数理解析研究所講究録
第1841巻 2013年 105-113

分数凸計画問題に対するDC最適化手法に基づく逐次近似解法

(A successive approximation method for solving a fractional programming problem based on DC optimization)

新潟大学大学院自然科学研究科　平野 裕之
HIRANO Yasuyuki
Graduate School of Science and Technology, Niigata University
新潟大学大学院自然科学研究科　山田 修司
YAMADA Syuuji
Graduate School of Science and Technology, Niigata University
新潟大学大学院自然科学研究科　田中 環
TANAKA Tamaki
Graduate School of Science and Technology, Niigata University
大阪大学大学院工学研究科　谷野 哲三
TANINO Tetsuzo
Graduate School of Engineering, Osaka University

1　はじめに

本研究では, 2つの凸関数の比で表わされた目的関数をコンパクト凸集合上で最小化する分数凸計画問題 (FP) を考える。(FP) は2つの凸関数の差で表わされる関数 (dc 関数) を目的関数としたDC計画問題に変換できることが知られている。また, DC計画問題の解法に対しては Tuy[3, 5] によって提案された外部近似法が強力な逐次近似解法の1つであることが知られている。したがって本研究では (FP) の近似解を効率よく計算するために Tuy[5] と Dinkelbach[1] によってそれぞれ提案された外部近似法とパラメトリック最適化手法を組み合わせた新たな解法を提案する。

2　分数凸計画問題

本研究では次の最適化問題について考えるものとする。

$$(\mathrm{FP}) \quad \begin{cases} \text{minimize } \theta(\boldsymbol{x}) := \dfrac{f(\boldsymbol{x})}{g(\boldsymbol{x})} \\ \text{subject to } h(\boldsymbol{x}) \leq 0 \end{cases}$$

ただし, $f(\boldsymbol{x}) := \max_{p=1,\ldots,m_f} f_p(\boldsymbol{x})$, $g(\boldsymbol{x}) := \max_{q=1,\ldots,m_g} g_q(\boldsymbol{x})$, $h(\boldsymbol{x}) := \max_{r=1,\ldots,m_h} h_r(\boldsymbol{x})$ とし, f_p, g_q, $h_r : \mathbb{R}^n \to \mathbb{R}$ $(p = 1, \ldots, m_f, q = 1, \ldots, m_g, r = 1, \ldots, m_h)$ は連続微分可能な凸関数とする。ただし, $\mathbb{R}$ と $\mathbb{R}^n$ は実数全体の集合と n 次元ユークリッド空間を意味する。ここで, f_p, g_q, h_r の凸性より, $f, g, h : \mathbb{R}^n \to \mathbb{R}$ は $\mathbb{R}^n$ 上の凸関数である。また, $X := \{\boldsymbol{x} \in \mathbb{R}^n : h(\boldsymbol{x}) \leq 0\}$ とおくと X は (FP) の実行可能集合であり, 閉凸集合である。本研究では, (FP) に対して, 以下を仮定する。

(A1) $\operatorname{int} X = \{\boldsymbol{x} \in \mathbb{R}^n : h(\boldsymbol{x}) < 0\} \neq \emptyset$

(A2) $\exists \rho \in \mathbb{R}$ s.t. $\rho \geq \max\{\|\boldsymbol{x} - \boldsymbol{y}\| : \boldsymbol{x}, \boldsymbol{y} \in X\}$

(A3) $\inf\{f(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^n\} > 0$ かつ $\inf\{g(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^n\} > 0$

ただし, $\operatorname{int} X$ は X の内部を表わし, $\|\boldsymbol{x}\|$ はベクトル $\boldsymbol{x} \in \mathbb{R}^n$ を与えたときの $\boldsymbol{x}$ のユークリッドノルムを意味する。仮定 (A1) と (A2) より, X は空でないコンパクト集合である。さらに, 仮定 (A3) より, 目的関数 θ は $\mathbb{R}^n$ 上で連続である。したがって, (FP) は大域的最適解をもつ。

ここで, 任意の $\omega > 0$ に対して, 以下のパラメトリック最適化問題を考える。

$$(\mathrm{P}(\omega)) \quad \begin{cases} \text{minimize } \psi(\boldsymbol{x};\omega) := f(\boldsymbol{x}) - \omega g(\boldsymbol{x}) \\ \text{subject to } \boldsymbol{x} \in X \end{cases}$$

関数 f と g の連続性から, 任意の ω に対して (P(ω)) は大域的最適解をもつ。また, $\omega > 0$ のとき f と g の凸性から (P(ω)) の目的関数 $\psi(\boldsymbol{x};\omega)$ は dc 関数である。

このとき, 以下の定理が成り立つ。

定理 2.1 (Jagannthan [2])
実行可能解 $\bar{\boldsymbol{x}} \in X$ が (FP) の大域的最適解であるための必要十分条件は $\bar{\boldsymbol{x}}$ が (P($\theta(\bar{\boldsymbol{x}})$)) を解くことである。

