行列計算を用いた機械学習法

今倉　暁

観測されたデータや現象を教師データとし，その振る舞いをモデルを用いて再現することで未知のデータの振る舞いを予測する“教師あり機械学習”は，データ解析における重要な問題設定であり，回帰問題や分類問題などに対して，現在さまざまな機械学習アルゴリズムが提案されている．本稿では，特に行列計算に基づく機械学習法として，Ridge 回帰や Lasso などの二乗誤差の最小化に基づく手法，および解析性能の向上のための技術であるカーネル法や次元削減法の概略について紹介する．また，アルゴリズムの実装の際に行列計算の観点からの注意すべき点について数値実験を交えて紹介する．

キーワード：行列計算，機械学習，効率的実装法，カーネル法，次元削減法

1. はじめに

演繹的アプローチに基づく数値解析は，データや現象の振る舞いを記述する支配方程式を立て，その求解によりデータや現象の原理を厳密に解明することを目的とする．一方データ駆動型アプローチに基づくデータ解析は，データや現象の振る舞いの結果を近似的に再現することを目的とし，近年活発に研究されさまざまな分野で利活用されている．特に，観測されたデータや現象を“教師データ”とし，その振る舞いをモデルを用いて再現することで，未知のデータの振る舞いを予測する“教師あり機械学習”は，回帰問題や分類問題などの問題設定に対して，重要な役割として現在さまざまなアルゴリズムが提案されている．

教師データを $\boldsymbol{x}_i$，対応する正解値を y_i とする．教師あり機械学習は，教師データセット $\{\boldsymbol{x}_i, y_i\}_{i=1,2,\ldots,n}$ を再現するモデル $y_i \approx f(\boldsymbol{x}_i, \boldsymbol{w})$ $(i = 1, 2, \ldots, n)$ を構築することで，テストデータ $\boldsymbol{x}_{\mathrm{test}}$ に対する正解値を予測することを目的とする．ここで，f はモデル関数，$\boldsymbol{w}$ は重みパラメータ，n は教師データ数である．

具体的には以下の四つのステップからなる．

Step.1 モデル $f(\boldsymbol{x}_i, \boldsymbol{w})$ の選択・設計

Step.2 目的関数 $D(\{f(\boldsymbol{x}_i, \boldsymbol{w}), y_i\}_{i=1,2,\ldots,n})$ の定義

Step.3 モデルの最適化：

$$\boldsymbol{w}_{\mathrm{opt}} = \arg\min_{\boldsymbol{w}} D(\{f(\boldsymbol{x}_i, \boldsymbol{w}), y_i\}_{i=1,2,\ldots,n})$$

Step.4 テストデータの解析：$y_{\mathrm{test}} = f(\boldsymbol{x}_{\mathrm{test}}, \boldsymbol{w}_{\mathrm{opt}})$

モデル f の選択，目的関数の定義，モデルの最適化法を具体的に設定することで，各種の教師あり機械学習法が設計される．各ステップの選択肢としては，たとえば下記のようなものが挙げられる．

Step.1 モデルの選択
- 線形／非線形回帰モデル
- （ディープ）ニューラルネットワーク
- 決定木
- ランダムフォレスト

Step.2 目的関数の定義
- 二乗誤差
- 相対エントロピー
- 交差エントロピー
- 正則化項

Step.3 モデルの最適化
- （確率的）勾配降下法
- （準）ニュートン法
- 行列分解：QR 分解，特異値分解
- Krylov 部分空間法

本稿では，特に行列計算に基づく機械学習法として，Ridge 回帰や Lasso などの二乗誤差の最小化に基づく手法，および解析性能の向上のための技術であるカーネル法や次元削減法の概略について紹介する．また，アルゴリズムの実装の際に行列計算の観点からの注意すべき点について数値実験を交えて紹介する．

2. 最小二乗法に基づく解法

データの特徴量数を m，データサンプル数を n とし，教師データセットを

$$X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times m},$$
$$\boldsymbol{y} = [y_1, y_2, \ldots, y_n]^\top \in \mathbb{R}^n$$

