

酒井研 基礎勉強会 テキスト

スペースモデリングと凸最適化

酒井智弥

Thursday 5th January, 2017

★ 目次

★ 準備	4
ベクトル・行列に関する表記法	4
★ かけると便利なスクランブル — ランダム射影 —	6
原理	6
試してみよう	7
演習問題	9
★ 方程式が足りない！どう解く？— 劣決定問題と正則化 —	12
劣決定の線形連立方程式	12
線形代数の基礎を少し復習	12
正則化の手法	14
演習問題	15
★ 多すぎるのは良くないことだ — 劣決定問題のスパース解 —	18
劣決定問題のスパース正則化	19
スパース解が一意になる条件	20
スパース解が一意になるための十分条件	21
まとめ	22
演習問題	23
★ スパース解法 — 貪欲法 —	26
OMP	26
演習問題	27
★ スパース解法 — 凸緩和 —	30
スパース解を持つ凸問題	30
最適化アルゴリズム	31
閾値処理に基づく手法	31
ADMM	32
まとめ：結局どの問題をどう解けばよいのか	33
演習問題	33
★ 固有ベクトルで分類 — グラフのスペクトルとデータのクラスタリング —	36
概要：グラフの解析によるデータの分類	36
代数的なグラフの取り扱い	37
グラフのスペクトルと分割	38
スペクトラルクラスタリングのアルゴリズム	39
演習問題	41

★ 2部グラフで同時に分類 — 共クラスタリングと特異値分解 —	44
対応が表すデータのクラスタ	44
2部グラフの切断によるデータの同時分類	44
2部グラフのスペクトラルクラスタリング	45
特異値分解	48
演習問題	48

★ 準備

ベクトル・行列に関する表記法

ベクトルや行列について本書が好んで用いる表記法を紹介する。

スカラーとベクトル 本書では、実数の集合を \mathbb{R} と表す。例えば、 $x \in \mathbb{R}$ という記述は、「 x は実数のスカラー (scalar) である」という意味である。また、2つの実数の組の集合を直積で表し、 $\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$ と記す。例えば、 $\mathbf{x} \in \mathbb{R}^2$ は、 \mathbf{x} が2つの実数の組で表される量であることを意味する。本書では、これを単に「 \mathbf{x} は2次元のベクトル (vector) である」と言う。同様に、 $\mathbf{x} \in \mathbb{R}^d$ (ただし d は正の整数) という記述は、「 \mathbf{x} は d 次元ベクトルである」という意味である。

空間 空間 (space) には、その性質や構造を備えるために内積や距離などが定義されている。例えば、実ベクトル空間は線形演算が定義された実数の配列の集合である。 \mathbb{R}^d は単に d 個の実数からなる配列の全体集合に過ぎないが、特に明示しない場合でも何らかの演算が定義されていることを意図して、本書では \mathbb{R}^d を「 d 次元空間」と単に呼ぶことがある。ただし、必ずしもお馴染みのユークリッド距離が定義されているユークリッド空間ではないかもしれない注意されたい。

行列 m 行 n 列の実数の配列で表される量を実行列 (real matrix) と呼び、その全体集合を $\mathbb{R}^{m \times n}$ と表す。例えば、 $\mathbf{A} \in \mathbb{R}^{m \times n}$ は、「 \mathbf{A} はサイズが $m \times n$ の行列である」という意味である。ふたつの集合 $[m] = \{1, \dots, m\}$ と $[n] = \{1, \dots, n\}$ の直積 $[m] \times [n] = \{(1, 1), (1, 2), \dots, (m, n)\}$ を添え字の集合とすると、行列は添え字から実数への対応付け $[m] \times [n] \rightarrow \mathbb{R}$ を表していると見なせる。その略記が $\mathbb{R}^{m \times n}$ の由来と考えられる。なお、本書では行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ の第 ij 要素を a_{ij} と表記することがある。また、第 j 列を $\mathbf{a}^{(j)} \in \mathbb{R}^m$ と書き表す。

$$\mathbf{A} = [\mathbf{a}^{(1)} \ \mathbf{a}^{(2)} \ \cdots \ \mathbf{a}^{(n)}] \in \mathbb{R}^{m \times n}$$

転置 行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ の行と列の添え字を入れ替えた行列を $\mathbf{A}^\top \in \mathbb{R}^{n \times m}$ と記す¹。これは行列の転置 (transposition) と呼ばれる操作であり、 \top は転置の演算子である。

例：

$$\mathbf{A} = \begin{bmatrix} 1.1 & 1.2 \\ 2.1 & 2.2 \\ 3.1 & 3.2 \end{bmatrix} \in \mathbb{R}^{3 \times 2} \quad \Leftrightarrow \quad \mathbf{A}^\top = \begin{bmatrix} 1.1 & 2.1 & 3.1 \\ 1.2 & 2.2 & 3.2 \end{bmatrix} \in \mathbb{R}^{2 \times 3}$$

この例のように、本書では行列の要素を並べた配列を大括弧で括って記述する。また、複素数を要素とする行列 $\mathbf{A} \in \mathbb{C}^{m \times n}$ に対して、転置行列の複素共役を随伴行列、エルミート転置行列などと呼び、 \mathbf{A}^\dagger や \mathbf{A}^H などと書き表す。

ベクトルの成分 サイズ $d \times 1$ の行列は d 次元の列ベクトルと呼ばれ、サイズ $1 \times d$ の行列は d 次元の行ベクトルと呼ばれる。本書で $\mathbf{x} \in \mathbb{R}^d$ と記したときは、特に断りがない場合、 \mathbf{x} は列ベクトルであるものとする。なお、 d 個の実数 x_1, x_2, \dots, x_d を成分にもつ列ベクトル $\mathbf{x} \in \mathbb{R}^d$ を、紙面の都合上、

¹文献によっては \mathbf{A}^T , ${}^t \mathbf{A}$, \mathbf{A}' などと記されることもある。

$\mathbf{x} = [x_1, \dots, x_d]^\top$ と書き表すことがある.

例 :

$$\mathbf{x} = \begin{bmatrix} 1.1 \\ 2.1 \\ 3.1 \\ 4.1 \end{bmatrix} \in \mathbb{R}^4 \Leftrightarrow \mathbf{x} = [1.1, 2.1, 3.1, 4.1]^\top \Leftrightarrow \mathbf{x}^\top = [1.1, 2.1, 3.1, 4.1]$$

内積 行列の乗算の規則を利用して、次元数が同じ 2 つのベクトル $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$ の内積を本書では $\mathbf{a}^\top \mathbf{b} = a_1 b_1 + a_2 b_2 + \dots + a_d b_d \in \mathbb{R}$ と書き表す. 例 :

$$\mathbf{a} = [1, 2, 3]^\top, \mathbf{b} = [2, -3, 4]^\top \Leftrightarrow \mathbf{a}^\top \mathbf{b} = 1 \cdot 2 + 2 \cdot (-3) + 3 \cdot 4 = 8$$

ノルム ノルム (norm) は、長さ・大きさを表す非負の関数であり、 $\|\cdot\|$ という記号を用いて書き表す. 本書には、何種類かのノルムが登場する. 最も単純な実数 $x \in \mathbb{R}$ に対するノルムは、実数の大きさを表す絶対値 $\|x\| = |x|$ に他ならない. ベクトル $\mathbf{x} \in \mathbb{R}^d$ の p ノルムまたは ℓ_p ノルムと呼ばれるノルムは

$$\|\mathbf{x}\|_p = (\|x_1\|^p + \|x_2\|^p + \dots + \|x_d\|^p)^{\frac{1}{p}}$$

と定義されている. $p = 2$ のときユークリッドノルム、 $p = 1$ のとき絶対値ノルムとも呼ばれる. 行列のノルムもいくつか定義されており、ベクトルのノルムを行列に対して一般化したものが多い. 例えば、行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ の ℓ_p ノルムは

$$\|\mathbf{A}\|_p = \left(\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^p \right)^{\frac{1}{p}}$$

と定義される. 特に、 $p = 2$ のときフロベニウスノルム (Frobenius norm) と呼ばれ、次式のように表記される.

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}$$

これとは異なる行列ノルムの定義もある. シャッテン p ノルム (Schatten p -norm) は、行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ の特異値 (singular values) $\kappa = [\kappa_1, \dots, \kappa_{\min(m,n)}]^\top$ を用いて $\|\mathbf{A}\|_p = \|\kappa\|_p$ と定義された行列ノルムである. \mathbf{A} の特異値は、正方行列 $\mathbf{A}\mathbf{A}^\top$ または $\mathbf{A}^\top \mathbf{A}$ の固有値の平方根であり、 \mathbf{A} の列ベクトルまたは行ベクトルが表すデータの分布の広がりを表す量である. $p = 2$ のとき、シャッテンノルムはフロベニウスノルムと一致する. また、 $p = 1$ のシャッテンノルムは特異値の総和を表す. これを核ノルム (nuclear norm) またはトレースノルム (trace norm)² と呼び、次式のように表記する.

$$\|\mathbf{A}\|_* = \sum_{k=1}^{\min(m,n)} \kappa_k$$

² $\|\mathbf{A}\|_* = \text{tr}(\sqrt{\mathbf{A}^\top \mathbf{A}})$ と書き表せることに由来する.

★ かけると便利なスクランブル — ランダム射影 —

原理

ランダム射影 (random projection) は、乱数で作った行列 \mathbf{R} による線形写像である。 d_x 次元ベクトル $\mathbf{x} \in \mathbb{R}^{d_x}$ から d_p 次元ベクトル $\mathbf{p} \in \mathbb{R}^{d_p}$ へのランダム射影は

$$\mathbf{p} = \mathbf{R}\mathbf{x} \quad (1)$$

と書ける。 \mathbf{x} は d_x 次元ベクトルで、実数の成分を持つどんなベクトルでも構わない。 \mathbf{p} は d_p 次元ベクトルである。 d_x 次元ベクトルを d_p 次元へ写像するランダム行列 \mathbf{R} は、サイズが $d_p \times d_x$ の行列で、次のように書き表すことができる。

$$\mathbf{R} = \begin{bmatrix} r_{11} & \cdots & r_{1d_x} \\ \vdots & \ddots & \vdots \\ r_{d_p 1} & \cdots & r_{d_p d_x} \end{bmatrix} = \begin{bmatrix} \mathbf{r}_1^\top \\ \vdots \\ \mathbf{r}_{d_p}^\top \end{bmatrix} \in \mathbb{R}^{d_p \times d_x}. \quad (2)$$

行列 \mathbf{R} の第 i 行は、横に転置した d_x 次元ベクトル \mathbf{r}_i^\top である。 d_p 本の d_x 次元ベクトル \mathbf{r}_i ($i = 1, \dots, d_p < d_x$) は、図 1 に示すように、 d_x 次元空間 \mathbb{R}^{d_x} における d_p 次元の部分空間を張る。よって、行列 \mathbf{R} によるランダム射影は、任意の d_x 次元ベクトル \mathbf{x} (図中の $\mathbf{x}^{(1)}$ や $\mathbf{x}^{(2)}$) を、この部分空間に正射影して d_p 次元ベクトル \mathbf{p} を作る演算になっている。

$$\mathbf{p} = \mathbf{R}\mathbf{x} = \begin{bmatrix} \mathbf{r}_1^\top \mathbf{x} \\ \vdots \\ \mathbf{r}_{d_p}^\top \mathbf{x} \end{bmatrix} \in \mathbb{R}^{d_p} \quad (3)$$

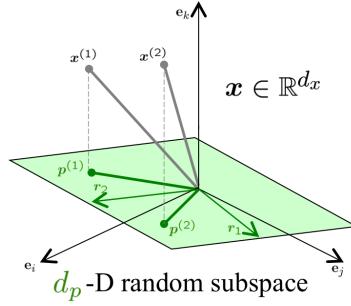


図 1 $\mathbf{x} \in \mathbb{R}^{d_x}$ から $\mathbf{p} \in \mathbb{R}^{d_p}$ へのランダム射影。

例えば、平均ゼロ、分散 $1/d_p$ の正規分布からランダム行列 \mathbf{R} を生成するとしよう。すると、 d_x 次元ベクトルで表されたデータの集合 $X \subset \mathbb{R}^{d_x}$ をランダム射影した低次元データ集合 $P = \{\mathbf{p} \mid \mathbf{p} = \mathbf{R}\mathbf{x}, \mathbf{x} \in X\} \subset \mathbb{R}^{d_p}$ は、 X の様々な性質を高い確率で近似的に保持することが知られている [1]。例えば、2 本のベクトル $\mathbf{x}^{(i)}, \mathbf{x}^{(j)} \in X$ をランダム射影したベクトルをそれぞれ $\mathbf{p}^{(i)}, \mathbf{p}^{(j)} \in P$ とする。

$$\mathbf{p}^{(i)} = \mathbf{R}\mathbf{x}^{(i)}, \quad \mathbf{p}^{(j)} = \mathbf{R}\mathbf{x}^{(j)}$$

すると、 $\mathbf{p}^{(i)}$ のノルムは $\mathbf{x}^{(i)}$ のノルムの良い近似値になる。同様に、2点 $\mathbf{x}^{(i)}, \mathbf{x}^{(j)} \in \mathbb{R}^{d_x}$ の間の距離や内積も高確率で近似的に保存される³。

$$\|\mathbf{p}^{(i)}\|_2 \approx \|\mathbf{x}^{(i)}\|_2, \quad (4)$$

$$\|\mathbf{p}^{(j)} - \mathbf{p}^{(i)}\|_2 \approx \|\mathbf{x}^{(j)} - \mathbf{x}^{(i)}\|_2, \quad (5)$$

$$\mathbf{p}^{(j)\top} \mathbf{p}^{(i)} \approx \mathbf{x}^{(j)\top} \mathbf{x}^{(i)} \quad (6)$$

ランダム射影で近似的に保存される性質を図2に列挙する。計量、多様体の構造、2クラス間のマージン、核関数、主成分等が保存されることが既に解明されている。

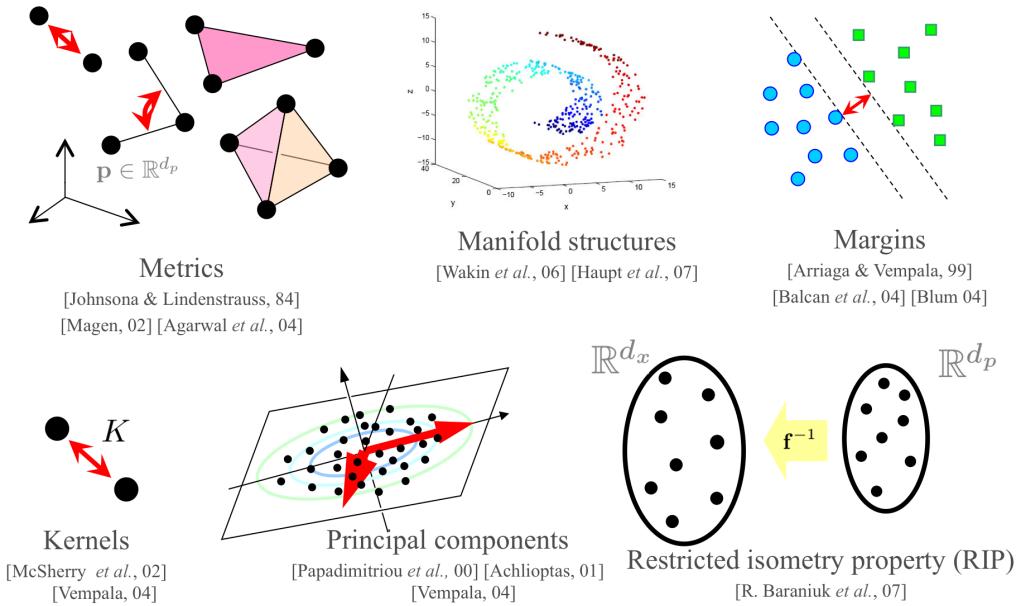


図2 What preserved after random projection.

パターン認識 (pattern recognition; PR) や機械学習 (machine learning; ML) と呼ばれる研究分野では、データの特徴を高次元ベクトルで表して分類や予測を達成する様々な解析手法が設計されている。内積や距離は、2つのデータの間の類似性・非類似性を測った量と見なせる。そのような量がランダム射影で近似的に保存されるということは、低次元データ集合 P を使って高次元データ集合 X を解析できることを意味している。つまり、ランダム射影を使って PR/ML の計算コストを低減できる⁴。

試してみよう

ノルムを比べてみる ランダム射影を実行する単純な MATLAB コードを示す。まず、ランダム行列を作ろう。

```
% データの次元数を設定する。
```

```
dx = 10^5;
dp = 300;
```

³≈は近似的に等しいの意味。‘≒’の記号を使うのは日本だけらしい(?)