問題 (FP) と (P($\theta(\bar{\boldsymbol{x}})$)) の最適値をそれぞれ min(FP) と min (P($\theta(\bar{\boldsymbol{x}})$)) とおき, $\bar{\boldsymbol{x}}$ を (FP) の大域的最適解とおく。このとき, 仮定 (A3) と定理 2.1 より

$$\min(\mathrm{FP}) = \theta(\bar{\boldsymbol{x}}) = \frac{f(\bar{\boldsymbol{x}})}{g(\bar{\boldsymbol{x}})} > 0,$$
$$\min(\mathrm{P}(\theta(\bar{\boldsymbol{x}}))) = f(\bar{\boldsymbol{x}}) - \theta(\bar{\boldsymbol{x}})g(\bar{\boldsymbol{x}}) = 0$$

が成り立つ。

3 逐次近似解法

問題 (FP) の近似解を求めるために, Tuy[5] と Dinkelebach[1] がそれぞれ提案した外部近似法とパラメトリック最適化手法を組み合わせた以下の逐次近似解法を提案する。

アルゴリズム SAM

ステップ 0.

ステップ 0-1. 実行可能解 $\boldsymbol{y^1} \in X$ を求め, $\omega_1 := \dfrac{f(\boldsymbol{y_1})}{g(\boldsymbol{y_1})}$ とおく。ステップ 0-2 へ。

ステップ 0-2. $S \supset X$ を満たす凸多面体 $\mathrm{S} \subset \mathbb{R}^n$ を生成する。S の頂点集合 $V(S)$ を計算する。$f(\boldsymbol{x'}) := \max\{f(\boldsymbol{x}) : \boldsymbol{x} \in V(S)\}$ を満たす $\boldsymbol{x'} \in V(S)$ を選ぶ。$\tilde{t} > f(\boldsymbol{x'})$ を満たす $\tilde{t} \in \mathbb{R}$ を定め, $D := \{(\boldsymbol{x}, t) \in \mathbb{R}^n \times \mathbb{R} : \boldsymbol{x} \in X, f(\boldsymbol{x}) \le t \le \tilde{t}\}$ とおく。ステップ 0-3 へ。

ステップ 0-3. $P_1 \supset D$ と $P_1 \subset \{(\boldsymbol{x}, t) \in \mathbb{R}^n \times \mathbb{R} : t \le \tilde{t}\}$ を満たす凸多面体 $P_1 \subset \mathbb{R}^{n+1}$ を生成する。ステップ 0-4 へ。

ステップ 0-4. $(\bar{\boldsymbol{y}}, \bar{t}) \in \operatorname{int} D$ を求める。許容誤差 $\tau \ge 0$ を定め, $k = 1$ とおく。ステップ 1 へ。

ステップ 1. $(\boldsymbol{x^k}, t^k) \in \operatorname{argmin}\{t - \omega_k g(\boldsymbol{x}) : (\boldsymbol{x}, t) \in V(P_k)\}$ を選ぶ。ステップ 2 へ。

ステップ 2. 次の停止条件 (SC) を満たすならばアルゴリズムを停止する。このとき, $\boldsymbol{y^k}$ と ω_k はそれぞれ (FP) の近似解と近似値とする。

$$(\mathrm{SC}) \quad t_k - \omega_k g(\boldsymbol{x^k}) \ge -\tau$$

その他の場合はステップ 3 へ。

ステップ 3. 以下のように $\boldsymbol{y^{k+1}}, \omega_{k+1}, P_{k+1}$ を更新する。

$$\boldsymbol{y^{k+1}} := \begin{cases} \boldsymbol{x^k} & f(\boldsymbol{x^k}) - \omega_k g(\boldsymbol{x^k}) < 0 \text{ かつ} \boldsymbol{x^k} \in X \text{ のとき} \\ \boldsymbol{y^k} & \text{その他の場合} \end{cases}$$

$$\omega_{k+1} := \begin{cases} \dfrac{f(\boldsymbol{x^k})}{g(\boldsymbol{x^k})} & f(\boldsymbol{x^k}) - \omega_k g(\boldsymbol{x^k}) < 0 \text{ かつ} \boldsymbol{x^k} \in X \text{ のとき} \\ \omega_k & \text{その他の場合} \end{cases}$$

$$P_{k+1} := P_k \cap \{(\boldsymbol{x}, t) \in \mathbb{R}^n \times \mathbb{R} : l_k(\boldsymbol{x}, t) < 0\}$$

