非凸計画問題≠解けない問題

—分枝限定法による大域的最適化—

久野　誉人

1.　はじめに

図書館にある本棚ほどの大きさだったコンピュータは名前の通りのノートサイズに縮んでしまい，値段の方も大きさに比例するように安くなった．性能はそれらに反比例して高速になったお陰で，今では数万変数の線形計画問題もそのノートの上でするりと解けてしまう．さらにはインターネットを通じて最適化コードが無料で手に入るので，難しそうな非線形計画問題でもソファに横たわったまま労なくして解決できる．一昔前までは夢想もできなかった理想的な研究環境が実現したようにも思えるが，その恩恵を受けられるのは凸関数を凸集合上で最小化する凸計画問題を対象とする場合だけ，と一般には考えられているようだ．

目的関数も制約式もすべて凸性を満たしていれば，逆さに差した傘の上の雨粒でさえ知っているように大域的な最小点を見つけだすことは（たぶん）難しくない．ところが，例えば目的関数が凹関数になると，正しく差した傘の上の雨粒が勝手なところから滴ってくるのと同じで，局所的な最小点がいくつも実行可能領域の中に存在する．傘の例は2次元の問題だから雨粒の滴ってくるところを一つ一つ列挙すれば大域的最小点もすぐに見つかるが，問題の次元が数十ともなればもう単純な列挙法では追い付かないことは明らかだ．過去の教科書達も，傘の例こそ出さないが本質的には同じ理由で「大規模非凸計画問題＝解決困難な問題」と教えている．それがいつの間にやら「非凸計画問題＝解けない問題」と実務家の間で解釈されているようで，非凸計画問題として定式化されたモデルには最近なかなかお目にかかれない．

コンピュータがまだ本棚サイズで気軽に触れられない時代には確かに「非凸計画問題＝解けない問題」というパラダイムは正しかったかもしれない．しかし，今でもそれが正しいのだろうか？ 本稿でこれを完全に否定することはできないものの，少なくともある種の構造をもつ非凸計画問題に限っては，結構大規模な問題まで現実的な時間のうちに解くことができることを紹介しよう．

くの　たかひと　筑波大学電子情報工学系
〒305-8573 つくば市天王台 1-1-1

2.　古きを温（たづ）ねて

非凸計画法の基本となる**凹最小化問題**(concave minimization problem)に対し，最初の大域的最適化アルゴリズムが発表されたのは1964年のことである[9]．この凸集合上における凹関数の最小化問題を解くために，Tuyは[9]の中で2つのアルゴリズムを提案している；一つは実行可能領域の大域的最小点を含まない部分を削り落とす**凹性カット法**(concavity-cut method)；もう一つは，整数計画・組合せ最適化でお馴染みの**分枝限定法**(branch-and-bound method)である．その後，凸計画法にも使われる**外部近似法**(outer approximation method)が加わるが，60年代に開発されたこの3つの手法が今もって非凸計画法の主役を続けている（詳しくは[2]を参照のこと）．さて，この節で紹介したいのはTuy[9]の5年後に発表されたFalkとSolandの分枝限定法[1]とSolandによるその廉価版[8]である．

2.1.　Falk-Solandの分枝限定法

Falk-Solandのアルゴリズム[1]が扱える凹最小化問題には，下に示すような少し特別な構造が必要である：

$$\left|\begin{array}{ll} \text{最小化} & f(\boldsymbol{x}) \equiv \sum_{j=1}^{n} f_j(x_j) \\ \text{条　件} & \boldsymbol{x} \in D. \end{array}\right. \tag{1}$$

ここで，$D \subset \mathbb{R}^n$はコンパクトな凸集合，$f : \mathbb{R}^n \to \mathbb{R}$は凹関数であるが，$\boldsymbol{x}$の成分$x_j$達それぞれの凹関数$f_j$の和として表されなければならない．このとき$f$は**分離可能**(separable)であるというが，(1)のように分離可能な凹最小化問題は後述するようにネットワーク流問題などの中にたくさんある．

