最小２乗法の一般化
Generalizations of Least-Squares Algorithms

浅野　哲夫
Tetsuo Asano

北陸先端科学技術大学院大学情報科学研究科
School of Information Science,
Japan Advanced Institute of Science and Technology
Nomi, 923-1292, Japan
t-asano@jaist.ac.jp

概要

最小２乗法というと，実験データの直線当てはめが思い浮かぶが，本講演ではその一般化と，同様の手法の別の問題への応用について述べる．

２次元平面上の点列 $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ として指定される実験データを最も良く近似する直線 $y = ax+b$ は，この直線との垂直方向距離の差の２乗和 $\sum_{i=1}^{n}(ax_i+b-y_i)^2$ を最小にする定数 a, b によって定まる．そのような２つの定数を求めるには，２乗和の式を a と b を変数と見なして，それぞれで偏微分して 0 と置いて得られる連立方程式を解けばよい．しかし，基準を垂直方向の距離の差（絶対値）の和に変更すると，一般的な偏微分が取れなくなって問題が難しくなる．しかし，計算量的には同じ線形時間で最適な直線が求まることが既に知られている．

本講演では，別の意味での一般化を考える．すなわち，１本の直線で近似するのではなく，1 点から左右に延びる２本の半直線で近似する．近似の度合いは，点から半直線までの距離を用いて評価するが，このとき差の２乗和を用いても差の絶対値和を用いても同様に効率よく解けることを示す．

直線当てはめ以外の問題にも同じ手法を用いることができる．たとえば，平面上に多数の点が配置されているとして，それぞれの点への距離を指定して，なるべくその距離になるように新たな点を挿入するという問題を考える．実際に配置したときの距離と最初に与えられていた距離との差の２乗和を最小にするという問題を考えたとき，計算幾何学におけるアレンジメントの概念と併用すると，同じ手法で解けることを示す．

Keywords: アルゴリズム，計算幾何学，最小２乗法，最適化問題，直線当てはめ

1　点列の多角形近似[1]

与えられた x 単調な点列を１本の直線で近似する問題はよく研究されているが，本節では，これを一般化して１点から左右に延びる２本の半直線（長さ２の線分列と呼ぶ）に当てはめる問題について考える．ここでは最適な線分列として，データ点と線分列のずれの最大値または総和を最小にするものを選びたい．

[1] ここで述べる結果は参考文献 [2] で発表したものの一部である．

線分列までの最大垂直距離を最小にする線分列を求める問題はよく研究されている．たとえば，Hakimi と Schmeichel による $O(n^2 \log n)$ 時間のアルゴリズムなどがそうである [6]．計算時間に関しては，後に $O(n^2)$ [12] さらに $O(n \log n)$ [5] に改善されている．別のアプローチとして，点と線分列とのユークリッド距離の最大値を最小にする線分列を求めるものもある．この問題も多項式時間の解が知られている [11]．これらの問題は曲線の単純化問題にも関係している．これは点列の代わりに線分列が入力として与えられたときに，少ない線分数で近似する問題であり，地理情報処理などに関係している（詳しくはサーベイ論文 [13] を参照のこと）．もちろん，計算幾何学の分野でも精力的に研究されている [1, 7, 9, 10]．

最大値を最小化する問題はパターン認識の分野でよく研究されているが，１点だけ他と離れた点があると，それによって最適な線分列が大きく影響を受けることがある．このような影響を受けにくくする一つの方法は，最も離れた点までの距離を最小化するのではなく，各点から線分列までの距離の総和を最小にするように目的関数を変更することである．このとき，点と線分列との距離をどのように定義するかで問題の難しさも変わってくる．最も一般的な基準は，点から線分列までの L_2-距離の２乗和を最小にするというものである．L_1 距離の和を最小にするという基準でも最適解を求めることができる．直線による近似が線形時間でできることも知られている [8] が，アルゴリズム的には非常に複雑である．

本文では，距離としては L_1 または L_2 を用いて長さ２の線分列で点列を近似するという問題に焦点を当てる．ここでは，暗黙のうちに，線分列は x-単調であると仮定している．与えられた点列を１本の直線ではなく，線分列で近似するというのは非常に素直な拡張であるが，著者の知る限り過去に研究はないようである．実際，L_1 距離に関して言えば，Imai らの結果 [8] を２線分の場合に拡張することですら理論的には非常に難しいように思われる．

