アルゴリズムとデータ構造

コンピュータサイエンスコース
知能コンピューティングコース

第9回

アルゴリズムの設計（動的計画法）
数値計算

塩浦昭義

情報科学研究科 准教授

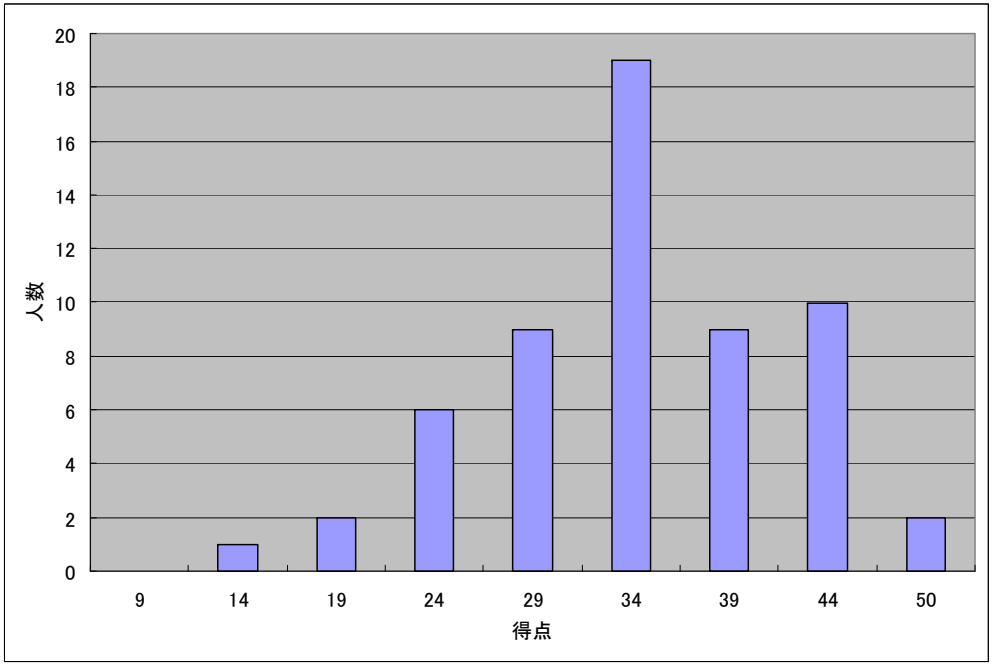
shioura@dais.is.tohoku.ac.jp
http://www.dais.is.tohoku.ac.jp/~shioura/teaching

中間試験の結果

受験者数 58名　合格 49名　不合格 9名

平均点　32．4点／50点満点

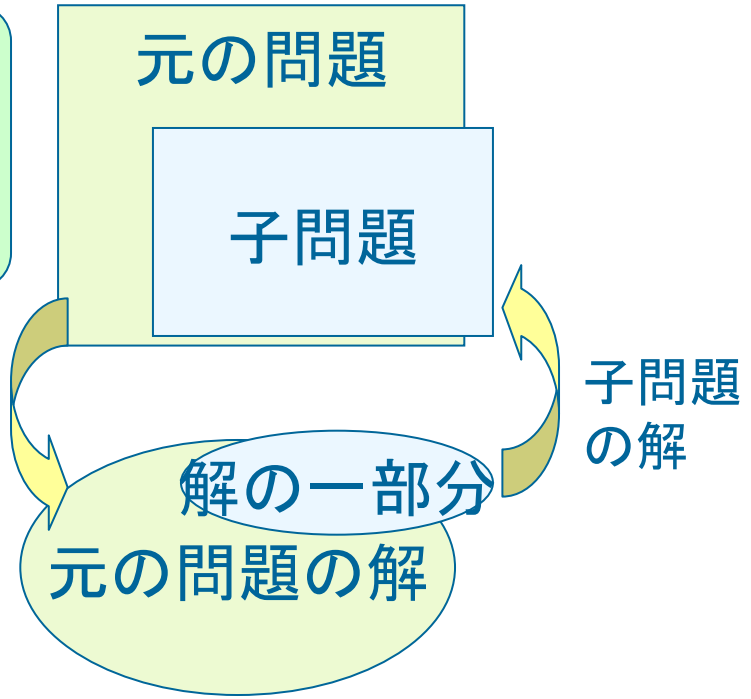
- 19点未満は単位不可決定
- 20点～24点は救済の可能性あり．7／7（水）までに問い合わせること．
- 得点の問い合わせは，出来るだけメイルにて．
- 試験結果の詳細が知りたい場合は，事前にメイルにてアポを取ってください



