Derivative Convergence for Iterative Equation Solvers

Andreas Griewank
Christian Bischof
George Corliss
Alan Carle
Karen Williamson

CRPC-TR93326
July 1993

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

Accepted for publication in Optimization Methods and Software.

Derivative Convergence for Iterative Equation Solvers*

Andreas Griewank and Christian Bischof, Argonne National Laboratory
George Corliss, Marquette University and Argonne National Laboratory
Alan Carle and Karen Williamson, Rice University

July 16, 1993

When nonlinear equation solvers are applied to parameter-dependent problems, their iterates can be interpreted as functions of these variable parameters. The derivatives (if they exist) of these iterated functions can be recursively evaluated by the forward mode of automatic differentiation. Then one may ask whether and how fast these derivative values converge to the derivative of the implicit solution function, which may be needed for parameter identification, sensitivity studies, or design optimization.

It is shown here that derivative convergence is achieved with an R-linear or possibly R-superlinear rate for a large class of memoryless contractions or secant updating methods. For a wider class of multistep contractions, we obtain R-linear convergence of a simplified derivative recurrence, which is more economical and can be easily generalized to higher-order derivatives. We also formulate a constructive criterion for derivative convergence based on the implicit function theorem. All theoretical results are confirmed by numerical experiments on small test examples.

KEY WORDS: Derivative convergence, automatic differentiation, implicit functions, preconditioning, Newton-like methods, secant updates.

1 INTRODUCTION AND ASSUMPTIONS ON $F(x,t)=0$

Many functions of practical interest are defined implicitly as solutions to differential or algebraic equations. The values of these functions are typically evaluated by iterative procedures with a variable number of steps and with various, often discontinuous, adjustments. The corresponding computer programs contain branches, and the results are often (strictly speaking) not everywhere differentiable in the data. Then one may ask whether and how automatic differentiation can still be expected to yield derivative values that are reasonable approximations to the underlying implicitly defined derivatives.

Automatic, or computational, differentiation is a chain rule based technique for evaluating the derivatives of functions defined by algorithms, usually in the form of computer programs written in Fortran, C, or some other high level language. If the program theoretically can be unrolled into a finite sequence of arithmetic operations and elementary function calls, then derivatives can be propagated recursively. Exceptions arise when there is a division by zero or when one of the elementary functions is evaluated at a point of non-differentiability. These local contingencies are easily detected and arise only in marginal situations where the undifferentiated evaluation algorithm is already poorly conditioned. For a general review of the theory, implementation, and application of automatic differentiation, see [16].

Rather than as a practical problem for automatic differentiation, one can also view the question raised here as a purely theoretical one, namely, whether the iterates generated for parameter-dependent problems converge not only pointwise, but also with respect to some Sobolev norm involving derivatives with respect

*This work was supported by the Office of Scientific Computing, U.S. Department of Energy, under Contract W-31-109-Eng-38, by the National Aerospace Agency under Purchase Order L25935D, by the National Science Foundation through the Center for Research on Parallel Computation under Cooperative Agreement CCR-9120008, and by the W. M. Keck Foundation.

to the parameters. This theoretical aspect will not be fully explored here, as only pointwise convergence of the derivatives is established. Throughout we will analyze the situation where a nonlinear system

$$F(x,t) = 0 \quad \text{with} \quad F : \mathbb{R}^n \times \mathbb{R} \mapsto \mathbb{R}^n$$

is solved for $x(t)$ for fixed t by an iteration of the form

$$x_{k+1} = \Phi_k(x_k,t) \equiv x_k - P_k F(x_k,t) , \tag{1}$$

where P_k is some $n \times n$ matrix, which we will consider as a preconditioner. Without loss of generality, we have restricted our framework to the case of a single scalar parameter $t \in \mathbb{R}$ since multivariate derivatives can always be constructed from families of univariate derivatives [2]. Total derivatives with respect to t will be denoted by primes, and partial derivatives (with x kept constant) by the subscript t.

In this paper, we consider two approaches to computing the desired implicitly defined derivative $x'(t)$. The **"simplified"** approach treats the P_k as if they were independent of t. The **"fully differentiated"** approach differentiates the entire iterative algorithm.

Obviously, any sequence $\{x_k\}_{k \geq 0}$ for which $F(x_k,t)$ never vanishes exactly can be written in form (1), unless we place some restriction on the $n \times n$ matrices P_k and thus the sequence of iteration functions Φ_k. The assumptions on the P_k that we will make are quite natural and almost necessary for a numerically stable iterative process.

Assumption 1 (Regularity) *For some fixed t, the iteration converges to a solution, so that*

$$x_k \longrightarrow x_* = x(t) \quad \text{with} \quad F(x(t),t) = 0 .$$

Moreover, on some ball with radius $\rho > 0$ centered at $(x(t),t)$, the function F is jointly Lipschitz-continuously differentiable and has a nonsingular Jacobian $F_x(x,t) = \partial F(x,t)/\partial x$ with respect to x, so that for two constants c_0, L, and for all $\|x - x(t)\| < \rho$,

$$\|F_x^{-1}(x,t)\| + \|[F_x(x,t), F_t(x,t)]\| \leq c_0 ,$$

and

$$\|[F_x(x,t), F_t(x,t)] - [F_x(x_*,t), F_t(x_*,t)]\| \leq L\|x - x_*\| ,$$

where we may use l_2 norms without loss of generality.

Under this assumption, local convergence is guaranteed for Newton's method with $P_k = F_x(x_k,t)^{-1}$ or for the Picard iteration with $P_k = I$ if the spectral radius of $(I - F_x)$ is less than one. If this condition is not met by the original system $F = 0$, one might try to find a fixed preconditioner $P_k = P$ so that $(I - P\,F_x)$ is a contractive mapping. Alternatively, one may select P_k as a function of x_k, for example by performing an incomplete triangular decomposition, so that we can write

$$P_k = P(x_k,t) .$$

Then P_k does not directly depend on the previous iterates, and we will refer to the iteration (1) as a memoryless contraction provided the following condition is met.

Assumption 2 (Contractivity) *The discrepancies*

$$D_k = [I - P_k\, F_x(x_k,t)]$$

satisfy

$$\delta_k \equiv \|D_k\| \leq \delta < 1 \tag{2}$$

with respect to some induced matrix norm so that in the limit

$$\delta_* \equiv \overline{\lim}_k\, \delta_k \leq \delta .$$

For the class of methods satisfying this contractivity assumption (which includes Newton's method with analytical Jacobians or divided difference approximations), convergence of the derivatives can be obtained easily. As an immediate consequence of Assumptions 1 and 2, we note that by standard arguments

$$\|P_k\| \leq c_0(1+\delta) \quad \text{and} \quad \|P_k^{-1}\| \leq c_0/(1-\delta)\,. \tag{3}$$

In the case of secant methods [13], the condition (2) is usually imposed for $k = 0$ and deduced for $k > 1$ to guarantee local convergence. If one assumes a certain kind of uniform linear independence for the sequence of the search directions, it can be shown [19] that $\delta_* = 0$. This is a sufficient, but by no means necessary, condition for Q-superlinear convergence. It can be enforced by taking so-called special steps [19] for the sole purpose of reducing the discrepancy D_k. We will see that $\delta_* = 0$ implies R-superlinear rather than just R-linear convergence of the derivatives. Hence, the extra expense of special updating steps might be justified on parameter-dependent problems. Secant methods are not memoryless because the preconditioners P_k are computed recursively from step to step. Therefore, they must be considered as functions of all previous points x_k and of the initial choice P_0. Since in formula (1), the matrix P_k must also absorb step multipliers, this functional dependence need not be smooth and may have discontinuities. In that case, the transition from x_k to x_{k+1} may also be nondifferentiable, so that the classical chain rule is not directly applicable.

Even when x_0, P_0, and all subsequent P_k are smooth functions of t, it may be uneconomical to calculate the corresponding derivatives explicitly. For example, in the case of Newton's method, the explicit calculation of derivatives would involve the propagation of derivatives through the triangular decomposition of the Jacobian, a process that involves $n^3/3$ arithmetic operations in the dense case. However, we know from the implicit function theorem that

$$F_x(x(t),t)\,x'(t) \;=\; -F_t(x(t),t)\,. \tag{4}$$

In particular, this means that $x'(t)$ is defined in terms of the first derivatives of F alone and does not depend on the second derivatives F_{xx} and F_{xt} . Yet these tensors come implicitly into play if derivatives with respect to x are propagated through the Newton iteration function $\Phi_k(x_k,t) = x_k - F_x(x_k,t)^{-1}F(x_k,t)$. The same applies to any other iteration where the preconditioner P_k depends in some way on derivatives of F with respect to x or t. Therefore, we will examine a simplified derivative recurrence, where the P_k are considered as (piecewise) constants with respect to the total differentiation of the recurrence (1) with respect to t. We call this the *simplified approach.*

On the other hand, it may be difficult to determine which quantities in a complicated nonlinear equation solver need to be differentiated and which can be considered as constants because they belong to the calculation of the preconditioner P_k. This distinction must then be conveyed to the automatic differentiation software by suitably annotating the code or retyping some of its variables. Therefore, one may prefer to adopt a black box approach and differentiate the whole iterative algorithm as though it were a straight line code. This is what we call the *fully differentiated approach.* Also, the derivative $x'_k = dx_k/dt$ of the iterate x_k that is finally accepted does represent the local tangent of the approximate solution set, which should be close to the exact solution curve if the convergence occurs with some degree of uniformity.

For either the simplified or fully differentiated approach, it seems pretty clear that the derivatives cannot converge faster than the iterates themselves, unless the problem is linear or has some other very special structure. We will show for Newton's method and for secant updating methods that the derivatives converge R-quadratically and R-linearly, respectively. Especially in the case of secant updating methods, we must therefore expect that the derivatives may lag behind the iterates during the final approach to the solution. Fortunately, we can constructively check the accuracy of any derivative approximation so that a premature termination can be avoided if accurate derivative values are required.

Gilbert showed in [15] that the derivatives dx_k/dt converge in the limit to the desired tangent $x' = x'(t)$ provided that the spectral radius of $\partial\Phi(x,t)/\partial x$ is less than one in the vicinity of (x_*, t). This fundamental result has removed some serious doubts regarding the general applicability of automatic differentiation. It has been verified on several large codes, including cases [3] where the assumptions of Gilbert's theorem do not appear to be satisfied. Therefore, we wish to relax the hypothesis and avoid derivatives that are not needed either from a theoretical or from a practical point of view. We will also establish rates of convergence, provide a practical stopping criterion, and extend the theory to higher derivatives and multistep contractions.

The paper is organized as follows. In the next section, we motivate the simplified and fully differentiated derivative recurrences and develop some basic mathematical relations. In Section 3, we establish R-linear derivative convergence for the simplified recurrence under Assumptions 1 and 2 alone and for the fully differentiated recurrence under the additional assumption that the update function of the P_k satisfies a certain differentiability condition. In Section 4, we present limited numerical experiments to illustrate our theoretical results. We compare the simplified and the full derivative recurrences for the Davidson-Fletcher-Powell update, and we consider using automatic differentiation to generate higher-order derivatives. We also investigate the performance of an optimization code in the more realistic setting of parameter identification problems when the necessary derivative information is supplied by the application of automatic differentiation to a function that is defined implicitly. In Section 5, we extend the results in Section 3 on R-linear convergence to methods such as cyclic reduction that are multistep contractive, but not one-step contractive. We conclude with a discussion of opportunities for further work.

2 SIMPLIFIED AND FULLY DIFFERENTIATED RECURRENCES

As we have indicated above, the basic recurrence (1) can be interpreted as one step of a Picard, or Richardson, iteration on the preconditioned nonlinear system

$$F_k(x,t) \equiv P_k\, F(x;t) = 0. \tag{5}$$

Provided P_k is nonsingular, as we will assume throughout, the solution set of each $F_k = 0$ is exactly the same as that of the original system $F = 0$. Consequently, the implicitly defined function $x(t)$ and its derivatives are independent from the sequence of preconditioners P_k. Their iterative evaluation certainly need not depend on the derivatives of P_k, which may not even exist.

Differentiating equation (5) with respect to t with P_k considered a constant, one obtains the equation defining $x'(t)$

$$P_k\, F_x(x(t),t)\, x'(t) = -P_k\, F_t(x(t),t) \quad . \tag{6}$$

In the following formulae, we will often suppress the dependence on t, which should be understood. Applying the Richardson iteration to the preconditioned linear system (6) of equations evaluated at the "current" iterate x_k, one obtains the recurrence

$$\tilde{x}'_{k+1} = \tilde{x}'_k - P_k\, [F_x(x_k,t)\, \tilde{x}'_k + F_t(x_k,t)] \ . \tag{7}$$

Here the tilde over $\tilde{x}'_k$ indicates that these approximations to the derivative $x'(t)$ are in general not the derivatives of the x_k with respect to t, which may or may not exist. Subtracting the actual implicitly defined derivative

$$x'_* \equiv -F_x(x_*,t)^{-1} F_t(x_*,t)$$

from both sides, we find that

$$\tilde{x}'_{k+1} - x'_* = D_k(\tilde{x}'_k - x'_*) + r'_k \ , \tag{8}$$

where

$$r'_k \equiv P_k[F_x(x_k,t)x'_* + F_t(x_k,t)] = \mathcal{O}(\|x_k - x_*\|) \ . \tag{9}$$

Since the perturbation r'_k tends to zero, equation (8) looks very much like a contraction and promises convergence of the x'_k to x'_*.

