ce-12. ニュートン法による方程式の求解

（C プログラミング応用）（全１４回）

URL: https://www.kkaneko.jp/pro/c/index.html

金子邦彦

ニュートン法による求解

ニュートン法

- ニュートン法による非線形方程式 f (x) = 0 の求解

入力：関数 **f** と,
初期近似値 x_0

出力：$f(x) = 0$ の
近似解の1つ

再帰による繰返し計算により，x_1，x_2 … を求める
（手順は次ページ）
⇒　収束すれば，解の近似値が得られる
（注）収束しない場合もありえる

ニュートン法

- 初期近似値 x0 から出発
- 反復公式

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

を繰り返す

- x1，x2，x3 … は，徐々に解に近づいていく（と期待できる）

ニュートン法

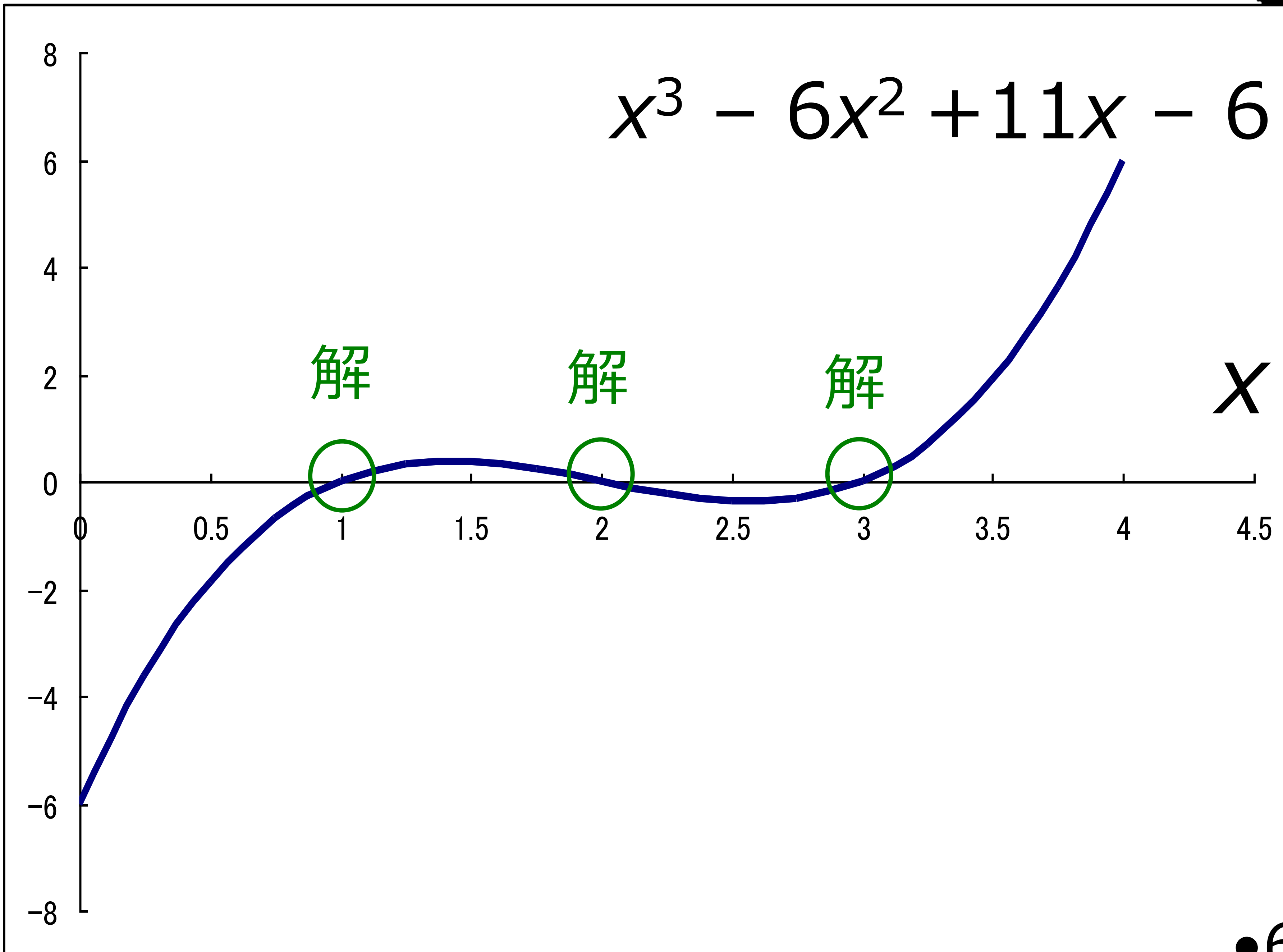
- 初期近似値 x0 から出発
- 反復公式

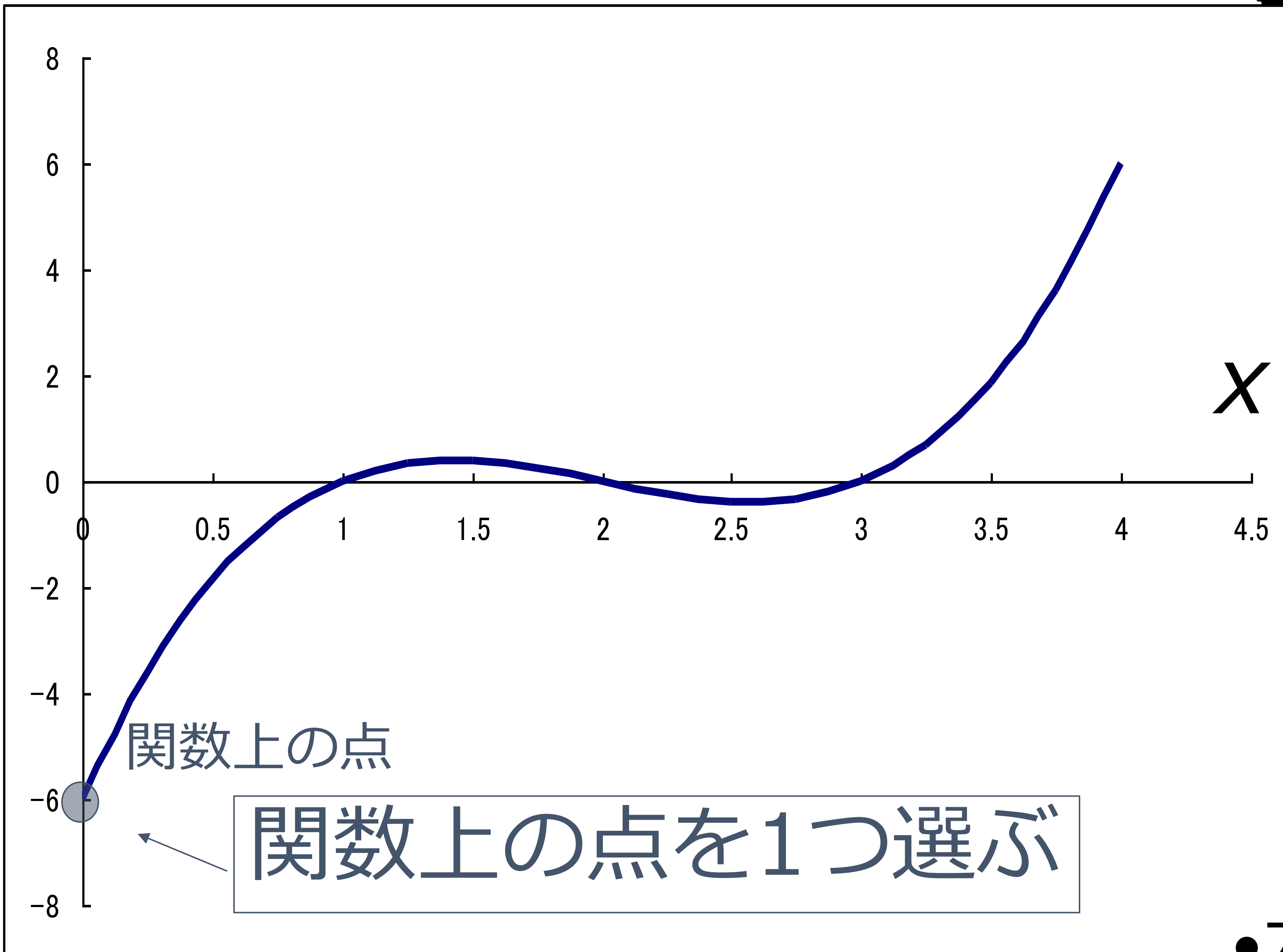
$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

