

スライド1 反復法と縮小写像の原理 p.77

2分法 解の存在範囲を探索しながら狭めていく. 「」法

ニュートン法 初期値 x_0 から出発して解 α に収束するような列 $\{x_n\}$ を作り, x_n が α に十分近づいたときに計算を打ち切る. 「」法, あるいは「」法

• 反復法による列 $\{x_n\}$

$$x = g(x) \tag{1}$$

$$x_{n+1} = g(x_n) \tag{2}$$

$g(x)$: 「」関数, 式(1) x : 「」, 式(2): 「」反復

$\phi(x) \neq 0$ となるように選んで,

$$g(x) = x - \phi(x)f(x)$$

$$x = g(x) \Leftrightarrow \phi(x)f(x) = 0 \Leftrightarrow f(x) = 0$$

スライド2 ニュートン法 p.77

ニュートン法の原理

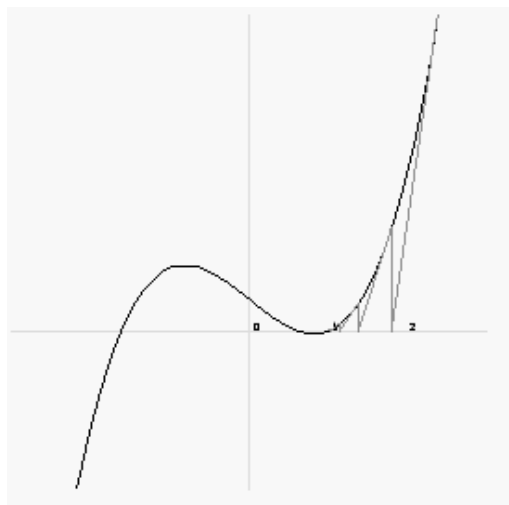
- 解 α の近傍で C^2 級とする. (「」関数が連続)
- 「」展開
- ニュートン反復列

$$0 = f(\alpha) = f(x_0) + f'(x_0)(\alpha - x_0) + \frac{1}{2}f''(\xi)(\alpha - x_0)^2, \xi \in (x_0, \alpha) \text{ or } (\alpha, x_0)$$

$$0 = f(\alpha) \approx f(x_0) + f'(x_0)\Delta x$$

$$f(x_0) + f'(x_0)\Delta x = 0 \quad \text{Hence} \quad \Delta x = -\frac{f(x_0)}{f'(x_0)}$$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$



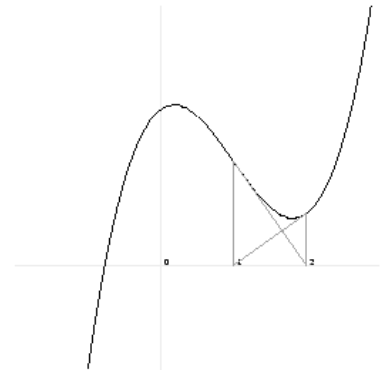
スライド3 ニュートン法：プログラム p.81

```

#include <stdio.h>
#include <math.h>
#define EPS pow(10.0,-8.0) /* epsilon の設定 */
#define NMAX 10 /* 最大反復回数 */
void newton(double x); /* Newton 法 */
double f(double x); /* f(x)の計算 */
double df(double x); /* f'(x)の計算 */
int main(void){
    double x;
    printf("初期値 x0 を入力してください¥n");
    scanf("%lf",&x);
    newton(x);
    return 0;
}
void newton(double x){/* Newton 法 */
    int n=0; double d;
    do{
        d=-f(x)/df(x);
        x=x+d;
        n++;
    }while(fabs(d)>EPS&& n<NMAX);
    if(n==NMAX){
        printf("答えは見つかりませんでした¥n");
    }
    else{
        printf("答えは x=%f です. ¥n",x);
    }
}
/* 関数の定義 */
double f(double x){
    return x*cos(x);
}
double df(double x){
    return 1.0+sin(x);
}

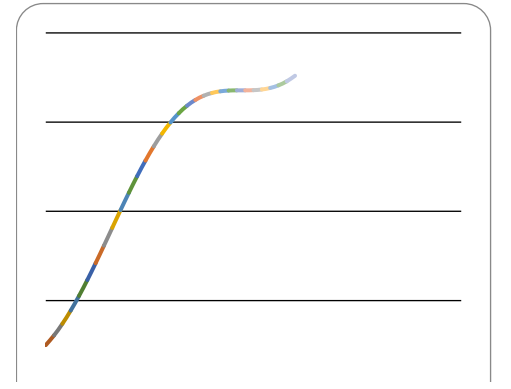
```

