Chapter 2

線形代数の基礎

本章の目的は，線形代数と呼ばれる数学の入門的な計算ルールや，基本的な性質を学ぶことにある．これらの知識は，後に述べる様々な多変量解析法の背後にある考え方を数学的に理解し，なおかつ，関連する手法との類似性や相違点を理解する上で必須のものとなる．なお，本章で扱う線形代数の内容は，大学の教養課程で学習するものとは大きく異なり，多変量解析法を理解するために必要最小限の内容に限定していることに注意する必要がある．線形代数に関するより網羅的なテキストとしては，Harville (1998)[1] などがある．

2.1 ベクトルと行列

「1」,「5」,「-4」,「0.8」などのような数値（**スカラー** (scalar) と呼ばれる）を複数並べ，一つのまとまりとして扱うことを考えよう．そのようなまとまりは**ベクトル** (vector) と呼ばれる．例えば，

$$\begin{bmatrix} 1 \\ 5 \\ -4 \\ 0.8 \end{bmatrix} \tag{2.1}$$

のように表記される．スカラーを縦方向に並べたベクトルは特に，列ベクトルと呼ばれる．一方，横方向に並べたベクトル

$$\begin{bmatrix} 1 & 5 & -4 & 0.8 \end{bmatrix} \tag{2.2}$$

を**行ベクトル** (row vector) と呼ぶ．また，ベクトルを構成するスカラーのそれぞれを，ベクトルの**要素** (element) と呼ぶ．ベクトルの要素の数は，そのベクトルの**次元数** (dimensionality) を表す．すなわち，上の二つのベクトルはともに，4 次元ベクトルである，という．この考え方を適用すれば，スカラーは 1 次元のベクトルである，とも言える．

さらに，同じ次元数のベクトルを複数束ねたものを，**行列** (matrix) と呼ぶ．例えば

$$\begin{bmatrix} 1 \\ 5 \\ -4 \\ 0.8 \end{bmatrix}, \begin{bmatrix} 2 \\ -7 \\ 3 \\ 4 \end{bmatrix} \tag{2.3}$$

という二つの 4 次元列ベクトルを横に束ねた

$$\begin{bmatrix} 1 & 2 \\ 5 & -7 \\ -4 & 3 \\ 0.8 & 4 \end{bmatrix} \tag{2.4}$$

がその一例である．ここで，行列の縦方向の広がりを**行** (row)，横方向の広がりを**列** (column) と呼び，行の数と列の数によって表される行列のサイズを，行列の次元数と呼ぶ．上の例であれば，行数は 4，列数は 3 であり，この行列の次元数は「4×3」と表される．このことに基づいて，上の行列は「4×3 の行列」と呼ばれる．なお，行列を構成する各スカラーを，ベクトルと同様に要素と呼ぶ．

ベクトルはスカラーの一般化であり，さらに行列はベクトルの一般化である．すなわち，行数または列数が 1 の行列は，行ベクトルまたは列ベクトルである．さらに，次元数が 1 のベクトルはスカラーであり，スカラーは 1×1 の行列である．言い換えれば，普段慣れ親しんでいるスカラーを拡張した概念が行列でありベクトルである．以降では，行列の和や積を計算するための演算規則やさまざまな性質を紹介するが，そのほとんどは，スカラーおよびベクトルの演算に対しても，その一般性を損なうことなく適用可能なものである．

本書では，行列を「$\mathbf{X}$」や「$\boldsymbol{\Psi}$」などの大文字のアルファベットまたはギリシャ文字で表記する．またベクトルを,「$\mathbf{x}$」や「$\boldsymbol{\theta}$」などの小文字のアルファベットまたはギリシャ文字で表記する．行列の $\mathbf{X}$ の i 番目の行かつ j 番目の列に位置する要素は第 (i, j) 要素と表記され，$\mathbf{X}$ 中の特定の要素の位置を示す i と j の組み合わせを，その要素の**インデックス** (index) と呼ぶことがある．さらに，この要素を「x_{ij}」と，行列のアルファベットまたはギリシャ文字の小文字のイタリック体に，インデックスの添え字を「行番号，列番号」の順で付した形で表記する．このことを，$\mathbf{X} = \{x_{ij}\}$ と表すことがある．ベクトルの場合も同様にして，ベクトル $\mathbf{y}$ の i 番目の要素は $\mathbf{y}$ の第 i 要素と呼ばれ，$\mathbf{y}_i$ と表記される．ベクトルの場合はインデックスが一つしかないため，下付き文字が一つしかつかないことに注意する．

行列は行ベクトルを縦に並べた，あるいは列ベクトルを横に束ねたものであるため，行列から行または列を取り出せば，行ベクトルか列ベクトルが得られる．そこで，特に断らない限り，$\mathbf{X}$ の第 i 行を取り出した行ベクトルを $\mathbf{x}_{(i)}$ と表す．また，第 j 列を取り出した列ベクトルを $\mathbf{x}_j$ と表す．つまり，ベクトルに括弧付きの下付き文字がつけば行ベクトル，括弧がつかなければ列ベクトルであると表現する．このことを利用すれば，$n \times p$ の $\mathbf{X}$ は以下のように表現できる．

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_{(1)} \\ \mathbf{x}_{(2)} \\ \vdots \\ \mathbf{x}_{(i)} \\ \vdots \\ \mathbf{x}_{(n)} \end{bmatrix} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_j & \cdots & \mathbf{x}_p \end{bmatrix} \tag{2.5}$$

2.2 行列の足し算・掛け算

行数と列数，すなわち次元数が一致する任意の二つの行列に対して，行列の足し算（**和** (sum)）を定義することができる．例えば，行列の和

$$\begin{bmatrix} 1 & 2 \\ 5 & -7 \\ -4 & 3 \\ 0.8 & 4 \end{bmatrix} + \begin{bmatrix} -3 & 10 \\ 4 & 1 \\ -0.2 & 2 \\ 6 & -1 \end{bmatrix} \tag{2.6}$$

や，ベクトルの和である

$$\begin{bmatrix} 1 \\ 5 \\ -4 \\ 0.8 \end{bmatrix} + \begin{bmatrix} 2 \\ -7 \\ 3 \\ 4 \end{bmatrix} \tag{2.7}$$

は定義できる．つまり計算可能である．しかし

$$\begin{bmatrix} 1 & 2 \\ 5 & -7 \\ -4 & 3 \\ 0.8 & 4 \end{bmatrix} + \begin{bmatrix} 1 \\ 5 \\ -4 \\ 0.8 \end{bmatrix} \tag{2.8}$$

や

$$\begin{bmatrix} 1 \\ 5 \\ -4 \\ 0.8 \end{bmatrix} + \begin{bmatrix} 2 \\ -7 \\ 3 \end{bmatrix} \tag{2.9}$$

定義不可能であり計算できない．

行列およびベクトルの和は，同じ行・列に位置する要素の足し算により定義される．例えば，(2.6) の結果は

$$\begin{bmatrix} 1 & 2 \\ 5 & -7 \\ -4 & 3 \\ 0.8 & 4 \end{bmatrix} + \begin{bmatrix} -3 & 10 \\ 4 & 1 \\ -0.2 & 2 \\ 6 & -1 \end{bmatrix} = \begin{bmatrix} 1-3 & 2+10 \\ 5+4 & -7+1 \\ -4-0.2 & 3+2 \\ 0.8+6 & 4-1 \end{bmatrix} = \begin{bmatrix} -2 & 12 \\ 9 & -6 \\ -4.2 & 5 \\ 6.8 & 3 \end{bmatrix} \tag{2.10}$$

と計算できる．より一般に，同じ $n \times p$ の行列 $\mathbf{X} = \{x_{ij}\}$ と $\mathbf{Y} = \{y_{ij}\}$（ただし，$i = 1, \cdots, n, j = 1, \cdots, p$）に対して，両者の和 $\mathbf{X} + \mathbf{Y}$ の第 (i, j) 要素は $x_{ij} + y_{ij}$ によって求められる．なお，この演算規則はベクトルの和に対しても適用される．

