GIẢI TÍCH SỐ
NUMERICAL ANALYSIS
CHƯƠNG 1: SAI SỐ

Giảng viên: TS. Hà Phi

NỘI DUNG MÔN HỌC

- Chương 1: Giới thiệu về sai số
- Chương 2: Giải số hệ phương trình tuyến tính = các phương pháp trực tiếp
- Chương 3: Giải số phương trình phi tuyến
- Chương 4: Bài toán xấp xỉ hàm số
- Chương 5: Tính gần đúng đạo hàm và tích phân
- ***Chương 6: Giải số các phương trình vi phân***
- ***Chương 7: Giải số hệ phương trình tuyến tính = các phương pháp gián tiếp. Mối liên hệ với "Giải số các phương trình đạo hàm riêng".***

Chương 1: SAI SỐ

◦ Nguồn gốc của sai số: 4 nguồn gốc

Ví dụ 1: Tính y = e^{sqrt{2}} sử dụng khai triển MacLaurin đến cấp 10 tại $x_0 = \sqrt{2}$

$$e^x = 1 + x + \frac{x^2}{2!} + ... + \frac{x^{10}}{10!} + R_n(x)$$

Sai số do mô hình toán học

Sai số dữ liệu đầu vào của x_0

Sai số thuật toán là $R_n(x)$

Sai số tính toán (do làm tròn số)

Vai trò của Sai số tính toán (làm tròn số)

Ta xét dãy số sau
Ta muốn đi tính x_{100}

$$x_{k+1} = \begin{cases} 2x_k, & x_k \in [0, \frac{1}{2}], \\ 2x_k - 1, & x_k \in (\frac{1}{2}, 1]. \end{cases}$$

Let $x_0 = 1/10$. Then $x_1 = 2/10$, $x_2 = 4/10$, $x_3 = 8/10$, $x_4 = 6/10$, and $x_5 = x_1$.

Ta tính toán sử dụng 5 chữ số thập phân.
Hãy xem kết quả tính toán số bằng lập trình có chính xác không?

Vì sao kết quả này lại sai?

k	True x_k	Computed x_k
0	0.10000	0.10000
1	0.20000	0.20000
2	0.40000	0.40000
3	0.80000	0.80000
4	0.60000	0.60000
5	0.20000	0.20000
10	0.40000	0.40000
20	0.60000	0.60000
40	0.60000	0.60001
42	0.40000	0.40002
44	0.60000	0.60010
50	0.40000	0.40625
54	0.40000	0.50000
55	0.80000	1.00000

Tại sao IBM lỗ hơn 420 triệu đô la năm 1994 vì dốt giải tích số? (Chap. 5, Greenbaum/ Chartier)

Vào mùa hè năm 1994, Intel đã dự đoán thành công thương mại của chip Pentium mới của mình. Con chip mới có tốc độ phân chia nhanh gấp đôi so với các chip Intel trước đây chạy cùng tốc độ xung nhịp. Đồng thời, Giáo sư Thomas R. Nicely, một nhà toán học tại Đại học Lynchburg ở Virginia, đã tính toán tổng nghịch đảo của các số nguyên tố bằng máy tính có chip Pentium mới. Các kết quả tính toán và lý thuyết khác nhau đáng kể. Tuy nhiên, kết quả chạy trên máy tính sử dụng CPU 486 cũ hơn đã tính ra kết quả chính xác. Đúng lúc, Nicely đã theo dõi lỗi trên chip Intel. Sau khi liên hệ với Intel và nhận được ít phản hồi cho các truy vấn ban đầu của mình, Nicely đã đăng một thông báo chung trên Internet yêu cầu những người khác xác nhận phát hiện của mình. Bài đăng (ngày 30 tháng 10 năm 1994) với dòng tiêu đề Lỗi trong Pentium FPU [78] bắt đầu:

"Có vẻ như có một lỗi trong đơn vị dấu chấm động (bộ xử lý số) của nhiều, và có lẽ là tất cả, bộ xử lý Pentium".

Figure 5.1. A bug in the Pentium chip cost Intel millions of dollars. (Courtesy of CPU Collection Konstantin Lanzet.)

