Finite Differences vs Automatic Differentiation for Restartable Iterative Procedures

Alan Carle and Mike Fagan

CRPC-TR97733
December 1997

Center for Research on Parallel Computation
Rice University
6100 South Main Street
CRPC - MS 41
Houston, TX 77005

Submitted December 1997

Finite Differences vs Automatic Differentiation for Restartable Iterative Procedures*

CRPC-TR97733

Alan Carle Mike Fagan

December 4,1997

Abstract

Many scientific programs that use iterative processes to compute their results also include a *restart* capability. This report investigates some methods that use restarts to improve the performance of sensitivity calculations. The investigation covers both finite-difference sensitivity methods as well as forward-mode automatic differentiation.

1 Introduction

Many useful programs compute their results by producing a sequence of iterates, terminating when the iterates are "sufficiently close" to a true fixed point of the iterative procedure. A prototypical iterative procedure has as its principal component a stepping function s that operates on two components: the independent input G, and the previous iterate q_n. The result of s is the next iterate q_{n+1}. The object of the iterative procedure is to find (approximate) the fixed point of s. That is, the iterative procedure seeks to approximately compute a q_* so that

$$q_* \approx s(G, q_*).$$

The iterative process begins with some canonical choice of initial iterate q_0.

Furthermore, to add a measure of fault tolerance, many iterative programs employ a "restart" mechanism. A restart mechanism periodically saves iterates for possible use at a later time. If a fault occurs, the most recently saved iterate can be used to *restart* the iterative procedure. More concretely, suppose an iterative computation begins with G and the canonical q_0. After 100 steps, the process saves iterate q_{100} as a restart. At step 102, a fault occurs. By using a restart, the iterative procedure may *resume* the computation using the previously saved q_{100}, rather than starting over with q_0.

In addition to fault tolerance, a restart mechanism may also facilitate the computation of sensitivity information. To understand the essential idea, consider a typical finite-difference sensitivity calculation for an iterative procedure. Using input G and the canonical starting point q_0, say the iterative procedure converges to q_*. The finite-difference step requires a perturbed input, say $G + \delta$. Then, again using the canonical starting point q_0, the iterative procedure converges to q_*^δ. The sensitivity of the iterative procedure, then, is

$$\frac{q_*^\delta - q_*}{\delta}.$$

*This work was supported by the National Aeronautics and Space Administration under Cooperative Agreement No. NCC 1 236, and by the National Science Foundation, through the Center for Research on Parallel Computation, under Cooperative Agreement No. CCR-9120008. Additional computing resources provided by NASA Ames High Performance Computing and Communication Program Testbed 1 Cooperative Research Agreement (through IBM P.O. # 835600Z-E1).

For "small" values of δ, one would expect q_*^δ to be "close" to q_*. Consequently, if the perturbation step uses q_* as its starting point, instead of the canonical q_0, then the iterative procedure might converge to q_*^δ in fewer steps, since it started "closer" to the final value. If the iterative procedure employs restarts, then the perturbation step of a finite-difference calculation may be accomplished by "restarting" the iterative procedure with $G + \delta$ and q_*.

The restart feature also has favorable implications when computing sensitivities by automatic differentiation (AD). If some "close" approximation to the derivatives are available, then restarting the derivative iteration with the close approximation may take less iterations to converge to the desired results. In this report, we examine more closely both finite differences and automatic differentiation with respect to restarts. We divide our analysis into four sections. Section 2 details our model of the iterative process, including our assumptions on the mathematical properties of the processes in question. Section 3 uses the model to analyze finite differences under restart. Section 4 uses the model to analyze the AD process. Section 5 summarizes our findings.

2 The Model

Our model of the iterative procedure consists of three components: a stepping function s that advances a given iterate to the next iterate, a canonical starting point, and a stopping criterion. As mentioned in section 1, the stepping function s takes two arguments: the independent variable G, and the current iterate q. Both G and q may be vectors. We model the stepping function in a general way, making no assumptions about its internal structure. We will later discuss the relevant mathematical properties. We notate a single step of function s as follows:

$$q_{k+1} = s(G, q_k).$$

We identify the canonical starting point as q_0.

For any *fixed* number of steps, we can model the iterative process as a "correction" function $s^n(G, q)$, where

$$s^n(G,q) = s(G, s^{n-1}(G,q)) = s(G, s(G, s^{n-2}(G,q)))\ldots.$$

For consistency, define $s^0(G,q) = q$.

The stopping criterion controls the *number* of iterations in the process. This number may be highly dependent on the input variables. For example, the "run until converged" criterion could certainly take fewer steps if the starting q is "close" to the final answer than if q is "far away" from the final answer.