と置く．以下では，最小二乗法に基づく解法の概要について記す．

いまくら　あきら
筑波大学システム情報系
〒 305–8573 茨城県つくば市天王台 1–1–1
imakura@cs.tsukuba.ac.jp

2.1 最小二乗法

最小二乗法は最も単純な機械学習法の一つであり，モデル関数 f として線形回帰モデル

$$\boldsymbol{y} \approx f(X, \boldsymbol{w}) = X\boldsymbol{w},$$
$$\boldsymbol{w} = [w_1, w_2, \ldots, w_m]^\top \in \mathbb{R}^m$$

を用い，誤差として二乗誤差

$$D(f(X, \boldsymbol{w}), \boldsymbol{y}) = \|\boldsymbol{y} - X\boldsymbol{w}\|_2^2 = \sum_i (y_i - \boldsymbol{x}_i^\top \boldsymbol{w})^2$$

を用いる．

このとき，モデルの最適化は線形最小二乗問題

$$\min_{\boldsymbol{w} \in \mathbb{R}^m} \|\boldsymbol{y} - X\boldsymbol{w}\|_2^2 \tag{1}$$

として定式化され，行列 X のサイズやランクに注意する必要はあるが，行列 X の QR 分解や特異値分解を用いて厳密に求解できる．

一方データ解析においては，モデルの最適化は必ずしも厳密に行う必要はなく，近似解で十分である場合が多い．この場合，最小二乗問題 (1) は，Krylov 部分空間法などの反復法や，近似 QR 分解や近似特異値分解などの行列の近似分解を用いて効率的に求解することができる．最小二乗問題の数値解法については，たとえば文献 [1, 2] を参照されたい．

2.2 正則化

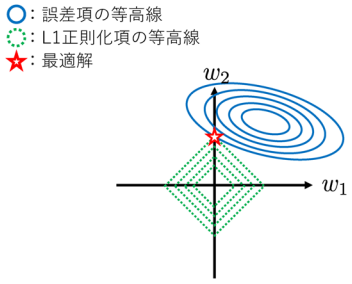
データ解析では，有限個の教師データを用いてモデルの最適化を行う．このとき，単に設定した誤差の最小化を行うと教師データに過剰適合し，未知のデータに対して性能が劣化する．このような過学習を抑制するために，誤差項に加えてモデルの複雑さに対するペナルティを課すことが一般的である．このペナルティを正則化と呼ぶ．代表的な正則化は以下の通りである．

- L2 正則化：$\|\boldsymbol{w}\|_2^2 = \sum_i w_i^2$
- L1 正則化：$\|\boldsymbol{w}\|_1 = \sum_i |w_i|$
- L0 正則化：$\|\boldsymbol{w}\|_0 = \boldsymbol{w}$ の非零要素数
- 組み合わせ：（例）L1 正則化と L2 正則化

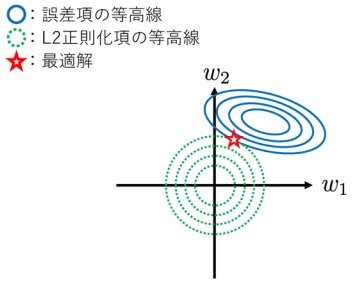
図 1 に 2 次元データに対する L1 正則化と L2 正則化のイメージを示す．L2 正則化を加えることで，教師データへの過剰適合により重み w_1, w_2 が過剰に大きな値を取ることが抑制される．また，L1 正則化を加えることで，重みにゼロ要素が多くなる（図 1(a) の例では $w_1 = 0$ となっている）．

2.3 具体的な機械学習法

最小二乗問題 (1) に前述の正則化を加えることで，具体的な機械学習法が設計される．以下では代表的手法



(a) L1 正則化



(b) L2 正則化

図 1 (a) L1 正則化と (b) L2 正則化のイメージ

である Ridge 回帰 [3], Lasso [4] および Elastic Net [5] の概要について示す．

2.3.1 Ridge 回帰

線形最小二乗問題 (1) に L2 正則化を加えた方法を Ridge 回帰と呼ぶ（Tikhonov の正則化とも呼ばれる）．Ridge 回帰におけるモデルの最適化は

$$\min_{\boldsymbol{w} \in \mathbb{R}^m} \|\boldsymbol{y} - X\boldsymbol{w}\|_2^2 + \lambda \|\boldsymbol{w}\|_2^2 \tag{2}$$