アルゴリズムは，まずD上における各変数x_jの下界値l_jと上界値u_jを計算し，Dを含む矩形Mを定める：

$$D \subset M \equiv \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{l} \leq \boldsymbol{x} \leq \boldsymbol{u}\}.$$

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

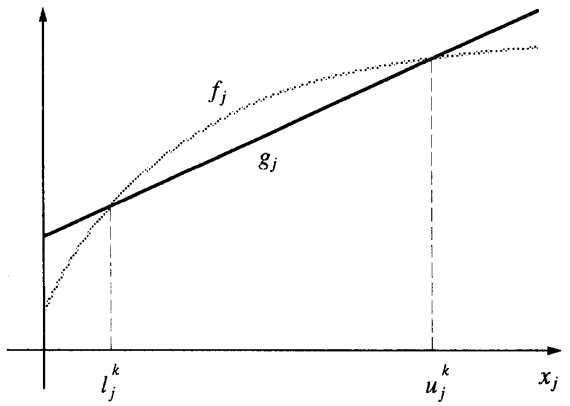

図1　凹関数 f_j と下包関数 g_j

この M を，通常の分枝限定法の手続きにしたがって，いくつかの矩形達

$$M^k = \{\boldsymbol{x} \in \mathbb{R}^n \mid \boldsymbol{l}^k \leq \boldsymbol{x} \leq \boldsymbol{u}^k\},\ k = 1, \ldots, r,$$

に分割して行く．つまり，以下の3つのステップを繰り返して最適解を含む領域を絞り込むのである:

ステップ1. 適当な矩形 M^k を選ぶ;

ステップ2. $D \cap M^k$ 上での f の下界値 $\bar{f}$ が暫定値 f° 以上ならば M^k を捨て去る（**限定操作**）;

ステップ3. $\bar{f} < f^\circ$ ならば，M^k を2つの矩形 M^{r+1}, M^{r+2} に分割する（**分枝操作**）.

効率の鍵を握るのはもちろんステップ2における目的関数 f の下界値 $\bar{f}$ であるが，その計算には f の分離可能性が効力を発揮する．まず，各 j に対して $(l_j^k, f_j(l_j^k))$ と $(u_j^k, f_j(u_j^k))$ を結ぶアフィン関数 g_j を定義する:

$$g_j(x_j) = \frac{f(u_j^k) - f(l_j^k)}{u_j^k - l_j^k} x_j + \frac{u_j^k f(l_j^k) - l_j^k f(u_j^k)}{u_j^k - l_j^k}. \quad (2)$$

すると図1からもわかるように，区間 $[l_j^k, u_j^k]$ では g_j の値が f_j のそれよりも大きくなることはないので，

$$g(\boldsymbol{x}) \equiv \textstyle\sum_{j=1}^n g_j(\boldsymbol{x}) \leq f(\boldsymbol{x}),\ \forall \boldsymbol{x} \in M^k, \quad (3)$$

が成り立つ．このように定義された g を f の M^k 上における**下包関数** (lower envelop function) と呼ぶ．式(3)から直ちに，凸最小化問題:

$$\left|\begin{array}{ll} \text{最小化} & g(\boldsymbol{x}) \\ \text{条　件} & \boldsymbol{x} \in D \cap M^k \end{array}\right. \quad (4)$$

を解くことで f の下界値 $\bar{f}$ の得られることがわかる．同時に(4)の最適解 $\bar{\boldsymbol{x}}$ は(1)の実行可能解でもあるから，$f(\bar{\boldsymbol{x}}) < f^\circ$ ならば暫定値を $f^\circ = f(\bar{\boldsymbol{x}})$ と更新できる．

次に効率を左右するのはステップ2における M^k の分割法である．これには様々な方法が提案されており，中にはアルゴリズムの収束が保証されないものもある [2]. 最も簡単で確実なのは，

$$s \in \arg\max\{f_j(\bar{x}_j) - g_j(\bar{x}_j) \mid j = 1, \ldots, n\}$$

に対し，区間 $[l_s^k, u_s^k]$ の真ん中で矩形 M^k を2分割して M^{r+1}, M^{r+2} を生成する方法である．

ここまで説明すれば，整数計画法を少しでも勉強したことのある人なら誰でも，このアルゴリズムが線形整数計画問題に用いられている普通の分枝限定法とそっくりなことに気がつくはずだ．つまり，下界値 $\bar{f}$ を求めるために解いた凸最小化問題(4)は，(1)の子問題:

$$\left|\begin{array}{ll} \text{最小化} & f(\boldsymbol{x}) \\ \text{条　件} & \boldsymbol{x} \in D \cap M^k \end{array}\right. \quad (5)$$

