Vol.2019-AL-175 No.18
2019/11/29

# $n$ 次元正軸体を半空間で切り取った際の体積

安藤　映[1,a)]　土屋　翔一[1,b)]

**概要**：本稿では $n$ 次元正軸体を半空間で切り取ってできる多面体の体積を考える．正軸体には指数的に多くの面が存在するため，この多面体の体積は単にシンプレックスに分解する方法では効率的に計算できない．提案アルゴリズムはこの体積を計算するのに $O(n^6)$ で実行を完了する．この結果は，$0-1$ ナップサック多面体の体積を計算する問題が $\#P$-困難であるのと対照的である．本研究でこの体積を計算する動機は，次の問いに肯定的な答えがありうると考えるからである：もしも幾何双対の関係にある 2 つの多面体の一方に，体積を計算する多項式時間アルゴリズムが存在する場合，もう一方の多面体の体積の計算にも多項式時間アルゴリズムがいつも存在するか？本稿の結果はその予想を真として矛盾はしない例である．

Ando Ei[1,a)]　Tsuchiya Shoichi[1,b)]

## 1. はじめに

多面体の体積を求める問題は，$n$ 次元では単純なケースでも困難な問題になることがある．例として体積の計算に関する次の結果を挙げる．

- Dyer と Frieze [10] は $0-1$ ナップサック多面体の体積を計算する問題が $\#P$-困難であることを示した．ここで $0-1$ ナップサック多面体とは，$n$ 次元単位ハイパーキューブ $[0,1]^n$ と半空間 $\{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{a} \cdot \boldsymbol{x} \leq b\}$ の重なり部分である．ただし，$\boldsymbol{a} \in \mathbb{Z}_+^n$ を問題の入力とする．
- Khachiyan [13], [14] は $0-1$ ナップサック多面体の幾何双対に当たる多面体の体積を求める問題が $\#P$-困難であることを示した．$0-1$ ナップサック多面体の幾何双対は，正軸体 $\mathrm{conv}(\{\pm\boldsymbol{e}_1, \ldots, \pm\boldsymbol{e}_n\})$ と点 $\boldsymbol{a} \in \mathbb{Z}_+^n$ の凸包である．問題の入力は $\boldsymbol{a}$ である．
- Dyer ら [9] は Zonotope の体積を計算する問題が $\#P$-困難であることを示した． Zonotope とは，$m$ 個のベクトル $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m \in \mathbb{Z}^n$ $(m \geq n)$ によって決まる線分のミンコフスキー和として与えられる多面体である．問題の入力は $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ である．
- 安藤と来嶋 [4] は 2 つの正軸体の重なり部分の体積を求める問題が $\#P$-困難であることを示した．一般性を失うことなく，一方の正軸体は原点を中心とする多面体であり，もう一方の正軸体の中心座標 $\boldsymbol{a}$ および半径 $r$ を問題の入力とする．

一般の $n$ 次元凸体 $K$ を考える．この $K$ はメンバーシップオラクルのみによって定義され，オラクルに問い合わせた座標が $K$ に含まれるかどうかだけの答えが得られる場合を考える．すると，この場合はどのような決定性多項式時間アルゴリズムでも $K$ の体積を計算できないことが知られている．正確には，決定性多項式時間アルゴリズム $\mathrm{Vol}(K)$ を近似する場合，指数的に大きな近似比すら達成できない場合が知られている [6], [11].

興味深いことに，こうした凸体の体積を近似する乱択化アルゴリズムが存在する．特に先に述べた一般の凸体 $K$ の体積については完全多項式時間乱択化近似スキーム (fully polynomial time randomized approximation schemes, FPRAS) [7], [8], [16] が知られている．

最近の研究結果では正確な体積の計算が $\#P$-困難であるような問題でも，決定性完全多項式時間近似スキーム (fully polynomial time approximation scheme, FPTAS) が存在しうることがわかっている．以下にいくつか例を挙げる．

- Lee と Shi [15] は離散的な確率変数の和の分布関数についての FPTAS を示した．このアルゴリズムは $0-1$ ナップサック多面体の体積を近似することができる．彼らのアルゴリズムのアイディアは Štefankovič ら [12], [18] による動的計画法に基づいている．
- 安藤と来嶋 [3] はハイパーキューブを定数個の半空間で切り取ってできる多面体の体積に対する FPTAS を示した ([1] も参照).

[1] 専修大学ネットワーク情報学部
[a)] ando.ei@isc.senshu-u.ac.jp
[b)] s.tsuchiya@isc.senshu-u.ac.jp

- 安藤と来嶋 [2] は $0-1$ ナップサック多面体の幾何双対の体積を求める問題に対して FPTAS を示した．

さて $0-1$ ナップサック多面体とその幾何双対に限って見てみると，次の疑問がある：幾何双対の関係にある 2 つの多面体について，一方の体積を（正確に，または近似的に）多項式時間で計算できるアルゴリズムが存在する場合，もう一方の多面体の体積を計算する多項式アルゴリズムは存在するか？

この疑問の答えを考えるため，本稿では幾何双対の関係にある 2 つの多面体の体積を効率的に計算する方法を考える．

最も単純な例はシンプレックスである．まず $S = \mathrm{conv}(\{\boldsymbol{a}_0, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_n\})$ とすると，$n$ 次元シンプレックス $S$ の幾何双対は別のシンプレックス $S^*$ であるので，明らかに $\mathrm{Vol}(S)$ と $\mathrm{Vol}(S^*)$ の両方について，多項式時間で計算するアルゴリズムが存在する．