の求解により行う．ここで，$\lambda > 0$ は正則化の重みパラメータである．

ベクトルの 2 ノルムの性質 $\|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2 = \|[\boldsymbol{a}^\top, \boldsymbol{b}^\top]^\top\|_2^2$ から，最適化問題 (2) の目的関数は

$$\|\boldsymbol{y} - X\boldsymbol{w}\|_2^2 + \lambda \|\boldsymbol{w}\|_2^2 = \left\| \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{bmatrix} - \begin{bmatrix} X \\ \sqrt{\lambda} I_m \end{bmatrix} \boldsymbol{w} \right\|_2^2$$

のように変形できる．ここで，I_m は m 次の単位行列である．このため，最適化問題 (2) は，線形最小二乗問題 (1) と同様に，行列 $[X^\top, \sqrt{\lambda} I_m]^\top$ の QR 分解や特異値分解などの厳密解法や Krylov 部分空間法などの近似解法により求解できる．

2.3.2 Lasso

線形最小二乗問題 (1) に L1 正則化を加えた方法を Lasso (Least Absolute Shrinkage and Selection Operator) と呼ぶ．Lasso におけるモデルの最適化は

$$\min_{\boldsymbol{w}\in\mathbb{R}^m} \|\boldsymbol{y}-X\boldsymbol{w}\|_2^2+\lambda\|\boldsymbol{w}\|_1 \tag{3}$$

の求解により行う．

最適化問題 (3) は，直接厳密解を求めることができない．このため，反復法で近似的に求解される．代表的な反復法として，座標降下法（Coordinate Descent, CD 法）や交互方向乗数法（Alternating Direction Method of Multipliers, ADMM 法）[6] が知られている．MATLAB では ADMM 法が標準解法として用いられている．

2.3.3 Elastic Net

線形最小二乗問題 (1) に L1 正則化と L2 正則化の両方を加えた方法を Elastic Net と呼ぶ．Ridge 回帰と同様に，L2 正則化項については二乗誤差と結合し，Lasso と同様に CD 法や ADMM 法などの反復法により求解する．

2.4 正解データが多次元の場合

正解データが ℓ (≥ 2) 次元である場合，つまり $\boldsymbol{y}_i\in\mathbb{R}^\ell$，について考える．このとき，

$$Y=[\boldsymbol{y}_1,\boldsymbol{y}_2,\ldots,\boldsymbol{y}_n]^\top\in\mathbb{R}^{n\times\ell}$$

と置くと，線形回帰モデルの重みは $m\times\ell$ 次の行列 W となり，行列の Frobenius ノルム ($\|A\|_{\mathrm{F}}=\sqrt{\sum_{ij}a_{ij}^2}$) を用いて，最小二乗問題 (1) は

$$\min_{W\in\mathbb{R}^{m\times\ell}} \|Y-XW\|_{\mathrm{F}}^2$$

と書き換えられる．このとき，正則化項としては，L2 正則化の拡張である $\|W\|_{\mathrm{F}}^2$ や L1 正則化の拡張として，

$$\|W\|_{2,1}=\sum_i\sqrt{\sum_j w_{ij}^2}$$

などが用いられる．

3. 解析性能の向上のための技術

本節では，解析性能向上のための技術として，カーネル法および次元削減法について記す．

3.1 カーネル法

解析性能の改善のため，入力データ $\boldsymbol{x}_i\in\mathbb{R}^m$ を

$$\widehat{\boldsymbol{x}}_i=\phi(\boldsymbol{x}_i)\in\mathbb{R}^{\widehat{m}},\quad \widehat{m}>m$$

