F-017

SVM を用いた多変量 2 標本検定のパス追跡による高速化とその遺伝子群解析への応用

An Efficient Algorithm for SVM-based Multivariate Two-sample Test Using Path-Following Approach and Its Application to Gene Set

磯部　浩太 †　石川　勇太 †　烏山　昌幸 ‡　泉　泰介 ‡　竹内　一郎 ‡
Kota Isobe　Yuta Ishikawa　Masayuki Karasuyama　Taisuke Izumi　Ichiro Takeuchi

1　まえがき

本稿では, サポートベクトルマシン (SVM) に基づく多変量 2 標本検定について検討する. 多変量 2 標本検定とは, 多変量分布が 2 標本で異なっているかどうかを統計的に評価することである. 2 標本での分布の違いを評価するのに 2 クラス分類器から計算される何らかの検定統計量を用い, その統計的信頼性を評価することで多変量 2 標本検定を行うことができる. 非線形やノンパラメトリックな 2 クラス分類器を利用すれば, 複雑に分かれている 2 標本の違いを検出できる利点がある [1][2]. 本稿では, 2 クラス分類器として SVM を用い, 学習後の SVM から計算される検定統計量を用いて多変量 2 標本検定を行う問題を検討する. しかし, 2 クラス分類器の検定統計量の帰無分布は一般に未知であり, ラベル並べ替え演算などのランダムシミュレーションを用いて推定しなくてはならない [3]. ラベル並べ替え演算により帰無分布を推定するアプローチでは, 分類器 (SVM) の学習 (最適化) をラベル並べ替えの回数分だけ行わなくてはならず, 計算コストが非常に大きくなってしまう. 本稿では, これを効率的に計算するため, ラベル並べ替えサンプルの SVM を効率的に計算するアプローチとして, 階層型クラスタリングによるスケジューリングとパス追跡を用いた方法を提案する.

以下では, n 次元縦ベクトルを $\boldsymbol{v} \in \mathbb{R}^n$ のように表し, $n \times m$ 行列を $\boldsymbol{M} \in \mathbb{R}^{n \times m}$ のように表記する. また, $\mathbb{N}_n$ は 1 から n までの自然数の集合 $\{1, 2, \ldots, n\}$ を表すものとする. さらに, $\boldsymbol{I}$ は適当な次元の単位行列とする.

2　SVM を用いた多変量 2 標本検定

本節では SVM の分類性能を検定統計量とした多変量 2 標本検定についての説明をする.

2.1　多変量 2 標本検定

本節では多変量 2 標本検定について説明する. $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i \in \mathbb{N}_n}$ を n 個の観測データ, $\boldsymbol{x}_i \in \mathbb{R}^p$ を p 次元の入力ベクトル, $y_i \in \{-1, +1\}$ をデータの所属クラスとする. また, 2 つのクラスをそれぞれ C_-, C_+ とし, 各々のデータ数を n_-, n_+ $(n = n_- + n_+)$ とする. さらに, クラス C_-, C_+ に属するデータはそれぞれ多変量分布 P_-, P_+ から生成されたものとする. 多変量 2 標本検定とは, データ $\mathcal{D}$ を用いて, 2 つの多変量分布 P_-, P_+ が同一の分布であるかどうかを統計的仮説検定により評価する問題である. 多変量 2 標本検定の検定統計量として学習後の 2 クラス分類器から計算される検定統計量を利用することができる. 2 標本にラベルをつけて分類器に適用すると, 直感的には, 分類性能がよいほど 2 標本の差が大きく, 分類性能が悪いほど 2 標本の差が小さいと解釈できる. 本稿では, 代表的な 2 クラス分類器であるサポートベクトルマシン (SVM) に関する検定統計量 (後述) を用いた多変量 2 標本検定を行う.