⁴ただし、ランダム射影自体の計算コストが問題にならないことが前提である。

```
% ランダム行列を生成する.  
R = randn(dp, dx) / sqrt(dp);
```

そして、適当なベクトル x を作ってランダム射影を適用する。

```
% ランダムなベクトル x を作成する.  
x = randn(dx, 1);  
  
% ランダム射影を計算する.  
p = R * x;  
  
% ノルムを表示する.  
fprintf('||x|| = %1.3e, ||p|| = %1.3e\n', norm(x), norm(p));
```

表示された x のノルムとそのランダム射影 p のノルムがかなり近い値になっていることが確認できるであろう。他の次元数（例えば $dp = 1000$ ）でも試してみるとよい。

顔を比べてみる 顔画像どうしの距離を比べてみよう。

```
% データを読み込む。この例ではYale B cropped face images.  
data = load('YaleB_Ext_Cropped_192x168_all.mat');  
dx = size(datafea, 2);  
dp = 300;  
  
% ランダム行列を生成する.  
R = randn(dp, dx) / sqrt(dp);  
  
% 同一人物の2枚の解画像を表示する.  
imshow( reshape(datafea(100,:), 192, 168) );  
imshow( reshape(datafea(101,:), 192, 168) );  
  
% ランダム射影を計算し、距離を比較する。  
x1 = double(datafea(100,:));  
x2 = double(datafea(101,:));  
p1 = R * x1;  
p2 = R * x2;  
fprintf('||x2-x1|| = %1.3e, ||p2-p1|| = %1.3e\n', norm(x2-x1),  
norm(p2-p1));  
  
% show an image of a different person  
imshow( reshape(datafea(200,:), 192, 168) );  
  
% compute its random projection and compare the distances
```

```

x3 = double(datafea(100,:));
p3 = R * x3;
fprintf('||x3-x1|| = %1.3e, ||p3-p1|| = %1.3e\n', norm(x3-x1),
norm(p3-p1));

```

演習問題

1. 適当に作成した n 本のベクトルについて、ランダム射影前後のノルムの相対誤差 $\varepsilon^{(j)}$ を計算せよ。

$$\varepsilon^{(j)} = \frac{\|\mathbf{p}^{(j)}\| - \|\mathbf{x}^{(j)}\|}{\|\mathbf{x}^{(j)}\|} \quad (j = 1, \dots, n).$$

2. 相対誤差のヒストグラムを作成し、分布を観察せよ。ランダム射影先の次元数 d_p が小さいと誤差はどうなるか。いくつか異なる次元数 d_p についてヒストグラムを作成し、観察せよ。

ヒント：MATLAB では、 e をベクトルとすると、 $hist(e, 30)$ は e の成分についてビン数 30 のヒストグラムを作る。

3. ランダム射影とノルム（または距離）の計算時間を測定せよ。

ヒント：`tic; p = R * x; toc` とすると、 $p = R * x$ の実行にかかった時間が表示される。

- 4.* 計算時間とメモリ使用量がとても少ない効率的なランダム射影のアルゴリズムがある [2, 3]。そのアルゴリズムは、サイズ $d_p \times d_x$ のランダム行列を生成する必要がない。どのようなアルゴリズムなのか調べよ。

- 5.* Johnson-Lindenstrauss の埋め込み [4, 5, 6, 1] とは何か調査せよ。また、ランダム射影後の 2 点間の距離の誤差に関する命題を見つけ、説明せよ。

参考文献

- [1] S.S. Vempala, The Random Projection Method, Series in Discrete Mathematics and Theoretical Computer Science, vol.65, American Mathematical Society, 2004.
- [2] T. Sakai, “An efficient algorithm of random projection by spectrum spreading and circular convolution (in Japanese),” The 24th SIP Symposium Proceedings, pp.507–512, 2009.
- [3] T. Sakai and A. Imiya, “Practical algorithms of spectral clustering: Toward large-scale vision-based motion analysis,” in Machine Learning for Vision-Based Motion Analysis, Advances in Pattern Recognition, ch. 1, pp.3–26, Springer, 2011.
- [4] W. Johnson and J. Lindenstrauss, “Extensions of Lipschitz maps into a Hilbert space,” Contemporary Mathematics, vol.26, pp.189–206, 1984.
- [5] D. Achlioptas, “Database-friendly random projections: Johnson-lindenstrauss with binary coins,” Journal of Computer and System Sciences, vol.66, pp.671–687, 2003.
- [6] S. Dasgupta and A. Gupta, “An elementary proof of the Johnson-Lindenstrauss lemma,” tech. rep., UC Berkeley, 1999.

See also:

[Sakai, 09][2] <https://drive.google.com/file/d/0Bx4bEpaTSFcgOFA5dVZoZXFkRmc>

[Sakai&Imiya,11][3] <https://drive.google.com/file/d/0Bx4bEpaTSFcgZWp2ZDRLa2c1SEE>

The Extended Yale Face Database B <http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html>

cropped images in MATLAB file: [YaleB_Ext_Cropped_192x168_all.mat](#)

<https://drive.google.com/file/d/0Bx4bEpaTSFcgRm15NVlua1pOTWM>

★ 方程式が足りない！どう解く？— 劣決定問題と正則化 —

劣決定の線形連立方程式

未知数の数よりも方程式の数が少ない線形連立方程式 (a system of linear equations) を考えよう。そのような連立方程式は、無数の解を持ち、解をひとつに決められない問題なので、劣決定問題 (underdetermined problem) と呼ばれる。

劣決定問題の例を見てみよう。3個の未知数 x_1, x_2, x_3 ⁵ に対して 2本の方程式しかない連立方程式

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \quad (7)$$

の解は、任意の定数 $t \in \mathbb{R}$ を使って

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} t. \quad (8)$$

と表すことができる。 $x_1 = 0, x_2 = 1, x_3 = 0$ ($t = 0$) でもよいし、 $x_1 = 1, x_2 = -1, x_3 = 1$ ($t = 1$) でもよい。

連立方程式 (7) は、3次元空間における2つの平面 $x_1 + x_2 + x_3 = 1$ と $x_1 + 2x_2 + 3x_3 = 2$ を表している。式 (8) はこの2平面の交わりを表す直線である。直線は3次元空間の点 $[0, 1, 0]^\top$ を通り、方向ベクトルが $[1, -2, 1]^\top$ である。この直線上の任意の点 $\mathbf{x} \in \mathbb{R}^3$ すべてが解の候補であり、与えられた方程式 (7) を満たす。よって、この例は、解が無数に存在し一意に定まらない劣決定問題である。

解を一意に定めるためには、解について更に何か条件が必要である。例えば、線形方程式がもう1本追加されれば、3個の未知数に対して方程式が3本になる。もはや劣決定ではなくなり、3本の線形方程式が表す3平面が交わる1点に解が定まる。

しかし、式 (7) のような劣決定問題に対して、追加の方程式が全く手に入らないときは、どうしようか。無数にある解の候補から、ひとつの解を選ぶ合理的な方法はあるだろうか。方程式の他に、解について何らかの事前情報または事前知識 (prior information/knowledge) を持っているならば、解を特定する条件としてそれを使えないだろうか。例えば、「未知数はゼロになるものが多い」という情報を知っていれば、式 (8) で表される解の候補の中から解 $\mathbf{x} = [0, 1, 0]^\top$ を選べるだろう。ここでは、解をひとつの絞り込む正則化と呼ばれる手法を紹介しよう。

線形代数の基礎を少し復習

連立方程式を行列とベクトルで書き表す 任意の線形連立方程式は、行列とベクトルを使って

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad (9)$$

と書き表せる。ここで、行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ とベクトル $\mathbf{b} \in \mathbb{R}^m$ は与えられるものであるとし、 m 本の連立方程式を表している⁶。ベクトル \mathbf{x} は n 個の未知数を成分を持つベクトルである。連立方程式の本数 m と未知数の個数 n によって、行列 \mathbf{A} は横長 ($m < n$)、正方 ($m = n$)、縦長 ($m > n$) のいずれかになる。

⁵未知数が3個のとき x, y, z の記号を使うことに慣れているかもしれないが、実用上は未知数が3個以上あるかもしれないのに、本書では添え字付きの記号を使う。

⁶例えば、式 (7) の連立方程式では、 $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ である。

連立方程式は線形結合係数を求める問題である 行列 \mathbf{A} は、 m 次元ベクトルが列に n 本並んでいると見ることができる。本書では、行列 \mathbf{A} の第 j 列を $\mathbf{a}^{(j)} \in \mathbb{R}^m$ と表すことにする。連立方程式 (9) は、与えられた n 本の m 次元ベクトル $\mathbf{a}^{(j)}$ ($j = 1, \dots, n$) を使って、 m 次元ベクトル \mathbf{b} を合成するための線形結合係数を求める問題であると解釈できる。なぜならば、

$$\mathbf{b} = \mathbf{Ax} = \begin{bmatrix} \mathbf{a}^{(1)} & \dots & \mathbf{a}^{(n)} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1 \mathbf{a}^{(1)} + \dots + x_n \mathbf{a}^{(n)} \quad (10)$$

と表せるからである。 n 本の m 次元ベクトル $\mathbf{a}^{(j)}$ ($j = 1, \dots, n$) は、 m 次元ベクトル \mathbf{b} を線形結合で合成するための基底 (basis) と呼ばれる。

例えば、式 (7) の劣決定問題は、3 本の列ベクトル $\mathbf{a}^{(1)} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{a}^{(2)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \mathbf{a}^{(3)} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$ を基底として、2 次元ベクトル $\mathbf{b} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$ を

$$\begin{bmatrix} 1 \\ 2 \end{bmatrix} = x_1 \begin{bmatrix} 1 \\ 1 \end{bmatrix} + x_2 \begin{bmatrix} 1 \\ 2 \end{bmatrix} + x_3 \begin{bmatrix} 1 \\ 3 \end{bmatrix}$$

のように合成するとき、線形結合係数 x_1, x_2, x_3 を求める問題であると解釈できる。2 次元ベクトル \mathbf{b} を合成するための基底 $\mathbf{a}^{(1)}, \mathbf{a}^{(2)}, \mathbf{a}^{(3)}$ の使い方は、一意に定まらない。 $x_1 = 0, x_2 = 1, x_3 = 0$ でもよいし、 $x_1 = 1, x_2 = -1, x_3 = 1$ でもよい。式 (8) のように、無数に解がある。

ランクとは？ 行列 \mathbf{A} から線形独立⁷な列ベクトルをできるだけ多く選び出したとき、その列ベクトルの本数を \mathbf{A} のランク (rank) と呼び、 $\text{rank } \mathbf{A}$ と記す⁸。 m 次元ベクトルなので、線形独立なベクトルは高々 m 本である。 $\text{rank } \mathbf{A} = \min(m, n)$ のとき⁹、 \mathbf{A} はフルランク (full rank) であるという。

線形連立方程式が一意の解を持つかどうかは行列 \mathbf{A} の性質次第

- もし、行列 \mathbf{A} が正則 (regular) ならば（逆行列を持つならば）、一意の解 $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$ が存在する。
- \mathbf{A} が正則である必要十分条件は、 \mathbf{A} がフルランクの正方行列であることである。すなわち、 $m = n = \text{rank } \mathbf{A}$ である。正則行列が表す基底は完備 (complete) であると呼ばれる。その理由は、式 (10) のように、どのようなベクトル $\mathbf{b} \in \mathbb{R}^m$ も行列 \mathbf{A} の $n = m$ 本の列ベクトルの線形結合で一意に表せるからである。
- 劣決定の連立方程式、つまり未知数より方程式が少ない場合 ($m < n$)、解は無数に存在する。 \mathbf{A} の n 本の列ベクトルの線形結合は、任意の m 次元ベクトル $\mathbf{b} \in \mathbb{R}^m$ を表現できるが、表現が無数に存在する。行列 \mathbf{A} は横長であり、次元数 m よりも基底ベクトルの本数 n が多い。 \mathbf{A} の n 本の列ベクトルから m 本をどのように選んでも完備な基底が作れるならば、そのような横長行列が表す基底は過完備 (overcomplete) であると呼ばれる。

⁷ 線形結合 $c_1 \mathbf{v}^{(1)} + c_2 \mathbf{v}^{(2)} + \dots + c_k \mathbf{v}^{(k)}$ について、 $c_1 = c_2 = \dots = c_k = 0$ とする以外にゼロベクトルが作れないならば、 k 本のベクトル $\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \dots, \mathbf{v}^{(k)}$ は線形独立であるという。線形独立な k 本のベクトルは、ちょうど打ち消しあうことができないので、どのベクトルもそのベクトルを使わないと作り出せない個性を持っているということ。

⁸ 階数とも呼ばれる

⁹ $\min(m, n)$ とは、 m と n のどちらか小さい方のことである。もし $m > n$ の場合は $\min(m, n) = n$ 本までしか列ベクトルを選べないので、ランクは $\min(m, n)$ となる。

- 優決定問題 (overdetermined problem) の連立方程式、つまり未知数より方程式が多い場合 ($m > n$) は、一般に「解なし」である¹⁰。 m 本の方程式から n 本選んで求めた解 \mathbf{x} は、選ばなかつた他の方程式を満たさないという矛盾が必ず生じる。 \mathbf{A} は m 次元の列ベクトルを n 本しか持たないので、任意の m 次元ベクトル \mathbf{b} を合成するためには基底が不十分である。

ベクトルのノルムとは？ ベクトル \mathbf{x} の大きさを表す非負の実数をノルム (norm) と呼び、 $\|\mathbf{x}\|$ と表す。原点 O から位置ベクトル \mathbf{x} の点までの距離を表すと考えてよい。しかし、ノルムの測り方は色々ある。最もよく用いられるノルムは ℓ_2 ノルムである。 n 次元ベクトル $\mathbf{x} = [x_1, x_2, \dots, x_n]^\top \in \mathbb{R}^n$ の ℓ_2 ノルムは $\|\mathbf{x}\|_2$ と書き表され、下記のように定義されている。

$$\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^\top \mathbf{x}} = \sqrt{x_1^2 + \dots + x_n^2} \quad (11)$$

これは、原点と位置ベクトル \mathbf{x} の点の間のユークリッド距離である。

ℓ_2 ノルム以外にもノルムの定義がある。例えば、 \mathbf{x} の ℓ_1 ノルムは

$$\|\mathbf{x}\|_1 = |x_1| + |x_2| + \dots + |x_n| \quad (12)$$

と定義される。 ℓ_1 ノルムで測った距離はマンハッタン距離 (Manhattan distance) とも呼ばれる。

正則化の手法

正則化 (regularization) とは、何らかの条件や情報・事前知識等に基づき、問題の解をひとつに限定することである。新たな方程式を追加することなく、劣決定の連立方程式をどのように正則化できるだろうか。

ℓ_2 最小長さ解 劣決定の連立方程式は、解 \mathbf{x} のノルムによって単純に正則化できる。「解 \mathbf{x} は原点 O に近い（ベクトル \mathbf{x} が短い）ほど良い解である」という事前知識が正しければ、これを使って最良の解を一意に特定できる。この解を最小長さ解 (minimum length solution) と呼ぶ。

n 次元空間の原点から点 $\mathbf{x} \in \mathbb{R}^n$ までのユークリッド距離は、式 (11) のように ℓ_2 ノルム $\|\mathbf{x}\|_2$ で測ることができる。劣決定の連立方程式の解のうち、原点からのユークリッド距離が最も小さい (ℓ_2 ノルムが最も小さい) 最小長さ解を求める問題は、次のような最小化問題として記述できる。

$$\min_{\mathbf{x}} \|\mathbf{x}\|_2^2 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b}. \quad (13)$$

この記述は、「 \mathbf{x} の ℓ_2 ノルムの 2 乗 $\|\mathbf{x}\|_2^2$ の最小値を求めよ。ただし、 \mathbf{x} は $\mathbf{A}\mathbf{x} = \mathbf{b}$ を満たすものとする。」という意味である。式 (13) の最小値から決まる一意の解 \mathbf{x}^* は、 \mathbf{A} と \mathbf{b} を使って書き表すことができる（演習問題 2）。

$$\mathbf{x}^* = \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^{-1} \mathbf{b} \quad (14)$$

ℓ_2 正則化 ℓ_2 最小長さ解と似ているが、少し異なる別の正則化の手法として、 ℓ_2 正則化を紹介しよう。 ℓ_2 正則化は、連立方程式から解 \mathbf{x} がずれている量（誤差）と、解の ℓ_2 ノルムの 2 乗 $\|\mathbf{x}\|_2^2$ を、一緒に最小化する。

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_2^2 \quad (15)$$

¹⁰縦長行列 \mathbf{A} が $\text{rank } \mathbf{A} = n$ ならば解なしである。 $\text{rank } \mathbf{A} < n$ ならば縦長行列であっても劣決定問題であり、解は無数に存在する。

この記述は、「 \mathbf{b} と \mathbf{Ax} のずれの大きさ $\|\mathbf{b} - \mathbf{Ax}\|_2^2$ と、 \mathbf{x} の長さを表す量 $\lambda\|\mathbf{x}\|_2^2$ を計算した。 \mathbf{x} を調節して、この和の最小値を求めよ。」という意味である。 ℓ_2 正則化は、統計学の分野でリッジ回帰（ridge regression）として知られている。式(15)から決まる一意の解 \mathbf{x}^* も、 \mathbf{A} , \mathbf{b} , λ を使って書き表すことができる（演習問題4）¹¹。

$$\mathbf{x}^* = (\mathbf{A}^\top \mathbf{A} + 2\lambda \mathbf{I})^{-1} \mathbf{A}^\top \mathbf{b} \quad (16)$$

式(15)の最小値から決まる解 \mathbf{x} は、もはや $\mathbf{Ax} = \mathbf{b}$ を厳密には満たさない。 \mathbf{b} と \mathbf{Ax} のずれの大きさ $\|\mathbf{b} - \mathbf{Ax}\|_2^2$ は損失関数（loss function）と呼ばれる。 ℓ_2 ノルムで測った誤差と考えてよい。正の定数 $\lambda \in \mathbb{R}_+$ は、損失関数と解の ℓ_2 ノルムのどちらも小さくなるようにバランスをとる役割があり、あらかじめ適切な値を設定しておく必要がある。設定した λ が小さいほど、損失関数が小さい（連立方程式を近似的に良く満たす）解が選ばれるようになるが、 ℓ_2 ノルムが小さい解が選ばれ難くなる（つまり事前知識が活かされない）。逆に、設定した λ が大きいほど、解の ℓ_2 ノルムが小さいという事前知識が重視されるが、損失関数があまり小さくない（連立方程式をあまり満たさない）解が選ばれるようになる。

