科学計算研究室 Python ゼミ

～ 1. 二分法と Newton 法～

2021-02-10 福田　浩

1　二分法

1.1　原理

1.1.1　前提条件

- $f(x) = 0$ の解が ans，解の探査区間が $b \leq x \leq a$ のとき，$b \leq ans \leq a$ であること．
- 解は探査区間が $b \leq x \leq a$ の中に 1 つしかないこと．

1.1.2　アルゴリズム

1. a と b の中間点 c を求める
2. $f(a) \times f(c) < 0$ であれば，ans は $c \leq ans \leq a$ を満たすので，b に c を代入する
3. $f(a) \times f(c) \geq 0$ であれば，ans は $b \leq ans \leq c$ を満たすので，a に c を代入する
4. 1. に戻る

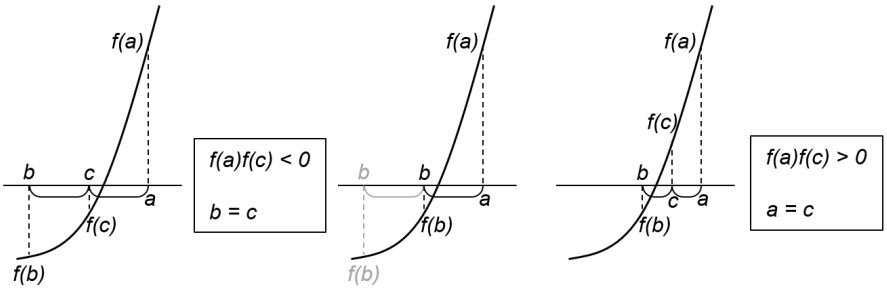

Figure 1: 二分法の原理.

1.1.3　課題

二分法を用いて 2 次方程式の解を求めるプログラムを Python で書き，動作を検証せよ．

- 条件
  - 解 ans が，$-1 \leq ans \leq 1$ の範囲にある場合に限定してよい
  - 2 次方程式の係数を標準入力から取得する機能を備えること
  - 解を標準出力へ出力する機能を備えること
- 課題
  - 繰り返し演算から抜ける条件を考えよ
  - matplotlib を用いて解を可視化し，二分法の計算結果の妥当性を検証せよ

1.2 参考： C++のソースコード例

2 次の係数が a_1，1 次の係数が a_2，定数が a_3 の 2 次方程式の解を二分法により求める．アルゴリズムの参考例であり，エラー処理は実装していない．

```
#include <stdio.h>
#include <math.h>
#include <iostream>
#define EPS 0.00001
using namespace std;

double func_y(double x, double a1, double a2, double a3){
    return a1*x*x + a2*x +a3;
}

int main(void){
    double a, b, c;
    double a1, a2, a3;
    a = -1.0;
    b = 1.0;
    c = 0.0;
    cin >> a1 >> a2 >> a3;
    while(fabs(func_y(c,a1,a2,a3))>EPS){
        c = (a+b)/2;
        if(func_y(a,a1,a2,a3)*func_y(c,a1,a2,a3)<0) b = c;
        else a = c;
    }
    cout << "ans=" << c << endl;
    cout << "func_y=" << func_y(c,a1,a2,a3) << endl;
    return 0;
}
```

1.3 更なる検討

2 次方程式は「解の公式」 $\frac{-a_2 \pm \sqrt{a_2^2 - 4a_1a_3}}{2a_1}$ でも求められるが，係数が極端に大きい場合はオーバーフローを起こすことがあるし，極端に小さい場合は計算誤差が累積して正解を得られない場合がある．アルゴリズムが大筋で正しくても，正解に辿り着けないケースを「コーナー条件」と呼ぶ．このような時，ササっと二分法を用いるほうが確実な場合も多い．解の公式のコーナー条件を探し，それを二分法で解決してみよ．

2 Newton 法

2.1 原理

2.1.1 前提条件

- $f(x) = 0$ の解が ans，解の探査区間が $b \leq x \leq a$ のとき，$b \leq ans \leq a$ であること．
- 解は探査区間が $b \leq x \leq a$ の中に 1 つしかないこと．
- 探査区間が $b \leq x \leq a$ に変曲点がないこと．

2.1.2 アルゴリズム

1. $f(x)$ に $x = a$ で接する直線を求める（ $y = f'(a)(x - a) + f(a)$ ）
2. 上記直前が x 軸と交わる点を b とする
3. a に b を代入する
4. 1. に戻る

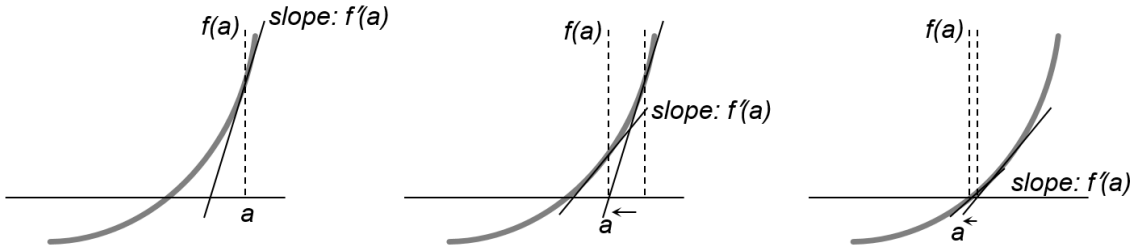

Figure 2: Newton 法の原理.

2.1.3 課題

Newton 法を用いて 2 次方程式の解を求めるプログラムを Python で書き，動作を検証せよ．

- 条件
  - 解 ans が，$-1 \leq ans \leq 1$ の範囲にある場合に限定してよい
  - 2 次方程式の係数を標準入力から取得する機能を備えること
  - 解を標準出力へ出力する機能を備えること
- 課題
  - 繰り返し演算から抜ける条件を考えよ
  - 三角関数に適用してみよ ($x - \sin x + 0.5 = 0$ など)
  - 三角関数にそもまま適用した場合と，三角関数を Taylor 展開 ($\sin(x) = x - \frac{x^3}{3} + \frac{x^5}{5} \cdots, \cos(x) = 1 - \frac{x}{2} + \frac{x^2}{4} \cdots$) した近似式に適用した場合の差を考察せよ
  - matplotlib を用いて解を可視化し，Newton 法の計算結果の妥当性を検証せよ

2.2 参考： C++のソースコード例

2 次の係数が a_1，1 次の係数が a_2，定数が a_3 の 2 次方程式の解を Newton 法により求める．アルゴリズムの参考例であり，エラー処理は実装していない．

```
#include <stdio.h>
#include <math.h>
#include <iostream>
#define EPS 0.00001
using namespace std;

double nt(double x, double a1, double a2, double a3){
    return (x*(2*a1*x+a2)-(a1*x*x+a2*x+a3))/(2*a1*x+a2);
}

int main(void){
    double a,b;
    double a1, a2, a3;
    a = 1.0;
    b = 0.0;
    cin >> a1 >> a2 >> a3;
    while(fabs(a-b)>EPS){
        a = b;
        b = nt(a, a1, a2, a3);
    }
    cout << "ans=" << b << endl;
    return 0;
}
```

2.3 更なる検討

- 探索区間に変曲点がある場合の振る舞いを予測し，仮説を立てて検証してみよ．
- Newton 法は二分法よりも収束が速い (と言われている)．どのくらい速度の違いがあるか，検証してみよ．ほとんど変わらない条件と，大きく変わる条件があるか？