行列およびベクトルの和は非常に直感的に計算できるのに対し，その掛け算（**積** (product)）はやや複雑である．まず，行列 $\mathbf{X}$ と $\mathbf{Y}$ の積 $\mathbf{XY}$ が計算できるためには，$\mathbf{X}$ の列数と $\mathbf{Y}$ の行数が一致していなければならない．例えば，$\mathbf{X}$ と $\mathbf{Y}$ がそれぞれ $m \times n$ および $t \times u$ ならば，$n = t$ である必要がある．さらに，計算の結果出力される行列の次元数は $m \times u$ となる．すなわち「$\mathbf{X}$ の行数 $\times \mathbf{Y}$ の列数」の行列が出力される．また，$\mathbf{Z} = \mathbf{XY} = \{z_{ij}\}$ とすると，z_{ij} は次式で得られる

$$z_{ij} = \sum_{k}^{n} x_{ik} y_{kj}. \tag{2.11}$$

つまり，$\mathbf{X}$ の第 i 行ベクトル $\mathbf{x}_{(i)}$ と $\mathbf{Y}$ の第 j 列ベクトル $\mathbf{Y}_j$ を用意し，それらの対応する要素の掛け算を全て足し合わせたものが，z_{ij} である．この「対応する要素の掛け算」を行うためには，$\mathbf{x}_{(i)}$ と $\mathbf{Y}_j$ の次元数が一致している必要があり，前者は $\mathbf{X}$ の行数を，後者は $\mathbf{Y}$ の列数をそれぞれ表す．そのため，行列の掛け算 $\mathbf{XY}$ の計算では，$\mathbf{X}$ の行数と $\mathbf{Y}$ の列数が一致している必要がある．

行列の掛け算の計算例を以下に示す．

$$\begin{aligned} \begin{bmatrix} 1 & 2 \\ 5 & -7 \\ -4 & 3 \end{bmatrix} \begin{bmatrix} -3 & 4 & -2 \\ 1 & 5 & -1 \end{bmatrix} &= \begin{bmatrix} 1 \times (-3) + 2 \times 1 & 1 \times 4 + 2 \times 5 & 1 \times (-2) + 2 \times (-1) \\ 5 \times (-3) - 7 \times 1 & 5 \times 4 - 7 \times 5 & 5 \times (-2) - 7 \times (-1) \\ -4 \times (-3) + 3 \times 1 & -4 \times 4 + 3 \times 5 & -4 \times (-2) + 3 \times (-1) \end{bmatrix} \\ &= \begin{bmatrix} -1 & 14 & -4 \\ -22 & -15 & -3 \\ 15 & -1 & 5 \end{bmatrix} \end{aligned} \tag{2.12}$$

この計算例では，3×2 に 2×3 の行列をかけているので，行数と列数が一致し，掛け算を計算することができる．ただし，それらが一致しない

$$\begin{bmatrix} 1 & 2 \\ 5 & -7 \\ -4 & 3 \end{bmatrix} \begin{bmatrix} -3 & 4 & -2 \\ 1 & 5 & -1 \\ 2 & 4 & 5 \end{bmatrix} \tag{2.13}$$

は計算できない．

以上の点から，行列の掛け算について，以下の重要な性質が導かれる．

- $\mathbf{XY}$ が定義できても，両者を入れ替えた $\mathbf{YX}$ が定義できるとは限らない．
- 一般に，$\mathbf{XY} \neq \mathbf{YX}$ である．

この二つの性質のため，行列の掛け算では，掛けられる行列 ($\mathbf{XY}$ の $\mathbf{X}$) と掛ける行列 ($\mathbf{XY}$ の $\mathbf{Y}$) を区別する必要がある．また，$\mathbf{XY}$ のことを「$\mathbf{X}$ の後ろ（または右）から $\mathbf{Y}$ を掛ける ($\mathbf{X}$ is post/right-multiplied by $\mathbf{Y}$)」や，逆に「$\mathbf{Y}$ の前（または左）から $\mathbf{X}$ を掛ける ($\mathbf{Y}$ is pre/left-multiplied by $\mathbf{X}$)」などという．また，$\mathbf{XY} = \mathbf{YX}$ が成立する場合，$\mathbf{X}$ と $\mathbf{Y}$ は**可換** (commutative) であるという．スカラーの場合は $2 \times 3 = 3 \times 2$ のように，この可換性が常に成り立つが，行列の場合はそうではなく，むしろ一部の行列だけが持つ特殊な性質であることに注意する．

ベクトルの掛け算も同様にして計算することができる．計算を定義可能にするためには，掛けられる側の行列の列数と，掛ける側の行列の行数が一致する必要があるから，行ベクトルに列ベクトルを掛けることしかできない．また，掛け算の結果は 1×1 の行列，つまりスカラーである．例えば

$$\begin{bmatrix} 1 & 5 & -4 & 8 \end{bmatrix} \begin{bmatrix} 2 \\ -1 \\ 3 \\ 2 \end{bmatrix} = 1 \times 2 + 5 \times (-1) + (-4) \times 3 + 8 \times 2 = 1 \tag{2.14}$$

のように計算できる．ベクトル同士の掛け算は，特に，ベクトルの**内積** (inner product) と呼ばれる．ある列ベクトルの行と列を入れ替えた行ベクトルと，元の列ベクトルの内積は，常に定義可能である．例えば

$$\mathbf{x} = \begin{bmatrix} 1 \\ 2 \\ -4 \\ 3 \end{bmatrix} \tag{2.15}$$

という列ベクトルと，その列ベクトルの行と列を入れ替えた行ベクトル

$$\mathbf{x}' = \begin{bmatrix} 1 & 2 & -4 & 3 \end{bmatrix} \tag{2.16}$$

に対して（このように行列やベクトルの行と列を入れ替えたものを**転置** (transpose) と呼び，のちに詳述するが，行列やベクトルの右上に「プライム：′」をつけて表現する），両者の掛け算は

$$\mathbf{x}'\mathbf{x} = \begin{bmatrix} 1 & 2 & -4 & 3 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ -4 \\ 3 \end{bmatrix} = 30 \tag{2.17}$$

のように計算できる．この内積は，$\mathbf{x}$ の要素の二乗和に等しい．さらにこの内積の平方根をとったものは，ベクトルの**長さ** (length) と呼ばれる．

2.3 行列の転置

ある行列やベクトルの行と列を入れ替えたものを，その行列の転置と呼ぶ．転置された行列は，元の行列の右上に′を付して表記する．例えば，(2.4) の 3×2 の $\mathbf{X}$ の転置は $\mathbf{X}'$ と表記され

$$\mathbf{X}' = \begin{bmatrix} 1 & 5 & -4 & 0.8 \\ 2 & -7 & 3 & 4 \end{bmatrix} \tag{2.18}$$

と表現される．

2.4 特殊な行列

行列の基本的な性質および演算規則を導入した上で，それらに基づいて，様々な特殊な行列を紹介する．

2.4.1

正方行列行数と列数が等しい行列を**正方行列** (square matrix) と呼ぶ．例えば，3×3 や，10×10 の次元数を持つ行列が正方行列である．また，あらゆるスカラーは 1×1 の行列なので，正方行列でもある．さらに，同じ次元数を持つ正方行列同士の掛け算は，掛ける側・掛けられる側を入れ替えても常に定義可能である．しかしその結果は一致しない（可換ではない）．

