How and How Not to Sum Floating Point Numbers

Nick Higham
Department of Mathematics
University of Manchester

higham@ma.man.ac.uk

Book in preparation:
Accuracy and Stability of Numerical Algorithms
(SIAM, 1995)

THE UNIVERSITY
of MANCHESTER

Swedish Advice on Summation

Dahlquist and Björck, *Numerical Methods*, Prentice-Hall, 1974.

- Example of $\sum_{n=1}^{10,000} n^{-2}$. Reverse ordering much more accurate (error 650 times smaller) than natural ordering.

- "If the terms in a sum (of positive terms) are of varying orders of magnitude, one should add the terms in increasing order if one needs high accuracy in the result."

- Kahan's summation algorithm described (IFIP Congress, 1971): "gives a very small error bound in the sum . . . error bound is (essentially) independent of N".

Summation

$$S_n = \sum_{i=1}^{n} x_i$$

Recursive summation:

$s = 0$
for $i = 1{:}\, n$
    $s = s + x_i$
end

Increasing order: $|x_1| \le |x_2| \le \cdots \le |x_n|,$

Decreasing order: $|x_1| \ge |x_2| \ge \cdots \ge |x_n|.$

Easy to show that $E_n = S_n - \widehat{S}_n$ satisfies

$$|E_n| \le (|x_1| + |x_2|)\gamma_{n-1} + \sum_{i=3}^{n} |x_i|\gamma_{n-i+1},$$

where

$$\gamma_n := \frac{nu}{1 - nu}.$$

Two More Methods

Pairwise summation:

$x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7 \quad x_8$

Insertion method: (assume $|x_1| \leq \cdots \leq |x_n|$)

$x_1 \quad x_2 \quad x_3 \quad x_4 \quad x_5 \quad x_6 \quad x_7 \quad x_8$

General Algorithm

Algorithm for $S_n = \sum_{i=1}^{n} x_i$.

Let $\mathcal{S} = \{ x_1, \ldots, x_n \}$.
repeat while $\mathcal{S}$ contains more than one element
Remove two numbers x and y from $\mathcal{S}$,
and add their sum $x + y$ to $\mathcal{S}$.
end
Assign the remaining element of $\mathcal{S}$ to S_n.

Error Analysis

Express ith execution of the repeat loop as
$T_i = x_{i_1} + y_{i_1}$. The computed sums satisfy

$$\widehat{T}_i = \frac{x_{i_1} + y_{i_1}}{1 + \delta_i}, \quad |\delta_i| \leq u, \quad i = 1\colon n - 1.$$

Local error is

$$\delta_i \widehat{T}_i.$$

Error Bound

Total error is sum of local errors:

$$E_n := S_n - \widehat{S}_n = \sum_{i=1}^{n-1} \delta_i \widehat{T}_i.$$

Smallest possible error bound:

$$|E_n| \leq u \sum_{i=1}^{n-1} |\widehat{T}_i|.$$

Since $|\widehat{T}_i| \leq \sum_{j=1}^{n} |x_j| + O(u)$, have also the weaker bound

$$|E_n| \leq (n-1)u \sum_{i=1}^{n} |x_i| + O(u^2).$$

> In designing or choosing a summation method to achieve high accuracy, the aim should be to minimize the absolute values of the intermediate sums T_i.

Application

Recursive Summation

- Here,

$$T_{i-1} = S_i := \sum_{j=1}^{i} x_j.$$

  Ideally, choose ordering to minimize $\sum_{i=2}^{n} |\widehat{S}_i|$.

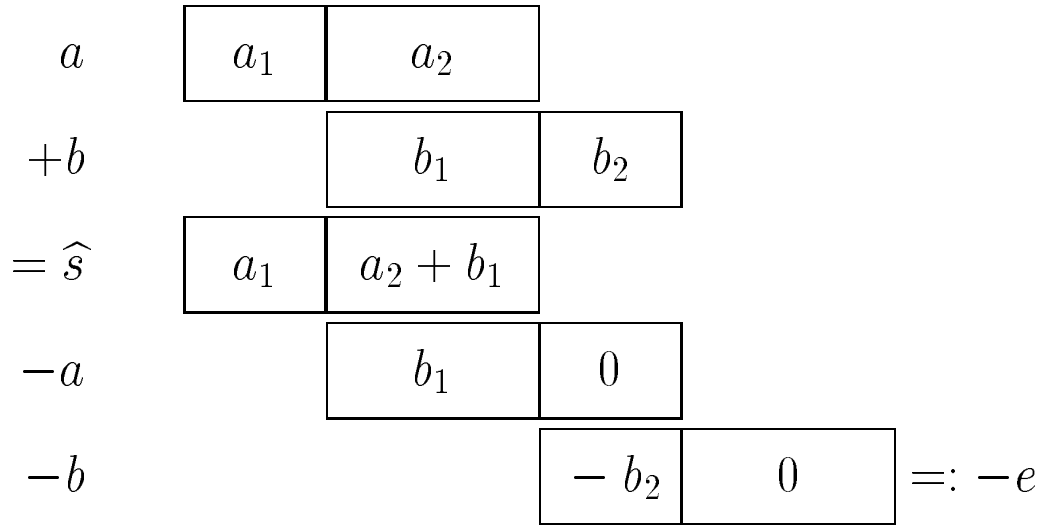
- Compromise (Psum): minimize, in turn, $|x_1|$, $|\widehat{S}_2|, \ldots, |\widehat{S}_{n-1}|$. ($O(n \log n)$ comparisons.)
- Further compromise: increasing ordering.
- If $x_i \geq 0$, increasing ordering is optimal.