１個の接合点をもつ線分列で近似する場合，ここで与えるアルゴリズムの実行時間は，L_2 距離の場合は $O(n)$，L_1 時間の場合は $\tilde{O}(n^{4/3})$ である．ただし，$\tilde{O}$ の記号は定数 $c > 0$ に対して $\log^c n$ の係数を無視したものである．L_2 距離の場合の近似問題は比較的単純で実用的であるが，L_1 距離を用いた場合は複雑なデータ構造とパラメトリック探索の技法が必要になるので，実際の実装には困難が予想される．長さ２の線分列を $P = (e_1, v, e_2)$ と表す．ただし，e_1 と e_2 は半直線であり，それらは共通の端点 v をもつものとする．半直線 e_1 と e_2 を表す式をそれぞれ，$y = a_i x - b_i, i = 1, 2$ と表す．接点 v の座標を (v_x, v_y) とするとき，２つの半直線が端点 v を共有することから，

$$v_y = a_1 v_x + b_1 = a_2 v_x + b_2 \tag{1}$$

が成り立っている．

さて，点集合 $S = \{p_1 = (x_1, y_1), p_2 = (x_2, y_2), \ldots, p_n = (x_n, y_n)\}$，ただし，$x_1 < x_2 < \ldots < x_n$ 与えられるものとしよう．このとき，次の２つの値のうちの１つを最小化する長さ２の線分列 $P = (e_1, v, e_2)$ を求めたい．

$$\begin{aligned} L_1\colon &\quad \textstyle\sum_{s=1}^{k+1} \sum_{u_{s-1} < x_i \le u_s} |a_s x_i - b_s - y_i|, \\ L_2\colon &\quad \textstyle\sum_{s=1}^{k+1} \sum_{u_{s-1} < x_i \le u_s} (a_s x_i - b_s - y_i)^2. \end{aligned}$$

１本の直線だけで近似する場合がよく知られた最小２乗近似である．その場合，$\mathrm{A}_n = \sum_{i=1}^n x_i$, $\mathrm{B}_n = \sum_{i=1}^n y_i$, $\mathrm{C}_n = \sum_{i=1}^n x_i^2$, $\mathrm{D}_n = \sum_{i=1}^n x_i^2$, $\mathrm{E}_n = \sum_{i=1}^n x_i y_i$ という値をそれぞれ線形時間で求めておけば，最適な近似直線 $y = ax - b$ を求めることができる．

ここでは２つの段階に分けて問題を考えよう．まず，接合点は固定の区間 $[x_q, x_{q+1}]$ に存在するものと仮定する．この仮定は後で取り除く．$S_1(q) = \{p_1, p_2, \ldots, p_q\}$ と $S_2(q) = \{p_{q+1}, \ldots, p_n\}$ という記号を用意する．求めたいのは２本の直線からなる線分列である．それらを $\ell_1 \colon y = a_1 x - b_1$ および $\ell_2 \colon y = a_2 x - b_2$ としよう．ここでは，２本の直線 $y = a_1 x - b_1$ と $y = a_2 x - b_2$ が x 方向の区間 $[x_q, x_{q+1}]$ で交差するとき，４つ組 (a_1, b_1, a_2, b_2) は実行可能であるということにする．ここでの目標は，次の値を最小にする実行可能な４つ組を求めることである：

$$\sum_{i=1}^{q} |a_1 x_i - b_1 - y_i| + \sum_{i=q+1}^{n} |a_2 x_i - b_2 - y_i| \quad \text{かつ} \tag{2}$$

$$\sum_{i=1}^{q} (a_1 x_i - b_1 - y_i)^2 + \sum_{i=q+1}^{n} (a_2 x_i - b_2 - y_i)^2. \tag{3}$$

式 (2) を最小化することは，$a_1 \neq a_2$ であれば，次の制約条件の下に $\sum_{i=1}^{n} w_i$ の値を最小化することと等価である．

$$\begin{aligned} &-w_i \leq a_1 x_i - b_1 - y_i \leq w_i, \quad i \leq q \text{ のとき}, \\ &-w_i \leq a_2 x_i - b_2 - y_i \leq w_i, \quad i \geq q+1 \text{ のとき}, \\ &x_q \leq \tfrac{b_1 - b_2}{a_1 - a_2} \leq x_{q+1}, \end{aligned} \tag{4}$$