2.4.2 対称行列

正方行列であり，なおかつ，(i, j) 要素と (j, i) 要素が全ての $i, j (i \neq j)$ について等しい行列を，対称行列と呼ぶ．正方行列において，行番号と列番号が等しい要素のことを**対角要素** (diagonal elements) と呼び，それに対して，対角要素ではない要素のことを**非対角要素** (non-diagonal elements) とよぶ．つまり，対称行列は非対角要素が対角要素を境にして対称であるような行列である．例えば

$$\begin{bmatrix} 3 & 2 & 4 \\ 2 & 1 & 5 \\ 4 & 5 & -2 \end{bmatrix} \tag{2.19}$$

は対称行列である．任意の対称行列 $\mathbf{X}$ に対して

$$\mathbf{X} = \mathbf{X}' \tag{2.20}$$

が成り立つ．対称行列の例として，p 個の変数に対して，(i, j) 要素が i 番目と j 番目の変数の共分散，(i, i) 要素が i 番目の要素の分散となっているような行列は，**分散共分散行列** (covariance matrix)（あるいは，単に共分散行列）と呼ばれる．任意の (i, j) のペアに関して，i 番目と j 番目の変数の共分散と，それを入れ替えた，j 番目と i 番目の変数の共分散は等しいので，非対角要素が等しく，よって分散共分散行列は対称行列である．

2.4.3 対角行列

非対角要素が全て 0 である行列を**対角行列** (diagonal matrix) と呼ぶ．例えば

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -2 & 0 \\ 0 & 0 & 3 \end{bmatrix} \tag{2.21}$$

は対角行列である．対角行列をある行列の左側からかけることは，右側の行列の i 番目の列の要素全てを，左側の行列（対角行列）の i 番目の対角要素倍する計算を意味する．逆に，対角行列をある行列の右側からかけると，左側の行列の j 番目の列が対角要素の j 番目の要素倍される．例えば，次のような計算が成り立つ．

このような性質は，データ行列の特定の変数を 2 倍する（例えばあるテストの点数を 2 倍して評価する）ような単純な計算や，ある変数をその標準偏差で割ることにより，割った後の変数の標準偏差が 1 になるようにするなどの計算に用いられる．

2.4.4 単位行列

対角要素が全て 1 であるような対角行列を，**単位行列** (identity matrix) と呼ぶ．単位行列は，スカラーにおける「1」に相当するものであり，任意の行列 $\mathbf{X}$ の右からかけて左からかけても，その結果は $\mathbf{X}$ に等しい．つまり

$$\mathbf{XI} = \mathbf{IX} = \mathbf{X} \tag{2.22}$$

であり，従って可換である．$p \times p$ の単位行列を「p 次の単位行列」と呼び，$\mathbf{I}_p$ と次数の添字をつけて表す．ただし，添え字がない場合の $\mathbf{I}$ は，和や積が定義可能な適当な次数の単位行列を表すものとする．

2.5 線形代数の利用例 1：データ行列の標準化

ここまで紹介した様々な行列やその演算規則を用いて，データの代表的な前処理法である，**標準化** (standardization) を実行してみよう．標準化とは，個体 × 変数のデータ行列（例えば 100 名の学生の 5 教科の試験成績を記録したデータであれば，100×5 のデータ行列）が与えられた時に，各変数に関して，その平均が 0，分散が 1 になるようにデータを変換することを表す．異なる平均や分散を持った変数に対してこの変換を施すことにより，変換後の変数は相互に比較することが可能になる．

標準化は，以下の二つの計算処理によって構成される．

- データ中の各数値から変数の平均を引く（**中心化** (centering)）
- データ中の各数値を変数の標準偏差で割る（**スケーリング** (scaling)）

まず，この処理のそれぞれをスカラーで表現してみよう．n 個体 $\times p$ 変数のデータ行列を $\mathbf{X} = \{x_{ij}\}$ としよう．さらに，第 j 変数の平均値を $\bar{x}_j = n^{-1} \sum_i x_{ij}$ と，標準偏差を σ_j とそれぞれ表す．この時，1 つ目の処理は

$$x_{ij} \leftarrow x_{ij} - \bar{x}_j \tag{2.23}$$

で表される．ここで「$\leftarrow$」は，右辺の結果で左辺を上書きすることを意味する．さらに，2 番目の処理は

$$x_{ij} \leftarrow x_{ij} / \sigma_j \tag{2.24}$$

と表される．

さて，まず (2.23) の中心化を線形代数で表現し直すことを考えよう．中心化は，データ行列の各要素から，各列の平均値を引く計算なので，$\mathbf{X}$ と同サイズで，第 j 列が全て $\bar{x}_j$ であるような行列 $\bar{\mathbf{X}}$ を用意して，$\mathbf{X} \leftarrow \mathbf{X} - \bar{\mathbf{X}}$ とすれば良い．さらに，$\bar{\mathbf{X}}$ は次のような演算によって得ることができる．

$$\bar{\mathbf{X}} = \frac{1}{n} \begin{bmatrix} 1 & \cdots & 1 \\ & \vdots & \\ 1 & \cdots & 1 \end{bmatrix} \mathbf{X} = \frac{1}{n} \mathbf{1}_n \mathbf{1}_n' \mathbf{X} \tag{2.25}$$

ここで $\mathbf{1}_n$ は要素が全て 1 の n 次元列ベクトルである．上式を用いると，中心化は

$$\begin{aligned} \mathbf{X} \leftarrow \mathbf{X} - \bar{\mathbf{X}} \quad &= \mathbf{X} - \tfrac{1}{n} \mathbf{1}_n \mathbf{1}_n' \mathbf{X} \\ &= \left(\mathbf{I}_n - \tfrac{1}{n} \mathbf{1}_n \mathbf{1}_n' \right) \mathbf{X} \\ &= \mathbf{J}_n \mathbf{X} \end{aligned} \tag{2.26}$$

と，データ行列の左から $\mathbf{J}_n = \mathbf{I}_n - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n'$ をかけるという非常に単純な演算として表現できる．ここで $\mathbf{J}_n$ は**中心化行列** (centering matrix) と呼ばれる．$\mathbf{J}_n$ は n さえ与えれば自動的に決定される．

次に，(2.24) のスケーリングの演算を線形代数で表現しよう．分散共分散行列は，j 番目の変数の分散が j 番目の対角要素に，j 番目と k 番目 $(j \neq k)$ の共分散が (j,k) 要素に配置される対称行列である．中心化したデータ行列を $\mathbf{X}_c = \mathbf{J}_n\mathbf{X}$ とすると，$\mathbf{X}$ の分散共分散行列 $\mathbf{S}_\mathbf{X}$ は次式で得られる．

$$\mathbf{S}_\mathbf{X} = \frac{1}{n}\mathbf{X}_c'\mathbf{X}_c = \frac{1}{n}\mathbf{X}_c'\mathbf{J}_n\mathbf{J}_n\mathbf{X}_c = \frac{1}{n}\mathbf{X}_c'\mathbf{J}_n\mathbf{X}_c \tag{2.27}$$

上式では，$\mathbf{J}_n$ が対称行列であること（$\mathbf{J}_n = \mathbf{J}_n'$）と冪等（$\mathbf{J}_n\mathbf{J}_n = \mathbf{J}_n$）であることを用いた．$\mathbf{S}_\mathbf{X}$ の非対角要素を 0 と置いた対角行列 $\mathrm{diag}(\mathbf{S}_\mathbf{X}) = \mathbf{D}$ を用いて変数の標準偏差の逆数，つまり分散の正の平方根の逆数が並んだ対角行列を

$$\mathbf{D}_s = \mathbf{D}^{-1/2} \tag{2.28}$$

として求める．ここで，$\mathbf{D}^{-1/2}$ は $\mathbf{D}$ の対角要素を $-1/2$ 乗することを表す．スケーリングでは，データ行列の各列を標準偏差で割る．つまり，各列を分散の $-1/2$ 乗倍すれば良い．そのような演算は，データ行列の右側から $\mathbf{D}_s$ をかけることによって実現できる．すなわちスケーリングは

$$\mathbf{X} \leftarrow \mathbf{X}\mathbf{D}_s \tag{2.29}$$

によって実現できる．

以上をまとめると，標準化は，(2.26) と (2.29) を組み合わせて

$$\mathbf{X} \leftarrow \mathbf{J}_n\mathbf{X}\mathbf{D}_s \tag{2.30}$$

という計算として定義できる．

2.6 行列の階数

$p \times a(p \geq q)$ の行列 $\mathbf{X}$ に関して，他の列ベクトルの定数倍，もしくは定数倍後の和によって表すことのできない列ベクトルを，**一次独立** (linearly independent) な列ベクトルと呼ぶ．そうでない場合，**一次従属** (linearly dependent) な列ベクトルであると呼ぶ．例えば

$$\mathbf{X} = \begin{bmatrix} 3 & -4 & 6 \\ 2 & -2 & 4 \\ 4 & -5 & 8 \\ -1 & 1 & -2 \\ 0 & -5 & 0 \end{bmatrix} \tag{2.31}$$