"±" method

- Compute sum of the positive numbers, S_+, and the nonpositive numbers, S_-, separately, then form $S_n = S_- + S_+$.
- Maximizes $\max_i |T_i|$.

Recovering the Rounding Error

Let a and b be floating point numbers with $|a| \geq |b|$, let $\widehat{s} = fl(a+b)$. Consider



Suggests that if we evaluate

$$e = -\big[\big((a+b) - a\big) - b\big] = (a - \widehat{s}) + b$$

then $\widehat{e} \approx (a+b) - \widehat{s}$. In fact, for rounded floating point arithmetic in base 2,

$$a + b = \widehat{s} + \widehat{e},$$

Compensated Summation

Compensated summation for $\sum_{i=1}^{n} x_i$, Kahan (1965):

$s = 0;\ e = 0$
for $i = 1{:}\,n$
    $temp = s$
    $y = x_i + e$
    $s = temp + y$ % "$s = s + x_i$"
    $e = (temp - s) + y$
end

Knuth (1981) shows that, *with a guard digit in addition*,

$$\widehat{S}_n = \sum_{i=1}^{n} (1 + \mu_i) x_i, \qquad |\mu_i| \leq 2u + O(nu^2),$$

so that

$$|E_n| \leq \left(2u + O(nu^2)\right) \sum_{i=1}^{n} |x_i|.$$

Application: Euler's Method

$$y' = f(x, y), \qquad y(a) \text{ given.}$$

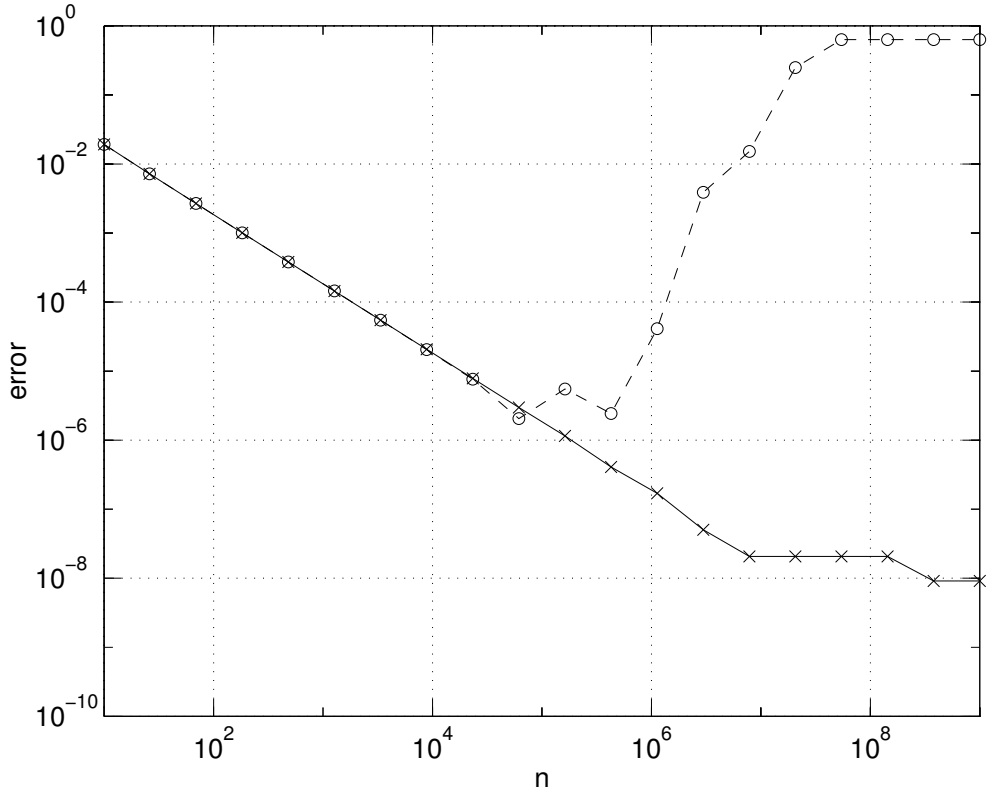
Euler: $y_{n+1} = y_n + hf_n$, $y_0 = y(a)$.

$\% \; x = x + h$
$dx = h + cx$
$new\_x = x + dx$
$cx = (x - new\_x) + dx$ $\%$ Parentheses vital!
$x = new\_x$

$\% \; y = y + h * f(x, y)$
$dy = h * f(x, y) + cy$
$new\_y = y + dy$
$cy = (y - new\_y) + dy$ $\%$ Ditto.
$y = new\_y$

Euler Example

Solved $y' = -y$ with $y(0) = 1$ over $[0, 1]$ using n steps of Euler's method. Single precision in NAG FTN90 ($u \approx 6 \times 10^{-8}$).



Summary

- ▷ A very simple error analysis is applicable to a general class of summation algorithms.
- ▷ For recursive summation we want the partial sums to be small. Equivalent to increasing ordering *only* if x_i all of one sign.
- ▷ Compensated summation fully exploits properties of floating point arithmetic to recover the error in additions and feed them back into the summation process.

  Works well and deserves to be better known!