbf{H}}_{k+1}^0 = \frac{\mathbf{y}_k^T \mathbf{s}_k}{\mathbf{y}_k^T \mathbf{y}_k} \mathbf{I}.$$

Previous studies have shown that storage of a small number m of correction pairs is in general sufficient to provide good performance.

The Truncated Newton Algorithm

In this method, a search direction is computed by finding an approximate solution to the Newton equations

$$\mathbf{G}_k \mathbf{p}_k = -\mathbf{g}_k, \tag{2}$$

where $\mathbf{p}_k$ is a descent direction and $\mathbf{G}_k$ is the Hessian matrix of the cost function $\mathbf{G}_k = \nabla^2 f(\mathbf{x}_k)$.

The use of an approximate search direction is justified by the fact that an exact solution of the Newton equation at a point far from the minimum is computationally wasteful in the framework of a basic descent method. Thus, for each outer iteration (2) there is an inner iteration loop applying the conjugate gradient method that computes this approximate direction, $\mathbf{p}_k$, and attempts to satisfy a termination test of the form:

$$\|\mathbf{r}_k\| \leq \eta_k \|\mathbf{g}_k\|$$

where

$$\mathbf{r}_k = \mathbf{G}_k \mathbf{p}_k + \mathbf{g}_k$$

and the sequence $\{\eta_k\}$ satisfies $0 < \eta_k < 1$ for all k.

The conjugate gradient inner algorithm is preconditioned by a scaled two-step limited memory BFGS method in Nash TN method with Powell's restarting strategy used to reset the preconditioner periodically. A detailed description of the preconditioner may be found in Nash (1985).

In the TN method, given a vector $\mathbf{v}$, the Hessian/vector product $\mathbf{G}_k\mathbf{v}$ required by the inner conjugate gradient algorithm may be obtained by a finite difference approximation,

$$\mathbf{G}_k \mathbf{v} \approx [\mathbf{g}(\mathbf{x}_k + h\mathbf{v}) - \mathbf{g}(\mathbf{x}_k)]/h.$$

A major issue is how to adequately choose h; in this work we use

$$h = \sqrt{\epsilon}(1 + \|\mathbf{x}_k\|_2)$$

where ϵ is the machine precision. The inner algorithm is terminated using the quadratic truncation test, which monitors a sufficient decrease of the quadratic model

$$\mathrm{q}_k = \mathbf{p}_k^T \mathbf{G}_k \mathbf{p}_k/2 + \mathbf{p}_k^T \mathbf{g}_k \times (1 - \mathrm{q}_k^{i-1})/\mathrm{q}_k^i \leq c_q/i$$

where i is the counter for the inner iteration and c_q is a tolerance, satisfying $0 < c_q < 1$.

If

$$c_q \leq \|\nabla f(\mathbf{x}_k)\|$$

then the truncated-Newton method will converge quadratically (see Dembo *et al.*, 1982; Nash and Sofer, 1990; 1996). This fact explains the faster rate of convergence and the better quality of results obtained with the TN method. For similar results in optimal control see LeDimet *et al.* (2002).

The Hybrid Method

The hybrid method aims at dynamically combining the best features of both systems in the manner of alternating l steps of L-BFGS with t steps of HFN

$$l{*}(\mathrm{L-BFGS}) \rightarrow t{*}(\mathrm{HFN(PCG)}) \rightarrow \mathbf{H}(m), \quad \text{repeat}$$

where $\mathbf{H}(m)$ is a limited memory matrix approximating the inverse of the Hessian while m denotes the number of correction pairs stored.

In the L-BFGS cycle, $\mathbf{H}(m)$ (starting say from the initial unit, or weighted unit matrix) is updated using most recent m pairs. The matrix obtained at the end of L-BFGS cycle is used to precondition the first of the t HFN iterations. In the remaining $t - 1$ iterations the limited memory matrix $\mathbf{H}(m)$ is updated using information generated by the inner preconditioned conjugate-gradient (PCG) iteration and it is used to precondition the next HFN iteration. A detailed description of the preconditioning process is presented by Morales and Nocedal (2001). At the end of t HFN steps, the most current $\mathbf{H}(m)$ matrix is used as the initial matrix in a new cycle of L-BFGS steps.

THE OPTIMIZATION PROBLEM

In this section, we present the variational data assimilation problem for a two-dimensional shallow-water model. The simplicity of the shallow water equations facilitates a thorough investigation of various numerical methods (Williamson *et al.*, 1992) while capturing important characteristics present in more comprehensive oceanographic and atmospheric models.