のように非線形変換することを考える．このような非線形変換は，線形変換のみでは判別・分類できない問題に対して有効である．さまざまな非線形関数 $\phi(\boldsymbol{x})$ が考えられるが，たとえば 2 次元データ $\boldsymbol{x}=[x_1,x_2]^\top$ に対する 2 次の多項式を用いると，以下のようになる．

$$\widehat{\boldsymbol{x}}=\phi(\boldsymbol{x})=[1,x_1,x_2,x_1x_2,x_1^2,x_2^2]^\top\in\mathbb{R}^6.$$

簡単のため正則化項を無視すると，非線形変換されたデータ $\widehat{X}=[\widehat{\boldsymbol{x}}_1,\widehat{\boldsymbol{x}}_2,\ldots,\widehat{\boldsymbol{x}}_n]^\top\in\mathbb{R}^{n\times\widehat{m}}$ に対する線形回帰は，

$$\min_{\widehat{\boldsymbol{w}}\in\mathbb{R}^{\widehat{m}}} \|\boldsymbol{y}-\widehat{X}\widehat{\boldsymbol{w}}\|_2^2 \tag{4}$$

のように定式化され，非線形変換されたデータ $\widehat{X}$ に対する重み $\widehat{\boldsymbol{w}}\in\mathbb{R}^{\widehat{m}}$ が得られる．このような非線形変換により判別・分類性能は向上するものの，一般に $\widehat{m}\gg m$ であるため，最小化問題 (4) の計算コストは大きく増大する．

一方，重み $\widehat{\boldsymbol{w}}$ を

$$\widehat{\boldsymbol{w}}=\widehat{X}^\top\widetilde{\boldsymbol{w}}=\sum_i\widetilde{w}_i\widehat{\boldsymbol{x}}_i,$$
$$\widetilde{\boldsymbol{w}}=[\widetilde{w}_1,\widetilde{w}_2,\ldots,\widetilde{w}_n]^\top\in\mathbb{R}^n$$

のように $\widehat{\boldsymbol{x}}_i$ の線形結合とすると，最小化問題 (4) は

$$\min_{\widetilde{\boldsymbol{w}}\in\mathbb{R}^n} \|\boldsymbol{y}-K\widetilde{\boldsymbol{w}}\|_2^2,\quad K=\widehat{X}\widehat{X}^\top\in\mathbb{R}^{n\times n} \tag{5}$$

と変形される．ここで，$K=\widehat{X}\widehat{X}^\top$ をグラム行列と呼ぶ．高次元データ $\widehat{\boldsymbol{x}}_i=\phi(\boldsymbol{x}_i)$ を計算することなく，直接グラム行列 K を計算することができれば，最小化問題 (5) を用いることで，計算の効率化が可能である．

このように元データの非線形変換を行うことなく，グラム行列 K を

$$K=[k_{ij}],\quad k_{ij}=\phi(\boldsymbol{x}_i)^\top\phi(\boldsymbol{x}_j)=k(\boldsymbol{x}_i,\boldsymbol{x}_j)$$

のように直接計算し，最小化問題 (5) を解くことで線形回帰を行う手法をカーネルトリックと呼ぶ．ここで，関数 $k(\boldsymbol{x}_i,\boldsymbol{x}_j)$ をカーネル関数と呼ぶ．カーネルトリックを用いることで，計算量の削減の他，無限次元の非線形関数を利用することも可能となる．

代表的なカーネルとして，多項式カーネル（$c\geq 0, d$：自然数）

$$k(\boldsymbol{x}_i,\boldsymbol{x}_j)=\left(\boldsymbol{x}_i^\top\boldsymbol{x}_j+c\right)^d,$$

シグモイドカーネル

$$k(\boldsymbol{x}_i,\boldsymbol{x}_j)=\frac{1}{1+\exp(-\gamma\boldsymbol{x}_i^\top\boldsymbol{x}_j)},$$

ガウスカーネル（RBF カーネル）

$$k(\boldsymbol{x}_i,\boldsymbol{x}_j)=\exp\left(-\frac{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2^2}{\sigma^2}\right), \tag{6}$$

