# Limitations of Backward Error Analysis

S. Qiao and D. L. Parnas
Department of Computing and Software
McMaster University
Hamilton, Ontario L8S 4L7

July 2000

**Abstract**

We know that a small backward error given by a backward error analysis of a numerical method ensures the stability of the method. In this paper, we show, through examples, that a large backward error or non-existence of backward error does not imply instability. In fact, a method can be stable and deliver accurate results although the backward error is large.

Backward error analysis is a powerful technique systematically developed by Wilkinson [6] in 1960s. An early and implicit version can be dated back to 1948 by Turing [5]. In this note, we discuss some issues in backward error analysis. First we give a brief description of the technique.

An algorithm can be viewed as a function

$$y = g(x)$$

mapping the input vector $x$ into the output vector $y$. Because of rounding errors, the computed result, denoted by $\widehat{y} = fl(g(x))$, is an approximation of the exact result $y$. Instead of measuring the error between the computed $\widehat{y}$ and $y$, which is usually unknown, backward error analysis interprets rounding errors as being equivalent to the perturbations in the input data. Specifically, we express the computed result as

$$\widehat{y} = g(x + \Delta x).$$

The input data are often subject to uncertainty caused by measuring errors and storing numbers in the computer. Thus, if such $\Delta x$ exists and is no larger than the uncertainty in the input, we are happy with the computed result. Then, we call the algorithm backward stable. If there is more than one possible value of $\Delta x$, we use the smallest one. The following definition is from [3]. Similar definitions are given in [1, 2, 4] and other books.

**Definition 1 (Backward Stable)** *A method for computing $y = g(x)$ is called backward stable if, for any $x$, it produces a computed $\widehat{y}$ with a small backward error, that is, $\widehat{y} = g(x + \Delta x)$ for some small $\Delta x$.*

The above definition implicitly assumes that such $\Delta x$ exists and says nothing about the case when $\Delta x$ is large. However, the implication might be that when $\Delta x$ does not exist or is large, it is because the result is inaccurate.

Now, we address the following issues in backward error analysis.

1. The existence of $\Delta x$;

2. The implication of a large $\Delta x$.

In the following examples, we assume that the machine precision is four decimal digits and we carry decimal arithmetic operations in the nearest rounding mode. Thus the unit of roundoff is $u = 0.0005$.

In the first example, we show that such $\Delta x$ may not exist.

**Example 1**. Consider the method

$$y = g(x) = \begin{cases} x + 1.0 & x < 1.0 \\ x + 2.0 & x \geq 1.0 \end{cases}$$

When $x = 0.9999$, the computed value

$$\widehat{y} = fl(g(0.9999)) = 2.0.$$

Since $g(x) < 2.0$ or $g(x) \geq 3.0$, there does not exist a $\Delta x$ such that $g(0.9999 + \Delta x) = 2.0$ exactly.

To be precise, we present a backward error analysis of this example. Assuming $0 < x < 1.0$, rounding error analysis shows that

$$\widehat{y} = fl(g(x)) = (x + 1.0)(1 + \delta), \quad |\delta| \leq u. \tag{1}$$

In backward analysis, we find a backward error $\Delta x$ such that

$$g(x + \Delta x) = \widehat{y}. \tag{2}$$

In order for $\Delta x$ to exist, it is necessary that $\widehat{y}$ be in the range, i.e., $\widehat{y} < 2$ or $\widehat{y} \geq 3$. From (1), we get

$$((x + 1)(1 + \delta) < 2) \vee ((x + 1)(1 + \delta) \geq 3),$$

which implies

$$\left(\delta < \frac{1-x}{1+x}\right) \vee \left(\delta \geq \frac{2-x}{1+x}\right) \tag{3}$$

respectively. In this example where $x = 0.9999$ and $\widehat{y} = 2.0$, we have

$$\delta = \frac{\widehat{y}}{1+x} - 1 = \frac{0.0001}{1.9999} = \frac{1-x}{1+x}.$$

Thus $\Delta x$ does not exist because the inequalities in (3) are not satisfied.