2.2　ラベル並べ替え検定

統計的検定において 2 標本の差の有意性を定量化する (p 値などを求める) には, 検定統計量の帰無分布を知る必要がある. 本節ではラベル並べ替え演算を利用して帰無分布を推定するラベル並べ替え検定を説明する. $\boldsymbol{y} = \{y_1, \ldots, y_n\}$ を観測データのラベルの n 次元ベクトルとする. ラベル並べ替え検定では, $\boldsymbol{y}$ をランダムに並べ替え, 並べ替えたラベルに対して検定統計量を求めることを繰り返すことで, 検定統計量の帰無分布を推定する. ラベルをランダムに並べ替えることによって仮想的な差のない 2 つの分布 (帰無仮説) を作成しているものと解釈することができる. 実用的には, ランダムなラベル並べ替え演算を 1000〜10000 回繰り返して検定統計量の帰無分布を推定し, 得られた帰無分布と正しいラベルの基での統計量を比較することで統計的信頼性を評価することができる.

†名古屋工業大学, Nagoya Institute of Technology
‡東京工業大学, Tokyo Institute of Technology

2.3 サポートベクトルマシンと検定統計量

本節ではサポートベクトルマシン (SVM) について説明を行う．学習データを $\{(\boldsymbol{x}_i, y_i)\}_{i\in\mathbb{N}_n}, \boldsymbol{x}_i \in \mathbb{R}^p, y_i \in \{-1, +1\}$ とすると, SVM の学習は以下のような二次計画問題として定式化される:

$$\min_{b,\boldsymbol{w}} \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i\in\mathbb{N}_n} \max(0, 1 - y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b)). \quad (1)$$

ここで, $C \in [0, \infty)$ は正則化パラメータである．ラグランジュ未定乗数として $\{\alpha_i\}_{i\in\mathbb{N}_n}$ を導入すると, (1) の双対問題は

$$\begin{aligned} \max_{\{\alpha_i\}_{i\in\mathbb{N}_n}} \quad & -\frac{1}{2}\sum_{i\in\mathbb{N}_n}\sum_{j\in\mathbb{N}_n} \alpha_i\alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sum_{i\in\mathbb{N}_n} \alpha_i \\ \text{s.t.} \quad & \sum_{i\in\mathbb{N}_n} \alpha_i y_i = 0,\ 0 \le \alpha_i \le C, \end{aligned}$$

と表される．ただし, $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ はカーネル関数を表している．主形式および双対形式における分類境界は, それぞれ,

$$f(\boldsymbol{x}) = \boldsymbol{w}^\top \boldsymbol{x} + b = 0, \quad (2)$$

および,

$$f(\boldsymbol{x}) = \sum_{i\in\mathbb{N}_n} \alpha_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}_j) + b = 0, \quad (3)$$

と与えられる．また, 最適性条件 (KKT 条件) は,

$$\begin{aligned} \alpha_i = 0 \quad &\Leftrightarrow \quad y_i f(\boldsymbol{x}_i) \ge 1, & (4)\\ 0 \le \alpha_i \le C \quad &\Leftrightarrow \quad y_i f(x_i) = 1, & (5)\\ \alpha_i = C \quad &\Leftrightarrow \quad y_i f(x_i) \le 1, & (6)\\ \sum_{i\in\mathbb{N}_n} y_i\alpha_i \quad &= \quad 0, & (7) \end{aligned}$$

と整理される．

2 標本が大きく離れている場合, 各データ点と分類境界 ((2), もしくは, (3)) の距離が大きくなると思われる (データ点 $(\boldsymbol{x}_i, y_i)$ の分類境界までの距離は $y_i f(\boldsymbol{x}_i)$ と計算される). 本稿では, 評価サンプルに対する SVM 分類境界までの平均距離を検定統計量とする．Leave-one-out 交差確認により評価サンプルを作成するとし, データ i を除いて学習された SVM 分類境界を $f^{(-i)}$ とすると, 本稿で採用する検定統計量は

$$s := n^{-1}\sum_{i\in\mathbb{N}_n} y_i f^{(-i)}(\boldsymbol{x}_i), \quad (8)$$

と定義される．なお, 本稿の趣旨は帰無分布推定時の計算コストを削減することであるため, 検定統計量のよさ (8) に関しては本稿で議論しない.