などが知られている．

3.2 特徴量選択・次元削減法

機械学習において，数学的には，特徴量の次元数 m が大きいほど教師データにおける最適化問題の関数値を小さくすることができる．しかしながら，実用上は特徴量の次元数が大きすぎると過学習などにより解析性能が低下することが知られており，また解析手法の計算コストが増大するといったデメリットがある．

このような観点から，特徴量の次元数を削減し解析手法の計算時間の削減および解析性能の向上を行う手法として，特徴量選択法および次元削減法がある．

3.2.1 特徴量選択法

元データの一部の特徴量を選択的に抽出し低次元データを作成する方法を特徴量選択と呼ぶ．特徴量選択法は，各特徴量の統計量を独立に評価し選択する Filter 法，選択した特徴量に対する解析モデルの性能により適切な特徴量の組み合わせを選択する Wrapper 法，解析モデルで計算した重みをもとに特徴量の重要度を算出する Embedded 法に分けられる [7].

2.3.2 節で示した Lasso は，L1 正則化により重みにゼロ要素が多くなり，これにより重要度の高い特徴量を選択する Embedded 型の特徴量選択手法としても用いられる．

3.2.2 行列トレースの最適化に基づく次元削減法

射影行列 $B \in \mathbb{R}^{m\times\widetilde{m}}$ を用い，高次元データ $\boldsymbol{x}_i \in \mathbb{R}^m$ を低次元（$\widetilde{m} < m$ 次元）に射影する手法

$$\widetilde{\boldsymbol{x}}_i = B^\top \boldsymbol{x}_i, \quad B \in \mathbb{R}^{m\times\widetilde{m}}$$

を次元削減法と呼び，教師なし次元削減法である主成分分析 (Principal Component Analysis, PCA)，局所性保存射影 (Locality Preserving Projections, LPP)[8] および教師あり次元削減法であるフィッシャー判別分析 (Fisher Discriminant Analysis, FDA)[9]，局所フィッシャー判別分析 (Local FDA, LFDA)[10] などが代表的な手法として知られている．

これらの手法は，各解法においてそれぞれ設定される対称行列 $A_1 \in \mathbb{R}^{m\times m}$ および正定値対称行列 $A_2 \in \mathbb{R}^{m\times m}$ を用い，行列トレースの最適化問題

$$\min_{B\in\mathbb{R}^{m\times\widetilde{m}}} \mathrm{Tr}\left(B^\top A_1 B\right) \text{ or } \max_{B\in\mathbb{R}^{m\times\widetilde{m}}} \mathrm{Tr}\left(B^\top A_1 B\right)$$
$$\text{s.t.} \quad \mathrm{Tr}\left(B^\top A_2 B\right) = 1$$

に帰着され，実対称一般化固有値問題

$$A_1\boldsymbol{u} = \lambda A_2 \boldsymbol{u}, \quad \lambda \in \mathbb{R}, \quad \boldsymbol{u} \in \mathbb{R}^m \setminus \{\boldsymbol{0}\}$$

の最小／最大から $\widetilde{m}$ 個の固有値に対応する固有ベクトルとして求解される．

一方で，上記の方法は $\widetilde{m}$ 次元に次元削減する際に $\widetilde{m}+1$ 番目以降の固有ベクトルを捨てるため，有用な情報を捨てている可能性がある．これに対して近年，複素モーメント型並列固有値解法 [11, 12] のアイディアに基づき，$\widetilde{m}$ 次元データの構築に際し $\widetilde{m}+1$ 本目以降の固有ベクトルも併せて利用する複素モーメント型次元削減法 (Complex Moment-based Supervised Eigenmap, CMSE)[13] が提案され，高い認識性能を実現することが示されている．

4. 行列計算の観点での計算の工夫

近年の計算機においては，CPU における浮動小数点演算の演算速度に対して，CPU とメモリの間でのデータ転送速度が非常に遅い．このため，計算時間は演算回数だけでなくデータ転送量やデータ転送回数に強く依存し，使用するデータ量に対して演算回数が多い計算ほど計算性能（単位時間あたりの浮動小数点演算回数）が高くなることが知られている．基本線形代数ライブラリ (Basic Linear Algebra Subprograms, BLAS) のレベル 1, 2, 3 に対応する代表的な行列計算である，ベクトルの加減算やノルム計算などのベクトル演算（行列の加減算も含む），行列ベクトル積，行列行列積の計算性能は以下の通りである．

行列行列積 $\gg$ 行列ベクトル積 $\gg$ ベクトル演算.