Tại sao IBM lỗ hơn 420 triệu đô la năm 1994 vì dốt giải tích số? (Chap. 5, Greenbaum/ Chartier)

Email này đã bắt đầu một hoạt động sôi nổi, đến nỗi chỉ vài tuần sau vào ngày 13 tháng 12, IBM đã tạm dừng giao hàng các máy Pentium của họ và vào cuối tháng 12, Intel đã đồng ý thay thế tất cả các chip Pentium bị lỗi theo yêu cầu. Công ty đã dành ra một khoản dự phòng 420 triệu đô la để trang trải chi phí, một khoản đầu tư lớn cho một sai sót. Với một loạt các hoạt động Internet từ ngày 29 tháng 11 đến ngày 11 tháng 12, Intel đã trở thành trò cười trong trò đùa trên Internet, nhưng đối với Intel thì điều đó không vui chút nào. Vào thứ Sáu, ngày 16 tháng 12, cổ phiếu Intel đóng cửa ở mức 59,50 đô la, giảm 3,25 đô la trong tuần.

Figure 5.1. A bug in the Pentium chip cost Intel millions of dollars. (Courtesy of CPU Collection Konstantin Lanzet.)

What type of error could the chip make in its arithmetic? The *New York Times* printed the following example of the Pentium bug: Let $A = 4{,}195{,}835.0$ and $B = 3{,}145{,}727.0$, and consider the quantity

$$A - (A/B) * B.$$

In exact arithmetic, of course, this would be 0, but the Pentium computed 256, because the quotient A/B was accurate to only about 5 decimal places. Is this *close enough*? For many applications it probably is, but we will see in later sections that we need to be able to count on computers to do better than this.

While such an example can make one wonder how Intel missed such an error, it should be noted that subsequent analysis confirmed the subtlety of the mistake. Alan Edelman, professor of mathematics at Massachusetts Institute of Technology, writes in his article of 1997 published in *SIAM Review* [37]:

> We also wish to emphasize that, despite the jokes, the bug is far more subtle than many people realize. . . . The bug in the Pentium was an easy mistake to make, and a difficult one to catch.

Tại sao năm 1996 Viện hàng không châu Âu (European Space Agency) lỗ đến 7 tỉ đô la và mất 10 năm công sức vì 0 học giải tích số cẩn thận?