ただし，最後の式は実行可能性の条件である．

補題 1.1 q を固定したとき，L_1-または L_2-距離を用いた最適線分列当てはめ問題は，２つの凸計画問題を解くことに帰着できる．

証明 制約条件を無視すると，問題は明らかに２次計画問題（L_2 距離の場合）と線形計画問題（L_1 距離の場合）となる．２本の直線が指定された区間において交差するという実行可能性の条件は，別の線形制約となる．したがって，問題を２つに分解することができる．$a_1 \leq a_2$ の場合，２線分が指定区間で交差するための必要十分条件は，x_q において ℓ_1 が ℓ_2 より下にはなく，x_{q+1} においては上には来ないことである．したがって，追加の制約式は次のようになる．

$$x_q(a_2 - a_1) \leq b_2 - b_1 \leq x_{q+1}(a_2 - a_1)\,. \tag{5}$$

逆の場合には次のようになる．

$$x_{q+1}(a_2 - a_1) \leq b_2 - b_1 \leq x_q(a_2 - a_1)\,. \tag{6}$$

明らかに，各部分問題は凸計画問題である．

上の補題より，最適な長さ２の線分列が凸計画問題として線形時間で求まることは明らかである．しかしここでは別のアルゴリズムを設計したい．そこで，問題を２つのタイプに分類する．(a) 不等式 (5), (6) が等式で成り立つ．(b) (5),(6) の不等式がすべて等号なしで成り立つ．前者を固定問題と呼び，後者を自由問題と呼ぶことにする．

補題 1.2 固定問題の場合，接合点は２本の垂直線 $x = x_q$, $x = x_{q+1}$ のどちらか一方の上にある．

接合部が直線 $x = x_{q+1}$ 上にある場合，それを点集合 S の $S_1(q+1) = S_1(q) \cup \{p_{q+1}\}$ と $S_2(q+1) = S_2(q) \setminus \{p_{q+1}\}$ への分割に対する解とみなす．したがって，それぞれの分割に対して，(1) 自由問題と (2) 接合点が垂直線 $x = x_q$ にある場合の固定問題の 2 つを解けばよい．これから次の汎用アルゴリズムが得られる：すなわち，集合 S の各分割 (S_1, S_2) に対して，実行可能性に関する制約を無視して自由問題を解き，得られた解が実行可能かどうか，すなわち 2 直線の交点が 2 点 p_q と p_{q+1} の間にあるかどうかを確かめる．もし実行可能なら，それが最適解である．そうでない場合は，接合点を直線 $x = x_q$ 上にあるとした固定問題を解き，その解を最適解として出力する．これをすべての分割について行い，目的関数の最小値を与える解を全体の最適解として出力する．

上記の汎用アルゴリズムは，近似線分列までの距離の 2 乗和を最小にする場合でも，垂直距離の和を最小にする場合でも，単に目的関数を変更するだけで適用可能である．しかし，距離の 2 乗和の場合には，さらに次のような工夫をすることによって計算時間を線形時間に削減することができる．

アルゴリズムでは，$q = 1$ の場合から始めて $q = n - 1$ の場合まで順に処理して行く．このとき，和，分散，共分散の値を常に管理するものとする．具体的には，$\mathrm{A}_q = \sum_{i=1}^{q} x_i$, $\mathrm{B}_q = \sum_{i=1}^{q} y_i$, $\mathrm{C}_q = \sum_{i=1}^{q} x_i^2$, $\mathrm{D}_q = \sum_{i=1}^{q} y_i^2$, $\mathrm{E}_q = \sum_{i=1}^{q} x_i y_i$ の値を毎回更新していく．この更新が全体で線形時間でできることは明らかである．また，$S_2(q)$ についても同様の値を管理しておく．