収束する場合



収束しない場合

実行結果 x*cos(x)=0
 (1 回目)
 初期値 x0 を入力してください.
 3
 答えは x=0.739085 です.
 (2 回目)
 初期値 x0 を入力してください.
 4
 答えがみつかりませんでした.



スライド4 収束の速さ p.82

(1) 線形収束

α に収束する反復列 $\{x_n\}$ が

$$(x_{n+1} - \alpha) = (A + \epsilon)(x_n - \alpha), \quad |A| < 1, \quad \lim_{n \rightarrow \infty} \epsilon_n = 0$$

A は 「 」 に依存しない定数

(2) p 次収束

$$|x_{n+1} - \alpha| \leq M|x_n - \alpha|^p, \quad p > 1, \quad 0 < M < \infty$$

M は 「 」 に依存しない定数

スライド5 収束の速さ p.82

定理 4.3 $f(x)=0$ の解 $x = \alpha$ が単解のとき, ニュートン法は2次収束する.

(証明)

$f(\alpha) = 0, f'(\alpha) \neq 0$ に注意すると, テイラーの公式より

$$\begin{aligned}x_{n+1} - \alpha &= x_n - \alpha - \frac{f(x_n)}{f'(x_n)} \\ &= \frac{-(f(x_n) + (\alpha - x_n)f'(x_n))}{f'(x_n)}\end{aligned}$$

なぜならば,

$$f(\alpha) = f(x_n) + (\alpha - x_n)f'(x_n) + \frac{f''(\xi)}{2}(\alpha - x_n)^2$$

スライド6 収束の速さ p.82

ニュートン法が収束するような区間において. $0 < A \leq |f'(x)|, |f''(x)| \leq B$

となるような定数 A, B を選べば,

$$|x_{n+1} - \alpha| \leq \frac{B}{2A}|x_n - \alpha|^2$$

が成り立つ. これはニュートン法が (収束するならば) 2次収束することを示している.

スライド7 Excel によるグラフ作成

2次関数のグラフ

```
#include <stdio.h>
```

```
#include <stdlib.h>
```

```
int main(void)
```

```
{
```

```
    int i; double x,y;
```

```
    FILE *fout;
```

```
    if((fout=fopen("output.csv","w"))==NULL){
```

```
        printf("ファイルは見つかりません:output.dat ¥n");
```

```
        exit(1);
```

```
    }
```

```
    for(x=-3;x<=3;x+=0.2){
```

```
        y=x*x;
```

```
        fprintf("%lf %lf¥n",x,y);
```

```
    }
```

```
    fclose(fout);
```

```
}
```

output.csv を Excel で開く。データの書かれたセルを選択して、グラフ（散布図）を作成する。

-3	9
-2.8	7.84
-2.6	6.76
-2.4	5.76
-2.2	4.84
-2	4
-1.8	3.24
-1.6	2.56
-1.4	1.96
-1.2	1.44
-1	1
-0.8	0.64
-0.6	0.36
-0.4	0.16
-0.2	0.04
0	0
0.2	0.04
0.4	0.16
0.6	0.36
0.8	0.64
1	1
1.2	1.44
1.4	1.96
1.6	2.56
1.8	3.24
2	4
2.2	4.84
2.4	5.76
2.6	6.76
2.8	7.84
3	9