ただし,

$$l_k(\boldsymbol{x}, t) := \left\langle \begin{pmatrix} \boldsymbol{d^k} \\ \xi_k \end{pmatrix}, \begin{pmatrix} \boldsymbol{x} \\ t \end{pmatrix} - \begin{pmatrix} \boldsymbol{z^k} \\ \eta_k \end{pmatrix} \right\rangle = \langle \boldsymbol{d^k}, \boldsymbol{x} - \boldsymbol{z^k} \rangle + \xi_k (t - \eta_k)$$

$$(\boldsymbol{d^k}, \xi_k) \in \partial\phi(\boldsymbol{z^k}, \eta_k) \ (\boldsymbol{d^k} \in \mathbb{R}^n, \xi_k \in \mathbb{R})$$

$$(\boldsymbol{z^k}, \eta_k) \in [(\boldsymbol{x^k}, t_k), (\bar{\boldsymbol{y}}, \bar{t})] \cap (\mathrm{bd}\, D)$$

$$\phi(\boldsymbol{x}, t) := \max\{h(\boldsymbol{x}), f(\boldsymbol{x}) - t\}$$

ここで, $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ はベクトル $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ を与えたときの $\boldsymbol{a}$ と $\boldsymbol{b}$ の内積, $\partial\phi(\boldsymbol{z^k}, \eta_k)$ は点 $(\boldsymbol{z^k}, \eta_k)$ における劣微分, $[(\boldsymbol{x^k}, t_k), (\bar{\boldsymbol{y}}, \bar{t})]$ は点 $(\boldsymbol{x^k}, t_k)$ と点 $(\bar{\boldsymbol{y}}, \bar{t})$ を結ぶ線分, $\mathrm{bd}\, D$ は D の境界集合を意味する。頂点集合 $V(P_{k+1})$ を計算する。 $k \leftarrow k+1$ としてステップ 1 へ戻る。

アルゴリズム SAM によって生成される凸多面体列 $\{P_k\}$ は次を満たす。

$$(\boldsymbol{x^k}, \boldsymbol{t_k}) \notin P_{k+1}$$

よって, 以下が成り立つ

$$P_1 \supsetneq P_2 \supsetneq \cdots \supsetneq P_k \supsetneq \cdots \supsetneq D$$

また, アルゴリズム SAM の反復 k において, $f(\boldsymbol{x^k}) - \omega_k g(\boldsymbol{x^k}) < 0$ かつ $\boldsymbol{x^k} \in X$ が成立するならば, 次が成立する。

$$0 \leq \theta(\boldsymbol{x^k}) = \frac{f(\boldsymbol{x^k})}{g(\boldsymbol{x^k})} < \omega_k = \frac{f(\boldsymbol{y^k})}{g(\boldsymbol{y^k})} = \theta(\boldsymbol{y^k})$$

このとき, 定理 2.1 より $\boldsymbol{y^k}$ は (FP) の大域的最適解でないことがわかる。したがって, 次が成り立つ。

$$\omega^k \geq \omega^{k+1} \geq \min(\mathrm{FP})$$

アルゴリズム SAM において以下の定理が成り立つ。

定理 3.1
許容誤差 $\tau = 0$ とおく。アルゴリズム SAM の反復 k において, 停止条件 (SC) を満たしたとする。このとき $\boldsymbol{y^k}$ は (FP) の大域的最適解である。

定理 3.2
許容誤差 $\tau = 0$ かつ $\{(\boldsymbol{x^k}, \boldsymbol{t_k})\}$ をアルゴリズム SAM によって生成された無限列とする。このとき, $\{(\boldsymbol{x^k}, \boldsymbol{t_k})\}$ のすべての集積点は D に含まれる。

定理 3.3
許容誤差 $\tau = 0$ かつ $\{(\boldsymbol{x^k}, \boldsymbol{t_k})\}$ をアルゴリズム SAM によって生成された無限列とする。このとき, $\{\boldsymbol{x^k}\}$ のすべての集積点は (FP) の大域的最適解である。さらに, $\{\boldsymbol{y^k}\}$ のすべての集積点は (FP) を解く。

アルゴリズム SAM によって無限列 $\{\boldsymbol{x^k}\}$ と $\{\boldsymbol{y^k}\}$ が生成された場合, 定理 3.3 より $\{\boldsymbol{x^k}\}$ と $\{\boldsymbol{y^k}\}$ のすべての集積点は (FP) の大域的最適解となる。さらに, $\tau > 0$ とおくことで, 定理 3.2 から, アルゴリズム SAM は有限回の反復で終了することがわかる。

4 頂点集合の計算に対する処理

3章で提案したアルゴリズムSAMのステップ3において頂点集合 $V(P_{k+1})$ が計算される。従来, この $V(P_{k+1})$ は連立1次方程式を解くことにより計算される。しかし, この方法は大規模な問題に対して効率が悪いことが知られている。そこで, 凸多面体の辺による頂点間の連結情報を用いた $V(P_{k+1})$ の生成方法を提案する。