The Shallow Water Model

In a Cartesian coordinate system, the dynamical model is represented as a system of nonlinear partial differential equations

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} - fv + \frac{\partial \phi}{\partial x} = 0 \tag{3}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + fu + \frac{\partial \phi}{\partial y} = 0 \tag{4}$$

$$\frac{\partial \phi}{\partial t} + \frac{\partial u\phi}{\partial x} + \frac{\partial v\phi}{\partial y} = 0 \tag{5}$$

where u and v are the components of the horizontal velocity, ϕ is the geopotential and f the Coriolis parameter. The spatial domain considered is a 6000 km × 4400 km channel with a uniform 21×21 spatial grid, such that the dimension of the discrete state vector $\mathbf{x} = (u, v, \phi)$ is 1083. The initial conditions are specified as in Grammeltvedt (1969) and the model is integrated for 10 h using a leap-frog scheme with a time increment $\Delta t = 600$ s. The geopotential at $t_0 = 0$ and $t = 10$ h is displayed in Fig. 1.

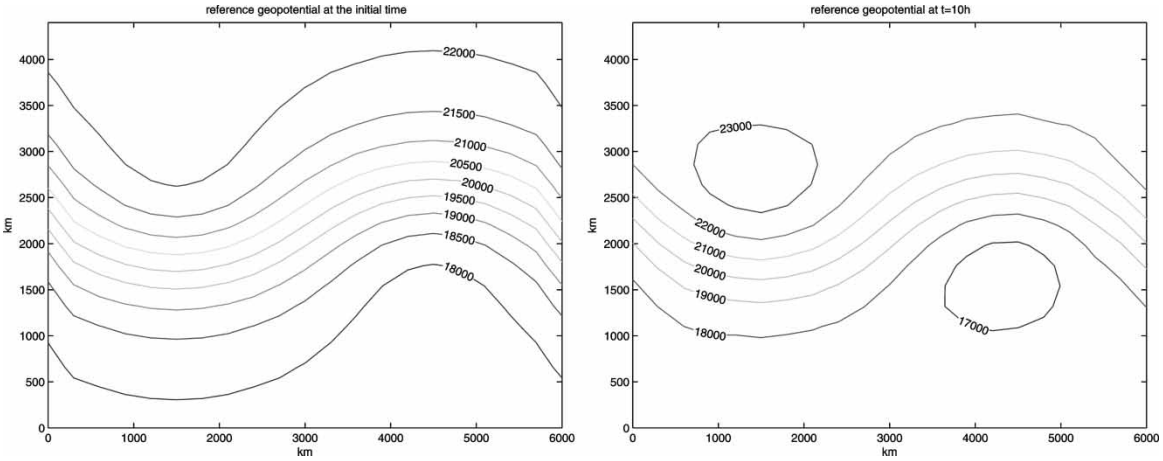

FIGURE 1 The configuration of the geopotential ($m^2\,s^{-2}$) for the reference run at the initial time $t = 0$ h and at $t = 10$ h.

4D-Var Data Assimilation

The data assimilation problem is implemented in an idealized framework (twin experiments) by considering a truth and a control experiment with the initial conditions as control parameters. We assume that the initial conditions described in the previous section represent the true (reference) state $\mathbf{x}_0^{\text{ref}}$ and the model trajectory $\mathbf{x}^{\text{ref}}(t)$ obtained during the reference run is used to provide observations. Next, the initial conditions $\mathbf{x}_0^{\text{ref}}$ are perturbed with random values chosen from a uniform distribution to obtain a perturbed state $\mathbf{x}_0^p$ that serves as initial conditions for the control run. In Fig. 2, we show the configuration of the perturbed geopotential at t_0 and after a 10 h run. A cost functional that measures the least-squares distance between the reference run $\mathbf{x}^{\text{ref}}$ and the control run $\mathbf{x}$ in the assimilation window $[0, 10]$ h is defined as follows:

$$f(\mathbf{x}) = \frac{1}{2}\sum_{i=0}^{n}\left[\mathbf{x}(t_i) - \mathbf{x}^{\text{ref}}(t_i)\right]^T \mathbf{W}_i\left[\mathbf{x}(t_i) - \mathbf{x}^{\text{ref}}(t_i)\right]. \quad (6)$$

In this study, we selected time-invariant diagonal weight matrices $\mathbf{W}_i = \text{diag}(1, 1, 0.01)$.