無数の解の候補のうち、 ℓ_2 ノルムが小さいものを選ぶことは適切なのだろうか。連立方程式に帰着する何かの応用問題で、もし \mathbf{x} の成分が物理量（例えば電流や電圧、速度等）を表しているならば、式(11)のような2乗和はエネルギーに相当するので、 ℓ_2 最小長さ解はエネルギーが最小の解を選んでいると解釈できるかもしれない。しかし、式(12)の ℓ_1 ノルムのように、 ℓ_2 以外のノルムを用いた正則化も非常に重要であり、科学技術の様々な分野で次々と革新的な成果を生んでいる。次回はそのような正則化を紹介しよう。

演習問題

1. 式(7)の劣決定問題の ℓ_2 最小長さ解を求めよ。

ヒント：解は t を使って式(8)のように表される。 $\|\mathbf{x}\|_2^2$ は t の2次関数になる。これが最小となる t を求めよ。

- 2*. 劣決定問題 $\mathbf{Ax} = \mathbf{b}$ ($\mathbf{A} \in \mathbb{R}^{m \times n}$, $m < n$) について、 ℓ_2 最小長さ解

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_2^2 \quad \text{subject to} \quad \mathbf{Ax} = \mathbf{b} \quad (17)$$

は、 \mathbf{A} と \mathbf{b} を用いて式(14)のように陽に書き表せることを示せ。

ヒント：ラグランジュ乗数法（the Lagrange multiplier method）を適用する。最小長さ解は、下記のラグランジュ関数の停留点である。

$$L(\mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{2} \mathbf{x}^\top \mathbf{x} + \boldsymbol{\lambda}^\top (\mathbf{b} - \mathbf{Ax}).$$

3. 式(7)の劣決定問題について、 $\mathbf{x}^* = \mathbf{A}^\top (\mathbf{A} \mathbf{A}^\top)^{-1} \mathbf{b}$ を計算せよ。 \mathbf{x}^* は ℓ_2 最小長さ解である。

- 4*. 劣決定問題 $\mathbf{Ax} = \mathbf{b}$ ($\mathbf{A} \in \mathbb{R}^{m \times n}$, $m < n$) について、 ℓ_2 正則化問題の解

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \frac{1}{2} \|\mathbf{b} - \mathbf{Ax}\|_2^2 + \lambda \|\mathbf{x}\|_2^2 \quad (18)$$

は、 \mathbf{A} , \mathbf{b} , λ を用いて式(16)のように陽に書き表せることを示せ。

ヒント： \mathbf{x} のすべての成分について偏微分してゼロと置く。勾配（gradient），リッジ回帰またはチホノフ正則化（Tikhonov regularization）について調査せよ。

¹¹式(15)に $\frac{1}{2}$ が付いていなければ $\mathbf{x}^* = (\mathbf{A}^\top \mathbf{A} + \lambda \mathbf{I})^{-1} \mathbf{A}^\top \mathbf{b}$ が最小解である。

5.* 優決定問題 $\mathbf{A}\mathbf{x} = \mathbf{b}$ ($\mathbf{A} \in \mathbb{R}^{m \times n}$, $m > n$) の解は一般に存在しない. ℓ_2 ノルムで測った誤差 $\|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$ を最小にする \mathbf{x} を近似解とする手法は最小二乗法 (least squares method) と呼ばれる. 最小二乗解

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 \quad (19)$$

は, \mathbf{A} と \mathbf{b} を用いて

$$\mathbf{x}^* = (\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b} \quad (20)$$

のように陽に書き表せることを示せ.

★ 多すぎるのは良くないことだ — 劣決定問題のスパース解 —

劣決定問題 (underdetermined problem) が解ければ、例えばこんなことができるぞ。

<h3>Inpainting</h3> <p>Known pixel values $\mathbf{y} = \begin{bmatrix} \Phi & \Psi & \alpha \end{bmatrix}$</p> <p>Extraction of normal pixel values Wavelet</p> <p>Sparse wavelet coef.</p>	<h3>Deblurring / Superresolution</h3> <p>Low res. image $\mathbf{y} = \begin{bmatrix} \Phi & \Psi & \alpha \end{bmatrix}$</p> <p>Blurring & downsampling Wavelet</p> <p>Sparse wavelet coef.</p>										
<h3>Show-Through Cancellation</h3> <p>Front \mathbf{y}_f Back \mathbf{y}_b</p> <p>Recovered back</p> <p>Recovered front</p> <p>Show-through images $\mathbf{y}_f, \mathbf{y}_b = \begin{bmatrix} \mathbf{I} & \mathbf{B} & \Psi & \mathbf{O} & \alpha \\ \mathbf{B} & \mathbf{I} & \mathbf{O} & \Psi & \end{bmatrix}$</p> <p>Mixing & Blurring Wavelet</p> <p>Sparse wavelet coef.</p>	<h3>Signal Separation and Classification</h3> <p>Original signal Sparse Fourier signal Sparse wavelet signal</p> <p>Extraction and classification of lung sound components [Sakai et al., 11]</p> <p>(a) Degraded auscultation signal, (b) extracted breath sound, and (c) extracted fine crackles.</p> <p>signal $\mathbf{y} = \begin{bmatrix} \mathbf{F}^{-1} & \mathbf{W}^{-1} & \alpha \end{bmatrix}$</p> <p>components</p>										
<h3>Robust Face Recognition</h3> <p>[Wright et al., 09]</p> <p>Sparse representation of query data by training data</p> <p>$\mathbf{q}_{\text{(query)}} = \begin{bmatrix} \mathbf{S} & \alpha \end{bmatrix}$</p> <p>Training data (sparse vector)</p> <ul style="list-style-type: none"> Robust against noise and occlusion Efficient with low-dimensional image representation 	<h3>Multiple Object Recognition</h3> <p>[Sakai, 09]</p> <p>1. SIFT</p> <p>2. Bag-of-Keypoints representation</p> <p>3. Sparse decomposition</p> <p>Results</p> <table border="1"> <tr> <td>aeroplane, car</td> <td>bird, boat</td> <td>dog, person, sofa</td> <td>bus, car, person</td> <td>chair, person, sofa, tvmonitor</td> </tr> <tr> <td>aeroplane, car</td> <td>bird, boat</td> <td>person, cat, sofa</td> <td>bus, car, train, person</td> <td>tvmonitor, chair</td> </tr> </table> <p>Fig. 4. Multi-labeling results. First row: test images, second row: ground-truth labels, third row: labels by our method. The true positive labels are in bold.</p>	aeroplane, car	bird, boat	dog, person, sofa	bus, car, person	chair, person, sofa, tvmonitor	aeroplane, car	bird, boat	person, cat, sofa	bus, car, train, person	tvmonitor, chair
aeroplane, car	bird, boat	dog, person, sofa	bus, car, person	chair, person, sofa, tvmonitor							
aeroplane, car	bird, boat	person, cat, sofa	bus, car, train, person	tvmonitor, chair							

劣決定問題のスパース正則化

前回までのあらすじ

1. 線形連立方程式は、行列とベクトルを使って

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad (21)$$

と書き表せる。 $\mathbf{x} \in \mathbb{R}^n$ は n 個の未知数を成分に持つ n 次元ベクトルである。 $\mathbf{A} \in \mathbb{R}^{m \times n}$ は、 m 本の方程式における未知数の係数を要素に持つ $m \times n$ 行列である。 $\mathbf{b} \in \mathbb{R}^m$ は、 m 個の定数を成分に持つ m 次元ベクトルである。

2. 連立方程式 (21) は、行列 \mathbf{A} の n 本の列ベクトル $\mathbf{a}^{(j)}$ ($j = 1, \dots, n$) を基底に使って \mathbf{b} を合成するための線形結合係数を求める問題であると解釈できる。なぜならば、

$$\mathbf{b} = \mathbf{A}\mathbf{x} = \begin{bmatrix} \mathbf{a}^{(1)} & \dots & \mathbf{a}^{(n)} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = x_1\mathbf{a}^{(1)} + \dots + x_n\mathbf{a}^{(n)} \quad (22)$$

と表せるからである。

3. 未知数の数よりも方程式の数が少ない連立方程式は、無数の解を持ち、解をひとつに決められない劣決定問題である。方程式の他に、解に関する情報や事前知識があるとき、それを使って解をひとつに限定することを正則化と呼ぶ。

方程式が足りなくても、正則化で正解が得られるならば、 n 個の未知数について $m = n$ 本の方程式を用意するのは無駄ではなかろうか。どのような方程式の解について事前知識がある場合に、方程式が足りなくても正解が得されることを保証できるだろうか。

スパース解 「未知数はゼロになるものが多い」という事前知識は、非常に重要な正則化になる。無数に存在する解のうち、非ゼロとなる未知数の数が最も少ない（ゼロになる未知数がの数が最も多い）解を求める問題は、次のような最小化問題として定式化される。

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \quad (23)$$

この記述は、「ベクトル \mathbf{x} の成分のうち、ゼロでないものの個数 $\|\mathbf{x}\|_0$ の最小値を求めよ。ただし、 \mathbf{x} は $\mathbf{A}\mathbf{x} = \mathbf{b}$ を満たすものとする。」という意味である。 $\|\mathbf{x}\|_0$ は、 \mathbf{x} の ℓ_0 ノルムと呼ばれ、 \mathbf{x} の非ゼロ成分の個数を表す¹²。すなわち、 $\|\mathbf{x}\|_0 = |\{i | x_i \neq 0\}|$ （ゼロでない成分の集合の要素の数）と定義する。

式 (23) の解をスパース解 (sparse solution) と呼ぶ。また、ベクトルの非ゼロ成分の数が k 個以下の場合、そのベクトルは k スパース (k -sparse) であると言う。式 (22) のように、 \mathbf{x} は線形結合係数を成分に持つベクトルであるから、スパース解の非ゼロ成分に対応する \mathbf{A} の列ベクトルのみによってベクトル \mathbf{b} が簡潔に表現される。よって、式 (23) の ℓ_0 最小化問題は、 \mathbf{A} の列から最少の本数で \mathbf{b}

¹²これは便宜上の記述である（演習問題 2）。正確には下記のように記述すべきである。

$$\min_{\mathbf{x}} \lim_{p \rightarrow 0} \|\mathbf{x}\|_p^p \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b}$$

を合成できる組合せを探すという NP 困難な「組合せ最適化問題」である。また、式(23)の解は必ずしも一意ではない。

数学では、関数 $f(x)$ の値がゼロにならないような x の集合を、その関数の台 (support) と呼ぶ。同様に、ベクトルの非ゼロ成分の番号 (添え字) の集合を、ベクトルの台と呼ぶ。1から n までの整数の集合を $[n] = \{1, \dots, n\}$ と表すならば、 n 次元ベクトルの台 S は集合 $[n]$ の部分集合である ($S \subset [n]$)。また、非ゼロ成分の個数はこの集合の要素の数 $|S|$ である。

例 $\mathbf{x} = [0, 2, 0, 0, 1]^\top$ の台は $S = \{2, 5\}$ である ($x_2 \neq 0, x_5 \neq 0$)。もちろん、 $S \subset [5] = \{1, 2, 3, 4, 5\}$ が成り立つ。

本書では、 n 本の列を持つ行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ について、番号の集合を $T \subset [n]$ とするとき、 T で指定した \mathbf{A} の列を抜き出して作った $m \times |T|$ 行列を \mathbf{A}_T と記すことにする。同様に、 T で指定した番号の成分を \mathbf{x} から抜き出して作った $|T|$ 次元ベクトルを \mathbf{x}_T と記す。

例 $\mathbf{A} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix}$, $\mathbf{x} = [0, 1, 2]^\top$, $T = \{1, 3\}$ のとき, $\mathbf{A}_T = \begin{bmatrix} 1 & 1 \\ 1 & 3 \end{bmatrix}$, $\mathbf{x}_T = \begin{bmatrix} 0 \\ 2 \end{bmatrix}$ である。

スパース解が一意になる条件

スパース解を求める方法を考える前に、式(23)のスパース解が複数存在せず一意に定まる条件は何なのか知っておきたい。式(23)の k スパース解の一意性は、 \mathbf{A} の列ベクトルの組合せの線形独立性に依存している。

単純な例で考察しよう 次のような 4×5 の行列で考えてみよう。

$$\mathbf{A}^n = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{A}^u = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}. \quad (24)$$

これらはフルランクである ($\text{rank } \mathbf{A}^n = 4$, $\text{rank } \mathbf{A}^u = 4$)。行列の列の組合せを見てみよう。4本の列を選ぶとする。行列 \mathbf{A}^n には、線形独立にならない4本の列の組合せが存在する。第1, 2, 3, 5列という組合せである（第1, 2, 5列を足し合わせると第3列が作れてしまう）。一方、 \mathbf{A}^u は、どの4本の列を組合せても必ず線形独立になる。

まず、 $\mathbf{A} = \mathbf{A}^n$, $\mathbf{b} = [1, 1, 1, 0]^\top$ として、式(23)のスパース解を見つけてみよう。解 \mathbf{x} は、式(22)のように \mathbf{b} を合成するための線形結合係数なので、スパース解の台は合成に使用する \mathbf{A}^n の列に対応する。 \mathbf{A}^n の $n = 5$ 本の列から線形独立な $m = 4$ 本のベクトルを選べば、どんな4次元ベクトルも合成できる。 ${}_5C_4 = 5$ 通りの列の組合せのうち、線形独立な列ベクトルの組合せは4通りある。 $T = \{1, 2, 3, 4\}$, $\{1, 2, 4, 5\}$, $\{1, 3, 4, 5\}$, $\{2, 3, 4, 5\}$ である。これら4通りの組合せから4通りの解が得られる¹³。

$$\mathbf{x} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ -1 \end{bmatrix} \quad (25)$$

¹³ 例えば、 $T = \{2, 3, 4, 5\}$ ならば、 \mathbf{A}^n の第1列を使用しないから $x_1 = 0$ である。 $\mathbf{A}_T^n \mathbf{x}_T = \mathbf{b}$ の方程式から $\mathbf{x}_T = [x_2, x_3, x_4, x_5]^\top = [0, 1, 0, -1]^\top$ を得る。

この結果から、解はスパースであるが、2種類存在することがわかる。ゆえに、 \mathbf{A}^n は \mathbf{b} に対して一意のスパース解を持たない。 \mathbf{b} を簡潔に合成するためには、 \mathbf{A}^n の第1列と第2列の組合せでもよいし、第3列と第5列の組合せでもよいということを2種類のスパース解が示している。つまり、 \mathbf{A}^n の過完備基底のように、互いに代用可能な基底ベクトルの組合せが存在するとき、解が一意にならない。

次に、 $\mathbf{A} = \mathbf{A}^u$, $\mathbf{b} = [1, 1, 1, 0]^\top$ として、同様にスパース解を見つけてみよう。4本選ぶ列の組合せは ${}_5C_4 = 5$ 通りあるが、どれも線形独立になる。 $T = \{1, 2, 3, 4\}, \{1, 2, 3, 5\}, \{1, 2, 4, 5\}, \{1, 3, 4, 5\}, \{2, 3, 4, 5\}$ の5通りの組合せから、それぞれ次のような解が得られる。

$$\mathbf{x} = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0.5 \\ 0 \\ 0.5 \\ 0.5 \\ -0.5 \end{bmatrix}, \text{ and } \begin{bmatrix} 0 \\ -1 \\ 1 \\ 1 \\ -1 \end{bmatrix} \quad (26)$$

ゆえに、 \mathbf{A}^u は \mathbf{b} に対して一意の $k = 2$ スパース解 $\mathbf{x} = [1, 1, 0, 0, 0]^\top$ を持つ。 \mathbf{A}^u の第1列と第2列の組合せだけが \mathbf{b} を簡潔に合成できる。 \mathbf{A}^u の過完備基底は、互いに代用可能な基底ベクトルの組合せが存在しないので、解が一意になる。

スパース解が一意になるための十分条件

スパーク

定義 1. 行列 \mathbf{A} から、線形独立にならない列ベクトルとなるべく少なく選び出したとき、その列ベクトルの本数を \mathbf{A} のスパーク (spark) と呼び、 $\text{spark } \mathbf{A}$ と記す。

$$\text{spark } \mathbf{A} = \min_{\mathbf{x} \neq \mathbf{0}} \|\mathbf{x}\|_0 \quad \text{subject to } \mathbf{Ax} = \mathbf{0}. \quad (27)$$

定理 1

$\text{spark } \mathbf{A} > 2k$ ならば、式 (23) の k スパース解は一意である。

この証明は演習問題 4 とする。

スパークはランクに似ているが混同しないように注意されたい。例えば、式 (24) の行列 \mathbf{A}^n と \mathbf{A}^u は、ランクがどちらも $\text{rank } \mathbf{A}^n = \text{rank } \mathbf{A}^u = 4$ であるが、スパークは $\text{spark } \mathbf{A}^n = 4$, $\text{spark } \mathbf{A}^u = 5$ である。ただし、スパークを求める問題は、線形独立にならない列の組合せを探す組合せ問題である（演習問題 3）。定理 1 から、 \mathbf{A}^u の $k = 2$ スパース解は一意であると言える。

制限等長性

定義 2. 任意の k スパースなベクトル \mathbf{x} について不等式

$$(1 - \delta) \|\mathbf{x}\|_2^2 \leq \|\mathbf{Ax}\|_2^2 \leq (1 + \delta) \|\mathbf{x}\|_2^2. \quad (28)$$