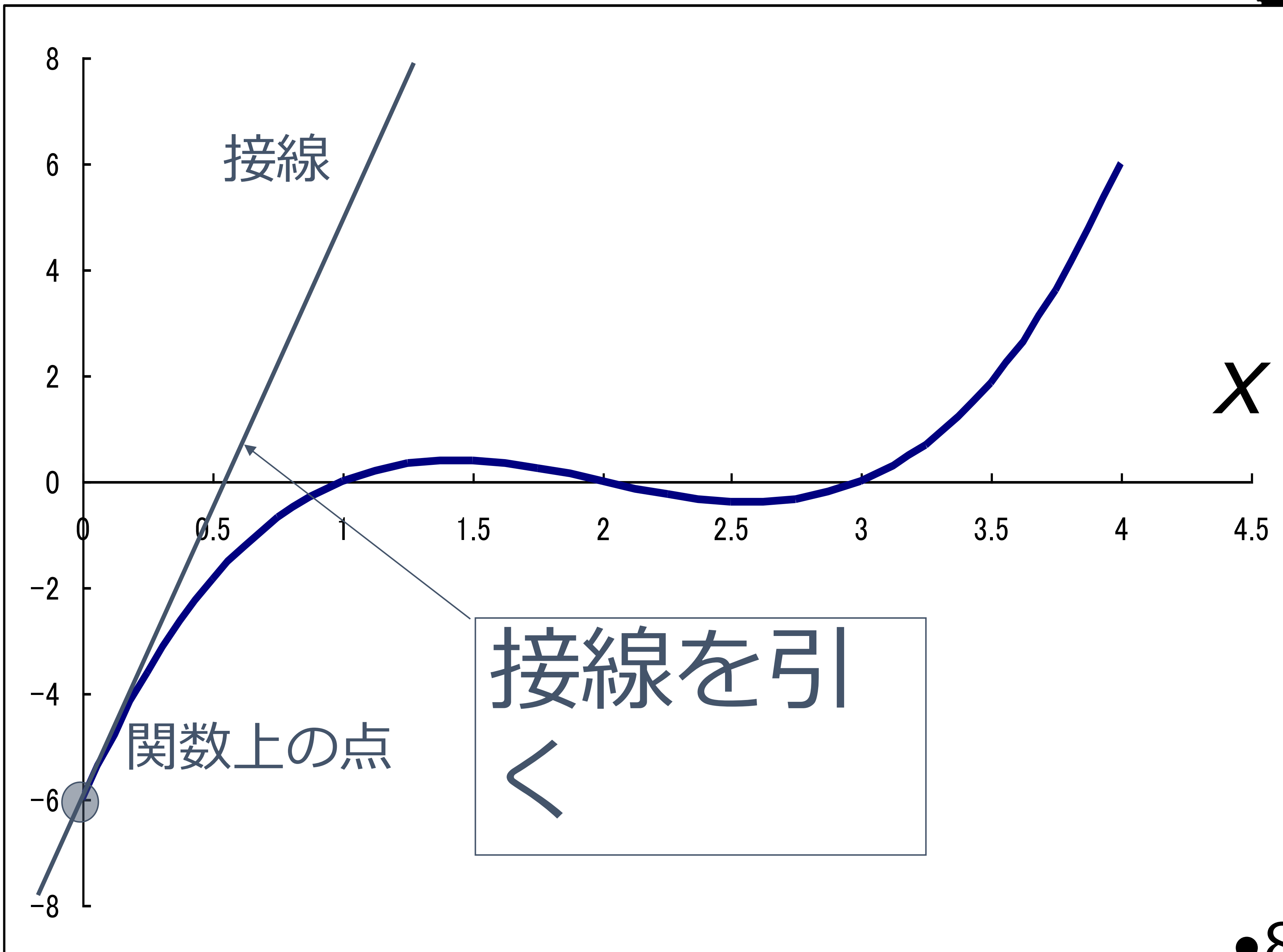
を繰り返す

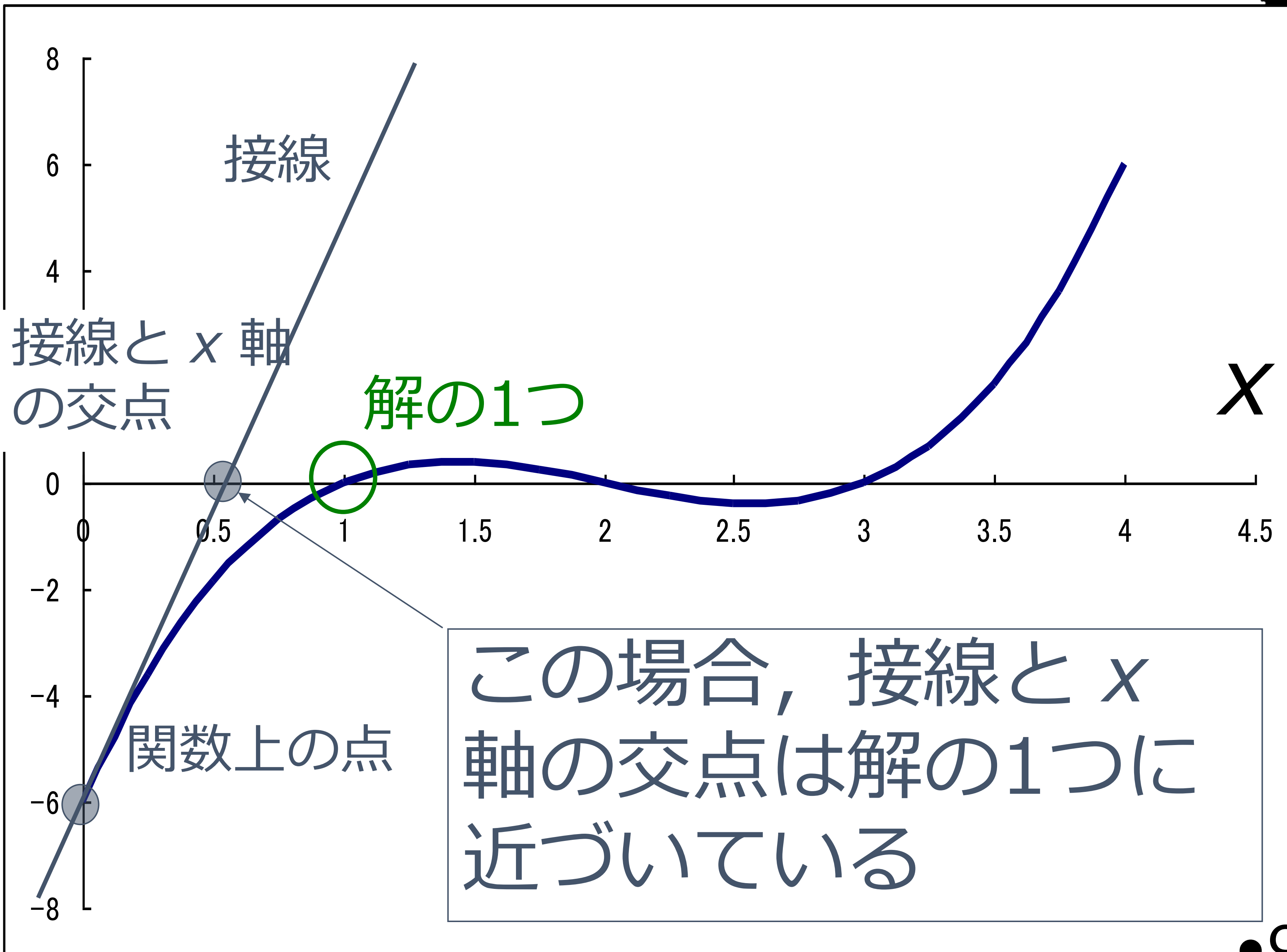
- x1，x2，x3 … は，徐々に解に近づいていく（と期待できる）

これは，点（x_i, $f(x_i)$）における 接線と，x軸（y = 0）との交点（x_{i+1}, 0）を求めてい

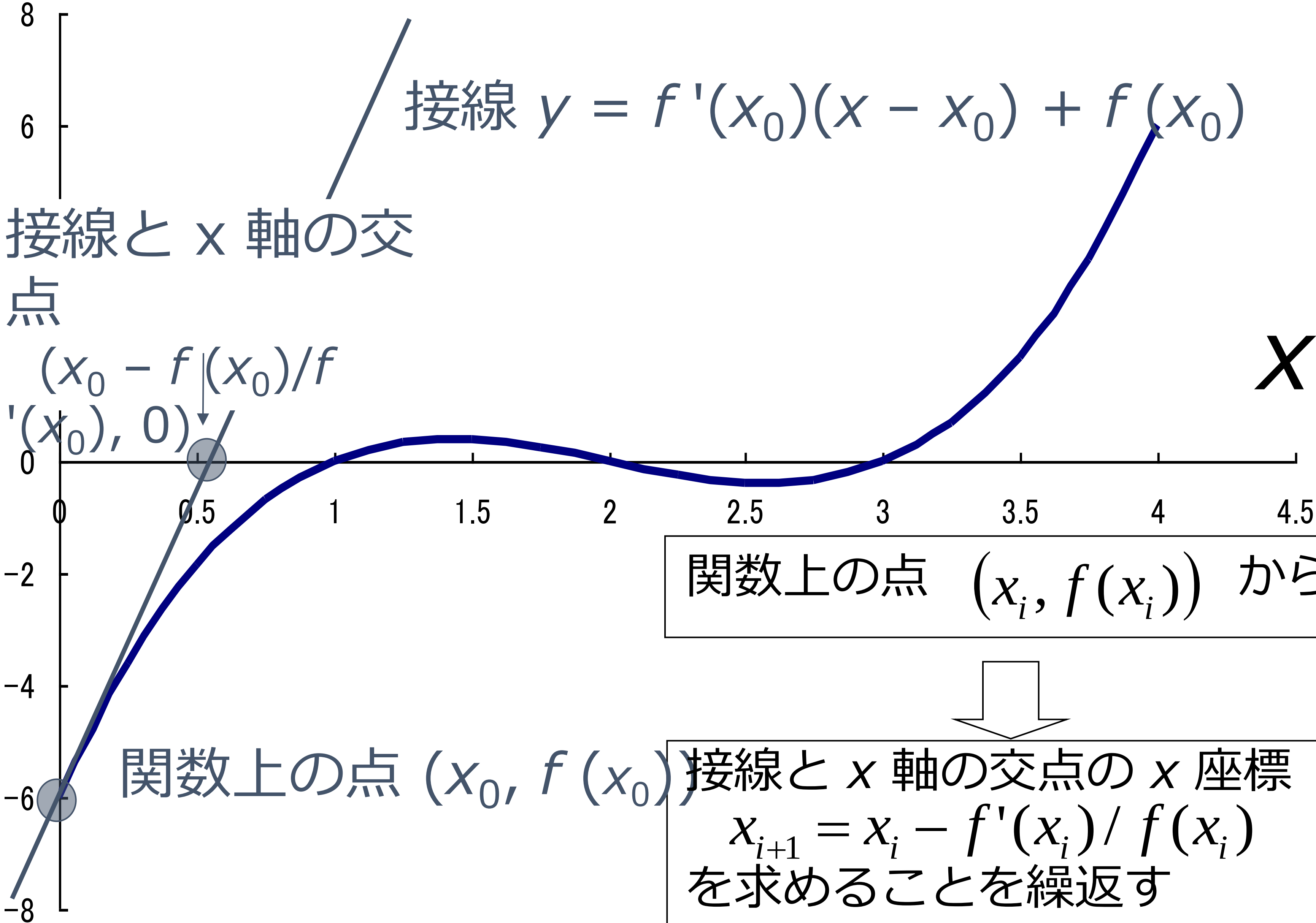

$x^3 - 6x^2 + 11x - 6$
x
解
解
解
8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5