If the P_k are at least locally smooth functions of t so that the matrices $P'_k = dP_k(x(t))/dt$ are continuous, then the derivatives $x'_k = x'_k(t)$ exist, and (1) implies that they satisfy the recurrence

$$x'_{k+1} = x'_k - P_k\, [F_x(x_k,t)x'_k + F_t(x_k,t)] - P'_k\, F(x_k,t) \ , \tag{10}$$

which can be rewritten in the contractive form as

$$x'_{k+1} - x'_* = D_k(x'_k - x'_*) + r'_k \ - \ P'_k F(x_k,t) \ . \tag{11}$$

We refer to equation (7) as the *simplified* recurrence and to equation (10) as the *fully differentiated* recurrence. We hope this terminology helps avoid the danger of confusion with the Jacobian and Hessian update formulas ([4] and [5]) that lie at the heart of secant methods.

Provided the P'_k stay bounded or do not blow up too fast with increasing k, the last term in the linear recurrence (11) becomes more and more negligible as the residual $F(x_k,t)$ approaches zero. In the remainder, we will analyze equation (7) as a special case of equation (10) with P_k considered as constant on some neighborhood of the current t. Obviously the two-stage iteration defined by (1) and (10) can be stationary only at the (locally) unique fixed point $(x_k, x'_k) = (x_*, x'_*)$. In general, iteration (10) will never reach this fixed point exactly. However, the derivative approximations x'_k can have no limit other than the correct value x'_* unless the $P'_k F(x_k, t)$ converge to a nonzero vector. This possibility is remote: it can occur only if $\|P'_k\|$ tends to infinity exactly at the same rate as the reciprocal $1/\|F(x_k, t)\|$. Note that this cannot happen in the simplified derivative recurrence (7) for which $P'_k \equiv 0$ by definition. In the case of the full recurrence applied to secant methods, the Q-superlinear convergence rate ensures that the perturbation $P'_k F(x_k, t)$ tends to zero R-linearly, as we will show in the proof of Proposition 2 in Section 3.

In general, we expect the derivatives x'_k to exhibit roughly the same convergence behavior as the iterates x_k. To justify this optimism, we note that by Taylor's theorem,

$$P_k F(x_k, t) = P_k F_x(x_k, t)\,(x_k - x_*) - r_k\ ,$$

where

$$r_k = -P_k[F(x_k, t) - F_x(x_k, t)(x_k - x*)] = \mathcal{O}(\|x_k - x_*\|^2)\ . \tag{12}$$

Consequently, the iterates x_k defined by (1) satisfy the contractive recurrence

$$x_{k+1} - x_* = D_k(x_k - x_*) + r_k\ . \tag{13}$$

Hence we have essentially the same leading term in (8), (11), and (13). Taking norms, one obtains

$$\|x_{k+1} - x_*\| \le \|D_k\|\|x_k - x_*\| + \|r_k\|\ ,$$

so that the errors $\|x_k - x_*\|$ converge Q-linearly because of the contractivity assumption:

$$\overline{\lim}_k \frac{\|x_{k+1} - x_*\|}{\|x_k - x_*\|} \le \delta_*\ .$$

No matter how a derivative approximation x'_k was generated, its quality can be checked by evaluating the directional derivative

$$F'(x_k, t, x'_k) \equiv \left.\frac{\partial F(x_k + \tau\, x'_k, t + \tau)}{\partial \tau}\right|_{\tau=0} \tag{14}$$

$$= F_x(x_k, t)x'_k + F_t(x_k, t)\ . \tag{15}$$

This vector can be evaluated cheaply in the forward mode of automatic differentiation, without the need to form the (potentially very large) Jacobian $F_x(x_k, t)$. Note that $P_k F'(x_k, t, x'_*) = r'_k$ as defined in (9). When $F'(x_k, t, x'_k)$ vanishes exactly, x'_k represents the tangent of the perturbed solution set

$$\{x \in \mathbb{R}^n : F(x, t) = F(x_k, t)\}\ .$$

If $F'(x_k, t, x'_k)$ does not vanish, one can substitute into the right-hand side of (7) or (10) to improve the approximation. In general, the x'_k can approximate x'_* only as well as the x_k approximate x_*. Abbreviating

$$\rho_k \equiv \|x_k - x_*\| \quad \text{and} \quad \mu_k \equiv \|x'_k - x'_*\|$$

and setting

$$\eta_k \equiv (Lc_1 + \|P'_k\|)\rho_k \quad \text{with} \quad c_1 \equiv 2(c_0^2 + 1)\ , \tag{16}$$

one can bound the derivative errors as follows.

Lemma 1 *The regularity and contractivity imposed by Assumptions 1 and 2 imply that*

$$\mu_k \le \frac{1}{(1-\delta)}\|P_k F'(x_k, t, x'_k)\| + \frac{1}{2}Lc_0c_1\rho_k\ , \tag{17}$$

$$\mu_{k+1} \le \delta_k\mu_k + c_0\eta_k\ , \quad \text{and} \quad \|r'_k\| \le c_1c_0L\rho_k\ , \tag{18}$$

for all $\rho_k < \rho$.

Proof. First we show that the function $F_x(x,t)^{-1}F_t(x,t) : \mathbb{R}^n \rightarrow \mathbb{R}^n$ with t fixed has the Lipschitz constant $Lc_0c_1/2$ at x_*.

$$\begin{aligned}
&||F_x(x,t)^{-1}F_t(x,t) - F_x(x_*,t)^{-1}F_t(x_*,t)|| \\
&\quad \leq \ ||F_x(x,t)^{-1}[F_t(x,t) - F_t(x_*,t)]|| + ||[F_x(x,t)^{-1} - F_x(x_*,t)^{-1}]F_t(x_*,t)|| \\
&\quad \leq \ c_0\, L\, ||x - x_*|| + ||F_x(x,t)^{-1}|| \cdot ||F_x(x_*,t) - F_x(x,t)|| \cdot ||F_x(x_*,t)^{-1}|| \cdot ||F_t(x_*,t)|| \\
&\quad \leq \ c_0\, L\, ||x - x_*|| + c_0 L ||x - x_*|| c_0\, c_0 = c_0\, L\,(c_0^2 + 1)||x - x_*|| \ .
\end{aligned}$$

From the definition of $F'(x_k, t, x_k')$ in (14), we have

$$x_k' - x_*' = F_x^{-1}(x_k,t)F'(x_k,t,x_k') - \left[F_x(x_k,t)^{-1}F_t(x_k,t) + x_*'\right] \ .$$

After taking norms and using the Lipschitz constant just derived, we get

$$\mu_k \leq ||F_x^{-1}(x_k,t)F'(x_k,t,x_k')|| + c_0\, L\, c_1 \rho_k/2 \ .$$

One can replace the inverse $F_x^{-1}(x_k,t)$ in the first term on the right-hand side by P_k, noting that by the Banach perturbation lemma (e.g., [20]) and the definition of D_k in Assumption 2

$$||F_x^{-1}(x_k,t)P_k^{-1}|| = ||(I - D_k)^{-1}|| \leq 1/(1 - ||D_k||) \ ,$$

which establishes the first assertion.

To prove the third inequality, we derive from (9) by taking norms

$$\begin{aligned}
||r_k'|| \ &= \ ||P_k F_x(x_k,t)x_*' + P_k F_t(x_k,t)|| \\
&\leq \ ||P_k[F_x(x_k,t) - F_x(x_*,t)]|| \cdot ||x_*'|| + ||P_k[F_t(x_k,t) - F_t(x_*,t)|| \\
&\leq \ ||P_k|| L(c_0^2 + 1)\rho_k \leq 2c_0 L(c_0^2 + 1)\rho_k = Lc_0c_1\rho_k \ .
\end{aligned}$$

Here we have used that $||x_*'|| \leq ||F_x^{-1}(x_*,t)|| \cdot ||F_t(x_*,t)|| \leq c_0^2$ by Assumption 1. The last inequality follows since $||P_k|| = ||(I - D_k)F_x^{-1}|| \leq (1 + \delta)c_0$ as a consequence of Assumption 2. Finally, we derive from (11),

$$\begin{aligned}
\mu_{k+1} \ &\leq \ \delta_k \mu_k + ||r_k'|| + ||P_k' F(x_k,t)|| \\
&\leq \ \delta_k \mu_k + (Lc_0c_1 + ||P_k'||c_0)\rho_k \\
&\leq \ \delta_k \mu_k + c_0 \eta_k \ ,
\end{aligned}$$

where c_0 is a bound on the Jacobian F_x and hence a Lipschitz-constant for F, so that $||F(x,t)|| = ||F(x,t) - F(x_*,t)|| \leq c_0\rho_k$. ∎

The first equation of Lemma 1 provides us with a constructive stopping criterion for the derivative iteration, provided we can make some reasonable assumption regarding the sizes of L, c_0, and δ, which are also needed to bound $||x_k - x_*||$ in terms of $||F(x_k,t)||$ or $||P_k F(x_k,t)||$. The second inequality is the key to our convergence analysis in the following section.

3 DERIVATIVE CONVERGENCE FOR Q-LINEAR METHODS

First we will consider memoryless methods, where we may assume that $P_k = P(x_k,t)$ is continuously differentiable near (x,t) so that for some c_2 and all $\rho_k < \rho$,

$$||P_k'|| \ = \ ||P_x x_k' + P_t|| \ \leq \ c_2(\mu_k + 1) \quad . \tag{19}$$

This relation holds trivially with $c_2 = 0$ for the simplified iteration (7) where $P_k' = 0$.

Proposition 1 *Under Assumptions 1 and 2, condition (19) implies* **R-linear or R-superlinear** *convergence for the derivative recurrence (10). That is,*

$$\overline{\lim}_k \ ||x_k' - x_*'||^{1/k} \ \leq \ \delta_*. \tag{20}$$

Moreover, for all sufficiently small weights $\omega > 0$, the Sobolev norms

$$\|x_k - x_*\| + \omega\|x'_k - x'_*\|$$

converge **Q-linearly** *to zero. Furthermore, if $\delta_k \leq c\|x_k - x_*\|$, then we have* **R-quadratic** *convergence in that*

$$\overline{\lim}_k \, \|x'_k - x'_*\|^{1/2^k} \; < \; 1 \; ,$$

which applies for Newton's method, in particular.

Proof. Substituting (19) into the definition (16), we obtain

$$\eta_k \; \leq \; (Lc_1 + c_2)\rho_k + c_2\mu_k\rho_k \; ,$$

so that by (18)

$$\mu_{k+1} \; \leq \; (\delta_k + c_0c_2\rho_k)\mu_k + c_3\rho_k \; ,$$

where $c_3 = c_0(Lc_1 + c_2)$. Because of (12) and (13), we have by standard arguments,

$$\rho_{k+1} \leq \delta_k\rho_k + Lc_0\rho_k^2 \; .$$

Combining the last two inequalities for any ω, one obtains the ratio

$$\begin{aligned} \frac{(\rho_{k+1} + \omega\mu_{k+1})}{(\rho_k + \omega\mu_k)} \; &\leq \; \frac{(\delta_k + \omega c_3 + Lc_0\rho_k)\rho_k + \omega(\delta_k + c_0c_2\rho_k)\mu_k}{(\rho_k + \omega\mu_k)} \\ &\leq \; \delta_k + \omega c_3 + c_0(Lc_0 + c_2)\rho_k \; . \end{aligned}$$

The last bound has the limit superior $\delta_* + \omega c_3$, since we already know that the ρ_k converge to zero. This limiting ratio implies Q-linear convergence of the Sobolev norm, provided we choose $0 < \omega < (1 - \delta_*)/c_3$. Consequently, the linear R-factor of the sequence μ_k is less than or equal to any $\delta_* + c_3\omega$, and thus is not greater than δ_*, as asserted in (20). With the additional assumption on δ_k, we have for some c_4,

$$\mu_{k+1} \leq c_4(\mu_k + 1)\rho_k \; ,$$

which means that the convergent sequence $\{\mu_k\}$ is bounded by a multiple of the Q-quadratically convergent sequence $\{\rho_{k-1}\}$. ▮

Proposition 1 shows that for memoryless contractions, the fully differentiated recurrence (10) yields R-linear convergence and potentially R-superlinear convergence, a possibility that can occur only if the iterates themselves converge superlinearly. The same convergence rates are achieved by the simplified derivative recurrence (7), even when the preconditioners are updated recursively and are not differentiable. In the important case of Newton's method, either derivative recurrence converges R-quadratically—a rather satisfactory result.