を満たす定数 $\delta < 1$ が存在するとき、行列 $\mathbf{A} \in \mathbb{R}^{m \times n}$ は k 次の制限等長性 (restricted isometry property; RIP) を持つという。この性質を (δ, k) -RIP と表す。等価な定義として、 k 個以下の要素を持つ任意の部分集合 $S \subset [n]$ ($|S| \leq k$) について不等式

$$\|\mathbf{A}_S^\top \mathbf{A}_S - \mathbf{I}_k\|_2 \leq \delta \quad (29)$$

が成り立つとき、行列 \mathbf{A} は (δ, k) -RIP の制限等長性を持つという。ここで、 \mathbf{I}_k は $k \times k$ の単位行列である。

言い換えると、 (δ, k) -RIP 条件を満たす行列 \mathbf{A} による線形写像は、 k スパースな任意のベクトルのノルムを近似的に維持する。この性質は、ランダム射影のそれと同様である。実際、 k スパースなベクトルの幾何（ノルムや距離）を維持するように JL の補題 [1] を適用すると、 $m \times n$ のランダム行列 \mathbf{A} について

$$m > m_0 \approx \frac{1}{\delta^2} k \log \frac{n}{k} \quad (30)$$

が成立するならば、 \mathbf{A} は (δ, k) -RIP 条件を満たす（演習問題 5）。また、もし \mathbf{A} が $(\delta, 2k)$ -RIP 条件を満たすならば、式 (23) の k スパース解が一意であることを簡単に確認できる（演習問題 6）。

更に、下記の定理は、凸最適化問題を解くことで一意のスパース解が得られることを保証している。

定理 2 ([2, 3])

\mathbf{A} が $\delta < \sqrt{2} - 1$ の $(\delta, 2k)$ -RIP 条件を満たすならば、式 (23) の k スパース解は一意であり、下記の ℓ_1 最小化問題の解と一致する。

$$\min \|\mathbf{x}\|_1 \quad \text{subject to} \quad \mathbf{Ax} = \mathbf{b} \quad (31)$$

この定理 2において、 $\delta < \sqrt{2} - 1$ の $(\delta, 2k)$ -RIP は、スパース解が一意かつ ℓ_1 最小化の解と一致するための十分条件である。要するに、どんな $2k$ スパースなベクトルに行列 \mathbf{A} を乗じても、ノルムが高々 $1 \pm \delta$ 倍しか伸縮しないという性質を \mathbf{A} が持っているならば、 k スパース解 \mathbf{x} は一意で、しかも ℓ_1 最小化問題の解とも一致する。 ℓ_1 最小化は解が必ず一意になる凸問題であり、最小化のアルゴリズムも充実している。

ただし、定理 2 の RIP 条件を確認する計算も組合せの手間数が必要で現実的ではない。 $|S| \leq 2k$ のあらゆる台について $\mathbf{A}_S^\top \mathbf{A}_S$ の最小固有値と最大固有値が $1 \pm \delta$ 以内に収まることを確認する必要があるからである。台が S のベクトル \mathbf{x} について $\|\mathbf{Ax}\|_2^2 = \mathbf{x}_S^\top \mathbf{A}_S^\top \mathbf{A}_S \mathbf{x}_S$ と書けるので、 \mathbf{A} が \mathbf{x} を何倍に収縮させるかは $\mathbf{A}_S^\top \mathbf{A}_S$ の最小固有値と最大固有値で表される。

まとめ

線形方程式の数が未知数の数より少なくて、解がスパースならば一意な解が存在する。方程式 $\mathbf{Ax} = \mathbf{b}$ は、与えられたベクトル \mathbf{b} を行列 \mathbf{A} の列ベクトルの線形結合で合成するための線形結合係数を求める問題である。解がスパースであるということは、 \mathbf{A} の列を数本組合せるだけで \mathbf{b} を合成できるという意味である。スパース解 \mathbf{x} の非ゼロ成分は、 \mathbf{b} の合成に使用する \mathbf{A} の列の組合せを表している。スパース解が一意である（その他にスパース解がない）ということは、 \mathbf{A} の数本の列の組合せで \mathbf{b} を線形合成できたときに、それ以下の本数では \mathbf{b} を合成できる組合せが他に見つからないということである。つまり、方程式の係数を表す行列 \mathbf{A} の列ベクトルが互いに線形結合で表し難い性質があるならば、ある列の組合せで \mathbf{b} を合成できたときに、その代用になる列の組合せが存在せず、解が一意になり易いと言える。この性質を定量的に表しているのが spark \mathbf{A} や制限等長性 (δ, k) -RIP である。特に、 \mathbf{A} がランダム行列の場合はスパース解が一意になり易い。また、証明の解説は省略したが、定理 2 は、解の一意性だけでなく、 ℓ_0 最小解と ℓ_1 最小解の一致性も保証している。

定理 1 と定理 2 は、非ゼロ成分の数よりも無闇にたくさんの方程式は無くとも解が求まる事を示している。ベクトル \mathbf{b} の各成分は、 \mathbf{A} の行ベクトルと \mathbf{x} の内積によって \mathbf{x} の特徴を測った観測値であると見なせる。方程式の本数が観測値の数であるから、解 \mathbf{x} を得るために無闇に何度も \mathbf{x} を観測しなくともよいとも言える。この性質を活かすために、更に次の点について学ぶべきであろう。

- \mathbf{A} と \mathbf{b} からスパース解 \mathbf{x} を効率的に求めるアルゴリズムは？
- \mathbf{b} をスパース表現できる \mathbf{A} とは？

演習問題

1. 式(24)の \mathbf{A}^u について, $T = \{1, 2, 4, 5\}$ のとき, \mathbf{A}_T^u を書き表せ. また, $\mathbf{b} = [1, 1, 1, 0]^\top$ のとき, 方程式 $\mathbf{A}_T^u \mathbf{x}_T = \mathbf{b}$ の解が $\mathbf{x}_T = [1, 1, 0, 0]^\top$ であることを示せ. この解は, 式(26)の3番目の解である. MATLABを使って, 他の解も同様に確認せよ.

ヒント: 次のようなMATLABコードを試してみよ.

```
A=[1 0 1 0 0; 0 1 2 0 0; 0 1 1 1 0; 0 0 1 0 1];
C=combnk(1:5,4)
T=C(3,:)
A(:,T)
help inv
```

2. $\|\mathbf{x}\|_p^p$ (ℓ_p ノルムの p 乗) は, $p \in \mathbb{R}_+$ がゼロに近づくと, \mathbf{x} の非ゼロ成分数に収束することを示せ. すなわち, 次式を証明せよ.

$$\lim_{p \rightarrow 0+} \|\mathbf{x}\|_p^p = \lim_{p \rightarrow 0+} \sum_{i=1}^n |x_i|^p = \#\{i | x_i \neq 0\} \quad (32)$$

なお, 下記のように収束の様子をMATLABで観察できる.

```
% x の 1p ノルムを計算する関数を定義する.
lpnormp=@(x,p) sum(abs(x).^p);
% 適当なベクトルについて, 非ゼロ成分数へ収束するかどうか値を見る.
lpnormp([1 2 0 3 0 0 4], 1)
lpnormp([1 2 0 3 0 0 4], 0.5)
lpnormp([1 2 0 3 0 0 4], 0.1)
lpnormp([1 2 0 3 0 0 4], 0.01)
```

3. spark \mathbf{A} を求める問題は組合せ問題であることを説明せよ.

ヒント: sparkを求めるMATLAB関数を下記に示す. 関数として使うには, これをファイル名 spark.mとして保存せよ.

```
function sp = spark(A)

[m, n] = size(A);
% これを大きなサイズの行列に使わないこと.
if m > 16, sp = -1; return; end

for sp = 2:m,
    C = combnk(1:n, sp);
    nc = size(C, 1);
    for i = 1:nc,
```

```

if rank(A(:,C(i,:))) < sp,
    return;
end
end
sp = m + 1;
end

```

4. 定理 1 を証明せよ.

ヒント: 対偶を証明せよ. $\mathbf{A}\mathbf{x} = \mathbf{b}$ について 2 つの異なる k スパース解があると仮定し, $\text{spark } \mathbf{A} \leq 2k$ を示せ.

5.* 式 (30) を導出せよ.

ヒント: スターリングの近似 (Stirling's approximation) から ${}_n C_k \leq (en/k)^k$ が成り立つ.

6. \mathbf{A} が $(\delta, 2k)$ -RIP ならば, 式 (23) の k スパース解は一意であることを証明せよ.

ヒント: 背理法で証明できる. ふたつの異なる k スパース解 \mathbf{x}_1 と \mathbf{x}_2 を仮定し, $2k$ スパースのベクトル $(\mathbf{x}_1 - \mathbf{x}_2)$ に RIP 条件を適用せよ.

7.* 式 (24) の \mathbf{A}^u は $k = 2$ の $(\delta, 2k)$ -RIP 条件を満たすことを確認せよ. また, δ の最小値を数値的に求めよ.

参考文献

- [1] W. Johnson and J. Lindenstrauss, "Extensions of Lipschitz maps into a Hilbert space," Contemporary Mathematics, vol.26, pp.189–206, 1984.
- [2] E.J. Candès, J. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," Comm. on Pure and Applied Math, vol.59, no.8, pp.1207–1223, 2006.
- [3] E.J. Candès, "The restricted isometry property and its implications for compressed sensing," Comptes Rendus Mathematique, vol.346, no.9-10, pp.589–592, May 2008.

★ スパース解法 —貪欲法 —

次のような ℓ_0 最小化問題の解き方を考えよう.

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{Ax} = \mathbf{b}. \quad (33)$$

貪欲法¹⁴ (greedy algorithm) とは何か? おつりを作る作業を思い出そう. なるべく少ない枚数のコインで 623 円をどのように作るか? 内輪で残額に最も近い額のコインを選ぶことを繰り返せばよい¹⁵.

線形結合 $\mathbf{b} = x_1 \mathbf{a}^{(1)} + \cdots + x_n \mathbf{a}^{(n)}$ を思い出そう. 基底ベクトル $\mathbf{a}^{(j)}$ はコインである. 金額 \mathbf{b} を作るために使うコイン $\mathbf{a}^{(j)}$ の係数 x_j だけ非ゼロにする. 選んだコインを表す非ゼロ係数の番号 (添え字) の集合を T とする. なるべく少ない枚数のコインで金額 \mathbf{b} を作ろう. 残額 $\mathbf{r} = \mathbf{b} - \mathbf{Ax}$ に最も似ているコイン $\mathbf{a}^{(s)}$ を選び, 残額が最も減るように非ゼロ成分の $\mathbf{x}_{T \cup \{s\}}$ を最小二乗法で更新すればよい.

OMP

OMP (orthogonal matching pursuit) [1, 2, 3] は, アルゴリズム 1 のようにスパース解を求める貪欲法である.

アルゴリズム 1 Orthogonal matching pursuit: $\mathbf{x} = \text{OMP}(\mathbf{b}, \mathbf{A}, k, \delta)$

入力: \mathbf{b} : m 次元ベクトル, \mathbf{A} : $m \times n$ 行列, k : 非ゼロ成分の最大個数, δ : 許容誤差;

出力: \mathbf{x} : k スパースな n 次元ベクトル;

- 1 $\mathbf{x} := \mathbf{0} \in \mathbb{R}^n$;
- 2 台の初期値を $T := \emptyset$ (空集合) とする;
- 3 $\mathbf{r} := \mathbf{b}$;
- 4 **while** $|T| \leq k$ かつ $\|\mathbf{r}\|_2 / \|\mathbf{b}\|_2 > \delta$ **do**
- 5 各基底と残差の類似度 $\mathbf{c} := \mathbf{A}^\top \mathbf{r}$ を計算する;
- 6 最も類似している列の番号 $s := \arg \max_j |c_j|$ を見つける;
- 7 $T := T \cup \{s\}$ として s を台に加える;
- 8 台 T で示された非ゼロ成分を $\mathbf{x}_T := \arg \min_{\mathbf{z} \in \mathbb{R}^{|T|}} \|\mathbf{b} - \mathbf{A}_T \mathbf{z}\|_2^2$ のように最小二乗解で更新する.
- 9 残差 $\mathbf{r} := \mathbf{b} - \mathbf{Ax}$ を計算する;
- 10 **end while**

貪欲法は, ゼロベクトルを解 \mathbf{x} の初期値とし, 残差が十分に小さくなる (\mathbf{Ax} が十分に \mathbf{b} の近似になる) まで \mathbf{x} の非ゼロ成分を次々と見つけるアルゴリズムになっている. よって, 非ゼロ成分の個数程度の反復回数で解が得られる. 各反復において最も計算量が大きい手続きは, ステップ 5 の類似度計算 ($\mathcal{O}(mn)$) と, ステップ 8 の最小二乗解 \mathbf{x}_T の計算である¹⁶. 最小二乗法の手間数は $\mathcal{O}(m|T|^2)$ なので, $k^2 \ll n$ ならば OMP の手間数は $\mathcal{O}(mk^3)$ と見積もられる.

¹⁴欲張り法ともいう.

¹⁵用意されているコインが 1, 5, 10, 50, 100, 500 円玉ならば必ず最適解が得られる. しかし, 貪欲法はどんなコインの組合せ問題も解けるわけではない. 例えば, もし 1, 6, 13 円玉の 3 種類だとしたら, 18 円を 3 枚の 6 円玉で作れるのに, 貪欲法では 1 枚の 13 円玉と 5 枚の 1 円玉の計 6 枚が選ばれる.

¹⁶この最小二乗解は $\mathbf{x}_T = (\mathbf{A}_T^\top \mathbf{A}_T)^{-1} \mathbf{A}_T \mathbf{b}$ と書ける. これを右から順に直接に計算する方法の他に, \mathbf{A}_T の特異値分解を利用する方法, 連立方程式 $\mathbf{A}_T^\top \mathbf{A}_T \mathbf{x}_T = \mathbf{A}_T \mathbf{b}$ を共役勾配法で解く方法などがある.

演習問題

1. アルゴリズム 1 の OMP を MATLAB 関数として実装せよ.

```
function x = OMP(b, A, k, delta)
```

この行から書き始め, `OMP.m` として保存せよ.

ヒント : MATLAB では, ステップ 6, 7, 8 は次のように書ける.

```
[maxc, s] = max([abs(c)]);
T = [T, s];
x(T) = A(:,T) \ b;
```

2. A , x , b を以下のように生成し, x を未知として式 (33) を OMP で解け.

- A をサイズ $m \times n$ のランダム行列とする. 要素は正規分布から生成する.
- x を k スパースなベクトルとし, 非ゼロ成分は正規分布から生成する.
- $b = Ax$ とする.

次に, 非ゼロ成分の個数 k に対する OMP の解の誤差を観察せよ. すなわち, 横軸を k , 縦軸を相対残差 $\|b - Ax\|_2/\|b\|_2$ としてグラフを描け.

ヒント : 以下のコードを試し, OMP で解が得られるかどうか確認せよ.

```
m = 100; n = 1000; A = randn(m,n);
k = 10; S = randperm(n); S = S(1:k);
x = zeros(n,1); x(S) = randn(k,1);
b = A * x;
x_est = OMP(b, A, k, 1e-7);
figure, plot(1:n, x, '.-', 1:n, x_est, 'o');
```

このコードを改造して相対残差を観察せよ. 誤差を統計的に評価するため, 反復試行する必要がある.

3.* スパース表現による顔認識 [?]

J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust face recognition via sparse representation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 2, pp. 210–227, 2009.

について調査し, 下記の課題に取り組むこと.

- “Sparse Representation-based Classification (SRC)” と呼ばれる手法の概要を述べよ.
- [The Extended Yale Face Database B](#)
(<http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html>) をランダム射影したものを用いて, SRC による顔認識を実行せよ. `YaleB_Ext_Cropped_192x168_all.mat` は, 顔を切り出した画像の画素値を並べた行列を MATLAB ファイルにしたものである.

- (a) ランダム射影後の次元数, (b) ノイズの強さに関して顔認識の正答率を評価せよ.

ヒント：SRCによる顔認識を実行するコードを以下に示す.

```
% Yale B cropped face images を読み込む.
data = load('YaleB_Ext_Cropped_192x168_all.mat');
[n, m] = size(datafea);

% 訓練データとテストデータに分ける.
ntest = round(n/2); ntrain = n - ntest;
jtest = randperm(n, ntest); jtrain = setdiff(1:n, jtest);
idtest = data.gnd(jtest); idtrain = data.gnd(jtrain);
idmax = max(idtrain);

% データ行列
D = double(datafea(:, :)' );

% ランダム射影(192*168次元から 200次元へ)
D = randn(200, 192*168) * D;

% 列ベクトルが正規化された訓練データ行列
A = D(:, jtrain);
A = A * spdiags(1./sqrt(sum(A.^2))', 0, ntrain, ntrain);

%for j = 1:ntest,
j = 700;
    b = D(:, jtest(j)); % + 100*rand(size(y)); % ノイズを印加する場合
    fprintf('testing id = %d\n', idtest(j));
    x = OMP(b, A, 100, 1e-1);
    fprintf('nnz = %d\n', nnz(x));

% 残差の計算
residual = zeros(idmax, 1);
for id = 1:idmax,
    b_reconst = A(:, idtrain == id) * x(idtrain == id);
    residual(id) = norm(b - b_reconst);
end
figure(11), bar(residual);

% 識別
[~, id_est] = min(residual);
fprintf('estimated id = %d\n', id_est);
%end
```

参考文献

- [1] Y.C. Pati, R. Rezaifar, Y.C.P.R. Rezaifar, and P.S. Krishnaprasad, “Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition,” Proceedings of the 27 th Annual Asilomar Conference on Signals, Systems, and Computers, pp.40–44, 1993.
- [2] J.A. Tropp, “Greed is good: Algorithmic results for sparse approximation,” IEEE Trans. Information Theory, vol.50, no.10, pp.2231–2242, 2004.
- [3] J.A. Tropp, Anna, and C. Gilbert, “Signal recovery from random measurements via orthogonal matching pursuit,” IEEE Trans. Information Theory, vol.53, no.12, pp.4655–4666, 2007.