8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
x
関数上の点
関数上の点を1つ選ぶ


8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
接線
x
関数上の点
接線を引く



この場合，接線と x 軸の交点は解の1つに近づいている



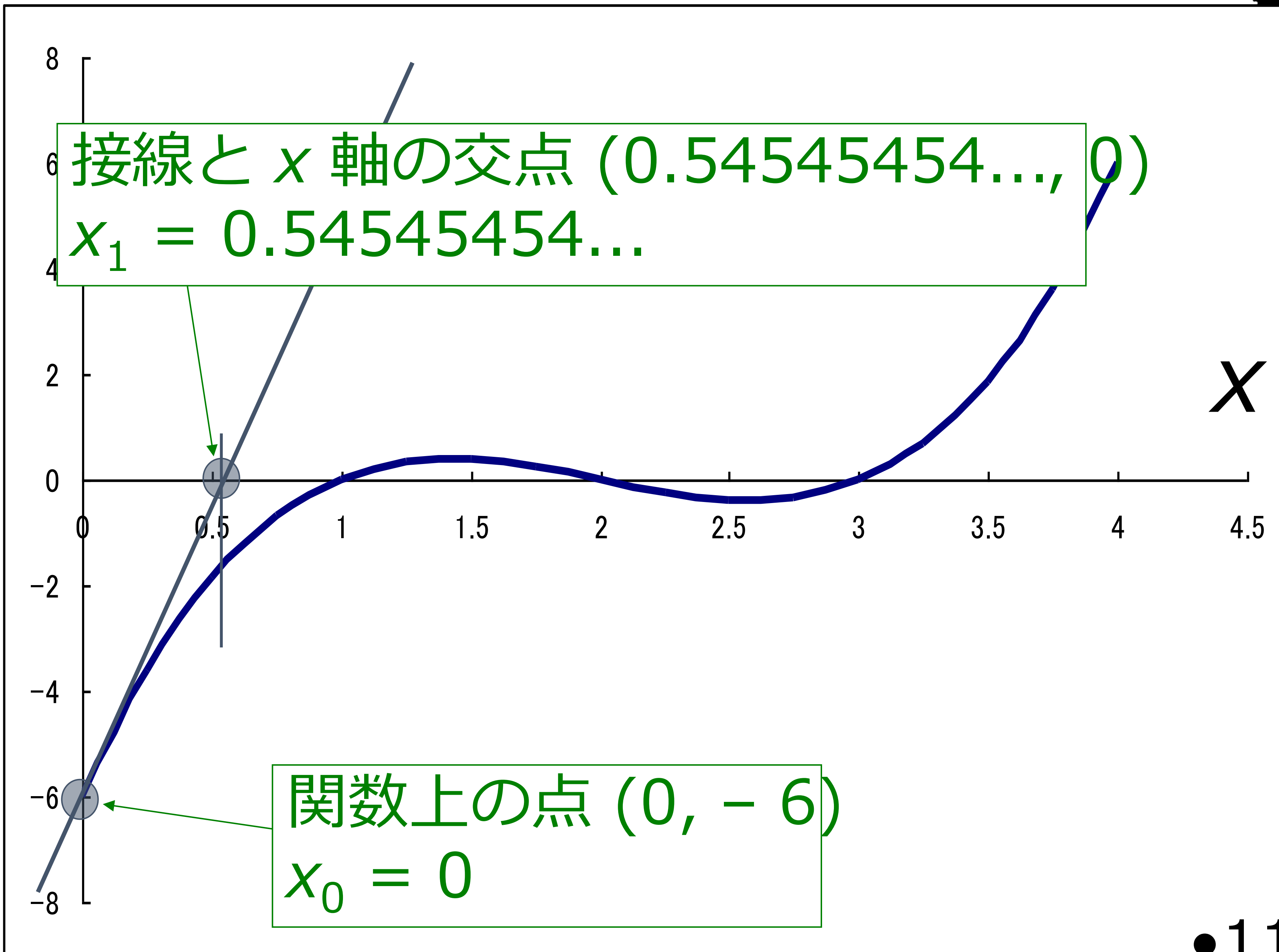
関数上の点 $(x_i, f(x_i))$ から

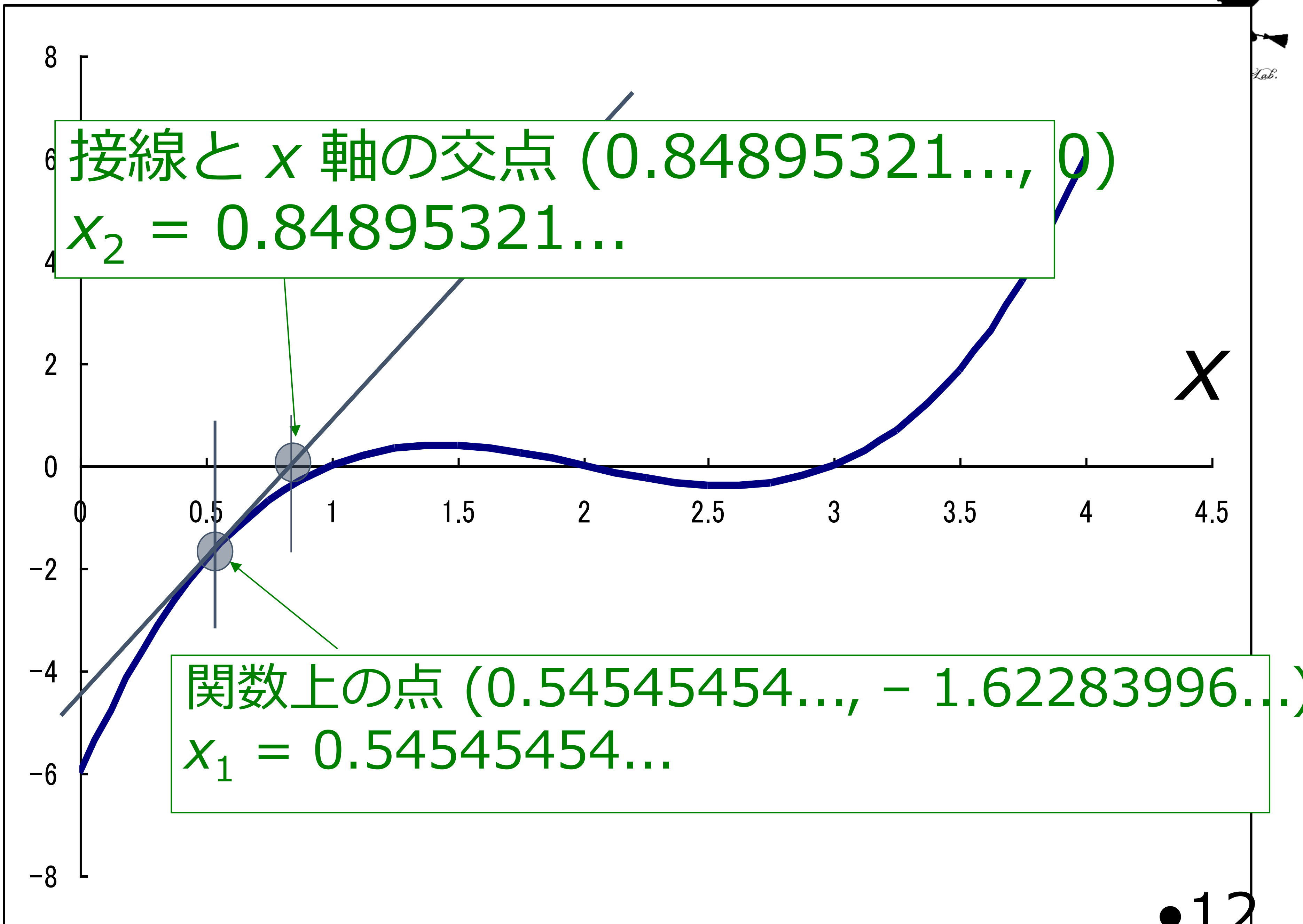
⇓

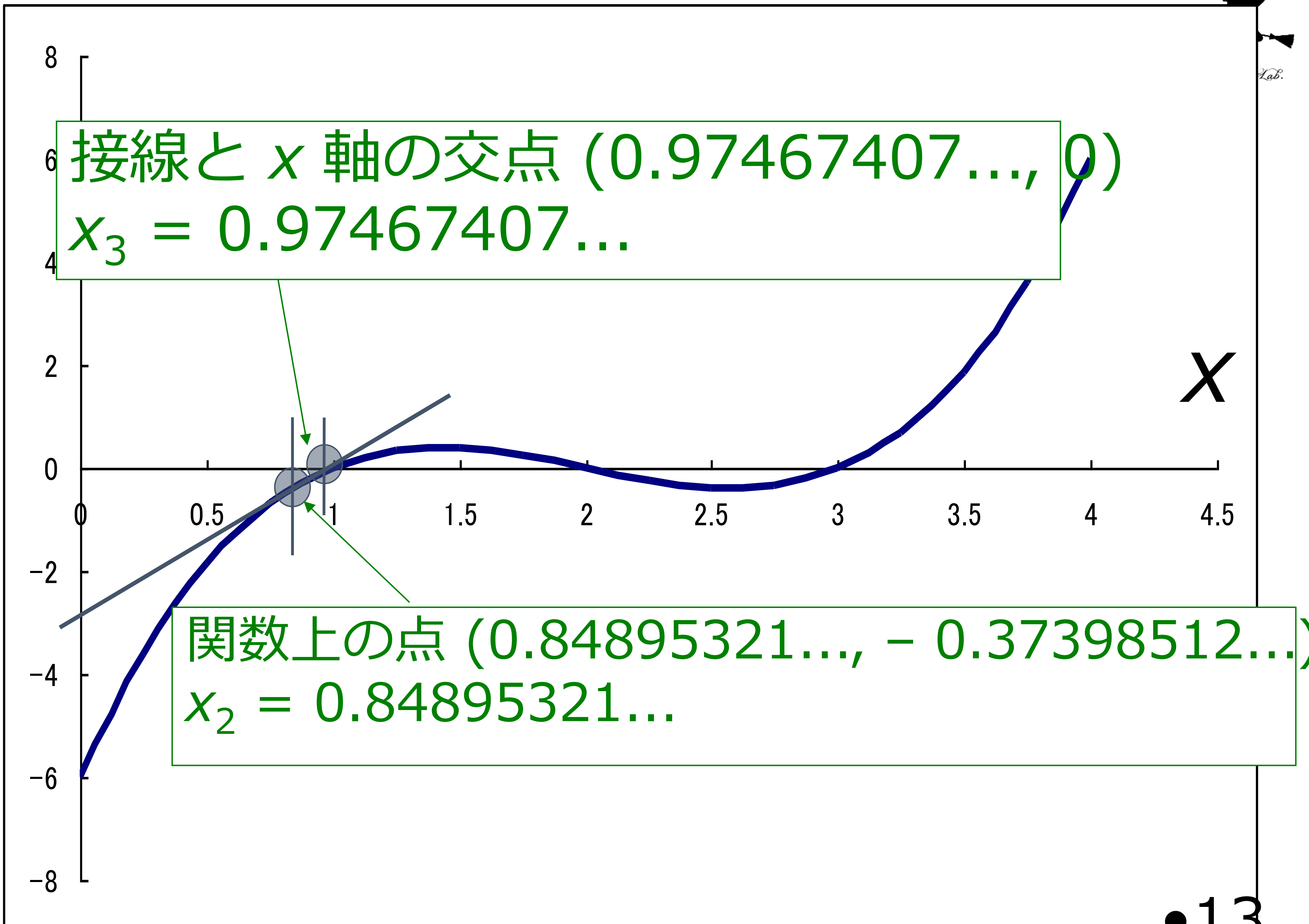
接線と x 軸の交点の x 座標

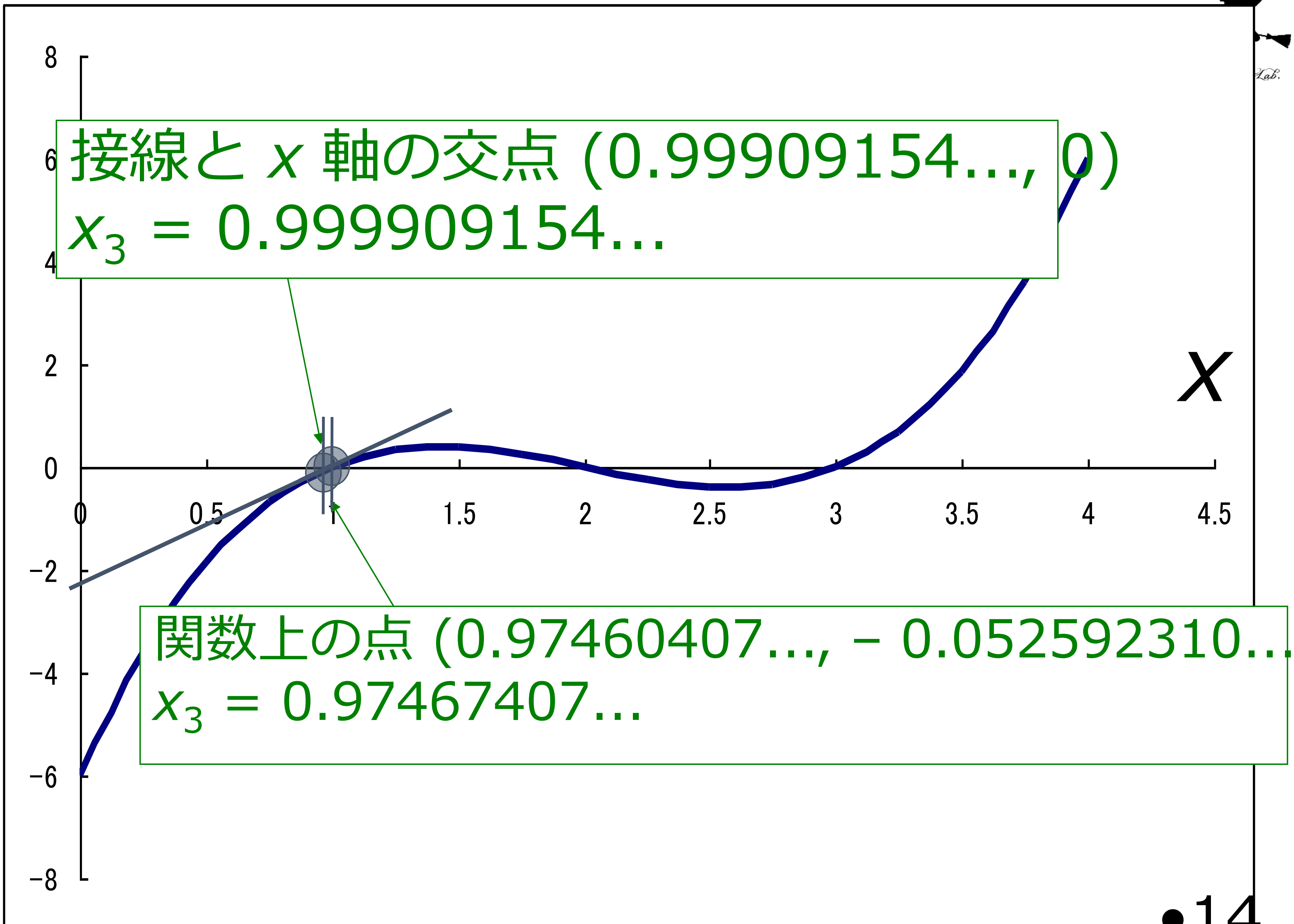
$$x_{i+1} = x_i - f'(x_i) / f(x_i)$$

を求めることを繰返す


接線と x 軸の交点 (0.54545454..., 0)
x_1 = 0.54545454...
