第 1 章

非線形方程式の根

1 次方程式や 2 次方程式の根を求める場合には，根の公式があるためわざわざ数字で根を求める必要はない．3 次方程式や 4 次方程式にも少し複雑になるが，ともかく根の公式がある．しかし，5 次以上の代数方程式では，解は確かに複素数の範囲で存在するが根の公式は存在しない．そのような場合にどのようにして根を求めればよいのだろうか．さらに代数方程式でない場合，たとえば

$$\cos x = x^2 \tag{1.1}$$

といったように，ふつうの方法では解けそうにもない方程式の根はどのようにして求めればよいのであろうか．本章では，このように公式のあてはめでは解けない方程式の解を数値で少なくとも 1 つ求める方法をいくつか紹介しよう．

1.1 反　復　法

0.2 節では 2 次方程式を反復法を用いて解いたが，この方法は一般の方程式に対しても適用することができる．根を求めたい方程式を

$$f(x) = 0 \tag{1.2}$$

とする．反復法はこの方程式を

$$x = F(x) \tag{1.3}$$

の形に変形し，粗い近似値 x_0 から出発して

$$\begin{aligned} x_1 &= F(x_0) \\ x_2 &= F(x_1) \\ &\cdots \\ x_{n+1} &= F(x_n) \end{aligned}$$

と反復しながら根を求めていく方法で，一般に反復式

$$x_{n+1} = F(x_n) \quad (n = 0, 1, 2, \cdots) \tag{1.4}$$

を，収束判定条件を満足するまで繰り返す．もちろん，この方法が使えるためには反復式が収束する必要がある．

例題 1-1　反復式が収束するための条件

式 (1.4) の反復式が収束するための条件を求めてみよう．式 (1.2) の根を α とすると，式 (1.3) から

$$\alpha = F(\alpha) \tag{1.5}$$

さらに

$$x_{n+1} = F(x_n) \tag{1.6}$$

でもあるから，式 (1.5) から式 (1.6) を引くと

$$\alpha - x_{n+1} = F(\alpha) - F(x_n) \tag{1.7}$$

となる．関数 $F(x)$ が閉区間 $[\alpha, x_n]$ で連続で，開区間 (α, x_n) で微分可能であれば，平均値の定理より

$$F'(c) = \frac{F(\alpha) - F(x_n)}{\alpha - x_n} \quad (\alpha < c < x_n) \tag{1.8}$$

が成り立つ．式 (1.8) を式 (1.7) に代入することにより

$$\alpha - x_{n+1} = (\alpha - x_n)F'(c) \quad (\alpha < c < x_n) \tag{1.9}$$

となり，式 (1.9) に $n = 0, 1, 2, 3 \cdots$ を代入すると

$$\left.\begin{array}{l}
\alpha - x_1 \sim F'(c)(\alpha - x_0) \\
\alpha - x_2 \sim F'(c)(\alpha - x_1) = \{F'(c)\}^2 (\alpha - x_0) \\
\alpha - x_3 \sim F'(c)(\alpha - x_2) = \{F'(c)\}^3 (\alpha - x_0) \\
\vdots \\
\alpha - x_{n+1} \sim F'(c)(\alpha - x_k) = \{F'(c)\}^{n+1} (\alpha - x_0)
\end{array}\right\} \tag{1.10}$$

となる．したがって $|F'(c)| < 1$ ならば収束する．

《反復法のアルゴリズム》

① 初期値 x_0 を適当に決める.

② $y_0 = F(x_0)$ を計算する.

③ $|x_0 - y_0| < \varepsilon$（$\varepsilon$：小さな正数）のとき⑤へ.

④ $x_0 = y_0$ として②へ.

⑤ 近似値 x_0 を出力する.

例題 1-2　$e^x - 5\sin x + 1.36x = 0$ を反復法を用いて解け

上式を式(1.3) にしたがって, たとえば次のように変形する.

$$x = e^x - 5\sin x + 2.36x$$

したがって

$$F(x) = e^x - 5\sin x + 2.36x$$

となる．例として

$$x_0 = 1$$

$$\varepsilon = 0.00001$$

のとき 21 回の反復で

$$x = 0.4535$$

反復法により上記の例題を解くプログラムを以下のプログラム 1.1 に示す.

プログラム 1.1　反復法

```
Option Explicit

Private Sub CMD_Calculation_Click()
    Dim X0 As Double
    Dim X1 As Double
    Dim epsl As Double

    X0 = Range("_X0")
    epsl = 0.000001
    X1 = Exp(X0) - 5# * Sin(X0) + 2.36 * X0
```

```
    Do Until (Abs(X1 - X0) < epsl)
        X0 = X1
        X1 = Exp(X0) - 5# * Sin(X0) + 2.36 * X0
    Loop

    Range("_X1") = X1

End Sub
```