の時，第 3 列ベクトルは第 1 列ベクトルの 2 倍になっている．しかし，第 2 列ベクトルは第 1 および第 3 列ベクトルの定数倍もしくはその和によって表現することはできない．よって，$\mathbf{X}$ の第 2 列は一次独立だが，それ以外の列ベクトルは一次従属である．

同様の議論は，行ベクトルについても成り立つ．(2.31) の第 3 行は，第 1 行と第 2 行を 0.5 倍したものの和として表現できるが，第 4，5 行はそのように表すことはできない．従って，第 4，5 行ベクトルは一次独立であり，その他の行ベクトルは一次従属である．

任意の行列に関して，一次独立な列ベクトルの本数を，その行列の**列階数** (column rank) もしくは**列ランク**と呼ぶ．また，列ランクがその行列の列数に等しい時，つまり，行列の全ての列が一次独立の場合，その行列は**フル列ランク** (full column-rank) であると呼ばれる．行ベクトルに関しても同様に，一次独立な行ベクトルの本数を，その行列の**行階数** (row rank) もしくは**行ランク**と呼び，行ランクが行数に等しいとき，その行列は**フル行ランク** (full row-rank) と呼ばれる．さらに，行ランクもしくは列ランクのどちらか小さい方を，その行列の**階数** (rank) もしくは**ランク**と呼び，$\mathrm{rank}(\bullet)$ という関数で表す．例えば $\mathbf{X}$ の列ランクは 2（第 3 列は一次従属なので，$3-1$），行ランクは 4（第 3 行は一次従属なので $5-1$）であるから，$\mathrm{rank}(\mathbf{X}) = 2$ である．

ランクに関して，明らかに次式が成り立つ．

$$\mathrm{rank}(\mathbf{X}) \leq min(p, q) \tag{2.32}$$

上の等号が成り立つ時，$\mathbf{X}$ は**フルランク** (full rank) であるという．そうでない場合，$\mathbf{X}$ はランクが**落ちている** (rank deficient)，と表現される．

行列のランクの直感的な意味としては「その行列に含まれるユニークな情報の量」であると考えられる．ある行ないし列ベクトルが一次従属であるということは，その行ないし列ベクトルが，他の行ないし列ベクトルの定数倍または和で表現可能であるということであり，そのように表すことのできるベクトルは，いわば，他の行列と「被った」情報を持っていると捉えられる．このことから，フル行ランクまたはフル列ランクの行列は，そうでない行列と比べて，情報が被ったベクトルが少ない，ユニークな情報を多く持ったベクトルであると捉えることができよう．

2.7 逆行列

$p \times p$ 正方行列 $\mathbf{A}$ に対して，次式を満たす $p \times p$ 正方行列 $\mathbf{A}^{-1}$ を，$\mathbf{A}$ の**逆行列**と呼ぶ．

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_p \tag{2.33}$$

逆行列は，スカラーにおける逆数に相当するものであり，元の行列の右または左から掛けると，単位行列が得られる．

逆行列の性質として，以下のものがある．

まず，逆行列の逆行列は元の行列である．

$$(\mathbf{A}^{-1})^{-1} = \mathbf{A} \tag{2.34}$$

このことは次の通り示すことができる．まず，逆行列の定義より，次式が成り立つ．

$$(\mathbf{A}^{-1})^{-1}(\mathbf{A}^{-1}) = \mathbf{I}, (\mathbf{A}^{-1})(\mathbf{A}^{-1})^{-1} = \mathbf{I} \tag{2.35}$$

(2.33) より (2.34) を得る．

次に，逆行列の転置は，転置の逆行列である．

$$(\mathbf{A}')^{-1} = (\mathbf{A}^{-1})' \tag{2.36}$$

行列の転置に関する性質から

$$(\mathbf{A}\mathbf{A}^{-1})' = \mathbf{I} \leftrightarrow (\mathbf{A}^{-1})'\mathbf{A}' = \mathbf{I} \tag{2.37}$$

と

$$(\mathbf{A}^{-1}\mathbf{A})' = \mathbf{I} \leftrightarrow \mathbf{A}'(\mathbf{A}^{-1})' = \mathbf{I} \tag{2.38}$$

が成り立つ．これらから，(2.36) が成り立つ．

最後に，行列の積の逆行列は，積の順番を入れ替えた逆行列の積である．

$$(\mathbf{A}\mathbf{B})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1} \tag{2.39}$$

このことは

$$(\mathbf{A}\mathbf{B})(\mathbf{B}^{-1}\mathbf{A}^{-1}) = \mathbf{A}\mathbf{A}^{-1} = \mathbf{I} \tag{2.40}$$

と

$$(\mathbf{B}^{-1}\mathbf{A}^{-1})(\mathbf{A}\mathbf{B}) = \mathbf{B}^{-1}\mathbf{B} = \mathbf{I} \tag{2.41}$$

によって示される．

しかしながら，(2.33) を満たす逆行列が常に存在するとは限らない．行列 $\mathbf{A}$ に対して，その逆行列が存在するための必要十分条件は

- $\mathbf{A}$ が正方行列であること
- $\mathbf{A}$ がフルランクであること

である．上の条件のどちらかが満たされない場合，その行列の逆行列は存在しない．

2.7.1 連立方程式を解く

3 つの未知数 x, y, z を含む 3 元 1 次方程式

$$5x - 4y + 6z = 8 \tag{2.42}$$
$$7x - 6y + 10z = 14 \tag{2.43}$$
$$4x + 9y + 7z = 74 \tag{2.44}$$

を同時に満たす x, y, z を求めることを考えよう．そのような未知数の組は，以下に示すとおり，行列の逆行列を用いれば簡単に求めることができる．連立方程式の係数を並べた 3×3 行列 $\mathbf{A}$，未知数を並べた 3 次元ベクトル，連立方程式の右辺を並べた 3 次元次元ベクトルを用いると，(2.44) は

$$\begin{bmatrix} 5 & -4 & 6 \\ 7 & -6 & 10 \\ 4 & 9 & 7 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 8 \\ 14 \\ 74 \end{bmatrix} \tag{2.45}$$
$$\mathbf{Ax} = \mathbf{b} \tag{2.46}$$

と表現できる．逆行列 $\mathbf{A}^{-1}$ をこの式の両辺の左からかけると

$$\mathbf{A}^{-1}\mathbf{Ax} = \mathbf{A}^{-1}\mathbf{b} \tag{2.47}$$
$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \tag{2.48}$$

が得られる．

しかしながら，連立方程式の中には，解くことのできないもの，より正確には，解が一意に定まらず，方程式を同時に満たす未知数が複数あるものが存在する．例えば，次の 3 元 1 次連立方程式

$$5x - 4y + 6z = 8 \tag{2.49}$$
$$7x - 6y + 10z = 14 \tag{2.50}$$
$$10x - 8y + 12z = 16 \tag{2.51}$$

