アルゴリズム 65－1
（1998. 10. 28）

クラス間分散最大区間を求めるアルゴリズムと多次元への拡張

浅野哲夫[1], 加藤直樹[2], 徳山豪[3]

[1] 北陸先端科学技術大学院大学情報科学研究科
[2] 京都大学大学院工学研究科
[3] 日本IBM東京基礎研究所

n 個の実数の重み $w_1, w_2, \ldots, w_n$ が一次元配列に与えられている。このとき $[1,n]$ の部分区間 I に属する重み集合と $[1,n]-I$ に属する重み集合のクラス間分散を最大にする区間 I を求める問題を解く $O(n \log n)$ 時間アルゴリズムを提案する。さらにこのアルゴリズムを二次元配列の問題に拡張する。つまり，実数の重みが $N \times N$ の二次元配列に与えられた場合に、クラス間分散を最大にする矩形領域を求める問題に対する $O(N^3)$ 時間アルゴリズムを提案する。

An algorithm for finding an interval with maximum interclass variance and its extension to higher dimensions

Tesuo Asano[1], Naoki Katoh[2] and Takeshi Tokuyama[3]

[1] School of Information Science, JAIST, Asahidai, Tatsunokuchi, 923-1292 JAPAN.
email:t-asano@jaist.ac.jp

[2] Department of Architecture and Architectural Systems, Kyoto University
Yoshida-Honmachi, Kyoto, 606-8501 JAPAN email:naoki@archi.kyoto-u.ac.jp

[3] IBM Tokyo Research Laboratory, Yamato 242, JAPAN. email:ttoku@trl.ibm.co.jp

Given n real weights $w_1, w_2, \ldots, w_n$ that are stored in one-dimensional array, we shall consider the problem of finding an interval $I \in [1,n]$ that maximizes the interclass variance between weights in I and $[1,n]-I$. We shall present an $O(n \log n)$ time algorithm for this problem. We then extend this algorithm to two-dimensional case. Namely, given a $N \times N$ two-dimensional array in which each component is associated with real weight, the problem seeks to find a rectangular subarray R such that the interclass variance between weights in R and $A-R$ is maximized. We shall present an $O(N^3)$ algorithm for this problem.

1 はじめに

n 個の実数 $w_1, w_2, \ldots, w_n$ が一次元配列に与えられているものとしよう。このとき $[1,n]$ の部分区間 I に属する重み集合と $[1,n]-I$ に属する重み集合のクラス間分散を最大にする区間 I を求める問題を考察する。問題を定式化すると次のようになる。μ_i $(i=0,1)$ をそれぞれ集合 I および $[1,n]-I$ の重みの平均値とする。つまり，

$$\mu = \frac{1}{n}\sum_{i=1}^{n} w_i, \mu_0 = \frac{1}{n_0}\sum_{i \in I} w_i, \mu_1 = \frac{1}{n_1}\sum_{i \in [1,n]-I} w_i.$$

このとき，次の最大化問題を考察する。

$$P_1: \quad \text{maximize} \quad |I|(\mu-\mu_0)^2+(n-|I|)(\mu-\mu_1)^2$$
$$\text{subject to} \quad I \subset [1,n].$$

本論文は，この問題を解く $O(n\log n)$ 時間アルゴリズムを提案する。さらにこのアルゴリズムを二次元配列の問題に拡張する。つまり，実数の重みが $N\times N$ の二次元配列 A に与えられた場合に、クラス間分散を最大にする矩形領域 R を求める問題を扱う。いま，行列 A の (i,j) 成分の重みを w_{ij} とし，μ,μ_0,μ_1 を一次元の場合と同様に次のように定義する。

$$\mu=(\sum_{(i,j)\in A} w_{ij})/N^2, \mu_0=\frac{1}{|R|}\sum_{(i,j)\in R} w_{ij}, \mu_1=\frac{1}{N^2-|R|}\sum_{(i,j)\notin R} w_{ij}.$$

この問題は次のように定式化される。

$$P_2: \quad \text{maximize} \quad |R|(\mu-\mu_0)^2+(N^2-|R|)(\mu-\mu_1)^2$$
$$\text{subject to} \quad R: \text{行列 } A \text{ の矩形部分行列.}$$