https://youtu.be/5tJPXYA0Nec

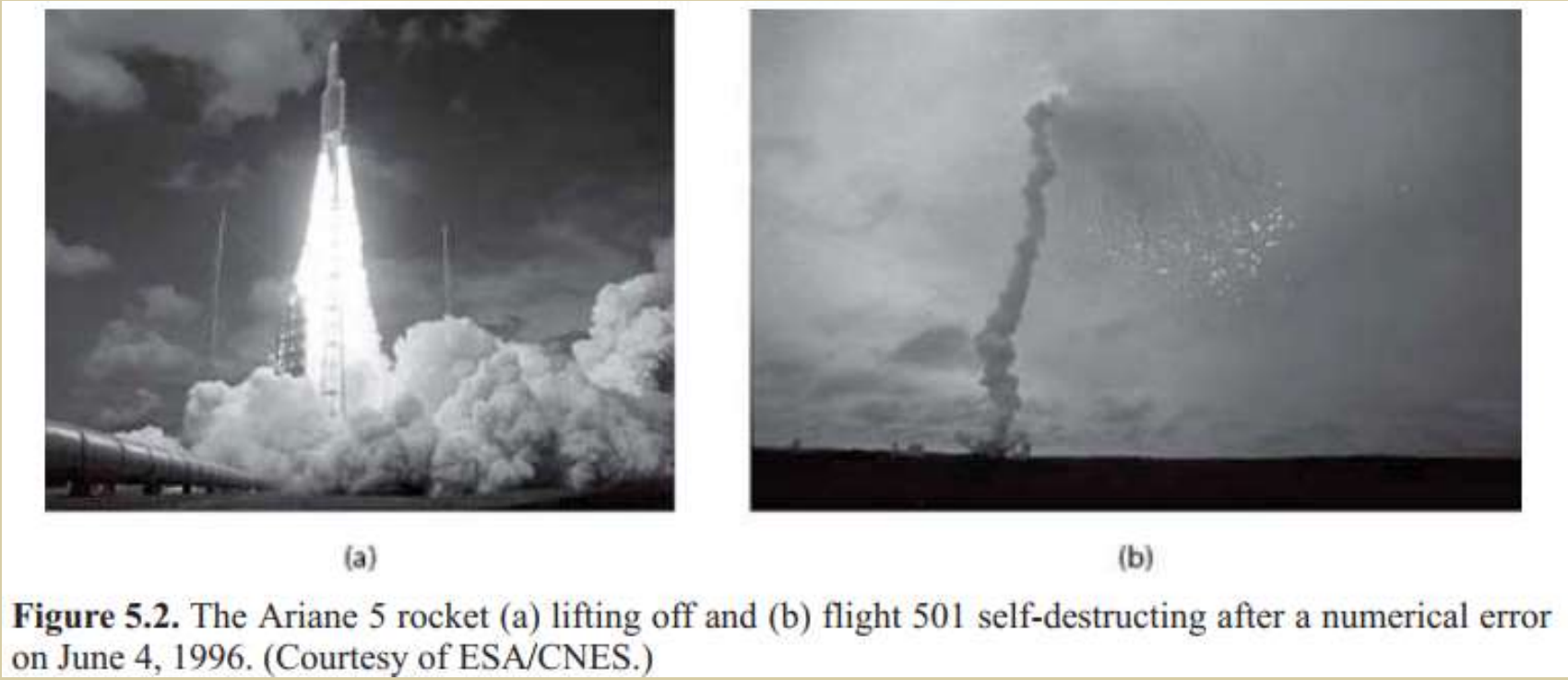

Figure 5.2. The Ariane 5 rocket (a) lifting off and (b) flight 501 self-destructing after a numerical error on June 4, 1996. (Courtesy of ESA/CNES.)

Ariane 5 Disaster. [4, 45] The Ariane 5, a giant rocket capable of sending a pair of three-ton satellites into orbit with each launch, took 10 years and 7 billion dollars for the European Space Agency to build. Its maiden launch was met with eager anticipation as the rocket was intended to propel Europe far into the lead in the commercial space business.

On June 4, 1996, the unmanned rocket took off cleanly but veered off course and exploded in just under 40 seconds after liftoff. Why? To answer this question, we must step back into the programming of the onboard computers.

During the design of an earlier rocket, programmers decided to implement an additional "feature" that would leave the horizontal positioning function (designed for positioning the rocket on the ground) running after the count-down had started, anticipating the possibility of a delayed takeoff. Since the expected deviation while on the ground was minimal, only a small amount of memory (16 bits) was allocated to the storage of this information. After the launch, however, the horizontal deviation was large enough that the number could not be stored correctly with 16 bits, resulting in an exception error. This

Millions of dollars would have been saved if the data had simply been saved in a larger variable rather than the 16-bit memory location allocated in the program.

We wish to emphasize that, the bug is far more subtle than many people realize.

(a)

(b)

Figure 5.2. The Ariane 5 rocket (a) lifting off and (b) flight 501 self-destructing after a numerical error on June 4, 1996. (Courtesy of ESA/CNES.)

Biểu diễn dấu chấm động/part 1
floating point representation

- Biểu diễn nhị phân (binary) $27 = 1\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 \Rightarrow 27 = \overline{11011}_2$
- Biểu diễn dấu chấm tĩnh (fixed point representation). Ví dụ 32 bit

$$n = \pm \left(\overline{x_{16}x_{15}...x_1.y_1y_2...y_{15}} \right)_2 , \quad x_i = 0, 1, y_i = 0, 1.$$

Khi đó số dương lớn nhất & số dương nhỏ nhất (số 0 máy/epsilon machine) là

$$n_{\max} = \left(\overline{\underbrace{11...1}_{16\ digits}.\underbrace{11...1}_{15 digits}} \right)_2 = 1\times 2^{15} + 1\times 2^{14} + ... + 1\times 2^{-13} + 1\times 2^{-13} \approx 2^{16}$$

$$eps = n_{\min} = \left(\overline{\underbrace{00...0}_{16\ digits}.\underbrace{0...01}_{15 digits}} \right)_2 = 2^{-15}$$

- Hạn chế của cách biểu diễn này là phạm vi biểu diễn quá nhỏ.

