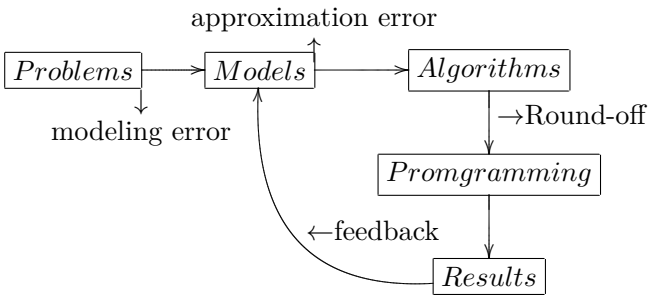
Numerical Analysis

Iydon

2018 年 9 月 6 日

目录

1 Preface


approximation error
Problems
Models
Algorithms
modeling error
→Round-off
Promgramming
←feedback
Results

2 Mathematical Preliminaries and Error Analysis

2.1 Round-off Errors and Computer Arithmetic

2.1.1 Binary machine numbers

定义 2.1 舍入误差 舍入误差形成原因:进行有限位的运算(*finite digits arithmetic*)

其中，*IEEE:754-2008* 规定二进制机器数（*Binary machine numbers*）中浮点数（*floating-point*）存储规范如下:

$$(-1)^S 2^{c-1023}(1+f)$$

$$\begin{cases} S: 0/1 & signpart \\ c: 11\, digits & exponential\ part. \\ f: 52\, digits & mantissa\ part \end{cases}$$

由于实数的稠密性，可知找不到比某一个数大的最小的数或小的最大的数，但是在计算机中可以找到，所以计算机不能表示所有的数。

2.1.2 Decimal machine numbers

$\pm 0.d_1 d_2 \cdots d_n \times 10^n$ 其中 $1 \le d_1 \le 9$，$0 \le d_i \le 9$，$\forall i \ge 2$。如果记真实的数为 y，其浮点数表示为 $fl(y)$。

当存在 $y = 0.d_1 d_2 \cdots d_k d_{k+1} \cdots \times 10^n$，其浮点数表示有如下两种方式:

1. Chopping: chop off digits，say $d_{k+1} d_{k+2} \cdots$.
2. Rounding: $y + 5^{n-(k+1)}$, then chopping.

例 2.1

$\pi = 3.14159265\cdots$，取 5 位。

- Chopping: $fl(y) = 0.31415 \times 10^1$.
- Rounding: $fl(y) = 0.31416 \times 10^1$.

π

定义 2.2 *Suppose $p^\star$ is an approximation of p.*

$$\begin{cases} absolute\,error & = |p^\star - p| \\ relative\,error & = \frac{|p^\star - p|}{p} \end{cases}$$

定义 2.3 有效数字（*Significant digits*） *$p^\star$ is said to approximate p with* t *significant digits. If t is the* largest nonnegative *integer, s.t.*

$$\frac{|p - p^\star|}{p} \leq 5 \times 10^{-t}$$

.

Chopping floating:

$$y = 0.d_1 \cdots d_k d_{k+1} \cdots \times 10^n$$

$$fl(y) = 0.d_1 \cdots d_k \times 10^n$$

Chopping:（其有效位数至少为 k-1）

$$\frac{|fl(y) - y|}{|y|} = \frac{0.0 \cdots 0 d_{k+1} \cdots \times 10^n}{0.d_1 \cdots d_k d_{k+1} \cdots \times 10^n} \leq 10^{1-k}$$

Rounding:（其有效位数至少为 k）

$$\frac{|fl(y) - y|}{|y|} \leq \frac{0.0 \cdots 1 d_{k+1} \cdots \times 10^n}{0.d_1 \cdots d_k d_{k+1} \cdots \times 10^n} \leq 10^{-k}$$

2.1.3 Machine Operators

记计算机的加减乘除为 $\oplus \ominus \otimes \oslash$，于是有

$$x \textcircled{op} y = fl(fl(x) \textcircled{op} fl(y))$$

Four cases to avoid:

1. 两个十分接近的数（two nearly equal）。
2. 分子远大于分母（numerator » denominator）。
3. 避免大数吃掉小数。

2.1.4 Nested method（秦九韶算法）

input : $a_0, a_1, \cdots, a_n(given)$; x
output: $P_n(x)$

$S_n \leftarrow a_n$;
for $k \leftarrow n-2$ **to** 0 **do**
  $S_k \leftarrow xS_{k+1} + a_k$;
end
$P_n(x) \leftarrow S_0$;

```python
def nested(poly:list=[1], x:float=0.0)->float:
    """
    Horner nested polynomial calculation.

    Args:
        poly: List, store the coefficient of the polynomial.
        x: Float, specify the variable in the polynomial.

    Returns:
        Float, result.

    Raises:
```

```
        If `poly' is empty, raise IndexError.
        If type(args) does not correspond, raise TypeError.
    """
    result = poly[0]
    for i in range(1, poly.__len__()):
        result = x*result + poly[i]
    return result
```