動的計画法

「最適性の原理」
元の問題の解の部分解は，
元の問題の部分問題の解である

再帰的なアルゴリズム



このアプローチ，もしくはこのアプローチにより得られる
再帰的アルゴリズムを動的計画法という

動的計画法を使って部分和問題と0-1ナップサック問題を解く

部分和問題 (subset-sum problem)

- 入力: n+1 個の正整数 (a_0, a_1, …, a_{n-1}; b)
- 出力: $a_0x_0+a_1x_1+\ldots+a_{n-1}x_{n-1}=b$
  を満たす0-1ベクトルが存在する➔yes, しない➔no

（別の見方: a_0,a_1, …,a_{n-1}の幾つかを足し合わせて bに一致する➔yes, どう足し合わせても一致しない➔no）

例1: $a_0=5$, $a_1=3$, $a_2=2$, b =7 のとき,
$a_0+a_2=b$ が成り立つ ➔ yes

例2: a0=5, a1=3, a2=2, b =6 のとき,
a0, a1, a2をどのように足し合わせても b に一致しない
➔ no

部分和問題の解の性質

部分和問題 $(a_0, a_1, \ldots, a_{n-2}, a_{n-1}; b)$ は

$x_{n-1}=1$ を満たす解をもつ

←→部分和問題 $(a_0, a_1, \ldots, a_{n-2}; b-a_{n-1})$ は解をもつ

$x_{n-1}=0$ を満たす解をもつ

←→部分和問題 $(a_0, a_1, \ldots, a_{n-2}; b)$ は解をもつ

元の問題が解をもつ←→部分問題は解をもつ

最適性の原理

部分和問題 $(a_0, a_1, \ldots, a_{n-2}, a_{n-1}; b)$ の

部分問題 $(a_0, a_1, \ldots, a_{k-1}, a_k; p)$

ただし，$0 \leqq k \leqq n-1$, $0 \leqq p \leqq b$

※ $k=n-1$, $p=b$ のとき，元の問題に一致

部分和問題の解法

解の性質を使うと，部分問題を繰り返し解くことで
元の問題の解を得ることが可能

部分問題(a_0; p) を 0≦ p ≦b について解く

部分問題(a_0, a_1; p) を 0≦ p ≦b について解く

部分問題(a_0,…, a_{n-2}; p) を 0≦ p ≦b について解く

部分問題(a_0,…, a_{n-2}, a_{n-1}; p) を0≦ p ≦bについて解く

一つの部分問題は定数時間で解ける➔時間計算量O(nb)

部分和問題の解法の例

部分和問題(3,7,5,8,2; 11) を解く

部分問題(3; p) を 0≦ p ≦11 について解く

k＼p	0	1	2	3	4	5	6	7	8	9	10	11
0	○	×	×	○	×	×	×	×	×	×	×	×
1												
2												
3												
4												

p＝0 のときは
常に
解をもつ

p＝a_0 のときは
解をもつ
それ以外は
解をもたない

部分和問題の解法の例

部分和問題(3,7,5,8,2; 11) を解く

部分問題(3, 7; p) を 0≦ p ≦11 について解く

k＼p	0	1	2	3	4	5	6	7	8	9	10	11
0	○	×	×	○	×	×	×	×	×	×	×	×
1	○	×	×	○	×	×	×	○	×	×	○	×
2												
3												
4												

(a_0; p)が解をもつならば,
(a_0,a_1; p)も解をもつ

(a_0; p-a_1)が解をもつならば,
(a_0,a_1; p)も解をもつ

部分和問題の解法の例

部分和問題(3,7,5,8,2; 11) を解く

k＼p	0	1	2	3	4	5	6	7	8	9	10	11
0	○	×	×	○	×	×	×	×	×	×	×	×
1	○	×	×	○	×	×	×	○	×	×	○	×
2	○	×	×	○	×	○	×	○	○	×	○	×
3	○	×	×	○	×	○	×	○	○	×	○	○
4	○	×	○	○	×	○	×	○	○	○	○	○