Biểu diễn dấu chấm động/part 2
floating point representation

- Biểu diễn dấu chấm động: $n = \pm\, m \cdot 2^E,\ 1 \le m < 2.$

  Ta gọi m là phần định trị (mantissa) và E là số mũ (Exponent)
- Biểu diễn bộ cộng đơn (single precision): 32 bits = 1 bit cho dấu (0 cho +, 1 cho -), 23 bit cho mantissa, 8 bit cho Exponent.
- Ví dụ $10 = 1.010_2 \times 2^3$ sẽ được lưu trong máy dạng

0	E=3	1.010 . . . 0

- Biểu diễn bộ cộng kép (double precision): 64 bits = 1 bit cho dấu (0 cho +, 1 cho -), 52 bit cho mantissa, 11 bit cho Exponent.
- Chú ý rằng trong phần định trị m thì số đầu tiên luôn là 1, còn $n \ge 1 \Leftrightarrow E \ge 0$

Biểu diễn dấu chấm động/part 3 floating point representation

The gap between 1 and the next larger floating-point number is called the **machine precision** and is often denoted by ϵ. (In MATLAB, this number is called `eps`.) [Note: Some sources define the machine precision to be $\frac{1}{2}$ times this number.] In single precision, the next floating-point number after 1 is $1 + 2^{-23}$, which is stored as

0	E=0	00000000000000000000001

Thus, for single precision, we have $\epsilon = 2^{-23} \approx 1.2 \times 10^{-7}$.

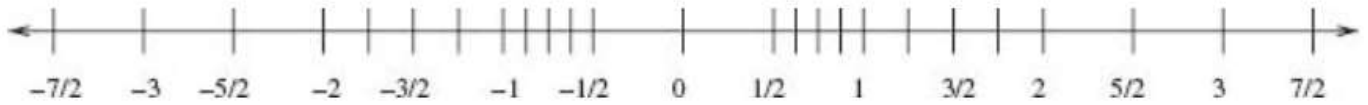

Figure 5.3. Number line for toy floating-point system.

The default precision in MATLAB is not single but **double precision**, where a word consists of 64 bits: 1 for the sign, 11 for the exponent, and 52 for the significand. Hence in double precision the next larger floating-point number after 1 is $1 + 2^{-52}$, so that the machine precision is $2^{-52} \approx 2.2 \times 10^{-16}$.

Sai số tuyệt đối/tương đối

- Bài toán: Tính y_{exact}, giả sử ta tính được y_{approx}
- Sai số tuyệt đối (absolute error): (quan trọng khi $|y_{exact}|$ nhỏ) đ/n bởi $|y_{exact} - y_{approx}|$
- Sai số tương đối (relative error): (quan trọng khi $|y_{exact}|$ lớn) đ/n bởi

$$\frac{|y_{exact} - y_{approx}|}{|y_{approx}|}$$

- Ví dụ: nếu giá trị chính xác là 50 và giá trị gần đúng là 49.9, thì sai số tuyệt đối là 0.1 và sai số tương đối là 0.1 / 50 = 0.002 = 0.2%.
- Các phép tính về sai số: Chương 1, Mục 2 sách Giáo trình Phương Pháp Tính và MATLAB, Lê Trọng Vinh, Trần Minh Toàn, 2013, ĐHBKHN nhưng chúng ta sẽ không đi quá sâu vào vấn đề này.

§2. CÁC PHÉP TÍNH VỀ SAI SỐ

2.1 Các phép tính

Giả sử đại lượng f có sai số tuyệt đối giới hạn là Δ_f và sai số tương đối là δ_f. Ta đã biết $\Delta_f = |\Delta f|$ (Δf là số gia của đại lượng f). Khi đó:

1. Nếu $u = x + y + z$ thì $\Delta_u = \Delta_x + \Delta_y + \Delta_z, \quad (x, y, z > 0).$
2. Nếu $u = x - y$ thì $\delta_u = \dfrac{\Delta_x + \Delta_y}{|x - y|}, \quad (x, y > 0).$
3. Nếu $u = xyz$ thì $\delta_u = \delta_x + \delta_y + \delta_z, \quad (x, y, z > 0).$
4. Nếu $u = \dfrac{x}{y}$ thì $\delta_u = \delta_x + \delta_y, \quad (x, y > 0).$

Chú ý 1.1. Trong công thức hiệu, nếu $|x - y|$ quá bé thì sai số sẽ lớn. Vì vậy, trong tính toán người ta tìm cách tránh phép trừ các số khá gần nhau khi tính sai số tương đối của hiệu hai số.