本論文は，この問題を解く $O(N^3)$ 時間アルゴリズムを提案する。

筆者等 ([1, 2, 7]) は，画像処理における領域分割への応用を目的として、二次元領域におけるクラス間分散を最大にする x-単調領域を求める問題を考察し，$O(N^4)$ 時間アルゴリズムを提案した (ここで N は画像の一辺のサイズである)。そのアルゴリズムは，クラス間分散最大化問題を各重み w_{ij} をパラメータ付きの重み $w_{ij}-\theta$ に変換し，重み和最大の x-単調領域を求める問題に帰着させるものである。適当なパラメータを選ぶと，この重み和最大化問題の解がクラス間分散最大化問題の最適解になることが保証されている。[2] は，パラメータを固定したとき，重み和を最大にする x-単調領域を求める $O(N^2)$ 時間で解く高速アルゴリズムを開発した ([1] 参照)。また，パラメータを変化させたときに現れる異なる解の数の上限は $O(N^2)$ であることから，Eisner and Severence [4] のアルゴリズムにもとづいて $O(N^4)$ 時間アルゴリズムが構築されている。

上記のパラメトリック問題への帰着は，本論文で考察する一次元配列のクラス間分散を最大化する部分区間を求める問題 P_1 にも適用可能である。すると，パラメータを固定したとき，Bentley [3] が考察した、一次元配列内の重み和最大の部分区間を求める問題になる。[3] はその問題に対して線形時間アルゴリズムを提案している。またパラメータを変化させたときに現れる異なる解の数の上限は，異なる区間サイズの数であるので $O(n)$ である。したがって，問題 P_1 に対する $O(n^2)$ 時間アルゴリズムが得られる。

しかし，異なる区間幅ごとにクラス間分散を最大にする区間が自明な方法で $O(n)$ 時間で求まるので，繰り返しこれを用いることにより，$O(n^2)$ 時間で P_1 を解くことができる。したがって，上で述べたパラメトリック問題に変換する方法を用いた $O(n^2)$ 時間アルゴリズムは，この自明な方法と同じ計算量を有することになり，わざわざパラメトリック問題に変換する方法を用いる利点はない。

それでは $O(n^2)$ 時間を改良することは可能であろうか？本論文はこの問に対して肯定的な解答を与える。その解法はパラメトリック問題に変換するところまでは同じである。パラメトリック問題を用いる $O(n^2)$ 時間アルゴリズムは固定したパラメータに対して [3] による

$O(n)$ 時間の動的計画法によるアルゴリズムを用いるが，提案する方法は，同じく [3] によって提案された一次元配列内の重み和最大の部分区間を求める分割統治法による $O(n\log n)$ 時間アルゴリズムを用いる。$O(n^2)$ 時間アルゴリズムと違って，すべての異なるパラメータに対する解を同時に計算しながら分割統治アルゴリズムを進めていく。これにより，全体で $O(n\log n)$ 時間アルゴリズムが実現できる。

本論文ではさらにこのアルゴリズムを二次元配列の問題に拡張する。つまり，実数の重みが $N\times N$ の二次元配列に与えられた場合に、クラス間分散を最大にする矩形領域を求める問題を考察し，$O(N^3)$ 時間アルゴリズムを構築する。ちなみにこの問題に対する自明なアルゴリズムは一次元の場合と同様な方法で実現でき，その計算量は $O(N^4)$ である。本論文で考察する二次元配列のクラス間分散を最大にする矩形領域を求める問題は最近注目を集めているデータマイニングへの応用がある (データマイニングへの応用に関する関連の研究成果については [5, 6, 9, 10] を参照されたい)。また，関連の研究としては，玉木，徳山 [8] による研究があり，重み和最大の矩形領域を求める $o(N^3)$ 時間のアルゴリズムを提案している。

2 一次元問題のアルゴリズム

本節では一次元配列におけるクラス間分散を最大にする部分区間を求める $O(n\log n)$ 時間アルゴリズムについて述べる。まず，各重み w_i を $w_i-\mu$ と置き換える。このような変換に対してクラス間分散は不変であることに注意しておく。

補題 2.1 ([2]) 次のパラメトリック問題の最適解 $S^*_\theta(I)$ が P_1 の最適解であるようなパラメータ θ が存在する。

$$P_\theta([1,n]) : maximize_{I\subseteq[1,n]} S_\theta(I)=\sum_{i\in I}(w_i-\theta)\left(=\sum_{i\in I}w_i-\theta|I|\right)$$