2.4 遺伝子群解析への応用

本節では多変量 2 標本検定が遺伝子群解析に応用できることを説明する．遺伝子群解析では遺伝子群と呼ばれる単位でマイクロアレイデータの解析を行う．遺伝子群解析の目的は多数の遺伝子群の中でもっとも発現パターンの異なる遺伝子群を同定し, 統計的信頼性を評価することである．この時, 多数の遺伝子群に対して統計的検定を行うと本来有意でない遺伝子群が有意であると判断されてしまう可能性が増加してしまう．この問題は多重検定問題と呼ばれ, 多重検定補正を行わなければならない．検定統計量が独立である多重検定の場合, 様々な補正法を適用できる．一方, 遺伝子群解析のように検定統計量が複雑な相関を持つ場合には相関を考慮した多重検定補正が必要となる．このような状況では 2.2 節で述べたラベル並べ替え検定が有用である．

3 SVM ラベル並べ替え解の計算の効率化

ラベル並べ替え検定を行う際に, ラベル並べ替え回数は 1000〜10000 回と多い．従って, これらのラベル群それぞれに対して SVM の学習を行うと, 計算コストが膨大になる．そのため本稿では以下の 2 つのアプローチにより SVM の学習の効率化を図る:

(i) パス追跡による並べ替えラベル群の最適解の追跡

(ii) 階層型クラスタリングによる効率的なパス追跡スケジューリング

(i) のパス追跡とは, 学習データ点の追加, 削除や正則化係数の変化などが起こった場合, 最初から学習しなおすのではなく, 最適解の感度分析に基づいて変更後の最適解を効率的に計算する方法である．本稿では, 複数のデータ点を追加することができるパス追跡 [4] を用いる．学習データが 0 の時点から, パス追跡により学習データを追加して最適解を求めることは可能である．その場合, 計算量は一般的な SVM の最適解を求める学習アルゴリズムより遅くなってしまう．そのため, 本稿ではデータを一度に追加するのではなく, 共通のラベルのデータのみ追加し, その後残りのデータを追加することで効率化を図る．3.1 節にて (i) について説明する.

続いて, どのようにして効率的なデータの追加を行うかについて説明する．パス追跡はデータの追加数が多い場合, 計算コストが高くなるため, できるだけ総追加数を少なくする必要がある．この問題は (ii) の方法により

対処する．並べ替え検定で生成されたラベル群をクラスタとし, 新しく生成されるクラスタをラベル群の共通部分とした階層型クラスタリングを行う．階層型クラスタリングにより作成された樹形図をもとにパス追跡により並べ替えラベルの最適解を求める．(ii) については 3.2 節にて詳しく説明する．

3.1 SVM パス追跡によるラベル並べ替え解の効率的計算

本節では, 複数の学習データを追加した場合の SVM の最適解のパス追跡を説明する．

まず, (4) - (6) から以下のような集合を定義する:

$$\begin{aligned}\mathcal{O} &:= \{i \mid \alpha_i = 0\},\\ \mathcal{M} &:= \{i \mid 0 < \alpha_i < C\},\\ \mathcal{I} &:= \{i \mid \alpha_i = C\}.\end{aligned}$$

また, 各集合の要素数を $|O|$, $|M|$, $|I|$ と表記する．今後, ベクトル $\boldsymbol{v} \in \mathbb{R}^n$ に対して, $\boldsymbol{v}_{\mathcal{I}}$ と表記する場合, 集合 $\mathcal{I}$ に含まれるインデックスの要素を取り出した部分ベクトルを表すものとする．同様に, 行列 $M \in \mathbb{R}^{n\times n}$ に対して $M_{\mathcal{M},\mathcal{O}}$ とした場合, 行列 M の集合 $\mathcal{M}$ に含まれるインデックスの要素を行および行列 M の集合 $\mathcal{O}$ に含まれるインデックスの要素の列を取り出した部分行列を表すものとする．また, $M_{\mathcal{M},\mathcal{M}}$ のような部分行列は $M_{\mathcal{M}}$ と略記する．