0-1ナップサック問題と部分問題

- 0-1ナップサック問題（入力は全て正の整数）

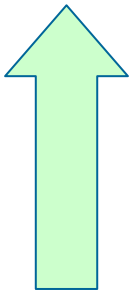
$$\text{最大化} \quad \sum_{j=1}^{n} c_j x_j$$

$$\text{条件} \quad \sum_{j=1}^{n} a_j x_j \leq b$$

$$x_j \in \{0,1\} \ (j = 1,2,\ldots,n)$$

元の問題の最適値

$f_n(b)$



- 部分問題（r = 1, 2,..., n, λ = 0, 1, ..., b)

$$(P_r(\lambda)) \quad \text{最大化} \quad \sum_{j=1}^{r} c_j x_j$$

$$\text{条件} \quad \sum_{j=1}^{r} a_j x_j \leq \lambda$$

$$x_j \in \{0,1\} \ (j = 1,2,\ldots,r)$$

この部分問題の最適値

$f_r(\lambda)$

0-1ナップサック問題の最適解の性質

- 部分問題(r = 1, 2,..., n, λ = 0, 1, ..., b)

$$(P_r(\lambda)) \quad \text{最大化} \quad \sum_{j=1}^{r} c_j x_j$$

$$\text{条件} \quad \sum_{j=1}^{r} a_j x_j \leq \lambda$$

$$x_j \in \{0,1\} \ (j = 1,2,\ldots,r)$$

この部分問題の最適値

$$f_r(\lambda)$$

最適性の原理

- $P_r(\lambda)$の最適解 x^* において $x^*_r = 0$ ならば
  $(x^*_1, \ldots, x^*_{r-1})$は$P_{r-1}(\lambda)$の最適解

$$f_r(\lambda) = f_{r-1}(\lambda)$$

- $P_r(\lambda)$の最適解 x^* において $x^*_r = 1$ ならば
  $(x^*_1, \ldots, x^*_{r-1})$は$P_{r-1}(\lambda - a_r)$の最適解

$$f_r(\lambda) = f_{r-1}(\lambda - a_r) + c_r$$

0-1ナップサック問題に関する再帰式

$$x_r^* = 0 \Longrightarrow f_r(\lambda) = f_{r-1}(\lambda)$$
$$x_r^* = 1 \Longrightarrow f_r(\lambda) = f_{r-1}(\lambda - a_r) + c_r$$

$$f_r(\lambda) = \max\{f_{r-1}(\lambda), f_{r-1}(\lambda - a_r) + c_r\}$$

ただし $f_1(\lambda) = \begin{cases} 0 & (0 \le \lambda < a_1) \\ c_1 & (a_1 \le \lambda \le b) \end{cases}$

動的計画法

上記の再帰式をつかって,

$f_2(0), f_2(1), \ldots, f_2(b),\ f_3(0), f_3(1), \ldots, f_3(b),$
$\ldots,\ f_n(0), f_n(1), \ldots, f_n(b)$
を順に計算 $\Longrightarrow$ 元問題の最適値が得られる

計算時間
O(nb)

0-1ナップサック問題の最適解の計算

- 最適解も再帰的に計算可能

$$f_r(\lambda) = \max\{f_{r-1}(\lambda), f_{r-1}(\lambda - a_r) + c_r\}$$

ここで

$$x_r^* = 0 \iff f_r(\lambda) = f_{r-1}(\lambda)$$
$$x_r^* = 1 \iff f_r(\lambda) = f_{r-1}(\lambda - a_r) + c_r$$

よって,

$f_r(\lambda) = f_{r-1}(\lambda)$
ならば $P_r(\lambda)$ の最適解は $(x_1^*, \ldots, x_{r-1}^*, 0)$
ここで $(x_1^*, \ldots, x_{r-1}^*)$ は $P_{r-1}(\lambda)$ の最適解