この性質に着目し，本稿では，ガウスカーネルのグラム行列 K の計算 (6) および LPP や LFDA などトレースの最適化に基づく次元削減法で用いられる行列

$$S = \sum_{ij} w_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top \in \mathbb{R}^{m\times m}, \quad (7)$$
$$w_{ij} = w_{ji}$$

の計算について，実装法による計算時間の違いを評価する．本稿では，例として MATLAB コードを示す．

4.1 グラム行列 K の計算法

サンプル数 n，パラメータ σ および教師データ $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times m}$ が `n`，`sigma` および 2 次元配列 `X` として与えられたとし，グラム行列 K を保持する 2 次元配列 `K` を計算する．

4.1.1 ナイーブな実装

ガウスカーネルのグラム行列 K は，式 (6) に基づき下記のように計算される．

```
K = zeros(n);
for j = 1:n
  for i = 1:n
```

```
    K(i,j) =...
      exp(-norm(X(i,:)-X(j,:))^2 / sigma^2);
  end
end
```

上記のナイーブな実装法では，n^2 回のノルム計算が計算コストの主要部である．

4.1.2 対称性を利用した実装

グラム行列の対称性 $K = K^\top$ を利用すると，グラム行列 K の計算は下記のように書き換えられる．

```
K = zeros(n);
for j = 1:n
  K(j,j) = 1;
  for i = 1:j-1
    K(i,j) =...
      exp(-norm(X(i,:)-X(j,:))^2 / sigma^2);
    K(j,i) = k(i,j);
  end
end
```

対称性を利用した実装法では，$n(n-1)/2$ 回のノルム計算が計算コストの主要部である．

4.1.3 行列行列積に基づく効率的な実装

計算性能の低いベクトル演算に基づく上記の実装法に対して，計算効率の高い行列行列積に基づく実装を考える．グラム行列 K の計算式 (6) は

$$K = [k_{ij}], \quad k_{ij} = \exp\left(-\frac{g_{ij}}{\sigma^2}\right), \quad g_{ij} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2$$

のように書くことができる．ここで，

$$g_{ij} = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2 = \boldsymbol{x}_i^\top \boldsymbol{x}_i + \boldsymbol{x}_j^\top \boldsymbol{x}_j - 2\boldsymbol{x}_i^\top \boldsymbol{x}_j$$

であるため，行列 $G = [g_{ij}]$ は，

$$\boldsymbol{g} = [\|\boldsymbol{x}_1\|_2^2, \|\boldsymbol{x}_2\|_2^2, \ldots, \|\boldsymbol{x}_n\|_2^2]^\top \in \mathbb{R}^n \qquad (8)$$

および $\mathbf{1} = [1, 1, \ldots, 1]^\top \in \mathbb{R}^n$ を用い，

$$G = [g_{ij}] = \boldsymbol{g}\mathbf{1}^\top + \mathbf{1}\boldsymbol{g}^\top - 2XX^\top$$

のように計算できる．したがって，ガウスカーネルのグラム行列 K は行列行列積に基づき下記の通り計算される．

```
g = sum(X.^2,2);
G = repmat(g,1,n) + repmat(g',n,1) - 2*X*X';
K = exp(-G/sigma^2);
```

ここで，`g = sum(X.^2,2);` はベクトル $\boldsymbol{g}$ の計算 (8) の MATLAB コマンドであり，`repmat(g,1,n)` および `repmat(g',n,1)` は $\boldsymbol{g}\mathbf{1}^\top$ および $\mathbf{1}\boldsymbol{g}^\top$ を計算する MATLAB コマンドである．