m 個の新しいデータを追加する場合について考える．追加するデータの集合を

$$\mathcal{A} = \{n+1, n+2, \cdots, n+m\},$$

とし, 学習データおよび, パラメータは $\{(\boldsymbol{x}_i, y_i)\}_{i\in\mathbb{N}_{(n+m)}}$, $\boldsymbol{\alpha} = [\alpha_1, \cdots, \alpha_{n+m}]^\top$ とする．追加するデータの初期値は $\alpha_i = 0, i \in \mathcal{A}$ とする．もし $y_i f(x_i) > 1, i \in \mathcal{A}$ を満たす要素があるならば, その要素はすでに KKT 条件 (4) を満たしている．そのため, 集合 $\mathcal{A}$ から削除し, 集合 $\mathcal{O}$ に追加する．同様に, $y_i f(x_i) = 1, i \in \mathcal{A}$ ならば, 集合 $\mathcal{A}$ から削除し, 集合 $\mathcal{M}$ に追加する．その結果, α_i, $y_i f(\boldsymbol{x}_i)$, $i \in \mathcal{A}$ は以下の条件を満たしていることがわかる:

$$0 \le \alpha_i \le C,\ y_i f(\boldsymbol{x}_i) < 1,\ i \in \mathcal{A}. \tag{9}$$

パス追跡の目的は, $\boldsymbol{\alpha}_{\mathcal{A}}$ を少しずつ増やしていき, すべての要素が KKT 条件を満たすようにすることである．しかし, $\boldsymbol{\alpha}_{\mathcal{A}}$ の最適解はわからないため, どのような方向に進めていけばよいのかわからない．そのため, 更新幅 $\Delta\boldsymbol{\alpha}_{\mathcal{A}}$ をステップ幅 $\eta > 0$ を用いて, 以下のように定義する:

$$\Delta\boldsymbol{\alpha}_{\mathcal{A}} := \eta(C\mathbf{1} - \boldsymbol{\alpha}_{\mathcal{A}}).$$

今後, 演算子 Δ は各変数の変化量を表すものとする．次に $\boldsymbol{\alpha}_{\mathcal{A}}$ を $\Delta\boldsymbol{\alpha}_{\mathcal{A}}$ だけ変化させた時の $\Delta\boldsymbol{\alpha}$, Δb について考える．$\boldsymbol{\alpha}_{\mathcal{A}}$ を変化させても, KKT 条件を満たさなければならないため, KKT 条件 (4) - (7) から

$$\sum_{j\in\mathcal{A}} Q_{ij}\Delta\alpha_j + \sum_{j\in\mathcal{M}} Q_{ij}\Delta\alpha_j + y_i\Delta b \ = \ 0,\ i \in \mathcal{M}, \tag{10}$$

$$\sum_{j\in\mathcal{A}} y_j\Delta\alpha_j + \sum_{j\in\mathcal{M}} y_j\Delta\alpha_j \ = \ 0, \tag{11}$$

となることがわかる．次に行列式 M を以下のように定義する:

$$M := \begin{bmatrix} 0 & \boldsymbol{y}_{\mathcal{M}}^\top \\ \boldsymbol{y}_{\mathcal{M}} & \boldsymbol{Q}_{\mathcal{M}} \end{bmatrix},$$

この行列式 M および式 (10), (11) を用いて Δb, $\Delta\boldsymbol{\alpha}_{\mathcal{M}}$ について解くと,

$$\begin{bmatrix} \Delta b \\ \Delta\boldsymbol{\alpha}_{\mathcal{M}} \end{bmatrix} = -M^{-1}\begin{bmatrix} \boldsymbol{y}_{\mathcal{A}}^\top \\ \boldsymbol{Q}_{\mathcal{M},\mathcal{A}} \end{bmatrix}\Delta\boldsymbol{\alpha}_{\mathcal{A}}. \tag{12}$$

となる．Δb, $\Delta\boldsymbol{\alpha}_{\mathcal{M}}$ は $\Delta\boldsymbol{\alpha}_{\mathcal{A}}$ から求めることができることがわかる．また, 他のパラメータは集合 $\mathcal{O}$,$\mathcal{I}$ の定義から

$$\alpha_{\mathcal{O}} = \mathbf{0}, \alpha_{\mathcal{I}} = C\mathbf{1}, \tag{13}$$