補題 2.2 $P_\theta([1,n])$ において，θ を変化させた時に得られる異なる解の総数は $O(n)$ である。

証明. $|I|$ の取り得る値は 1 から $n-1$ であることから，補題が直ちに証明される。 □

θ を固定すると、問題 $P_\theta([1,n])$ は Bentley[3] によって，DP を用いると $O(n)$ で解ける。また，[3] は分割統治法による $O(n\log n)$ も提案している。したがって、補題 2.2 と [4] の結果より、$O(n^2)$ で P_1 が解ける。しかし、もっと単純な方法ですべての区間を組織的に調べることによって $O(n^2)$ 時間で解けることは自明である。

これから述べるアルゴリズムは、$O(n\log n)$ 時間で問題 P_1 を解くものであり，[3] による $O(n\log n)$ で一次元配列におけるクラス間分散最大の区間を求める分割統治アルゴリズムを用いる。このアルゴリズムはもともと本論文が扱う問題 P_1 において，固定された θ に対して $O(n\log n)$ で解くというものである。しかし、アルゴリズムをよく見るとすべての θ に対して $P_\theta([1,n])$ を解くという問題が計算量を変えずに $O(n\log n)$ で解ける。以下その方法を説明する。

いま、区間 $[1,n]$ を $[1,n/2],[n/2+1,n]$ に分けて $P_\theta([1,n/2])$, $P_\theta([n/2+1,n])$ の部分問題を各々解く (以下の記述では簡単のために n は 2 の冪乗であると仮定する)。その最適区間を各々 $I^*_{left}(\theta)$, $I^*_{right}(\theta)$ とする。また、$I^*_{left}(\theta)$, $I^*_{right}(\theta)$ の重みを $S^*_{left}(\theta)$, $S^*_{right}(\theta)$ とする。すると、これらは θ に関する区分線形な凸関数になる。

また、$[1,n]$ を二つに分ける分割ラインを跨ぐ区間のなかでの最適区間を $I^*_{overlap}(\theta)$ とする。さらに上と同様に $S^*_{overlap}(\theta)$ を定義しておく。

$P_\theta([1,n])$ の最適区間とその重み関数をそれぞれ $I^*(\theta)$, $S^*(\theta)$ とすると次の再帰方程式を得る。

$$S^*(\theta) = \max\{S^*_{left}(\theta), S^*_{right}(\theta), S^*_{overlap}(\theta)\} \tag{1}$$

したがって、$S^*_{overlap}(\theta)$ の計算と上の再帰方程式の最大値を求める計算が $O(n)$ でできると全体で $O(n\log n)$ で解けることになる。

1. $S^*_{overlap}(\theta)$ **の計算**: 区間 $[i,n/2]$ の区間和を $S_\theta([i,n])$ とする。

$$f_{left}(\theta) = \max_{1\le i\le n/2} S_\theta([i,n])$$

を求める。この計算は計算幾何学の双対変換を使うと x 座標に関してソート済みの点列に対する凸包計算に対応するので、$O(n)$ で計算可能である。

$$f_{right}(\theta) = \max_{n/2+1\le j\le n} S_\theta([n/2+1,j])$$

も同様にして $O(n)$ で計算しておく。すると,

$$S^*_{overlap}(\theta) = f_{left}(\theta) + f_{right}(\theta)$$

であり、$f_{left}(\theta), f_{right}(\theta)$ ともに凸関数であるので関数 $S^*_{overlap}(\theta)$ は $O(n)$ で計算できる。($f_{left}(\theta)$, $f_{right}(\theta)$ は各々関数の breakpoint の点列をリスト構造で管理すれば、$S^*_{overlap}(\theta)$ の計算はリストマージアルゴリズムの要領で実行できる。)

2. **再帰方程式の最大値計算**: 関数 $S^*_{left}(\theta)$, $S^*_{right}(\theta)$ の complexity(breakpoint の数) は高々 $n/2$ であることに注意すると (区間幅の取り得る値は 0 から $n/2$)、3 つの区分線形凸関数の最大値計算が $O(n)$ で出来ることをいえばよい。これもリストマージアルゴリズムの方法で $O(n)$ で計算できる。

定理 2.3 問題 P_1 は $O(n\log n)$ 時間で解ける。

例題: 次の一次元配列を考えよう。

1	3	-2	5	-4	-2

この配列に上で述べたアルゴリズムを適用してみよう。区間を $[1,3]$ と $[4,6]$ の二つに分割する。すると,

$$S^*_{left}(\theta) = \max\{3-\theta(=S_\theta([2,2])), 4-2\theta(=S_\theta([1,2])), 2-3\theta(=S_\theta([1,3]))\},$$

$$S^*_{right}(\theta) = \max\{5-\theta(=S_\theta([4,4])), 1-2\theta(=S_\theta([4,5])), -1-3\theta(=S_\theta([4,6]))\}$$