★ スパース解法 —凸緩和—

次のような ℓ_0 最小化問題の解き方を考えよう.

$$\min_{\mathbf{x}} \|\mathbf{x}\|_0 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b}. \quad (34)$$

スパース解を持つ凸問題

貪欲法とは別の、もうひとつの代表的なスパース解法は、 ℓ_1 ノルムを用いた凸緩和である。式(34)の ℓ_0 ノルムを ℓ_1 ノルムに置き換えた最小化問題

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{subject to} \quad \mathbf{A}\mathbf{x} = \mathbf{b} \quad (35)$$

は基底追跡 (basis pursuit; BP) と呼ばれる [1]。 \mathbf{A} が $\delta < \sqrt{2} - 1$ の $(\delta, 2k)$ -RIP 条件を満たすならば、式(35)は式(34)と同一のスパース解を持つことが証明されている [2, 3]。 ℓ_p ノルムは、 $p \geq 1$ のとき凸関数 (convex function) ¹⁷ であり、最小化問題の解が必ず一意になる。

BP は線形計画法 (linear programming; LP) に書き直すことができる。

$$\|\mathbf{x}\|_1 = \min_{\mathbf{x}^+, \mathbf{x}^-} \mathbf{1}^\top (\mathbf{x}^+ + \mathbf{x}^-) \quad \text{subject to} \quad \mathbf{x} = \mathbf{x}^+ - \mathbf{x}^-, \quad \mathbf{x}^+ \geq \mathbf{0}, \quad \mathbf{x}^- \geq \mathbf{0}$$

また、BP は次式の基底追跡ノイズ除去 (basis pursuit denoising; BPDN) [1] の特別な場合である。

$$\min_{\mathbf{x}} \|\mathbf{x}\|_1 \quad \text{subject to} \quad \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 \leq \delta \quad (36)$$

ただし、 $\delta \geq 0$ は定数である。

次式のように ℓ_1 ノルムに上界を設けた残差の ℓ_2 最小化問題は LASSO (least absolute shrinkage and selection operator) と呼ばれる [4]。

$$\min_{\mathbf{x}} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2 \quad \text{subject to} \quad \|\mathbf{x}\|_1 \leq \tau \quad (37)$$

ただし、 $\tau > 0$ は定数である。

式(36)と(37)の最小化問題は、 ℓ_1 正則化最小二乗問題 (ℓ_1 -LS)

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2^2 + \lambda \|\mathbf{x}\|_1 \quad (38)$$

と等価である。すなわち、ある $\delta > 0$ に対する式(36)の解が式(38)の解と一致するような定数 $\lambda > 0$ が存在する。同様に、ある $\tau \geq 0$ に対する式(37)の解が式(38)の解と一致するような定数 $\lambda \geq 0$ が存在する [5]¹⁸。なお、式(38)は LASSO 回帰 (LASSO regression) とも呼ばれる。

ℓ_2 最小長さ解や ℓ_2 正則化問題の解 \mathbf{x}^* は、与えられた行列 \mathbf{A} や \mathbf{b} を用いて陽に書き表すことができた。これらの ℓ_2 最小化問題はそれぞれ式(35)の BP や式(38)の ℓ_1 -LS に非常に良く似ている。しかし、 ℓ_1 ノルムは ℓ_2 ノルムのように微分できないので、式(35)の BP や式(36)の BPDN、または式(37)の LASSO や式(38)の ℓ_1 -LS のような最小化問題の解は、もはや解を陽に導出できない。そこで、 ℓ_1 ノルムが関わる凸最適化問題の解を数値的に求める方法が研究されてきた（例えば [6, 4, 1] [5, 7, 8, 9] など）。

¹⁷集合 $C \subset \mathbb{R}^n$ について、任意の 2 点 $\mathbf{x}_1, \mathbf{x}_2 \in C$ を結ぶ線分上の点を $\mathbf{x}_\alpha = \alpha\mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2$ ($0 \leq \alpha \leq 1$) とする。必ず $\mathbf{x}_\alpha \in C$ ならば、 C は凸集合 (convex set) である。さらに、 $f(\mathbf{x}_\alpha) \leq \alpha f(\mathbf{x}_1) + (1 - \alpha)f(\mathbf{x}_2)$ が成り立つならば、 f は凸関数である。

¹⁸ただし、 δ や τ に対応する λ を求める一般的な方法はない。

最適化アルゴリズム

閾値処理に基づく手法

様々な解法があるが、代表的なものをいくつか紹介しよう。まず、式(38)で $\mathbf{A} = \mathbf{I}$ の場合は、単純であるが理解のため重要である。なぜならば、これは次のような成分毎の最小化問題になるからである。

$$\min_{x_j} \frac{1}{2} (b_j - x_j)^2 + \lambda |x_j| \quad (j = 1, \dots, n) \quad (39)$$

この最小値を与える解は、次のように陽に書ける（演習問題1）。

$$x_j^* = \begin{cases} b_j - \lambda & (0 < \lambda < b_j) \\ 0 & (-\lambda \leq b_j \leq \lambda) \\ b_j + \lambda & (b_j < -\lambda < 0) \end{cases} \quad (40)$$

要するに、 \mathbf{b} の各成分を原点Oに向けて λ だけ縮めると解 \mathbf{x}^* が得られる。 $|b_j|$ が λ より小さい成分はゼロになるので、解はスパースである。式(40)の処理はソフト閾値処理（soft thresholding/shrinkage）[6]と呼ばれ、 $\mathbf{x}^* = \text{soft}(\mathbf{b}, \lambda)$ と書く。

式(38)で $\mathbf{A} \neq \mathbf{I}$ の場合は、少々強引ではあるが $\mathbf{A}^\top \mathbf{A} \approx L\mathbf{I}$ (L はLipschitz定数)と近似すると、定点 $\mathbf{x}^{(k)}$ の近傍で

$$\frac{1}{2} \|\mathbf{b} - \mathbf{Ax}\|_2^2 \approx \frac{1}{2} \|\mathbf{b} - \mathbf{Ax}^{(k)}\|_2^2 + (\mathbf{x} - \mathbf{x}^{(k)})^\top \mathbf{A}^\top (\mathbf{Ax}^{(k)} - \mathbf{b}) + \frac{L}{2} \|\mathbf{x} - \mathbf{x}^{(k)}\|_2^2$$

と近似できるので、これに $\lambda \|\mathbf{x}\|_1$ を付加した関数

$$F(\mathbf{x}, \mathbf{x}^{(k)}) := \frac{1}{2} \|\mathbf{b} - \mathbf{Ax}^{(k)}\|_2^2 + (\mathbf{x} - \mathbf{x}^{(k)})^\top \mathbf{A}^\top (\mathbf{Ax}^{(k)} - \mathbf{b}) + \frac{L}{2} \|\mathbf{x} - \mathbf{x}^{(k)}\|_2^2 + \lambda \|\mathbf{x}\|_1$$

を \mathbf{x} について最小化すると、次式の反復式を得る（演習問題2）。

$$\begin{aligned} \mathbf{x}^{(k+1)} &:= \arg \min_{\mathbf{x}} F(\mathbf{x}, \mathbf{x}^{(k)}) = \arg \min_{\mathbf{x}} \left(\frac{1}{2} \|\mathbf{x}^{(k)} + \frac{1}{L} \mathbf{A}^\top (\mathbf{b} - \mathbf{Ax}^{(k)}) - \mathbf{x}\|_2^2 + \frac{\lambda}{L} \|\mathbf{x}\|_1 \right) \quad (41) \\ &= \text{soft} \left(\mathbf{x}^{(k)} + \frac{1}{L} \mathbf{A}^\top (\mathbf{b} - \mathbf{Ax}^{(k)}), \frac{\lambda}{L} \right) \end{aligned}$$

これはIST（iterative soft shrinkage）[10]と呼ばれるアルゴリズムである。初期値は任意でよいが、 $\mathbf{x}^{(0)} = \mathbf{A}^\top \mathbf{b}$ がよく使われる。また、ISTの収束を改善したFISTA（fast iterative shrinkage-thresholding algorithm）[11]は、近接点（proximal point） $\mathbf{w}^{(k)}$ を使った次のような反復法である[12, 13, 11]。

$$\begin{aligned} \mathbf{w}^{(1)} &:= \mathbf{x}^{(0)}, \quad t_1 := 1 \\ \mathbf{x}^{(k)} &:= \text{soft} \left(\mathbf{w}^{(k)} + \frac{1}{L} \mathbf{A}^\top (\mathbf{b} - \mathbf{Aw}^{(k)}), \frac{\lambda}{L} \right) \\ t_{k+1} &:= \frac{1 + \sqrt{1 + 4t_k^2}}{2} \\ \mathbf{w}^{(k+1)} &:= \mathbf{x}^{(k)} + \frac{t_k - 1}{t_{k+1}} (\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}) \end{aligned}$$

ADMM

交互方向乗数法 (alternating direction method of multipliers; ADMM) [14] は、次のような最適化問題を解く反復解法である。

$$\min_{(\mathbf{x}, \mathbf{z})} f(\mathbf{x}) + g(\mathbf{z}) \quad \text{subject to} \quad \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{z} = \mathbf{c} \quad (42)$$

ただし、 $f(\mathbf{x})$ と $g(\mathbf{z})$ は凸関数とする。最適解を得る反復式は次のとおりである。

$$\mathbf{x}^{(k+1)} := \arg \min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2} \|\mathbf{y}^{(k)} + \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{z}^{(k)} - \mathbf{c}\|_2^2 \quad (43)$$

$$\mathbf{z}^{(k+1)} := \arg \min_{\mathbf{z}} g(\mathbf{z}) + \frac{\rho}{2} \|\mathbf{y}^{(k)} + \mathbf{F}\mathbf{x}^{(k+1)} + \mathbf{G}\mathbf{z} - \mathbf{c}\|_2^2 \quad (44)$$

$$\mathbf{y}^{(k+1)} := \mathbf{y}^{(k)} + \mathbf{F}\mathbf{x}^{(k+1)} + \mathbf{G}\mathbf{z}^{(k+1)} - \mathbf{c} \quad (45)$$

ただし、 $\rho > 0$ は定数である。ADMM については Boyd らによる解説 [15] が詳しい。また、Web サイト “MATLAB scripts for alternating direction method of multipliers” では、BP や LASSO を含むいくつかの問題について、ADMM から導出された解法を MATLAB で実装した例が紹介されている。

スパース解法の導出例 ADMM から反復計算のアルゴリズムを導出する手順はおよそ次のとおりである。

1. 目的の最小化問題を式 (42) の形式に書き換える。
2. ADMM の反復式 (43) と (44) の最小化問題が解けることを確認する。

式 (35) の BP を例に、ADMM からスパース解法を導出してみよう。まず、式 (35) の BP を次のように書き換える。

$$\min_{(\mathbf{x}, \mathbf{z})} f(\mathbf{x}) + \|\mathbf{z}\|_1 \quad \text{subject to} \quad \mathbf{x} = \mathbf{z} \quad (46)$$

すなわち、式 (42) を以下のように設定したものと解釈できる。

$$f(\mathbf{x}) = \iota_C(\mathbf{x}) = \begin{cases} 0 & \text{if } \mathbf{x} \in C = \{\mathbf{x} \mid \mathbf{A}\mathbf{x} = \mathbf{b}\} \\ \infty & \text{otherwise} \end{cases} \quad (47)$$

$$g(\mathbf{z}) = \|\mathbf{z}\|_1 \quad (48)$$

$$\mathbf{F} = \mathbf{I}, \quad \mathbf{G} = -\mathbf{I}, \quad \mathbf{c} = \mathbf{0} \quad (49)$$

ここで、 $\iota_C(\mathbf{x})$ は指示関数 (indicator function) と呼ばれ、 \mathbf{x} が集合 C に属しているかどうかを判定する関数である。式 (47) のように集合を定義すれば、式 (46) を最小にできる \mathbf{x} は $\mathbf{b} = \mathbf{A}\mathbf{x}$ を満たす \mathbf{x} に限定される。この解釈と式 (43), (44), (45) から、次のような反復式が得られる。

$$\mathbf{x}^{(k+1)} := \arg \min_{\mathbf{x}} \iota_C(\mathbf{x}) + \frac{\rho}{2} \|\mathbf{x} - (\mathbf{z}^{(k)} - \mathbf{y}^{(k)})\|_2^2 \quad (50)$$

$$= (\mathbf{I} - \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^{-1} \mathbf{A})(\mathbf{z}^{(k)} - \mathbf{y}^{(k)}) + \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^{-1} \mathbf{b} \quad (51)$$

$$\mathbf{z}^{(k+1)} := \arg \min_{\mathbf{z}} \|\mathbf{z}\|_1 + \frac{\rho}{2} \|(\mathbf{x}^{(k+1)} + \mathbf{y}^{(k)}) - \mathbf{z}\|_2^2 \quad (52)$$

$$= \text{soft}(\mathbf{x}^{(k+1)} + \mathbf{y}^{(k)}, \frac{1}{\rho}) \quad (53)$$

$$\mathbf{y}^{(k+1)} := \mathbf{y}^{(k)} + \mathbf{x}^{(k+1)} - \mathbf{z}^{(k+1)} \quad (54)$$

式(50)は、「 $\mathbf{A}\mathbf{x} = \mathbf{b}$ を満たす \mathbf{x} のうち、点 $(\mathbf{z}^{(k)} - \mathbf{y}^{(k)})$ から ℓ_2 距離が最も近い点を $\mathbf{x}^{(k+1)}$ とする」という意味である。つまり、式(51)のように、点 $(\mathbf{z}^{(k)} - \mathbf{y}^{(k)})$ からの ℓ_2 最小長さ解が $\mathbf{x}^{(k+1)}$ となる（演習問題3）。また、式(52)は式(39)と同様の最小化問題であり、 $(\mathbf{x}^{(k+1)} + \mathbf{y}^{(k)})$ の成分を $\lambda = 1/\rho$ だけ縮めるソフト閾値処理で最小解 $\mathbf{z}^{(k+1)}$ が得られる。この例では、ADMMの反復式(43)と(44)がそれぞれ ℓ_2 最小長さ解とソフト閾値処理で解ける問題になった。

一般に、凸関数 $f(\mathbf{x})$ と定数 $\rho > 0$ が与えられたとき、点 \mathbf{v} からの ℓ_2 距離で正則化¹⁹した最小化問題の解を表す関数

$$\text{prox}_{f,\rho}(\mathbf{v}) := \arg \min_{\mathbf{x}} f(\mathbf{x}) + \frac{\rho}{2} \|\mathbf{x} - \mathbf{v}\|_2^2 \quad (55)$$

を近接写像（proximity operator）と呼ぶ。例えばBPの場合、式(50)と(52)は近接写像である。

$$\mathbf{x}^{(k+1)} = \text{prox}_{\ell_C,\rho}(\mathbf{z}^{(k)} - \mathbf{y}^{(k)}) \quad (56)$$

$$\mathbf{z}^{(k+1)} = \text{prox}_{\|\cdot\|_1,\rho}(\mathbf{x}^{(k+1)} + \mathbf{y}^{(k)}) \quad (57)$$

ADMMからのアルゴリズムの導出では、反復式(43)と(44)が計算しやすい近接写像になることが重要となる。

まとめ：結局どの問題をどう解けばよいのか

解が極めてスパースであるならば、その非ゼロ成分を次々と特定する貪欲法がお勧めである。実際、凸緩和の解法よりもずっと少ない手間数で解が得られることが多い。しかし、途中で台の選択に失敗するとそこで解への収束が破綻する。そのような失敗は基底 \mathbf{A} の性質に依存する。

一方、凸緩和の解法は、貪欲法よりも計算時間を要することが多いが、任意の初期値からスパース解へ収束する反復計算になっているので、極端にスパースではないスパース解を求める応用問題に向いている。実際、画像や音はフーリエまたはウェーブレット基底で表現すると、ある程度のスパース性が現れる。また、貪欲法で失敗しやすい基底 \mathbf{A} や低次元の \mathbf{b} に対しても、凸緩和の解法で適切なスパース解が得られることがある。

スパース解が $\mathbf{A}\mathbf{x} = \mathbf{b}$ を厳密に満たすならばOMPやBPで良いが、ノイズの混入を考慮するとBPDN, LASSO, ℓ_1 -LSなどが実用的であろう。ただし、凸緩和では定数を利用者が設定する必要がある。例えば式(36)のBPDNでは、 \mathbf{b} が何らかの観測値の場合、定数 δ をノイズの強さの目安として与えることになる。しかし、式(37)の τ や式(38)の λ の値の目安を具体的に見積もることは簡単ではなく、しばしば手探りで見つける作業を要する。

演習問題

1. 式(39)の解が式(40)となることを示せ。

ヒント：例えば、 $x_j \geq 0$ と $x_j < 0$ の場合に分けて x_j の2次関数を平方完成する。最小となる x_j は定数 λ と b_j の大小関係に依存する。

2. IST を導出せよ。

ヒント：式(41)は、 $F(\mathbf{x}, \mathbf{x}^{(k)})$ が最小になるような \mathbf{x} を求める問題なので、 $F(\mathbf{x}, \mathbf{x}^{(k)})$ を定数倍しても解は変わらない。また、 $\mathbf{x}^{(k)}$, \mathbf{A} , \mathbf{b} のみからなる項を省略または追加しても解は変わらない。

¹⁹Moreau-Yosida regularization と呼ばれる。

3. 式(51)を導出せよ.