となることがわかる．(12), (13) は集合 $\mathcal{I}$, $\mathcal{M}$, $\mathcal{O}$, $\mathcal{A}$ に変化が起きた場合, 更新する必要がある．そのため, 集合の変化を監視する必要がある．各集合は KKT 条件 (4) - (7) および (9) から以下の条件を満たしている:

$$0 \le \alpha_i + \Delta\alpha_i \le C, \qquad i \in \mathcal{M}, \tag{14}$$

$$y_i\{f(\boldsymbol{x}_i) + \Delta f(\boldsymbol{x}_i)\} > 1, \qquad i \in \mathcal{O}, \tag{15}$$

$$y_i\{f(\boldsymbol{x}_i) + \Delta f(\boldsymbol{x}_i)\} < 1, \qquad i \in \mathcal{I}, \tag{16}$$

$$y_i\{f(\boldsymbol{x}_i) + \Delta f(\boldsymbol{x}_i)\} < 1, \qquad i \in \mathcal{A}. \tag{17}$$

もし η を増加させた時に (14) - (17) の条件が破られる (パス追跡の文脈ではイベントと呼ばれる) ようであれば, 集合を変化させ (12), (13) を更新する必要がある．ここで, Δb, $\Delta\boldsymbol{\alpha}_{\mathcal{M}}$ を η を用いて表すと,

$$\begin{bmatrix} \Delta b \\ \Delta\boldsymbol{\alpha}_{\mathcal{M}} \end{bmatrix} = \eta\boldsymbol{\phi}, \tag{18}$$

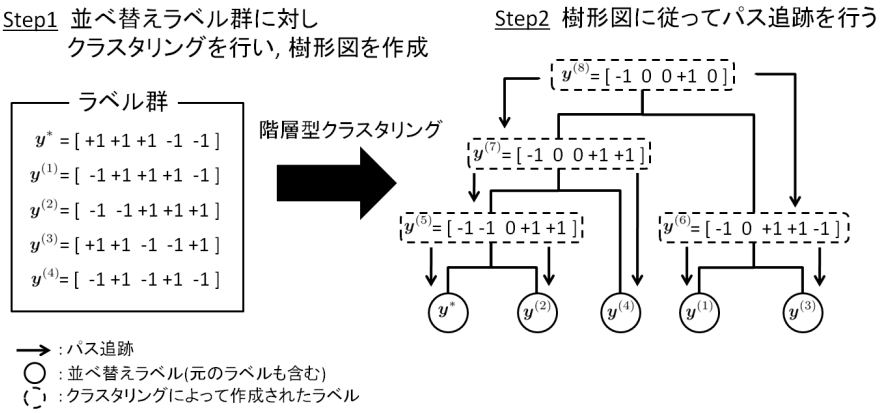

図 1: 階層型クラスタリングを用いたスケジューリングの例

となる. ただし,

$$\boldsymbol{\phi} := -M^{-1}\begin{bmatrix} \boldsymbol{y}_{\mathcal{A}}^{\top} \\ \boldsymbol{Q}_{\mathcal{M},\mathcal{A}} \end{bmatrix}(C\mathbf{1} - \boldsymbol{\alpha}_{\mathcal{A}}),$$

とした. また, $y_i\Delta f(\boldsymbol{x}_i)$ は

$$y_i\Delta f(\boldsymbol{x}_i) = \eta\psi_i,\ i \in \mathbb{N}_{n+m}, \tag{19}$$

となる. ただし,

$$\psi_i := y_i\boldsymbol{Q}_{i,\mathcal{M}}\boldsymbol{\phi} + Q_{i,\mathcal{A}}(C\mathbf{1} - \boldsymbol{\alpha}_{\mathcal{A}}),$$

とした. ここで, 集合 $\mathcal{M}$ 内の要素を $\{m_1, \cdots, m_{|\mathcal{M}|}\}$ とし ϕ_i は $\boldsymbol{\phi}$ の i 番目の要素を表すとする. また, $\min_i\{z_i\}_+$ は $\min_i\{z_i | z_i \geq 0\}$ を簡潔にしたものであるとする. (14) - (17),(18) および (19) から次のイベントが η は

$$\eta = \min_{i\in\{1,\cdots,|\mathcal{M}|\},j\in\mathcal{I},\mathcal{O},\mathcal{A}} \left\{ -\frac{\alpha_{m_i}}{\phi_{i+1}}, \frac{C_{m_i} - \alpha_{m_i}}{\phi_{i+1}}, \frac{1 - y_i f(\boldsymbol{x}_j)}{\psi_j}, 1 \right\}_+$$

となる. これにより得られた η により各パラメータを更新後, イベントの種類に応じて集合の更新を行う. これを $\eta = 1$ となるイベントが発生するまで行い, $\eta = 1$ となるイベントが発生したらパス追跡は終了し, $\mathcal{A}$ を追加した時の SVM の最適解が得られる.