もう一つの例として，ハイパーキューブと点 $\boldsymbol{a} \in \mathbb{R}^n$ の凸包を考える．この場合，その幾何双対は正軸体を一つの半空間 $\{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{a} \cdot \boldsymbol{x} \leq b\}$ で切り取ってできる多面体である．ただし，$\boldsymbol{a} \in \mathbb{R}^n$ と $b \in \mathbb{R}$ を問題の入力とする．さて，前者の正確な体積を計算する問題について多項式時間アルゴリズムが存在することはかんたんに確かめられる．この多面体をハイパーキューブと，$n$ 個の「ピラミッド」に分割し，それぞれの体積を計算して和を取ることで全体の体積を求める．このとき「ピラミッド」の底面はハイパーキューブの面 (facet) であり，頂上は $\boldsymbol{a}$ である．このアルゴリズムは線形時間で完了するので，残るは幾何双対である後者の体積に多項式時間アルゴリズムが存在するかどうかが課題である．本稿の主要な内容として，その方法を述べる．

本稿の主定理は次の通りである．

**定理 1.1.** ベクトル $\boldsymbol{a} \in \mathbb{R}^n$ と $b \in \mathbb{R}$ を入力として，原点を中心とする半径 1 の正軸体を半空間 $\{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{a} \cdot \boldsymbol{x} \leq b\}$ で切り取ってできる多面体 $K$ を考える．このとき，$\mathrm{Vol}(K)$ を計算する $O(n^6)$ 時間アルゴリズムが存在する．

この問題を考える上で解決するべき部分は，正軸体には $2^n$ の面があるという点である．したがって $K$ をシンプレックスに分割して，その後にシンプレックスの体積の和を取るようなアルゴリズムでは $n$ の指数に比例する時間がかかる場合が存在し，効率的ではない．この問題を解決するため，$K$ の体積は一種のステップ関数を使うことで定積分の繰り返しとして表現できることを示し，その積分を数式処理の要領で計算するアルゴリズムを示す．提案するアルゴリズムは決定性であるが，証明において確率の議論を用いることでアルゴリズムの正しさが示せる．

本稿の構成は次のとおりである．第 2 節ではハイパーキューブと 1 点の凸包の体積を計算する問題が $O(n)$ 時間で解けることを示す．第 3 節では，正軸体を切り取ってできる多面体 $K$ の体積を求める決定性多項式時間アルゴリズムを示す．第 4 節でまとめと今後の課題を述べる．

## 2. ハイパーキューブと 1 点の凸包の体積

ここでは，ハイパーキューブ $[-1/2, 1/2]^n$ と点 $\boldsymbol{a} \in \mathbb{R}^n$ の凸包の体積を計算する $O(n)$ 時間アルゴリズムについて述べる．

提案アルゴリズムは次のとおりである．ハイパーキューブの面（facet) を $F$ とし，それぞれの $F$ について，点 $\boldsymbol{a}$ から $F$ が可視であるかどうかを調べる．この実現にあたり，$\boldsymbol{f}$ を $F$ の法線（ハイパーキューブ内側から外側に向いている）とすると，次のことがわかる．

**観察 2.1.** 点 $\boldsymbol{a}$ と $\boldsymbol{f}$ が $\boldsymbol{a} \cdot \boldsymbol{f} > 1/2$ を満たすとき，またそのときに限り $F$ は $\boldsymbol{a}$ から可視である．

そして $F$ を底面とし $\boldsymbol{a}$ を頂上とするピラミッド（$F$ と $\boldsymbol{a}$ の凸包）の体積を計算する．このピラミッドの底面積 ($F$ の $n-1$ 次元体積) は 1 であって，高さは $\boldsymbol{a} \cdot \boldsymbol{f} - 1/2$ である．したがって，ピラミッドの体積は $(\boldsymbol{a} \cdot \boldsymbol{f} - 1/2)/n$．処理を高速にするために次のことに注意する：(1) もしも $F$ が $\boldsymbol{a}$ から可視である場合，反対側の面 $F'$ は $\boldsymbol{a}$ から可視でない; (2) $F$ の法線ベクトルは $\pm\boldsymbol{e}_1, \ldots, \pm\boldsymbol{e}_n$ の中のいずれかであり，$\boldsymbol{a} \cdot \boldsymbol{f}$ は $\pm a_1, \ldots, \pm a_n$ のいずれかである．

**Algorithm 1.** *ConvexHull-Hypercube-Point($\boldsymbol{a}$)*
*Input: Vector* $\boldsymbol{a} = (a_1, \ldots, a_n) \in \mathbb{R}^n$.
*1.* $V \leftarrow 0$*;*
*2. For* $i = 1, \ldots, n$ *do*
*3.* *If* $|a_i| \geq 1/2$ *then*
*4.* $V \leftarrow V + (|a_i| - 1/2)/n$*;*
*5. done;*
*6. Output* $V + 1$.

したがって下記が言える．

**命題 2.2.** 入力として $\boldsymbol{a} \in \mathbb{R}^n$ が与えらるとき，$\mathrm{conv}([-1/2, 1/2]^n \cup \{\boldsymbol{a}\})$ の体積は $O(n)$ 時間で計算できる．

## 3. 正軸体を半空間で切り取ってできる体積

本節では定理 1.1 のためのアルゴリズムについて説明する．まず 3.1. 節で基本的な定義を行い，3.2. 節でアルゴリズムの説明を行う．

### 3.1 基本的な定義

中心を $\mathbf{0}$ とする半径 1 の正軸体を $C(\mathbf{0}, 1)$ とする．つまり，

$$C(\mathbf{0}, 1) \stackrel{\mathrm{def}}{=} \mathrm{conv}(\{\pm\boldsymbol{e}_1, \ldots, \pm\boldsymbol{e}_n\}) = \left\{ \boldsymbol{x} \in \mathbb{R}^n \,\middle|\, ||\boldsymbol{x}||_1 = \sum_{i=1,\ldots,n} |x_i| \leq 1 \right\},$$

ただし $\boldsymbol{e}_i$ $(i = 1, \ldots, n)$ は $i$ 番目の成分だけが 1 で他の成

分が 0 のベクトルである．

$n$ 次元ベクトル $\boldsymbol{a} \in \mathbb{R}^n$ および $b \in \mathbb{R}$ に対し，半空間で切り取られた正軸体を $CT(\boldsymbol{a}, b)$ とする．つまり，

$$CT(\boldsymbol{a}, b) \stackrel{\text{def}}{=} C(\mathbf{0}, 1) \cap \{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{a} \cdot \boldsymbol{x} \leq b\}.$$