ヒント: 点 $\mathbf{p} = \mathbf{z}^{(k)} - \mathbf{y}^{(k)}$ からの ℓ_2 最小長さ解を求める問題である. ℓ_2 最小長さ解の導出と同様にラグランジュ乗数法を適用する.

$$L(\mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{2}(\mathbf{x} - \mathbf{p})^\top(\mathbf{x} - \mathbf{p}) + \boldsymbol{\lambda}^\top(\mathbf{b} - A\mathbf{x})$$

4. 凸集合 $C = \{\mathbf{x} \mid \|\mathbf{x} - \mathbf{p}\|_2 \leq \varepsilon\}$ の指示関数を $\iota_C(\mathbf{x})$ とする. 近接写像が

$$\text{prox}_{\iota_C, \rho}(\mathbf{v}) := \arg \min_{\mathbf{x}} \iota_C(\mathbf{x}) + \frac{\rho}{2} \|\mathbf{x} - \mathbf{v}\|_2^2 = \begin{cases} \mathbf{v} & \text{if } \mathbf{v} \in C \\ \mathbf{p} + \varepsilon \frac{\mathbf{v} - \mathbf{p}}{\|\mathbf{v} - \mathbf{p}\|_2} & \text{otherwise} \end{cases} \quad (58)$$

と書けることを図説せよ.

5. IST を MATLAB 関数として実装せよ.

```
function x = IST(b, A, lambda, L, delta)
```

この行から書き始め, `IST.m` として保存せよ.

ヒント: MATLAB では, ソフト閾値処理の関数を 1 行で書ける.

```
soft = @(x, th) sign(x) .* max(abs(x) - th, 0);
```

6. A , x , b を以下のように生成し, x を未知として式(34)を IST で解け.

- A をサイズ $m \times n$ のランダム行列とする. 要素は正規分布から生成する.
- x を k スペースなベクトルとし, 非ゼロ成分は正規分布から生成する.
- $b = Ax$ とする.

次に, 非ゼロ成分の個数 k に対する IST の解の誤差を観察せよ. すなわち, 横軸を k , 縦軸を相対残差 $\|b - Ax\|_2 / \|b\|_2$ としてグラフを描け.

ヒント: 以下のコードを試し, IST で解が得られるかどうか確認せよ.

```
m = 100; n = 1000; A = randn(m,n);
k = 10; S = randperm(n); S = S(1:k);
x = zeros(n,1); x(S) = randn(k,1);
b = A * x;
x_est = IST(b, A, 適当な値lambda, 適当な値L, 1e-7);
figure, plot(1:n, x, '.', 1:n, x_est, 'o')
```

このコードを改造して相対残差を観察せよ. 誤差を統計的に評価するため, 反復試行する必要がある. なお, 指定する λ と L の値によっては, IST の計算が発散することがあるので注意せよ.

参考文献

- [1] S.S. Chen, D.L. Donoho, and M.A. Saunders, “Atomic decomposition by basis pursuit,” SIAM J. Sci. Comput., vol.20, no.1, pp.33–61, 1998.
- [2] E.J. Candès, J. Romberg, and T. Tao, “Stable signal recovery from incomplete and inaccurate measurements,” Comm. on Pure and Applied Math, vol.59, no.8, pp.1207–1223, 2006.
- [3] E.J. Candès, “The restricted isometry property and its implications for compressed sensing,” Comptes Rendus Mathématique, vol.346, no.9-10, pp.589–592, May 2008.
- [4] R. Tibshirani, “Regression shrinkage and selection via the lasso,” Journal of the Royal Statistical Society, Series B, vol.58, pp.267–288, 1996.
- [5] M.A.T. Figueiredo, R.D. Nowak, and S.J. Wright, “Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems,” IEEE Journal of Selected Topics in Signal Processing, vol.1, no.4, pp.586–597, 2007.
- [6] D.L. Donoho, “De-noising by soft-thresholding,” Information Theory, IEEE Transactions on, vol.41, no.3, pp.613–627, 1995.
- [7] S.J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, “An interior-point method for large-scale l_1 -regularized least squares,” IEEE Journal on Selected Topics in Signal Processing, vol.1, no.4, pp.606–617, 2007.
- [8] S.J. Wright, R.D. Nowak, and M.A.T. Figueiredo, “Sparse reconstruction by separable approximation,” IEEE Trans. Signal Processing, vol.57, no.7, pp.2479–2493, 2009.
- [9] R. Tomioka and M. Sugiyama, “Dual augmented Lagrangian method for efficient sparse reconstruction,” IEEE Signal Processing Letters, vol.16, no.2, pp.1067–1070, 2009.
- [10] I. Daubechies, M. Defrise, and C. De Mol, “An iterative thresholding algorithm for linear inverse problems with a sparsity constraint,” Communications on Pure and Applied Mathematics, vol.57, no.11, pp.1413–1457, 2004.
- [11] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” SIAM J. Img. Sci., vol.2, no.1, pp.183–202, March 2009.
- [12] Y. Nesterov, “A method of solving a convex programming problem with convergence rate $O(1/k^2)$,” Soviet Mathematics Doklady, vol.27, no.2, pp.372–376, 1983.
- [13] Y. Nesterov, “Gradient methods for minimizing composite objective function,” 2007.
- [14] D. Gabay and B. Mercier, “A dual algorithm for the solution of nonlinear variational problems via finite element approximation,” Computers and Mathematics with Applications, vol.2, no.1, pp.17–40, 1976.
- [15] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” Found. Trends Mach. Learn., vol.3, no.1, pp.1–122, Jan. 2011.

★ 固有ベクトルで分類 — グラフのスペクトルとデータのクラスタリング —

概要：グラフの解析によるデータの分類

データを整理してグループにまとめることは基本的な解析のひとつである。データクラスタリングの目標は、データの集合からいくつかの自然なグループ（クラスタ）を見つけることである²⁰。様々な応用に対してクラスタリングの手法が数多く提案されているが[1, 2]、ここでは、クラスタリングのためのグラフ表現とその単純な固有値解析を紹介しよう。

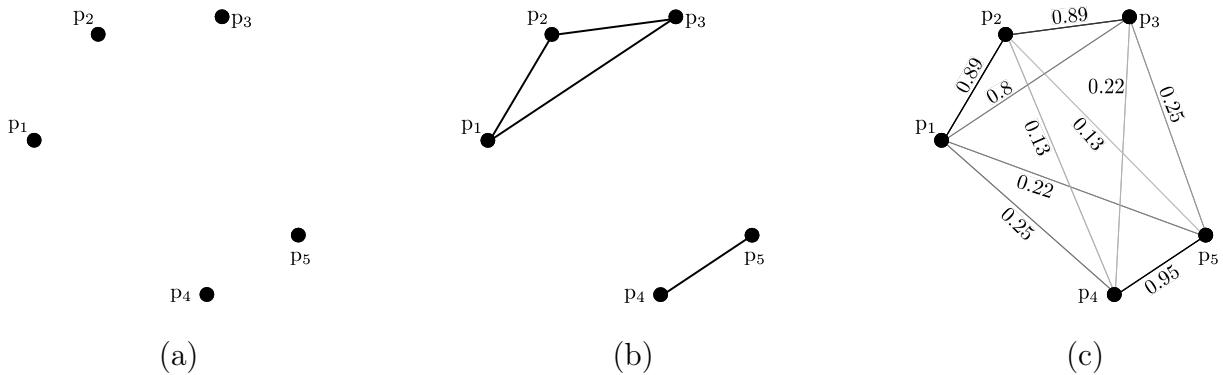


図 3 グラフの解析によるクラスタリング。グラフの節と枝はそれぞれデータとデータ間の関係の深さを表す。(a) 与えられたデータ点。(b) クラスタを表す非連結グラフ。(c) 枝がデータ間の類似度を持っている類似度グラフ。

データ集合が与えられたとき、グラフを使って自動的にクラスタを見つけられるだろうか？例えば、図 3(a) のように、5つのデータ点が与えられたときに2つのクラスタを見つけたいとする。任意のデータ点の間のつながり（隣接）が分かっているならば、図 3(b) のようなグラフを作ることができる。図の例では、他と連結していない2つのグラフの部分（連結成分）によってグラフが構成されており、2つのクラスタ $C_1 = \{p_1, p_2, p_3\}$ と $C_2 = \{p_4, p_5\}$ を特定できる。このように、クラスタを表す連結成分を「計算」によって見つけることが、グラフを用いたクラスタリングの目標である。

実際の応用では、データ間の隣接が明示的には与えられないかもしれないが、データが持つ属性や特徴を使ってデータどうしの類似性か非類似性を見積もれることがある（データ点の間の距離や内積など）。そのような場合は、例えば図 3(c) のような類似度グラフ（similarity graph）を作れる。類似度グラフは、データ間の類似度を枝の重みとして与えた単純グラフ²¹である。この類似度グラフの枝を切断して非連結のグラフにすることがクラスタリングであると解釈できる。その際、切断する枝の重みの和が最小になるように切断すべきである。

²⁰ クラスタは簡単に定義できるものではない。クラスタ内のデータが互いに近く、クラスタが互いに離れているという性質が望ましいとされるが、データやクラスタ間の近い・遠い、似ている・似ていないことを適切に測れるとは限らない。

²¹ 閉路も多重の枝も持たない無向グラフを単純グラフと呼ぶ。

代数的なグラフの取り扱い

隣接行列 グラフを解析するため、隣接行列（adjacency matrix）を用いてグラフを代数的に表すことができる。 n 個の節を持つグラフ G の隣接行列 \mathbf{A} は、節 i から節 j への枝の本数を第 i 行 j 列要素に持つ $n \times n$ 行列である。単純グラフは 0, 1 の 2 値の要素を持つ隣接行列で表される。例えば、図 3(b) の単純グラフの隣接行列は次のように書ける。

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (59)$$

次数行列 節 i から他の節への枝の本数を次数（degree）という。隣接行列の第 i 行の行和は節 i の次数に等しい。 n 個の成分がすべて 1 のベクトルを $\mathbf{1}_n$ とすると、 n 個の節の次数を成分に持つベクトルは $\mathbf{d} = \mathbf{A}\mathbf{1}_n$ のように計算できる。次数を対角要素とする行列を次数行列（degree matrix）と定義する。つまり、次数ベクトル \mathbf{d} を用いて $\mathbf{D} = \text{diag}(\mathbf{d})$ と書き表す。図 3(b) の単純グラフの次数ベクトルと次数行列は次のように書ける。

$$\mathbf{d} = \mathbf{A}\mathbf{1}_n = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 1 \\ 1 \end{bmatrix}, \quad \mathbf{D} = \text{diag} \left(\begin{bmatrix} 2 \\ 2 \\ 2 \\ 1 \\ 1 \end{bmatrix} \right) = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (60)$$

クラスタを表すベクトル n 個のデータ点について、成分がクラスタへの所属を表しているような n 次元ベクトルをクラスタ指示ベクトル（cluster indicator）と呼ぶ。図 3 のようなクラスタ $C_1 = \{p_1, p_2, p_3\}$ と $C_2 = \{p_4, p_5\}$ のクラスタ指示ベクトルは、それぞれ $\mathbf{h}^{(1)} = [1, 1, 1, 0, 0]$ と $\mathbf{h}^{(2)} = [0, 0, 0, 1, 1]$ である。

隣接行列 \mathbf{A} または次数行列 \mathbf{D} をクラスタ指示ベクトルに乘すと、クラスタに属す節の次数の総和が得られる。例えば、

$$\mathbf{A}\mathbf{h}^{(1)} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 0 \\ 0 \end{bmatrix} \quad \text{または} \quad \mathbf{D}\mathbf{h}^{(1)} = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 2 \\ 0 \\ 0 \end{bmatrix} \quad (61)$$

ラプラシアン行列 グラフ理論では、グラフのラプラシアン行列（Laplacian matrix）が $\mathbf{L} = \mathbf{D} - \mathbf{A}$ と定義されている。ラプラシアンとは、任意の点における値とその近傍の平均値の差を測る 2 階微分演算子のことである。もし、グラフの節を定義域として何らかの関数 f が定義されていたら、 \mathbf{L} はグラフ上で f のラプラシアンを計算する演算子になる。例えば、図 3(b) のグラフのラプラシアン行列 \mathbf{L} をグラフ上の関数 f に作用させると、

$$\mathbf{L} = \begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}, \quad \mathbf{L}\mathbf{f} = \mathbf{L} \begin{bmatrix} f(p_1) \\ f(p_2) \\ f(p_3) \\ f(p_4) \\ p(p_5) \end{bmatrix} = \begin{bmatrix} 2(f(p_1) - \frac{f(p_2)+f(p_3)}{2}) \\ 2(f(p_2) - \frac{f(p_1)+f(p_3)}{2}) \\ 2(f(p_3) - \frac{f(p_1)+f(p_2)}{2}) \\ f(p_4) - f(p_5) \\ f(p_5) - f(p_4) \end{bmatrix}. \quad (62)$$

アルゴリズム 2 連結成分を見つけるクラスタリング

入力：隣接行列 \mathbf{A} ;

出力：クラスタの集合 $\mathcal{C} = \{C_1, \dots, C_k\}$;

- 1 次数行列 $\mathbf{D} = \text{diag}(\mathbf{A}\mathbf{1}_n)$ を計算する;
 - 2 ラプラシアン行列 $\mathbf{L} = \mathbf{D} - \mathbf{A}$ を計算する;
 - 3 \mathbf{L} のゼロ固有値に属する固有ベクトル $\mathbf{h}^{(l)}$ を計算する;
 - 4 $h_i^{(l)} \neq 0$ ならば i を C_l に所属させる.
-

という計算になる. $\mathbf{L}\mathbf{f}$ の第 1 成分は, p_1 における f と, 隣接する p_2 および p_3 における f の平均値との差になっていることがわかる.

ラプラシアン行列にはいくつかの興味深い性質がある.

- 任意のグラフについて $\mathbf{L}\mathbf{1}_n = \mathbf{D}\mathbf{1}_n - \mathbf{A}\mathbf{1}_n = \mathbf{d} - \mathbf{d} = 0\mathbf{1}_n$ が必ず成り立つので, \mathbf{L} は少なくともひとつゼロの固有値を持ち, その固有ベクトルは $\mathbf{h} = \mathbf{1}_n$ である.
- グラフが k 個の連結成分からなるとき, 連結成分を表すクラスタ指示ベクトル $\mathbf{h}^{(l)}$ ($l = 1, \dots, k$) がラプラシアン行列 \mathbf{L} のゼロの固有値に属する固有ベクトルになる. このことは簡単に確認できる. クラスタ C_l について, 式 (61) のように $\mathbf{A}\mathbf{h}^{(l)} = \mathbf{D}\mathbf{h}^{(l)}$ が成り立つから, $\mathbf{L}\mathbf{h}^{(l)} = (\mathbf{D} - \mathbf{A})\mathbf{h}^{(l)} = 0\mathbf{h}^{(l)}$ を得る.
- 一般に, \mathbf{L} のゼロの固有値が k 個ならば, グラフの連結成分は k 個ある (演習問題 1).
- ラプラシアン行列の固有値は非負である (演習問題 2). ラプラシアン行列 \mathbf{L} の固有値はグラフスペクトル (graph spectra) と呼ばれる.

ひとつの連結成分からなるグラフのラプラシアン行列はゼロの固有値をひとつ持つ. グラフが複数の連結成分からなるかどうかは第 2 固有値から判明する. それゆえ, 第 2 固有値は代数的連結度 (algebraic connectivity) と呼ばれる. また, 第 2 固有値に属する固有ベクトルは Fiedler ベクトル (Fiedler vector) [3] とも呼ばれる.

グラフのスペクトルと分割

グラフの連結成分を見つける クラスタ指示ベクトルは, ラプラシアン行列のゼロの固有値に属する固有ベクトルである. この性質を利用して, グラフの連結成分としてクラスタを見つけるアルゴリズムを 2 に示す. このように, データの関係を表現するグラフを記述する行列の固有値解析によってクラスタを見つけることができる.