関数上の点 (0, − 6)
x_0 = 0
x
8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5


接線と x 軸の交点 (0.84895321…, 0)
x2 = 0.84895321…
関数上の点 (0.54545454…, − 1.62283996…)
x1 = 0.54545454…
x
8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5


接線と x 軸の交点 (0.97467407..., 0)
x3 = 0.97467407...
関数上の点 (0.84895321..., − 0.37398512...)
x2 = 0.84895321...
X
8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5


接線と x 軸の交点 (0.99909154..., 0)
x3 = 0.999909154...
関数上の点 (0.97460407..., – 0.052592310...)
x3 = 0.97467407...
x
8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5

ニュートン法での収束の判定

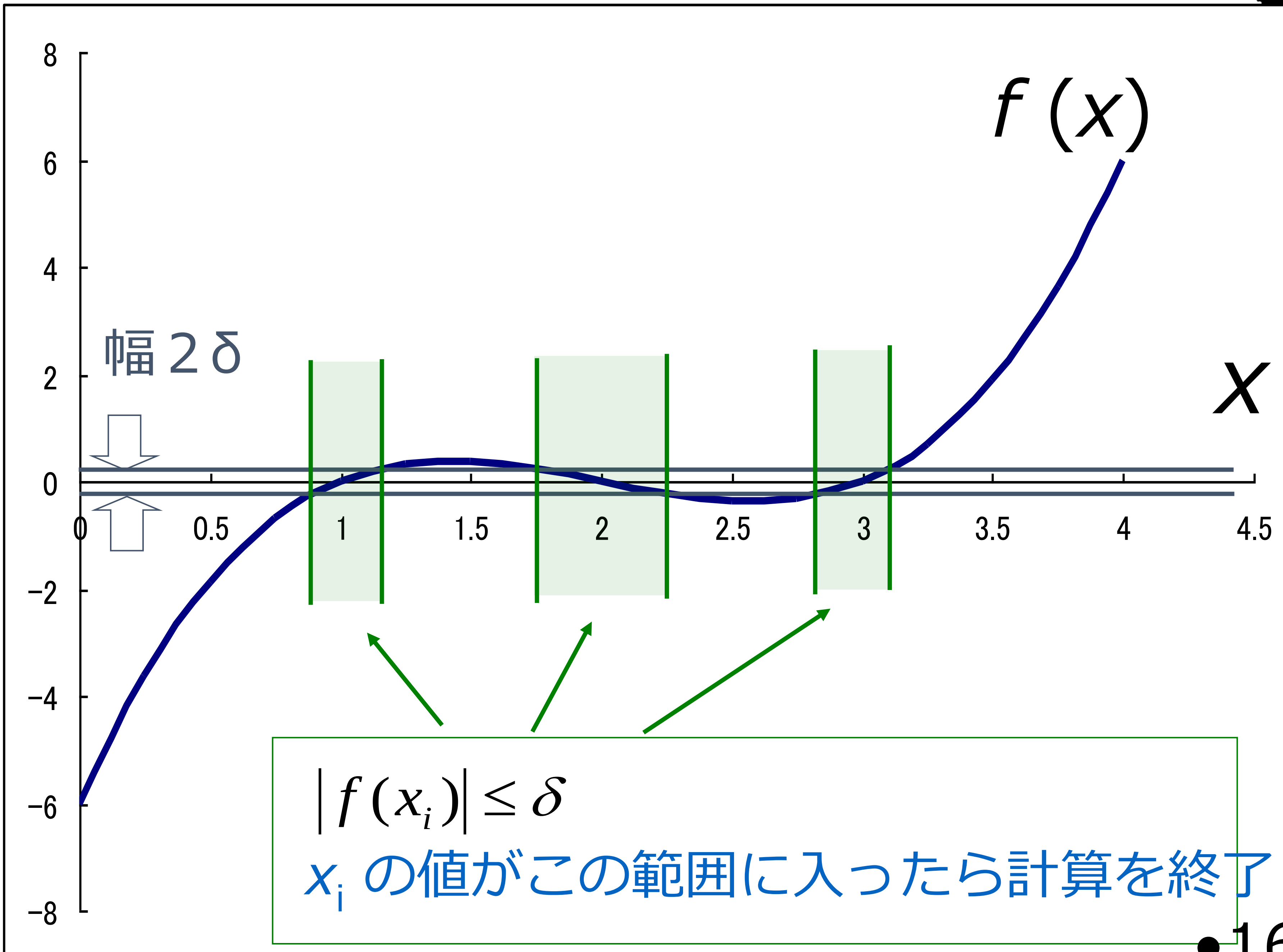
ニュートン法では，現在の x_i の誤差（どれだけ真の解に近いか）は，正確には分からない
（解そのものが分かっていないから）

今日の授業では次の方法で行ってみる

ある小さな正の数δに対して

$$|f(x_i)| \leq \delta$$

となった時点で計算を終了


8
6
4
2
0
−2
−4
−6
−8
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
f (x)
x
幅２δ
$|f(x_i)| \le \delta$
x_i の値がこの範囲に入ったら計算を終了

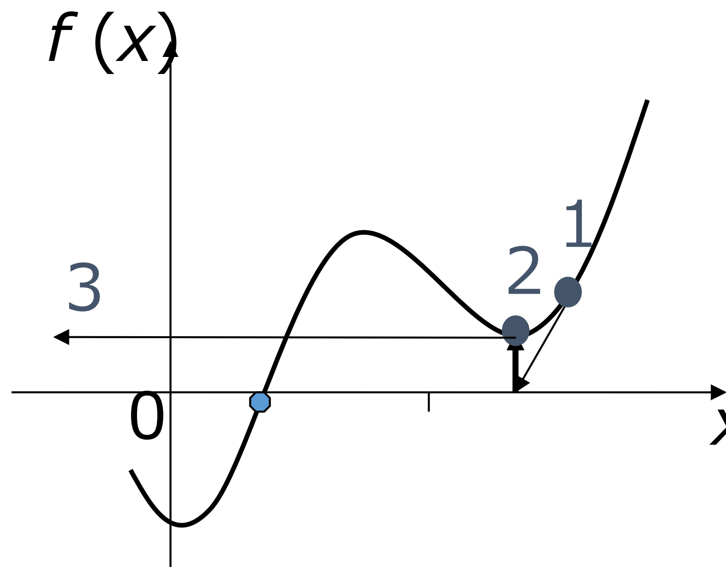
ニュートン法の能力と限界

初期近似値 x_0 で十分近い解を指定できれば，収束が早い.