本稿では $CT(\boldsymbol{a}, b)$ の体積を計算する方法について述べる．一般性を失うことなく，$\boldsymbol{a} \in \mathbb{R}^n_{\geq 0}$ を仮定する．

次の幾何双対性の定義は [17] によるものである．

**定義 3.1.** 点の集合 $K \subseteq \mathbb{R}^n$ について，$K$ の双対 $K^*$ とは次の式で定義される．

$$K^* \stackrel{\text{def}}{=} \{\boldsymbol{y} \in \mathbb{R}^n | \boldsymbol{x} \cdot \boldsymbol{y} \leq 1 \ \ \textit{for all} \ \ \boldsymbol{x} \in K\} \tag{1}$$

ちょうど $K$ の頂点は $K^*$ の面 (facet) を定義する半空間と対応する．例えば，ハイパーキューブ $[-1, 1]^n$ と正軸体 $C(\mathbf{0}, 1)$ は互いに幾何双対の関係にある．ハイパーキューブ $[-1, 1]^n$ の頂点 $\mathbf{1} = (1, 1, \ldots, 1)$ は，半空間 $\{\boldsymbol{x} \in \mathbb{R}^n | \mathbf{1} \cdot \boldsymbol{x} \leq 1\}$ と対応し，この半空間の境界である超平面は正軸体 $C(\mathbf{0}, 1)$ の面である．なお次のことが言える．

**命題 3.2.** $n$ 次元ベクトル $\boldsymbol{a} \in \mathbb{R}^n_{\geq 0}$ と $b \in \mathbb{R}_+$ について，$CT(\boldsymbol{a}, b)$ は $\mathrm{conv}([-1, 1]^n \cup \{\boldsymbol{a}/b\})$ と互いに幾何双対である．

先に見たとおり，$\mathrm{conv}([-1, 1]^n \cup \{\boldsymbol{a}/b\})$ の体積は $O(n)$ 時間で計算完了するということが，本稿で $CT(\boldsymbol{a}, b)$ の体積を多項式時間で計算するアルゴリズムの有無を考察する動機である．なお，提案アルゴリズムは $b \leq 0$ の場合も自然に扱えるので，以下では，$b \in \mathbb{R}$ とする．

さて $f(x)$ を区間 $[-1, 1]$ の一様分布の密度関数とする．つまり，

$$f(x) = \begin{cases} 1/2 & -1 \leq x \leq 1 \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

次にステップ関数 $H(x)$ を考える．本稿では，

$$H(x) = \begin{cases} 0 & x < 0 \\ 1/2 & x = 0 \\ 1 & x > 0 \end{cases}$$

とする．$H(x)$ は平均が 0 で分散が限りなく 0 に近い正規分布の分布関数とも考えられる．本稿では $H(x)$ を使って不連続点のある関数を簡潔に記述する．例えば，$x = \pm 1$ の場合の値の差異を無視して $H(x+1)H(1-x)/2$ を $f(x)$ の先の定義 (2) の代わりに用いる．

今挙げた例のように，ステップ関数を用いた表記を行うと場合分けの表記の場合と比べて不連続点の上 (例えば $x = \pm 1$ の場合の $f(x)$) で誤った値の関数となる．しかしながら，不連続点はステップ関数の引数が 0 である場合に限られていて，求める関数が連続であることが判っていれば，この誤った値は後で修正ができる．そこで次のような「ほぼ至るところで等しい」という関係を定義する．

**定義 3.3.** 2 つの区分的多項式関数 $F(x)$ と $G(x)$ を考える．もしも $x \in \mathbb{R} \setminus I$ において $F(x) = G(x)$ かつ，$I \subseteq \mathbb{R}$ が有限個の点しか含まない場合，

$$F(x) \stackrel{\text{a.e.}}{=} G(x) \tag{3}$$

と書くことにする．$F(x) \stackrel{\text{a.e.}}{=} G(x)$ のことを $F(x)$ と $G(x)$ はほぼ至るところで等しいと言う．

したがって，(2) によって定義される $f(x)$ について，$f(x) \stackrel{\text{a.e.}}{=} H(x+1)H(1-x)/2$ である．もし $F(x) \stackrel{\text{a.e.}}{=} G(x)$ $(F(x) = G(x) \text{ for } x \in \mathbb{R} \setminus I)$ であり，$F(x)$ が連続であれば，$G(x)$ を用いて $c \in I$ について $F(c)$ の値を得ることができる．つまり，$F(c) = \lim_{\epsilon \to 0} G(c - \epsilon)$ である．

2 変数関数についても，同様に「ほぼ至るところで等しい」関係を定義できる．

**定義 3.4.** 2 変数関数 $F(x, y)$ と $G(x, y)$ はどちらも区分的多項式関数であるとする．もし $(x, y) \in \mathbb{R} \setminus I$ について $F(x, y) = G(x, y)$ であり，$I \subseteq \mathbb{R}^2$ が有限個の直線からなる点集合ならば，

$$F(x, y) \stackrel{\text{a.e.}}{=} G(x, y) \tag{4}$$

と書く．

もしも $(c, d) \in I$ であって $F(x, y)$ が連続であるならば，適当な $a, b \in \mathbb{R}$ が存在して

$$F(c, d) = \lim_{\epsilon_1 \to 0} \lim_{\epsilon_2 \to 0} G(c - \epsilon_1 a, d - \epsilon_2 b) \tag{5}$$

として $F(c, d)$ を求められる．特に有限個の場合分けによる $G(x, y)$ の記述が利用できる場合，(5) が一つの値に収束してなおかつ $(a, b)$ と $(c, d)$ の間には $G(x, y)$ の不連続点が存在しないような $a$ と $b$ を選ぶことができる．

加えて，次の用語を定義することで区分的多項式関数の記述に必要な記憶容量の議論ができる．

**定義 3.5.** 区分的多項式関数 $F(x)$ のピース $I$ とは，次の条件を満たすような多項式関数 $G(x)$ が存在するような連続区間でである：

*(1)* $\int_{x \in I} 1 \mathrm{d}x > 0$;

*(2)* $F(z) = G(z)$ を満たすような $z \in I$ が存在する;

*(3)* $x \in I$ について $y \in [x, z] \cup [z, x]$ であるような任意の $y$ について $F(y) = G(y)$ である

### 3.2 提案アルゴリズム

ここでは $\mathrm{Vol}(CT(\boldsymbol{a}, b))$ を求めるアルゴリズムについて述べる．確率の議論を用いることによって，$\mathrm{Vol}(CT(\boldsymbol{a}, b))$ を定積分の繰り返しの形で表現する．この定積分をすべて計算するのにかかる時間が $n$ の多項式で抑えられる．証明は [5] を参照.