類似度グラフを分割する 類似度グラフの枝のうち小さな類似度を持つものを切断してクラスタを作ることを考える. データ間の関係が類似度グラフとして与えられたとき, これを類似度行列 (affinity/similarity matrix) によって代数的に表すことができる. 類似度行列は隣接行列を実数に拡張したようなものである. 類似度行列 \mathbf{W} の第 i 行 j 列要素は節 i から節 j への枝の重み (つまり類

似度) である. 図 3(c) の類似度グラフの類似度行列は次のように書ける.

$$\mathbf{W} = \begin{bmatrix} 0 & 0.89 & 0.80 & 0.25 & 0.22 \\ 0.89 & 0 & 0.89 & 0.13 & 0.13 \\ 0.80 & 0.89 & 0 & 0.22 & 0.25 \\ 0.25 & 0.13 & 0.22 & 0 & 0.95 \\ 0.22 & 0.13 & 0.25 & 0.95 & 0 \end{bmatrix} \quad (63)$$

類似度グラフの次数行列 \mathbf{D} とラプラシアン行列 \mathbf{L} も同様に類似度行列を用いて定義される.

$$\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1}_n) \quad (64)$$

$$\mathbf{L} = \mathbf{D} - \mathbf{W} \quad (65)$$

類似度グラフのどの節も非ゼロの重みの枝をひとつ以上持つならば, 類似度グラフには非連結成分がない. その類似度グラフのラプラシアン行列はひとつだけゼロ固有値を持ち, 固有ベクトルは $\mathbf{h} = \mathbf{1}_n$ である. たとえ類似度グラフが明らかに k 個の部分グラフからなっていても, それらは小さな重みの枝で弱く結ばれているので, アルゴリズム 2 によってクラスタ指示ベクトル $\mathbf{h}^{(l)}$ は得られない. その代わり, 類似度グラフのラプラシアン行列は k 個の小さな固有値を持ち, それらの固有ベクトルがクラスタを教えてくれる. なぜならば, 固有ベクトルがクラスタ指示ベクトルの近似になっているからである. $\mathbf{W}\mathbf{h}^{(l)} \neq \mathbf{D}\mathbf{h}^{(l)}$ なので, $\mathbf{L}\mathbf{h}^{(l)} = (\mathbf{D} - \mathbf{W})\mathbf{h}^{(l)} \neq 0\mathbf{h}^{(l)}$ である. しかし, $\mathbf{D}\mathbf{h}^{(l)}$ と $\mathbf{W}\mathbf{h}^{(l)}$ の差は, 第 l 部分グラフの節と他の部分グラフの節の間の枝が持つ小さな重みの総和を成分とする小さなベクトルに過ぎない.

図 3(c) の類似度グラフの場合, ラプラシアン行列の固有値は 0, 0.99, 2.50, 2.96, 3.01 である. 明らかに小さな固有値がふたつ存在し, それらに属する固有ベクトルを列に並べると

$$\mathbf{H}_2 = [\mathbf{h}^{(1)}, \mathbf{h}^{(2)}] = \begin{bmatrix} 0.45 & 0.33 \\ 0.45 & 0.43 \\ 0.45 & 0.33 \\ 0.45 & -0.55 \\ 0.45 & -0.55 \end{bmatrix}. \quad (66)$$

という行列を作れる. 固有ベクトル $\mathbf{h}^{(1)}$ と $\mathbf{h}^{(2)}$ はクラスタ指示ベクトルそのものではない. しかし, 行列 \mathbf{H}_2 の全 5 行を 2 次元空間の 5 点のデータと見なしてプロットすると, 図 4 のように 2 箇所に固まってクラスタが形成されていることがわかる. これらのクラスタは k 平均クラスタリング (k -means clustering) や閾値処理程度の簡単なベクトル量子化²² で特定することができる.

このように, グラフを表す行列から固有ベクトルを計算すると, 類似度グラフの分割による n 個のデータ点のクラスタリングは, k 次元空間の n 個の点のクラスタリングに帰着する.

スペクトラルクラスタリングのアルゴリズム

固有値解析に基づくデータクラスタリングをアルゴリズム 3 に示す. このアルゴリズムは, グラフの k 分割の「コスト」を最小化する問題について, クラスタ指示ベクトルを離散 2 値から実数へ緩和することで導出される. グラフを切斷するコストは, クラスタの下記の性質を考慮して設計されている.

²² 小数点以下を四捨五入するなど, 実数を何種類かの整数 (または記号) のひとつに置き換えることを量子化という. 同様に, ベクトルを何種類かの整数 (または記号) のひとつに対応させることをベクトル量子化と呼ぶ. ベクトルで表されるデータのクラスタリングはベクトル量子化の一手法である.

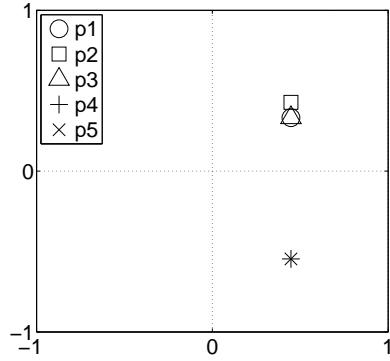


図 4 \mathbf{H}_2 の 5 行が 2 次元空間でふたつの密なクラスタを呈す。

アルゴリズム 3 スペクトラルクラスタリング

入力：類似度行列 \mathbf{W} , クラスタ数 k ;

出力：クラスタの集合 $\mathcal{C} = \{C_1, \dots, C_k\}$;

1 次数行列 $\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1}_n)$ を計算する;

2 Rcut[4] の場合:

ラプラシアン行列 $\mathbf{L} = \mathbf{D} - \mathbf{W}$ の k 個の小さな固有値に属する固有ベクトルを列に並べた行列 $\mathbf{X}_k = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(k)}] \in \mathbb{R}^{n \times k}$ を作る;

Ncut[5, 6]) の場合:

正規化類似度行列 $\mathbf{S} = \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$ の k 個の大きな固有値に属する固有ベクトルを列に並べた行列 $\mathbf{X}_k = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(k)}] \in \mathbb{R}^{n \times k}$ を作る;

3 \mathbf{X}_k の各行ベクトルを正規化する;

4 ベクトル量子化によって各データ点をクラスタのひとつ $C_l \in \mathcal{C}$ に所属させる: 典型的には, \mathbf{X}_k の n 本の行ベクトルに k 平均クラスタリングを適用する.

外的分離 (external isolation) クラスタが互いに離れていること. 第 l クラスタ C_l の全データ点と他のクラスタのデータ点をつなぐ枝の重みの和は, $\mathbf{h}^{(l)\top} \mathbf{L} \mathbf{h}^{(l)}$ のように計算できる. この値は小さいほどよい. C_l が他のクラスタから離れるほど小さくなるからである.

内的結合 (internal cohesion) クラスタ内のデータが互いに近いこと. 第 l クラスタ C_l 内のデータの個数は $\mathbf{h}^{(l)\top} \mathbf{h}^{(l)}$ である. C_l 内のデータから他の節への枝の重みの総和は $\mathbf{h}^{(l)\top} \mathbf{D} \mathbf{h}^{(l)}$ のように計算できる. どのクラスタについても, これらの量が大きいほどよい. 各クラスタがよくまとまっていて, バランスがとれているほど大きくなるからである.

スペクトラルクラスタリングのためのグラフ切断コストがいくつか設計されている.

$$\text{Ratio cut [4]}: \quad J_{\text{Rcut}}(\mathbf{H}) = \sum_{l=1}^k \frac{\mathbf{h}^{(l)\top} \mathbf{L} \mathbf{h}^{(l)}}{\mathbf{h}^{(l)\top} \mathbf{h}^{(l)}} \quad (67)$$

$$\text{Normalized cut [5, 6]}: \quad J_{\text{Ncut}}(\mathbf{H}) = \sum_{l=1}^k \frac{\mathbf{h}^{(l)\top} \mathbf{L} \mathbf{h}^{(l)}}{\mathbf{h}^{(l)\top} \mathbf{D} \mathbf{h}^{(l)}} \quad (68)$$

コスト $J_{\text{Rcut}}(\mathbf{H})$ や $J_{\text{Ncut}}(\mathbf{H})$ は, k 本のクラスタ指示ベクトル $\mathbf{h}^{(l)}$ ($l = 1, \dots, k$) の関数である. コストが最小になる $\mathbf{H} = [\mathbf{h}^{(1)}, \dots, \mathbf{h}^{(k)}]$ を探す問題は組合せ最適化問題である. 式(67)や式(68)を最小化する問題を制約付き対角和最小問題に書き換え, クラスタ指示ベクトル $\mathbf{h}^{(l)}$ を実ベクトル $\mathbf{x}^{(l)} \in \mathbb{R}^n$

と見なすと近似的な解法が得られる。実数のクラスタ指示ベクトル $\mathbf{x}^{(l)}$ は次のような固有値問題の固有ベクトルとして求まる。

$$\text{Ratio cut : } \mathbf{L}\mathbf{x}^{(l)} = \lambda_l \mathbf{x}^{(l)} \quad (69)$$

$$\text{Normalized cut : } \mathbf{S}\mathbf{x}^{(l)} = \mu_l \mathbf{x}^{(l)} \quad (70)$$

ただし、 λ_l ($l = 1, \dots, k$) は \mathbf{L} の k 個の小さな固有値、 μ_l ($l = 1, \dots, k$) は $\mathbf{S} = \mathbf{D}^{-1/2} \mathbf{W} \mathbf{D}^{-1/2}$ の k 個の大きな固有値である。導出の詳細は文献 [7] が詳しい。

演習問題

- 連結成分が k 個あるグラフ G のラプラシアン行列を \mathbf{L} とする。 \mathbf{L} のゼロの固有値が k 個ならば、 G の連結成分は k 個あることを証明せよ。

ヒント：連結成分を表す k 本のクラスタ指示ベクトル $\mathbf{h}^{(l)}$ ($l = 1, \dots, k$) が \mathbf{L} の固有ベクトルであり、その固有値はゼロであることから、 \mathbf{L} の固有値のうち少なくとも k 個はゼロであると言える。 \mathbf{L} のゼロの固有値に属する固有ベクトルは、クラスタ指示ベクトルまたはその線形結合で表せるベクトルに限られることを示せばよい。

- ラプラシアン行列の固有値は非負であることを証明せよ。

ヒント：関数 $q(\mathbf{h}) = \mathbf{h}^\top \mathbf{L} \mathbf{h}$ を、行列 \mathbf{L} の二次形式と呼ぶ。 $\|\mathbf{h}\|_2 = 1$ の条件下で二次形式 $q(\mathbf{h})$ の極値は \mathbf{L} の固有値となる。また、隣接行列 \mathbf{A} の要素を用いてラプラシアン行列 $\mathbf{L} = \mathbf{D} - \mathbf{A}$ の二次形式を書き表すと $q(\mathbf{h}) \geq 0$ が示せる。

- スペクトラルクラスタリングを 2 次元の点集合（例えば Clustering datasets

(<http://cs.joensuu.fi/sipu/datasets/>) に適用せよ。ただし、点 $\mathbf{p}^{(i)}$ と点 $\mathbf{p}^{(j)}$ ($i \neq j$) の類似度 w_{ij} を次式のように定義する。

$$w_{ij} = \exp\left(-\frac{\|\mathbf{p}^{(j)} - \mathbf{p}^{(i)}\|_2^2}{2\sigma^2}\right) \quad (71)$$

σ は尺度のパラメタで、近いと見なす点間の距離の目安である。例えば、下記の MATLAB コードのようにデータ集合 Spiral または D31 に試してみよ。

```
close all

% データ集合を読み込む.
data = load('spiral.txt');
n = size(data, 1);

% クラスタ数と尺度を設定する.
k = 3; sigma = 1.0;

% 類似度行列Wと正規化類似度行列Sを計算する.
pairdist = pdist(data(:,1:2));
W = exp(-squareform(pairdist.^2 / (2.0 * sigma.^2))) - eye(n, n);
Dinvsq = diag(1./sqrt(sum(W, 2)));
S = Dinvsq * W * Dinvsq;
```

```
% 固有ベクトルXと固有値Mを求める。
[X, M] = eig(S);
[M, idx] = sort(diag(M), 'descend');
figure, plot(1:n, M)

% 上位k本の固有ベクトルを取り出して行を正規化する。
X = X(:, idx(1:k));
X = diag(1./sqrt(sum(X.^2, 2))) * X;

% k平均クラスタリングでベクトル量子化する。
c = kmeans(X, k, 'emptyaction', 'singleton');

% 結果を表示する。
marker10 = {'+', '*', 'x', '^', 'v', 's', 'p', '^', 'd', 'v'};
col = colormap(lines(16));
figure;
hold on
for t = 1:k,
    p = find(c == t);
    plot(data(p,1), data(p,2), marker10{mod(t,10)+1}, 'Color', col(mod(t,16)+1,:));
end
hold off
```

数千個のデータ点のクラスタリング（数百万個以上の非ゼロ要素を持つ行列の固有ベクトル計算）は数分かかるので注意せよ。

4.* Web サイト [the University of Florida Sparse Matrix Collection
\(<http://www.cise.ufl.edu/research/sparse/matrices/>\)](http://www.cise.ufl.edu/research/sparse/matrices/)

を見て、ひとつ行列を選べ（例えば、c-45.mat, jazz.mat, wiki-Vote.mat, netscience.matなど）。その行列は無向グラフまたは2部グラフの類似度行列と見なせる。スペクトラルクラスタリングを適用せよ。固有値をプロットしてクラスタ数を推測せよ。できればグラフのクラスタを可視化せよ。

ヒント：Ncut を適用するサンプルコードは下記のとおりである。

```
close all
load('jazz.mat'); k = 4;
W = abs(Problem.A);
[m, n] = size(W);
Dinvsq1 = spdiags(1./sqrt(sum(W, 1)'), 0, n, n);
Dinvsq2 = spdiags(1./sqrt(sum(W, 2)), 0, m, m);
S = Dinvsq2 * W * Dinvsq1;
```

```
% スパース行列の特異値分解によって上位 20*  
k 個の固有値, 固有ベクトルを計算する.  
[X, M, ~] = svds(S, 20*k);  
M = diag(M);  
figure, plot(1:length(M), M, 'o')  
  
X = X(:, 1:k);  
X = spdiags(1./sqrt(sum(X.^2, 2)), 0, m, m) * X;  
c = kmeans(X, k, 'emptyaction', 'singleton');
```

参考文献

- [1] A.K. Jain, M. Murty, and P. Flynn, "Data clustering: a review," ACM Computing Surveys, vol.31, no.3, pp.254–323, 1999.
- [2] A.K. Jain, "Data clustering: 50 years beyond K-means," Pattern Recognition Letters, vol.31, no.8, pp.651–666, 2010.
- [3] M. Fiedler, "Algebraic connectivity of graphs," Czechoslovak Mathematical Journal, vol.23, no.2, pp.298–305, 1973.
- [4] L. Hagen and A. Kahng, "New spectral methods for ratio cut partitioning and clustering," IEEE Transactions on Computer-Aided Design, vol.11, no.9, pp.1074–1085, 1992.
- [5] J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.22, no.8, pp.888–905, 2000.
- [6] S.X. Yu and J. Shi, "Multiclass spectral clustering," International Conference on Computer Vision, pp.313–319, 2003.
- [7] U. Von Luxburg, "A tutorial on spectral clustering," Statistics and Computing, vol.17, no.4, pp.395–416, 2007.

★ 2部グラフで同時に分類 — 共クラスタリングと特異値分解 —

前回と今回あらすじ データを節、データ間のつながり（隣接）を枝で表したグラフを調べると、データのグループ（クラスタ）を見つけることができる。隣接の代わりにデータ間の類似度がわかる場合は、枝に類似度が付された類似度グラフを分割することでデータをグループ分けできる。なるべく小さな類似度が付された枝のみを切断してグラフを分割する問題は、グラフを表す行列の固有値問題に近似して解ける。この方法はスペクトルクラスタリングと呼ばれている。

もし類似度グラフが2部グラフになる場合はどうだろうか。集合が2つに分かれている、2つの集合の要素間にだけ対応または関連性が与えられている場合がそうである。そのような場合は、対応または関連性に基づき2つの集合を同時にクラスタリングできる。その際、固有値問題は特異値分解と呼ばれる長方形の行列の分解に帰着する。

対応が表すデータのクラスタ

ふたつのデータの集合があって、一方の集合の要素（データ）がもう一方の集合の部分集合に対応付けられていると見なせるものがある。単語（検索キーワード）と文書（Webページ）の対応や、顧客と商品の対応などがある。ふたつの集合の間に対応（correspondence）が与えられたとき、その対応次第で、ふたつの集合を同時にグループ分けできることがある。

図5(a)は、5種類の商品□と4人の顧客○の対応を表すグラフである。どの商品も顧客の部分集合に対応付けられている。例えば、商品5は顧客2と顧客4に買われている。よく見ると、商品2も同様に顧客2と顧客4に買われている。図5(b)のように少し並べ替えてみるとわかりやすい。商品間や顧客間のつながりは与えられていないが、対応に基づき、商品は{1,3,4}と{2,5}のグループに、顧客は{1,3}と{2,4}のグループにそれぞれ分かれている。このグループ分けは非常に有用である。商品1を買う顧客は商品3や4も買う実態が明らかであり、顧客1が買った商品を顧客3にお勧めする商法が有効であることもわかる。類似の商品を顧客に提示することをコンテンツベースフィルタリング（content-based filtering）、顧客の嗜好の類似性に基づき商品を提示することを協調フィルタリング（collaborative filtering）と呼ぶ。

2部グラフの切断によるデータの同時分類

2部グラフ (bipartite graph) とは、節がふたつの集合に分かれている、一方の集合 M の節ともう一方の集合 N の節の間にだけ枝があるようなグラフである。図5のように、2部グラフは対応（correspondence）を表しており、その対応次第で、ふたつの集合 M と N の要素をそれぞれ同時にグループ分けできことがある。そのようなグループ分けは共クラスタリング（biclustering）²³ と呼ばれる。

実際の応用では、要素間の対応の有無ではなく、関連性の強さで対応が表されていることがある。例えば、表1のように、顧客が購入した商品の数や与えた星の数などを参考に、商品と顧客の関係を調べることになるであろう。そのような場合は、図5(c)のような**2部類似度グラフ** (bipartite similarity graph) を作れる。2部類似度グラフは、集合 M の要素と集合 N の要素の対応を表す枝に重みが与えられている2部グラフである。なるべく小さな重みの枝を切断して2部類似度グラフを非連結のグラフに分割することが共クラスタリングであると解釈できる。

²³a.k.a. block clustering, co-clustering, or two-mode clustering.