は解くことができない．一般に，未知数の数よりも方程式の数が少ない場合，その連立方程式を同時に満たす未知数を一意に定めることができないことが知られている．ここで，(2.51) の 3 つ目の方程式は，1 つ目の方程式の両辺を 2 倍したものである．つまり，1 つ目と 3 つ目の方程式は同じものを表しており，(2.51) には 2 つの異なる方程式しか存在しない．よって未知数の数 3 に対して方程式が 2 つしか存在しないので，この連立方程式を解くことはできない．この現象は，直感的には「3 つの未知数に対して，それらを特定するに十分な情報の量（すなわち方程式）が足りないので，未知数を求めることができない」と理解できよう．

このことを逆行列の存在という観点から説明しよう．まず，(2.51) を

$$\begin{bmatrix} 5 & -4 & 6 \\ 7 & -6 & 10 \\ 10 & -8 & 12 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 8 \\ 14 \\ 16 \end{bmatrix} \tag{2.52}$$
$$\mathbf{Ax} = \mathbf{b} \tag{2.53}$$

のように行列で表記する．ここで，$\mathbf{A}$ の第 1 行と第 3 行に注目すると，第 3 行は第 1 行の 2 倍になっていることがわかる．従って，$\mathbf{A}$ の行ランクは 2 であり，$\mathrm{rank}(\mathbf{A}) = 2$ である．従って $\mathbf{A}$ はフルランクの行列ではないため，その逆行列は存在しない．従って，(2.48) のように $\mathbf{A}$ の両辺に $\mathbf{A}^{-1}$ をかけることができないため，この連立方程式を解くことができない．

2.8 正規直交行列

$p \times q (p \geq q)$ 行列 $\mathbf{T}$ が

$$\mathbf{T}'\mathbf{T} = \mathbf{I}_q \tag{2.54}$$

を満たすとき，$\mathbf{T}$ は**列正規直交行列** (column orthonormal matrix) であるという．この時，$\mathbf{T}$ の第 j 列ベクトル $\mathbf{t}_j$ に関して,$\mathbf{t}_j'\mathbf{t}_j = 1$，つまり列ベクトルの長さが 1 である．また，別の第 $k(\neq j)$ 列ベクトルに関して，$\mathbf{t}_j'\mathbf{t}_k = 0$ が成り立つが，このことを，二つのベクトルが**直交する** (orthogonal) という．

同様の議論は行についても成り立つ．すなわち $p \leq q$ の時，

$$\mathbf{T}\mathbf{T}' = \mathbf{I}_p \tag{2.55}$$

が成り立つ時，$\mathbf{T}$ は**行正規直交行列** (row orthonormal matrix) であるという．

$\mathbf{T}$ が正方であり，なおかつ，(2.54) と (2.55) を同時に満たすとき，$\mathbf{T}$ は単に**正規直交行列** (orthonormal matrix) と呼ばれる．

2.9 特異値分解と固有値分解

$p \times q (p \geq q)$ のフルランクの行列 $\mathbf{X}$ は

- $\mathbf{U}'\mathbf{U} = \mathbf{I}_q$ を満たす $p \times q$ 列正規直交行列 $\mathbf{U}$
- $\mathbf{V}'\mathbf{V} = \mathbf{V}\mathbf{V}' = \mathbf{I}_q$ を満たす $q \times q$ 列正規直交行列 $\mathbf{V}$
- 0 以上の対角要素が大きい順に並ぶ $q \times q$ 対角行列 $\mathbf{D}$

を用いて

$$\mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{V}' \tag{2.56}$$

と表現する，言い換えれば，3 つの行列の積に分解することができる．この分解を，行列の**特異値分解** (singular value decomposition) と呼ぶ．$\mathbf{U}$ の各列は $\mathbf{X}$ の**左特異ベクトル** (left singular vector) と，$\mathbf{V}$ の各列は $\mathbf{X}$ の**右特異ベクトル** (right singular vector) と呼ばれる．また，$\mathbf{D}$ の対角要素 $d_1, \cdots, d_q \geq 0$ は，$\mathbf{D}$ の**特異値** (singular value) と呼ばれる．上式は

$$\mathbf{X} = \sum_{j=1}^{q} d_j \mathbf{u}_j \mathbf{v}_j' \tag{2.57}$$

と書き換えられるが，$\mathbf{u}_j$ と $\mathbf{v}_j$ を，第 j 特異値 d_j に対応する左・右特異ベクトルとそれぞれ呼ぶ．

$\mathbf{X}$ がフルランクでない場合も包含するように特異値分解の定義を広げてみよう．$\mathrm{rank}(\mathbf{X}) = r \leq q$ であるとして，$\mathbf{X}$ の特異値分解は

$$\begin{aligned} \mathbf{X} &= \mathbf{U}\mathbf{D}\mathbf{V}' \\ &= \begin{bmatrix} \tilde{\mathbf{U}} & \mathbf{U}_\perp \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{D}} & \\ & {}_{(q-r)}\mathbf{O}_{(q-r)} \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{V}} & \mathbf{V}_\perp \end{bmatrix}' \end{aligned} \tag{2.58}$$

と定義される．ここで，$\tilde{\mathbf{U}}(p \times r)$ と $\mathbf{U}_\perp(p \times (q-r)$ の列正規直交行列であり，$\mathbf{U}$ が列正規直交行列であることから

$$\tilde{\mathbf{U}}'\mathbf{U}_\perp = {}_r\mathbf{O}_{(q-r)} \tag{2.59}$$

が成り立つ．ただし，${}_r\mathbf{O}_{(q-r)}$ は要素が全て 0 の $r \times (q-r)$ 行列である．また，$\tilde{\mathbf{V}}(q \times r)$ と $\mathbf{V}_\perp(q \times (q-r)$ の列正規直交行列であり，上と同様に

$$\tilde{\mathbf{V}}'\mathbf{V}_\perp = {}_r\mathbf{O}_{(q-r)} \tag{2.60}$$

が成り立つ．(2.4) で重要なことは，特異値分解によって得られる，0 よりも大きい特異値の数は，$\mathbf{X}$ のランク数に等しいことである．従って，特異値分解によって 0 よりも大きい特異値の数を調べれば，その行列のランクを得ることができる．

(2.56) を用いると

$$\mathbf{X}'\mathbf{X} = \mathbf{V}\mathbf{D}\mathbf{U}'\mathbf{U}\mathbf{D}\mathbf{V}' = \mathbf{V}\mathbf{D}^2\mathbf{V}' \tag{2.61}$$

が得られる．ただし，$\mathbf{D}^2$ は $\mathbf{D}$ の対角要素を平方した対角行列である．これは，行列 $\mathbf{X}'\mathbf{X}$ の**固有値分解** (eigenvalue decomposition) と呼ばれる．この分解において，$\mathbf{V}$ の各列は**固有ベクトル** (eigenvector) と呼ばれ，$\mathbf{D}^2$ の対角要素は $\mathbf{X}'\mathbf{X}$ の**固有値** (eigenvalue) と呼ばれる．特異値分解が，あらゆる次元数の行列に対して定義可能な一方，固有値分解が可能なのは，上式が表すように，その行列が対称行列である場合に限られる．また，$\mathbf{X}$ の特異値分解で得られる右特異ベクトルは，$\mathbf{X}'\mathbf{X}$ の固有ベクトルでもあり，特異値の二乗が固有値である．従って，特異値分解が固有ベクトルを包含する関係にある．