3.2 階層型クラスタリングを用いたスケジューリング

与えられたデータのラベルを $\boldsymbol{y}^* = [y_1, \cdots, y_n]^{\top}, y_i^* \in \{-1, +1\}$ とする. ラベル並べ替え検定により生成されたラベル集合を $\mathcal{Y}_B = \{\boldsymbol{y}^{(b)}\}_{b\in\mathbb{N}_B}, \mathcal{Y} = \{\boldsymbol{y}^*\} \cup \mathcal{Y}_B$ とする. ただし, B は並べ替え回数を表している. $\mathcal{Y}$ に対し階層型クラスタリングを行い, 得られた樹形図に沿ってパス追跡を行っていく. 本稿で用いる階層型クラスタリングは最短距離法を用いる. クラスタ間の類似度は

$$S(\boldsymbol{y}, \boldsymbol{y}') := \max\{\sum_{i\in\mathbb{N}_n} |y_i| I(y_i = y_i'), \sum_{i\in\mathbb{N}_n} |y_i| I(y_i = -y_i')\} \tag{20}$$

として定義する. これは学習データ点のラベルが完全に反転した場合 (y_i' を反転して $-y_i'$ とした場合) でも SVM が同様の学習結果となるため, このような類似度を定義した. このようにラベルを完全に反転させたものもクラスタリングの対象に含むことで, より類似度の高いラベルを見つけることができる.

また, 2 つのクラスタ $\boldsymbol{y}^{(k)}$, $\boldsymbol{y}^{(\ell)}$, から併合されてできた新しいクラスタ $\boldsymbol{y}^{(\mathrm{new})}$ は,

$$y_i^{(\mathrm{new})} = \begin{cases} 0, & \text{if } y_i^{(k)} \neq y_i^{(l)}, \\ y_i^{(k)}, & \text{if } y_i^{(k)} = y_i^{(l)}, \end{cases} \quad i \in \mathbb{N}_n, \tag{21}$$

とする. $y_i = 0$ とは i 番目のデータ点は学習に含まれないことを示している. 本稿で用いた階層型クラスタリングのアルゴリズムを Algorithm 1 に示す. また, 階層型クラスタリングを用いたスケジューリングの例を図 1 に示す.

4 計算機実験

本節では実データを用いた計算機実験を行い, 階層型クラスタリングとパス追跡 (提案法) による効率化の検証

Algorithm 1 階層型クラスタリング

入力: 並べ替えラベル集合 $\mathcal{Y} = \{\boldsymbol{y}^*\} \cup \mathcal{Y}_B$

$\mathcal{Y}_{\text{cluster}} \leftarrow \mathcal{Y}$

while $\mathcal{Y}_{\text{cluster}} > 1$ **do**

$(k, l) = \arg\max D(\boldsymbol{y}^{(k)}, \boldsymbol{y}^{(l)}), \boldsymbol{y}^{(k)}, \boldsymbol{y}^{(l)} \in \mathcal{Y}_{\text{cluster}}$

(21) より $\boldsymbol{y}^{(k)}, \boldsymbol{y}^{(l)}$ から $\boldsymbol{y}^{(\text{new})}$ を求める

$\mathcal{Y}_{\text{cluster}} \leftarrow \mathcal{Y}_{\text{cluster}} \cup \{\boldsymbol{y}^{(\text{new})}\} - \{\boldsymbol{y}^{(k)}\} \cup \{\boldsymbol{y}^{(l)}\}$

$\mathcal{Y} \leftarrow \mathcal{Y} \cup \{\boldsymbol{y}^{(\text{new})}\}$

end while

出力：ラベル群 $\mathcal{Y}$, クラスタリングにより生成された樹形図.