↕

初期近似値 x_0 の選び方によっては，収束が遅いことがある

収束しないこともありえる
（右図）



関数 f (x) が単調でなくて変曲点を持
（つまり f '(x) の符号が変わる）とき

例）上の図の1から開始
　　2（f '(x) = 0となる点)が選ばれ
　　3　y 軸との交点が求まらない
　　　（負の無限大に発散）
→　収束しない

ニュートン法の注意点

- 虚数解は求まらない
- f (x) = 0 の解が複数あっても，１回に求まる解は１つだけ
- 初期近似値 x0
  - x0 は，コンピュータでなく，人間が決める
  - x0 の値によっては，収束しないこともありえる
  - x0 の値によって，求まる解が変わってくる（f (x) = 0 が複数の解を持つ場合）
- 求まる解は近似解

例題１．ニュートン法のプログラム

- f(x)=x2 –2 をニュートン法で解くプログラム

```
#include "stdio.h"
#include <math.h>
/* f(x) */
double f(double x)
{
  return pow(x,2) - 2;
}
/* f(x)の導関数 */
double g(double x)
{
  return 2 * x;
}
int main()
{
  double x;
  double new_x;
  double delta;
  int i;
  int ch;
  /* 初期値 */
  x = 10;
  /* 収束判定のための delta 値 */
  delta = 0.000001;

  printf("繰り返し回数¥tnew_x¥t¥tf(x)¥t¥tg(x)¥n");
  /* for ... になっているのは 100 回繰り返しても収束しなかったら計算を終わりたいから */
  for(i = 0 ; i < 100 ; i++) {
    new_x = x - f(x) / g(x);
    printf("%2d¥t¥t%lf¥t%lf¥t%lf¥n",i,new_x,f(x),g(x));
    /* fabs は double 型の変数について絶対値を求める関数 */
    if( fabs(f(new_x)) < delta ) {
      printf( "収束した¥n" );
      printf( "解は %lf¥n", new_x);
      break;
    }
    x = new_x;
  }
  printf( "Enter キーを1,2回押してください. プログラムを終了します¥n");
  ch = getchar();
  ch = getchar();
  return 0;
}
```

反復公式

収束したかの判定を
行っている部分

実行結果の例

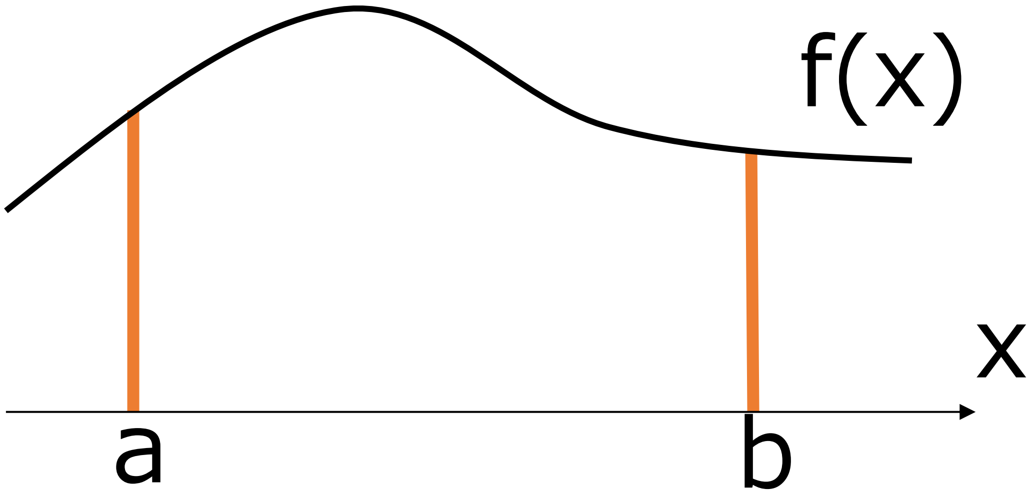
C:¥WINDOWS¥system32¥cmd.exe

```
繰り返し回数      new_x           f(x)            g(x)
 0              5.100000        98.000000       20.000000
 1              2.746078        24.010000       10.200000
 2              1.737195        5.540947        5.492157
 3              1.444238        1.017846        3.474390
 4              1.414526        0.085824        2.888476
 5              1.414214        0.000883        2.829051
収束した
解は 1.414214
Enter キーを1,2回押してください. プログラムを終了します
```

