# Unit 5: Mathematical considerations for mixed-precision

In this unit, we will introduce the computational mathematics terminology needed to understand the properties of algorithms. These properties can then be used to help determine whether the use of reduced precision is advisable. Most of the common algorithms have already been analysed in literature and, once the terminology is understood, you will be able to use the literature to understand the underlying properties of the specific method under consideration.

## Accuracy versus precision

Accuracy refers to the absolute or relative error of an approximate quantity. Let $\hat{x}$ be an approximation to a real number $x$, and $|a|$ be the absolute value of $a$.

Definition The *absolute error* is defined as

$$e_{abs}(\hat{x}) = |x - \hat{x}|$$

and the *relative error* as

$$e_{rel}(\hat{x}) = \frac{|x - \hat{x}|}{|x|}.$$

If $x$ and $\hat{x}$ are vectors, we have

$$e_{abs}(\hat{x}) = \|x - \hat{x}\|$$

and the *relative error* is

$$e_{rel}(\hat{x}) = \frac{\|x - \hat{x}\|}{\|x\|},$$

where $\|\cdot\|$ is a vector norm. Commonly used vector norms are

$$\|x\|_\infty := \max_i |x_i|,$$

$$\|x\|_1 := \sum_i |x_i|$$

and

$$\|x\|_2 := \sqrt{\sum_i x_i^2}.$$

The component-wise relative error

$$\max_i \frac{|x_i - \hat{x}_i|}{|x_i|}$$

is widely used in error analysis and perturbation analysis.

When considering whether to use mixed-precision components within your code, if is important to take a number of factors into account:

- Rounding;
- Data uncertainty;
- Truncation (discretization).

Truncation errors relate to the specific numerical methods being used and most of the popular methods will be well-understood with respect to these methods.

It is important to remember that precision is the accuracy for which the basic arithmetic operations are performed: the terms precision and accuracy should not be confused.

## Backward and forward errors

Let $y = f(x)$, where $f$ is a real scalar function. Suppose that an approximation to $y$, $\hat{y}$, is computed. A good question to ask is: how can we measure the quality of $\hat{y}$?

Normally, we would be happy for $e_{rel}(\hat{y})$ to be tiny but this is not always achievable. Instead, we can focus on the backward error and the question: for what set of $\Delta x$ do we have $\hat{y} = f(x + \Delta x)$? In general, $\Delta x$ may not be unique and, hence, we wish to find the smallest such $\Delta x$.

Definition The *backward error* is defined to be the value of $|\Delta x|$ (or $\min|\Delta x|$), possibly divided by $|x|$.

Definition The *forward error* is defined to be either $e_{abs}(\hat{y})$ or $e_{rel}(\hat{y})$.

*Backward error analysis* is the process of bounding the backward error of a computed solution. Backward error analysis allows the computational scientist to interpret the rounding errors as being equivalent to perturbations in the data and, additionally, reduces the estimation or bounding of the forward error to perturbation theory (which is, generally, well understood). Since data frequently contains uncertainties from previous computations or errors from storing numbers on the computers, if the backward error is no larger than these certainties, then it is difficult to overly criticise the computed solution.

Definition A method is called *backward stable* if, for any $x$, it produces a $\hat{y}$ with a small backward error, where "small" will depend on the context of the situation.

Definition An algorithm is called *numerically stable* if it is stable in the *mixed forward-backward error* sense:

$$\hat{y} + \Delta y = f(x + \Delta x), \text{ where } |\Delta y| \leq \epsilon |y| \text{ and } |\Delta x| \leq \mu |x|$$

provided $\epsilon$ and $\mu$ are sufficiently small.

Hence, a backward stable algorithm is numerically stable.

Definition A function $g(x)$ is $O(x^p)$ if and only if there exists a positive real number $d$ and real number $x_0$ such that $|g(x)| \leq dx^p$ for all $x \geq x_0$.

Definition A function is twice continuously differentiable if the function, its derivative and its second derivative are all continuous functions.

If $f(x)$ is twice continuously differentiable, then

$$\frac{\hat{y}-y}{y} = \left(\frac{xf'(x)}{f(x)}\right)\frac{\Delta x}{x} + O((\Delta x)^2).$$

Definition The quantity

$$c(x) = \left|\frac{xf'(x)}{f(x)}\right|$$

is called the *condition number* of $f$.

For small $\Delta x$, the relative change in the output is closely approximated by the relative change in $x$ multiplied by $c(x)$.