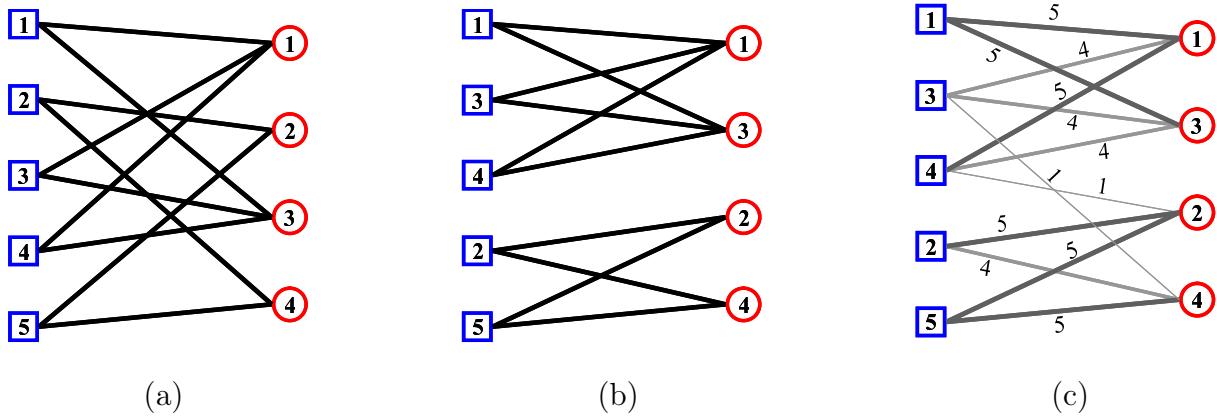


図 5 対応を表す2部グラフとクラスタリング。5個の□と4個の○はそれぞれ商品と顧客を表す節である。(a)商品とそれを買った顧客の対応を枝で表す2部グラフ。(b)(a)の商品と顧客を並べ替えたもの。(c)商品と顧客の関連性の強さ（購入数や評価点数など）を重みとした類似度グラフ。

表 1 関連性の強さで表された対応の例。顧客と商品の対応を示す値が与えられている。実際の応用では、顧客が商品を評価した点数かもしれないし、購入数かもしれない。

	顧客 1	顧客 2	顧客 3	顧客 4
商品 1	5	0	5	0
商品 2	0	5	0	4
商品 3	4	0	4	1
商品 4	5	1	4	0
商品 5	0	5	0	5

2部グラフのスペクトラルクラスタリング

集合 M と N の要素間の対応を表す m 対 n の2部類似度グラフが与えられたとき、集合 M と N のそれぞれ k 個のグループに分ける共クラスタリングを実行しよう。Ncut[1, 2]) のスペクトラルクラスタリングを2部類似度グラフに適用すると、次のような計算手順になる。

1. 与えられた2部類似度グラフを類似度行列 \mathbf{W} で表す。
2. 次数行列 $\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1}_{m+n})$ を作る。
3. 正規化類似度行列 $\mathbf{S} = \mathbf{D}^{-1/2}\mathbf{W}\mathbf{D}^{-1/2}$ を作る。
4. \mathbf{S} の k 個の大きな固有値に属する固有ベクトルを計算し、固有ベクトルを列に並べた行列 $\mathbf{X}_k = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(k)}] \in \mathbb{R}^{(m+n) \times k}$ を作る。
5. \mathbf{X}_k の各行を正規化し、ベクトル量子化する。

この手順を更に詳しく見てみよう。

類似度行列の作成 類似度行列 \mathbf{W} の第 i 行 j 列要素は第 i 節から第 j 節への枝の重みである。 m 対 n の2部類似度グラフは $m + n$ 個の節を持つので、 $m + n$ 次正方の類似度行列で表せる。 n 個の節を

第 $m+1, \dots, m+n$ 節とすると, $(i, j) \in \{1, \dots, m\} \times \{m+1, \dots, m+n\}$ のときだけ類似度行列の要素 $w_{ij} := c_{ij}$ は非ゼロになり得る.

$$\mathbf{W} = \begin{bmatrix} \mathbf{O} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{O} \end{bmatrix} \quad (72)$$

類似度行列のブロック行列 $\mathbf{C} \in \mathbb{R}^{m \times n}$ を本書では長方類似度行列 (rectangular similarity matrix) と呼ぶことにする.

例： 図 5(c) は $m=5$ 対 $n=4$ の 2 部類似度グラフであり, $m+n=9$ 個の節を持つ. 商品 1 から 5, 顧客 1 から 4 を, それぞれ第 1 から 5 節, 第 6 から 9 節とすると, 類似度行列は次のように書ける.

$$\mathbf{W} = \left[\begin{array}{cc|cc} 0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 & 4 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 5 & 1 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 5 \\ \hline 5 & 0 & 4 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 1 & 5 & 0 & 0 & 0 & 0 \\ 5 & 0 & 4 & 4 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 1 & 0 & 5 & 0 & 0 & 0 & 0 \end{array} \right] = \begin{bmatrix} \mathbf{O} & \mathbf{C} \\ \mathbf{C}^\top & \mathbf{O} \end{bmatrix} \quad \text{ただし, } \mathbf{C} = \begin{bmatrix} 5 & 0 & 5 & 0 \\ 0 & 5 & 0 & 4 \\ 4 & 0 & 4 & 1 \\ 5 & 1 & 4 & 0 \\ 0 & 5 & 0 & 5 \end{bmatrix} \quad (73)$$

このように, 2 部類似度グラフの類似度行列 \mathbf{W} は, 対角ブロックにゼロ行列, 非対角ブロックに \mathbf{C} と \mathbf{C}^\top を持つ. この長方類似度行列 \mathbf{C} は表 1 そのものである.

次数行列の作成 $\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1}_{m+n})$ は \mathbf{W} の行和

$$d_i = \sum_{j=1}^{m+n} w_{ij} = \begin{cases} \sum_{j=1}^n c_{ij} & \text{if } 1 \leq i \leq m \\ \sum_{j=1}^m c_{ji} & \text{if } m+1 \leq i \leq m+n \end{cases} \quad (\mathbf{C} \text{ の行和 } \mathbf{C}\mathbf{1}_n) \quad (74)$$

からなる対角行列である. また, $\mathbf{D}^{-1/2}$ も $1/\sqrt{d_i}$ からなる対角行列である.

例： 式 (73) の類似度行列から得られる次数行列 \mathbf{D} の対角要素は

$$\begin{aligned} \mathbf{W}\mathbf{1}_9 &= [5+5, 5+4, 4+4+1, 5+1+4, 5+5, 5+4+5, 5+1+5, 5+4+4, 4+1+5]^\top \\ &= [10, 9, 9, 10, 10, 14, 11, 13, 10]^\top \end{aligned}$$

である. 初めの $m=5$ 個の要素は \mathbf{C} の行和, 残りの $n=4$ 個の要素は \mathbf{C} の列和であることがわかる.

正規化類似度行列の作成 正規化類似度行列 \mathbf{S} の第 i 行 j 列要素は

$$s_{ij} = \frac{w_{ij}}{\sqrt{d_i d_j}}$$

であり、式(72)と同様に対角ブロックにゼロ行列を持っている。式(74)のように、 d_i は \mathbf{C} の行和と列和からなるので、正規化類似度行列 \mathbf{S} は次のように書ける。

$$\mathbf{S} = \begin{bmatrix} \mathbf{O} & \mathbf{Z} \\ \mathbf{Z}^\top & \mathbf{O} \end{bmatrix} \quad \text{ただし, } \mathbf{Z} = \mathbf{D}_2^{-1/2} \mathbf{C} \mathbf{D}_1^{-1/2}, \quad \mathbf{D}_2 = \text{diag}(\mathbf{C} \mathbf{1}_n), \quad \mathbf{D}_1 = \text{diag}(\mathbf{C}^\top \mathbf{1}_m) \quad (75)$$

固有ベクトルの計算 式(75)のように、 \mathbf{S} はブロック状の $m+n$ 次正方行列なので、 $m+n$ 次元の固有ベクトルを持つ。この固有ベクトルを、 m 次元ベクトル $\mathbf{u} \in \mathbb{R}^m$ と n 次元ベクトル $\mathbf{v} \in \mathbb{R}^n$ の連結と見なして $\mathbf{x} = [\mathbf{u}^\top \mathbf{v}^\top]^\top$ と置くと、

$$\begin{aligned} \begin{bmatrix} \mathbf{O} & \mathbf{Z} \\ \mathbf{Z}^\top & \mathbf{O} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \mathbf{v} \end{bmatrix} &= \begin{bmatrix} \lambda \mathbf{u} \\ \lambda \mathbf{v} \end{bmatrix} \\ \therefore \mathbf{Z}\mathbf{v} &= \lambda \mathbf{u}, \quad \mathbf{Z}^\top \mathbf{u} = \lambda \mathbf{v} \end{aligned} \quad (76)$$

\mathbf{u} または \mathbf{v} を消去すると、次式を得る。

$$\mathbf{Z}\mathbf{Z}^\top \mathbf{u} = \lambda^2 \mathbf{u}, \quad \mathbf{Z}^\top \mathbf{Z}\mathbf{v} = \lambda^2 \mathbf{v} \quad (77)$$

すなわち、 \mathbf{S} の固有ベクトル \mathbf{x} を構成する \mathbf{u} および \mathbf{v} は、それぞれ対称行列 $\mathbf{Z}\mathbf{Z}^\top \in \mathbb{R}^{m \times m}$ および $\mathbf{Z}^\top \mathbf{Z} \in \mathbb{R}^{n \times n}$ の固有ベクトルに他ならない。

ふたつの集合のクラスタリング \mathbf{S} の k 個の大きな固有値に属する固有ベクトル $\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(k)}$ を列に並べて、行列 $\mathbf{X}_k = [\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(k)}]$ を作成する。行列 \mathbf{X}_k の全 $m+n$ 行を k 次元空間の $m+n$ 点と見なすと、 k 箇所に固まってクラスタが形成されている。これは、2部グラフの節のふたつの集合 M と N の要素がまとめて k 個のクラスタに分類されたものである。集合 M の要素も N の要素も k 個のクラスタに分けられており、 M のクラスタと N のクラスタの対応も知ることができる。

例：式(73), (75)から得られる正規化類似度行列の固有値は

$$1.00, 0.92, 0.12, 0.03, 0.00, -0.03, -0.12, -0.92, -1.00$$

である。明らかに大きな固有値がふたつ存在し、それらに属する固有ベクトルを列に並べると

$$\mathbf{X}_2 = [\mathbf{x}^{(1)}, \mathbf{x}^{(2)}] = \begin{bmatrix} 0.32 & 0.31 \\ 0.31 & -0.38 \\ 0.31 & 0.22 \\ 0.32 & 0.24 \\ 0.32 & -0.40 \\ 0.38 & 0.34 \\ 0.34 & -0.39 \\ 0.37 & 0.33 \\ 0.32 & -0.36 \end{bmatrix} \quad \text{各行を正規化すると} \quad \bar{\mathbf{X}}_2 = \begin{bmatrix} 0.72 & 0.69 \\ 0.63 & -0.78 \\ 0.81 & 0.58 \\ 0.80 & 0.59 \\ 0.63 & -0.78 \\ 0.75 & 0.66 \\ 0.66 & -0.75 \\ 0.75 & 0.66 \\ 0.66 & -0.75 \end{bmatrix} \quad (78)$$

という行列を作れる。各行を $k=2$ 次元空間の点のデータと見なし、第1行から第5行の $m=5$ 点を集合 M の要素、第6行から第9行の $n=4$ 点を集合 N の要素としてプロットすると、図6のように2箇所に固まったクラスタが形成されていることがわかる。

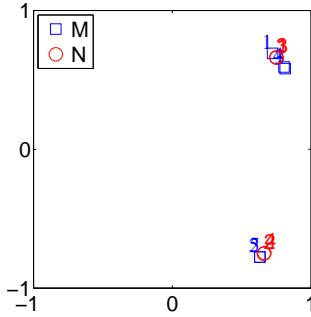


図 6 $\bar{\mathbf{X}}_2$ の 5 行が 2 次元空間でふたつの密なクラスタを呈す。右上のクラスタは集合 M の $\{1, 3, 4\}$ と集合 N の $\{1, 3\}$ からなる。右下のクラスタは集合 M の $\{2, 5\}$ と集合 N の $\{2, 4\}$ からなる。

特異値分解

Z のランクを $r := \text{rank } Z$ とし、式 (77) の非ゼロ固有値 λ_j^2 に属する固有ベクトル $\mathbf{u}^{(j)}$ と $\mathbf{v}^{(j)}$ ($j = 1, \dots, r$) は単位ベクトルであるとする。対称行列の固有ベクトルは互いに直交するので、

$$\mathbf{U}_r := [\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(r)}] \in \mathbb{R}^{m \times r}, \quad \mathbf{V}_r := [\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(r)}] \in \mathbb{R}^{n \times r} \quad (79)$$

という行列を定義すると、

$$\mathbf{U}_r^\top \mathbf{U}_r = \mathbf{V}_r^\top \mathbf{V}_r = \mathbf{I} \in \mathbb{R}^{r \times r} \quad (80)$$

が成立する。さらに、固有値からなる対角行列を $\mathbf{K}_r := \text{diag}(\lambda_1, \dots, \lambda_r)$ とすると、式 (76), (79), (80) から、次式が導出される（演習問題 1）。

$$Z \mathbf{V}_r = \mathbf{U}_r \mathbf{K}_r, \quad \therefore \quad \mathbf{U}_r^\top Z \mathbf{V}_r = \mathbf{K}_r \quad (81)$$

\mathbf{U}_r は m 次元実空間 \mathbb{R}^m の正規直交基底のうち r 本の基底ベクトルを並べた行列であると見なせる。 \mathbf{V}_r についても同様である。ゆえに、それぞれ残りの $m - r$ 本, $n - r$ 本の基底ベクトルを追加した行列 $\mathbf{U} \in \mathbb{R}^{m \times m}$ と $\mathbf{V} \in \mathbb{R}^{n \times n}$ を作成できる。これら \mathbf{U} と \mathbf{V} は直交行列であり、

$$\mathbf{U}^\top \mathbf{U} = \mathbf{I} \in \mathbb{R}^{m \times m}, \quad \mathbf{V}^\top \mathbf{V} = \mathbf{I} \in \mathbb{R}^{n \times n} \quad (82)$$

を満たす。ゆえに、

$$Z = \mathbf{U} \mathbf{K} \mathbf{V}^\top = \mathbf{U}_r \mathbf{K}_r \mathbf{V}_r^\top \quad (83)$$

が成立する。これは、行列 Z の特異値分解 (singular value decomposition; SVD) と呼ばれている。 $\lambda_1, \dots, \lambda_r$ は Z の特異値と呼ばれる。また、 $\mathbf{u}^{(1)}, \dots, \mathbf{u}^{(r)}$ と $\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(r)}$ は、それぞれ Z の左特異ベクトル、右特異ベクトルと呼ばれる。

特異値分解を用いると、Ncut による共クラスタリングはアルゴリズム 4 のような簡潔な手順になる。

演習問題

1. 式 (81) を導出せよ。また、特異値分解の式 (83) を導出せよ。

アルゴリズム 4 Ncut による共クラスタリング

入力： クラスタ数 k ,

m 個の項目と n 個の項目の間の関連性の強さを要素とする長方類似度行列 $\mathbf{C} \in \mathbb{R}^{m \times n}$;

出力： クラスタの集合 $\mathcal{M} = \{M_1, \dots, M_k\}$, $\mathcal{N} = \{N_1, \dots, N_k\}$

(ただし, $\sum_{l=1}^k |M_l| = m$, $\sum_{l=1}^k |N_l| = n$) ;

- 1 行和と列和による次数行列 $\mathbf{D}_2 = \text{diag}(\mathbf{C}\mathbf{1}_n)$ と $\mathbf{D}_1 = \text{diag}(\mathbf{C}^\top \mathbf{1}_m)$ を計算する;
 - 2 正規化した長方類似度行列 $\mathbf{Z} = \mathbf{D}_2^{-1/2} \mathbf{C} \mathbf{D}_1^{-1/2}$ の k 個の大きな特異値に属する左右の特異ベクトル $\mathbf{u}^{(l)}$ と $\mathbf{v}^{(l)}$ ($l = 1, \dots, k$) を計算する;
 - 3 $\mathbf{u}^{(l)}$ と $\mathbf{v}^{(l)}$ を結合して $\mathbf{X}_k = \begin{bmatrix} \mathbf{u}^{(1)} & \cdots & \mathbf{u}^{(k)} \\ \mathbf{v}^{(1)} & \cdots & \mathbf{v}^{(k)} \end{bmatrix} \in \mathbb{R}^{(m+n) \times k}$ を作る;
 - 4 \mathbf{X}_k の各行ベクトルを正規化する;
 - 5 \mathbf{X}_k の各行を k 個のビンにベクトル量子化して $\mathbf{y} \in [k]^{m+n}$ を作る：典型的には， \mathbf{X}_k の $m+n$ 本の行ベクトルを k 平均クラスタリングする；
 - 6 $y_i = l$ ($i = 1, \dots, m$) ならば項目 i を M_l に所属させる.
 - 7 $y_{m+j} = l$ ($j = 1, \dots, n$) ならば項目 j を N_l に所属させる.
-

参考文献

- [1] J. Shi and J. Malik, “Normalized cuts and image segmentation,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.22, no.8, pp.888–905, 2000.
- [2] S.X. Yu and J. Shi, “Multiclass spectral clustering,” International Conference on Computer Vision, pp.313–319, 2003.