最初に，単純なアプローチで $CT(\boldsymbol{a},b)$ の体積を計算することは効率的でないことを注意しておく．正軸体 $C(\mathbf{0},1)$ には $2^n$ 個の面 (facet) があるので，$CT(\boldsymbol{a},b)$ も指数個の面を持ちうる．もしも $CT(\boldsymbol{a},b)$ をシンプレックスに分割することを考えると，指数個のシンプレックスが現れうる．したがって，それらのシンプレックスの個々の体積を求めて和を取ろうとすると $n$ の指数に比例する時間がかかる．

代わりに，$CT(\boldsymbol{a},b)$ の体積を計算するアルゴリズムは確率を考えることで得られる．ランダムなベクトル $\boldsymbol{X}$ が $[-1,1]^n$ 内に一様に分布しているとする．すると，

$$\mathrm{Vol}(CT(\boldsymbol{a},b)) = 2^n \Pr[||\boldsymbol{X}||_1 \le 1 \wedge \boldsymbol{a}\cdot\boldsymbol{X} \le b].$$

この確率を定積分の繰り返しで表現することが，$\mathrm{Vol}(CT(\boldsymbol{a},b))$ を得るアルゴリズムのアイディアである．まず次を定義する．

$$\Phi_0(u,v) = \Pr[0 \le u \wedge 0 \le v] \overset{\text{a.e.}}{=} H(u)H(v)$$

そして $\boldsymbol{X} = (X_1,\ldots,X_n)$，また $\boldsymbol{a} = (a_1,\ldots,a_n)$ とする．すると $\Phi_i(u,v)$ を次の式で定義される 2 変数関数と定義できる．

$$\begin{aligned}
&\Phi_i(u,v)\\
&= 2^i \Pr[|X_1| + \cdots + |X_i| \le u \wedge a_1X_1 + \cdots + a_iX_i \le v]\\
&= 2^i \int_{x_i\in[-1,1]} \Pr\left[\sum_{j=1}^{i-1}|X_j| + |x_i| \le u \right.\\
&\qquad\qquad \left. \wedge \sum_{j=1}^{i-1} a_jX_j + a_ix_i \le v \,\middle|\, X_i = x_i\right] \mathrm{d}x_i\\
&= 2^i \int_{x_i\in\mathbb{R}} \Pr\left[\sum_{j=1}^{i-1}|X_j| + |x_i| \le u \right.\\
&\qquad\qquad \left. \wedge \sum_{j=1}^{i-1} a_jX_j + a_ix_i \le v \right] f(x_i)\mathrm{d}x_i\\
&= 2\int_{x_i\in\mathbb{R}} \Phi_{i-1}(u-|x_i|, v-a_ix_i) f(x_i)\mathrm{d}x_i, \qquad (6)
\end{aligned}$$

ただし $f(x) \overset{\text{a.e.}}{=} H(x+1)H(1-x)/2$ である．すると明らかに $\Phi_n(1,b) = \mathrm{Vol}(CT(\boldsymbol{a},b))$ である．この式 (6) の形から，$\Phi_i(u,v)$ の値のうち $u>1$ に対応する部分は $\Phi_n(1,b)$ の値の計算には不要である．提案アルゴリズムでは，式 (6) を $u \le 1$ について計算する．

定理 1.1 の証明のアイディアを説明する．まず，任意の固定された $u \le 1$ について $\Phi_i(u,v)$ は区分的多項式で，次数は高々 $i$ であり，また，$\Phi_i(u,v)$ のピースの数は高々 $2n+1$ であるといえる．これによって積分を多項式時間で処理できるため $\Phi_n(u,v)$ を多項式時間で計算できる．次の命題が証明に関して重要な役割をはたす．

**命題 3.6.** ベクトル $\boldsymbol{a} \in \mathbb{R}^n_{\ge 0}$ が与えられたとして，$K$ は凸多面体で $K = \mathrm{conv}(\{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_N\})$ とする．ただし，$\boldsymbol{v}_1,\ldots,\boldsymbol{v}_N \in \mathbb{R}^n$ なおかつ $\boldsymbol{a}\cdot\boldsymbol{v}_1 \le \boldsymbol{a}\cdot\boldsymbol{v}_2 \le \cdots \le \boldsymbol{a}\cdot\boldsymbol{v}_N$ を満たすとする．ここで $F_K(\boldsymbol{a},b)$ を

$$F_K(\boldsymbol{a},b) = \mathrm{Vol}(\{\boldsymbol{x}\in K | \boldsymbol{a}\cdot\boldsymbol{x} \le b\}). \qquad (7)$$

と定義すると次のことが言える．

*(1)* $F_K(\boldsymbol{a},b)$ は $b$ の区分的多項式関数であって， 次数は高々 $n$ である*;*

*(2)* 任意に固定された $\boldsymbol{a}$ について，$F_K(\boldsymbol{a},b)$ のピースは高々 $N+1$ である*;*

*(3)* 任意に固定された $\boldsymbol{a}$ について $F_K(\boldsymbol{a},b)$ のピースは $(-\infty,\boldsymbol{a}\cdot\boldsymbol{v}_1]$,$[\boldsymbol{a}\cdot\boldsymbol{v}_N,+\infty)$ と $[\boldsymbol{a}\cdot\boldsymbol{v}_i,\boldsymbol{a}\cdot\boldsymbol{v}_{i+1}]$ $(i=1,\ldots,N)$ で与えられる．