となる。

また,

$$f_{left}(\theta) = \max\{-2-\theta, 1-2\theta, 2-3\theta\},$$

$$f_{right}(\theta) = \max\{5-\theta, 1-2\theta, -1-3\theta\}$$

なので

$$S^*_{overlap}(\theta) = \max\{1-6\theta, 7-4\theta, 6-3\theta, 3-2\theta\}$$

となる。これらと (1) より,

$$S^*(\theta) = \max\{1-6\theta, 7-4\theta, 6-3\theta, 4-2\theta, 5-\theta\} \tag{2}$$

を得る。これより,

$$S^*(\theta) = \begin{cases} 1-6\theta & \theta \in (-\infty, -3] \text{ (区間 } [1,6]) \\ 7-4\theta & \theta \in [-3, 2/3] \text{ (区間 } [1,4]) \\ 5-\theta & \theta \in [2/3, +\infty) \text{ (区間 } [4,4]) \end{cases} \tag{3}$$

となる。以上から簡単な計算によりクラス間分散を最大にする区間は $[1,4]$ となる。

3 二次元への拡張

この節では, 前節のアルゴリズムを利用して問題 P_2 を $O(N^3)$ 時間で解くアルゴリズムを提案する。一節でも述べたようにこの問題に対する自明なアルゴリズムはすべての矩形部分行列とそのクラス間分散を組織的に求めていくもので $O(N^4)$ 時間かかることに注意しておく。

提案するアルゴリズムは二次元の分割統治法であり, その概略は以下の通りである。

ステップ 1: まず $N \times N$ 行列を $N/2 \times N$ の二つの横長の行列に分割する。そして, その分割線をまたぐ矩形部分行列 R のなかでクラス間分散最大のものを見つける。これは次のようにして求める。まず, R が第 j_1 列から始まり, 第 j_2 列で終るものだけを考えよう。すると, そのような行列のなかでクラス間分散最大のものをみつけるには, 第 i 要素の重みが

$$W_i = \sum_{j=j_1}^{j_2} w_{ij}$$

であるような一次元配列 W を考えると, この一次元配列 W のクラス間分散最大区間を求める問題となる。これは前節の結果から $O(N \log N)$ 時間で解ける。これを j_1 と j_2 のすべての組合せに対しておこなう。これに要する時間は $O(N^3 \log N)$ である。(図 1(1) を参照)

ステップ 2: ステップ 1で得られた二つの $N/2 \times N$ 行列の垂直二等分線を考える。各 $N/2 \times N$ 行列内の矩形部分行列でででその垂直二等分線をまたぐ部分行列を考える。そのような部分行列のなかでクラス間分散最大のものを求める。これは (1) で述べた方法と同様にして求められその計算時間は $O(N^3 \log N)$ である。(図 1(2) を参照)

ステップ 3: ステップ 1, 2で考えていない矩形部分行列は，$N \times N$ 行列を水平に二等分し，さらに垂直二等分して得られる4つの $N/2 \times N/2$ 行列の内部に含まれる。よって，この4つの行列に対してステップ 1, 2を再帰的に繰り返す。(図1(3)を参照)

以上の考察から上記のアルゴリズムの計算時間を $T(N)$ とすると，次の再帰方程式を得る。

$$T(N) \leq cN^3 \log N + 4T(N/2). \tag{4}$$

ここで c は正の定数である。この方程式を解くと $T(N) = O(N^3 \log N)$ を得る。しかし，これは以下の考察により $O(N^3)$ に改善できる。

上のステップ 1, 2では一次元配列のクラス間分散最大区間を求める問題を解くが，前節でも述べたようにそのアルゴリズムは分割統治法に基づいている。分割統治法では問題のサイズを毎回半分にしながら解くが，各レベルで要する計算時間は $O(N)$ である (レベルの数は $O(\log N)$)。したがって，上のステップ 1, 2で解く各問題も分割統治法のレベルでの計算だけをすればよいので, 再帰の各ステップにおける計算時間は $O(N^3 \log N)$ ではなく，$O(N^3)$ 時間である。よって再帰方程式は

$$T(N) \leq c'N^3 + 4T(N/2). \tag{5}$$

ここで c' は正の定数である。これを解くと $T(N) = O(N^3)$ となる。

定理 3.1 問題 P_2 は $O(N^3)$ 時間で解ける。

4 おわりに

本論文では, 一次元配列内のクラス間分散最大の部分区間を求める $O(n \log n)$ 時間アルゴリズムを提案した。さらにそれを二次元配列の問題に拡張し，クラス間分散を最大にする矩形領域を求める問題に対する $O(N^3)$ 時間アルゴリズムを提案した。本方法は三次元以上の場合にも容易に拡張できるものと思われる。最後に，ここで提案した手法は次の問題にも適用でき同じ計算量を有するアルゴリズムを得ることができることを指摘しておこう。