Let $\Delta y = \widehat{y} - g(x)$, then from (1)

$$|\Delta y| = |(x+1)\delta| \leq 2u \quad \text{when } 0 < x < 1.0,$$

recalling that $u = 0.0005$. Thus $\widehat{y}$ can be anywhere in the interval $[g(x) - 2u, g(x)+2u]$. However, as shown above, it is possible that some of the values, for example $\widehat{y} = 2.0$, in this interval are not in the range of $g(x)$ when $x$ is near 1. Consequently, there is no $\Delta x$ satisfying (2). In general, in order for $\Delta x$ to exist, the inverse function $g^{-1}(y)$ must exist for $y \in [g(x) - |\Delta y|, g(x) + |\Delta y|]$. If this interval is not contained in the range of $g(x)$, $\Delta x$ in (2) may not exist for some values of $\widehat{y}$.

One may suggest that the problem is caused by the discontinuity of $g(x)$. The computed value lies at the discontinuity and is out of the range of $g(x)$. The method in the second example is continuous.

**Example 2**. Let

$$y = g(x) = 1.0 + (0.9999 - x * x).$$

Note that the parentheses should not be removed. It is obvious that $g(x) \leq 1.9999$. However, consider the case where $x = 0.02$ and

$$\widehat{y} = fl(g(0.02)) = 2.0.$$

Again, there does not exist a $\Delta x$ such that $g(0.02 + \Delta x) = 2$ exactly.

Applying backward error analysis, we get

$$\widehat{y} = fl(g(x)) = (1 + (0.9999 - x^2(1+\delta_1))(1+\delta_2))(1+\delta_3), \quad |\delta_i| \leq u.$$

When $x = 0.02$, $x*x$ and $0.9999 - x*x$ are computed exactly. So, $\delta_1 = \delta_2 = 0$. Consequently,

$$\widehat{y} = (1.9999 - x^2)(1+\delta_3). \tag{4}$$

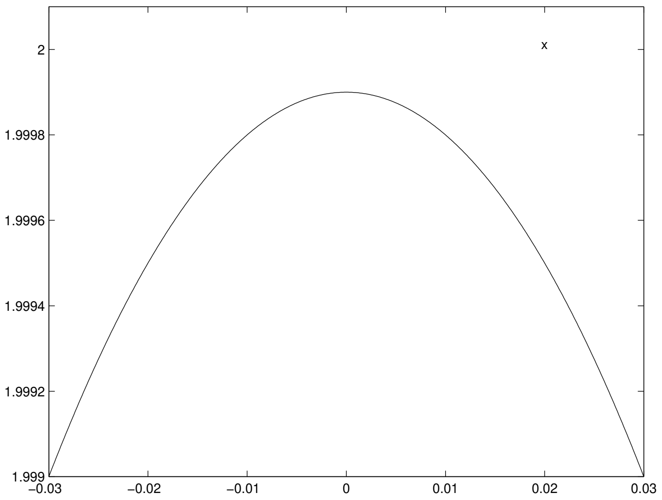


Figure 1: Exact $g(x)$ and computed $fl(g(0.02))$ (marked $\times$) in Example 2.

Setting $g(x + \Delta x) = \widehat{y}$, we get

$$1.9999 - (x + \Delta x)^2 = (1.9999 - x^2)(1 + \delta_3).$$

In order for $\Delta x$ to exist, we must have $(1.9999 - x^2)(1 + \delta_3) \leq 1.9999$, which implies

$$\delta_3 \leq \frac{0.0004}{1.9995}. \tag{5}$$

However, in this example, $\widehat{y} = 2.0$ and, from (4),

$$\delta_3 = \frac{\widehat{y}}{1.9999 - x^2} - 1 = \frac{0.0005}{1.9995}.$$

Thus (5) is not satisfied and $\Delta x$ does not exist.

Similar to Example 1, in general, $g^{-1}(y)$ should exist for $y \in [g(x) - |\Delta y|, g(x) + |\Delta y|]$, where $\Delta y = \widehat{y} - y$. In this example where $x = 0.02$, the computed result $\widehat{y} = 2.0$ exceeds the maximum $g(0) = 1.9999$, thus out of the range of $g(x)$. Figure 1 depicts the situation.