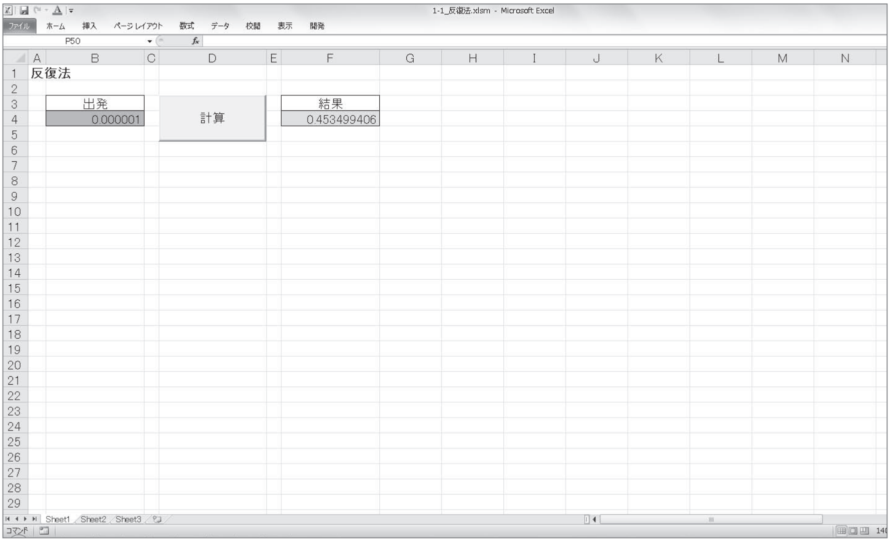

図 1.1　反復法（実行結果）

1.2　2　分　法

2 分法の原理は極めて単純である．はじめに，2 つの初期値を a，b（$a < b$）として

$$f(a)f(b) < 0 \tag{1.11}$$

を満足するように選ぶ．ただし $f(x)$ は連続とする．式(1.11) は $f(a)$ と $f(b)$ が異符号であることを意味しているため，曲線 $y = f(x)$ 上の 2 点$(a,\ f(a))$，$(b,\ f(b))$ は x 軸をはさんで反対側にある．$f(x)$ は連続であるため，図 1.2 に示すように曲線が $x=a$，$x=b$ の間のある点で少なくとも 1 回は x 軸と交わる（中間値の定理）．すなわち，根は a と b の間に少なくとも 1 つある．そこで a と b の中点を c とすれば，根は a と c の間かまたは c と b の間に少なくとも 1 つある[*1]．

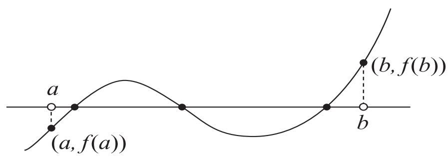

図 1.2　中間値の定理

このことは，$f(a)f(c) < 0$ または $f(c)f(b) < 0$ のいずれかが成り立つことを意味している．そこで $f(a)f(c) < 0$ の場合は c を新たに b の修正値とみなし $f(c)f(b) < 0$ の場合は c を新たに a の修正値とみなすことにすれば，上の式(1.11)と同じ状態になり, しかも区間の幅は半分になる．これらの手続きを図 1.3

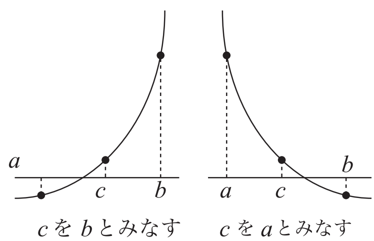

図 1.3　2 分法

＊1　偶然に c が根であることもあり得るが，そのときはそれで計算が終了することになる．

に示す．あとは，同じことを繰り返せば，根の含まれている区間幅を限りなく 0 に近づけることができる．実際には計算誤差 ε を指定し，区間幅がこの誤差以内になれば計算を打ち切ればよい．

《2 分法のアルゴリズム》

① $f(a)f(b) < 0$ となるような a，b（$a < b$）を初期値にとる．

② $c =$（$a + b$）$/2$ とする．

③ $b - a$ があらかじめ指定した誤差より小さければ根を c とする．

④ $f(a)f(\mathrm{c}) < 0$ ならば，c を新たに b の修正値とみなして②に戻る．

⑤ $f(\mathrm{c})f(b) < 0$ ならば，c を新たに a の修正値とみなして②に戻る．

初期値 a，b が見つかることが 2 分法の前提条件であるが，見つかればあとは確実に根が求まる．ただし，1 回の手続きで区間幅が半分になるだけであるので根を求めるのに時間がかかるという欠点がある．

具体例として $\cos x - x^2 = 0$ の根を，初期値を $a = 0$，$b = 1$ として求める場合に c の値を記すと以下のようになる．

$$
\begin{array}{rl}
1 & c = 0 \\
2 & c = 0.5 \\
3 & c = 0.75 \\
4 & c = 0.875 \\
 & \cdots \\
15 & c = 0.82415771 \\
 & \cdots \\
26 & c = 0.82413229 \\
27 & c = 0.82413231
\end{array}
\tag{1.12}
$$