2.10 トレースとノルム

次式で定義される値を，正方行列 $\mathbf{X}$ の**トレース** (trace) と呼ぶ．

$$\mathrm{tr}\mathbf{X} = \sum_i x_{ii} \tag{2.62}$$

つまり，対角行列の和である．また，次式で定義される値を，$p \times q$ 行列 $\mathbf{A}$ の**平方ノルム** (squared norm) と呼ぶ．

$$||\mathbf{A}||^2 = \sum_{i,j} a_{ij}^2 \tag{2.63}$$

トレースと平方ノルムに関して，以下の性質が成り立つ．

まず，明らかに，ある行列のトレースは，その行列の転置のトレースに等しい．$n \times n$ 正方行列について

$$\mathrm{tr}\mathbf{A} = \mathrm{tr}\mathbf{A}' \tag{2.64}$$

が成り立つ．また，トレースは行列の積を入れ替えても不変である（ただし，積が定義できる場合に限る）．$p \times p$ の正方行列 $\mathbf{B}$ を用いて

$$\mathrm{tr}\mathbf{A}\mathbf{B} = \mathrm{tr}\mathbf{B}\mathbf{A} \tag{2.65}$$

が成り立つ．このことは (2.64) を用いて

$$\mathrm{tr}\mathbf{A}\mathbf{B} = \mathrm{tr}(\mathbf{A}\mathbf{B})' = \mathrm{tr}\mathbf{B}'\mathbf{A}' = \sum_j \mathbf{a}_{(j)}\mathbf{b}_j \tag{2.66}$$

と表せることにより示される．

$p \times q$ の行列 $\mathbf{X}$ の平方ノルムはトレースを用いて

$$||\mathbf{X}||^2 = \mathrm{tr}\mathbf{X}'\mathbf{X} \tag{2.67}$$

と展開できる．

2.11 行列関数の最適化

2.11.1 行列関数

行列を引数として入力し，スカラーを出力する関数を，**行列関数** (matrix function) と呼ぶ．行列関数は，平方ノルムにより

$$f(\mathbf{W}) = ||\mathbf{XW}||^2 \tag{2.68}$$

などがその例として挙げられる．ここで，$\mathbf{W}$ は $q \times r$ の行列である．また，引数は複数の行列であってもよく

$$g(\mathbf{P}, \mathbf{Q}) = \mathrm{tr}\mathbf{PXQ} \tag{2.69}$$

も行列関数である．ここで，$\mathbf{P}$ は $m \times n$，$\mathbf{Q}$ は $p \times m$ の行列である．

2.11.2 最適化

行列関数を最小，または最大にするような引数の行列の具体値を求めることを，**行列関数の最適化** (optimization of matrix function) と呼ぶ．また，最適化の対象となる関数のことを**目的関数** (objective function) と呼ぶ．行列関数の最適化では，引数の行列のことを**パラメータ** (parameter) と呼ぶ．パラメータに対して，その値について何らかの条件が課されることが多い．そのような条件のことを**制約条件** (constraint) と呼ぶ．

例えば，(2.67) を，$\mathbf{W}$ が列正規直交行列であるという制約条件のもとで最小化することを考えよう．このことは，**最小化問題** (minimization problem) と呼ばれ（逆に，最大化する場合は**最大化問題** (maximization problem) と呼ぶ），フォーマルには

$$min.\ f(\mathbf{W}) = ||\mathbf{XW}||^2\ over\ \mathbf{W}\ s.t.\ \mathbf{W}'\mathbf{W} = \mathbf{I}_r \tag{2.70}$$

と表現される．最大化問題の時は「$min.$」が「$Max.$」に変わる．また，$s.t.$ は subject to の略である．さらに，目的関数を最小もしくは最大にするパラメータ行列の具体的な値を，**解** (solution) と呼ぶ．

2.11.3 多変量データ解析の定式化

次章以降紹介する様々な多変量データ解析法の多くは，解析法特有の行列関数を最適化するようにパラメータ行列を求める行為，として定義される．この定義は，上の最小化問題ないし最大化問題と同義であり，そのような問題の定義のことを，**数学的定式化** (mathematical formulation) と呼ぶ．例えば，ある多変量データ解析法が，(2.70) の最小化問題によって定式化されるとしよう（実はこれは，主成分分析の定式化の一種である）．この定式化は，一見難しいように思えるが，次のようにわかりやすく再表現することができる．

1. （多変量データ解析の目的）多変量データ解析法に対して，データ行列 $\mathbf{X}$ を入力した時に，何らかのロジックによって計算を行って，パラメータ行列 $\mathbf{W}$ の推定値（要素の値が具体的に決まったもの）を出力したい．
2. （目的関数の定義）その計算ロジックにはいろいろなものが考えられるが，ここでは $f(\mathbf{W}) = ||\mathbf{XW}||^2$ を最小化するように $\mathbf{W}$ を計算することにする．
3. （制約条件）ただし，$\mathbf{W}$ はどのような $\mathbf{W}$ でも良いわけではなく，$\mathbf{W}'\mathbf{W} = \mathbf{I}_r$ を守って欲しい．

2 では，目的関数の最小化によって $\mathbf{W}$ の解が決まるということを表しているが，このことは言い換えると，あらゆる可能な $\mathbf{W}$ の中で，$f(\mathbf{W})$ を最小化する $\mathbf{W}$ が「最も良い」と決めていることを意味する．従って,「最も良い」$\mathbf{W}$ を定めるための基準には，他の基準もあり得ることになる．

例えば，別の目的関数を設定しても良いし，データ行列 $\mathbf{X}$ を無視して，要素が全て 1 の $\mathbf{W}$ が「最も良い」と定めるような基準（とはもはや呼べないが）でも良い．また，同じ目的を設定している多変量データ解析法であっても，別の目的関数を設定することにより，複数の解が存在する場合もある．

2.11.4 微分による最小化

それでは，行列関数の最適化のためには，どのような方法が考えられるだろうか．関数の極値を求めるために一般的に用いられるのが，微分を用いた方法である．まず，最適化したい関数を，パラメータに関して微分（もしくは偏微分）する．関数が極値をとる時，微分，すなわち関数の傾きが 0 になるため，関数の極値を与えるパラメータは，微分を 0 とおいた方程式を満たす．この方程式のことを**正規方程式** (normal equation) と呼ぶ．よって，正規方程式をパラメータに関して解くことによって，関数の極値を与えるパラメータを求めることが可能である．注意すべきは，このような手続きによって求められるパラメータは，パラメータの取りうる全ての値に関して行列関数を最大化もしくは最小化する，**大域的最適解** (global optimum) であるとは限らないという点である．このことは，極値を複数与える行列関数の最適化を試みる場合などに問題になる．

例えば，図 2.1 に示す二次関数のグラフでは，$\times$ 印で示した部分で微分，すなわち波線の傾きが 0 になり，波線が水平になる．このように微分が 0 になる地点は二次関数には 1 箇所しか存在しないが，このことに基づいて，二次関数は**凸関数** (convex function) と呼ばれることがある．よって，$\times$ 印の地点の x の値が，この二次関数の大域的最適解である．一般に凸関数では，微分が 0 になる地点は，その関数の大域的最適解であることが知られている．

問題は凸関数ではない関数の最適化を行う場合である．図 2.2 に示す四次関数のグラフは，凸関数ではない関数の例として知られている．このグラフでは，微分が 0 になる地点が 3 つ存在する．従って，微分を 0 とおいた正規方程式を満たす x の値は 3 つ存在することになるので，正規方程式を解くことによって唯一の大域的最適解（この中では一番左の地点）が得られるわけではない．このことは，あるパラメータが正規方程式を満足することは，そのパラメータ値が大域的最適解であることの必要条件ではあるが，十分条件ではないことを意味している．大域的最適解ではないが，微分を 0 にするパラメータ値のことを，**局所解** (local optimum) と呼ぶ．

凸関数とそうでない関数の厳密な定義や，関数の最適化に関するより詳細な議論に関しては，金谷（2005）が参考になる．

このような問題点がありつつも，微分を用いた最適化は非常に良く用いられる．その例として，以下の最小化問題を考えよう．

$$min.\ f(\mathbf{W}) = ||\mathbf{Y} - \mathbf{X}\mathbf{W}||^2\ over\ \mathbf{W} \tag{2.71}$$