Roughly speaking, we can claim in all these cases that the derivatives converge satisfactorily whenever the iterates x_k converge in a reasonably rapid and stable fashion. The simplest condition under which the x_k, x'_k, and $\tilde{x}'_k$ must all converge linearly to their respective limits is that the shifted Jacobians $D_k = [I - P_k\, F_x(x_k, t)]$ converge to a limit whose spectral radius is less than one. This condition was implied by the hypothesis of Gilbert's theorem [15] but must be considered quite restrictive. For example, the condition does not hold for Broyden's method nor for other popular quasi-Newton schemes, where $P_k = \alpha_k B_k^{-1}$. Here, α_k is a step multiplier, and B_k is an approximation to the Jacobian $F_x(x_k, t)$, which is not guaranteed to converge to $F_x(x_*, t)$ or to any other limit. However, under the usual assumption for local convergence of secant updating methods, it can be shown [6] that $\alpha_k \rightarrow 1.0$ and that $\|D_k\| < 0.5$ in the l_2 norm for all k. Then it follows from Proposition 1 that the simplified recurrence (7) must converge to the unique limit x'_*. This does not necessarily apply in case of the fully differentiated recurrence (10) because *a priori* nothing is known about the existence or the size of the P'_k.

The differentiability of the secant updates is in question because they contain rank-one terms of the form $y_k/\|s_k\|$, where both difference vectors

$$s_k \; \equiv \; x_{k+1} - x_k \quad \text{and} \quad y_k \; \equiv \; F(x_{k+1}, t) - F(x_k, t) \; \approx \; F_x(x_*, t)\, s_k$$

converge to zero. To prove that the matrix derivatives $||P_k'||$ do not blow up too fast, we make the observation that all classical updates and many other possible schemes can be written in the form

$$P_{k+1} \;=\; U(P_k, x_k, t, s_k, y_k) \quad , \tag{21}$$

where the update function

$$U \;:\; \mathbb{R}^{n\times n} \times \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^n \;\mapsto\; \mathbb{R}^{n\times n}$$

has the following property.

Assumption 3 (Lipschitzian Update) *There exist constants $c \geq 1$, $\rho < \infty$, $\delta < 1$, and $\gamma < \infty$ such that the domain conditions*

$$||P||\,,\; ||P^{-1}|| < c\,,\; ||x - x_*||\,,\; ||s|| < \rho\,, \quad \text{and} \quad ||Py - s|| < \delta||s|| \tag{22}$$

imply that U is differentiable at the point (P, x, t, s, y), and its partial derivatives satisfy

$$||U_P||\,,\; ||U_x||\,,\; ||U_t|| \leq \gamma\,, \quad \text{and} \quad ||U_s||\,,\; ||U_y|| \leq \gamma/||s||\,, \tag{23}$$

where P may be restricted to the open cone of symmetric positive definite matrices in $\mathbb{R}^{n\times n}$.

The crucial point here is that the partial derivatives with respect to s and y are bounded only by a multiple of the reciprocal step size $1/||s||$, which allows unbounded growth of the matrix derivatives $||P_k'||$. The key observation of the following proof is that the Q-superlinear convergence rate

$$\lim_k \frac{||x_{k+1} - x_*||}{||x_k - x_*||} \;=\; 0 \tag{24}$$

implies that the residuals $||F_k||$ decline just a bit faster than the $||P_k'||$ may grow. Before we formulate the second major result, let us briefly show that the Broyden update [4] and the DFP formula [9], [14], which do not explicitly depend on (x, t), satisfy the condition above.

Lemma 2 *The Broyden update function*

$$U(P, s, y) \;=\; P + \frac{(s - Py)s^T P}{s^T P y}$$

and the Davidon-Fletcher-Powell formula

$$U(P, s, y) \;=\; P - \frac{Pyy^T P}{y^T P y} + \frac{ss^T}{y^T s}$$

satisfy Assumption 3 with all norms $||\cdot||$ induced by the Euclidean vector norm.

Proof. For the nonsymmetric Broyden update, ρ is arbitrary, and δ may be any number between zero and one. Then we derive from the last domain condition in Assumption 3 that $s \neq 0$ and that

$$||y|| = ||P^{-1}Py|| \leq c||Py|| \leq c(1+\delta)||s|| < 2c||s||$$

as well as

$$||s|| \cdot ||y||c \geq ||s|| \cdot ||Py|| \geq s^T P y = s^T(Py - s) + s^T s \geq (1-\delta)\; |s||^2\,.$$

In particular, $||y|| \geq ||s||(1-\delta)/c$. Now let $P(\tau) \equiv P + \tau\dot{P}$ for some $\dot{P}$, and compute the derivative $\dot{U}$ of $U(P(\tau), s, y)$ with respect to τ at $\tau = 0$. Then we have by the chain rule with s and y kept constant,

$$\dot{U} = \dot{P} - \dot{P}ys^T P + (s - Py)s^T \dot{P}\Big/ (s^T P y) - (s - Py)s^T P(s^T \dot{P} y)/(s^T P y)^2\,,$$

so that by the triangle inequality in the L_2 norm

$$\begin{aligned} ||\dot{U}|| \;&\leq\; ||\dot{P}|| \cdot [1 + (||y|| \cdot ||P^T s|| + ||s||^2 + ||s|| \cdot ||Py||)\,/(s^T P y) \\ &\qquad + (||s|| + ||Py||)||P^T s|| \cdot ||s|| \cdot ||y||/(s^T P y)^2] \\ &\leq\; ||\dot{P}|| \cdot [1 + (2c^2 + 1 + 2c^2)/(1-\delta) + (1 + 2c^2)2c^2/(1-\delta)^2]\,. \end{aligned}$$

Since the direction $\dot{P}$ is arbitrary, the derivative U_P is uniformly bounded as required. Similarly, we find for the differentiation in some direction $\dot{s}$ with $s(\tau) = s + \tau\dot{s}$,

$$\begin{aligned} \|\dot{U}\| &\le \|\dot{s}\| \cdot [(\|P^T s\| + \|s - Py\| \cdot \|P\|)/(s^T Py) + (\|s\| + \|Py\|)\|P^T s\| \cdot \|Py\|/(s^T Py)^2] \\ &\le (\|\dot{s}\|/\|s\|) \cdot [(1 + 1 + 2c^2)c/(1-\delta) + (2 + 2c^2)2c^3)/(1-\delta)^2] \, , \end{aligned}$$

which implies that $U_s\|s\|$ is indeed uniformly bounded. Finally, we derive in the direction $\dot{y}$

$$\begin{aligned} \|\dot{U}\| &\le \|\dot{y}\| \cdot [\|P\| \cdot \|P^T s\|/(s^T Py) + (\|s\| + \|Py\|)\|P^T s\| \cdot \|P^T s\|/(s^T Py)^2] \\ &\le (\|\dot{y}\|/\|s\|) \cdot [c^2/(1-\delta) + (c + 2c^2)c^2/(1-\delta)^2] \, , \end{aligned}$$

which ensures that $U_y\|s\|$ is also uniformly bounded.

For the DFP formula, we must impose the restriction $\delta < 0.2\,c^{-2}$. Then we have

$$y^T s = y^T P P^{-1} s \ge s^T P^{-1} s - \|Py - s\| \cdot \|P^{-1}\| \cdot \|s\| \ge (1/c - c\delta)\|s\|^2 \ge 0.8\|s\|^2/c \, ,$$

where we have used the assumed positive definiteness of P to bound

$$s^T P^{-1} s \ge \|s\|^2/\|P\| \ge \|s\|^2/c \, .$$

As an immediate consequence, we have

$$y^T Py \ge y^T s - y^T(s - Py) \ge 0.8\|s\|^2/c - \delta\|y\| \cdot \|s\| \ge (0.8/c - \delta 2c)\|s\|^2 \ge 0.4\|s\|^2/c \, .$$

The rest of the argument is almost the same as for the Broyden update. Differentiating in some direction $\dot{P}$ with s and y held constant, we find

$$\dot{U} = \dot{P} - [\dot{P}yy^T P + Pyy^T \dot{P}]/(y^T Py) - [Pyy^T P](y^T \dot{P} y)/(y^T Py)^2 \, ,$$

so that, after taking norms,

$$\begin{aligned} \|\dot{U}\| &\le \|\dot{P}\| \, [1 + 2\|y\| \cdot \|Py\| 2.5c/\|s\|^2 + \|Py\|^2\|y\|^2 6.25c^2/\|s\|^4] \\ &\le \|\dot{P}\| \, [1 + 20c^4 + 100c^8] \le \|\dot{P}\|(1 + 10c_0^4)^2 \, . \end{aligned}$$

The derivatives with respect to y and s can be bounded by multiples of $\|s\|^{-1}$ in exactly the same fashion.
▮

Since Assumption 3 can also be verified for the BFGS update, it applies for a wide range of methods. Now we obtain for these updating methods almost the same result as in the memoryless case. The rather stringent restriction $\delta \le 0.2c^{-2}$ used in the proof for the DFP formula could be avoided if other conditions were placed on $y^T s$ and $y^T Py$. This would make perfect sense in the context of convex optimization, but we did not introduce them here because our primary focus is on the nonlinear equations case.

Proposition 2 *Under Assumptions 1, 2, and 3 with ρ and δ sufficiently small, the fully differentiated recurrence (10) yields R-linear or R-superlinear derivative convergence:*

$$\overline{\lim}_k \, \|x_k' - x_*'\|^{1/k} \;\le\; \delta_* \, .$$

Moreover,

$$\overline{\lim}_k \, [\|P_k'\|\|x_k - x_*\|]^{\frac{1}{k}} \le \delta_* \, ,$$

which limits the potential growth of the P_k' relative to the decline of the errors $\|x_k - x_*\|$.

Proof. Differentiating (21), we use the chain rule, the triangular inequality, and (23) to obtain

$$\begin{aligned} \tfrac{1}{\gamma}\,\|P_{k+1}'\| &\le \tfrac{1}{\gamma}\|U_P\, P_k' + U_x\, x_k' + U_t + U_s\, s_k' + U_y\, y_k'\| \\ &\le \|P_k'\| + \mu_k + \|x_*'\| + 1 + (\|s_k'\| + \|y_k'\|)/\|s_k\| \, . \end{aligned}$$

To bound the last two terms, we note that by (18) of Lemma 3,

$$\begin{aligned} \|s'_k\| &= \|x'_{k+1} - x'_k\| \leq \mu_{k+1} + \mu_k \\ &\leq (1+\delta)\mu_k + c_0\eta_k \leq 2\mu_k + c_0\eta_k \,. \end{aligned}$$

Similarly, we find

$$\begin{aligned} \|y'_k\| &= \|F'(x_{k+1}, t, x'_{k+1}) - F'(x_k, t, x'_k)\| \\ &\leq \|F_x(x_{k+1}, t)x'_{k+1} - F_x(x_k, t)x'_k\| + \|F_t(x_{k+1}; t) - F_t(x_k, t)\| \\ &\leq \|F_x(x_{k+1}, t)(x'_{k+1} - x'_*)\| + \|F_x(x_k, t)(x'_k - x'_*)\| \\ &\qquad + \|[F_x(x_{k+1}, t) - F_x(x_{k+1}, t)]x'_*\| + L(\rho_{k+1} + \rho_k) \\ &\leq c_0(\mu_{k+1} + \mu_k) + (c_0^2 + 1)L(\rho_{k+1} + \rho_k) \\ &\leq 2c_0\mu_k + c_0^2\eta_k + \eta_k \leq (c_0^2 + 1)(\mu_k + \eta_k) \,. \end{aligned}$$

Adding the last two inequalities and noting that $\|s_k\| \geq \rho_k - \rho_{k+1} \geq 0.9(1-\delta)\rho_k$, we find that for some c_5,

$$(\|s'_k\| + \|y'_k\|)/\|s_k\| \leq c_5(\mu_k + \eta_k)/\rho_k \,.$$

Now, since ρ_k is bounded, and η_k/ρ_k is bounded away from zero, the first four terms in (25), and an additional Lc_1 can be subsumed into the last bound, with c_5 growing to some c_6, so that

$$Lc_1 + \|P'_{k+1}\| \leq c_6(\mu_k + \eta_k)/\rho_k \quad .$$

After multiplication by ρ_{k+1}, we get

$$\eta_{k+1} \leq q_k(\mu_k + \eta_k) \quad \text{with} \quad q_k \equiv c_6\rho_{k+1}/\rho_k \to 0 \,.$$

Adding $\omega > \delta_*$ times this inequality to the bound (18), we find that

$$\frac{(c_0\eta_{k+1} + \omega\mu_{k+1})}{(c_0\eta_k + \omega\mu_k)} \leq \frac{c_0(q_k + \omega)\eta_k + (c_0q_k + \omega\delta_k)\mu_k}{(c_0\eta_k + \omega\mu_k)} \leq \max\left\{\delta_k + \frac{c_0q_k}{\omega},\, q_k + \omega\right\} \,.$$

Since the limit superior of the maximum is ω, and one may choose ω arbitrarily close to δ_*, we have shown that the sequences $\{\eta_k\}_k$ and $\{\mu_k\}_k$ both have a linear R-factor no greater than δ_*. The last assertion follows directly from the definition of η_k in (16). ▮

This result applies to all standard classical secant methods and suggests that the rate at which the derivatives x'_k converge is the same whether or not the Jacobian updating procedure is differentiated. This conclusion is valid only if the globalization strategy eventually becomes inactive, so that all later steps are of unit length. On the one hand, this means that the fully differentiated, or black box, approach is reasonably safe. On the other hand, it appears that implicitly defined derivatives can be obtained at a much reduced cost by deactivating the P_k, that is, by treating them as constants as in the simplified updating scheme. Also, the theoretical possibility that the P'_k generated in the fully differentiated update may grow unbounded is numerically worrisome, as it may lead to exponent overflows.

