Section 1

Error analysis

Types of Errors

1. Round-off errors:

Rational numbers without recurring digits can be represented by a finite number of digits. These number of digits however may be larger than the size of memory. Rational numbers with recurring digits and irrational number cannot be represented by a finite number of digits.

Also, we record measurements only up to certain precision. Such errors are called round-off errors. See Sec. for discussion.

2. **Approximate errors**:

On many occasions we compute functions using series expansion. For example, sin(x). We truncate the series at some steps. The error due to such truncation is called *approximation error* or *truncation error.*

In a series of computation, the error could add up in two ways:

In a random manner: The errors in every step is random and uncorrelated, hence the total error after *N* steps will go as $\epsilon\sqrt{N}$, where the error in each step is ϵ.

In a systematic manner: The errors in different steps are somewhat similar, hence the total error after *N* steps will go as ϵN.

Errors in arithmetic operation

Suppose the computer representation of a variable x is x_c. Let us estimate the error in a subtraction operation

$$r = x - y$$

In computer representation

$$r_c \simeq x_c - y_c \simeq x(1+\epsilon_x) - y(1+\epsilon_y).$$

Therefore

$$\frac{r_c}{r} \simeq 1 + \epsilon_r \simeq 1 + \epsilon_x \frac{x}{r} - \epsilon_y \frac{y}{r}$$

The error ϵ_r is maximum and dangerous when $r \to 0$, or when we subtract two equal numbers. For this case, assuming worst case scenario when the errors could add up

$$|\epsilon_r| \simeq \frac{x}{r}(|\epsilon_x| + |\epsilon_y|)$$

Example: The solution of a quadratic equation

$$ax^2 + bx + c = 0$$

are

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}.$$

For small *c,* the solutions are

$x \to -b/a$ and 0.

For $a = 1, b = 1, c = 10^{-10}$, the numerical solution are

$$x = -0.9999999999, -1.00000008274 \times 10^{-10}$$

Program segment

```
import numpy as np
a = 1
b = 1
c = 10**(-10)

x1 = (-b+np.sqrt(b**2-4*a*c))/(2*a)
x2 = (-b-np.sqrt(b**2-4*a*c))/(2*a)
y1 = (-2*c)/(b+np.sqrt(b**2-4*a*c))
y2 = (-2*c)/(b+np.sqrt(b**2-4*a*c))

print x1, y1, x2, y2
```

The above equation can be rewritten as

$$cx^{-2} + bx^{-1} + a = 01+$$

whose solutions are

$$x'_{1,2} = \frac{-2c}{-b \pm \sqrt{b^2 - 4ac}}$$

For $a = 1, b = 1, c = 10^{-10}$, the numerical solution are

$$x = -1.000000000 \times 10^{-10}, -1.000000000 \times 10^{-10},$$

which are incorrect because the error in computation of $-b + \sqrt{b^2 - 4ac}$ is of the order of *c*, that is why the error in computation of $\dfrac{-2c}{-b + \sqrt{b^2 - 4ac}}$ is very large.

Correct this... for different c's.

For a product $r = xy$,

$$\frac{r_c}{r} = \frac{x_c y_c}{xy} \approx 1 + \epsilon_x + \epsilon_y.$$

Error in series expansion

Estimate error in a series computation of $\exp(x)$ and $\sin(x)$ for $x = 1$.

Error of the order of last term of the series.