ここで，$\mathbf{Y}$ は $n \times q$ 行列，$\mathbf{X}$ は $n \times p$ 行列，$\mathbf{W}$ は $p \times q$ のパラメータ行列である．この最小化問題は，n 個体の関する p 個の独立変数を用いて，q 個の従属変数の値を予測・説明する回帰分析の定式化である．なお，回帰分析に関しては，第 3 章で詳述する．

一般に，$p \times q$ のパラメータ行列 $\mathbf{A} = \{a_{ij}\}$ を引数とする行列関数 $f(\mathbf{A})$ の微分は，行列関数を $\mathbf{A}$ の各要素で微分し，その結果を $\mathbf{A}$ と同じように並べた，$p \times q$ の行列として得られる．すなわち，微分の結果の (i, j) 要素は

$$\left[\frac{df(\mathbf{A})}{d\mathbf{A}}\right]_{ij} = \frac{df(\mathbf{A})}{da_{ij}} \tag{2.72}$$

で得られる．ここで，$[\bullet]_{ij}$ は，括弧中の行列の (i, j) 要素を表す．行列の微分に関しての詳述は避けるが，(2.71) の微分のためには

$$\frac{d\mathrm{tr}\mathbf{B}\mathbf{A}}{d\mathbf{A}} = \mathbf{B}' \tag{2.73}$$

及び

$$\frac{d\mathrm{tr}\mathbf{A}'\mathbf{B}\mathbf{A}}{d\mathbf{A}} = 2\mathbf{B}'\mathbf{A} \tag{2.74}$$

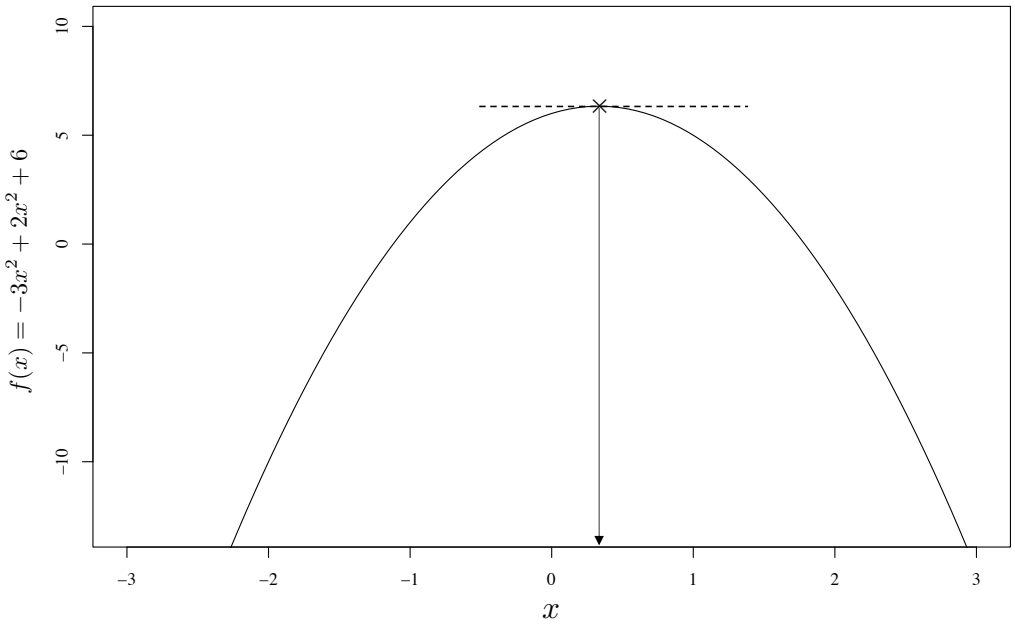

Figure 2.1: 二次関数の極値

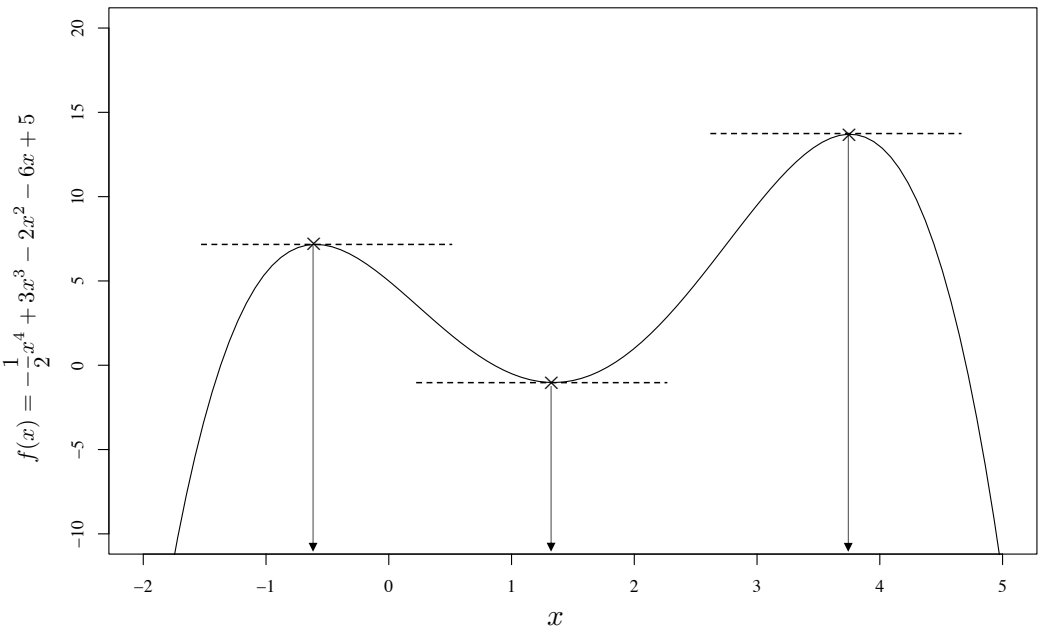

Figure 2.2: 四次関数の極値

が成り立つことだけを用いれば良い．その他の微分の性質に関しては，例えば，Magnus & Neudecker (2019) を参照せよ．

さて，上の性質を使うと，(2.71) の微分は

$$\begin{aligned}\frac{df(\mathbf{W})}{d\mathbf{W}} &= \frac{d(\mathrm{tr}\mathbf{Y}'\mathbf{Y} - 2\mathrm{tr}\mathbf{Y}'\mathbf{X}\mathbf{W} + \mathrm{tr}\mathbf{W}'\mathbf{X}'\mathbf{X}\mathbf{W})}{d\mathbf{W}} \\ &= -2\mathbf{X}'\mathbf{Y} + 2\mathbf{X}'\mathbf{X}\mathbf{W} \end{aligned} \tag{2.75}$$

で得られる．この結果を ${}_p\mathbf{O}_q$ とおいた正規方程式を $\mathbf{W}$ に関して解けば良い．すなわち

$$\begin{aligned} -2\mathbf{X}'\mathbf{Y} + 2\mathbf{X}'\mathbf{X}\mathbf{W} &= {}_p\mathbf{O}_q \\ \Leftrightarrow \mathbf{X}'\mathbf{X}\mathbf{W} &= \mathbf{X}'\mathbf{Y} \\ \Leftrightarrow \mathbf{W} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \end{aligned} \tag{2.76}$$

が得られる．上式では $\mathbf{X}'\mathbf{X}$ の逆行列を用いているが，この逆行列が存在するためには $n \geq p$ であり，なおかつ $\mathbf{X}$ がフルランクであれば良い．