の緩和問題であり，整数計画法における変数の連続緩和が，ここでは凹関数 f_j の下包 g_j をとる操作に置き代わっただけなのである．ただし，整数計画法とは違って一般に有限回での収束は保証されない．そのため，ステップ2で行なう M^k を捨て去るかどうかの判定は，$\bar{f} \geq f^\circ$ ではなく，あらかじめ定めておいた許容誤差 $\epsilon > 0$ に対して

$$\bar{f} + \epsilon \geq f^\circ$$

で行なうことが多い．その結果，アルゴリズムの出力 $\boldsymbol{x}^\circ$ は厳密な最適解ではなく，ϵ-**最適解** (ϵ-optimal solution) と呼ばれる．しかし，ここで注意したいのは，$\boldsymbol{x}^\circ$ が発見的な近似解法によって得られる局所的な近似解ではなく，大域的に見ても真の最適値との誤差を ϵ しかもたない点である．その意味で，この Falk-Soland の分枝限定法は大域的最適化に分類される．

2.2. Soland の廉価版

分離可能な凹最小化問題(1)を解くには，それと同じような制約条件をもつ凸最小化問題(4)をいくつか解けばよいことがわかった．「同じような」と書いたのは，(4)には(1)にない新たな制約 $\boldsymbol{x} \in M^k$ が加わっているからである．ところが，この制約のせいで(4)がずいぶん解きにくくなってしまうことがある．

例えば，D がネットワーク流によって定義される凸多

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず.

面体で，目的関数は

$$f(\boldsymbol{x}) = \sum_{i=1}^{m} f_i\left(\sum_{j\in J_i} c_j x_j\right)$$

によって与えられているとしよう．ただし，$\bigcup_{i=1}^{m} J_i = \{1,\ldots,n\}$ である．新たに m 個の変数

$$y_i = \sum_{j\in J_i} c_j x_j,\quad i = 1,\ldots,m,$$

を導入すれば，問題は $\boldsymbol{y}$ に関して分離可能な凹最小化問題となり，Falk-Soland のアルゴリズムの守備範囲に入る．しかし，その中で何度も解かれる凸最小化問題 (3) には

$$l_i^k \le \sum_{j\in J_i} c_j x_j \le u_i^k,\quad i = 1,\ldots,m, \qquad (6)$$

なる**付帯制約** (side consraint) が加わるため，せっかくのネットワーク構造は破壊されて利用できない．

そこで Soland はこの厄介な制約 (6) を無視してしまった．つまり，(4) の代わりに

$$\left|\begin{array}{ll} \text{最小化} & g(\boldsymbol{x}) \\ \text{条　件} & \boldsymbol{x}\in D \end{array}\right. \qquad (7)$$

を子問題 (5) の緩和問題として提案したのである [8]．問題 (7) の実行可能領域は (4) の部分集合なので，(7) の最適値を $\tilde{f}$ とすれば，$\bar{f}$ および (5) の最適値 f^* との間に

$$\tilde{f} \le \bar{f} \le f^* \qquad (8)$$

の関係が成り立つ．したがって，下界値の精度はいくらか悪くなるものの，分枝限定法の各ステップで $\tilde{f}$ は $\bar{f}$ の代役を立派に務めることができるのである．また，(7) は元の問題構造をそのまま引き継いおり，制約条件が常に同じである．そのため，その最適解 $\tilde{\boldsymbol{x}}$ は次の反復における緩和問題の初期実行可能解として利用することもできる．さらに D が凸多面体であることを仮定すれば，以下に示すような棚ぼたもある．