上記の実装法の計算コストの主要部は $XX^\top$ の行列積計算であり，計算量はベクトル演算に基づく実装法とおおむね変わらないものの，計算効率が高く計算時間が短くなることが予想される．

4.2 行列 S の計算法

特徴量次元数 m，サンプル数 n，重み行列 $W = [w_{ij}]$ および教師データ $X = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]^\top \in \mathbb{R}^{n\times m}$ が `m`，`n` および 2 次元配列 `W`，`X` として与えられたとし，式 (7) の行列 S を保持する 2 次元配列 `S` を計算する．

4.2.1 ナイーブな実装

行列 S は，式 (7) に基づき下記のように計算される．

```
S = zeros(m);
for j = 1:n
  for i = 1:n
    xd = X(i,:) - X(j,:);
    S = S + W(i,j) * xd' * xd;
  end
end
```

上記のナイーブな実装法では，n^2 回の行列の加算が計算コストの主要部である．

4.2.2 対称性を利用した実装

重みの対称性 $w_{ij} = w_{ji}$ を利用すると，行列 S の計算は下記のように書き換えられる．

```
S = zeros(m);
for j = 1:n
  for i = 1:j-1
    xd = X(i,:) - X(j,:);
    S = S + W(i,j) * xd' * xd;
  end
end
S = 2 * S;
```

対称性を利用した実装法では，$n(n-1)/2$ 回の行列の加算が計算コストの主要部である．行列 S の対称性も利用することでさらなる計算量の削減が可能である．

4.2.3 行列行列積に基づく効率的な実装

計算性能の低いベクトル演算に基づく上記の実装法に対して，計算効率の高い行列行列積に基づく実装を考える．行列 S の定義式 (7) は

$$\begin{aligned} S &= \sum_{ij} w_{ij}(\boldsymbol{x}_i - \boldsymbol{x}_j)(\boldsymbol{x}_i - \boldsymbol{x}_j)^\top \\ &= 2\sum_i \left(\sum_j w_{ij}\right)\boldsymbol{x}_i\boldsymbol{x}_i^\top - 2\sum_{ij} w_{ij}\boldsymbol{x}_i\boldsymbol{x}_j^\top \\ &= 2X^\top(D - W)X, \\ W &= [w_{ij}], \quad D = \mathrm{diag}(d_i), \quad d_i = \sum_j w_{ij} \end{aligned} \qquad (9)$$

のように書き換えることができる．したがって，行

表 1　グラム行列 K の計算時間（秒）

実装法	サンプル数 n		
	1000	2000	4000
ナイーブな実装	7.78	40.04	202.41
対称性を利用した実装	3.85	19.95	99.35
行列行列積に基づく実装	**0.03**	**0.11**	**0.43**

表 2　行列 S の計算時間（秒）

実装法	サンプル数 n		
	1000	2000	4000
ナイーブな実装	15.29	72.88	300.67
対称性を利用した実装	8.31	36.11	150.72
行列行列積に基づく実装	**0.01**	**0.03**	**0.10**

列 S は行列行列積に基づき下記の通り計算される．

```
D = diag(sum(W));
S = 2 * X' * (D-W) * X;
```

ここで，`D = diag(sum(W));` は対角行列 D の計算 (9) の MATLAB コマンドである．

上記の実装法では，計算量が $1/m$ 程度に削減されている．さらに，計算コストの主要部は $X^\top(D-W)X$ の行列積計算であり，計算効率が高く計算時間が短くなることが予想される．

4.3　各実装法の計算時間の評価

ガウスカーネルのグラム行列 K の計算 (6) および行列 S の計算 (7) に対して，実装法による計算時間の違いを評価する．実験環境は，Windows 10 Pro, Intel® Core™ i7-10710U CPU @ 1.10GHz, 16GB RAM であり，MATALB2019b を使用した．

グラム行列 K の計算 (6) に対しては，特徴量数を $m = 1000$ とし，サンプル数を $n = 1000, 2000, 4000$, $\sigma = 1$ とし，データ X は乱数により生成した．一方，行列 S の計算 (7) に対しては，特徴量数を $m = 100$ とし，サンプル数を $n = 1000, 2000, 4000$ とし，データ X および重み W は乱数により生成した．