まず, 以下の定義を紹介する。

定義 4.1 ([4])
集合 $P \subset \mathbb{R}^n$ を $\dim P = n$ を満たす凸多面体とし, H を P の支持超平面とする。ただし, $\dim P$ は P の次元を表わす。このとき共通部分 $F := H \cap P$ は P の面と呼ばれる。もし, $\dim F = 1$ であるならば F は P の辺と呼ばれ, $\dim F = n-1$ であるならば F は P のファセットと呼ばれる。

凸多面体 P_k をアルゴリズムSAMの反復 $k-1$ において生成されたものとする。このとき, $(v(i), t(i))$ $(i \in \Delta_k)$, F_j $(j \in \Gamma_k)$, $\mathcal{V}_i$ $(i \in \Delta_k)$, $\mathcal{F}_i$ $(i \in \Gamma_k)$, $\alpha(k), \beta(k)$ を以下のように定義する。

- $(v(i), t(i))$ $(i \in \Delta_k)$ は P_k の頂点を表わす。ただし, Δ_k は P_k のすべての頂点の添え字集合とする。
- F_j $(j \in \Gamma_k)$ は P_k のファセットを表わす。ただし, Γ_k は P_k のすべてのファセットの添え字集合とする。
- $\mathcal{V}_i := \{j : [(v(i), t(i)), (v(j), t(j))]$ は P_k の辺, $j \in \Delta_k \setminus \{i\}\}$.
  ただし, $[(v(i), t(i)), (v(j), t(j))]$ は $(v(i), t(i))$ と $(v(j), t(j))$ を結ぶ線分を表わす。
- $\mathcal{F}_i := \{j : (v(j), t(j)) \in F_j, j \in \Gamma_k\}$.
- $\alpha(k) := \max\{i : i \in \Delta_k\}$.
- $\beta(k) := \max\{i : i \in \Gamma_k\}$.

4.1 初期の凸多面体と添え字集合の生成

関数 $h_r (r = 1, \ldots, m_h)$ が連続微分可能な凸関数なので, 仮定 (A1) より $h(\bar{y}) < 0$ を満たす $\bar{y} \in \mathbb{R}^n$ が存在する (すなわち, $\bar{y} \in \mathrm{int} X$) 。仮定 (A2) より, $X \subset B(\bar{y}, 2\rho)$ が成り立つ。したがって, 以下のような $S \supset X$ を満たす n 次元単体 S を得られる。

$$S := \mathrm{co}\{\tilde{v}(1), \ldots, \tilde{v}(n+1)\} \text{ (すなわち,} V(S) = \{\tilde{v}(1), \ldots, \tilde{v}(n+1)\})$$

ただし,

$$\tilde{v}(1) := (\bar{y}_1 + 2\rho, \ldots, \bar{y}_n + 2\rho)^{\mathsf{T}}$$

$$\tilde{v}(i) := (\tilde{v}(1)_1, \ldots, \tilde{v}(2)_{i-2}, \bar{y}_{i-1} - 2\rho(n - 1 + \sqrt{n}), \tilde{v}(1)_i, \ldots, \tilde{v}(1)_n)^{\mathsf{T}} \quad (\forall i = 2, \ldots, n+1)$$

ここで, $\mathrm{co}\{\tilde{v}(1), \ldots, \tilde{v}(n+1)\}$ は集合 $\{\tilde{v}(1), \ldots, \tilde{v}(n+1)\}$ の凸包を表わし, $(\bar{y}_1 + 2\rho, \ldots, \bar{y}_n + 2\rho)^{\mathsf{T}}$ は $(\bar{y}_1 + 2\rho, \ldots, \bar{y}_n + 2\rho)$ の転置ベクトルを表わす。S がコンパクト凸集合であるから $\check{t} := \min\{f(x); x \in S\}$ が存在する。ここで以下のような P_1 を考える。

$$P_1 := \{(x, t) \in \mathbb{R}^n \times \mathbb{R} : x \in S, \check{t} \leq t \leq \tilde{t}\}$$

このとき,

$$P_1 \supset D \text{ かつ } V(P_1) = \{(v(1), t(1)), \ldots, (v(2n+2), t(2n+2))\}$$

ただし, 任意の $i \in \{1,\ldots,2n+2\}$ に対して,

$$(v(i),ti) := \begin{cases} (\tilde{v}(i),\check{t}) & (1 \le i \le n+1) \\ (\tilde{v}(i-n-1),\tilde{t}) & (n+2 \le i \le 2n+2) \end{cases}$$

したがって,

$$\Delta_1 := \{1,\ldots,2n+2\}$$

とおく。このとき, P_1 は $n+3$ つのファセットを持つ。したがって, Γ_k+1 と P_1 のすべてのファセット $F_1,\ldots,F_{n+3}$ を以下のようにおく。

$$\Gamma_{k+1} := \{1,\ldots,n+3\}$$

$$F_i := \begin{cases} \mathrm{co}\{(v(j),t(j)) : j \in \Delta_1 \backslash \{i, i+n+1\}\} & (1 \le i \le n+1) \\ \mathrm{co}\{(v(1),t(1)),\ldots,(v(n+1),t(n+1))\} & (n = n+2) \\ \mathrm{co}\{(v(n+2),t(n+2)),\ldots,(v(2n+2),t(2n+2))\} & (n = n+3) \end{cases}$$