$f_r(\lambda) = f_{r-1}(\lambda - a_r) + c_r$
ならば $P_r(\lambda)$ の最適解は $(x_1^*, \ldots, x_{r-1}^*, 1)$
ここで $(x_1^*, \ldots, x_{r-1}^*)$ は $P_{r-1}(\lambda - a_r)$ の最適解

数値計算

- 数値計算の問題---所望の数値を計算する問題
  - 連立一次方程式や非線形方程式の解の計算
  - 微分不等式の解の計算
  - 行列の固有値の計算
- 答え，および計算の途中で表れる数値は一般の場合，実数（複素数の場合もあり）

  ➔ 数値誤差を考慮する必要がある
  - コンピュータ上で無理数を厳密に表現することは不可能
  - 有理数でも表現可能な桁数に制限有り

    (例：1/3＝0.333333 と途中で打ち切り)

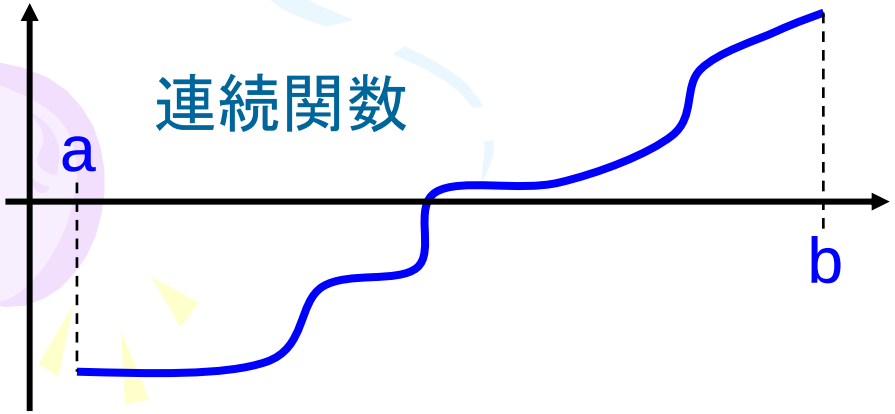
数値計算のアルゴリズム

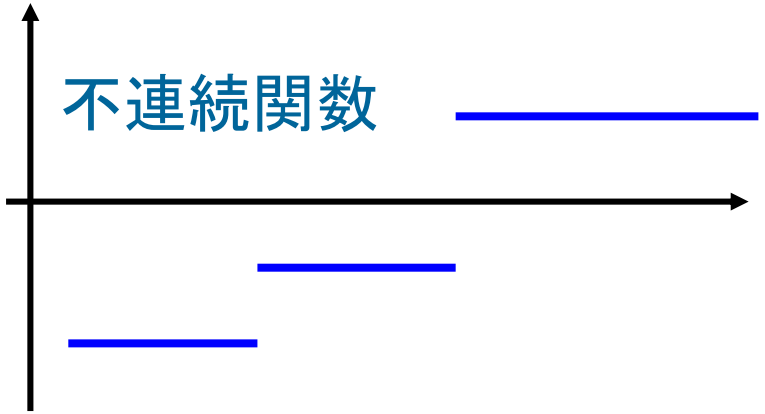
- 数値計算のアルゴリズムの要件
  - アルゴリズムがわかりやすい，プログラムを作りやすい
  - 計算時間が理論・実用の両面で少ない
  - 空間計算量が少ない
  - 計算結果の誤差が小さい
  - 計算結果の誤差の範囲が保証されている

単調関数の零点を求める

- 授業で扱う問題
  - 単調非減少な連続関数の零点を求める
  - x は関数 f の零点 ←→ f(x) = 0

中間値定理: 一変数の連続関数 f と実数 a, b（ただし a < b）が f(a) < 0 < f(b) を満たす
➔区間 [a, b] において関数 f は零点をもつ
（∃x*∈[a, b]: f(x*) = 0）

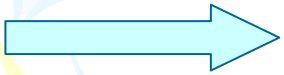




単調関数の零点を求める問題

零点を求める問題（数学版）
入力: 一変数の関数 f, 実数 a, b, ただし $a < b$, $f(a) < 0 < f(b)$
区間 [a, b] において関数 f は単調非減少かつ連続
出力: 関数 f の零点 $x \in [a, b]$