を行う. また, SVM を用いた多変量 2 標本検定を遺伝子群解析に応用し, 得られた結果を GSEA[5] と比較する.

4.1 階層型クラスタリングとパス追跡による効率化の検証

実データを用いて並べ替えたラベル群に対して SVM の学習および Leave-one-out 交差確認を行い, 計算時間の比較する. 比較対象には SVM の代表的な学習法である SMO アルゴリズムを用いる. SMO アルゴリズムは LIBSVM[6] のプログラムに若干の改良を加えたものを用いた. SMO アルゴリズムは初期値を $\boldsymbol{\alpha} = \mathbf{0}$ と設定した場合 (SMO) と, 階層型クラスタリングによるスケジューリングを用いてホットスタートを行った場合 (SMO-hot) の 2 通りで実験を行った. また, Leave-one-out 交差確認には, SMO, SMO-hot ともに全データでの $\boldsymbol{\alpha}$ を初期値としたホットスタートを行った. SMO アルゴリズムの終了条件は 10^{-6} とし, 正則化係数は $C \in \{1, 10, 100\}$ とした. また, カーネル関数はガウシアンカーネル $K(\boldsymbol{x_i}, \boldsymbol{x_j}) = \exp(-\gamma||\boldsymbol{x_i} - \boldsymbol{x_j}||^2)$ を用いた. ハイパーパラメータ γ は $\gamma \in \{0.1/p, 0.5/p, 1/p\}$ とし, 次元数 p で正規化した値を用いた. ラベル並べ替え回数は $B = 1000$ とした. 使用する実データは UCI Machine Learning Repository[1] から取得した Parkinsons($n = 195$, $p = 22$), Ionosphere($n = 351$, $p = 33$), ConnectionistBench($n = 208$, $p = 60$) を用いた. 実験結果を表 1, 2, および, 3 に示す. ただし, 表中の提案法はパス追跡にかかる計算時間のみであるため, 提案法には階層型クラスタリングにかかる計算時間 (表 4) が余分にかかることに注意されたい.

実験結果から, 階層型クラスタリングとパス追跡を用いることで, 検定統計量を (8) とした多変量検定を効率的に行うことができることが分かる.

[1] http://archive.ics.uci.edu/ml/

表 1: 計算時間の比較 (sec.): Ionosphere

γ	手法	$C = 1$	$C = 10$	$C = 100$
$0.1/p$	提案法	262.644	319.298	504.889
	SMO-hot	1369.660	2520.630	9395.180
	SMO	1377.410	2505.370	10254.500
$0.5/p$	提案法	362.606	574.163	702.575
	SMO-hot	1179.800	2898.300	7934.500
	SMO	1178.500	3014.940	8150.150
$1/p$	提案法	394.144	686.599	892.360
	SMO-hot	1084.390	2534.560	7007.930
	SMO	1065.260	2665.980	7161.060

表 2: 計算時間の比較 (sec.): Parkinsons

γ	手法	$C = 1$	$C = 10$	$C = 100$
$0.1/p$	提案法	62.509	63.483	68.446
	SMO-hot	637.747	979.564	1632.840
	SMO	607.413	990.591	1605.840
$0.5/p$	提案法	69.907	77.661	110.525
	SMO-hot	300.847	442.790	1139.220
	SMO	290.053	429.249	1110.290
$1/p$	提案法	72.900	89.212	133.949
	SMO-hot	215.493	333.675	938.300
	SMO	206.761	333.235	910.468

表 3: 計算時間の比較 (sec.): ConnectionistBench

γ	手法	$C = 1$	$C = 10$	$C = 100$
$0.1/p$	提案法	37.348	73.363	205.720
	SMO-hot	75.605	245.266	1563.930
	SMO	76.334	242.405	1499.120
$0.5/p$	提案法	49.590	231.112	355.299
	SMO-hot	104.423	589.487	1153.940
	SMO	106.882	571.481	1147.230
$1/p$	提案法	69.012	353.957	387.078
	SMO-hot	128.320	563.064	703.353
	SMO	124.111	571.842	702.344

表 4: 階層型クラスタリングの計算時間 (sec.)