命題 3.6 により，次のことが言える．

**観察 3.7.** ベクトル $\boldsymbol{a} = (a_1,\ldots,a_i)$ について，$a_1 \ge a_2 \ge \cdots \ge a_i \ge 0$ を仮定する．すると，次のことが成立する．

*(1)* $\Phi_i(u,v) = 0$ $(u \le 1, v \le -a_1u)$ また，$\Phi_i(u,v) = u^i/i!$ $(0 \le u \le 1, v \ge a_1u)$ である．

*(2)* 変数 $u \le 1$ を固定したとき $\Phi_i(u,v)$ は区分的多項式関数である．ピースの数はたかだか $2i+1$ である．

*(3)* 関数 $\Phi_i(u,v)$ のピースは $(-\infty,\alpha_1u]$, $[\alpha_{2i}u,+\infty)$ および，$[\alpha_ku,\alpha_{k+1}u]$ $(k=1,\ldots,2i)$ である．ただし，

$$\alpha_k = \begin{cases} -a_k & for\ \ k=1,\ldots,i,\\ a_{2i-k+1} & for\ \ k=i+1,\ldots,2i. \end{cases} \qquad (8)$$

次に，以下のような定義をする．

**定義 3.8.** 2 変数関数 $V_k(u,v)$ を $V_k(u,v) \overset{\text{def}}{=} H(v-\alpha_ku)H(\alpha_{k+1}u-v)$ $(k=1,\ldots,2n-1)$ と定義する．ここで $\Phi_i(u,v)$ $(u\le 1)$ の標準形を次のように定義する．

$$\begin{aligned}
\Phi_i(u,v) \overset{\text{a.e.}}{=}\ & H(u)H(1-u)\sum_{k=0}^{2i} V_k(u,v)p_k(u,v)\\
&+ H(u)H(1-u)H(v+\alpha_{2i}u)u^i/i!
\end{aligned}$$

ただし，$p_k(u,v)$ は $u$ と $v$ の多項式とする．

ここで $u>1$ の場合については計算する必要がないことに注意しておく．もし $u>1$ である場合，$\Phi_n(u,v)$ は多面体 $C(\mathbf{0},u)\cap[-1,1]^n \cap \{\boldsymbol{x}\in\mathbb{R}^n | \boldsymbol{a}\cdot\boldsymbol{x}\le v\}$ の体積と等しい．この多面体は指数的に多くの頂点を持ちうる．

提案アルゴリズムは $u\le 1$ である場合について $\Phi_i(u,v)$ の標準形を $\Phi_{i-1}(u,v)$ $(i=1,\ldots,n)$ の標準形から計算する．ここで $\Phi_i(u,v)$ の標準形は配列変数 $A_i(k)$ と三次元配列変数 $P_i(k,d_u,d_v)$ $(k=0,\ldots,2i,\ d_u=0,\ldots,i,\ d_v=0,\ldots,i)$ によって与えられる．添字の $k$ は先に定義した標準形に現れるシグマの中の変数 $k$ と同じものである．添字 $d_u,d_v$ はそれぞれ $p_k(u,v)$ 中での $u$ および $v$ の次数を表す．配列変数 $A_i(k)$ および $P_i(k,d_u,d_v)$ の値は $\Phi_i(u,v)$ において，標準形の定義中の値 $\alpha_k$ および，多項式 $p_k(u,v)$ で $u,v$ の次

数がそれぞれ $d_u, d_v$ で与えられる項の係数を表す．以下，提案アルゴリズムを示す．

**Algorithm 2.** *Input:* $\boldsymbol{a} \in \mathbb{R}_+^n$ *and* $b \in \mathbb{R}$.
*1. Set* $P_0(0,0,0) := 0, A_0(0) := 0$;
*2. For* $i = 1, \ldots, n$ *do*
*3.  For* $k = 0, \ldots, 2i$ *do*
*4.  Set* $A_i(k) = \alpha_k$ *as given in Observation 2;*
*5.  For all* $(d_u, d_v) \in \{0, 1, \ldots, i\}^2$ *do*
*6.  Compute* $P_i(k, d_u, d_v)$;
*7.  done;*
*8.  done;*
*9. done;*
*10. Find* $k$ *such that* $A_n(k-1) \le b \le A_n(k)$;
*11. Output* $\sum_{d_u=0}^n \sum_{d_v=0}^n P_n(k, d_u, d_v) b^{d_v}$.

最後に $\Phi_n(u,v)$ の値，特に $(u,v) = (1,b)$ の場合の値を求める際には注意が必要である．得られる $\Phi_n(u,v)$ の標準形は $u \le 1$ について「ほとんど至るところで等しい」だけであるので，単純に $(u,v) = (1,b)$ を代入すると誤った値を得る恐れがある．これは $(u,v) = (1,b)$ のときに，標準形の中のステップ関数の引数のうち一つでも 0 になった場合に起こりうる．そのような誤った値を避けるためには，$\Phi_n(1-\epsilon, b-\epsilon')$ の値を適当な $\epsilon, \epsilon' \to 0$ について求める必要があり，それは $\sum_{d_u=0}^n \sum_{d_v=0}^n P_n(k, d_u, d_v) b^{d_v}$ に収束する．

定理 1 は次の補題によって証明できる．

**補題 3.9.** 固定された $\boldsymbol{a} \in \mathbb{R}_+^n$ について，$\Phi_i(u,v)$ の標準形のステップ関数引数が全て 0 でないとき，$\Phi_i(u,v)$ の値と $\Phi_i(u,v)$ の標準形の値は一致する．更に，標準形の中の $p_k(u,v)$ は $u, v$ の次数が高々 $i$ の多項式である．

