

Stopping tests for optimization and nonlinear equations

Mark S. Gockenbach

1 Introduction

Since algorithms for nonlinear optimization are iterative in nature, with computed solutions converging to the true solution only in the limit, stopping criteria form an important part of an implementable algorithm. A good optimization code will stop the iteration when one of several conditions has been met. The following material is adapted from Dennis and Schnabel [1], Chapter 7.

2 Stopping tests for optimization

2.1 The gradient tolerance

In a quasi-Newton (descent) algorithm, it is (implicitly) assumed that approximating a stationary point is equivalent to solving a minimization problem. Therefore, the most important stopping test should be something like

$$\|\nabla f(x^{(k)})\| \leq \epsilon_1. \quad (1)$$

However, a little thought shows that such a simplistic test is inadequate; the reason is that it fails to take into account the scale of the problem. It should be obvious that a given $x^{(k)}$ is an equally good (or bad) approximate solution to

$$\min f(x)$$

and

$$\min \tilde{f}(x)$$

if $\tilde{f}(x) = 10^8 f(x)$. However, for a given ϵ_1 , $x^{(k)}$ is likely to fail to satisfy $\|\nabla \tilde{f}(x^{(k)})\| \leq \epsilon_1$ even if $\|\nabla f(x^{(k)})\| \leq \epsilon_1$ holds. For this reason, the stopping test should take into account the scale of f .

Somewhat less obvious is that the scale of x also must be taken into account. The quantity $-\|\nabla f(x)\|$ is the predicted decrease in f if a step of length one is taken in the steepest descent direction. Whether $\|\nabla f(x)\|$ is considered large depends on whether a step of length one is large or small. If, for instance, $\|x^{(k)}\| = 10^8$, then a step of length one is a very small step, and $\|\nabla f(x^{(k)})\| = 10^{-2}$ might be considered large. On the other hand, if $\|x^{(k)}\| = 10^{-6}$, then $\|\nabla f(x^{(k)})\| = 10^{-2}$ should probably be regarded as rather small.

Because of the above arguments, it is natural to measure the *relative gradient* g_{rel} of f , defined by

$$g_{rel}(x)_i = \lim_{h \rightarrow 0} \frac{\frac{f(x+he_i) - f(x)}{f(x)}}{\frac{h}{x_i}} = \frac{\nabla f(x)_i x_i}{f(x)}.$$

The iteration can then be stopped if

$$\|g_{rel}(x^{(k)})\|_\infty = \max \left\{ \left| g_{rel}(x^{(k)})_i \right| : i = 1, 2, \dots, n \right\} \leq \epsilon_1. \quad (2)$$

There is still one difficulty with (2), though: relative measures are not well-defined when the base quantity (in this case $f(x^{(k)})$ or x_i) approaches zero. To protect against this, the user can be asked

to provide typical values for $f(x)$ and $|x_i|$, and these are used to protect against division by zero (or a very small quantity). The test then becomes

$$\max \left\{ \frac{\nabla f(x^{(k)})_i \max\{|x_i^{(k)}|, \text{typx}_i\}}{\max\{|f(x^{(k)})|, \text{typf}\}} \right\} \leq \epsilon_1. \quad (3)$$

A simpler version of (3) is

$$\frac{\|\nabla f(x^{(k)})\| \max\{\|x^{(k)}\|, \text{typxnorm}\}}{\max\{|f(x^{(k)})|, \text{typf}\}} \leq \epsilon_1. \quad (4)$$

Condition (4) is usually satisfactory, although (3) is preferable if the components of x tend to differ by orders of magnitude.

The quantity ϵ_1 in (3) or (4) is called the *gradient tolerance*.

2.2 The step tolerance

In some problems, it is difficult to satisfy the gradient tolerance, particularly if the gradient is computed only approximately. For this reason, a secondary stopping test is usually implemented that halts the iteration if the step taken becomes small. The reader will recall that, for a superlinearly convergent algorithm, this condition suggests that a good approximation of x^* has been obtained. Since it is important to take the scale of the problem into account in this stopping test as well, the relevant conditions are

$$\max \left\{ \frac{|x_i^{(k+1)} - x_i^{(k)}|}{\max\{|x_i^{(k)}|, \text{typx}_i\}} : i = 1, 2, \dots, n \right\} \leq \epsilon_2 \quad (5)$$

or

$$\frac{\|x^{(k+1)} - x^{(k)}\|}{\max\{\|x^{(k)}\|, \text{typxnorm}\}} \leq \epsilon_2. \quad (6)$$

The quantity ϵ_2 is referred to as the *step tolerance*.

2.3 Safeguards

In spite of the fact that there are global convergence theorems for many optimization algorithms, some problems are so nonlinear that convergence is impossible in a reasonable number of iterations. Moreover, users sometimes try to solve incorrectly-posed problems (such as trying to minimize a function that is unbounded below). For this reason, every good optimization code allows the user to specify a maximum number of iterations. If neither the gradient tolerance nor the step tolerance has been satisfied when this limit is reached, the algorithm simply halts and reports failure.

Sometimes another stopping test is implemented specifically to detect divergence (as may occur, for instance, with a unbounded objective function). The algorithm can keep track of the lengths of the steps, and if several consecutive steps are taken that exceed a pre-determined limit, the algorithm assumes that the sequence is diverging.

3 Stopping tests for nonlinear equations

The step tolerance test is equally valid for the problem of solving $F(x) = 0$, as are tests for divergence or an iteration limit. The gradient test should be modified to a properly scaled version of

$$\|F(x^{(k)})\| \leq \epsilon_3.$$

Here there is the difficulty that $F(x^{(k)})$ is *supposed* to converge, so the obvious tests for the relative size of $F(x^{(k)})$ are meaningless. It is therefore essential that the user provide a typical size for $F(x)$ when x is not too close to the solution (failing input from the user, the value of $F(x^{(0)})$ could be used to determine this size).

The primary stopping test for nonlinear equations therefore takes the form

$$\max \left\{ \frac{|F(x^{(k)})_i|}{\text{typF}_i} : i = 1, 2, \dots, n \right\} \leq \epsilon_3 \quad (7)$$

or

$$\frac{\|F(x^{(k)})\|}{\text{typFnorm}} \leq \epsilon_3. \quad (8)$$

In (8), typFnorm is a positive scalar, while in (7) typF is a vector of positive scalars. The quantity ϵ_3 is called the *function tolerance*.

Finally, an algorithm for nonlinear equations should check explicitly whether the iteration is converging to a local, nonglobal minimizer of $f(x) = (1/2)\|F(x)\|^2$. A relative gradient check similar to (3) or (4) should be employed.

All codes, whether for optimization or nonlinear equations, should report which stopping test caused the algorithm to halt.

References

- [1] J.E. Dennis, Jr. and R.B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Prentice-Hall, Englewood Cliffs, 1983.