4 NUMERICAL RESULTS

Our limited numerical experiments confirm and illustrate our theoretical results. In Section 4.1, we test the predictions of Propositions 1 and 2 for the DFP update, and we consider the generation of higher-order derivatives. We compare the simplified and the fully differentiated derivative recurrences, both without and with line searches. We find general agreement with the theory, but we cannot choose between the simplified and the full derivative recurrences in general on the basis of these experiments.

In Section 4.2, we consider the effect on a class of parameter identification problems of using automatic differentiation of an implicitly defined function to compute the derivative information needed by the optimization algorithm. The fully differentiated Newton's method using the more accurate derivative values computed by automatic differentiation typically required fewer iterations to find a satisfactory solution than the corresponding finite difference algorithm, but the finite difference version required less total CPU time. Similar results are given for the fully differentiated Broyden's method. Finally, we give some numerical results for a (differentiated) simplified Newton's method using slightly modified stopping criteria which indicates that the simplified approach is quite promising.

4.1 *THE DFP UPDATE AND HIGHER-ORDER DERIVATIVE RECURRENCES*

Our limited numerical experience confirms the theoretical results. We found only a moderate growth of the P'_k for our test case, the Davidon-Fletcher-Powell (DFP) secant method. However, there is clear evidence that the convergence of the first derivatives x'_k lags significantly behind the convergence of the iterates x_k themselves. This phenomenon is much more pronounced for secant methods than for Newton's method, where the d-th derivative can be shown to lag roughly d steps behind the functional iterate. We have also propagated higher derivatives for secant methods and found that they converge in a staggered fashion and at about the same rate whether or not P_k is deactivated.

Our numerical experiments were conducted on the test function

$$F(x,t) \equiv \nabla_x f(x,t) \quad \text{with} \quad f(x,t) \equiv \frac{1}{2}\left(x^T H x + t\|x\|^4\right) ,$$

where $H = [1/(i+j-1)]$ is the Hilbert matrix of order n, and $\|x\|$ denotes the Euclidean norm. Locally, the minimizers of f are characterized as roots of the stationarity conditions $F = 0$, so that minimization methods behave eventually like equation solvers. In the general nonlinear equations case, the progress towards the solution is usually gauged in terms of some norm of the residual vector F. Often a monotonic decrease of such a merit function is enforced by a suitable line-search or trust-region strategy [13]. In the optimization case, one may use the objective function f itself, which we have utilized for a line-search consisting of a single parabolic interpolation. This simple strategy works here because the objective function is convex and very smooth.

Since the unique solution $x_* = 0$ is independent of the parameter t, all derivatives $x'_*, x''_*, \ldots x_*^{(j)}$ must also vanish, a situation that makes monitoring their errors exceedingly simple. The approximate inverse Hessian was initialized as $P_0 = \text{diag}(i)_{i=1,\ldots,n}$, which is somewhat "smaller" than the exact inverse H^{-1}. Consequently, the inverse form of the DFP update takes a very long time before P_k and the resulting steps $s_k = -P_k F(x_k,t)$ become large enough to achieve superlinear convergence. The starting point was always the vector of ones $x_0 = e$, and the parameter was set to $t = 1$.

The simplified iteration depicted in Figure 1 proceeds rather slowly until the Frobenius norm of the error matrix $\tilde{D}_k = I - P_k H$ drops below 1 at about the 25-th step. Hence, our theoretical results apply at most for the last five iterations. Note that $\tilde{D}_k$ is not exactly equal to D_k since we have neglected the nonquadratic term. Over the whole range the iterates x_k, their "derivatives" x'_k and the corresponding residuals $F(x_k,t)$ and $F'(x_k,t,x'_k)$ converge more or less monotonically at about the same rate. Since the iterates themselves converge so slowly, the derivatives do not noticeably lag behind. The situation is not radically different when the iteration is fully differentiated as depicted in Figure 2. However, as one can see from the top line, the preconditioner derivative P'_k grows to a Frobenius norm in the hundreds before it finally begins to decline. As a result, the first derivative of iterates and residuals seems to behave a little bit more erratically in the intermediate stage of the iteration. While we have not made a timing comparison to see how much overhead the differentiation of the preconditioner entails, it would seem so far that there is no reward for incurring that extra cost. On the other hand, if the identification and "deactivation" of P_k appears to require a significant recoding effort, one may also just differentiate the whole iteration. In both Figures 1 and 2, the residual derivative $\|F'(x_k,t)\|$ is a fairly reliable indicator of the actual error $\|x'_k - x'_*\|$.

In order to speed up the convergence, let us introduce a line search in the form of exactly one parabolic interpolation of the objective f per step. The step multiplier was adjusted even when it was close to the first trial value one, which is known to happen during the final convergence to the solution. The resulting iteration with an increased dimension $n = 3$ and the simplified derivative recurrence is depicted in Figure 3. The convergence is now much more rapid throughout and accelerates again when the Frobenius norm of the discrepancy $\tilde{D}_k$ drops below one, which happens at about the eleventh iteration. Curiously, at exactly that stage, the derivatives of iterates and the residuals appear to deteriorate significantly before they begin to converge with a noticeable lag behind the iterates and their residuals. As we can see in

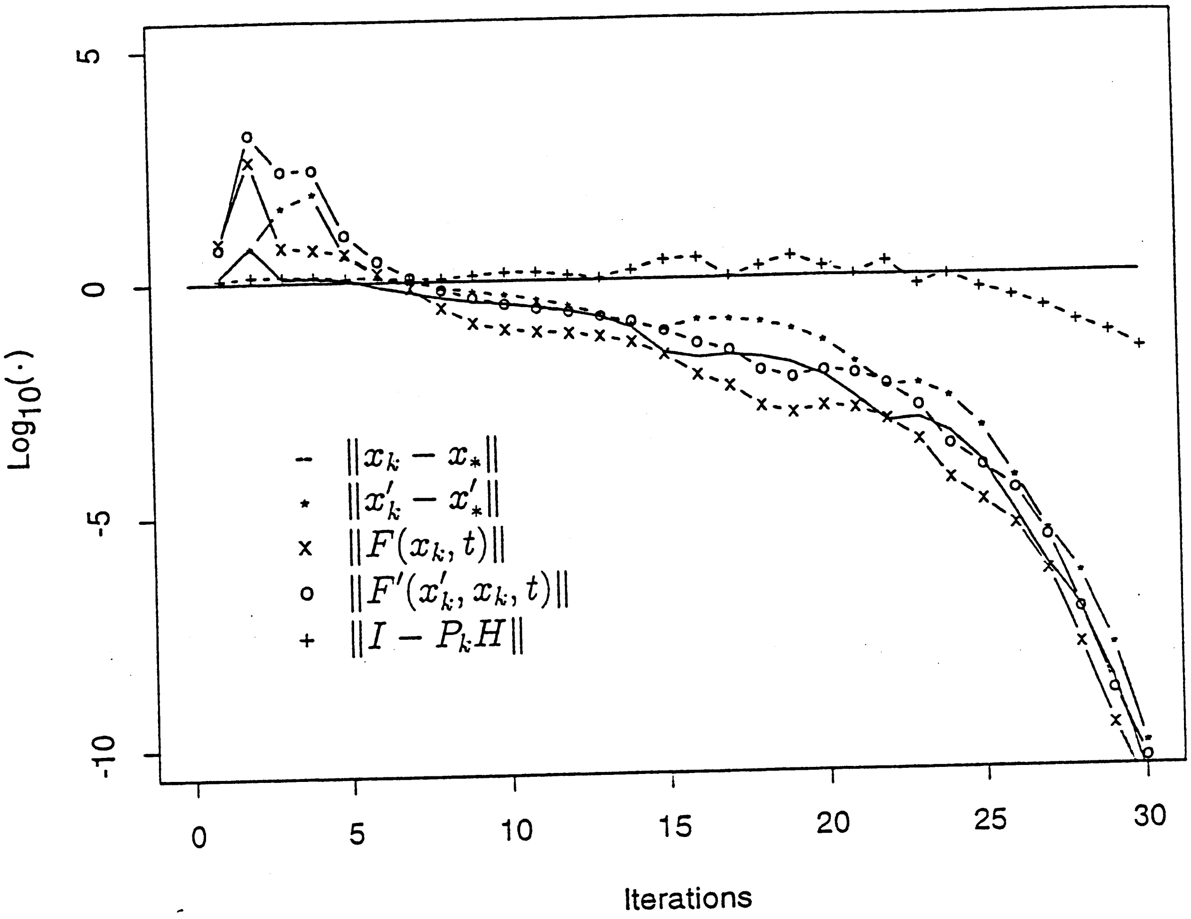

Figure 1: Simplified Derivative Recurrence, no line-search ($n = 2$)

Figure 4, the associated fully differentiated recurrence does not generate that hump and achieves faster derivative convergence. However, this time P'_k actually show signs of blowing up as its Frobenius norm almost reaches a thousand. Obviously, our very limited numerical experiments do not allow us to draw any general conclusions regarding the relative merits of the simplified and fully differentiated iteration.

Finally, let us consider the computation of second and higher derivatives. For the simplified recurrence, where the P_k are deactivated, the following informal argument establishes the convergence of the higher derivatives $x^{(j)} \equiv d^j x(t)/dt^j$. Differentiating equation (6) $j < m$ times with respect to t, we obtain the following linear system for the $(j+1)$-st derivative from Leibnitz's rule:

$$P_k\, F_x(x(t),t)\, x^{(j+1)} = -P_k \left(\frac{\partial^j}{\partial t^j} [F_t(x(t),t)] + \sum_{i=1}^{j} \binom{j}{i} \frac{\partial^{j-i}}{\partial t^{j-i}} [F_x(x(t),t)]\, x^{(i)}(t) \right) . \tag{25}$$

Here we have assumed that $F(x,t)$ is m times jointly Lipschitz-continuously differentiable. Replacing the $x^{(i)}(t)$ by approximations $\bar{x}_k^{(i)}$ for $i = 0, 1, \ldots, j$, one may interpret the right-hand side as a vector function

$$-P_k R_k^{(j)} \equiv -P_k R_k^{(j)} \left(t, x_k, \bar{x}'_k, \ldots, \bar{x}_k^{(j)} \right) .$$

While this may seem a very messy expression, the residual vectors

$$F_x(x_k(t),t)\, x_k^{(i+1)} + R_k(i), \quad \text{for } i = 0, 1, \ldots, j$$

can be evaluated simultaneously for any given t and $(\bar{x}_k^{(i)})_{i=0,1,\ldots,j+1}$ by one forward sweep of automatic differentiation [8]. The complexity of this Taylor series propagation is $\mathcal{O}(j^2)$ times that of one function

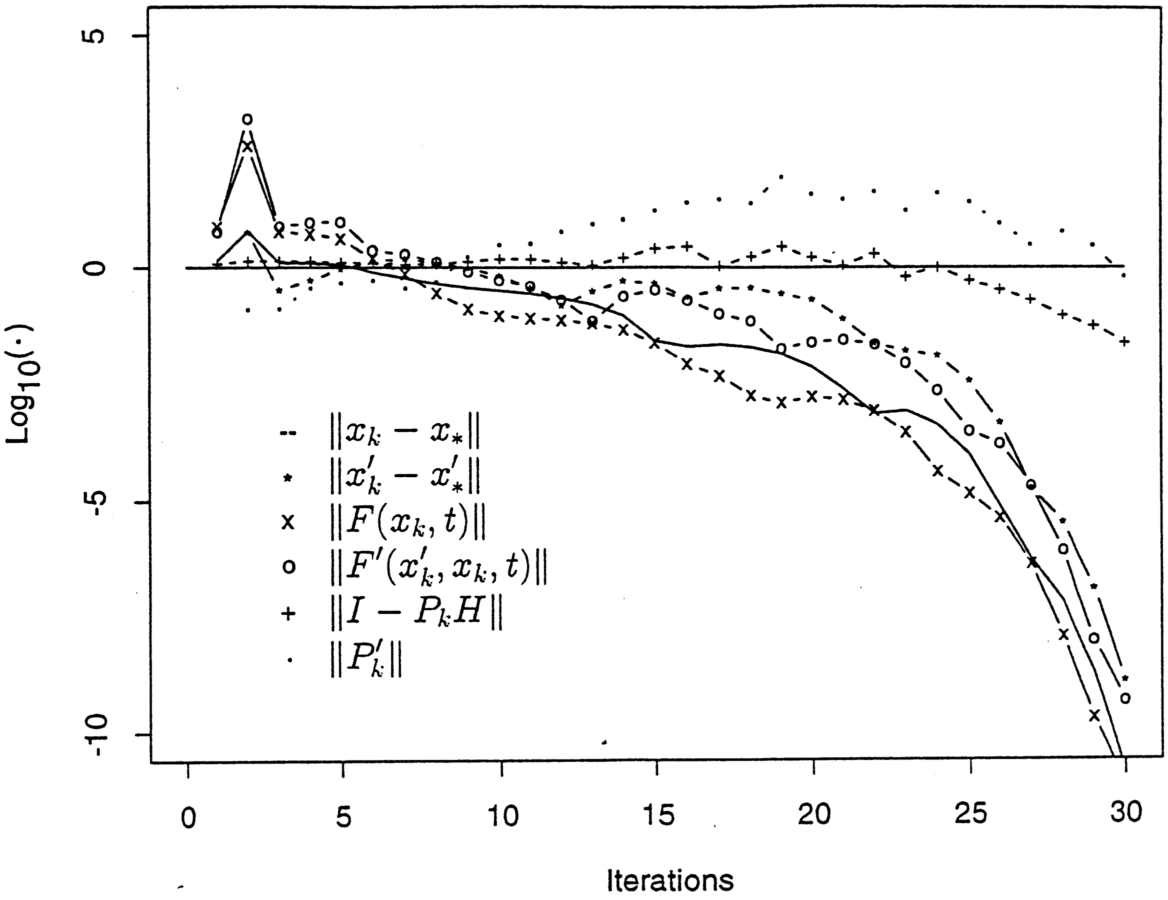

Figure 2: Full Derivative Recurrence, no line-search ($n = 2$)

evaluation $F(x,t)$ if ordinary polynomial arithmetic is used. This asymptotic complexity bound can be reduced to $\mathcal{O}(j \log j)$ through the use of the fast Fourier transform, but that is likely to pay off only when j is significantly larger than 10. As a generalization of (7), one may now iterate for $j = 0, 1, \ldots, m-1$ and $k = 0, 1, \ldots$

$$\tilde{x}^{(j+1)}_{k+1} = \tilde{x}^{(j+1)}_{k} - P_k \left[F_x(x_k(t), t)\, \tilde{x}^{(j+1)}_{k} + R^{(j)}_{k}\right] .$$