このとき, 各 $i \in \Delta_1$ に対して $\mathcal{V}_i$ と $\mathcal{F}_i$ は以下のようになる。

$$\mathcal{V}_i := \begin{cases} (\{1,\ldots,n+1\}\backslash\{i\}) \cup \{i+n+1\} & (1 \le i \le n+1) \\ (\{n+2,\ldots,2n+2\}\backslash\{i\}) \cup \{i-n-1\} & (n+2 \le i \le 2n+2) \end{cases}$$

$$\mathcal{F}_i := \begin{cases} (\{1,\ldots,n+1\}\backslash\{i\}) \cup \{n+2\} & (1 \le i \le n+1) \\ (\{1,\ldots,n+1\}\backslash\{i-n-1\}) \cup \{n+3\} & (n+2 \le i \le 2n+2) \end{cases}$$

さらに,

$$\alpha(1) := 2n+2$$

$$\beta(1) := n+3$$

である。

4.2 頂点集合の生成

アルゴリズムSAMのステップ3より, $P_{k+1} = P_k \cap \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : l_k(x,t) \le 0\}$ かつ $l_k(v(i_k),t(i_k)) > 0$ なので, $V(P_k) \setminus P_{k+1} \neq \emptyset$ となる。したがって, 頂点集合 $V(P_{k+1})$ を生成するために, $l_k(v(i),t(i)) > 0$ を満足するすべての $i \in \Delta_k$ を調べる必要がある。このとき, 以下の命題が成り立つ。

命題 4.1
集合 $P \subset \mathbb{R}^n$ を凸多面体とし, $l : \mathbb{R}^n \to \mathbb{R}$ をアフィン関数とする。頂点 $v', v'' \in V(P)$ が $l(v') > 0$ と $l(v'') > 0$ を満たしたとする。このとき, $l(\hat{v}) > 0$ を満たす $\hat{v} \in V(P) \setminus \{v', v''\}$ が存在し、また線分 $[v', v'']$ は P の辺となる。

命題 4.2
集合 $P \subset \mathbb{R}^n$ を凸多面体とし, ある $\hat{v} \in V(P)$ に対して $P' := \mathrm{co}(V(P) \setminus \{\hat{v}\})$ とする。頂点 $v', v'' \in V(P)$ $(v' \neq v'')$ を $[v', v'']$ が P' の辺であることを満たしているとする。このとき, 次の条件の1つまたは両方が成り立つ。

1. 線分 $[v', v'']$ が P の辺である
2. 線分 $[\hat{v}, v']$ と $[\hat{v}, v'']$ がともに P の辺である

命題 4.3
頂点 $P \subset \mathbb{R}^n$ を凸多面体とし, $l : \mathbb{R}^n \to \mathbb{R}$ をアフィン関数とする。頂点 $v', v'' \in V(P)$ $(v' \neq v'')$ が $l(v') > 0$ と $l(v'') > 0$ を満たしたとする。このとき, $\hat{v}(1) = v', \hat{v}(s) = v'', l(\hat{v}(i)) > 0$ $(\forall i \in \{1,\ldots,s\})$ を満たす $\hat{v}(1),\ldots,\hat{v}(s) \in V(P)$ が存在し, また線分 $[\hat{v}(i-1), \hat{v}(i)]$ $(i \in \{2,\ldots,s\})$ は P の辺となる。

命題 4.3. に基づいてアルゴリズム SAM の反復 k において, $l_k(v(i),t(i)) > 0$ $(\forall i \in \mathcal{M}_{k+1})$ かつ $l_k(v(i),t(i)) \leq 0$ $(\forall i \in \Delta_k \setminus \mathcal{M}_{k+1})$ を満たす頂点のリスト $\mathcal{M}_{k+1} \subset \Delta_k$ を生成できる。ここで, $i_1 \in \mathcal{M}_{k+1}$ と $i_2 \in \mathcal{M}_{k+1}$ が $l_k(v(i_1),t(i_2)) < 0$ を満たしたとする。このとき, $[(v(i_1),t(i_1)),(v(i_2),t(i_2))]$ は P_k の辺となり, また $(v',t') \in [(v(i_1),t(i_1)),(v(i_2),t(i_2))]$ を満たす頂点 $(v',t') \in V(P_k+1) \setminus P_k$ が存在する。さらに, 以下が成り立つ。