2.2 Công thức tổng quát về sai số

Giả sử $u = f(x_1, x_2, \ldots, x_n)$ với f là hàm khả vi liên tục thì

$$\Delta_u = \sum_{i=1}^{n} \left| \frac{\partial f}{\partial x_i} \right| \Delta_{x_i}. \tag{1.2}$$

và nếu $f > 0$ thì ta có:

$$\delta_u = \sum_{i=1}^{n} \left| \frac{\partial \ln f}{\partial x_i} \right| \Delta_{x_i}. \tag{1.3}$$

Ví dụ 1.4. Tìm sai số tuyệt đối và sai số tương đối giới hạn của thể tích hình cầu $V = \frac{1}{6}\pi d^3$, nếu đường kính $d = 3.7 \pm 0.05$ (cm) và $\pi = 3.14 \pm 0.0016$.

Giải: Xem d và π là các đối số của hàm V, ta có:

$$\begin{aligned} \frac{\partial V}{\partial \pi} &= \frac{1}{6}d^3 = \frac{1}{6}(3.7)^3 = 8.44; \\ \frac{\partial V}{\partial d} &= \frac{1}{2}\pi d^2 = \frac{1}{2}(3.14)(3.7)^2 = 21.5. \end{aligned}$$

Theo công thức (1.2) ta được sai số tuyệt đối giới hạn

$$\begin{aligned} \Delta_V &= \left| \frac{\partial V}{\partial \pi} \right| \Delta_\pi + \left| \frac{\partial V}{\partial d} \right| \Delta_d = 8.44 \times 0.0016 + 21.5 \times 0.05 \\ &= 1.088 \approx 1.1(cm^3). \end{aligned}$$

Vậy $V = \frac{1}{6}\pi d^3 = \frac{1}{6}(3.14)(3.7)^3 \pm 1.1 = 27.4 \pm 1.1(cm^3)$.

Sai số tương đối $\delta_V = \frac{\Delta_V}{|V|} = \frac{1.088}{27.4} \approx 0.04 = 4\%$.

2.3 Bài toán ngược về sai số

Trong phần trước ta đã tính sai số của hàm số tùy thuộc vào sai số của đối số. Bây giờ ta xét trường hợp cho sai số của hàm thì giới hạn sai số của đối số cần cho phép là bao nhiêu?

Bài toán: Cho hàm số $u = f(x_1, x_2, \ldots, x_n)$. Hỏi sai số tuyệt đối (hay tương đối) giới hạn của mỗi đối số là bao nhiêu để sai số tuyệt đối (hay tương đối) giới hạn của hàm số không vượt quá một số dương cho trước (số dương này được gọi là sai số cho phép)?
Để giải quyết vấn đề đã đặt ra, có hai cách thường được sử dụng.

1. Cách giải quyết theo nguyên tắc sai số của các đối số ngang bằng nhau.
   Theo công thức (1.2) ta có:

$$\Delta_u = \sum_{i=1}^{n} \left|\frac{\partial f}{\partial x_i}\right| \Delta_{x_i}.$$

   Ta xem ảnh hưởng của mọi thành phần như nhau, tức là

$$\left|\frac{\partial f}{\partial x_1}\right| \Delta_{x_1} = \left|\frac{\partial f}{\partial x_2}\right| \Delta_{x_2} = \cdots = \left|\frac{\partial f}{\partial x_n}\right| \Delta_{x_n}$$

$$\Longrightarrow \Delta_u = \sum_{i=1}^{n} \left|\frac{\partial f}{\partial x_i}\right| \Delta_{x_i} = n \left|\frac{\partial f}{\partial x_i}\right| \Delta_{x_i}.$$

   Từ đó ta có công thức sai số tuyệt đối giới hạn cho phép của mỗi thành phần:

$$\Delta_{x_i} = \frac{\Delta_u}{n \left|\frac{\partial f}{\partial x_i}\right|} \quad i = 1, 2, \ldots, n. \tag{1.4}$$

Ví dụ 1.5. Cho hình trụ đứng, bán kính $R = 2(m)$; chiều cao $H = 3(m)$. Hỏi sai số tuyệt đối của R và H là bao nhiêu để thể tích $V = \pi R^2 H$ của hình trụ đạt sai số tuyệt đối giới hạn là $0.1(m^3)$.

