

スライド 1 講義アウトライン

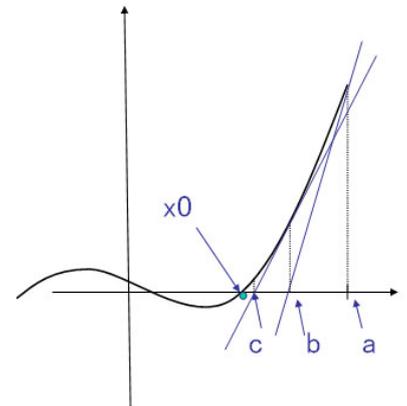
- 非線形方程式
 - 「 \dots 」法
 - ニュートン法 (「 \dots 」変数)

スライド 2 連立 1 次方程式 p.70

- 代数方程式の解 $f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0 = 0$
 - 「 \dots 」次方程式までは公式があり，解ける.
 - 「 \dots 」次以上の方程式に代数的解法は存在しない.
 - 数値的に解くしかない.
- 高次の連立方程式，三角関数など変数の「 \dots 」で表せない項を含む方程式

$$f(x) = x - \cos x = 0$$

- 数値的に解くしかない.

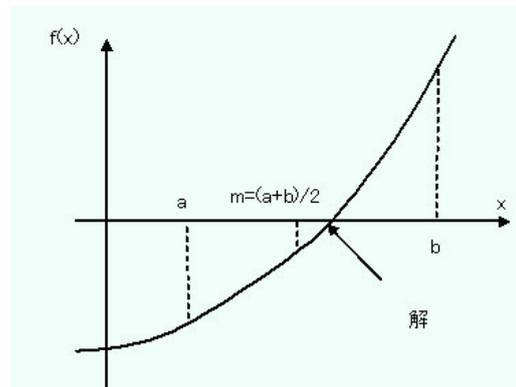


スライド 3 2 分法 p.70

- 定理 4.1 (「 \dots 」の定理)
関数 $f(x)$ は閉区間 $[a, b]$ で「 \dots 」で $f(a) \neq f(b)$ ならば， $f(a)$ と $f(b)$ の間の任意の数 k に対して $f(c) = k$ となる c ($a < c < b$) が存在する.

2 分法

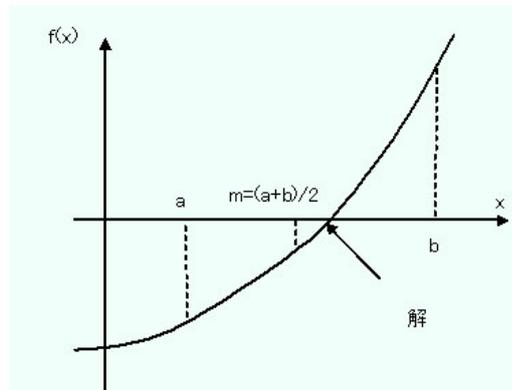
- 中間値の定理より， $f(a) f(b) < 0$ ならば， $f(c) = 0$ ， $a < c < b$ となる c が存在する.
- (1) 何らかの方法で $f(a) f(b) < 0$ となる閉区間 $[a, b]$ を求める.
- (2) $c_1 = (a+b)/2$
 - (2a) $f(a) f(c_1) < 0$ のとき，解 α は $[a, c_1]$ に存在する.
 - (2b) $f(c_1) f(b) < 0$ のとき，解 α は $[c_1, b]$ に存在する.
- (2a) では $[a, b]$ の代わりに $[a, c_1]$ ，(2b) では $[c_1, b]$ とする.
- ステップ(2)に戻り，「 \dots 」が十分に小さくなるまで繰り返す



スライド4 2分法のアルゴリズム p.71

```

Input a, b, ε
Do
  c ← (a+b)/2
  if f(a)f(b) < 0 then
    b ← c
  else
    a ← c
  end if
while( |a - b| ≥ ε )
c ← (a+b)/2
Output c
    
```



(図中の m はアルゴリズムの c に対応)

スライド5 2分法 p.71

区間 [a, b] に解が 1 つしか存在しないのならば,

- n 回目の区間
- $d_n < \epsilon$ のときに計算を打ち切るとすると,
必要な「 n 」は,

$$d_n = \frac{d_{n-1}}{2} = \frac{d_{n-2}}{2^2} = \dots = \frac{d_0}{2^n}$$

$$d_n = \frac{d_0}{2^n} < \epsilon$$

$$n > \frac{\log(\frac{d_0}{\epsilon})}{\log 2} \quad \text{を満たす最小の自然数}$$

「 n 」誤差 $|c_n - \alpha| < \frac{d_n}{2}$

ステップ1) 区間 [a, b] の求め方

- (1) 最初の区間 $[x_{\min}, x_{\max}]$, および 微小区間の幅 h を与える.
- (2) $n=(x_{\max}-x_{\min})/h$ として分割数を定め, $x_0=x_{\min}$ とする.
- (3) $k=1,2,\dots,n$ に対して, $x_k=x_{\min}+kh$ とし, $f(x_{k-1})f(x_k) < 0$ なら $[x_{k-1}, x_k]$ を対象区間にする.

スライド6 2分法: プログラム p.73

```

#include <stdio.h>
#include <math.h>
double bisection(double a,double b,double eps); /* 2分法 */
double f(double x); /* 関数の定義 */
int main(void){
  double a,b,x,h,y1,y2,eps=pow(2.0,-30.0);
  int n;
  printf("初期区間[a,b]を入力してください. ---> a b\n");
  scanf("%lf%lf",&a,&b);
  printf("区間の分割数 n を入力してください. ---> n\n");
  scanf("%d",&n);
  /* 対象区間を探索しながら 2分法を適用 */
  h=(b-a)/n; y1=f(a);
    
```

```

for(x=a+h;x<=b;x+=h){
y2=f(x);
if(y1*y2<0.0){
printf("求める答えは x=%f です.\n",bisection(x-h,x,eps));
}
y1=y2;
}
return 0;
}
/* 2 分法 */
double bisection(double a,double b,double eps){
double c;
do{
c=0.5*(a+b);
if(f(a)*f(c)<0){
b=c;
}
else{
a=c;
}
}while(fabs(b-a) >=eps) ; /* fabs()は絶対値を返す.
/* 「C 言語入門」 p.264 */

c=0.5*(a+b);
return c;
}
/* 関数の定義 */
double f(double x){
return x*(x*x*(x*x-5.0)+4.0);
/* x*x*x*x*x-5.0*x*x*x+4.0*x */
}

```

実行結果

初期区間[a,b]を入力してください. --->a b

-3 3

区間の分割数 n を入力してください. --->

n

10

求める答えは x=-2.000000 です.

求める答えは x=-1.000000 です.

求める答えは x=-0.000000 です.

求める答えは x=1.000000 です.

求める答えは x=2.000000 です.