- $(v',t') \in F_j$ $(\forall j \in \mathcal{F}_{i_1} \cap \mathcal{F}_{i_2})$
- $(v',t') \in F_{\beta(k+1)}$
  ただし, $\beta(k+1) := \beta(k)+1$ かつ $F_{\beta(k+1)} := P_{k+1} \cap \{(x,t) \in \mathbb{R}^n \times \mathbb{R} : l_k(x,t) = 0\}$

アルゴリズム SAM の反復 k において生成されるすべての頂点を計算するために, 以下の処理を提案する。

処理 A

ステップ 0. $\mathcal{M}_{k+1} := \mathcal{M}'_{k+1} := \{i_k\}, \mathcal{W}_{k+1} := \emptyset$, $\alpha(k) := \alpha(k)$, $\beta(k+1) := \beta(k)+1$ とおく。ステップ 1 へ。

ステップ 1. $\mathcal{M}'_{k+1} := \emptyset$ ならば $\Delta_{k+1} := (\Delta_k \cup \{\alpha(k)+1,\ldots,\alpha(k+1)\}) \setminus \mathcal{M}_{k+1}$ とし, ストップ。その他の場合は $j \in \mathcal{M}'_{k+1}$ を選びステップ 2 へ。

ステップ 2.

ステップ 2-0. $\mathcal{T} := \mathcal{V}_j$ とし, ステップ 2-1 へ。

ステップ 2-1. $\mathcal{T} = \emptyset$ ならばステップ 3 へ。その他の場合は $\kappa \in \mathcal{T}$ を選びステップ 2-2 へ。

ステップ 2-2. $l_k(v(\kappa),t(\kappa)) > 0$ ならば, ステップ 2-3 へ。 $l_k(v(\kappa),t(\kappa)) < 0$ ならば, ステップ 2-4 へ。その他の場合はステップ 2-5 へ。

ステップ 2-3. $\kappa \notin \mathcal{M}'_{k+1}$ ならば, $\mathcal{M}_{k+1} \leftarrow \mathcal{M}_{k+1} \cup \{\kappa\}$, $\mathcal{M}'_{k+1} \leftarrow \mathcal{M}'_{k+1} \cup \{\kappa\}$, $\mathcal{V}_\kappa \leftarrow \mathcal{V}_\kappa \setminus \{j\}$ とする。ステップ 2-6 へ。

ステップ 2-4. $(v(\alpha(k+1)+1),t(\alpha(k+1)+1))$, $\mathcal{F}_{\alpha(k+1)+1}$, $\mathcal{V}_{\alpha(k+1)+1}$ を以下のように定める。

$$(v(\alpha(k+1)+1),t(\alpha(k+1)+1)) := (1-\lambda)(v(j),t(j)) + \lambda(v(\kappa),t(\kappa))$$

$$\mathcal{F}_{\alpha(k+1)+1} := (\mathcal{F}_j \cap \mathcal{F}_\kappa) \cup \{\beta(k+1)\}$$

$$\mathcal{V}_{\alpha(k+1)+1} := \{\kappa\}$$

$$\mathcal{V}_\kappa \leftarrow (\mathcal{V}_\kappa \setminus \{j\}) \cup \{\alpha(k+1)+1\}$$

$$\mathcal{W}_{k+1} \leftarrow \mathcal{W}_{k+1} \cup \{\alpha(k+1)+1\}$$

$$\alpha(k+1) \leftarrow \alpha(k+1)+1$$

ただし,

$$\lambda := \frac{l_k(v(\kappa),t(\kappa))}{l_k(v(j),t(j)) - l_k(v(\kappa),t(\kappa))}$$

このステップにおいて, $\mathcal{V}_{\alpha(k+1)}$ の更新は不完全である。4 章 4 節において提案する処理 C により添え字集合 $\mathcal{V}_{\alpha(k+1)+1}$ を完成させる。ステップ 2-6 へ。

ステップ 2-5. $\mathcal{V}_\kappa$, $\mathcal{F}_\kappa$, $\mathcal{W}_{k+1}$ を以下のように更新する。

$$\mathcal{V}_\kappa \leftarrow \mathcal{V}_\kappa \setminus \{j\}$$

$$\mathcal{F}_\kappa \leftarrow \begin{cases} \mathcal{F}_\kappa \cup \{\beta(k+1)\} & (\beta(k+1) \notin \mathcal{F}_\kappa \text{ のとき}) \\ \mathcal{F}_\kappa & (\text{その他}) \end{cases}$$

$$\mathcal{W}_{k+1} \leftarrow \begin{cases} \mathcal{W}_{k+1} \cup \{\kappa\} & (\kappa \notin \mathcal{F}_\kappa \text{ のとき}) \\ \mathcal{W}_{k+1} & (\text{その他}) \end{cases}$$

ステップ 2-6 へ。

ステップ 2-6. $\mathcal{T} \leftarrow \mathcal{T} \setminus \{\kappa\}$ として, ステップ 2-1 に戻る。

ステップ 3. $\mathcal{M}'_{k+1} \leftarrow \mathcal{M}'_{k+1} \setminus \{j\}$ として, ステップ 1 に戻る。