しかし，零点は実数値➔厳密に計算することは困難
零点であること ($f(x)=0$) を厳密に判定することも困難



誤差を許すことにする（近似解で我慢する）

零点を求める問題（数値計算版）
入力: 一変数の関数 f, 実数 a, b, ただし $a < b$, $f(a) < 0 < f(b)$
区間 [a, b] において関数 f は単調非減少かつ連続
正の実数 ε , δ
出力: 関数 f の零点 x^* に対して $|x - x^*| \leqq \varepsilon$ を満たす $x \in [a, b]$
もしくは $|f(x)| \leqq \delta$ を満たす $x \in [a, b]$

零点の計算に対する二分探索

- 二分探索により零点の近似解は計算可能

(1): $x_L := a$, $x_R := b$ とおく.

(2): $x_M := (i_L + i_R)/2$ とおく.

(3): $|f(x_M)| \leq \delta$ ならば終了. x を出力.

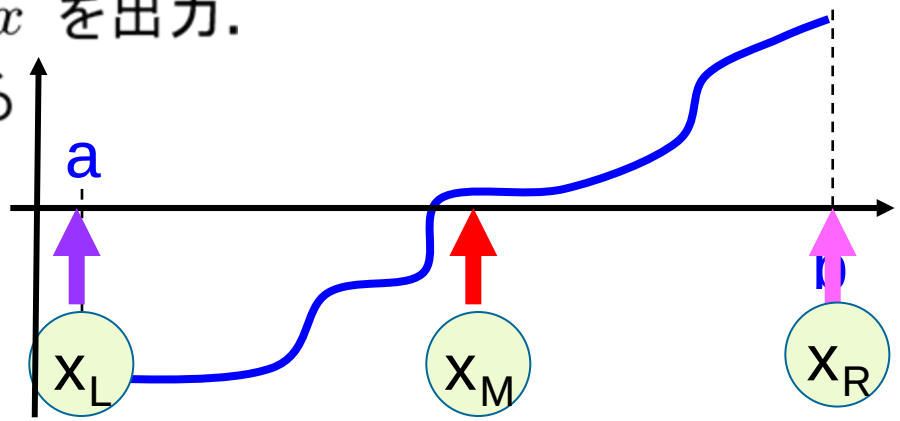
$f(x_M) > \delta$ ならば $x_R := x_M$ とおく.

$f(x_M) < -\delta$ ならば $x_L := i_M$ とおく.

(4): $x_R - X_L \leq \epsilon$ ならば終了. x を出力.

$x_R - X_L > \epsilon$ ならば **(2)** に戻る

常に
$f(x_L) < 0 < f(x_R)$
を満たす



零点の計算に対する二分探索

- 二分探索により零点の近似解は計算可能

(1): $x_L := a$, $x_R := b$ とおく.

(2): $x_M := (i_L + i_R)/2$ とおく.

(3): $|f(x_M)| \leq \delta$ ならば終了. x を出力.

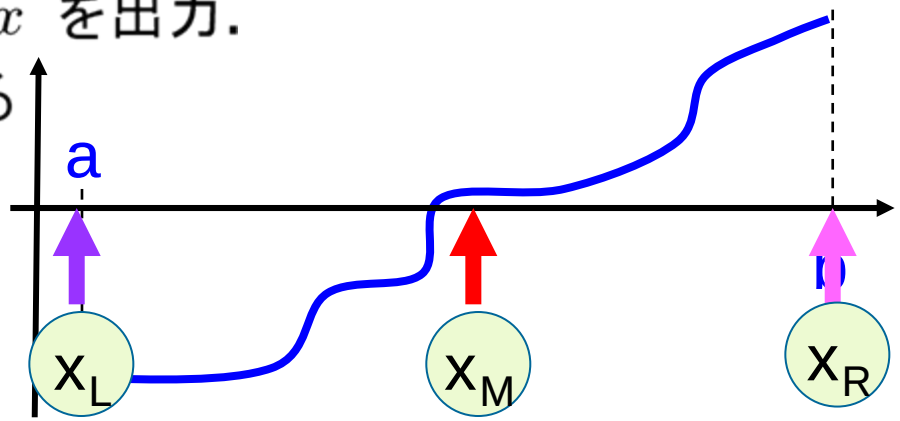
$f(x_M) > \delta$ ならば $x_R := x_M$ とおく.