The data assimilation procedure aims to provide an optimal analysis of the state evolution by minimizing the cost functional (6). Numerical integration of the model equations (3)–(5) provides an explicit dependence of the state evolution in terms of the initial conditions $\mathbf{x}(t_i) = M_i(\mathbf{x}_0)$. Therefore, an optimal initial state $\mathbf{x}_0^*$ is obtained by solving the unconstrained optimization problem

$$\min_{\mathbf{x}_0} f(\mathbf{x}_0). \quad (7)$$

For our idealized experimental settings, $\mathbf{x}_0^* = \mathbf{x}_0^{\text{ref}}$ such that the value of the cost functional at the optimal point must be zero.

Characteristics of the Cost Functional

In this section, we investigate several properties of the optimization problem that have a high impact on the performance of the optimization algorithm: the cost of the function-gradient evaluation, the "degree of nonlinearity" of f, convexity, and the condition number of the Hessian matrix $\mathbf{G}(\mathbf{x}) = \nabla^2 f(\mathbf{x})$.

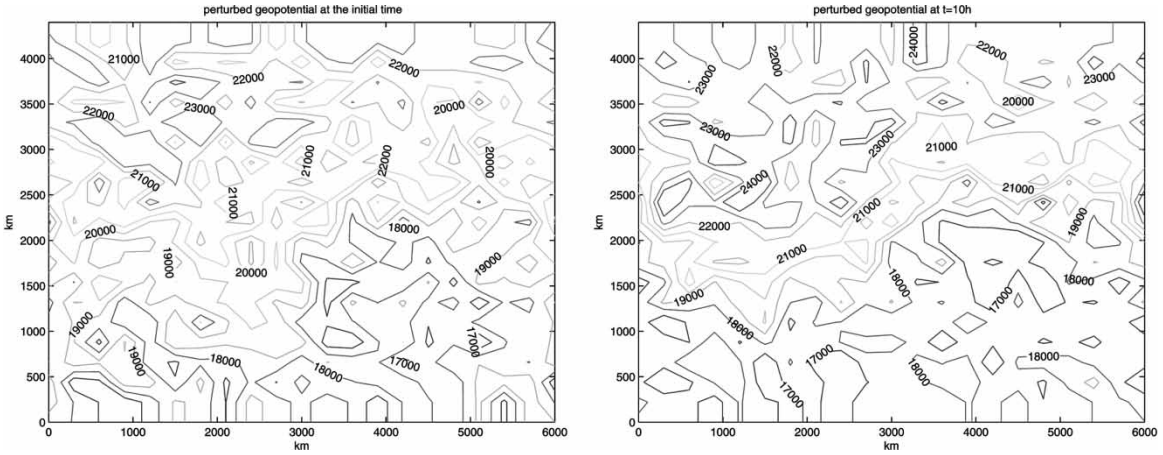

FIGURE 2 The configuration of the geopotential ($m^2\,s^{-2}$) for the initial guess run at the initial time $t = 0$ and at $t = 10$ h.

Implementation of large-scale minimization algorithms for the data assimilation procedure requires the evaluation of both the cost functional (7) and its gradient $\nabla_{\mathbf{x}_0} f$. Each evaluation of the cost functional requires an integration of the model equations (3)–(5), whereas the gradient is obtained through the backward integration of the adjoint model associated with the forward model equations (3)–(5) as explained in LeDimet *et al.* (2002). Using a hand-coded discrete adjoint model, we obtained an average ratio $\mathrm{CPU}(\nabla_{\mathbf{x}_0} f)/\mathrm{CPU}(f) \approx 2.4$ of the CPU time required to evaluate the gradient to the CPU time required to evaluate the cost functional.

The degree of nonlinearity of the cost function may be assessed by considering the "deviation from quadratic"

$$\mathrm{DQ} = \frac{\|g(\mathbf{x}_0) - g(\mathbf{x}_0^*) - \mathbf{G}(\mathbf{x}_0^*)(\mathbf{x}_0 - \mathbf{x}_0^*)\|_\infty}{\|(\mathbf{x}_0 - \mathbf{x}_0^*)\|_\infty^2} \qquad (8)$$

which gives a measure of the size of the third-order derivatives. If DQ has a small value, then the problem is approximately quadratic. The evolution of the deviation from quadratic of the functional (7) during the L-BFGS iteration is shown in Fig. 3 and reveals a small degree of nonlinearity. For this type of problem, the results presented in the study of Nash and Nocedal (1991) suggest that the TN method is expected to outperform L-BFGS.