台形則による数値積分

定積分

- 区間 [a, b] で，連続関数 f, x軸, x=a, x=b で囲まれた面積

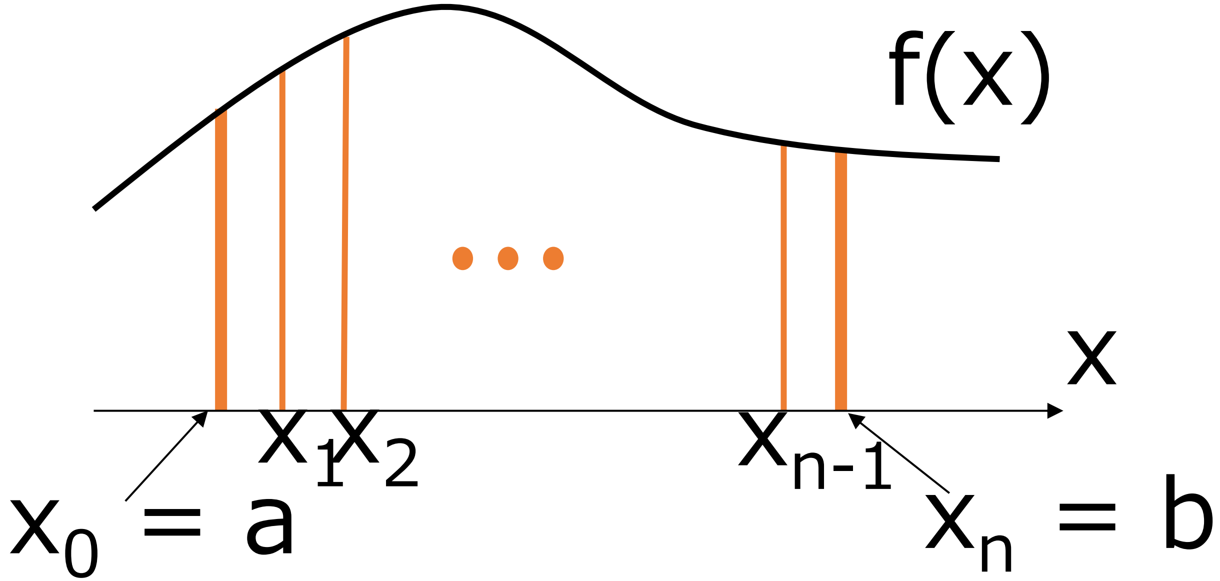


$$定積分：I = \int_a^b f(x)dx$$

区間 [a, b] の小区間への分割

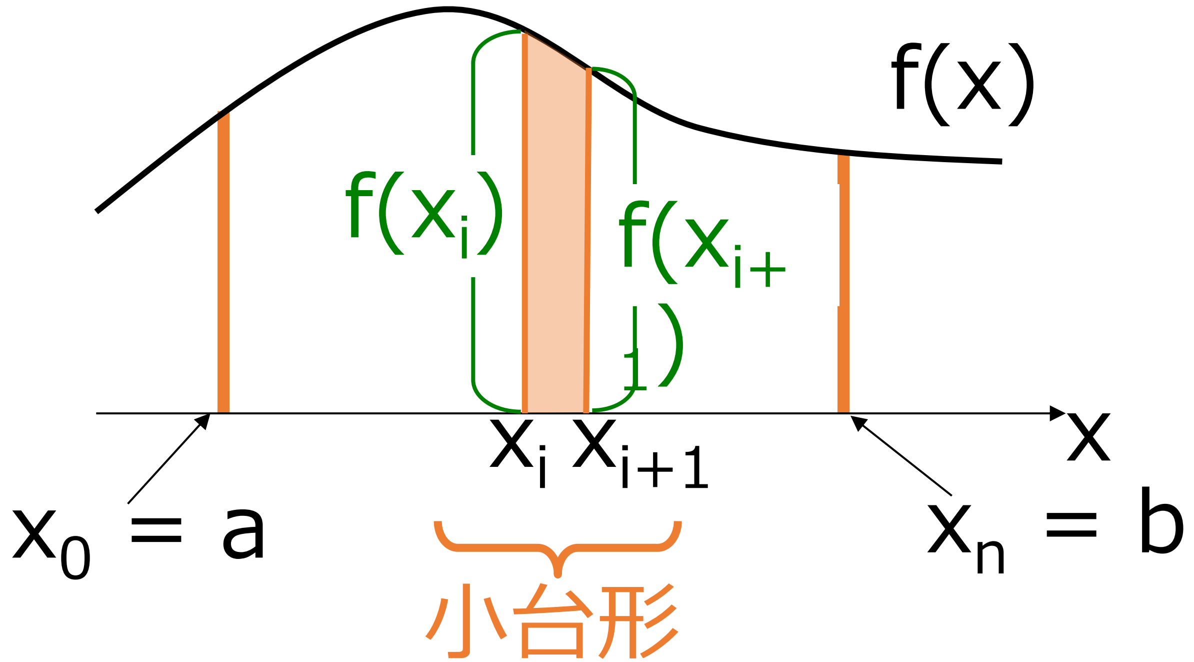
- n 個の等間隔な小区間に分割
  - 幅：　$h = (b-a) / n$
  - 小区間：　[x0, x1], [x1, x2], ..., [xn-1, xn]

但し，x0 = a, xi = x0 + i × h



小台形

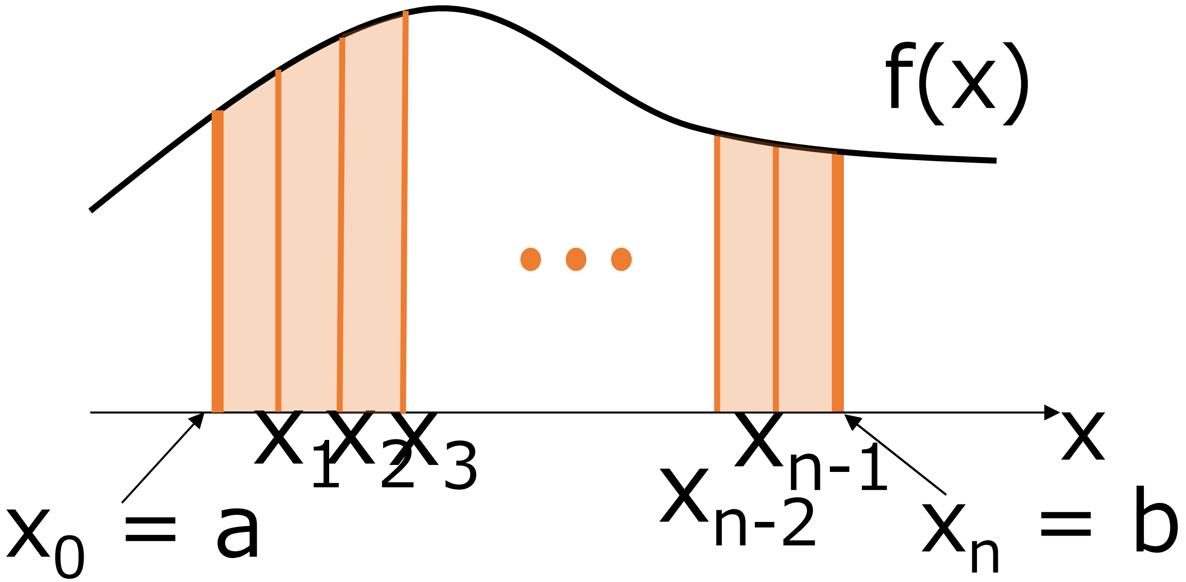
- 小台形の面積は



$$\frac{h}{2}\left(f(x_i) + f(x_{i+1})\right)$$

台形則 trapezoidal rule

- 小台形の面積の和は



- 定積分 I を，この和 Sn で近似 ⇒ 台形則という

$$S_n = \sum_{i=0}^{n-1} \frac{h}{2}(f(x_i) + f(x_{i+1}))$$

台形則

- 両端 x0 = a と xn = b を除いて，f(xi) は2度出現

$$
\begin{aligned}
S_n &= \sum_{i=0}^{n-1} \frac{h}{2}\left(f(x_i) + f(x_{i+1})\right) \\
&= \frac{h}{2}\left(f(x_0) + 2\underbrace{(f(x_1) + \cdots + f(x_{n-1}))}_{\text{2回現れる部分}} + f(x_n)\right) \\
&= h \cdot \left(\frac{1}{2} f(x_0) + \sum_{i=1}^{n-1} f(x_i) + \frac{1}{2} f(x_n) \right) \\
&= h \cdot \left(\frac{1}{2} f(a) + \sum_{i=1}^{n-1} f(a+ih) + \frac{1}{2} f(b) \right)
\end{aligned}
$$

但し　h = (b - a) / n

台形則による数値積分

- 区間［a, b］を n 等分 (1区間の幅h=(b-a)/n)
- n 個の台形を考え, その面積の和 Sn で，定積分 I を近似
  - f(x) が連続関数のときは，n を無限大に近づければ，Sn は I に近づく

$$I = \lim_{n\to\infty}\sum_{i=0}^{n-1}\frac{h}{2}(f(x_i)+f(x_{i+1})) = \lim_{n\to\infty} S_n$$

- 但し，単純に「n を大きくすればよい」とは言えない
  - n を大きくすると ⇒ 計算時間の問題，丸め誤差の問題が発生