$f(x_M) < -\delta$ ならば $x_L := i_M$ とおく.

(4): $x_R - X_L \leq \epsilon$ ならば終了. x を出力.

$x_R - X_L > \epsilon$ ならば **(2)** に戻る

常に
$f(x_L) < 0 < f(x_R)$
を満たす



零点の計算に対する二分探索

- 二分探索により零点の近似解は計算可能

(1): $x_L := a$, $x_R := b$ とおく.

(2): $x_M := (i_L + i_R)/2$ とおく.

(3): $|f(x_M)| \leq \delta$ ならば終了. x を出力.

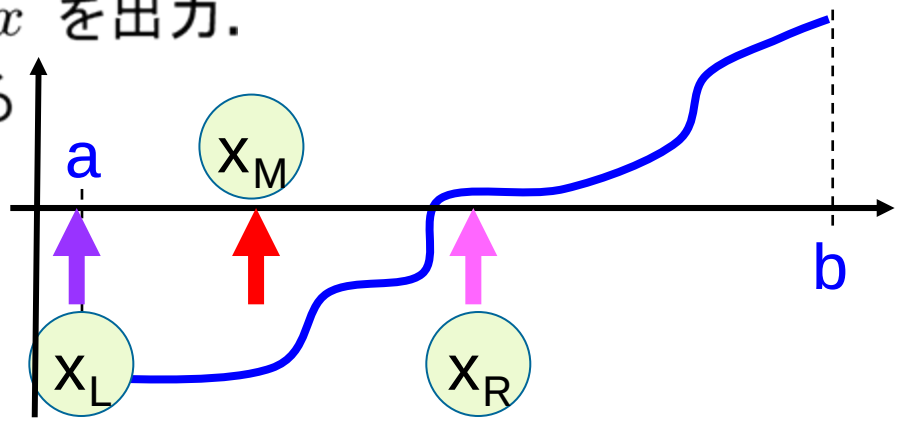
$f(x_M) > \delta$ ならば $x_R := x_M$ とおく.

$f(x_M) < -\delta$ ならば $x_L := i_M$ とおく.

(4): $x_R - X_L \leq \epsilon$ ならば終了. x を出力.

$x_R - X_L > \epsilon$ ならば **(2)** に戻る

常に
$f(x_L) < 0 < f(x_R)$
を満たす



零点の計算に対する二分探索

- 二分探索により零点の近似解は計算可能

(1): $x_L := a$, $x_R := b$ とおく.

(2): $x_M := (i_L + i_R)/2$ とおく.

(3): $|f(x_M)| \leq \delta$ ならば終了. x を出力.

$f(x_M) > \delta$ ならば $x_R := x_M$ とおく.

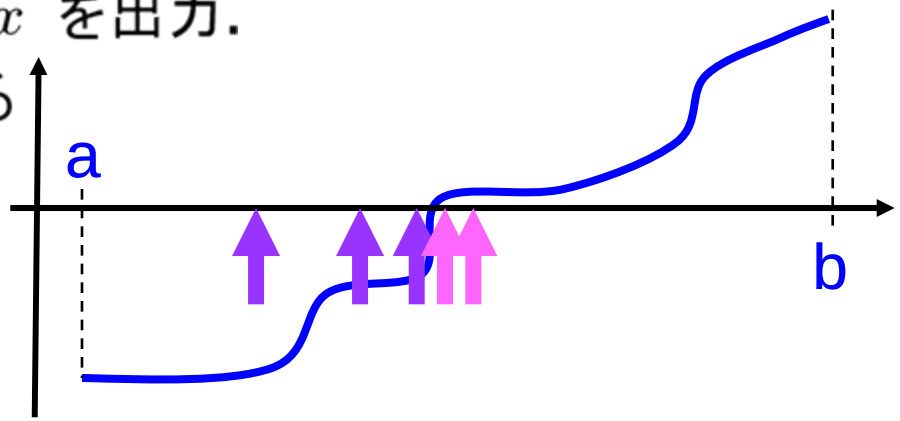
$f(x_M) < -\delta$ ならば $x_L := i_M$ とおく.