This family of linear recurrences is again of form (7) with the same leading linear term. By induction, one sees that if all $\tilde{x}^{(i)}_k$ for $i < j$ converge to the correct values $x^{(i)}_*$, then the $R^{(j)}_k$ converge to the right-hand side of (25), and the $\tilde{x}^{(j+1)}_k$ can converge only to the unique fixed point $x^{(j+1)}_*$ of its recurrence. The linear R-factor is again at least δ_*, but the higher derivatives tend to converge in a staggered fashion.

To demonstrate this, let us look at the second, third, fourth and fifth derivatives of the iterates x_k generated by the simplified recurrence for $n = 2$ without line search and the full recurrence for $n = 3$ with line-search. These two cases are depicted in Figure 5 and 6, respectively. The most striking feature of both graphs is that the derivatives are staggered very nicely behind the iterates themselves. The convergence behavior is quite smooth and looks ultimately superlinear. This seems to suggest that the errors D_k tend to zero, so that our theorems assert R-superlinear convergence of the derivatives as $\delta_* = 0$.

The fully differentiated recurrence without line-search for $n = 2$ and the simplified recurrence with line search for $n = 3$ exhibit about the same average speed, but the convergence looks significantly rougher. We have not included the corresponding graphs in the paper and are still investigating the reasons for this less desirable behavior. In fact, we have currently no explanation why the full recurrence appears to work better on the iteration with line-search. In a practical code, the line-search becomes eventually inactive, so that one might expect the simplified recurrence to yield more accurate derivatives at a lower cost.

The numerical results reported in this section were obtained in double precision on a SPARCstation 2

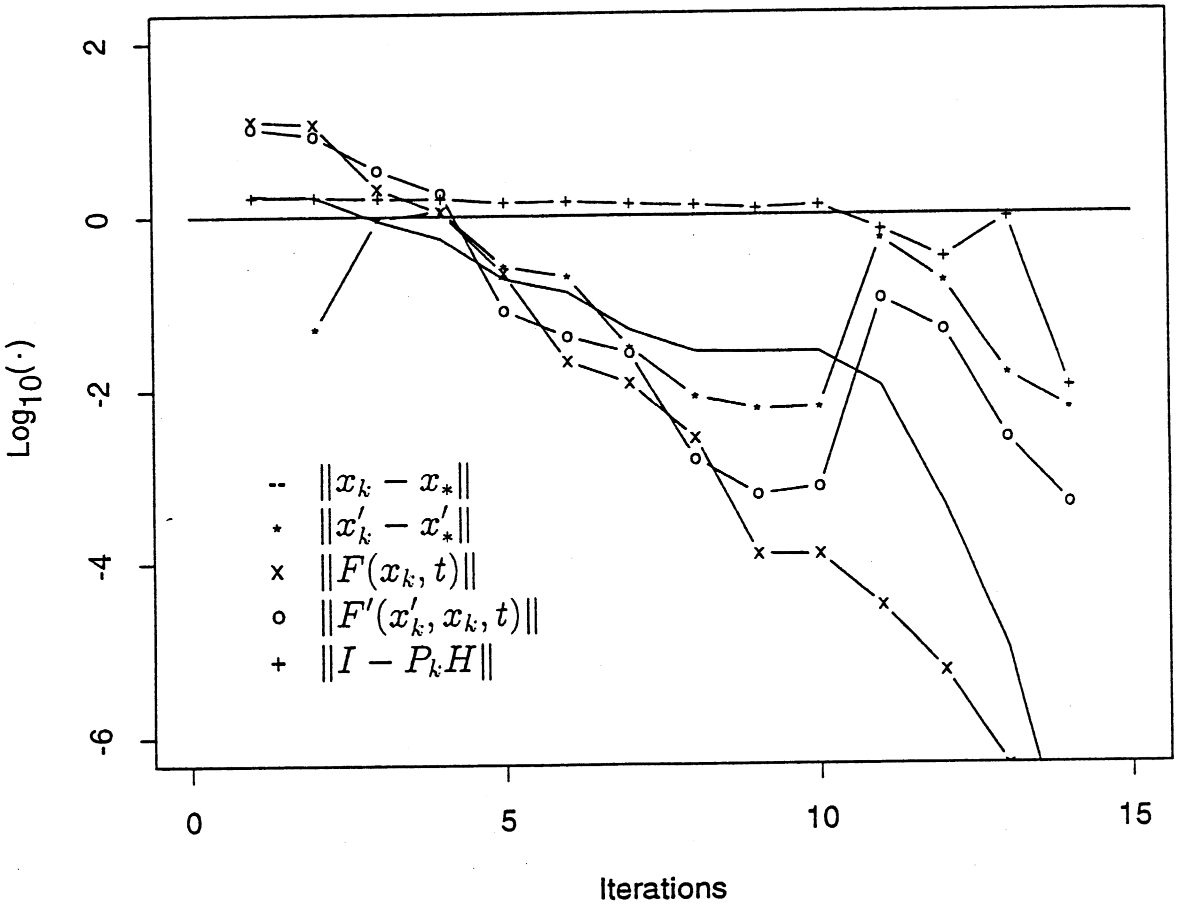

Figure 3: Simplified Derivative Recurrence, parabolic line-search ($n = 3$)

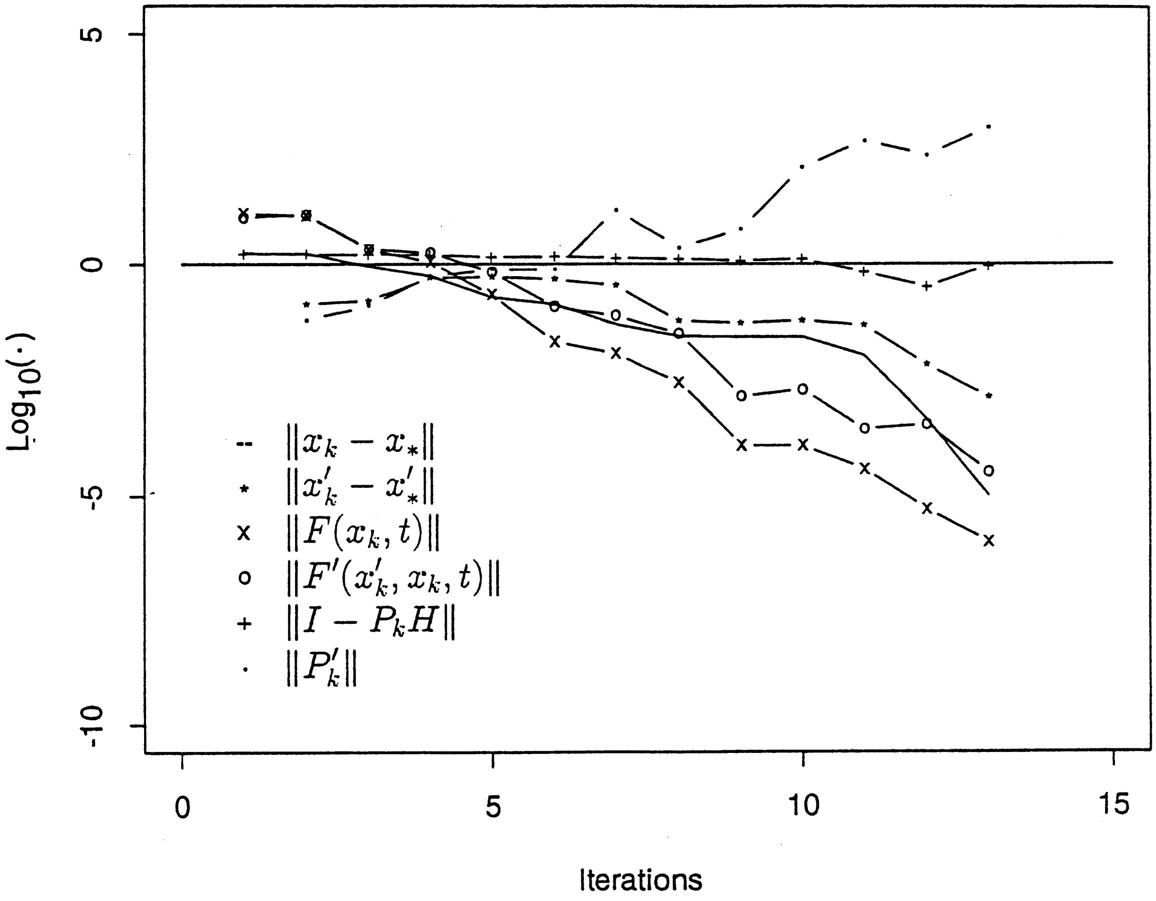

Figure 4: Full Derivative Recurrence, parabolic line-search ($n = 3$)

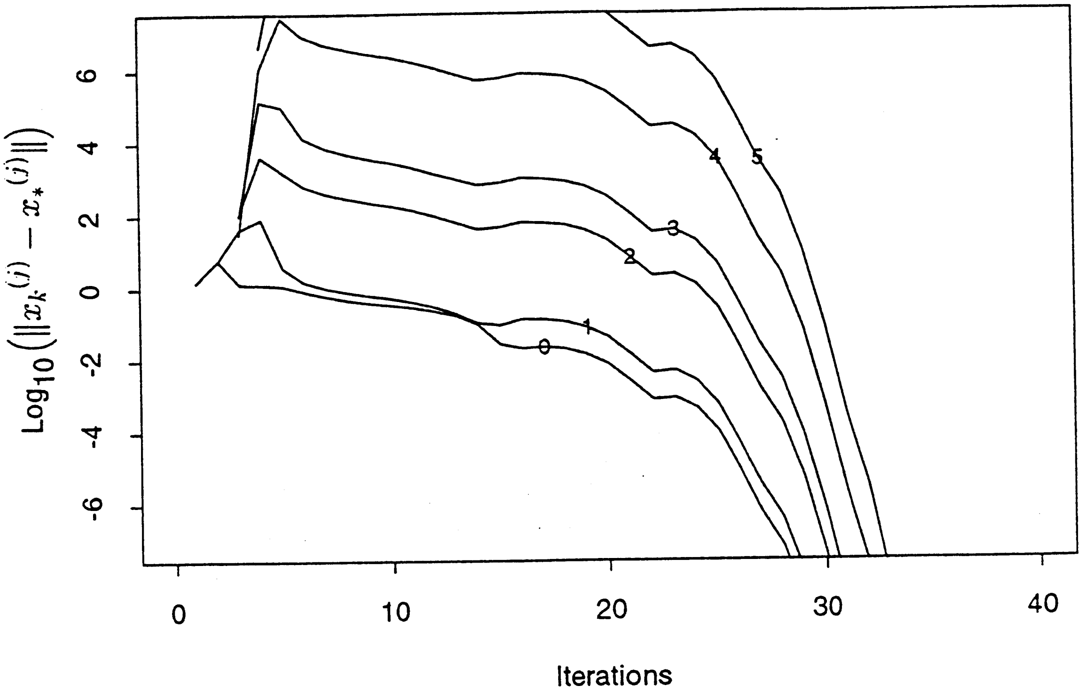

Figure 5: Simplified Derivative Recurrence for first five derivatives ($n = 2$)