Hessian information is crucial in many aspects of both constrained and unconstrained minimization. To obtain sufficient conditions for the existence of the minimum of the multivariate unconstrained problem, the Hessian matrix must be positive definite at $\mathbf{x}_0^*$. Iterative methods (e.g. the Lanczos method) that require only matrix/vector products may be used to evaluate few extremal eigenvalues of the Hessian matrix and its condition number $k(\mathbf{G}) = \lambda_{\max}/\lambda_{\min}$. This information may be used to predict the behavior and convergence rate for unconstrained optimization algorithms. Using the ARPACK package (Lehoucq *et al.*, 1998), we evaluated the largest and smallest Hessian eigenvalues. We obtained at the initial guess point $\lambda_{\max} \approx 529.4$, $\lambda_{\min} \approx 0.03$ and at the optimal point $\lambda_{\max} \approx 512.9$, $\lambda_{\min} \approx 0.014$. The condition number is $k(\mathbf{G}) = O(10^4)$ and a slow convergence rate is expected for optimization methods that use only gradient information.

NUMERICAL RESULTS AND DISCUSSION

In this section, we analyze the L-BFGS, TN, and hybrid optimization algorithms when applied to the 4D-Var data assimilation problem. The performance of each method is evaluated in terms of the number of iterations *NIT*, number of function/gradient evaluations *NFG*, and CPU time

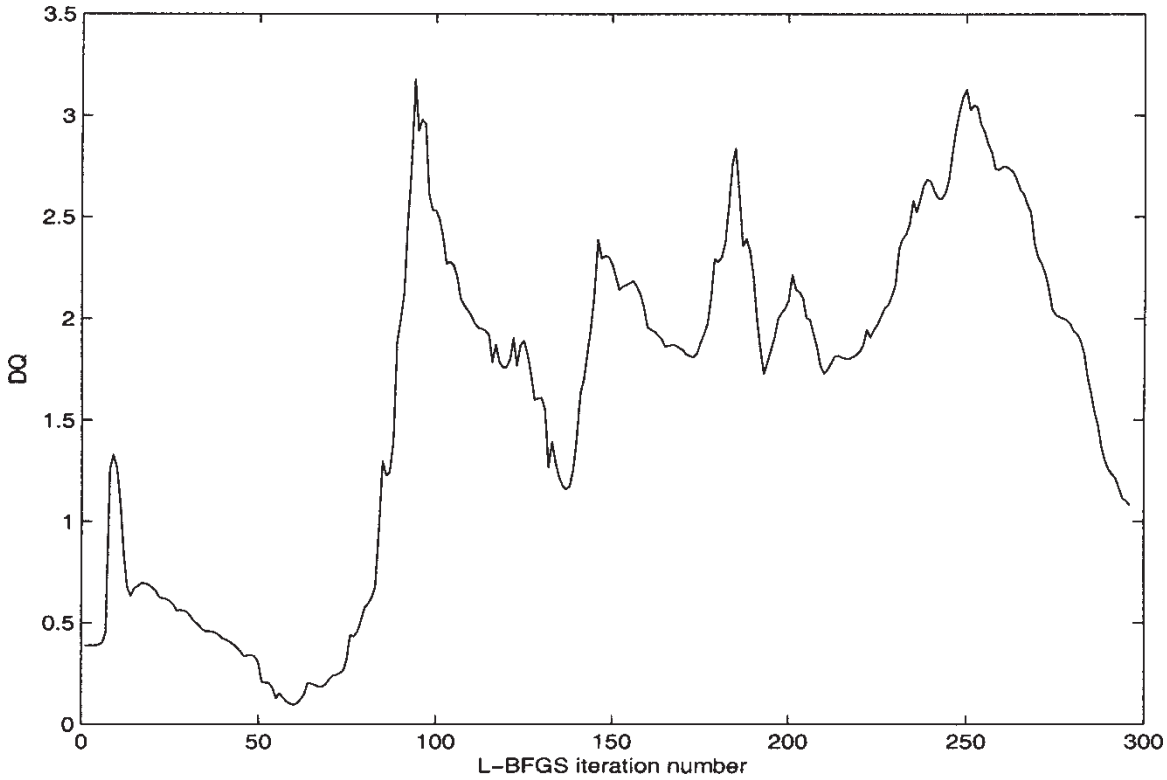

FIGURE 3 Deviation from quadratic (DQ) of the cost functional (7) during the L-BFGS iteration. Small DQ values indicate that the optimization problem is approximately quadratic.