---

Example Let $f(x) = e^x$, then $c(x) = |x|$. Hence, for large values of $x$, a small relative change in $x$ can produce a much larger relative change in $e^x$. For example, if $x = 1000$ and $\Delta x = 0.1$, then

$$\frac{\Delta x}{x} = 10^{-4} \text{ and } \frac{\hat{y}-y}{y} = \frac{e^{1000.1} - e^{1000}}{e^{1000}} = e^{0.1} - 1 = 0.10517.$$

This satisfies

$$\frac{\hat{y}-y}{y} \approx \left(\frac{xf'(x)}{f(x)}\right)\frac{\Delta x}{x}.$$

---

Note: If $x$ is a vector of length $n$ and the output of $f(x)$ is a vector of length $m$, the condition number is similarly defined using norms:

$$c(x) = \frac{\|J(x)\|\|x\|}{\|f(x)\|},$$

where $J(x)$ is the Jacobian matrix of $f(x)$:

$$J(x) = \begin{bmatrix} \frac{\delta f_1}{\delta x_1} & \cdots & \frac{\delta f_1}{\delta x_n} \\ \vdots & \ddots & \vdots \\ \frac{\delta f_m}{\delta x_1} & \cdots & \frac{\delta f_m}{\delta x_n} \end{bmatrix} \text{ and } f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{bmatrix}.$$

It is assumed that the matrix is induced from the vector norm:

$$\|J\| = \sup_{v\ (v\neq 0)} \frac{\|Jv\|}{\|v\|}.$$

## Case Study: Solution of linear systems $Ay = b$

A typical problem in the scientific applications is the solution of linear systems $Ay = b$, where $A$ and $b$ are given and we wish to calculate $y$. $A$ is assumed to be real $n \times n$ matrix and non-singular (i.e., for each $b$ there exists a unique $y$) and $b$ is assumed to be a real vector of length $n$.

The Gaussian Elimination Method can be used to solve such problems. It reduces the problem to an equivalent matrix problem (in exact arithmetic): solve

$$A^{(n+1)}y = b^{n+1},$$

where $A^{(n+1)}$ is upper triangular (all entries below the diagonal are zero). At the beginning of the $k^{th}$ stage of the Gaussian elimination method, the system has been converted to $A^{(k)}y = b^{(k)}$, where

$$A^{(k)} = \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ 0 & A_{22}^{(k)} \end{bmatrix}$$

and $A_{11}^{(k)}$ is upper triangular. The $k^{th}$ stage of the Gaussian elimination method aims to create zeros below the diagonal in the $k^{th}$ column of $A^{(k)}$. This is done by the following operations:

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{ik}a_{kj}^{(k)}, i = k+1:n, j = k+1:n,$$

$$b_i^{(k+1)} = b_i^{(k)} - l_{ik}b_k^{(k)}, i = k+1:n,$$

where $l_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}},\ i = k+1:n$. Gaussian elimination requires $\frac{2}{3}n^3$ flops. If $a_{kk}^{(k)} = 0$ for some $k$, then the method will breakdown. Even if the method does not encounter $a_{kk}^{(k)} = 0$, when working in finite precision, if some multiplier $l_{ik}$ is large, then there is possible loss in accuracy when performing $a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{ik}a_{kj}^{(k)}$: this would correspond to making a relatively large change to $A$. In an attempt to overcome these problems, partial pivoting may be used. Namely, at the start of the $k^{th}$ stage, the $k^{th}$ row of $A^{(k)}$ and $b^{(k)}$ is interchanged with row $r$, where

$$\left|a_{rk}^{(k)}\right| := \max_{k \le i \le n} \left|a_{ik}^{(k)}\right|.$$

Thus, $l_{ik} \le 1$ for $i = k+1:n$. The following theorem provides a result about the numerical stability of the method.

Theorem Let $A$ be a real, non-singular, $n \times n$ matrix and suppose that Gaussian elimination with partial pivoting produces a computed solution $\hat{y}$ to $Ay = b$. Then

$$(A + \Delta A)\hat{y} = b,$$

where

$$\|\Delta A\| \le 2n^2\gamma_n\rho_n\|A\|_\infty,$$

$$\gamma_n = \frac{nu}{1-nu},$$

$$\rho_n = \frac{\max_{ijk}\left|a_{ij}^{(k)}\right|}{\max_{ij}|a_{ij}|},$$

and $u$ is the unit roundoff (Unit 2).

From this theorem, we can see that the floating-point precision will affect the upper bound on the value of $\|\Delta A\|$, along with the properties of $A$. Further bounds on $\rho_n$ are also available along with other theorems relating the stability of Gaussian Elimination-based methods to the properties of $A$ and $u$, see Chapters 7-11 of [1] for a good summary.

Also, see [1] for stability results for a wealth of other methods.

[1] "Accuracy and Stability of Numerical Algorithms (Second Edition)", *N. J. Higham*, SIAM, 2002.