2026 年度秋学期　画像情報処理　第８回

第２部・画像情報圧縮／ 行列の直交変換と基底画像

前回は，画像をベクトルで表し，ベクトルを直交行列で主成分に変換することで情報量を圧縮する考え方について説明しました。今回は，画像を行列で表現した場合に，前回説明したベクトルの行列による直交変換がどのような変換に対応するかを説明し，これにもとづく直交変換・ユニタリー変換の考え方を説明します。

前回の最後に説明したように，取り扱う画像群の分散共分散行列を求めることは一般にはできないので，適切な変換は経験的に求める必要があります。画像を行列で表すと，直交変換が目に見えるようになり，経験的に適切な変換を考えやすくなります。このことを具体的に行う方法として，画像を「基底画像」の組み合わせで表現し，基底画像を省略することで情報量を圧縮する方法について説明します。

行列の Kronecker 積と変換

ベクトルの変換から行列の変換へ

前回は画像をベクトルで表して，ベクトルの直交変換について説明しました。しかし，ディジタル画像は輝度を表す数値が２次元に配置されているものですから，画像は行列で表すほうが自然です。そこで，行列の変換について考えてみましょう。

前回，画像を表すベクトル $\boldsymbol{x}$ を，行列 P によってベクトル $\boldsymbol{z}$ に変換するという操作を考えました。つまり，

$$\boldsymbol{z} = P'\boldsymbol{x} \tag{1}$$

という変換です。このとき，列ベクトル $\boldsymbol{x}$ と $\boldsymbol{z}$ は m^2 個の要素でできているとし，P' は $m^2 \times m^2$ 行列であるとします。

ここで，画像を表すベクトル $\boldsymbol{x}$ を画像を表す行列に書き換えてみます。列ベクトル $\boldsymbol{x}$ を m 個の m 要素の列ベクトル $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j, \ldots, \boldsymbol{x}_m$ に分けます。すなわち

$$\boldsymbol{x} = \left(\begin{array}{c} \boldsymbol{x}_1 \\ \hline \vdots \\ \hline \boldsymbol{x}_j \\ \hline \vdots \\ \hline \boldsymbol{x}_m \end{array} \right) \tag{2}$$

とします。例えば，9 要素からなる列ベクトルを 3 要素の列ベクトル 3 個に分けるということです。分けられた各列ベクトルを行方向に並べ替えて，$m \times m$ 行列 X を得ます。すなわち，

$$X = \left(\begin{array}{ccccc} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_j & \cdots & \boldsymbol{x}_m \end{array} \right) \tag{3}$$

とします。同様にして，ベクトル $\boldsymbol{z}$ から行列 Z を得ます。

行列の Kronecker 積

さて，$m^2 \times m^2$ 行列 P' が，次のような 2 つの $m \times m$ 行列 C と R を使って，次のように表されるとします。

$$C = \begin{pmatrix} c_{11} & \cdots & c_{1m} \\ \vdots & \ddots & \vdots \\ c_{m1} & \cdots & c_{mm} \end{pmatrix},\ R = \begin{pmatrix} r_{11} & \cdots & r_{1m} \\ \vdots & \ddots & \vdots \\ r_{m1} & \cdots & r_{mm} \end{pmatrix} \tag{4}$$

$$P' = \begin{pmatrix} r_{11}c_{11} & \cdots & r_{11}c_{1m} & & r_{1m}c_{11} & \cdots & r_{1m}c_{1m} \\ \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ r_{11}c_{m1} & \cdots & r_{11}c_{mm} & & r_{1m}c_{m1} & \cdots & r_{1m}c_{mm} \\ & \vdots & & \ddots & & \vdots & \\ r_{m1}c_{11} & \cdots & r_{m1}c_{1m} & & r_{mm}c_{11} & \cdots & r_{mm}c_{1m} \\ \vdots & \ddots & \vdots & \cdots & \vdots & \ddots & \vdots \\ r_{m1}c_{m1} & \cdots & r_{m1}c_{mm} & & r_{mm}c_{m1} & \cdots & r_{mm}c_{mm} \end{pmatrix} \tag{5}$$