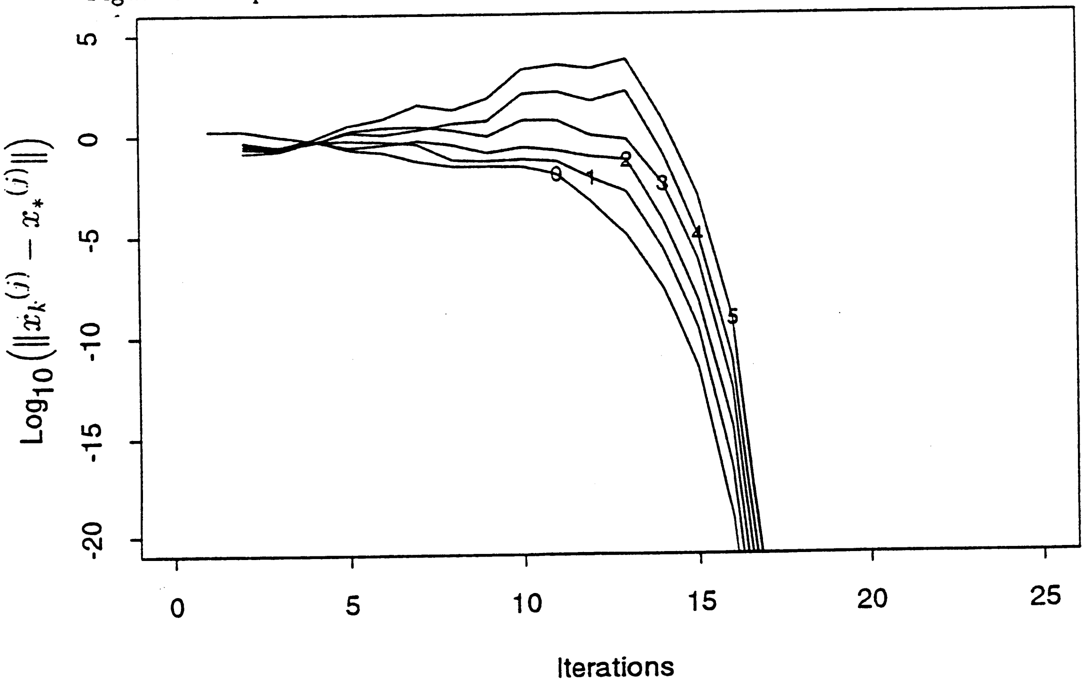

Figure 6: Full Derivative Recurrence for first five derivatives ($n = 3$)

using the automatic differentiation package ADOL-C described in [17].

4.2 APPLICATION TO PARAMETER IDENTIFICATION PROBLEMS

In this section, we study the effect of using the automatic differentiation of an implicitly defined function to generate the first-order derivative information needed at each iteration of an optimization algorithm. In particular, we consider the impact of both the fully differentiated and simplified approaches on the solution of parameter identification problems.

Given a parameterized system of ordinary differential equations

$$y' = g(\tau, y; p) \tag{26}$$

and a set of data points $(\tau_j, \hat{y}_j)$, the parameter identification problem is to find values of the parameters p_* to minimize the discrepancy between the solution $y(\tau; p_*)$ of the ODE model (26) and the data points. The data points are measurements of the solution trajectory $y(\tau_j, p_*)$ at various *times* τ_j. Given a particular vector of parameters p_i, the elements of the residual vector $R(p_i)$ are the discrepancies between the solution of the model $y' = g(\tau, y; p_i)$ and the data points. Then, the optimization problem to be solved can be loosely stated as

$$\begin{aligned} \underset{p}{\text{minimize}} \quad & \frac{1}{2} R(p)^T R(p) \qquad (27) \\ \text{subject to} \quad & y' = g(\tau, y; p) \,. \end{aligned}$$

The initial values for the ODE (26) can either be treated as fixed constants, $y(\tau_0) \equiv \hat{y}_0$, or as additional unknown parameters. We apologize for the change in notation; in this section the parameters are denoted by p instead of t since t usually represents time in the context of differential equations.

Using orthogonal collocation at the Gauss points, the differential equation (26) is discretized so that the solution trajectory is approximated by a piecewise polynomial with coefficients x. Certain conditions must be imposed to ensure that the approximating polynomial adequately represents the solution to the differential equation, and these collocation and continuity conditions yield a nonlinear system of equations $F(x; p) = 0$. For each parameter vector p_i, this nonlinear system must be solved to obtain the coefficients $x(p_i)$ of the polynomial approximation to the solution trajectory $y(\tau; p_i)$. A more detailed description of the problem formulation and the collocation scheme can be found in [12].

Thus, using collocation, we can replace the ODE in (27) with a nonlinear system of equations. The "black box" formulation of the parameter identification problem can then be written as

$$\underset{p}{\text{minimize}} \quad \frac{1}{2} R(p, x(p))^T R(p, x(p)) \,, \tag{28}$$

where $x(p)$ solves

$$F(x; p) \quad = \quad 0 \tag{29}$$

for a fixed p. To solve the resulting optimization problem (28), we will use the nonlinear least-squares package NL2SOL [11]. NL2SOL is an implementation of an augmented Gauss-Newton trust region method [10] that exploits the structure of the nonlinear least-squares problem.

For each parameter vector p_i, (where i denotes the i-th optimization iteration), the evaluation of the residuals $R(p_i, x(p_i))$ requires the solution of the nonlinear system of equations (29) for the implicit variables $x(p_i)$. We consider both Newton's method and Broyden's method for the solution of this nonlinear system; in particular, the implementations found in MINPACK [18], (LMDER and HYBRJ). Thus, given p_i and an initial guess for $x(p_i)$, the nonlinear equation solver generates a sequence of iterates

$$x_0(p_i),\ x_1(p_i),\ x_2(p_i), \ldots, x_*(p_i) \equiv x(p_i)$$

until an acceptable solution is found. At the first optimization iteration, the initial guess $x_0(p_0)$ is generated by linear interpolation of the data points, but at each subsequent iteration, the starting point is chosen to be the solution of (29) at the previous iterate, i.e., $x_0(p_i) = x_*(p_{i-1})$. LMDER uses analytical derivatives, $F_x(x,p)$, while HYBRJ starts with the analytical $F_x(x,p)$ and then uses Broyden's method to update the factorization at subsequent iterations. Both algorithms use a trust region globalization strategy.

At each optimization iteration, we must estimate the necessary first-order derivative information, $J(p, x(p))$, where J denotes the Jacobian of the residuals $R(p, x(p))$, i.e., $J_{j\cdot} = \partial R_j(p, x(p))/\partial p$. The Jacobian can be computed by finite differences, or it can be constructed from values of $dx(p_i)/dp$. We will apply automatic differentiation to the nonlinear equation solver to obtain an iterative procedure for computing $dx(p_i)/dp$. Applying the automatic differentiation tool ADIFOR [1] to LMDER and HYBRJ, which are both written in Fortran, yields a fully differentiated Newton's method and a fully differentiated Broyden's method. Given the current optimization iterate p_i and initial guesses $x_0(p_i)$ and $dx_0(p_i)/dp$, the resulting (Fortran) code generates a sequence of derivatives

$$\frac{dx_0(p_i)}{dp}, \frac{dx_1(p_i)}{dp}, \frac{dx_2(p_i)}{dp}, \ldots, \frac{dx_*(p_i)}{dp} \equiv \frac{dx(p_i)}{dp},$$

in addition to the original sequence of iterates $\{x_k(p_i)\}$.

We used six parameter identification problems from [12] to evaluate the effectiveness of the algorithms. We tested two formulations of each problem, and Table 4 identifies the problems using the convention that "problem 1" refers to problem 1 with fixed initial conditions while "problem 1i" refers to problem 1 with the initial conditions treated as additional, unknown parameters. Table 4 also indicates the size of each problem, both the dimension of the parameter vector p and the size of the nonlinear system $F(x,p) = 0$. Each ODE model (26) was discretized using a collocation scheme with ten uniformly spaced subintervals and a polynomial approximation of degree three on each subinterval. While this discretization is adequate to solve all of the problems, a more efficient discretization could be chosen for each individual problem. In addition to the standard starting point, (which can be found in [12]), problems 1a and 1ai used $p_0 = (15, 20)^T$, problems 4a and 4ai used $p_0 = (1 \times 10^{-5}, 1 \times 10^{-5})^T$, and problems 4b and 4bi used $p_0 = (1 \times 10^{-4}, 1 \times 10^{-4})^T$, resulting in sixteen different test cases.

Using Newton's method (LMDER) to solve (29) for the implicit variables $x(p_i)$, we first compared the effect of a forward difference Jacobian with a Jacobian constructed from $dx_*(p_i)/dp$ produced by the fully differentiated Newton's method. We tested the fully differentiated code as if it were a "black box" in the sense that we did not modify the code produced by the automatic differentiation tool to take into account our knowledge that we had differentiated Newton's method. Specifically, the stopping criteria in the fully differentiated code are the original stopping criteria used to judge the convergence of $x_k(p_i) \rightarrow x_*(p_i)$ and have not been modified to take into consideration the convergence of $dx_k(p_i)/dp \rightarrow dx_*(p_i)/dp$.

The top section of Table 2 gives the number of optimization iterations, the number of residual calculations, (i.e., the number of objective function evaluations), and the approximate running time of these algorithms for each problem. As the numbers show, the fully differentiated Newton's method performed better than the finite difference code in terms of the number of optimization iterations required to reach an acceptable solution. In fact, the fully differentiated code saved ten optimization iterations overall.

However, the total running time for the fully differentiated Newton's method was slightly more than twice that of the finite difference code. With the exception of problem 4bi, the individual times show that the fully differentiated algorithm required roughly two to five times more CPU time than the finite difference method. It should be noted that the performance of the fully differentiated code suffers since the small number of parameters for these problems magnifies the loop overhead for each of the intermediate gradient computations.

Some of the running time differences are due to the *iterative* nature of the residual calculation. One would expect that each column of the finite difference Jacobian would require approximately the same amount of work as the corresponding residual computation, but this was often not the case. For example, consider problem 3, which has three unknown parameters, (i.e., the Jacobian has three columns.) Both methods required six optimization iterations to solve the problem, and they also used the same number of iterations to compute each $R(p_i, x(p_i))$. The number of LMDER iterations it took to solve $F(x, p_i) = 0$ for each residual calculation is given in Table 1. Overall, 35 iterations were performed by the nonlinear equation solver. Thus, if each column of the finite difference Jacobian cost as much as the corresponding residual calculation, then we would predict that it would take $35 * 3 = 105$ LMDER iterations to compute all of the necessary finite difference Jacobians. However, instead of the predicted 105 iterations, Table 1 shows that only 63 iterations of LMDER were required. So, due to the iterative nature of the residual calculation, the finite difference algorithm was substantially cheaper than it would have been if the cost of each column of the Jacobian were as expensive as the corresponding residual calculation.

Also included in Table 2 is an estimate of the relative error in the computed values of $dx_*(p_i)/dp$. Unlike the example in the previous section, we do not have an analytical expression for either $x_*(p_i)$ or $dx_*(p_i)/dp$. Therefore, we use $dx(p_i)/dp$ to denote the numerical solution determined by the differentiated nonlinear equation solvers. From the implicit function theorem, we know $dx_*(p_i)/dp$ satisfies the linear equation (4). Thus, for each Jacobian calculation, we can estimate the relative error in the computed value of $dx(p_i)/dp$ as

$$\text{Relative Error } (dx(p_i)/dp) \approx \qquad (30)$$

$$\frac{||F_x(x_*(p_i),p_i)\cdot dx(p_i)/dp + F_p(x_*(p_i),p_i)||\cdot \text{Cond}\,(F_x(x_*(p_i),p_i))}{||F_x(x_*(p_i),p_i)||\cdot||dx(p_i)/dp||}.$$

We then have an estimate of the relative error in $dx(p_i)/dp$ for each optimization iteration i. The numbers reported in Table 2 in the column labelled "Rel Error" are the maximum relative error estimates over all of the optimization iterations for each problem.

Although the fully differentiated Newton's method (LMDER) used the stopping criteria for the underlying $x_k(p_i) \longrightarrow x_*(p_i)$ convergence, the relative error estimates for $dx(p)/dp$ are on the order of machine precision. This indicates that the convergence of derivative did not noticeably lag behind the convergence of the x_k iterates for Newton's method. This is not surprising, for the work of Gilbert [15] shows that theoretically Newton's method is "special" in that the derivative convergence should only lag one iteration behind the iterate convergence. On the other hand, based on the step size, we would estimate a relative error of around 10^{-8} for the finite difference Jacobian. Over all of the test cases, the extra accuracy in the Jacobian obtained from the fully differentiated Newton's method (LMDER) saved ten optimization iterations and six Jacobian calculations over the finite difference method. However, the finite difference method required slightly less than half of the time needed by the fully differentiated Newton approach.

In a similar manner, the fully differentiated Broyden's method uses Broyden's method (HYBRJ) to solve the nonlinear system (29) and a fully differentiated Broyden's (HYBRJ) method to estimate $dx_*(p_i)/dp$. Again, the algorithm was tested as a "black box," and the fully differentiated code used only the stopping criteria for the $x_k(p_i) \longrightarrow x_*(p_i)$ iteration.

As one would expect, the fully differentiated Broyden's method required slightly more optimization iterations but slightly less CPU time than the fully differentiated Newton's method. These results are shown in Table 2. Each Broyden iteration is cheaper than a Newton iteration, $O(dim(x)^2)$ versus $O(dim(x)^3)$, and so, each fully differentiated Broyden iteration is cheaper than a fully differentiated Newton iteration. However, the relative error (30) is significantly larger for some of the problems then the relative error estimate generated by the fully differentiated Newton's method.