このとき, 次が成り立つ。

- 処理 A は各 $i \in \mathcal{M}_{k+1}$ に対して

$$(\boldsymbol{v}(i), t(i)) \in V(P_k) \text{ かつ } (\boldsymbol{v}(i), t(i)) \notin V(P_{k+1})$$

であるような添え字集合 $\mathcal{M}_{k+1}$ を生成する。

- 処理 A は $(\boldsymbol{v}(\alpha(k)+1), t(\alpha(k)+1)), \ldots, (\boldsymbol{v}(\alpha(k+1)), t(\alpha(k+1)))$, Δ_{k+1}, α_{k+1}, β_{k+1} を計算する。したがって, $V(P_{k+1}) = \{(\boldsymbol{v}(i), t(i)) : i \in \Delta_{k+1}\}$ がいえる。

- 処理 A は各 $i \in \mathcal{W}_{k+1}$ に対して $l_k(\boldsymbol{v}(i), t(i)) = 0$ であるような添え字集合 $\mathcal{W}_{k+1} \subset \Delta_{k+1}$ を生成する。したがって, $\mathcal{W}_{k+1} = \{\alpha(k)+1, \ldots, \alpha(k+1)\} \cup \{i \in \Delta_k : l_k(\boldsymbol{v}(i), t(i)) = 0\}$ となる。

- 4 章 4 節で提案する処理 C は $\mathcal{V}_i$ $(i \in \mathcal{W}_{k+1})$ を生成または, 更新する。

- 各 $i \in \Delta_{k+1} \setminus \mathcal{W}_{k+1}$ に対して, $\mathcal{V}_i$ は処理 A によって完全に更新される。

4.3 ファセットの添え字集合の更新

4 章 2 節において, $V(P_{k+1})$ の生成に対する処理 A を提案した。$(\boldsymbol{v}(\alpha(k)+1), t(\alpha(k)+1)), \ldots, (\boldsymbol{v}(\alpha(k+1)), t(\alpha(k+1)))$ を計算するために処理 A は $\mathcal{F}_i$ $(i \in \Delta_k)$ を利用する。したがって, この節では Γ_{k+1} の計算方法と各 $i \in \Delta_k$ に対する $\mathcal{F}_i$ の更新に対する処理を提案する。このとき, 以下の命題は明らかである。

命題 4.4
集合 P を n 次元凸多面体集合とし, $F(1), \ldots, F(\beta)$ を P のファセットとする。このとき, 各 $i \in \{1, \ldots, \beta\}$ と $j \in \{1, \ldots, \beta\} \setminus \{i\}$ に対して, $F_i \not\subset F_j$ である。

命題 4.4 より, 各 $i \in \Gamma_k \cup \{\beta(k+1)\}$ に対して, ある $j \in (\Gamma_k \cup \{\beta(k+1)) \setminus \{i\}$ が $F_i \cap V(P_{k+1}) \subset F_j$ を満たすならば, $i \notin \Gamma_{k+1}$ である。したがって, 以下のように任意の $i \in \Delta_{k+1}$ に対する Γ_{k+1} と $\mathcal{F}_i$ を生成する処理を提案する。

処理 B

ステップ 0. $\Gamma_{k+1} := \Gamma_k \cup \{\beta(k+1)\}$ とし, $i := 1$ とする。ステップ 1 へ。

ステップ 1. $i = \beta(k+1)$ ならばストップ。$i \notin \Gamma_{k+1}$ ならばステップ 4 へ。その他の場合はステップ 2 へ。

ステップ 2. $\Psi_i := \{\kappa \in \Delta_{k+1} : i \in \mathcal{F}_\kappa\}$ とする。$\Psi_i = \emptyset$ または $|\Psi_i| < n$ ならば $\Gamma_{k+1} \leftarrow \Gamma_{k+1} \setminus \{i\}$ とし, ステップ 4 へ。ただし, $|\Psi_i|$ は Ψ_i の元の数を表わす。その他の場合はステップ 3 へ。

ステップ 3.

ステップ 3-0. $j := i+1$ としステップ 3-1 へ。

ステップ 3-1. $j := \beta(k+1)+1$ ならばステップ 4 へ。$j \notin \Gamma_{k+1}$ ならばステップ 3-4 へ。その他の場合はステップ 3-2 へ。

ステップ 3-2. $\Psi_j := \{\kappa \in \Delta_{k+1} : j \in \mathcal{F}_\kappa\}$ とする。$\Psi_j = \emptyset$, $|\Psi_j| < n$ または $\Psi_j \subset \Psi_i$ であるならば, $\Gamma_{k+1} \leftarrow \Gamma_{k+1} \setminus \{j\}$, $\mathcal{F}_\kappa \leftarrow \mathcal{F}_\kappa \setminus \{j\}$ とし, ステップ 3-4 へ。その他の場合はステップ 3-3 へ。