つまり，行列 P' は，行列 R の各要素に行列 C を「貼り付け」たものになっています。この関係を

$$P' = R \otimes C \tag{6}$$

で表し，「P' が R と C の **Kronecker 積** (Kronecker product) で表される」といいます。

分離可能な変換

これらの表現を使うと，ベクトル $\boldsymbol{x}$ の行列 P による変換は

$$Z = CXR' \tag{7}$$

と，行列 X の行列 C, R による変換で表されます。証明は，付録 1 に示します。また，R と C が直交行列のとき，P' も直交行列となります。このことの証明は，付録 2 に示します。

以上から，画像を表すベクトル $\boldsymbol{x}$ を直交行列 P' で変換する演算は，ベクトル $\boldsymbol{x}$ を行列 X で表した場合，P' が直交行列 C と直交行列 R の Kronecker 積で表されるならば，(7) 式のように表されることがわかりました。ところで，(7) 式を見ると，行列 C は行列 X の列に，行列 R は行列 X の行に作用していることがわかります[1]。つまり，「P' が C と R の Kronecker 積で表される」とは，「P' の作用を，画像 X の列に対する作用と行に対する作用に分解できる」ことを意味しています。このことを，変換 P' が **分離可能** (separable) であるといいます。

行列の直交変換，ユニタリー変換

やりたいのは「どんな画像でもひとつの変換で圧縮する」こと

前回説明したように，画像の直交変換は，画像を「重要な成分」と「重要でない成分」に分けることを目的としています。そこで，前回説明した主成分分析（KL 変換）では，(1) 式の変換で P' をうまく選ぶことで，ベクトル $\boldsymbol{z}$ の下の方の要素をほとんど 0 にしてしまっていました。

しかし，KL 変換では，変換するための行列（今回の例でいえば，P'，あるいは C と R）は，それぞれの画像 X 1 つ 1 つに対して，別々に選ばなければなりません。どんな画像を圧縮しなければならないかを事前に知ることは，一般にはできません。前回の説明で，対象とする画像群の分散共分散行列がわ

[1] ここで文字 C, R を選んでいるのは，C は列 (column) の頭文字，R は行 (row) の頭文字だからです。

からなければならないので，一般に KL 変換を画像圧縮に用いることはできない，と言っていたのと同じ問題です。われわれの目的は，「いろいろな画像に一定の変換を施して，少ない情報量で画像を伝達する」ことなのです。

直交変換とユニタリー変換

そこで，行列 C, R を画像に依存せずにあらかじめ選んでおいて，変換後の画像 Z を，値 0 の要素（あるいはほとんど 0 とみなせる要素）が，「たいていの」画像について「なるべく多く」なるようにする，という方向で話を進めることにしましょう。この際，画像の行と列について異なる取り扱いをする理由は通常ないので，$C = R$ とします。すると (7) 式は

$$Z = RXR' \tag{8}$$

となります。R は直交行列ですから $RR' = I$ なので（I は単位行列），逆変換は

$$X = R'ZR \tag{9}$$

となります。この形の変換を画像（を表す行列）の**直交変換** (orthogonal transformation) といいます。また，R の要素に複素数を考えるときは，$RR'^* = I$ となるような行列 R を用います。記号 * は共役行列 (各要素の複素共役 (複素数の i を $-i$ に変えたもの) をとった行列) を意味します。このような行列を**ユニタリー行列** (unitary matrix) といい，このとき (8)(9) の変換を**ユニタリー変換** (unitary transformation) といいます。

基底画像と画像情報圧縮

基底画像

さて，どのような直交変換あるいはユニタリー変換を行なえば，つまりどんな R を選べば，「たいていの」画像を圧縮できるのでしょうか？ それは後で説明するとして，ここでは，とりあえず，どのようなユニタリー変換 R を用いるかを決めたものとして，先に進みましょう。

変換後の画像 Z を，「要素のうち 1 つだけが 0 でない行列 m^2 個の和」，すなわち

$$Z = \begin{pmatrix} z_{11} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} + \begin{pmatrix} 0 & z_{12} & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} + \cdots + \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & z_{mm} \end{pmatrix} \tag{10}$$