実装法毎のグラム行列 K および行列 S の計算時間をそれぞれ表 1 および表 2 に示す．実験結果から，対称性を利用した実装はナイーブな実装と比較して計算時間が半減していることがわかる．これは，対称性を利用することで計算量が半減していることに由来する．

一方で，行列行列積に基づく実装法は，ベクトル演算に基づく実装法と比較して，特に大きな n に対して大幅に計算時間が減少していることがわかる．なお行列 S の計算において，行列行列積に基づく実装法は，ベクトル演算に基づく実装法と比較して計算量が 1/100 程度に削減されているが，計算量削減効果以上の大幅な高速化を実現している．これは，計算時間が計算量だけではなく計算効率に強く依存し，行列行列積がベクトル演算と比較して高速であることに由来する．

5.　おわりに

本稿では，特に行列計算に基づく機械学習法として，Ridge 回帰や Lasso などの二乗誤差の最小化に基づく手法，および解析性能の向上のための技術であるカーネル法や次元削減法の概略について紹介した．また，アルゴリズムの実装の際に行列計算の観点からの注意すべき点について数値実験を交えて紹介した．

数値実験で示されたように，アルゴリズムの計算時間は計算量だけでなく計算効率に強く依存し，効率的な実装を用いることで，時として大幅な高速化が可能である．このため，計算効率の高い行列行列積計算を基盤とした実装を行うことが非常に重要である．また，アルゴリズムによっては行列行列積に基づく効率的な実装を行うことができない場合もあるため，アルゴリズム選択の段階で計算効率を考慮する必要がある．

各種行列計算のアルゴリズムや高性能実装については文献 [1, 2] を参照されたい．

参考文献

[1] 日本応用数理学会監修，櫻井鉄也，松尾宇泰，片桐孝洋編，『数値線形代数の数理と HPC』共立出版，2018.

[2] G. H. Golub and C. F. Van Loan, *Matrix Computations, Fourth Edition*, JHU Press, 2013.

[3] A. E. Hoerl and R. W. Kennard, “Ridge regression: biased estimatin for nonorthogonal problems,” *Technometrics*, **12**, pp. 55–67, 1970.

[4] R. Tibshirani, “Regression shrinkage and selection via the Lasso,” *Journal of the Royal Statistical Society (Series B)*, **58**, pp. 267–288, 1996.

[5] H. Zou and T. Hastie, “Regularization and variable selection via the elastic net,” *Journal of the Royal Statistical Society (Series B)*, **67**, pp. 301–320, 2005.

[6] S. Boyd, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends in Machine Learning*, **3**, pp. 1–122, 2010.

[7] G. Chandrashekar and F. Sahin, “A survey on feature selection methods,” *Computers and Electrical Engineering*, **40**, pp. 16–28, 2014.

[8] X. He and P. Niyogi, “Locality preserving projections,” In *Advances in neural information processing systems*, pp. 153–160, 2004.

[9] R. A. Fisher, “The use of multiple measurements in taxonomic problems,” *Annals of Human Genetics*, **7**, pp. 179–188, 1936.

[10] M. Sugiyama, “Dimensionality reduction of mul-

timodal labeled data by local Fisher discriminant analysis," *Journal of Machine Learning Research*, **8**, pp. 1027–1061, 2007.
[11] T. Sakurai and H. Sugiura, "A projection method for generalized eigenvalue problems using numerical integration," *Journal of Computational and Applied Mathematics*, **159**, pp. 119–128, 2003.
[12] A. Imakura, L. Du and T. Sakurai, "Relationships among contour integral-based methods for solving generalized eigenvalue problems," *Japan Journal of Industrial and Applied Mathematics*, **33**, pp. 721–750, 2017.
[13] A. Imakura, M. Matsuda, X. Ye and T. Sakurai, "Complex moment-based supervised eigenmap for dimensionality reduction," In *Proceedings of the AAAI Conference on Artificial Intelligence*, **33**, pp. 3910–3918, 2019.