2.1.5 Convergence（收敛性）

Stable: small change in initial data and the error is small.
若 E_0 为初始值误差，E_n 为 n 步的误差，

- $E_n \approx C$（不依赖 n），称之为线性。
- $E_n \approx C^n E_0$ 则可由 C 的取值判断是否稳定。

定义 2.4 *Rates of Convergence* 当 $n \to \infty$，$\alpha_n \to \alpha$，$\beta_n \to 0$，其中 $|\alpha_n - \alpha| \leq k\,|\beta_n|$（与 n 的取值无关），则称 α_n 是以 β_n 的速度收敛到 α 的。

$$\alpha_n = \alpha + o(\beta_n).$$

3 Root-finding problem

定理 3.1 *Intermediate Value Theorem* $f \in [a,b]$，$\forall k \in f([a,b])$，$\exists c \in [a,b]$，$s.t.\ f(c) = k$。

```
def Bisection(fun, a:float, b:float, max_step:int=128, ...
    eps:float=1e-6)->float:
    mid_last = a
    if fun(a)*fun(b) < 0:
        for i in range(0, max_step):
            mid = (a+b) / 2
            if abs(mid-mid_last)<eps or abs(fun(mid))<eps:
                print("Step: %d\nZero: %fc"%(i, mid))
                return mid
            else:
                if fun(mid)*fun(a)<0:
                    b = mid
                else:
                    a = mid
            mid_last = mid
        print('Bisection cannot be convergent within..
               the pre-set steps.')
```

定理 3.2 $f \in C[a,b](continuous)$，根据如上算法，P_i 为 mid 的序列。如果 $\exists$ $root$ $P \in [a,b]$，则有 $|P_n - P| \leq \frac{b-a}{x^n}$。

【证明】 $|b_n - a_n| = \frac{b-a}{2^{n-1}}$，

$$|P_n - P| \leq \frac{1}{2}(b_n - a_n) = \frac{b-a}{2^n}$$

于是 $P_n = P + o(2_{-n})$。 □

定义 3.1 *Fixed-point Iteration* 对 $g(P)$，如果 $\forall x \in [a,b]$，如果 $\exists\ P$ *s.t.* $g(P)=P$，则称 P 为不动点（*fixed point*）。
如果 $g(x) \in C[a,b]$ 并且 $g([a,b]) \subset [a,b]$，*there exists at least one* $p \in [a,b]$，*s.t.* $g(p)=p$。

定理 3.3 不动点迭代根的存在唯一性定理 $g(x) \in C[a,b]$，$g([a,b]) \subset [a,b]$。$\forall x \in [a,b]$，都有 $g'(x) \le \kappa < 1$。

【证明】
存在性：

$$\begin{cases} h(a) = \quad g(a) - a \ge 0 \\ h(b) = \quad g(b) - b \le 0 \end{cases}$$

于是有 $h(a)h(b) \le 0$，则 $\exists$ p，*s.t.* h(p)=0。
唯一性：
假设存在两个根 P_1，P_2，使得 $P_1 = g(P_1)$，$P_2 = g(P_2)$，但是 $P_1 \ne P_2$。

$$\begin{aligned} |g(P_1) - g(P_2)| &= |g'(\xi)|\,|P_1 - P_2||\,, \quad \xi \in [P_1, P_2]. \\ &\le \kappa\,|P_1 - P_2|\,, contradiction. \end{aligned}$$

□

定理 3.4 不动点收敛的充分条件 $g \in C[a,b]$，$g([a,b]) \subset [a,b]$，$g'(x)$ 存在，并且 $|g'(x)| \le \kappa < 1$。$\forall P_0 \in [a,b]$，定义序列 $P_i = g(P_{i-1}),\quad i = 1,2,\cdots$，则 $\lim_{n\to\infty} P_n = P$（$P$ 为不动点）。

【证明】

$$\begin{aligned} |P_n - P_n| &= |g(P_{n-1} - g(P))| \\ &= |g'(\xi_{n-1})|\,|P_{n-1} - P| \\ &\le \kappa\,|P_{n-1} - P| \\ &\le \cdots \le \cdots \\ &\le \kappa^n\,|P_0 - P| \to 0. \end{aligned}$$

□

其中，寻找不动点的代码如下：

```
def fixed_point(fun, start:float=0, max_step:int=128, ...
    eps:float=1e-6)->float:
    new_val = fun(start)
    for i in range(0, max_step):
        old_val = new_val
        new_val = fun(old_val)
        if -eps<old_val-new_val<eps:
            print(i)
            return new_val
```