Stopping criteria are strongly application dependent (some say they are strongly application *user* dependent). In light of the large possible variation, we will limit our analysis to the "small" residual convergence criterion. We model that as follows:

Definition 1 (Tolerance R fixed-point approximation(RFPA)) *For iterative step $q_k = s(G, q_{k-1})$, we say step k has converged to within residual R if*

$$||q_k - s(G, q_k)|| \le R.$$

The q_k quantity is a tolerance R fixed-point approximation for s. We will use the abbreviation RFPA for "tolerance R fixed-point approximation".

Note that RFPAs need not be unique. For further analysis, we need to identify a unique such approximation. We will distinguish a canonical RFPA based on the canonical starting point q_0.

Definition 2 (Canonical Tolerance R fixed-point approximation) *Let q_0 be the canonical starting point for s. Let q_N be an RFPA derived from N steps of s, starting at q_0. That is,*

$$q_N = s^N(G, q_0)$$

where

$$||q_N - s(G, q_N)|| \leq R$$

but

$$\left\| q_{N-1} - s^{(N-1)}(G, q_0) \right\| > R.$$

Using this model of the computation, we wish to compute the sensitivity of the canonical RFPA of s with respect to the inputs G.

Assumptions on the iterative step function s

Following Christianson[1] and Griewank *et al*[3] [1], we assume that the iterative step function s has certain reasonable convergence properties. First, we assume that for any given G, s has a fixed point. That is, we assume:

$$\text{for all } G \text{ there exists } q^* \text{ s.t. } q^* = s(G, q^*).$$

In addition, we assume the behavior of s is regular in that s has the following properties:

1. s is continuously differentiable.
2. $s' \equiv (\frac{\partial s}{\partial G}, \frac{\partial s}{\partial q})$ is Lipschitz (with constant C).
3. s is attractive with attraction constant τ (that is, $\left\| \frac{\partial s}{\partial q} \right\| \leq \tau$).

We use the contractivity property to establish a simple estimation procedure for RFPAs.

Estimate 1 *[(Distance between RFPAs)] Let q_1 and q_2 be any two RFPAs for the same input variables, that is*

$$||q_i - s(G, q_i)|| \leq R, \;\; i = 1, 2.$$

Then

$$||q_1 - q_2|| \leq \frac{2R}{1 - \tau}.$$

We establish this estimate as follows:

Letting q^* be the actual fixed point of $s(G, q)$, we use a result of Christianson (derived by simple line integration) to get

$$||q^* - s(G, q)|| \leq \tau \, ||q^* - q|| \, .$$

Using the fact that q_i are RFPAs, we note

$$||q_i - s(G, q_i)|| \leq R, \;\; i = 1, 2.$$

[1] Griewank[3] prefer a different, less restrictive (but more complicated) condition than attractivity (property 3). We will follow the simpler case here, and conjecture that similar results hold for the Griewank conditions.

So

$$\begin{aligned}
||q^* - q_i|| &= ||q^* - s(G, q_i) + s(G, q_i) - q_i|| \\
&\leq ||q^* - s(G, q_i)|| + ||s(G, q_i) - q_i|| \\
&\leq \tau\, ||q^* - q_i|| + R \\
&\quad \text{for } i = 1, 2.
\end{aligned}$$

Rearranging terms yields

$$||q^* - q_i|| \leq \frac{R}{1-\tau},\ i = 1, 2.$$

We can now get our estimate by simply adding and subtracting q^* to the desired difference:

$$\begin{aligned}
||q_1 - q_2|| &= ||q_1 - q^* + q^* - q_2|| \\
&\leq ||q_1 - q^*|| + ||q^* - q_2|| \\
&\leq \frac{R}{1-\tau} + \frac{R}{1-\tau} \text{ by previous reasoning} \\
&= \frac{2R}{1-\tau}.
\end{aligned}$$

3 Finite Differences

To preface our analysis of finite differencing, we need to clarify the finite-difference methods under examination. We will elaborate the "standard" finite-difference method, as well as a prototypical finite-difference method that uses restarts. A straightforward finite-difference protocol works as follows:

Protocol 1 (Standard finite differences)

1. Starting with "baseline" inputs G_b, compute the canonical tolerance R fixed-point approximation q_b. Assume this step takes N iterations.
2. Perturb one of the inputs, yielding $G_b + d$, and find the canonical tolerance R fixed point approximation of this new input. Subtract q_b from this approximation, and divide by d.
3. Repeat step 2 as many times as there are inputs.