2.11.5 テン・ベルジの定理による最大化

正規直交行列を含む行列関数の最大化では，**テン・ベルジの定理** (ten Berge's theorem) と呼ばれる定理を用いると便利である．

まず，定理の導入のために，**部分直交行列** (sub-orthonormal matrix) と呼ばれる行列を導入する．部分直交行列は，正規直交行列の一部の行や列ベクトルを取り除いた行列である．例えば，$n \times n$ 正規直交行列 $\mathbf{T}$（ただし，$n \geq 4$）に対して，第 1 列から第 4 列までを取り出した $n \times 4$ の列直交行列 $\mathbf{T}_4$ は，部分直交行列である．さらに，$\mathbf{T}_4$ の第 1 行から第 4 行を取り出した 4×4 行列も部分直交行列である．また，部分直交行列同士の積は，部分直交行列であり，正規直交行列もまた部分直交行列である．

その上で，テン・ベルジの定理は次のようなものである．

定理 1 $\mathbf{G}$ を $r \times r$ 部分直交行列であるとする．この時，$r \times r$ の任意の対角行列 $\mathbf{D}$ に対して，$\mathbf{G}$ を引数とする行列関数

$$f(\mathbf{G}) = \mathrm{tr}\mathbf{G}\mathbf{D} \tag{2.77}$$

の最大値について，次の不等式が成り立つ．

$$f(\mathbf{G}) \leq \mathrm{tr}\mathbf{D} \tag{2.78}$$

等号成立は $\mathbf{G} = \mathbf{I}_r$ の時である．

証明は，ten Berge (1983)[3] を参照せよ．

次に，$p \times q$ の行列 $\mathbf{A}$（ただし，$p \geq q$ であり $\mathrm{rank}(\mathbf{A}) = q$）と $p \times q$ の行列 $\mathbf{W}$ に関して，次の最大化問題を考えよう．

$$Max.\ f(\mathbf{W}) = \mathrm{tr}\mathbf{A}'\mathbf{W}\ over\ \mathbf{W}\ s.t.\ \mathbf{W}'\mathbf{W} = \mathbf{I}_q \tag{2.79}$$

パラメータ $\mathbf{W}$ に関する制約条件があることに注意しよう．最大化のために，$\mathbf{A}$ の特異値分解

$$\mathbf{A} = \mathbf{U}\mathbf{D}\mathbf{V}' \tag{2.80}$$

を用いる．ここで，$\mathbf{U}$ は $\mathbf{U}'\mathbf{U} = \mathbf{I}_q$ を満たす $p \times q$ 列正規直交行列，$\mathbf{V}$ は $\mathbf{V}'\mathbf{V} = \mathbf{V}\mathbf{V}' = \mathbf{I}_q$ を満たす $q \times q$ 列正規直交行列，$\mathbf{D}$ は q 個の特異値が降順対角に並ぶ $q \times q$ 対角行列である．この時，目的関数は

$$f(\mathbf{W}) = \mathrm{tr}\mathbf{V}\mathbf{D}\mathbf{U}'\mathbf{W} = \mathrm{tr}(\mathbf{U}'\mathbf{W}\mathbf{V})\mathbf{D} \tag{2.81}$$

と書き換えられる．ここで，$p \times p$ 行列 $\mathbf{U}'\mathbf{W}\mathbf{V}$ は，3 つの列正規直交行列の積なので，部分直交行列である．ここで，テン・ベルジの定理を用いて，次の不等式が成り立つ．

$$f(\mathbf{W}) = \mathrm{tr}\mathbf{U}'\mathbf{W}\mathbf{V}\mathbf{D} \leq \mathrm{tr}\mathbf{D} \tag{2.82}$$

つまり $\mathbf{A}$ の特異値の和が $f(\mathbf{W})$ の最大値である．その等号が成り立つ時，つまり，目的関数の最大値を与える $\mathbf{W}$ は

$$\mathbf{U}'\mathbf{W}\mathbf{V} = \mathbf{I}_q \Leftrightarrow \mathbf{W} = \mathbf{U}\mathbf{V}' \tag{2.83}$$

で得られる．この $\mathbf{W}$ は

$$\mathbf{W}'\mathbf{W} = (\mathbf{U}\mathbf{V}')'(\mathbf{U}\mathbf{V}') = \mathbf{I}_q \tag{2.84}$$

となり，確かに制約条件を満たす．

2.11.6 その他の最適化

上で紹介した微分およびテン・ベルジの定理を用いた方法は「パラメータ行列 = 」という形で解が得られる．このような形式で得られる解のことを**解析解** (analytical solution) と呼び，解析解を持つ最適化問題を，**解析的に解ける**と表現する．しかしながら，実際のところ，解析的に解ける最適化問題は稀である．解析的に解くことのできない最適化問題に対しては，様々な最適化の方法を用いることがあるが，本書で紹介する多くの最適化問題では，**繰り返しアルゴリズム**を用いる．繰り返しアルゴリズムの概略は以下のようなものである．まず，解析的に解くことのできない最適化問題における目的関数を $f(\mathbf{A}, \mathbf{B})$ とし，これは 2 つのパラメータ行列をその引数として持つとする．これを最小化するために，以下のアルゴリズムを実行する．

1. $\mathbf{A}$ と $\mathbf{B}$ に適当な初期値を与える．
2. 現在の $\mathbf{A}$ と $\mathbf{B}$ を用いて目的関数の値を記録する．
3. $\mathbf{A}$ を現在の値で固定した上で，目的関数を $\mathbf{B}$ に関して最小化する $\mathbf{B}$ を求め，それによって $\mathbf{B}$ を更新する．
4. $\mathbf{B}$ を現在の値で固定した上で，目的関数を $\mathbf{A}$ に関して最小化する $\mathbf{A}$ を求め，それによって $\mathbf{A}$ を更新する．
5. 2 で記録した目的関数の値と，現在の $\mathbf{A}$ と $\mathbf{B}$ による関数値の差が十分に小さければアルゴリズムを停止し，現在の $\mathbf{A}$ と $\mathbf{B}$ を出力する．そうでなければ 2 に戻る．

3 と 4 で目的関数が減少することが保証されていれば，これらのステップを繰り返すことによって目的関数の値はどんどん減少していく．目的関数の値が無限に小さくなっていくような最適化問題でなければ，ステップを十分に繰り返すことによって目的関数の値を十分に小さくすることができ，関数値の γ 根省はやがて収束することが期待できる．上の繰り返しアルゴリズムでは，そのようなロジックに基づいて，目的関数の数値的な最適化を試みる．

このアルゴリズムは簡便あり，解析的に解けない問題に対しても一応の解を出してくれる点で優れたものであるが，問題点として，初期値への依存性が挙げられる．上のアルゴリズムでは，その出発点として，$\mathbf{A}$ と $\mathbf{B}$ の初期値を用いているが，当然ながら，初期値が変わるとアルゴリズムの出力も異なる．つまり，ある初期値から出発してアルゴリズムを実行して得た解と，別の初期値でアルゴリズムを実行して得た解が一致する保証はない．そのため，乱数のように実行するたびに異なる初期値を設定する場合，アルゴリズムを実行するたびに解が変わるという不都合な現象が起きてしまう．あるいは，アルゴリズムを実行して得られる解よりも小さな目的関数の値を与える解（すなわち，1 回目の実行時よりも良い解）が得られる可能性もあるだろう．このように，アルゴリズムの実行によって最良の解として得られたのにも関わらず，それよりも良い解が得られた場合，前者の解を**局所解**と呼ぶ．一般に，得られた解が局所解であるかどうかは未知である．そのため，局所解を避けて，なるべく大域的最適解に近い解を得るために，複数（例えば 100 個）の初期値から別々にアルゴリズムを開始し，その結果得られた 100 個の解のなかで，最も小さな関数値を与える解を最終的な解として採用するなどの方略が取られることがある．

Bibliography

[1] Harville, D. A. (1998). *Matrix algebra from a statistician's perspective*. Taylor & Francis.

[2] 金谷健一. (2005). これなら分かる最適化数学-基礎原理から計算手法まで. 共立出版.

[3] ten Berge, J.M.F. (1983). A generalization of Kristof's theorem on the trace of certain matrix products. *Psychometrika*, 48, 519–523.