*Proof.* 帰納法で証明を行う．まず基礎段階として，$\Phi_1(u,v)$ は次によって与えられる．

$$\begin{aligned}
\Phi_1(u,v) &= 2\int_{-\infty}^{\infty} \Phi_0(u-|x_1|, v-a_1x_1) f(x_1) \mathrm{d}x_1 \\
&= \int_{-1}^{1} H(u-|x_1|) H(v-a_1x_1) \mathrm{d}x_1 \\
&= \int_{-1}^{0} H(u) H(u+x_1) H(v-a_1x_1) dx_1 \\
&\qquad + \int_{0}^{1} H(u-x_1) H(v-a_1x_1) \mathrm{d}x_1 \\
&\overset{\text{a.e.}}{=} \min\{1, u, v/a_1\} H(\min\{1, u, v/a_1\}) + \\
&\qquad (\min\{0, v/a_1\} - \max\{-1, -u\}) \\
&\qquad\quad H(\min\{0, v/a_1\} - \max\{-1, -u\})
\end{aligned}$$

これらの max および min はステップ関数をつかって下記のように書き直せる．

$$\max_{i=1,\ldots,\ell}\{p_\ell\} \overset{\text{a.e.}}{=} \sum_{i=1,\ldots,\ell} \prod_{j\neq i} H(p_i - p_j) p_i \tag{9}$$

$$\min_{i=1,\ldots,\ell}\{q_\ell\} \overset{\text{a.e.}}{=} \sum_{i=1,\ldots,\ell} \prod_{j\neq i} H(q_j - q_i) q_i \tag{10}$$

したがって，次のことが言える．

$$\begin{aligned}
&\min\{0, v/a_1\} \overset{\text{a.e.}}{=} H(-v/a_1) v/a_1, \\
&\max\{-1, -u\} \overset{\text{a.e.}}{=} H(-1+u)(-1) + H(-u+1)(-u), \\
&\min\{1, u, v/a_1\} \overset{\text{a.e.}}{=} H(u-1) H(v/a_1 - 1) \\
&\qquad\qquad + H(1-u) H(v/a_1 - u) u \\
&\qquad\qquad + H(1 - v/a_1) H(u - v/a_1) v/a_1
\end{aligned}$$

また，

$$\begin{aligned}
&H\left(\min_{i=1,\ldots,\ell}\{p_i\} - \max_{j=1,\ldots,\ell'}\{q_j\}\right) \\
&\qquad \overset{\text{a.e.}}{=} \sum_{i=1,\ldots,\ell} \prod_{j=1,\ldots,\ell'} H(p_i - q_j)
\end{aligned}$$

が成立するので，

$$\begin{aligned}
&H(\min\{0, v/a_1\} - \max\{-1, -u\}) \\
&\qquad \overset{\text{a.e.}}{=} H(u) H(v/a_1 + 1) H(v/a_1 + u)
\end{aligned}$$

かつ

$$H(\min\{1, u, v/a_1\}) \overset{\text{a.e.}}{=} H(u) H(v/a_1)$$

であるといえる．このやり方で，次のように式を変形できる．

$$\begin{aligned}
&\Phi_1(u,v) \\
&\overset{\text{a.e.}}{=} H(u) H(v/a_1 + 1) H(v/a_1 + u) H(-v/a_1) v/a_1 \\
&\qquad - H(u) H(v/a_1 + 1) H(v/a_1 + u) H(-1+u)(-1) \\
&\qquad - H(u) H(v/a_1 + 1) H(v/a_1 + u) H(-u+1)(-u) \\
&\qquad + H(u) H(v/a_1) H(u-1) H(v/a_1 - 1) \\
&\qquad + H(u) H(v/a_1) H(1-u) H(v/a_1 - u) u \\
&\qquad + H(u) H(v/a_1) H(1 - v/a_1) H(u - v/a_1) v/a_1
\end{aligned} \tag{11}$$

$$\overset{\text{a.e.}}{=} \begin{cases}
0 & u < 0 \ \text{ or } \ v < -a_1 u \\
u + v/a_1 & 0 < u < 1 \ \text{ and } \ -a_1 u < v < a_1 u \\
2u & 0 < u < 1 \ \text{ and } \ a_1 u < v \\
1 + v/a_1 & 1 < u \ \text{ and } \ -a_1 < v < a_1 \\
2 & 1 < u \ \text{ and } \ a_1 < v.
\end{cases} \tag{12}$$

標準形は (12) から直ちに得られ，$u \le 1$ について

$$\begin{aligned}
&\Phi_1(u,v) \\
&\overset{\text{a.e.}}{=} H(u) H(1-u) H(v + a_1 u) H(a_1 u - v)(u + v/a_1) \\
&\qquad + H(u) H(1-u) H(v - a_1 u) 2u
\end{aligned}$$

である．以上により，補題の主張は基礎段階では成立する．

次に帰納段階について述べる．帰納法の仮定として，$\Phi_{i-1}(u,v)$ の標準形が得られており，$p_k(u,v)$ $(k=0,\dots,2i)$ は次数が高々 $i-1$ の多項式とする．関数 $\Phi_i(u,v)$ のピースが与えられる形は観察 3.7 の第三の主張の通りであるので，ここでは多重積分の計算方法を述べる．つまり，$0\le k\le 2i$ について各項 $H(u-|x_i|)H(1-(u-|x_i|))V_k(u-|x_i|,v-a_ix_i)p_k(u-|x_i|,v-a_ix_i)$ を積分するときの手続きを説明する．簡単のため，最終項については少し異なるが同様の処理を行うために省略をする．