それにはステップ 3 での分割方法を少し改め，

$$s \in \arg\max\{f_j(\tilde{x}_j) - g_j(\tilde{x}_j) \mid j = 1,\ldots,n\}$$

に対する区間 $[l_s^k, u_s^k]$ の，真ん中ではなく，$\tilde{x}_s$ を境にして M^k を 2 分割する．つまり，新たに生成される矩形 M^{r+1} と M^{r+2} の s に対する辺はそれぞれ

$$[l_s^{r+1}, u_s^{r+1}] = [l_s^k, \tilde{x}_s],\ [l_s^{r+2}, u_s^{r+2}] = [\tilde{x}_s, u_s^k]$$

となる．問題 (7) には $\boldsymbol{x}\in M^k$ を要求する制約がないので，$\tilde{x}_s$ が区間 $[l_s^k, u_s^k]$ の内部にあるとは限らない．しかし，その場合，任意の j に対して $f_j(\tilde{x}_j) \le g_j(\tilde{x}_j)$ が成り立つので，$f^\circ \le f(\tilde{\boldsymbol{x}}) \le \tilde{f} = \sum_j g_j(\tilde{x}_j)$ となって M^k はステップ 2 で捨てられているはずである．

このような方法で矩形 M の分割を進めていくと，D が凸多面体ならば $\tilde{\boldsymbol{x}}$ はその端点となるので，M^k 達の各辺 $[l_j^k, u_j^k]$ の両端は D のいずれかの端点と対応することになる．ところが D の端点の数は有限個しかない．したがって，生成される M^k の数も有限個となり，分枝限定法は誤差 ϵ を考慮しなくとも有限回での収束が保証される．このときの出力 $\boldsymbol{x}^\circ$ は 0- 最適解，つまり厳密な大域的最適解である．

3. 新しきを知る

30 年も昔の大域的最適化アルゴリズムを 2 つ紹介したが，果してその実力はといえば，[1, 8] を紐解くかぎり，おもちゃサイズの問題がいくつか解かれているだけで魅力に乏しい．しかし，その実験は 30 年も昔のコンピュータを使って行なわれているのである．2 つのアルゴリズムを現代のノート上でプログラムして実験してみれば，その意外な効率のよさに驚かされるに違いない．

さて，この 2 つの分枝限定法，一体どちらが優れているだろうか？ 両者の用いる下界値 $\bar{f}$，$\tilde{f}$ に (8) の関係があることから，生成される分枝木は Falk-Soland の方が小さいことを予想できる．しかし，Falk の方は元の問題構造を存分に活用できるうえ，D が凸多面体ならば，各緩和問題を解くのに強力な**パラメトリック費用単体法** (parametric cost simplex method) を使うこともできる．そこで，この節では Falk のアルゴリズムの利点を活かせる 2 つの非凸計画問題を紹介し，古典的な方法によってアルゴリズムのリニューアルを試みよう．

3.1. 生産輸送問題

Hitchcock 型輸送問題で，流入口 $i = 1,\ldots,m$ と流出口 $j = 1,\ldots,n$ をそれぞれ工場，倉庫に見立て，生産と輸送に関わる総費用を最小化する問題を**生産輸送問題** (production-transportation problem) と呼ぶ:

$$\left|\begin{array}{lll} \text{最小化} & f(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{m}\left(\sum_{j=1}^{n} c_{ij}x_{ij} + f_i(y_i)\right) & \\ \text{条　件} & \sum_{j=1}^{n} x_{ij} = y_i, & i = 1,\ldots,m \\ & \sum_{i=1}^{m} x_{ij} = b_j, & j = 1,\ldots,n \\ & \boldsymbol{x}\ge \boldsymbol{0},\quad \boldsymbol{0}\le \boldsymbol{y}\le \boldsymbol{u}. & \end{array}\right. \qquad (9)$$

各工場 i の生産費用 f_i は，規模の経済の働きによって単調非減少な凹関数が仮定される．各工場 i から各倉庫 j への輸送費用は線形であるが，線形関数も凹関数なので (9) は分離可能な凹最小化問題である．この問題のネット

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

ワークに頂点 s を加えて各流入口 i と枝 (s,i) で結べば，60年代後半から70年代にかけて盛んに研究された**最小凹費用流問題**(minimum concave cost flow problem)に帰着されるが，(9)では線形費用枝の数 mn に比べて非線形費用枝の数 m が著しく少ないことが大きな特徴となっている．また，工場の数 m が2に固定されれば，(9)は $O(n\log n)$ の計算時間で大域的に最適化できることも知られている[10].