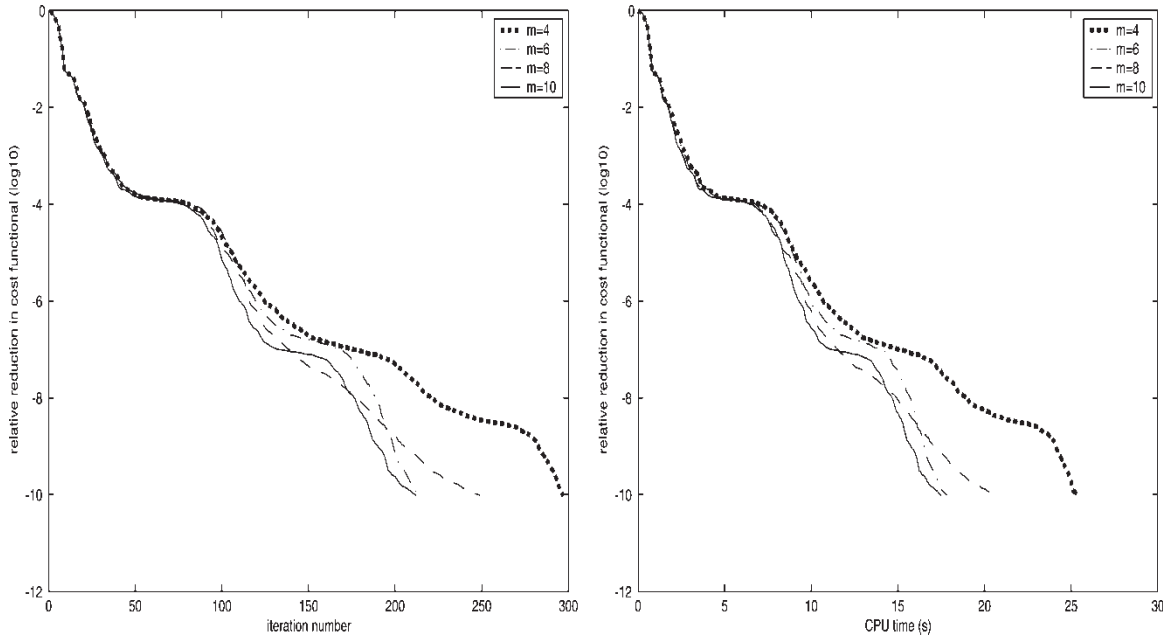

FIGURE 4 The performance of the L-BFGS algorithm for various values of the number m of correction pairs stored. Relative reduction of the cost functional is shown as a function of the number of iterations (left) and as a function of the CPU time (right) required by the optimization algorithm. Results are displayed for $m = 4, 6, 8$ and 10 and it can be seen that the selection of m has a significant impact on the algorithm performance in the vicinity of the optimal point.

required to achieve a prescribed relative reduction in the cost functional.

Each of the three optimization methods proceeds until a relative reduction in the cost functional (7)

$$f(\mathbf{x}_0^k)/f(\mathbf{x}_0) \leq 10^{-10} \tag{9}$$

is obtained.

The L-BFGS algorithm is sensitive to the number m of correction pairs stored. We tested several values in the range $4 \leq m \leq 10$ and the best results were obtained with $m = 10$, as shown in Fig. 4. In Table I, we report the performance of the algorithm to satisfy the criteria (9), for each value $m = 4$, 6, 8 and 10 we notice that the best results were obtained for $m = 10$. Further increasing m did not exhibit any significant improvements and for the hybrid code experiments presented next in this section we selected $m = 10$.

The TN iterations are highly sensitive to the termination criteria used for the inner CG iteration, and a trade-off between the computational cost and the performance must be considered. The inner iteration was terminated if the CG residual satisfied the convergence criteria

$$\|r\|_2 \leq \eta_k \|g_k\|_2, \quad \eta_k = \min(0.5/k, \|g_k\|_2)$$

or a user specified maximum number of inner iterations *maxit* was achieved. The impact of the selection of the method parameter *maxit* on the performance of the algorithm is shown in Fig. 5 where we show the number of iterations and the CPU time vs. the relative reduction in the cost functional for *maxit* = 5, 10, 20 and 30. Increasing *maxit* will result in a smaller number of HFN (outer) iterations, since the Newton equation (2) is solved more accurately. However, each CG (inner) iteration becomes more expensive and choosing a large *maxit* may lower the efficiency of the algorithm by increasing the number of function/gradient evaluations (and therefore, the CPU time). The performance of the HFN algorithm to satisfy the criteria (9) is outlined in Table II for *maxit* = 5, 10, 20 and 30. We notice that although using *maxit* = 5 will take more than twice as many outer iterations as with *maxit* = 20, the results are much closer in terms of the overall CPU time required by optimization.