自由問題の場合には，目的関数は分離可能である．すなわち，$\sum_{i=1}^{q}(a_1 x_i - b_1 - y_i)^2$ を最小にする (a_1, b_1) と $\sum_{j=q+1}^{n}(a_2 x_j - b_2 - y_j)^2$ を最小にする (a_2, b_2) を独立に求めることによって解を得ることができる．$\mathrm{A}_q, \ldots, \mathrm{E}_q$ の値が利用可能であるから，この計算は定数時間でできる．実行可能性の判定も定数時間でできる．残っているのは，接合点を $x = x_q$ 上にもつという制約を加えた部分問題である．$f(\cdot)$ を最小化すべき目的関数とし，接合点に関する制約を $g(\cdot) = 0$ で表すものとする．

$$f(a_1, b_1, a_2, b_2) = \sum_{i=1}^{q}(a_1 x_i - b_1 - y_i)^2 + \sum_{j=q+1}^{n}(a_2 x_j - b_2 - y_j)^2, \tag{7}$$

$$g(a_1, b_1, a_2, b_2) = a_1 x_q - b_1 - a_2 x_q + b_2, \tag{8}$$

$$L(a_1, b_1, a_2, b_2) = f(a_1, b_1, a_2, b_2) - \lambda g(a_1, b_1, a_2, b_2), \tag{9}$$

このとき，Kuhn-Tucker 条件により，距離の 2 乗和を最小にする線分列を与える最適解 $Z_{\mathrm{opt}} = (a_1^0, b_1^0, a_2^0, b_2^0)$ は次式を満たす．

$$\left.\frac{\partial L}{\partial a_1}\right|_{Z_{\mathrm{opt}}} = \left.\frac{\partial L}{\partial b_1}\right|_{Z_{\mathrm{opt}}} = \left.\frac{\partial L}{\partial a_2}\right|_{Z_{\mathrm{opt}}} = \left.\frac{\partial L}{\partial b_2}\right|_{Z_{\mathrm{opt}}} = 0, \tag{10}$$

および

$$g(Z_{\mathrm{opt}}) = 0. \tag{11}$$

これで a_1, b_1, a_2, b_2 の最適な値とラグランジェ乗数 λ が満たさなければならない 5 個の線形方程式の集合が得られる．係数は $x_q, \mathrm{A}_q, \ldots, \mathrm{E}_q$ を用いて表現できるので，この連立方程式は定数時間で解くことができる．よって，次の定理を得る．

定理 1.3 線分列までの距離の 2 乗和を最小にする長さ 2 の線分列は線形時間で求めることができる．

2　指定した距離を実現する点の挿入

同じような技法を用いて全く別の問題が解けることを示そう．ここでは，予め平面上に配置された点集合 $S = \{p_1, \ldots, p_n\}$ が与えられているものとして，ここに新たな点を挿入する問題を考える．新たな点から既存の点までの距離が $d_1, \ldots, d_n$ で指定されるとき，これらの距離をなるだけ正確に実現するように点を挿入したい．正確には，挿入すべき点を p とし，S の点 p_i との距離を $d(p, p_i)$ とすれば，最小化すべき目的関数は

$$f(p) = \sum_{i=1}^{n} |d(p, p_i)^2 - d_i^2| \tag{12}$$

である．上記の式で 2 乗をせずに，単に距離の差の和を用いて目的関数を定義することもできるが，その場合には d_i の値をすべて 0 とした特殊な問題がフェルマーの問題と呼ばれる計算困難な問題となってしまうことに注意しておかなければならない．点 p の座標を (x, y)，各点 p_i の座標を (x_i, y_i) とするとき，式 (13) は具体的に次のように書くことができる．

$$f(x, y) = \sum_{i=1}^{n} |(x - x_i)^2 + (y - y_i)^2 - d_i^2|. \tag{13}$$

絶対値がなければ，前節と同様に x と y に関する偏微分によって方程式を求めることができる．言い換えれば，絶対値を外すことができれば，最小 2 乗法を用いることができる．

ここでは計算幾何学で標準的な技法を用いる．与えられたそれぞれの点 p_i を中心とする半径 d_i の円 C_i を描く．これによって平面は $O(n^2)$ 個の領域に分かれることになる．それぞれの領域（以下，セルと呼ぶ）R は，それを内部に含む円の集合と，それ以外の円の集合によって特徴づけることができる．R を含む円の中心点の集合を $\mathcal{S}(R)$ とするとき，この集合の点 p_i については，点 p までの距離が対応する半径より小さく，それ以外の点については対応する半径より大きい．したがって，セル R では上記の目的関数を次のように分解して計算することができる．

$$f_R(x, y) = \sum_{p_i \in S(R)} (d_i^2 - (x - x_i)^2 + (y - y_i)^2) + \sum_{p_i \notin S(R)} ((x - x_i)^2 + (y - y_i)^2 - d_i^2). \tag{14}$$