問題(9)に対して分枝限定法を用いる場合，輸送費用は線形なので $\boldsymbol{y}$-空間の矩形 $M=\{\boldsymbol{y}\in\mathbb{R}^m \mid \boldsymbol{0}\le\boldsymbol{y}\le\boldsymbol{u}\}$ にだけ分割を行ない，

$$M^k=\{\boldsymbol{y}\in\mathbb{R}^m \mid \boldsymbol{l}^k\le\boldsymbol{y}\le\boldsymbol{u}^k\},\quad k=1,\ldots,r,$$

を生成すればよい．区間 $[l_i^k,u_i^k]$ における f_i の下包 g_i をとってSolandの緩和問題(7)を求めれば，M^k 上の子問題は結局，通常のHitchcock問題に緩和されることがわかる:

$$\left|\begin{array}{ll} \text{最小化} & g(\boldsymbol{x})=\sum_{i=1}^m\sum_{j=1}^n \tilde{c}_{ij}x_{ij}+d \\ \text{条　件} & \sum_{j=1}^n x_{ij}\le u_i,\quad i=1,\ldots,m \\ & \sum_{i=1}^m x_{ij}=b_j,\quad j=1,\ldots,n \\ & \boldsymbol{x}\ge\boldsymbol{0}. \end{array}\right. \tag{10}$$

ただし，g の各定数は下のようにまとめてある:

$$\tilde{c}_{ij}=c_{ij}+\frac{f_i(u_i^k)-f_i(l_i^k)}{u_i^k-l_i^k}$$
$$d=\sum_{i=1}^m\left(f_i(l_i^k)-\frac{f_i(u_i^k)-f_i(l_i^k)}{u_i^k-l_i^k}l_i^k\right).$$

こして得られた(10)の構造には満足できるが，それが与える非力な下界値 $\tilde{f}$ はSolandのアルゴリズム最大の弱点である．それでは，その下界値を強化するには一体どうすればよいだろうか？

問題(10)が整数計画問題に使われる普通の分枝限定法の連続緩和に相当することは前節の中に述べた．整数計画法に照らして次に思いつくのはラグランジュ緩和である．各 j に対してラグランジュ乗数 λ_j を導入し，制約 $\sum_i x_{ij}=b_j$ を緩和してみよう．同時に x_{ij} が倉庫 j の要求量 b_j を越えられないことも考慮すれば，下のラグランジュ緩和問題が得られる:

$$\left|\begin{array}{ll} \text{最小化} & h(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\lambda})= \\ & \sum_{i=1}^m\left(\sum_{j=1}^n c_{ij}(\boldsymbol{\lambda})x_{ij}+f_i(y_i)\right)+b(\boldsymbol{\lambda}) \\ \text{条　件} & \sum_{j\in N}x_{ij}=y_i,\qquad i=1,\ldots,m \\ & \boldsymbol{0}\le\boldsymbol{x}\le\boldsymbol{b},\quad \boldsymbol{y}\in M^k. \end{array}\right. \tag{11}$$

ただし，

$$c_{ij}(\boldsymbol{\lambda})=c_{ij}-\lambda_j;\quad b(\boldsymbol{\lambda})=\sum_{j=1}^n\lambda_j b_j.$$

ラグランジュ乗数 $\boldsymbol{\lambda}$ の値としては様々なものが考えられる（例えば[5]）が，ここでは $\tilde{f}$ よりも強力な下界値がわずかな手間で得られなければ意味がない．そこで提案するのは，線形計画問題(10)対する双対最適解の利用である．つまり，制約 $\sum_i x_{ij}=b_j$ に対する双対最適解 $\tilde{\lambda}_j$ を(11)のラグランジュ乗数 λ_j の値として用いるのである．