(4): $x_R - X_L \leq \epsilon$ ならば終了. x を出力.

$x_R - X_L > \epsilon$ ならば **(2)** に戻る

常に
$f(x_L) < 0 < f(x_R)$
を満たす

x_L と x_R が
十分近くなったら
終了



零点の計算に対する二分探索の時間計算量

- 各反復で $x_R - x_L$ は半分に減少
- 初期値は $b - a$
- $x_R - x_L \leqq \varepsilon$ となったら終了

➔反復回数を k とすると,

$$(b-a)\left(\frac{1}{2}\right)^{k-1} > \varepsilon \geq (b-a)\left(\frac{1}{2}\right)^{k}$$

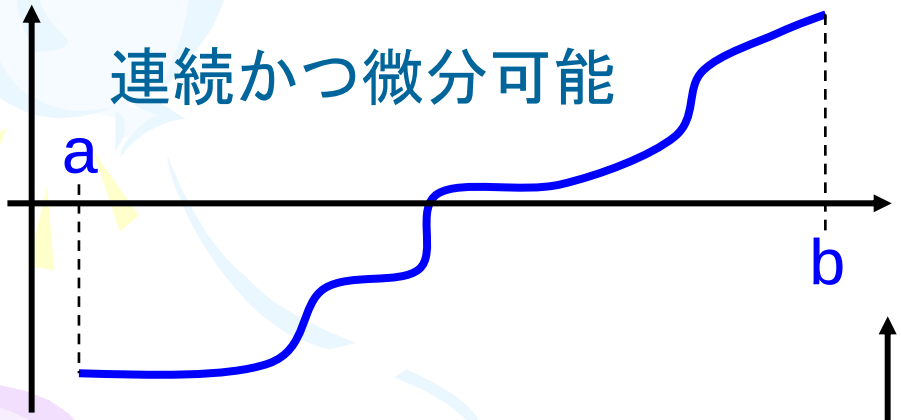
$$\therefore\ k = \left\lceil \log_2 \frac{b-a}{\varepsilon} \right\rceil$$

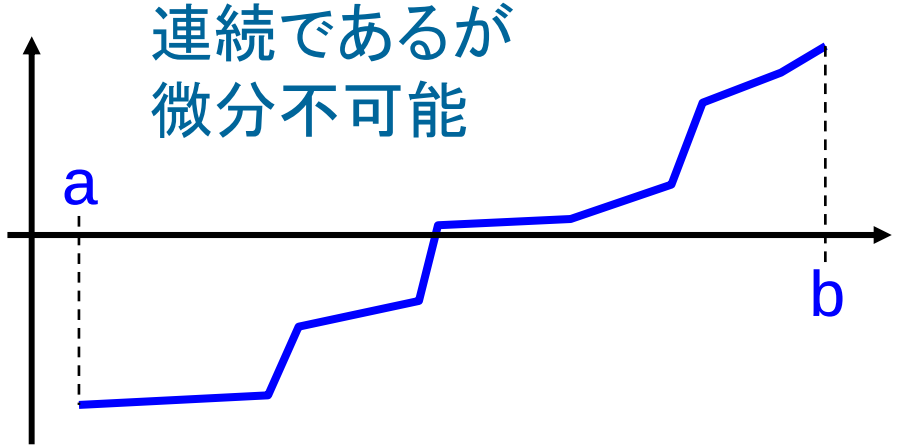
関数値 $f(x)$ の計算が定数時間で出来るならば,

時間計算量は $O\left(\log_2 \frac{b-a}{\varepsilon}\right)$

零点の計算に対するニュートン法

- 関数 f が単調増加かつ微分可能ならば，ニュートン法が利用可能
  - f が単調増加 ⇔ x < y ならば f(x) < f(y)