これで絶対値が外せたので，この目的関数を x と y に関して偏微分して線形の連立方程式を求めれば，セル R における最適解が求まる．したがって，すべてのセルについて解を求めれば，全体の最適解が得られる．

上記のアルゴリズムを単純に実現すると，円のアレンジメントを蓄えるのに $O(n^2)$ の記憶スペースが必要になるだけではなく，それぞれのセルでも $O(n)$ の計算時間が必要となるから，全体の計算時間も $O(n^3)$ となる．記憶スペースを線形 $O(n)$ に改善するには，計算幾何学における常套手段である平面走査法を用いればよい．これは走査線と呼ばれる垂直線を左から右に動かして行きながら，走査線と交差するセルでの計算を行うというものである．上に述べた目的関数の偏微分によって得られる連立方程式は，走査線が新たな状態（円の最も左の点，円の最も右の点，円と円の交点）に達するたびに定数時間で更新できる．走査線での状態の管理に毎回 $O(\log n)$ の時間がかかることを考慮すると，全体の計算時間は $O(n^2 \log n)$ となる．

計算時間を $O(n^2)$ に改善できるかどうかは未解決問題である．円ではなくて直線であれば，トポロジカルスィープ [4] またはトポロジカルウォーク [3] と呼ばれる技法を適用することで解決できるが，円のアレンジメントでも同じことができるかどうかは知られていない．

参考文献

[1] P. K. Agarwal and K. R. Varadarajan, "Efficient algorithms for approximating polygonal chains," *Discrete Comput. Geom.*, **23** (2000) 273–291.

[2] B. Aronov, T. Asano, N. Katoh, K. Mehlhorn, and T. Tokuyama: Polyline fitting of planar points under min-sum criterion, to appear in *International Journal on Computational Geometry and Applications*.

[3] T. Asano, L. J. Guibas and T. Tokuyama: Walking in an Arrangement Topologically, *International Journal on Computational Geometry and Applications*, Vol. 4, No. 1, pp. 123-151, 1994.

[4] H. Edelsbrunner and L.J. Guibas: "Topologically Sweeping an Arrangement," *J. Comp. and Sys. Sci.*, 38 pp.165-194 (1989).

[5] M. Goodrich, "Efficient piecewise-linear function approximation using the uniform metric," *Discrete Comput. Geom.*, **14** (1995) 445–462.

[6] S. Hakimi and E. Schmeichel, "Fitting polygonal functions to a set of points in the plane," *Graphical Models and Image Processing*, **53** (1991) 132–136.

[7] H. Imai and M. Iri: "Polygonal approximations of a curve - Formulations and algorithms," *Computational Morphology*, Elsevier Science Publishers B.V. (North Holland), 1988, 71–86.

[8] H. Imai, K. Kato, and P. Yamamoto: "A linear-time algorithm for linear L_1 approximation of points," *Algorithmica*, **4** (1989) 77–96.

[9] Y. Kurozumi and W.A. Davis: "Polygonal approximation by the minimax method," *Computer Graphics and Image Processing*, **19** (1982) 248–264.

[10] A. Melkman and J. O'Rourke, "On polygonal chain approximation," *Computational Morphology*, Elsevier Science Publishers B.V. (North Holland), 1988, 87–95.

[11] J. O'Rourke and G. Toussaint, "Pattern recognition," Chapter 43 of *Handbook of Discrete and Computational Geometry* (eds. J. Goodman and J. O'Rourke), CRC Press, 1997.

[12] D. P. Wang, N. F. Huang, H. S. Chao, and R. C. T. Lee, "Plane sweep algorithms for polygonal approximation problems with applications," *Proc. 4th Internat. Symp. Algorithms Comput. (ISAAC 2003)*, LNCS 762, 1993, pp. 515–522.

[13] Robert Weibel, "Generalization of spatial data: principles and selected algorithms," *Algorithmic Foundations of Geographic Information Systems*, LNCS 1340, 1997, pp. 99–152.