[平均値の差を最大にする2クラス分割を求める問題]

まず一次元の場合について述べる。これまでと同様に n 個の実数の重み $w_1, w_2, \ldots, w_n$ が一次元配列に与えられているものとする。このとき対象となる問題は次のように定式化される。

$$P_3: \quad \text{maximize} \quad \left| \frac{1}{|I|} \sum_{i \in I} w_i - \frac{1}{n - |I|} \sum_{i \notin I} w_i \right|$$
$$\text{subject to} \quad I \subset [1, n].$$

二次元の場合も P_2 と同様に定式化されるので詳しい記述は省略する。

参考文献

[1] 浅野哲夫, 離散システムとしての画像処理, 「離散構造とアルゴリズム　V」第3章(101-146頁), 藤重　悟　編著, 近代科学社, 1 998年6月.

[2] T. Asano, D. Chen, N. Katoh, and T. Tokuyama, "Polynomial Time Solution to Image Segmentation," *Proceedings of 7th ACM-SIAM Symposium on Discrete Algorithms,* (1996), pp.104-113.

[3] J. Bentley, Programming Pearls – Algorithm design techniques, CACM **27-9** (1984), 865-871.

[4] M.J. Eisner and D.G. Severence, Mathematical techniques for efficient record segmentation, JACM **23** (1976), 619-635.

[5] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama: "Data Mining Using Two-Dimensional Optimized Association Rules: Scheme, Algoirthms, and Visualization," *Proc. ACM SIGMOD International Conference on Management of Data* (1996) pp. 13-23.

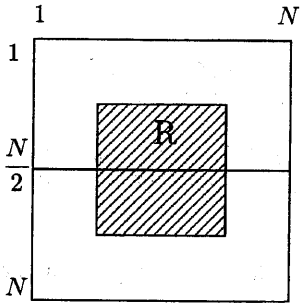
[6] T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama: "Mining Optimized Association Rules for Numeric Attributes," *Proc. 15th ACM Principle of Database Systems* (1996) pp. 182-191. A Journal version will appear in Journal of Computer Systems and Science.

[7] 加藤直樹、画像処理における組合せ最適化問題、オペレーションズ・リサーチ、Vol.40 No.7 (1994), 363-369.

[8] H. Tamaki and T. Tokuyama: "Algorithms for the Maximum Subarray Problem Based on Matrix Multiplication," *Proceedings of 9th ACM-SIAM Symposium on Discrete Algorithms,* (1998), pp.446-452.

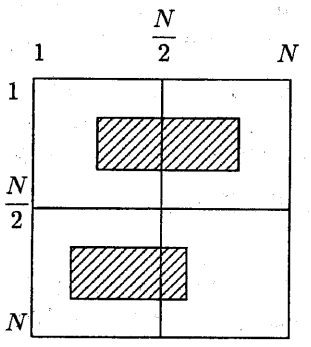
[9] 徳山 豪: "データマイニングに使われる最適化の数理," 応用数理 6 － 4 （1996） pp.49-59.

[10] K. Yoda, T. Fukuda, Y. Morimoto, S. Morishita, and T. Tokuyama: "Computing Optimized Rectilinear Regions for Association Rules," *Proc. 3rd Int. Conf. on Knowledge Discovery and Data Mining,* (1997) 96-103.



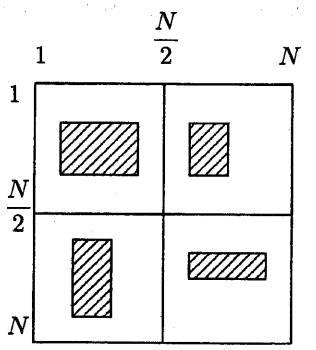
行列 A

(1)



行列 A

(2)



行列 A

(3)

図 1: ステップ 1, 2, 3 の説明図