ステップ 3-3. $\Psi_i \subset \Psi_j$ ならば, $\Gamma_{k+1} \leftarrow \Gamma_{k+1} \setminus \{i\}$, $\mathcal{F}_k \leftarrow \mathcal{F}_\kappa \setminus \{i\}$ とし, ステップ 4 へ。その他の場合はステップ 3-4 へ。

ステップ 3-4. $j \leftarrow j+1$ とし, ステップ 3-1 に戻る。

ステップ 4. $i \leftarrow i+1$ とし, ステップ 1 に戻る。

4.4 頂点間の連結情報の更新

処理 A は $\mathcal{V}_i$ $(i \in \mathcal{M}_{k+1})$ を利用することで $V(P_{k+1})$ を生成した。したがって, アルゴリズム SAM の反復 k の後, 頂点集合 $V(P_{k'})$ $(k' > k)$ を生成するために, $\mathcal{V}_{\alpha(k)+1}, \ldots, \mathcal{V}_{\alpha(k+1)}$ を生成することと, $\mathcal{V}_i$ $(i \in \{i \in \Gamma_k : l_k(v(i), t(i)) = 0\})$ を更新することが必要である。このとき, 以下の命題が成り立つ。

命題 4.5
集合 $P \subset \mathbb{R}^n$ を $\dim P = n$ を満たす凸多面体とし, $\{F_i : i \in \Gamma\}$ を P のすべてのファセットの集合とする。ただし, Γ は P のファセットの添え字集合とする。線分 $[v', v'']$ $(v', v'' \in V(P), v' \neq v'')$ が P の辺であることの必要十分条件は各 $j = 1, \ldots, n-1$ に対して $[v', v''] \subset F_{i_j}$ であるような $i_1, \ldots, i_{n-1} \in \Gamma$ が存在することである。

処理 B は P_{k+1} のファセットの添え字集合 Γ_{k+1} を生成する。したがって, 以下に各 $i \in \mathcal{W}_{k+1}$ ($\mathcal{W}_{k+1}$ は処理 A によって生成された集合) に対して, $\mathcal{V}_i$ を生成するための処理を提案する。

処理 C

ステップ 0. $j = 1$ として, ステップ 1 へ。

ステップ 1. $j = \alpha(k+1)$ ならばストップ。$j \notin \mathcal{W}_{k+1}$ ならばステップ 3 へ。その他の場合はステップ 2 へ。

ステップ 2.

ステップ 2-0. $\kappa := j+1$ とし, ステップ 2-1 へ。

ステップ 2-1. $\kappa = \alpha(k+1)+1$ ならばステップ 3 へ。$\kappa \notin \mathcal{W}_{k+1}$ ならばステップ 2-3 へ。その他の場合はステップ 2-2 へ

ステップ 2-2. $|\mathcal{F}_j \cap \mathcal{F}_\kappa| = n-1$ ならば $\mathcal{V}_j \leftarrow \mathcal{V}_j \cup \{\kappa\}$ と $\mathcal{V}_\kappa \leftarrow \mathcal{V}_\kappa \cup \{j\}$ とする。ステップ 2-3 へ。

ステップ 2-3. $\kappa \leftarrow \kappa+1$ とし, ステップ 2-1 に戻る。

ステップ 3. $j \leftarrow j+1$ として, ステップ 1 に戻る。

処理 A と C より, すべての添え字集合 $\mathcal{V}_i$ $(i \in \Delta_{k+1})$ が完全に更新される。

5 おわりに

本研究では 2 つの凸関数の比で表わされた目的関数をコンパクト凸集合上で最小化する分数凸計画問題に対して, 近似解を効率的に計算するために, 外部近似法とパラメトリック最適化手法を組み合わせた逐次近似解法を提案した。このアルゴリズム SAM は大域的収束を保証するために凸多面体列を生成する。生成された凸多面体列は外部から X 上の f のエピグラフの境界を近似する。さらにアルゴリズム SAM の計算効率を向上させるために連立 1 次方程式を解かずに頂点集合を更新する, 凸多面体の辺による頂点間の連結情報を利用した手法を提案をした。

参考文献

[1] Dinkelbach, W. *Complementary Geometric Programming SIAM J. Appl. Math.* 13, 492–498 (1967).

[2] Jagannathan, R.: *On some properties of programming problem in parametric form pertainig to fractional programing, Management Sci.* 12, 609–615 (1966).

[3] Tuy, H.: *Canonical DC Programming: Outer Approximation Methods Revisited, Oper. Res. Let..* 18, 99–106 (1995).

[4] Tuy, H.: *Convex Analysis and Global Optimization, Kluwer Academic Publishers* (1998).

[5] Tuy, H.: *On global optimality conditions and cutting plane algorithms, J. Optim. Theory Appl.* 118, 201–216 (2003).