Figure 7 shows the convergence history of the initial Jacobian, $J(p_0, x(p_0))$, computation for problem 4b using the fully differentiated Broyden's method. The differentiated HYBRJ required 50 iterations to find an acceptable solution $x_*(p_0)$ and thus $dx(p_0)/dp$. Since neither $x_*(p)$ or $dx(p)/dp$ are zero, we used relative

Optimization iteration	0	1	2	3	4	5	6	Total
Number of iterations to compute $x(p_i)$	7	7	6	5	4	3	3	35
Number of iterations for each column of J	3,3,3	3,3,3	3,3,3	3.3,3	3,3,3	3,3,3	3,3,3	63

Table 1: The number of LMDER iterations required to compute $x(p_i)$ and the number of LMDER iterations used to compute each column of the finite difference Jacobian $J(p_i, x(p_i))$ at each optimization iteration for problem 3.

	Finite Difference Newton		Fully Differentiated Newton		
Problem	# Iter (# Res)	Time (sec)	# Iter (# Res)	Time (sec)	Rel Error
1	9 (13)	14.0	9 (13)	34.5	5×10^{-16}
1i	9 (13)	22.3	9 (13)	45.5	3×10^{-16}
1a	7 (9)	13.8	7 (9)	29.9	4×10^{-16}
1ai	9 (11)	25.2	9 (11)	53.0	5×10^{-16}
2	9 (11)	21.4	7 (9)	42.4	2×10^{-16}
2i	8 (10)	25.4	7 (9)	53.3	3×10^{-16}
3	6 (7)	15.8	6 (7)	61.6	2×10^{-16}
3i	6 (7)	20.5	6 (7)	78.3	3×10^{-16}
4	8 (11)	3.3	8 (11)	10.0	5×10^{-16}
4i	8 (11)	4.0	7 (10)	10.7	5×10^{-16}
4a	7 (12)	3.4	7 (12)	9.4	3×10^{-15}
4ai	6 (10)	3.4	6 (10)	9.1	6×10^{-16}
4b	13 (19)	14.0	12 (23)	23.3	1×10^{-15}
4bi	12 (17)	49.0	10 (15)	24.3	1×10^{-15}
5	9 (10)	26.5	8 (9)	127.8	6×10^{-16}
5i	10 (13)	38.6	8 (11)	184.1	3×10^{-15}
6	6 (7)	264.6	6 (7)	604.5	3×10^{-16}
6i	6 (7)	449.2	6 (7)	926.1	3×10^{-16}
Totals	148 (198)	1014.4	138 (193)	2327.8	

	Fully Differentiated Broyden			Simplified Newton		
Problem	# Iter (# Res)	Time (sec)	Rel Error	# Iter (# Res)	Time (sec)	Rel Error
1	9 (13)	30.7	4×10^{-15}	9 (13)	17.2	5×10^{-16}
1i	9 (13)	39.9	4×10^{-16}	9 (13)	15.4	3×10^{-16}
1a	9 (14)	36.9	3×10^{-15}	10 (12)	18.0	2×10^{-11}
1ai	9 (11)	46.9	8×10^{-16}	9 (11)	19.6	4×10^{-16}
2	9 (11)	43.8	9×10^{-17}	10 (12)	17.0	3×10^{-12}
2i	8 (10)	55.3	4×10^{-17}	8 (10)	16.3	3×10^{-16}
3	8 (9)	44.7	8×10^{-11}	6 (7)	20.5	2×10^{-12}
3i	7 (8)	51.4	4×10^{-11}	6 (7)	19.8	4×10^{-12}
4	8 (11)	7.8	1×10^{-11}	8 (11)	5.2	1×10^{-12}
4i	8 (11)	9.0	1×10^{-11}	8 (11)	5.1	2×10^{-13}
4a	7 (12)	7.4	1×10^{-11}	7 (12)	5.1	1×10^{-13}
4ai	6 (10)	7.6	2×10^{-11}	6 (10)	4.3	6×10^{-13}
4b	12 (23)	24.5	1×10^{-6}	12 (23)	15.2	1×10^{-12}
4bi	11 (16)	20.7	2×10^{-6}	11 (16)	16.0	5×10^{-12}
5	11 (12)	83.8	1×10^{-9}	9 (10)	42.5	2×10^{-13}
5i	10 (13)	100.7	7×10^{-9}	10 (13)	50.8	3×10^{-12}
6	6 (7)	579.2	2×10^{-15}	6 (7)	138.8	3×10^{-16}
6i	6 (7)	888.3	1×10^{-15}	6 (7)	139.1	3×10^{-16}
Totals	153 (211)	2078.6		150 (205)	565.9	

Table 2: The number of optimization iterations, (number of residual, $R(p, x(p))$, calculations) and the time in seconds required to solve each of the parameter identification problems, and a relative error estimate for dx/dp.

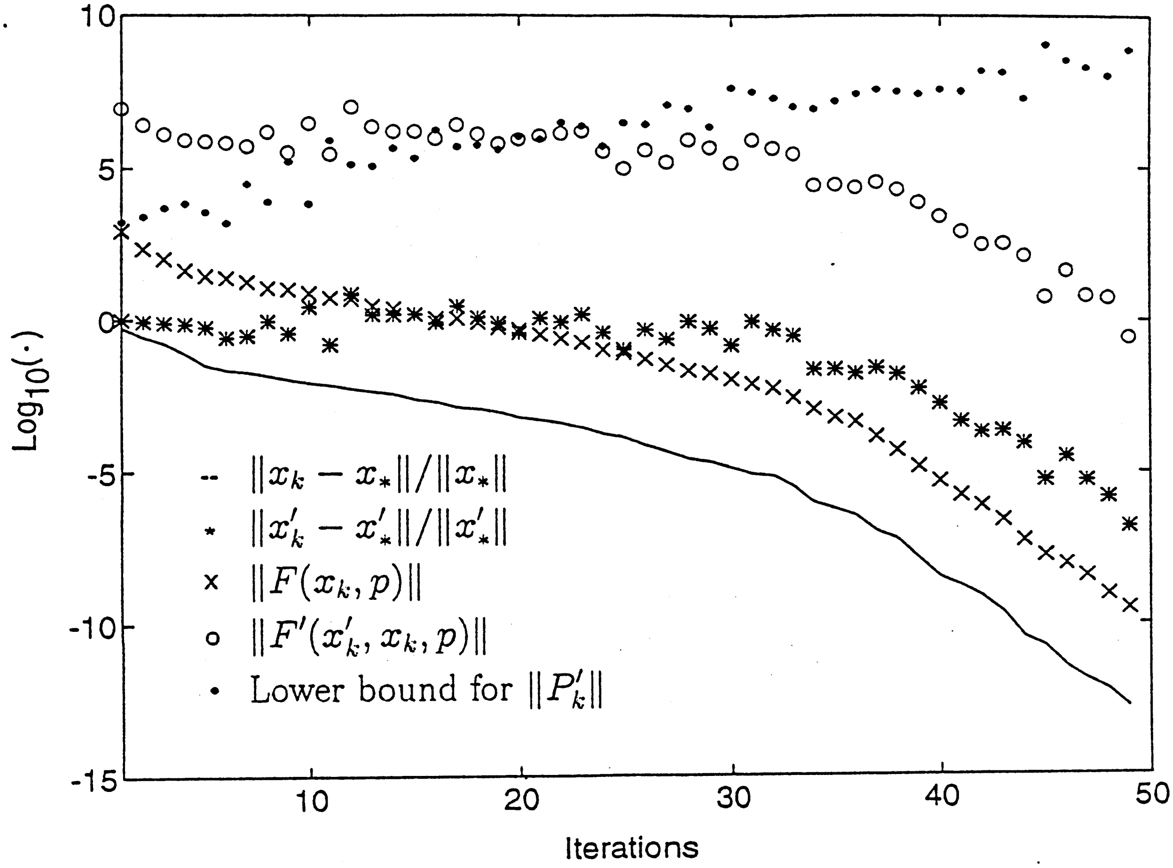

Figure 7: Fully differentiated Broyden's method for the first Jacobian calculation in problem 4b.

error estimates of the form

$$\frac{\|x_k(p) - x_*(p)\|_2}{\|x_*(p)\|_2} \quad \text{and} \quad \frac{\|x'_k(p) - x'(p)\|_F}{\|x'(p)\|_F},$$

where x'_k a is convenient shorthand for $dx_k(p)/dp$. (All of the plots in Figure 7 are $\log_{10}(\cdot)$ of the respective quantities.) Also included in Figure 7 is the norm of the nonlinear system, $\|F(x_k;p)\|_2$, and the graph shows that it closely follows the decrease in the relative error in x_k. Similarly, the plot of the norm of the total derivative, $\|F_x(x_k, p) \cdot x'_k + F_p(x_k, p)\|$ matches the trends in the relative error in dx_k/dp, but it is substantially larger.

We derived a lower bound for the derivative of the inverse of the Broyden approximation, P'_k. Since the implementation in HYBRJ uses the factored form of Broyden's method, it is difficult to obtain P'_k itself. The fully differentiated derivative recurrence (10) can be written as

$$x'_{k+1} - x'_k - P_k[F_x(x_k, p) \cdot x'_k + F_p(x_k, p)] = P'_k \cdot F(x_k, p) .$$

Taking the norm of both sides and applying the triangle inequality yields

$$\|x'_{k+1} - x'_k - P_k[F_x(x_k, p) \cdot x'_k + F_p(x_k, p)]\| \le \|P'_k\| \cdot \|F(x_k, p)\| .$$

This gives us a lower bound on P'_k of the form

$$\frac{\|x'_{k+1} - x'_k - P_k[F_x(x_k, p) \cdot x'_k + F_p(x_k, p)]\|}{\|F(x_k, p)\|} \le \|P'_k\| , \tag{31}$$

and we can compute the qualities on the left-hand side of (31) using the machinery already in HYBRJ to apply P_k to $F_x(x_k, p) \cdot x'_k + F_p(x_k, p)$. Since this is a fairly expensive computation, none of the calculations needed for Figure 7 are included in the timings given in Table 2. Note that the derivative recurrence (10), where P_k is the inverse of the Broyden approximation to $F_x(x_k, p)$, does not necessarily hold if the trust region globalization strategy modifies the direction and length of the step. However, full "Broyden" steps were always taken in this particular example. The graph of the lower bound for $||P_k'||$ given by (31) is shown in Figure 7. The plot indicates that the lower bound on $||P_k'||$ is *increasing* as $x_k \rightarrow x_*$, and this supports the hypothesis that the differentiation of P_k can be unstable. It should be noted that in this particular case, $||I - P_k F_x(x_k, p)|| > 1$, and this is most likely due to the fact that the starting guess p_0 is "far" from the solution p_*.

Finally, we obtained a (differentiated) simplified Newton's method (LMDER) from the fully differentiated Newton's method by deactivating the differentiation of P_k by hand. We tried using the simplified Newton's method with only the stopping criteria for the $x_k \rightarrow x_*$ iteration, but the results were not particularly successful on some of the problems. Thus, we introduced an additional stopping condition requiring that the relative error estimate, (30), must be less than a small constant. For these tests, we chose the constant to be $macheps^{(2/3)} \approx 4 \times 10^{-11}$. If the relative error estimate does not satisfy this stopping condition, we forced the simplified code to take additional simplified Newton steps until x_k' satisfied the new relative error test. The results for the (differentiated) simplified Newton's method are given in Table 2, and the relative error estimates range from 10^{-11} to 10^{-16}. The larger relative error estimates are probably an artifact of the constant in the new stopping condition, i.e., when the $x_k \rightarrow x_*$ iteration converged, $x'(p)$ just barely passed the relative error test, and no additional steps were taken. Over all of the test problems, an additional 40 (differentiated) simplified Newton steps were required to satisfy the relative error test, and the number of extra steps required by the simplified Newton's method to solve each optimization problem is given in Table 3. As this table shows, there is a definite correlation between the number of additional steps and the resulting maximum relative error in x'.

Returning to the results in Table 2, overall, the simplified Newton's method required slightly fewer optimization iterations and residual calculations than the fully differentiated Broyden's method but slightly more than either the finite difference method or the fully differentiated Newton's method. However, because the factorization of $F_x(x_k, p)$ is the dominant cost of each Newton iteration, deactivating the differentiation of P_k' makes the (differentiated) simplified Newton's method the fastest method we tested. Overall, it is about four times faster than the fully differentiated Newton's method and roughly twice as fast as the finite difference method.

Thus, we have demonstrated that the automatic differentiation tool ADIFOR can successfully differentiate the complex library routines LMDER and HYBRJ from MINPACK to provide derivatives of variables defined by implicit functions. The derivatives obtained directly from the fully differentiated Newton and Broyden codes provided sufficiently good approximations to the Jacobian, and using these Jacobians, the optimization code was able to solve this set of parameter identification problems. However, the fully differentiated codes were slower than the finite difference code. The results for the simplified (differentiated) Newton's method show that simplified, or deactivated approaches can work on practical problems, but currently, they may require some user (expert) intervention. However, the simplified approach shows the most promise for generating fast, accurate derivatives.