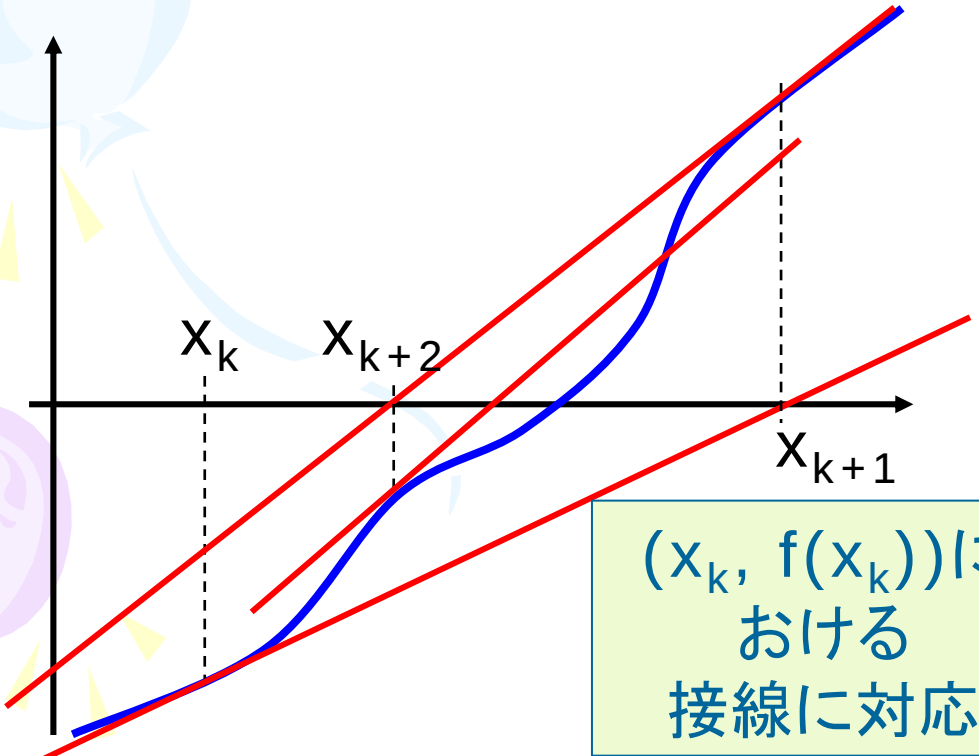


ニュートン法の考え方

- 現在の $x = x_k$ において関数を一次近似

$$f(x) \Rightarrow f(x_k) + f'(x_k)(x - x_k)$$

- 近似した関数の零点を次の $x = x_{k+1}$ とおく



$(x_k, f(x_k))$における接線に対応

x_kとx_{k+1}は次の式を満たす

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

アルゴリズムでは，この式を使って繰り返しx_kを更新

ニュートン法の収束性

- ニュートン法のように，解候補 x_k を繰り返し生成するアルゴリズムの性能は，

  解候補が解 x^* に収束する速さ

  によって評価
  - 正の定数 α が存在して，各反復で
    $|x_{k+1} - x^*| \leqq \alpha |x_k - x^*|$ が成り立つ➔一次収束
  - 正の定数 α が存在して，各反復で
    $|x_{k+1} - x^*| \leqq \alpha |x_k - x^*|^2$ が成り立つ➔二次収束
  - 二次収束の方が一次収束より速い
  - ニュートン法は（ある仮定の下で）二次収束

レポート問題1（締切：7月1日（木））

最大化　$10x_1 + 7x_2 + 25x_3 + 24x_4$

条件　$2x_1 + x_2 + 6x_3 + 5x_4 \leq 7$

$x_j \in \{0, 1\}\ (j = 1, 2, 3, 4)$

この0－1ナップサック問題を
動的計画法で解きなさい

λ	0	1	2	3	4	5	6	7
f_1	0	0	10	10	10	10	10	10
f_2								
f_3								
f_4								

レポート問題2

ニュートン法に関するレポート問題

（1）関数 $f(x) = (x+2)x(x-1) = x^3+x^2-2x$を図に書きなさい.

x の範囲は[-2.5, 2] くらいにすること. 適当なソフトウェアを使って書いても良い.

（2）x＝-1からスタートしたときのニュートン法の実行例を図に書きなさい.

（3）x＝2からスタートしたときのニュートン法の実行例を図に書きなさい.

（2），（3）はともに数値を正確に計算する必要はなく，図を使って計算すれば十分である.