問題(11)から制約 $\boldsymbol{x}\le\boldsymbol{b}$ と $\boldsymbol{y}\in M^k$ を取り去り，さらに f_i をその下包 g_i に置き換えれば(10)のラグランジュ緩和問題が得られる．ところが(10)は線形計画問題なので，そのラグランジュ緩和問題の最適値は $\boldsymbol{\lambda}=\tilde{\boldsymbol{\lambda}}$ で最大値 $\tilde{f}$ をとる．図1に示した f_i と g_i の関係，それに追加されている制約 $\boldsymbol{x}\le\boldsymbol{b}$ と $\boldsymbol{y}\in M^k$ の影響から，$\boldsymbol{\lambda}=\tilde{\boldsymbol{\lambda}}$ としたときの(11)の最適値 f^λ は，子問題の最適値 f^* と(10)の最適値 $\tilde{f}$ に対して

$$\tilde{f}\le f^\lambda\le f^* \tag{12}$$

の関係を満足することになる[6].

ラグランジュ緩和問題(11)を解けば，$\tilde{f}$ よりも優れた下界値 f^λ の得られることはわかった．しかし，(11)の目的関数は厄介な凹関数である．実は，このことも恐れるにたりない．問題(11)は m 個の凹最小化問題:

$$\left|\begin{array}{ll} \text{最小化} & \sum_{j=1}^n c_{ij}(\boldsymbol{\lambda})x_{ij}+f_i(y_i) \\ \text{条　件} & \sum_{j\in N}x_{ij}=y_i \\ & 0\le x_{ij}\le b_j,\ j=1,\ldots,n,\quad l_i^k\le y_i\le u_i^k \end{array}\right. \tag{13}$$

に分解できるが，このそれぞれは $m=2$ の生産輸送問題(9)にほかならないのである．したがって，(13)は $O(n\log n)$ の計算時間で解決することができる[10]．これより，ラグラジュ緩和による下界値 f^λ は，Hitchcock問題(10)に要する手間に加え，$O(mn\log n)$ だけ余計な仕事すれば得られることになる．

このラグランジュ緩和を組み込み，$\tilde{f}$ と f^λ の2段構えで限定操作を行なう分枝限定法の実験結果は[7]に報告されている．それによると，150MHzのワークステーション上で $(m,n)=(30,100)$ までのテスト問題が遅くとも5分程度のCPU時間のうちに解かれている．変数の総数は $mn+m=3030$ だから数万変数の問題でも解ける線形計画法に比べればいささか寂しい気もするが，Solandのアルゴリズムが同じテスト問題に対して $(m,n)=(15,50)$ のときにギブアップしていることを考えれば，リニューアルは成功したと考えてよいだろう．最近では400MHzを越えるPCも珍しくないので，現代版ノートの上ではこの倍のサイズの問題でも難なく処理できるはずだ．

© 日本オペレーションズ・リサーチ学会．無断複写・複製・転載を禁ず．

3.2. ナップサック乗法計画問題

次に紹介するのは非線形のナップサック問題である:

$$\left|\begin{array}{ll} \text{最小化} & f(\boldsymbol{x}) = \prod_{i=1}^{m} \left(\sum_{j\in J_i} c_j x_j + d_i\right) \\ \text{条　件} & \sum_{j=1}^{n} a_j x_j \geq b \\ & \boldsymbol{x} \in \{0,1\}^n. \end{array}\right. \tag{14}$$

係数はすべて正であることを仮定するが，このとき目的 f は正の象限上で**準凹関数** (quasiconcave function) となることが知られている [3]. ここではさらに，**多重選択ナップサック問題** (multiple-choice knapsack problem) のように

$$J_i \cap J_\ell = \emptyset,\ i \neq \ell;\ \bigcup_{i=1}^{m} J_i = \{1,\ldots,n\} \tag{15}$$

を仮定する.