The numerical results reported in this section were obtained in double precision on a SPARCstation 2 using the automatic differentiation package ADIFOR described in [1].

5 CONVERGENCE RESULTS FOR MULTISTEP CONTRACTIONS

Unfortunately, many methods of great practical importance are not one-step contractive in the sense that most or all of the D_k have a spectral radius greater than or equal to one. For example, this is true for any iterative method that keeps some components of x_k fixed at each step, such as cyclic reduction or any form of alternating projections. In those cases, one would still hope that over a cycle of iterations, a significant contraction is achieved in the following sense.

Assumption 4 *The preconditioners P_k are chosen uniformly bounded so that*

$$||P_k|| + ||P_k^{-1}|| \leq c_0 < \infty \quad \text{for all } k, \tag{32}$$

Problem	Additional steps	Relative Error
1	6	5×10^{-16}
1i	2	3×10^{-16}
1a	2	2×10^{-11}
1ai	2	4×10^{-16}
2	4	3×10^{-12}
2i	3	3×10^{-16}
3	1	2×10^{-12}
3i	0	4×10^{-12}
4	1	1×10^{-12}
4i	1	2×10^{-13}
4a	1	1×10^{-13}
4ai	2	6×10^{-13}
4b	1	1×10^{-12}
4bi	1	5×10^{-12}
5	2	2×10^{-13}
5i	3	3×10^{-12}
6	4	3×10^{-16}
6i	4	3×10^{-16}

Table 3: The number of additional simplified (differentiated) Newton steps required to satisfy the relative error test over the solution of each test problem.

	Problem	$dim(p)$	$dim(x)$
1	First-order irreversible chain reaction	2	80
1i		4	78
2	First-order reversible chain reaction	4	80
2i		6	78
3	Catalytic cracking of gasoil	3	80
3i		5	78
4	Bellman's problem	2	40
4i		3	39
5	Barnes' problem	3	80
5i		5	78
6	Thermal isomerization of α-pinene	5	200
6i		10	195

Table 4: Parameter Identification Test Problems

and there exists an induced matrix norm and a cycle length $m > 0$ such that

$$\delta_m \equiv \overline{\lim}_j \, \|D_{j+m} \cdot D_{j+m-1} \cdots D_{j+2} \cdot D_{j+1}\|^{\frac{1}{m}} < 1 \, . \tag{33}$$

We will argue at the end of this section that any method for which this condition is not met is numerically unstable.

Proposition 3 *Under Assumptions 1 and 4, the iterations (1) and (7) converge with a linear R-factor no less than*

$$\delta_* = \inf_m \delta_m < 1$$

to their respective limits x_* and x'_*. Thus, we have

$$\overline{\lim}_k \, \|x_k - x_*\|^{1/k} \leq \delta_* \quad \text{and} \quad \overline{\lim}_k \, \|\tilde{x}'_k - x'_*\|^{1/k} \leq \delta_* \, . \tag{34}$$

Proof. Abbreviating $\widehat{x}_k \equiv x_k - x_*$ and with r_k as defined in (12), we have by (13)

$$\widehat{x}_{k+m} = \left(\prod_{j=1}^{m} D_{k+m-j} \right) \widehat{x}_k + \sum_{i=1}^{m} \left(\prod_{j=1}^{m-i} D_{k+m-j} \right) r_{k+i-1} \tag{35}$$

over a cycle of m steps. Because of the assumed convergence of the x_k and (32), the D_i are uniformly bounded in norm by $(1 + c_0)^2$, so that by (33) for any ε and sufficiently large k, we have

$$\begin{aligned} \|\widehat{x}_{k+m}\| &\leq (\delta_m + \varepsilon)^m \, \|\widehat{x}_k\| + \max_{0 \leq i < m} \|r_{k+i}\| \sum_{i=1}^{m} (1 + c_0^2)^{i-1} \\ &\leq (\delta_m + \varepsilon)^m \, \|\widehat{x}_k\| + \max_{0 \leq i < m} \|r_{k+i}\| (1 + c_0^2)^m / c_0^2 \, . \end{aligned} \tag{36}$$

Because of (13), the assumed convergence, and the uniform boundedness of the D_i, the $\widehat{x}_k$ grow at most linearly. Therefore, for some constant $c_7 = c_7(m)$

$$\|r_{k+j}\| \leq c_7 \|\widehat{x}_k\|^2 \quad \text{for} \quad 0 \leq j < m \, . \tag{37}$$

Hence we have by (35) for fixed m

$$\overline{\lim}_k \, \|\widehat{x}_{k+m}\| / \|\widehat{x}_k\| \leq (\delta_m + \varepsilon)^m \, ,$$

which ensures m-step Q-linear convergence with a limiting ratio no greater than δ_m^m, since ϵ may be chosen arbitrarily small. This implies the R-linear convergence assertion for the x_k by well-known results [20] and by taking the infimum of δ_m over m.

For the derivatives, we obtain from (7) the recurrence for the $\widehat{x}'_k \equiv x'_k - x'_*$

$$\widehat{x}'_{k+m} = \left(\prod_{j=1}^{m} D_{k+m-j} \right) \widehat{x}'_k + \sum_{i=1}^{m} \left(\prod_{j=1}^{m-i} D_{k+m-j} \right) r'_{k+i-1} \, , \tag{38}$$

where r'_k is as defined in (8). Since the last bound in Lemma 1 was proven without any reference to Assumption 2, it can be used here to derive from the R-linear convergence of the x_k that for some constant $c_8 = c_8(m, \varepsilon)$

$$\max_{0 < i < m} \|r'_{k+i}\| (1 + c_0^2)^m / c_0^2 \leq c_8 (\delta_m + \epsilon)^{k+m} \, . \tag{39}$$

Substituting this bound into the "primed" version of (36) and then dividing by $(\delta_m + \varepsilon)^{k+m}$, we obtain the inequality

$$\frac{\|\widehat{x}'_{k+m}\|}{(\delta_m + \varepsilon)^{k+m}} - \frac{\|\widehat{x}'\|}{(\delta_m + \varepsilon)^k} \leq c_8 \, .$$

Summing for $k = i\,m$ over $i = 0, 1, \ldots, j-1$, we obtain

$$\|\widehat{x}'_{j\,m}\| / (\delta_m + \varepsilon)^{j\,m} \leq \|\widehat{x}'_0\| + j\,c_8 \, .$$

Since the mj−th root of the right-hand side converges to one, we obtain the asserted result, namely, that the x'_k converge with the same linear R-factor δ_* to x'_*. ▮

As we have noticed above, the x_k may converge superlinearly. In those cases, the recurrence for x'_k will soon be almost exactly linear, so that one may seriously consider accelerating the derivative convergence by Richardson extrapolation. Since we have a constructive test on the quality of these extrapolated derivatives, it should be easy to determine the best candidate.

Finally, let us briefly examine the possibility that an iterative method of the general form achieves convergence but that assumption (33) is never satisfied. Then equation (36) suggests that a small perturbation δx_k of the iterate x_k in the direction of the largest singular value of $D_{k+m} \cdot D_{k+m-1} \cdots D_{k+2} \cdot D_{k+1}$ will not be damped out over an arbitrarily large number m of steps. This would indicate that the method is numerically rather unstable. We cannot make this claim rigorously, however, because the perturbation δx_k might alter the D_{k+j} in such a fortuitous way that it is damped out after all. For example, it is currently not clear whether conjugate direction methods can be interpreted in form (1) such that assumption (33) is satisfied. Derivative convergence has been observed for the classical conjugate gradient method, but this experimental observation cannot be supported by Proposition 1 and its corollaries.

6 CONCLUSION AND DISCUSSION

In this paper, we have described conditions under which derivative convergence is achieved, albeit possibly at a slower rate than the underlying function iteration. This observation applies to the fully differentiated iteration as well as the simplified recurrence, in the latter case also for higher derivatives. Our purpose was mainly analytical, and we do not claim that either derivative recurrence is the most efficient procedure for calculating implicit derivatives. However, our numerical results do show that both the fully differentiated iteration and the simplified approach do provide sufficiently accurate derivatives.

One might argue that if a P_k with contractive $D_k = I - P_k F_x$ is known, the linear system $F_x x' = -F_t$ can be solved iteratively after the solution x_* has been computed with satisfactory accuracy. This approach has long been used by engineers, as evidenced for example in some references of [3]. It certainly may be advantageous to start the derivative recurrences (7) or (10) with an initial $x'_k = 0$ only when the underlying iteration has reached the vicinity of the solution point.

If for some weight vector w, one actually wants to calculate the adjoint sensitivity

$$w^T x'_k = -w^T F_x^{-1} F_t \, ,$$

then one should first compute $w^T F_x^{-1}$ iteratively using the approximate inverse P_k^T of F_x^T. This approach is particularly useful if t is actually a vector so that several linear systems need to be solved for computing x'_k. This iterative variant of the reverse mode for implicit gradients has been advocated and analyzed by Christianson in [7]. However, it should be noted that his analysis, if not the method itself, assumes that the Jacobian of the iteration function is not only contractive but also Lipschitz continuous in the current argument. This condition is certainly not satisfied by secant updating methods. Moreover, there are some important schemes like nonlinear conjugate gradients, which do not satisfy our slightly weaker assumptions either. The question of what happens under those circumstances and several practical implementation aspects remain to be investigated.

ACKNOWLEDGEMENTS

We are indebted to John Dennis for his insistence that black-box differentiation be analyzed.

References

[1] Christian Bischof, Alan Carle, George Corliss, Andreas Griewank, and Paul Hovland. ADIFOR–Generating Derivative codes from Fortran Programs. Scientific Programming, 1:11-29, 1992.

[2] Christian Bischof, George Corliss, and Andreas Griewank. Structured second- and higher-order derivatives through univariate Taylor series. Preprint MCS-P296-0392, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., March 1992. ADIFOR Working Note #6.

[3] Christian Bischof, George Corliss, Larry Green, Andreas Griewank, K. Haigler, and Perry Newman. Automatic differentiation of advanced CFD codes for multidisciplinary design. Preprint MCS-P339-1192, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., January 1993.

[4] C. G. Broyden. A class of methods for solving nonlinear simultaneous equations. Math. Comp. 19:577–593, 1965.

[5] C. G. Broyden. The convergence of a class of double rank minimization algorithms. J. Inst. Math. Appl. 6:222–231, 1970.

[6] C. G. Broyden, J. E. Dennis, J. J. Moré. On the local and superlinear convergence of quasi-Newton methods. J. Inst. Math. Appl. 12:223–245, 1973.

[7] Bruce Christianson. Reverse Accumulation and Attractive Fixed Points. Numerical Optimisation Centre Report 258, University of Hertfordshire, 1992.

[8] George F. Corliss. Overloading point and interval Taylor operators. In Andreas Griewank and George F. Corliss, editors, *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, pages 139–146. SIAM, Philadelphia, Penn., 1991.

[9] W. C. Davidon. Variable metric method for minimization. Report ANL-5990 (Rev.), Argonne National Laboratory, Argonne, Illinois, 1959.

[10] J. E. Dennis, Jr., D. M. Gay and R. E. Welsch. An Adaptive Nonlinear Least-Squares Algorithm. *TOMS*, 7:348-368, 1981.

[11] J. E. Dennis, Jr., D. M. Gay and R. E. Welsch. Algorithm 573 NL2SOL–An Adaptive Nonlinear Least-Squares Algorithm. *TOMS*, 7:369–383, 1981.

[12] John E. Dennis, Jr., Guangye Li and Karen A. Williamson. Optimization Algorithms for Parameter Identification. Technical Report CRPC-TR92277, Center for Research on Parallel Computation, Rice University, 1992.

[13] J. E. Dennis and J. J. Moré. Quasi-Newton methods, motivation and theory. *SIAM Review*, 19:46–89, 1977.

[14] R. Fletcher and M. J. D. Powell. A rapidly convergent descent method for minimization. *Comput. J.*, 6:163–168, 1963.

[15] J. Ch. Gilbert. Automatic differentiation and iterative processes. *Optimization Methods and Software*, 1:13–21, 1992. Also appeared as a preprint, INRIA, Le Chesnay, France, 1991.

[16] Andreas Griewank. Automatic evaluation of first- and higher-derivative vectors. In R. Seydel, F. W. Schneider, T. Küpper, and H. Troger, editors, *Proceedings of the Conference at Würzburg, Aug. 1990, Bifurcation and Chaos: Analysis, Algorithms, Applications*, volume 97, pages 135–148. Birkhäuser Verlag, Basel, Switzerland, 1991.

[17] Andreas Griewank, David Juedes, Jay Srinivasan, and Charles Tyner. ADOL-C, a package for the automatic differentiation of algorithms written in C/C++. *ACM Trans. Math. Software*, to appear. Also appeared as Preprint MCS-P180-1190, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Ill., November 1990.

[18] J. J. Moré, B. S. Garbow, and K. E. Hillstrom. User guide for MINPACK-1. Technical Report ANL-80-74, Argonne National Laboratory, 1980.

[19] M. J. D. Powell. A hybrid method for nonlinear equations. In P. Rabinowitz, editor, *Numerical Methods for Nonlinear Algebraic Equations*, pages 87–114. Gordon and Breach, London, 1970.

[20] J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.