To implement the hybrid algorithm, in addition to m and *maxit* the user must specify the initial number of L-BFGS iterations (l) and HFN iterations (t). Afterwards, a dynamic adjustment of the lengths (l, t) of the L-BFGS and HFN cycles is implemented in the hybrid code as explained by Morales and Nocedal (2002). If $l = 0$ then the pure HFN method is implemented, whereas by setting $t = 0$ the standard L-BFGS method is performed. We experimented with various settings of the parameters l and t while keeping constant *maxit* and found that setting $l = 20$, $t = 10$ provided in general the best performance.

TABLE I The performance of the L-BFGS algorithm applied to the variational data assimilation problem

m	*NIT*	*NFG*	CPU (s)
4	296	316	25.3
6	214	222	17.8
8	248	257	20.6
10	211	213	17.5

Results displayed for $m = 4$, 6, 8 and 10 show that the selection of the number m of pairs stored has a significant impact on the algorithm performance.

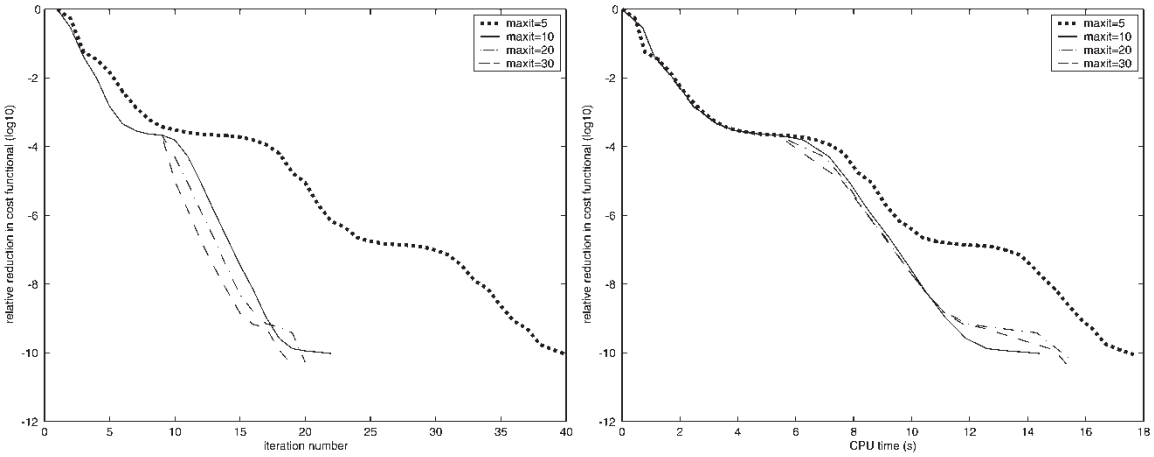

FIGURE 5 The performance of the TN algorithm for various specifications of the maximum number of iterations *maxit* used in the inner CG iteration. Relative reduction of the cost functional is shown as a function of the number of outer iterations (left) and as a function of the CPU time (right) required by the optimization algorithm. Results are displayed for *maxit* = 5, 10, 20 and 30.

The results for *maxit* = 20 are reported in Table III and a graphical illustration is presented in Fig. 6. By comparison with the results in Tables I and II, we notice that the hybrid algorithm with $m = 10$, $maxit = 20$, $l = 20$, $t = 10$ outperformed the best results obtained with each of the L-BFGS and TN methods implemented individually.

SUMMARY AND CONCLUDING REMARKS

A comparative analysis of the performance of a hybrid method vs. L-BFGS and Hessian-free TN optimization methods was presented in the context of 4D-Var data assimilation. The study presented in this paper is of particular interest since the optimization problem in variational data assimilation is often characterized by a large dimension of the vector of control variables, large condition number of the Hessian (ill-conditioned), and small deviation from quadratic. It is known that for problems that share these properties TN methods are more effective than L-BFGS (Nash and Nocedal, 1991). The characteristics of the cost functional were illustrated for a two-dimensional shallow-water model. All three optimization methods successfully solved the assimilation problem. However, the numerical experiments revealed that the performance of the algorithms is highly influenced by the selection of the method parameters. The L-BFGS iteration provided a slow convergence and we found its performance in the vicinity of the optimal point to be sensitive to the number of correction pairs stored. The HFN method provided in general the fastest convergence with an appropriate specification of the number of inner CG iterations. Due to the inherent algorithmic demands of 4D-Var data assimilation, the efficiency of the Hessian-free TN method should be weighted against its computational cost. Thus the number of inner preconditioned CG iterations should be restricted since each such iteration entails a function and gradient evaluation requiring integration of the forward model and its adjoint. This issue becomes of preponderant importance if we deal with 3D numerical weather prediction models due to the large computational burden. This explains why L-BFGS is still prevalent in operational 4D-Var implementation at major numerical weather prediction centers.

