Lecture 3, Sep 15, 2023

Numerical Methods and Stability

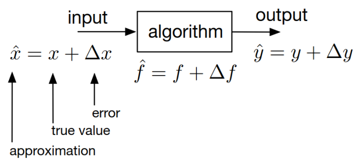

Figure 1: Model of errors in a numerical algorithm.

- Any numerical system will accumulate errors in a variety of ways; how can we quantify and evaluate these errors?
- Some common sources of error are:
  - Rounding/truncation errors, due to finite precision
  - Approximate numerical algorithms, in which we simplify complex models for efficiency
  - Input error, where the inputs to our algorithm are the outputs of an upstream algorithm which itself has errors
  - Modelling errors, where the model itself is a simplified representation of the real world (e.g. discretization)
- These errors can be categorized into two general sources: input error (Δx) and algorithm/approximation error (Δf) which combine to result in an output error (Δy)
- *Absolute* errors are simply the absolute value of the error, $|\Delta x|$; *relative* errors are the absolute errors divided by the parameter, $\delta \frac{|\Delta x|}{x}$
  - If the true values of x, f, y are not known, we cannot compute the relative error and might have to settle for an upper bound instead

Well- and Ill-Conditioned Problems

- First we will consider what happens when we have an ideal algorithm with some input error
- Intuitively, a problem is *well-conditioned* if, assuming an ideal algorithm ($\Delta f = 0$), the input error does not grow when propagated through the algorithm, i.e. $\Delta \bar{y} < \Delta x$
  - Consider the Taylor expansion: $f(x + \Delta x) = f(x) + \frac{\mathrm{d}f}{\mathrm{d}x}\Delta x + O(\Delta x^2) = y + \Delta \bar{y}$
  - $\frac{\Delta \bar{y}}{y} = \frac{1}{y}\left(f(x) + \frac{\mathrm{d}f}{\mathrm{d}x} \cdot x\frac{\Delta x}{x} + O(\Delta x^2)\right) = \frac{\mathrm{d}f}{\mathrm{d}x}\frac{x}{f(x)} \cdot \frac{\Delta x}{x} + O(\Delta x^2) = K_x \delta x + O(\Delta x^2)$
  - If $|\delta x| \leq \varepsilon$ then $|\delta y| \approx |K_x \delta x| \leq |K_x|\varepsilon$, so given a bound on δx we can find a bound on δy

Definition

The *absolute condition number* is defined as

$$\mathrm{Cond}_x = \left|\frac{\Delta\bar{y}}{\Delta x}\right| \approx \left|\frac{\mathrm{d}f}{\mathrm{d}x}\right|$$

The *relative condition number* is defined as

$$\mathrm{cond}_x = \left|\frac{\delta\bar{y}}{\delta x}\right| \approx |K_x| = \left|\frac{\mathrm{d}f}{\mathrm{d}x} \cdot \frac{x}{f(x)}\right|$$

(note, multiplication not differentiation)

A problem is *well-conditioned*:

1. If and only if Cond_x is small (using absolute error)
2. If and only if $\mathrm{cond}_x \leq 1$ (using relative error)

- We typically want the absolute condition number to be small (but how small depends on the problem), and we want the relative condition number to be less than 1; so most of the time the relative condition number is used since it is easier to interpret
- Conditioning is a property of the *problem*, not a particular algorithm (since we assumed a perfect algorithm to begin with)
- Example: linear function: $y = ax$
  - $\frac{\mathrm{d}f}{\mathrm{d}x} = a \implies K_x = \frac{\mathrm{d}x}{\mathrm{d}f} = a\frac{x}{ax} = 1$
  - The condition number is 1, so the relative error stays the same and the problem is well-conditioned
  - The absolute error is smaller than the absolute input error if $\left|\frac{\mathrm{d}f}{\mathrm{d}x}\right| = |a| < 1$
    - Intuitively, for a steeper function the error will get bigger, but for a smaller slope the error is smaller
- Example: linear equation: find y such that $ay + b = 0$ where b is the input and a is fixed
  - $y = -\frac{b}{a} = f(b)$
  - $\frac{\mathrm{d}f}{\mathrm{d}b} = -\frac{1}{a} \implies K_b = \frac{\mathrm{d}f}{\mathrm{d}b} \cdot \frac{1}{f(b)} b = -\frac{1}{a} \cdot -\frac{a}{b} \cdot b = 1$
  - This is another well-conditioned problem
  - $\left|\frac{\mathrm{d}f}{\mathrm{d}b}\right| = \left|\frac{1}{a}\right|$ so for small a, the absolute condition number is large
- Example: differential equation: find y such that $\dot{y} = (\lambda + \Delta\lambda)y, y(0) = y_0$ where λ is the input
  - $\hat{y}(t) = y_0 e^{(\lambda+\Delta\lambda)t} = y_0 e^{\lambda t} e^{\Delta\lambda t} \approx y_0 e^{\lambda t} + y_0 t e^{\lambda t} \cdot \Delta\lambda = y + \Delta\bar{y}$
    - Note we used $e^x \approx 1 + x$
  - The absolute condition number is $\frac{\Delta\bar{y}}{\Delta\lambda} = y_0 t e^{\lambda t}$
  - $\lim_{t\to\infty}\left|\frac{\Delta\bar{y}}{\Delta\lambda}\right| = \begin{cases} 0 & \lambda < 0 \\ \infty & \lambda \geq 0 \end{cases}$
  - This shows that asymptotically stable differential equations (DEs that approach some fixed value) are well-conditioned
    - Exercise: what we think of the error as being on the initial condition? $\dot{y} = \lambda y, y(0) = y_0 + \Delta y_0$
- Example: root finding: find y such that $g(y) = 0$
  - y is the output, but the input is hard to define since it is a function
  - We can think of it as $g(y) = 0, g(y + \Delta\bar{y}) + \varepsilon = 0$ so ε is an "input" representing an additive error in g (see diagram above); if we shift g up or down, the location of the root y will also change by an amount