のように分解すると考えます。このとき，行列 R を行ベクトルを使って

$$\boldsymbol{r}'_j = (r_{j1} \cdots r_{jk} \cdots r_{jm}), \; R = \begin{pmatrix} \boldsymbol{r}'_1 \\ \vdots \\ \boldsymbol{r}'_j \\ \vdots \\ \boldsymbol{r}'_m \end{pmatrix} \tag{11}$$

と表すと，(9) 式は

$$
\begin{aligned}
X &= R' \begin{pmatrix} z_{11} & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} R + R' \begin{pmatrix} 0 & z_{12} & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{pmatrix} R + \cdots + R' \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & z_{mm} \end{pmatrix} R \\
&= R' z_{11} \begin{pmatrix} \boldsymbol{r}'_1 \\ \mathbf{0}' \\ \vdots \\ \mathbf{0}' \end{pmatrix} + R' z_{12} \begin{pmatrix} \boldsymbol{r}'_2 \\ \mathbf{0}' \\ \vdots \\ \mathbf{0}' \end{pmatrix} + \cdots + R' z_{12} \begin{pmatrix} \mathbf{0}' \\ \mathbf{0}' \\ \vdots \\ \boldsymbol{r}'_m \end{pmatrix} \\
&= z_{11} \boldsymbol{r}_1 \boldsymbol{r}'_1 + z_{12} \boldsymbol{r}_1 \boldsymbol{r}'_2 + \cdots + z_{mm} \boldsymbol{r}_m \boldsymbol{r}'_m \\
&= \sum_{i=1}^{m} \sum_{j=1}^{m} z_{ij} \boldsymbol{r}_i \boldsymbol{r}'_j \qquad (12)
\end{aligned}
$$

のように，R の行（R' の列）どうしの直積でできる行列に分解されます（付録 3 参照)。

これらの直積による行列を，**基底画像**（basis image) といいます。このように書くと，変換後の画像（行列）の各要素 z_{ij} は各々の基底画像の係数と考えることができます。すなわち，「元の画像 X は，(ij) 番の基底画像を z_{ij} 倍して足しあわせたもので表される」ことになります。

基底画像と画像データ圧縮

そこで，画像 X を通信する際，事前に変換 R を決めたので，それから求められる基底画像を送信側と受信側に知らせておくとします。そうすると，もしも，m^2 個ある z_{ij} のうちいくつか以外を 0 とみなせるならば，そのいくつかの z_{ij} だけを伝えることで，情報量を圧縮して画像を伝達することができます。このとき，画像 X は，0 でない z_{ij} に対応する基底画像だけでほとんど表現されていることになります。

基底画像とフーリエ変換

どのような基底画像を用いればうまく情報量を圧縮できるかについては，これまでの経験にもとづいて，さまざまな考え方が提示されています。その中でもっとも広く受け入れられているのが，フーリエ変換あるいはその変形を用いる方法です。

ここまでの説明で，画像 X は「基底画像にそれぞれ何かの値をかけて合計する」ことで表されることを示しました。そこで，基底画像を「正弦波を表す指数関数」に置き換えると，これは，第 1 部で説明した，「さまざまな周波数の波に，それぞれ何かの値をかけて合計する」というフーリエ変換の説明と同じであることに気がつくと思います。

実は，２次元離散フーリエ変換はユニタリー変換の１つとして表現され，このときの基底画像には，いろいろな細かさの縦横の波の組み合わせが現れます。このとき，もしも画像 X に細かい波の成分（高い空間周波数成分）があまり含まれていないならば，細かい波を表す基底画像は用いなくてもよく，それに対応する z_{ij} は伝えなくてもよい，ということになります。

これまでの経験にもとづいて，画像圧縮のためのさまざまな基底画像の選び方が提案されていますが，その中でもっとも広く受け入れられているのが，実はこのフーリエ変換あるいはその変形を用いる方法なのです。次回の講義では，離散フーリエ変換がどのようにしてユニタリー変換で表されるかを説明します。さらに，フーリエ変換の変形で，JPEG 方式の画像圧縮で用いられる離散コサイン変換についても説明します。