If a second-order adjoint of the model is available, the number of required Hessian-free iterations may be reduced, since exact Hessian/vector products are computed (LeDimet *et al.*, 2002). The hybrid approach was found to be superior to the L-BFGS method and, with a proper tuning of the parameters *maxit*, *l* and *t*, also to the HFN method. However, finding the optimal values for the parameters is a delicate issue and depends on the optimization problem

TABLE II The performance of the truncated-Newton algorithm applied to the variational data assimilation problem

Maxit	*NIT*	*NFG*	CPU (s)
5	39	221	17.7
10	21	180	14.4
20	19	195	15.6
30	18	192	15.4

Results displayed for *maxit* = 5, 10, 20 and 30 show that the selection of the maximum number of inner CG iterations *maxit* has a significant impact on the algorithm performance. Best trade-off between the accuracy and the computational cost to solve for the Newton direction was obtained for *maxit* = 10.

TABLE III The performance of the hybrid algorithm applied to the variational data assimilation problem

l	*t*	*NIT*	*NFG*	CPU (s)
10	5	57	214	17.2
10	10	54	211	16.9
20	10	59	176	14.1
20	20	66	220	17.6

Results are shown for various specifications of the lengths (l, t) of the L-BFGS and TN cycle while keeping constant $m = 10$ and $maxit = 20$. *NIT* represents the number of L-BFGS iterations plus the number of outer TN iterations taken during the optimization. By comparison with the results in Tables I and II, we notice that the hybrid algorithm with $l = 20$, $t = 10$ provided the overall best performance.

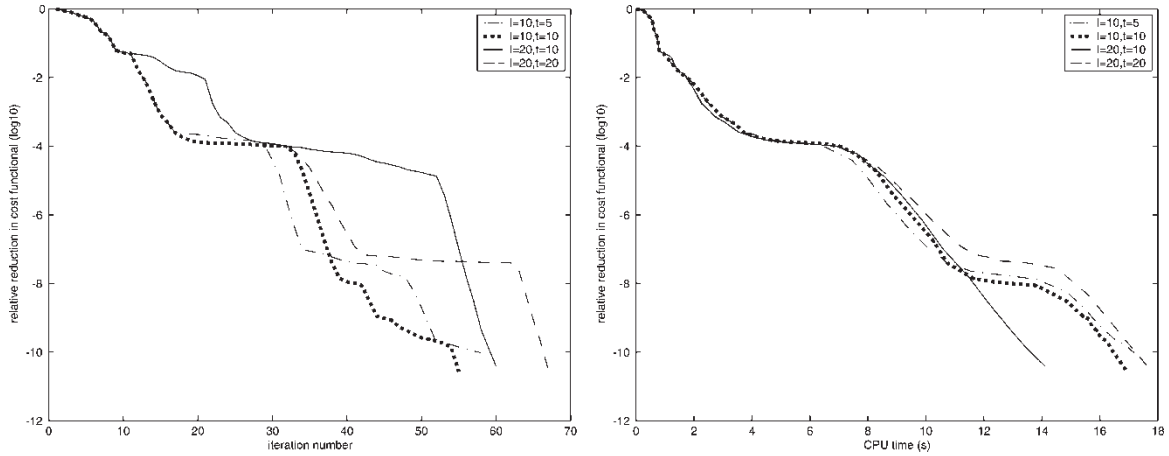

FIGURE 6 The performance of the hybrid algorithm for various specifications of the initial lengths (l, t) of the L-BFGS and HFN cycles. Relative reduction of the cost functional is shown as a function of the number of the L-BFGS + outer TN iterations (left), and as a function of the CPU time required by the optimization algorithm (right).

characteristics. For practical applications, the success of the hybrid code will largely depend on its efficient implementation in conjunction with the availability of second-order adjoint to 4D-Var with simplified physics using global circulation models.

The issue of dimensionality in 3D operational models will also be compounded by the impact of nonlinearity due to physical processes such as precipitation and radiation. Moreover, if an adjoint with physics is used for data assimilation with an operational model, and in particular if full physics is used, we may still encounter the issues of on/off processes. This in turn may require (if a version with simplified physics was not developed) the use of non-smooth optimization. The issue of proper preconditioning will also take another aspect since due to the higher dimensionality of the problem we should expect much larger condition numbers. Hence, further experiments are needed for 3D operational models where we may reach different conclusions than those reached here.