Ionosphere	2.91656
Parkinsons	1.75573
ConnectionistBench	1.93471

表 5: 遺伝子群解析結果の比較: 太字は SVM, GSEA で検出された遺伝子群を示している.

SVM		GSEA	
遺伝子群	p 値	遺伝子群	p 値
MAP03020 RNA polymerase	0.0028	P53 DOWN	0.0004
GNF FEMALE GENES	0.0120	**MAP00120 Bile acid biosynthesis**	0.0056
MAP00120 Bile acid biosynthesis	0.0207	VOXPHOS	0.0086
tRNA Synthetases	0.0268	ST T Cell Signal Transduction	0.0161
MAP00650 Butanoate metabolism	0.0269	Electron Transport Chain	0.0169
XINACT MERGED	0.0317	ST MONOCYTE AD PATHWAY	0.0304
MAP00020 Citrate cycle TCA cycle	0.0340	MAP00561 Glycerolipid metabolism	0.0322
MAP00710 Carbon fixation	0.0356	MAP00500 Starch and sucrose metabolism	0.0361
TCA	0.0364	SA B CELL RECEPTOR COMPLEXES	0.0385
mef2dPathway	0.0365	ucalpainPathway	0.0392
tumor supressor	0.0366		
MAP00010 Glycolysis Gluconeogenesis	0.0390		
MAP00052 Galactose metabolism	0.0420		

4.2 遺伝子群解析への応用

SVM を用いた遺伝子群解析と GSEA との結果の比較を行う. 実験に用いるデータは GSEA データベース[2][5] から取得可能な C2.Diabetes(症例数 = 34, 遺伝子群数 = 331) を用いる. なお, 遺伝子群は, 各々の遺伝子群に含まれる遺伝子数が 15 個以上 500 個以下となるもののみ用いた. SVM の設定は $\gamma = 10/p, C = 10$ で行った. また, ラベル並べ替え回数は $B = 10000$ とした. SVM および GSEA で有意 (p 値 < 0.05) と判断された遺伝子群を表 5 に示す. [3]

本研究の主目的は, SVM による多変量検定の効率化であるため, 本稿では遺伝子群解析の結果についての考察は行わない.

5 まとめと今後の課題

本稿では, SVM を用いた多変量 2 標本検定の高速化について考察を行った. 並べ替えたラベル群に対する SVM の学習において, 階層型クラスタリングを用いたスケジューリングとパス追跡を行うことで, ラベル並べ替え検定にかかる計算時間の削減を確認した. 今後の課題として, スケジューリングの改善, ラベル並べ替え検定に用いる検定統計量の考察, 遺伝子群解析の結果の医学生物学的知見からの考察などが挙げられる.

[2] http://www.broadinstitute.org/gsea/index.jsp

[3] GSEA の結果は GSEA の Web サイトからダウンロード可能なソフトウェアを利用した. Web サイトおよび, [5] の情報だけでは p 値の計算方法が明確ではないため, 我々が用いた p 値と基準が異なる可能性があることに注意されたい.

参考文献

[1] J. H. Friedman and L. C. Rafsky. Multivariate generalizations of the wald-wolfowits and smirnov two-sample tests. *Annals of Statistics*, 7(4):697–717, 1979.

[2] B. Henze. A multivariate two-sample test based on the number of nearest neighbor type coincidences. *Annals of Statistics*, 16(2):772–783, 1988.

[3] Y. Ge, S. Dudoit, and T. P. Speed. Resampling-based multiple testing for microarray data analysis. *The Statistician*, 45(4):407–436, 1996.

[4] M. Karasuyama and I. Takeuchi. Multiple incremental decremental learning of support vector machines. *IEEE Transactions on Neural Networks*, 21:1048–1059, 2010.

[5] A. Subramanian, P. Tamayo, V. K. Mootha, and S. Mukherjee et al. Gene set enrichment analysis : A knowledge-based approach for interpreting genome-wide expression profiles. *PNAS*, 102(43):15545–15550, 2005.

[6] CC. Chang and CC. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.