In the third example, the computed value is within the range and such $\Delta x$ exists, but is large.

**Example 3**. Evaluating

$$g(x) = \frac{x^3}{3} - x$$

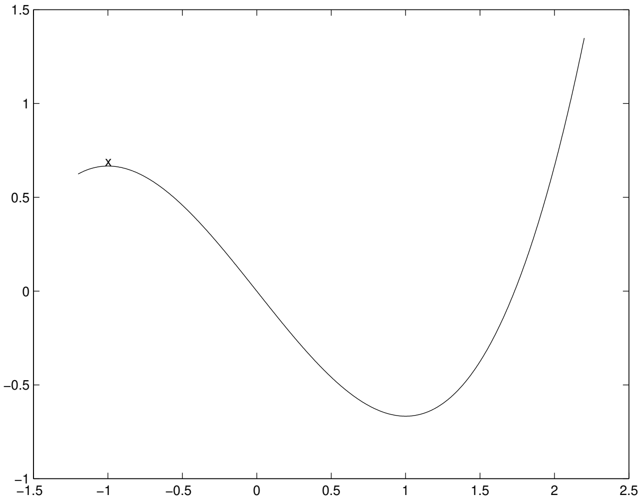


Figure 2: Exact $g(x)$ and computed $fl(g(-1.0))$ (marked $\times$) in Example 3.

at $x = -1.0$, we get

$$\widehat{y} = fl(g(-1.0)) = -0.3333 + 1.0 = 0.6667.$$

To find $\Delta x$ satisfying $g(x + \Delta x) = \widehat{y}$, we solve the cubic equation

$$\frac{x^3}{3} - x = 0.6667.$$

We get

$$x_0 = \sqrt[3]{1.00005 + \sqrt{0.00040001}} + \sqrt[3]{1.00005 - \sqrt{0.00040001}} \approx 1.9999444\cdots$$

Thus, the backward error

$$\Delta x = x_0 - x \approx 3.0$$

exists, but is very large. Figure 2 plots the exact $g(x)$ and the computed $fl(g(-1.0))$. Although the computed $\widehat{y} = fl(g(-1.0)) = 0.6667$ in this example is in the range of $g(x)$, it exceeds the local maximum $2/3$ at $x = -1$ and thus is mapped to an $x$ far from $-1$.

Although $\Delta x$ exists and is very large, this method is stable since it involves only multiplication, division, and subtraction. The computed result

$fl(g(-1.0))$ is obviously an accurate approximation of the exact $g(-1)$. This example shows that a large $\Delta x$ does not necessarily imply instability.

**Conclusion** We have shown that a stable accurate method may lead to large backward error. The above three examples show that in order for the backward error $\Delta x$ to exist and indicate the stability of the method, $g(x)$ must be invertible near the input $x$ and the inverse $g^{-1}(y)$ is defined in a neighbourhood of $y = g(x)$. Whenever the analysis gives a large backward error, the above conditions must be checked before we conclude that the method is unstable.

# References


[1] J.M. Demmel. *Applied Numerical Linear Algebra.* The Society for Industrial and Applied Mathematics, 1997.

[2] G.H. Golub and C.F. Van Loan. *Matrix Computations*, 3rd Edition. The Johns Hopkins University Press, 1996.

[3] N.J. Higham. *Accuracy and Stability of Numerical Algorithms.* The Society for Industrial and Applied Mathematics, 1996.

[4] G.W. Stewart. *Introduction to Matrix Computations.* Academic Press, Inc., 1973.

[5] A.M. Turing. Rounding-off errors in matrix process. *Quart. J. Mech. Appl. Math.*, 1:287–308, 1948.

[6] J.H. Wilkinson. *Rounding Errors in Algebraic Processes.* Notes on Applied Science No. 32, Her Majesty's Stationery Office, London, 1963. Also published by Prentice-Hall, Englewood Cliffs, NJ, USA.