Using restarts, protocol 1 generalizes to:

Protocol 2 (Restart-enhanced Finite Differences)

1. Using the base inputs G_b, the iterative process runs for P steps, starting with the canonical starting point q_0. After P steps, we have some updated output value q_s (not necessarily converged). The q_s output is saved in a restart file.
2. Restarting with q_s, continue on to tolerance R convergence. This should be g_b, the same value as step 1 of protocol 1.
3. Perturb one of the inputs, yielding $G_b + d$. Then, restarting with q_s from step 1 and the newly perturbed input, compute a new tolerance R fixed-point approximation q_d^r. The finite difference computation yields

   $$\frac{q_d^r - g_b}{d}.$$

4. Repeat step 3 for each input in G. We note that it is possible that different inputs *might* require different numbers of steps to converge.

A common variation of this protocol combines steps 1 and 2 so that $q_s = q_b$.

As mentioned in the introduction, the main advantage of protocol 2 is a potential decrease in execution time. If d is "small", then q_s should be "close" to the iterate that would have been computed at that point in the process. Hence, the number of steps to correct q_s to the fixed point for $G + d$ should be (or at least might be) less than the number of steps to correct the canonical starting point q_0.

Differences between Finite-Difference Protocols

To analyze the difference between protocol 1 and protocol 2, let us assume that d has been chosen so that

$$\left\| \frac{(s^N(G+d, q_0) - s^N(G, q_0))}{d} - \frac{\partial s^N}{\partial G}(G, q_0) \right\| < \epsilon.$$

Writing q_b for $s^N(G, q_0)$, q_d for $s^N(G+d, q0)$, and $q_r = s^N(G+d, q_s)$, and noting that q_i are all RFPAs, we add and subtract q_r from the previous inequality to get

$$\begin{aligned}
\left\| \frac{(q_d - q_r + q_r - q_b)}{d} - \frac{\partial s^N}{\partial G}(G, q_0) \right\| &< \epsilon \\
\left\| \frac{(q_d - q_r)}{d} + \frac{(q_r - q_b)}{d} - \frac{\partial s^N}{\partial G}(G, q_0) \right\| &< \epsilon \\
\left\| \frac{(q_r - q_b)}{d} - \frac{\partial s^N}{\partial G}(G, q_0) \right\| - \left\| \frac{(q_d - q_r)}{d} \right\| &< \epsilon \\
\left\| \frac{(q_r - q_b)}{d} - \frac{\partial s^N}{\partial G}(G, q_0) \right\| &< \left\| \frac{(q_d - q_r)}{d} \right\| + \epsilon \\
\left\| \frac{(q_r - q_b)}{d} - \frac{\partial s^N}{\partial G}(G, q_0) \right\| &< \frac{2R}{(1-\tau)d} + \epsilon \text{ by estimate 1.}
\end{aligned}$$

Examining this inequality shows that for careful choice of d and R, then reasonable accuracy may be obtained for protocol 2. Note however, that R should be a few orders of magnitude less than d, otherwise, the term $\frac{2R}{(1-\tau)d}$ may be too large.

4 Automatic Differentiation

Automatic differentiation of program code produces a new code that computes the derivative based on the well known rules of calculus, using the chain rule to link together the various operations in the differentiated program. The program under consideration iterates the computation of s, given G and some starting point q. Thus, automatic differentiation of s with respect to G yields a single step iterative procedure for $\frac{ds}{dG}$. Moreover, the automatic differentiation procedure in question *augments* the computer code with derivative computation, so the derivative-augmented s computes both s and $\frac{ds}{dG}$.

As with finite differences in section 3, the ability to restart a computation suggests multiple possible protocols for automatic differentiation as well.

The standard AD protocol is:

Protocol 3 (Standard AD)

> Using the base inputs G_b, the canonical starting point q_0, and the associated canonical q'_0, run the derivative computation for N steps, until the derivative is converged to the desired accuracy. We note that q'_0 is *usually*, but not always, 0.

Similarly, the standard restart-AD protocol is:

Protocol 4 (Restart-enhanced AD)

> Using the base inputs G_b, restart q_r, and an associated restart q'_r, run the derivative computation for M steps, until the derivative is converged to the desired accuracy.

There is, however, a variation of protocol 4 that has no analog with finite differences. It is possible to use restarts for function value only, yielding this variant:

Protocol 5 (Restart-function AD)

1. Using the (unaugmented) s procedure, generate a final q_f that is converged to the desired accuracy.
2. Using G_b, restart q_f, and the *canonical* derivative starting point q'_0, run the augmented s until the derivatives are converged to the desired accuracy.

Previous work by Christianson[1], Gilbert, and Griewank[3] has shown that the conditions we placed on s ensure Q-linear convergence of the function iteration, and R-linear convergence of the derivative iteration. In other words, the derivative iterates converge “at roughly the same rate” as the function iterates. So protocol 3 should take “about” as many steps to converge the derivatives as the function iteration required to converge. Similarly, protocol 4 should take approximately the same number of steps to converge derivatives from the derivative restart value as the function value does from the function restart. If the starting point is “closer” to the fixed point, then this iteration should take less steps to converge than protocol 3. Protocol 5 should take “about” the same number of steps as protocol 3. In other words, having a good starting point for only the function value is probably *not* sufficient to reduce the required number of steps for derivative convergence. We will analyze these protocols more precisely in the remainder of this section. Our main analytical tool is Estimate 2. To simplify notation, we will usually write q' for $\frac{dq}{dG}$.