付録 1：(7) 式の証明

ベクトル $\boldsymbol{x}, \boldsymbol{z}$ の内部にある（縦にならんでいる）j 番目のベクトルを，それぞれ (2) 式に示すように，$\boldsymbol{x}_1, \ldots, \boldsymbol{x}_j, \ldots, \boldsymbol{x}_m$ および $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_j, \ldots, \boldsymbol{z}_m$ で表します。さらに，（内部の）ベクトル $\boldsymbol{x}_j, \boldsymbol{z}_j$ のそれぞれ i 番目の要素をそれぞれ x_{ij}, z_{ij} とすると，(1)(5) 式から

$$z_{ij} = (r_{j1}c_{i1} \cdots r_{j1}c_{im}\ r_{jk}c_{i1} \cdots r_{jk}c_{im}\ r_{jm}c_{i1} \cdots r_{jm}c_{im}) \begin{pmatrix} x_{11} \\ \vdots \\ x_{m1} \\ \vdots \\ x_{1k} \\ \vdots \\ x_{mk} \\ \vdots \\ x_{1m} \\ \vdots \\ x_{mm} \end{pmatrix} = \sum_{k=1}^{m} r_{jk} \sum_{l=1}^{m} c_{il}x_{lk} \tag{A1}$$

と表されます。ここで，k はベクトル $\boldsymbol{x}$ の内部にあるベクトルの番号，l は k 番の（内部）ベクトル $\boldsymbol{x}_k$ の要素の番号です。一方，z_{ij} は行列 Z の i 行 j 列の要素ですから，(7) 式から

$$\begin{aligned} z_{ij} &= (CX)_{i\text{throw}}R'_{j\text{thcolumn}} \\ &= \left(\sum_{l=1}^{m} c_{il}x_{l1} \cdots \sum_{l=1}^{m} c_{il}x_{lk} \cdots \sum_{l=1}^{m} c_{il}x_{lm} \right) \begin{pmatrix} r_{j1} \\ \vdots \\ r_{jk} \\ \vdots \\ r_{jm} \end{pmatrix} \\ &= \sum_{k=1}^{m} r_{jk} \sum_{l=1}^{m} c_{il}x_{lk} \end{aligned} \tag{A2}$$

となり，(A1) 式と (A2) 式から，確かに (A1) 式と (A2) 式が同等であることがわかります。

付録 2：R と C が直交行列のとき P' が直交行列になることの証明

P' の 2 つの列の内積は

$$\begin{aligned} &\sum_{i=1} mr_{1k}c_{il} \cdot r_{1k'}c_{il'} + \cdots + \sum_{i=1} mr_{jk}c_{il} \cdot r_{jk'}c_{il'} + \cdots + \sum_{i=1} mr_{mk}c_{il} \cdot r_{mk'}c_{il'} \\ &= \sum_{j=1} m \sum_{i=1} mr_{jk}c_{il} \cdot r_{jk'}c_{il'} \\ &= \sum_{j=1} mr_{jk}r_{jk'} \sum_{i=1} mc_{il}c_{il'} \end{aligned} \tag{A3}$$

となります。この値は，P' が直交行列であるとすると，$k = k', l = l'$ のとき 1，その他の時は 0 となります。(A3) 式の前の $\sum$ は行列 R の k 列と k' 列の積，後の $\sum$ は行列 C の l 列と l' 列の積ですから，R と C がいずれも直交行列であれば，P' は直交行列となります。

付録 3：ベクトルの内積と直積

ベクトル $\boldsymbol{a} = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$ と $\boldsymbol{b} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$ に対して，

$$\boldsymbol{a}'\boldsymbol{b} = (a_1, a_2) \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} = a_1 b_1 + a_2 b_2 \tag{A4}$$

をベクトル $\boldsymbol{a}, \boldsymbol{b}$ の**内積** (inner product) といいます。また，

$$\boldsymbol{a}\boldsymbol{b}' = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} (b_1, b_2) = \begin{pmatrix} a_1 b_1 & a_1 b_2 \\ a_2 b_1 & a_2 b_2 \end{pmatrix} \tag{A5}$$

をベクトル $\boldsymbol{a}, \boldsymbol{b}$ の**直積** (outer product) といいます。なお，ベクトルの外積（クロス積，cross product）とは違いますので注意してください[2]。

[2]分野によっては，ここでいう「直積」を「外積」とよぶことがあるので，注意が必要です。