2 分法の変形として，2 分法の中点 c のかわりに，2 点（a, $f(a)$）と（b, $f(b)$）を結ぶ直線と x 軸の交点を c にとる方法もある（図 1.5）．この 2 点を結ぶ直線は

$$
y - f(a) = \frac{f(b) - f(a)}{b - a}(x - a) \tag{1.13}
$$

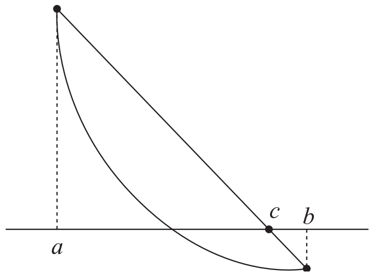

図 1.4　はさみうち法

であるから，x 軸上では

$$0 - f(a) = \frac{f(b) - f(a)}{b - a}(c - a) \tag{1.14}$$

となる．したがって，

$$c = \frac{af(b) - bf(a)}{f(b) - f(a)} \tag{1.15}$$

となる．この方法の計算手順は 2 分法の計算手順の②を

$$c = \frac{af(b) - bf(a)}{f(b) - f(a)} \tag{1.16}$$

で置き換えるだけで，あとはまったく同じである．この方法ははさみうち法とよばれている．

例題 1-3　方程式 $\cos x - x^2 = 0$ をはさみうち法で解け．

1	$c = 0.00000000$	6	$c = 0.82412287$
2	$c = 0.68507336$	7	$c = 0.82413148$
3	$c = 0.81069365$	8	$c = 0.82413224$
4	$c = 0.82293160$	9	$c = 0.82413231$
5	$c = 0.82402582$		

たまたまこの例では 2 分法よりは収束は速くなったが，このことはいつも成り立つとは限らず，逆に遅いこともある．

以下のプログラム 1.2 は，2 分法とはさみうち法を用いた非線形方程式の解法のプログラムであり，例として例題 1-3 を解いている．

プログラム 1.2　2 分法とはさみうち法による非線形方程式の解法

```
OOption Explicit

Private Sub CMD_Calculation_Click()
    Dim A As Double
    Dim B As Double
    Dim KTYPE As Integer
    Dim MSG As String
    Dim I As Integer
    Dim X1 As Double

    A = Range("_A")
    B = Range("_B")
    KTYPE = Range("_KTYPE")

    Call Calculation(A, B, KTYPE, MSG, I, X1)

    Range("_MSG") = MSG
    Range("_I") = I
    Range("_X1") = X1
End Sub

Sub Calculation(A As Double, _
                B As Double, _
                KTYPE As Integer, _
                ByRef MSG As String, _
                ByRef I As Integer, _
                ByRef X1 As Double)

    Dim EPS As Double


    EPS = 0.0000001
    For I = 1 To 50
        Select Case KTYPE
        Case 1: X1 = (A + B) / 2#
        Case 2: X1 = (A * F(B) - B * F(A)) / (F(B) - F(A))
        End Select

        If (F(X1) * F(B) < 0) Then
            If (Abs(A - X1) < EPS) Then Exit For
            A = X1
        Else
            If (Abs(A - X1) < EPS) Then Exit For
```

```
            B = X1
        End If
    Next I
    If I = 50 Then
        MSG = " 収束しません。"
        I = 0
        X1 = 0
    Else
        MSG = " 収束しました "
    End If

End Sub

Function F(X As Double) As Double

    F = Cos(X) - X * X

End Function
```

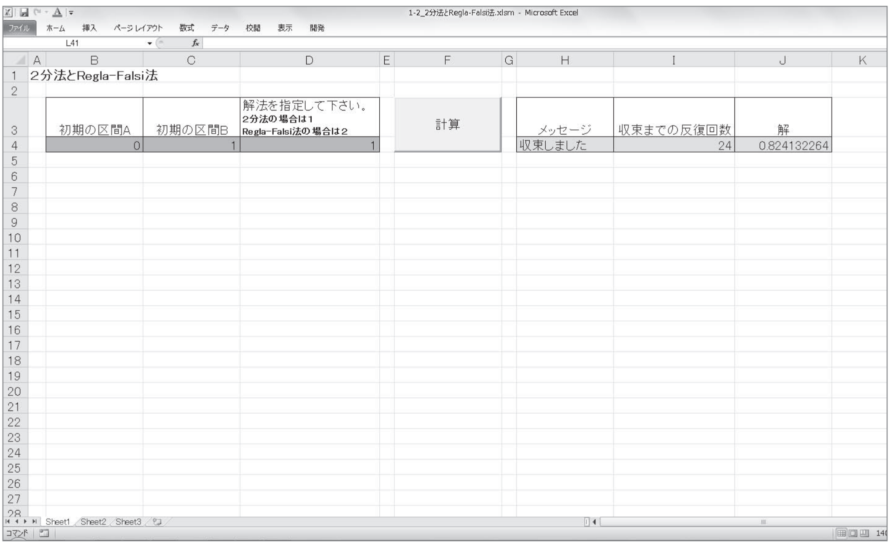

図 1.5　2 分法とはさみうち法による非線形方程式（実行結果）