Stability

- An algorithm may be *unstable*. A problem may be *ill-conditioned*.
- Why is stability an important issue in the design of an algorithm?
  - ◇ Since a computer can only represent finitely many numbers, there is a good chance that most real numbers will have to be rounded and, hence, carry errors.
  - ◇ Also, the law of arithmetic operations generally is *not* satisfied. For example, there exists a positive number ϵ such that, in the floating-point arithmetic, $1 + \epsilon = 1$.
  - ◇ These errors can be magnified or propagated through the sequence of executions required by an algorithm and eventually corrupt the desirable results.
- Two mathematically equivalent algorithms may give rise to two totally different numerical answers as the result of unstability.
  - ◇ Consider a machine with $\beta = 10$, $t = 5$, and unlimited digits for the exponents. Observe the two way of evaluating $e^{-5.5}$:

$$\begin{aligned} e^{-5.5} &= 1.000 - 5.5000 + 15.125 - 27.730 + 38.129 - 41.942 \\ &\quad +38.446 - 30.208 + \ldots = 0.00263363; \\ e^{-5.5} &= \frac{1}{e^{5.5}} = \frac{1}{1+5.5+15.125+27.730+\ldots} = 0.0040865, \end{aligned}$$

    whereas the true value should be $e^{-5.5} \approx 0.00408677$.

    - ▷ The wrongness in the first calculation originates from, for example, terms like 38.129 already have roundoff effor which is nearly as large as the final result. (38.12760417)
    - ▷ The second way is no better. The denomerator is just as bad the first way of calculation. However, the savage somehow is remedied by the division.

- A numerical method is said to be unstable if the roundoff errors introduced at one stage of the computation propagate with increasing magnitude in later stages.
- An example of efficient but unstable algorithem:
    - ◇ Suppose a machine with $\beta = 10$ and $t = 6$ is used to evaluate $E_n = \int_0^1 x^n e^{x-1}\,dx$, $n = 1, 2, \ldots$.
    - ◇ It appears the recursive formula

      $$E_n = 1 - nE_{n-1}$$

      obtained by integration by parts is an easy enough algorithm, if we know $E_1 = \frac{1}{e}$.
    - ◇ The results are

      $$\begin{array}{llll} E_1 & \approx 0.367879, & E_6 & \approx 0.127120, \\ E_2 & \approx 0.264242, & E_7 & \approx 0.110160, \\ E_3 & \approx 0.207274, & E_8 & \approx 0.118720, \\ E_4 & \approx 0.170904, & E_9 & \approx -0.0684800. \\ E_5 & \approx 0.145480, & & \end{array}$$

      and E_9 obviously is wrong. (Why?)
    - ◇ Note that the error in E_1 is magnified by a factor of $9! = 362880$. Suppose the initial error is $\approx .441 \times 10^{-6}$. Then the error in E_9 should be $\approx .1601$ which is even greater than the true value of $E_9 \approx 0.0916$.
- Generally speaking, any newly designed algorithm needs to pass the stability analysis first. Not always we can develop a stable algorithm, but unstable algorithms should clearly caution users about the possible breakdown.