Short Communications / Kurze Mitteilungen

Inner Product Rounding Error Analysis in the Presence of Underflow

A. Neumaier, Freiburg i. Br.

Received October 24, 1983; revised June 7, 1984

Abstract — Zusammenfassung

Inner Product Rounding Error Analysis in the Presence of Underflow. Wilkinson's classical error analysis for sums and inner products is extended to the case where underflow may occur. This is relevant for the construction of rigorous error bounds for an inner product evaluated on computers which do not give underflow messages. The analysis also covers calculations with gradual underflow.

AMS Subject Classification: 65 G05, 65 G10.

Key words: Error analysis, underflow, sums, inner products.


1. Introduction

In his now classical book on error analysis, Wilkinson [6] treats in detail the computation of inner products on a computer which performs all operations with bounded relative error. Real computers satisfy this requirement only if neither overflow nor underflow occurs. Whereas the danger of overflow is small, and usually interrupts the computation, the occurrence of underflow is more likely (e.g. in the computation of small residuals), and is usually not noticed by the user. Thus, for the design of a portable routine for the computation of inner products with rigorous error bounds, Wilkinson's error analysis has to be modified to handle underflow effects.

This is done in the present paper. The analysis given is general enough to cover fixpoint arithmetic, normalized floating point arithmetic, and floating point arithmetic with gradual underflow (the latter as described e.g. in Kahan and Palmer [1]). Moreover, emphasis is given to the problem of presenting the error bounds in such a form that they remain strict bounds even if their evaluation involves roundoff errors. The latter technique was introduced by Olver [4] in connection with absolute and relative precision; the treatment given here avoids the exponential expressions occurring in Olver's paper.
The paper is concluded with three portable algorithms (written in an informal programming language) performing three specific tasks:

**SUM** \((x, n, s)\) computes an upper bound \(s\) for a sum of nonnegative numbers \(x_i\) \((i = 1, \ldots, n)\):

\[
\sum_{i=1}^{n} x_i \leq s,
\]

with slight overestimation only.

**IPROD** \((A, B, c, i_1, j_2, k, r, e)\) computes an approximate inner product \(r\) with error bound \(e\):

\[
\left| c - \sum_{j=j_1}^{j_2} a_{ij} b_{jk} - r \right| \leq e,
\]

with small \(e\).

**DIV** \((a, b, c, e)\) computes an approximate quotient \(q\) and a small residual error bound \(e\) such that

\[
|a - c b| \leq e.
\]

These algorithms are applied in Neumaier [2] to the construction of a portable algorithm for the computation of a matrix inverse with rigorous, realistic, and componentwise error bounds with \(n^3 + O(n^2)\) multiplications.

### 2. Roundoff Error Control: Error Analysis

We begin with some remarks concerning computer arithmetic. Let \(M\) be the set of machine numbers used for our calculation on a given computer. We denote by \(\varphi\) the result of an arithmetic expression \(\varphi\) when evaluated on the computer, and make the following assumptions on \(\varphi\) and the basic arithmetical operations \(\circ \in \{+, -, \cdot, /\}\).

**A1**: There are small numbers \(\varepsilon, \eta\) such that for all \(a, b \in M\) for which the results \(\overline{a \circ b}\) is defined,

\[
\overline{a \circ b} = (a \circ b)(1 + \alpha) + \alpha', \ |\alpha| \leq \varepsilon, \ |\alpha'| \leq \eta, \ \alpha \alpha' = 0.
\]

The number \(\varepsilon\) is the relative precision and \(\eta\) the underflow threshold of the computer. We require that \(\eta\) (but not necessarily \(\varepsilon\)) is in \(M\).

**A2**: There is a large integer \(N\) such that if \(i, k\) are integers in the range \(-N \leq i, k \leq N\) then \(i, k \in M\) and

\[
i \circ k = (i \circ k)(1 + \alpha), \ \ |\alpha| \leq \varepsilon;
\]

moreover, \(\alpha = 0\) if \(i \circ k\) is also an integer in this range.

**A3**: If \(a \circ b \geq 0\) then \(\overline{a \circ b} \geq 0\).

If the computer works with normalized floating point numbers with basis \(B\), mantissa length \(L\), exponent range \([-E, F]\) with \(E, F \geq L\), and if the arithmetic registers have at least one guard digit then **A1** is satisfied with \(\varepsilon = B^{1-L}\) and