Giải: Ta có

$$V = \pi R^2 H, \quad \Delta_V = 0.1(m^3), \quad R = 2(m), \quad H = 3(m)$$

và số $\pi = 3.14$ thì khi đó:

$$\begin{aligned} \frac{\partial V}{\partial \pi} &= R^2 H = 2^2 \times 3 = 12; \\ \frac{\partial V}{\partial H} &= \pi R^2 = 3.14 \times 2^2 = 12.6; \\ \frac{\partial V}{\partial R} &= 2\pi RH = 2 \times 3.14 \times 2 \times 3 = 37.7. \end{aligned}$$

Ở đây $n = 3$, theo nguyên tắc ngang bằng nhau ta thu được:

$$\begin{aligned} \Delta_\pi &= \frac{0.1}{3 \times 12} < 0.003; \\ \Delta_R &= \frac{0.1}{3 \times 12.6} < 0.001; \\ \Delta_H &= \frac{0.1}{3 \times 37.7} < 0.003 \end{aligned}$$

là các sai số của mỗi thành phần để V có sai số tuyệt đối giới hạn không vượt quá $0.1(m^3)$.

2. Phương pháp điều chỉnh

Trong thực tế, khi sai số của thành phần nào đó tính theo nguyên tắc ngang bằng nhau là không phù hợp (không thực hiện được) ta phải điều chỉnh thành phần đó. Chẳng hạn đối với bài toán sau:

Ví dụ 1.6. Hỏi phải đo bán kính $R = 30.5(cm)$ có sai số tối đa là bao nhiêu và số π lấy là bao nhiêu để được hình tròn có diên tích đạt sai số tương đối không quá 1%?

Giải: Ta có diện tích hình tròn $S = \pi R^2$, nên $\ln S = \ln \pi + 2 \ln R$, suy ra sai số tương đối

$$\delta_S = \frac{\Delta_S}{S} = \frac{1}{\pi}\Delta_\pi + \frac{2}{R}\Delta_R = 1\% = 0.01.$$

Theo nguyên tắc ngang bằng nhau (có hai thành phần) thì

$$\frac{\Delta_\pi}{\pi} = 2\frac{\Delta_R}{R} = \frac{0.01}{2} = 0.005.$$

Từ đó suy ra

- $\Delta_\pi = \pi \times 0.005 \leq 0.0016$ với $\pi = 3.14$;
- $\Delta_R = \dfrac{R}{2} \times 0.005 \leq 0.07(cm)$ với $R = 30.5(cm)$.

Rõ ràng đối với R khó thực hiện được vì Δ_R quá nhỏ, do đó cần phải tăng Δ_R lên. Do tổng cả hai thành phần là 0.01 nên khi Δ_R tăng thì Δ_π phải giảm, nghĩa là cần phải lấy thêm chữ số thập phân trong số π.

Chẳng hạn, lấy $\pi = 3.142$ thì $\dfrac{\Delta_\pi}{\pi} = \dfrac{3.142 - 3.1416}{3.142} = 0.00013$. Khi đó $2\dfrac{\Delta_R}{R} = 0.01 - 0.00013 = 0.00983$, suy ra

$$\Delta_R = \frac{0.00983}{2} \times 30.5 < 0.15(cm).$$

Ta thấy yêu cầu của độ chính xác này có thể thực hiên được.

Qua ví dụ trên ta thấy, việc điều chỉnh là tùy thuộc công việc, không theo nguyên tắc chung nên chúng tôi chỉ nêu ví dụ để cùng suy ngẫm khi thực hiện công việc mà thực tế đặt ra.