さて一つの項を積分した結果は次のようなものになる．ここで $q_k(u-x_i,v-a_ix_i)$ および $\tilde{q}_k(u+x_i,v-a_ix_i)$ は $\frac{d}{dx_i}q_k(u-x_i,v-a_ix_i)=p_k(u-x_i,v-a_ix_i)$ かつ $\frac{d}{dx_i}\tilde{q}_k(u+x_i,v-a_ix_i)=p_k(u+x_i,v-a_ix_i)$ を満たすものとする．場合分けをして，$\alpha_k>0$ かつ $\alpha_{k+1}>0$ である場合，次が言える．

$$\begin{aligned}
&\int_{-1}^{1} H(u-|x_i|)H(1-(u-|x_i|))V_k(u-|x_i|,v-a_ix_i)\\
&\qquad p_k(u-|x_i|,v-a_ix_i)\mathrm{d}x_i\\
&=\int_0^1 H(u-x_i)H(1+x_i-u)V_k(u-x_i,v-a_ix_i)\\
&\qquad p_k(u-x_i,v-a_ix_i)\mathrm{d}x_i\\
&\quad+\int_{-1}^{0} H(u+x_i)H(1-x_i-u)V_k(u+x_i,v-a_ix_i)\\
&\qquad p_k(u+x_i,v-a_ix_i)\mathrm{d}x_i \qquad (13)\\
&\overset{\text{a.e.}}{=}[q_k(u-x_i,v-a_ix_i)]_{S_k(u,v)}^{T_k(u,v)}H(T_k(u,v)-S_k(u,v))\\
&\quad+[\tilde{q}_k(u+x_i,v-a_ix_i)]_{\tilde{S}_k(u,v)}^{\tilde{T}_k(u,v)}H(\tilde{T}_k(u,v)-\tilde{S}_k(u,v)) \qquad (14)
\end{aligned}$$

ただし，$T_k(u,v),\tilde{T}_k(u,v),S_k(u,v),\tilde{S}_k(u,v)$ は $\alpha_k$ および $\alpha_{k+1}$ の値によって決まる．被積分関数はステップ関数の因数 $H(u\mp x_i)H(1\pm x_i-u)$ および

$$\begin{aligned}
V_k(u\mp x_i,v-a_ix_i)=H(v-a_ix_i-\alpha_k(u\mp x_i))\\
H(\alpha_{k+1}(u\mp x_i)-v+a_ix_i)
\end{aligned}$$

を持っているのでので，これらのステップ関数因数は $x_i$ の係数の符号に応じて積分の上限または下限を決める．もしも $\alpha_k+a_i\ge\alpha_k-a_i>0$ かつ $\alpha_{k+1}+a_i\ge\alpha_{k+1}-a_i>0$ である場合には，

$$\begin{aligned}
T_k(u,v)&=\min\left\{1,u,\frac{-v+\alpha_{k+1}u}{\alpha_{k+1}-a_i}\right\}\\
S_k(u,v)&=\max\left\{0,u-1,\frac{-v+\alpha_ku}{\alpha_k-a_i}\right\}\\
\tilde{T}_k(u,v)&=\min\left\{0,1-u,\frac{v-\alpha_ku}{\alpha_k+a_i}\right\}\\
\tilde{S}_k(u,v)&=\max\left\{-1,-u,\frac{v-\alpha_{k+1}u}{\alpha_{k+1}+a_i}\right\}
\end{aligned}$$

である．ここで，$1,0,-1$ は (13) の積分の上下限から得られる．そして (14) は次の式と等しい．

$$\begin{aligned}
&q_k(u-T_k(u,v),v-T_k(u,v))H(T_k(u,v)-S_k)\\
&\quad-q_k(u-S_k(u,v),v-S_k(u,v))H(T_k(u,v)-S_k)\\
&\quad+\tilde{q}_k(u+\tilde{T}_k(u,v),v-\tilde{T}_k(u,v)))H(\tilde{T}_k(u,v)-\tilde{S}_k(u,v))\\
&\quad-\tilde{q}_k(u+\tilde{S}_k(u,v),v-\tilde{S}_k(u,v))H(\tilde{T}_k(u,v)-\tilde{S}_k(u,v))
\end{aligned}$$

ここで min や max はステップ関数を使って書き直しができるので，積分の結果は明らかにステップ関数および $u,v$ の高々次数 $i$ の多項式の積である．したがって，min と max をステップ関数で書き換えると，ステップ関数の引数は次のようになる．

$$\begin{aligned}
&T_k(u,v)-S_k(u,v)\\
&=\min\left\{1,u,\frac{-v+\alpha_{k+1}u}{\alpha_{k+1}-a_i}\right\}-\max\left\{0,u-1,\frac{-v+\alpha_ku}{\alpha_k-a_i}\right\},
\end{aligned}$$

かつ

$$\begin{aligned}
&\tilde{T}_k(u,v)-\tilde{S}_k(u,v)\\
&=\min\left\{0,1-u,\frac{v-\alpha_ku}{\alpha_k+a_i}\right\}-\max\left\{-1,-u,\frac{v-\alpha_{k+1}u}{\alpha_{k+1}+a_i}\right\}
\end{aligned}$$

他の $\alpha_k$ および $\alpha_{k+1}$ の値の組み合わせの場合も同様であるため，ここでは証明を省略する．

関数 $\Phi_i(u,v)$ の標準形は次のようにして得ることができる．任意に 0 に近い $\epsilon>0$ 及び $k=1,\dots,2i$ について，もしも $u=\epsilon$ かつ $v=\alpha_k+\epsilon$ であった場合にステップ関数因数が 0 であるか 1 であるかを調べる．つまり，$\epsilon$ を選ぶ際にどのステップ関数の引数も 0 でないように選ぶ．このとき多項式因数の部分に現れる $u$ や $v$ を $\epsilon$ や $\alpha_k+\epsilon$ で置き換えはしない．すると，$p_k(u,v)$ はステップ関数を 0 ないし 1 で置き換えたときに残る式である．以上により，補題は示された． □

定理 1.1 の証明は次のとおりである

*Proof.* (定理 1.1) 区分的多項式関数 $\Phi_i(u,v)$ 及び途中計算の積分式を記憶するのに必要な記憶領域を考える．関数 $\Phi_i(u,v)$ を格納するのに配列 $A_i(k)$ と配列 $P_i(k,d_u,d_v)$ を使うので $(k=0,\dots,2i,\ d_u=0,\dots,i,\ d_v=0,\dots,i)$，$O(i^3)$ の領域で $u\le 1$ について関数 $\Phi_i(u,v)$ を格納するのに十分である．