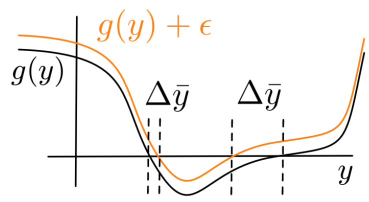

Figure 2: Representation of error in root finding.

- Taylor series expand: $g(y)+g'(y)\Delta\bar{y}+\varepsilon = g'(y)\Delta\bar{y}+\varepsilon \approx 0 \implies \left|\frac{\Delta\bar{y}}{\varepsilon}\right| \approx \frac{1}{|g'(y)|}$
- We find that the absolute condition number is inversely proportional to slope; notice that for the same shift, the left root with a larger slope is shifted a lot less

Order and Consistency

- We shall now consider what happens when we have an algorithmic error, while the input is ideal

Definition

Let $\varphi(x,\Delta)$ represent some approximate algorithm that models $f(x)$, where Δ are the parameters of the algorithm; φ's accuracy is of *order* p (alternatively, φ is $O(\Delta^p)$) if

$$\tilde{y} = \varphi(x,\Delta) - f(x) \propto \Delta^p$$

φ is *consistent* if

$$\lim_{\Delta\to 0}\varphi(x,\Delta) = f(x)$$

i.e. the approximate algorithm φ approaches the real model as the parameter approaches zero.

- The parameters can be e.g. a step size for numerical ODE solving; note that we assume that Δ is typically smaller than 1
- Example: numerical differentiation: $y = f'(x), \varphi(x,\Delta) = \frac{f(x+\Delta)-f(x)}{\Delta}$
  - Taylor expansion: $\varphi(x,\Delta) = \frac{f(x)+f'(x)\Delta+f''(x)\frac{\Delta^2}{2}+\cdots-f(x)}{\Delta} = f'(x)+f''(x)\frac{\Delta}{2}+\cdots$
  - Therefore $\varphi(x,\Delta)-f'(x) = f''(x)\frac{\Delta}{2}+f'''(x)\frac{\Delta^2}{6}+\cdots$ so φ is $O(\Delta)$, i.e. order 1

Stability and Convergence

- If we have both an input error and an approximate algorithm, the errors compound
- $\Delta y = \varphi(x+\Delta x,\Delta)-f(x) = \underbrace{(\varphi(x+\Delta x,\Delta)-\varphi(x,\Delta))}_{\Delta\tilde{y}} + \underbrace{(\varphi(x,\Delta)-f(x))}_{\tilde{y}} = \Delta\tilde{y}+\tilde{y}$
  - $\Delta\tilde{y}$ is the result of our input error, and $\tilde{y}$ is the result of our approximate algorithm
- The propagated error is $\Delta\tilde{y} = \varphi(x+\Delta x,\Delta)-\varphi(x,\Delta) \approx \frac{\mathrm{d}\varphi}{\mathrm{d}x}\cdot\Delta x$

Definition

φ is *numerically stable* if the ratio

$$\left|\frac{\Delta\tilde{y}}{\Delta x}\right| = \left|\frac{\mathrm{d}\varphi}{\mathrm{d}x}\right| < 1$$

If this ratio is greater than 1, then φ is *unstable*; if the ratio is exactly 1, then φ is *marginally stable*.

- The idea is that if you iteratively apply the algorithm, each time the error will be multiplied by this ratio; therefore a numerically stable algorithm will decrease in error, but an unstable algorithm will increase

Theorem

The numerical solution $\hat{y} = \varphi(\hat{x}, \Delta)$ with input $\hat{x} = x + \Delta x$ *converges* for $\Delta \to 0$ towards the exact solution $y = f(x)$ if

1. $\varphi(x, \Delta)$ is consistent, i.e. $\lim\limits_{\Delta \to 0} \varphi(x, \Delta) = f(x)$
2. $\varphi(x, \Delta)$ is at least marginally stable for $\Delta \to 0$, i.e. $\lim\limits_{\Delta \to 0} \left| \frac{\mathrm{d}\varphi}{\mathrm{d}x} \right| \leq 1$