Estimate 2 *Let $q_{n+1} = s(G, q_n)$ be the function iteration. Then*

$$\left\| q'_* - q'_{n+1} \right\| \le C \left\| q_* - q_n \right\| + \tau \left\| q'_* - q'_n \right\|$$

where C is the Lipschitz constant and τ is the attractivity constant.

We derive this estimate by the following steps:

> Differentiating s yields (by chain rule) and evaluating at q_n and q_* yields:
>
> $$\begin{aligned} \left.\frac{ds}{dG}\right|_{(G_s,q_n)} &= \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_n)} + \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_n)} \frac{dq_n}{dG}, \\ \left.\frac{ds}{dG}\right|_{(G_s,q_*)} &= \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_*)} + \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_*)} \frac{dq_*}{dG}. \end{aligned}$$
>
> Subtracting, and rewriting using our simplified notation gives:
>
> $$q'_{n+1} - q'_* = \left(\left.\frac{\partial s}{\partial G}\right|_{(G_s,q_n)} - \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_*)} \right) + \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_n)} q'_n - \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_*)} q'_*.$$

Taking norms yields:

$$\left\|q'_{n+1} - q'_*\right\| \le \left\| \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_n)} - \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_*)} \right\| + \left\| \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_n)} \right\| ||q'_n|| - \left\| \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_*)} \right\| ||q'_*|| \,.$$

Finally, we note that our original assumptions on the Lipschitz continuity of s' and the contractivity of s establish the bounds we need

$$\begin{aligned} \left\|q'_{n+1} - q'_*\right\| &\le \left\| \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_n)} - \left.\frac{\partial s}{\partial G}\right|_{(G_s,q_*)} \right\| + \left\| \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_n)} \right\| ||q'_n|| - \left\| \left.\frac{\partial s}{\partial q}\right|_{(G_s,q_*)} \right\| ||q'_*|| \\ &\le C\,||q_n - q_*|| + \tau\,||q'_n|| - \tau\,||q'_*|| \\ &\le C\,||q_n - q_*|| + \tau\,||q'_n - q'_*|| \,. \end{aligned}$$

Examining each of our protocols in light of this estimate gives us a better understanding of its behavior. For protocol 3, after N steps, say, the function value has converged to within R of the fixed point, then the derivatives will be

$$\left\|q'_{N+1} - q'_*\right\| \le CR + \tau\,||q'_N - q'_*|| \,.$$

So the derivatives are converged to "about" the same accuracy. If the CR product is fairly large, then a few more iterations may be required to ensure the desired accuracy of the derivative.

Similarly, for protocol 4, we note that if both q_r and q'_r are the same order of approximation to q_* and q'_*, then derivative and function value iterations will yield similar accuracies. The number of iterations to yield that accuracy depends on the accuracy of *both* the restart q_r and q'_r.

This observation leads us to conclude that protocol 5 will generally require about the same number of iterations as protocol 3 for derivative convergence. Suppose we start derivative iteration at q_* exactly. Then Estimate 2 still indicates

$$\left\|q'_* - q'_{n+1}\right\| \le \tau\,||q'_* - q'_n|| \,.$$

Hence, starting at q'_0 will still require about the same number of steps to converge derivatives as protocol 3. Experimental confirmation of this behavior for a computational fluid dynamics code is reported by Green[2].

5 Conclusions

Our main conclusion is that restart capability enables potential improvement in both finite difference and automatic differentiation sensitivity calculations. As always, for finite differences, the step size is important. It interacts inversely with the desired tolerance. The function tolerance should be several orders of magnitude smaller than the step size to ensure the desired approximation to the sensitivity.

For automatic differentiation, the main consideration is that both the function restart and the derivative restart be the same order of closeness to their respective fixed points. Estimate 2 indicates that the number of steps to converge derivatives depends on *both* function accuracy and derivative accuracy.

References

[1] Bruce Christianson. Reverse accumulation and attractive fixed points. *Optimization Methods and Software*, 3:311–326, 1994.

[2] Lawrence L. Green, Perry A. Newman, and Kara J. Haigler. Sensitivity derivatives for advanced cfd algorithm and viscous modeling parameters via automatic differentiation. *Journal of computational physics*, 125(2), 1996.

[3] Andreas Griewank, Christian Bischof, George Corliss, Alan Carle, and Karen Williamson. Derivative convergence for iterative equation solvers. *Optimization Methods and Software*, 2:321–355, 1993.