さて関数 $\Phi_{i-1}(u-|x_i|,v-a_ix_i)$ を格納するだけの記憶領域は一時的に関数 $\Phi_{i-1}(u,v)$ を格納するための領域に比べて高々 $i$ 倍程度あれば十分である．これは，$(u+x_i)^{i-1}$，$(u-x_i)^{i-1}$，$(v-a_ix_i)^{i-1}$ のような累乗を展開して積の和の形にする際にそれぞれ高々 $i$ 個の項が現れるためである．

ここで，積和の形の多項式の積分をシンボリックに行うということは，各項について積分後の形に書き換えを行うということであり，これは各項について定数時間で実行可

能である．つまり，積分を実行するのは被積分関数の長さについて線形時間で完了する．したがって，高々 $O(i^4)$ 時間あれば積分 $\int_{-1}^{1} \Phi_{i-1}(u-|x_i|, v-a_i x_i)\mathrm{d}x_i$ の結果を得るのに十分である．

積分の結果得られた多項式は標準形で必ずしも標準形ではないので，この式を標準形に書き換える．この処理は高々 $O(i^4) \times 2i = O(i^5)$ 時間で完了する．

以上により，関数 $\Phi_i(u,v)$ の標準形は $\Phi_{i-1}(u,v)$ の標準形から $O(i^5)$ 時間で得られる．よって定理は示された． □

## 4. 結論と今後の課題

本稿では，正軸体を一つの半空間で切り取ってできる多面体の体積について，多項式時間アルゴリズムが存在することを説明した．ハイパーキューブを一つの半空間で切り取ってできる多面体 ($0-1$ ナップサック多面体) の体積を求める問題が $\#P$-困難であるのとは対照的に，正軸体を半空間で切り取った場合は体積の計算は困難ではない．以前の研究 [3], [4] で得られた結果と本稿の結果を合わせてみると，幾何双対性が効率的な体積計算アルゴリズムが存在するかどうかを保存する可能性がある．

今後の課題として，幾何双対の関係にある２つの多面体の間で，効率的な体積計算アルゴリズムの存在が保存されるかどうか，証明あるいは反例を探したい．その他，次の２条件を満たすような多面体 $K$ の体積を求める問題は多項式時間で解決可能と考えている．

(1) $K'$ を $\mathrm{poly}(n)$ 個の頂点からなる多面体とする．
(2) $K$ は $K'$ を定数個の線形制約式（変数の絶対値を含むことを許す）によって切り取ってできる多面体である．

正軸体を半空間で一つの半空間で切り取ってできる多面体の体積を計算するのには $O(n^6)$ かかるのに対し，ハイパーキューブと１点の凸包の体積を計算するのは $O(n)$ で完了する．このような計算時間の差が本質的かどうかは興味深いと考えている．今回の提案アルゴリズムは実行時間を最適化していないため，より高速なアルゴリズムが存在する可能性がある．

### 参考文献

[1] Ando, E.: An FPTAS for computing the distribution function of the longest path length in dags with uniformly distributed edge lengths. Proc. of WALCOM2017, LNCS **10167**, 421–432 (2017)
[2] Ando, E., Kijima, S.: An FPTAS for the volume of a $\mathcal{V}$-polytopes – it is hard to compute the volume of the intersection of two cross-polytopes. arXiv:1607.06173
[3] Ando, E., Kijima, S.: An FPTAS for the volume computation of 0-1 knapsack polytopes based on approximate convolution. Algorithmica **76**(4), 1245–1263 (2016)
[4] Ando, E., Kijima, S.: An FPTAS for the volume of some $\mathcal{V}$-polytopes – it is hard to compute the volume of the intersection of two cross-polytopes. Proc. of COCOON2017, LNCS **10392**, 13–24 (2017)
[5] Ando, E., Tsuchiya, S.: The volume of a crosspolytope truncated by a halfspace. Proc. of TAMC2019, LNCS **11436**, 13–27 (2019)
[6] Bárány, I., Füredi, Z.: Computing the volume is difficult. Discrete Computational Geometry **2**, 319–326 (1987)
[7] Cousins, B., Vempala, S.: Bypassing, K.L.S.: Gaussian cooling and an $O^*(n^3)$ volume algorithm. Proc. of STOC 2015 pp. 539–548 (2015)
[8] Dyer, M., F., A., Kannan, R.: A random polynomial-time algorithm for approximating the volume of convex bodies. Journal of the Association for Computing Machinery **38**(1), 1–17 (1991)
[9] Dyer, M., G., P., Hufnagel, A.: On the complexity of computing mixed volumes. SIAM Journal of Computing **27**(2), 356–400 (1998)
[10] Dyer, M., Frieze, A.: On the complexity of computing the volume of a polyhedron. SIAM Journal on Computing **17**(5), 967–974 (1988)
[11] Elekes, G.: A geometric inequality and the complexity of computing volume. Discrete Computational Geometry **1**, 289–292 (1986)
[12] Gopalan, P., Klivans, A., Meka, R., Štefankovič, D., Vempala, S., Vigoda, E.: An FPTAS for #knapsack and related counting problems. Proc. of FOCS 2011 pp. 817–826 (2011)
[13] Khachiyan, L.: The problem of computing the volume of polytopes is $\#P$-hard. Uspekhi Mat. Nauk. **44**, 199–200 (1989)
[14] Khachiyan, L.: Complexity of polytope volume computation. In: Pach, J. (ed.) New Trends in Discrete and Computational Geometry. Springer (1993)
[15] Li, J., Shi, T.: A fully polynomial-time approximation scheme for approximating a sum of random variables. Operations Research Letters **42**, 197–202 (2014)
[16] Lovász, L., Vempala, S.: Simulated annealing in convex bodies and an $O^*(n^4)$ volume algorithm. Journal of Computer and System Sciences **72**, 392–417 (2006)
[17] Matoušek, J.: Lectures on Discrete Geometry. Graduate Texts in Mathematics, Springer (2002)
[18] Štefankovič, D., Vempala, S., Vigoda, E.: A deterministic polynomial-time approximation scheme for counting knapsack solutions. SIAM Journal on Computing **41**(2), 356–366 (2012)