ところで，問題 (14) は凹最小化でなければ分離可能でもない. しかし f の対数をとれば，れっきとした凹関数

$$\log f(\boldsymbol{x}) = \sum_{i=1}^{m} \log\left(\sum_{j\in J_i} c_j x_j + d_i\right)$$

となり，しかも (15) から $\sum_{j\in J_i} c_j x_j + d_i$ ごとに分離可能となる. 対数は単調関数なので，f の代わりに $\log f$ を最小化しても (14) の最適解は正しく求められる.

問題 (14) は整数計画問題でもあるので，Soland の下界値 $\tilde{f}$ を求めるには log の下包をとるだけでなく，0-1 変数 x_j 達の連続緩和も必要になる. しかし，それ以外の手続きは生産輸送問題の場合と同じである; つまり，連続な線形ナップサック問題を解いて $\tilde{f}$ と双対最適解 $\tilde{\lambda}$ を求める; 次に $\tilde{\lambda}$ を乗数とするラグランジュ緩和問題を解いて，より緊密な下界値 f^{λ} を生成する. このラグランジュ緩和問題は，仮定 (15) のお陰で $|J_i|$ 個の変数をもつ問題 m 個に分解できる. 分解されたそれぞれの問題は，凹最小化ではあるが変数をソートする手間だけで解くことができる. したがって $\tilde{f}$ と f^{λ} による2段階の限定操作は，連続線形ナップサック問題を解く手間も加えて $O(n \log n)$ の計算時間のうちに終わってしまう [6].

文献 [6] に報告されている実験結果もきわめて良好で，150MHz のワークステーション上で $(m,n) = (20,120)$ までのテスト問題が大体1分以内には処理されている. しかし，この問題 (15) の場合に頭を悩ませるのは効率性より，むしろその使い道である. **線形乗法関数** (linear multiplicative function) f の最適化によって，そこに含まれるアフィン関数達が同時に最適化されると解釈すれば，(14) は予算を共にする独立した m 個の部門からの異なる要求を同時に聞き入れようとするときに役立つかもしれない.

4. おわりに

後半は駆け足になってしまい，読者に「非凸計画問題≠解けない問題」と感じていただけたかどうか心配であるが，もしも引き出しの中に非凸型であるという理由でボツになったモデルがあったら，ほこりを払って問題構造の再確認を願いたい. 分離可能で，非凸に作用する変数の数が比較的少く，さらにネットワークなどの気の利いた構造が見つかれば，その問題はきっと「解ける問題」だ. それを見極めるときに役立つ参考書として最後に [4] を挙げておこう.

参考文献

[1] Falk, J.E. and R.M. Soland, "An algorithm for separable nonconvex programming problems," *Manag. Science* **15** (1969), 550 – 569.

[2] Horst, R. and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer (Berlin, 1993).

[3] Konno, H. and T. Kuno, "Linear multiplicative programming," *Mathematical Programming* **56** (1992), 51 – 64.

[4] Konno, H., P.T. Thach and H. Tuy, *Optimization on Low Rank Nonconvex Structures*, Kluwer (Dordrecht, 1997).

[5] Kubo, M. and H. Kasugai, "A Lagrangean approach to the facility location problem with concave costs," *J. the O.R. Society of Japan* **34** (1991), 125 – 136.

[6] Kuno, T., "Solving a class of multiplicative programs with 0-1 knapsack constraints," ISE-TR-98-147, University of Tsukuba (Ibaraki, 1998) to appear in *J. Opt. Theory and Applications*.

[7] Kuno, T. and T. Utsunomiya, "A Lagrangian based branch-and-bound algorithm for production-transportation problems," ISE-TR-98-150, University of Tsukuba (Ibaraki, 1998).

[8] Soland, R.M., "Optimal facility location with concave costs," *Oper. Res.* **22** (1974), 373 – 382.

[9] Tuy, H., "Concave programming under linear constraints," *Soviet Math.* **5** (1964), 1437 – 1440.

[10] Tuy, H., S. Ghannadan, A. Migdalas and P. Värbrand, "Strongly polynomial time algorithms for certain concave minimization problems on networks," *O.R. Letters* **14** (1993), 99 - 109.

© 日本オペレーションズ・リサーチ学会. 無断複写・複製・転載を禁ず.