Acknowledgements

The first author acknowledges the support from the Supercomputing Institute for Digital Simulation and Advanced Computation of the University of Minnesota. The second author would like to acknowledge the support from NSF grant ATM-0201808 managed by Dr Linda Peng, whom he would like to thank for her support.

References

Byrd, R.H., Nocedal, J. and Schnabel, R.B. (1994) "Representation of quasi-Newton matrices and their use in limited memory methods", *Math. Program.* **63**(4), 129–156.

Dembo, R.S., Eisenstat, S.C. and Steihaug, T. (1982) "Inexact Newton methods", *SIAM J. Num. Anal.* **19**, 400–408.

Gilbert, J.C. and Lemarechal, C. (1989) "Some numerical experiments with variable-storage quasi-Newton algorithms", *Math. Prog.* **45**, 407–435.

Grammeltvedt, A. (1969) "A survey of finite-difference schemes for the primitive equations for a barotropic fluid", *Mon. Weather Rev.* **97**, 387–404.

Kalnay, E. (2002) Atmospheric Modeling, Data Assimilation and Predictability (Cambridge University Press, Cambridge) p 512.

LeDimet, F.X., Navon, I.M. and Daescu, D.N. (2002) "Second order information in data assimilation", *Mon. Weather Rev.* **130**(3), 629–648.

Lehoucq, R.B., Sorensen, D.C. and Yang, C. (1998) "ARPACK user's guide: solution of large-scale eigenvalue problems with implicitly restarted arnoldi methods", Software, Environments, and Tools (SIAM) Vol. **6**, p 160.

Liu, D.C. and Nocedal, J. (1989) "On the limited memory BFGS method for large scale minimization", *Math. Prog.* **45**, 503–528.

Morales, J.L. and Nocedal, J. (2000) "Automatic preconditioning by limited memory quasi-Newton updating", *SIAM J. Optim.* **10**, 1079–1096.

Morales, J.L. and Nocedal, J. (2001) "Algorithm 809: PREQN: Fortran 77 subroutines for preconditioning the conjugate gradient method", *ACM Trans. Math. Soft.* **27**(1), 83–91.

Morales, J.L. and Nocedal, J. (2002) "Enriched methods for large-scale unconstrained optimization", *Comput. Optim. Appl.* **21**, 143–154.

Nash, S.G. (1984) "Truncated-Newton methods for large-scale function minimization", In: Rauch, H.E., ed, Applications of Nonlinear Programming to Optimization and Control (Pergamon Press, Oxford) pp 91–100.

Nash, S.G. (1985) "Preconditioning of truncated-Newton methods", *SIAM J. Sci. Statist. Comput.* **6**, 599–616.

Nash, S.G. and Nocedal, J. (1991) "A numerical study of the limited memory BFGS method and the truncated-Newton method for large scale optimization", *SIAM J. Optim.* **1**, 358–372.

Nash, S.G. and Sofer, A. (1990) "Assessing a search direction within a truncated-Newton method", *Oper. Res. Lett.* **9**, 219–221.

Nash, S.G. and Sofer, A. (1996) Linear and Nonlinear Programming (McGraw-Hill, New York, NY).

Navon, I.M., Zou, X., Derber, J. and Sela, J. (1992) "Variational data assimilation with the N.M.C. spectral model. Part I: adiabatic model tests", *Mon. Weather Rev.* **120**(7), 1433–1446.

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

Nocedal, J. and Wright, S. (1999) Numerical Optimization (Springer Verlag, New York).

Schlick, T. and Fogelson, A. (1992) "TNPACK-A truncated Newton minimization package for large-scale problems: I. Algorithm and usage", *ACM Trans. Math. Soft.* **18**(1), 46–70.

Wang, Z., Navon, I.M., Zou, X. and LeDimet, F.X. (1995) "A truncated-Newton optimization algorithm in meteorology applications with analytic Hessian/vector products", *Comput. Optim. Appl.* **4**, 241–262.

Wang, Z., Droegemeier, K.K. and White, L. (1998) "The adjoint Newton algorithm for large-scale unconstrained optimization in meteorology applications", *Comput. Optim. Appl.* **10**, 283–320.

Williamson, D.L., Drake, J.B., Hack, J.J., Jakob, R. and Swarztrauber, P.N. (1992) "A standard test set for numerical approximations to the shallow water equations in spherical geometry", *J. Comput. Phys.* **102**, 211–224.

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