画像データ行列の
直交変換・分解

フーリエ変換を
行列計算として表すことで
見えてくるもの

行列計算としてのフーリエ変換（1／2）

- 離散フーリエ変換

$$y(k) = \sum_{n=0}^{N-1} x(n) e^{-i2\pi \frac{k}{N} n}$$

周期 k/N の波を
乗じて，総和を計算

入力信号に対して，
基本となる係数 F_N を
$n\,k$ 乗した値を乗じている
ともみなせる

$$y(k) = \sum_{n=0}^{N} x(n) (F_N)^{nk} \qquad F_N = \exp(-i2\pi/N)$$

x と y を配列とみなして
インデックスを $k=1$
から開始する場合
$x(0) \rightarrow x[1],\ \ y(0) \rightarrow y[1]$

$$y[k] = \sum_{n=1}^{N} x[n] F_N^{(n-1)(k-1)}$$

行列計算としてのフーリエ変換（２／２）

- 線形和の計算　→　行列演算で書ける

$$\forall_k \; y[k] = \sum_{n=1}^{N} x[n] \; F_N^{(n-1)(k-1)}$$

- フーリエ変換行列

$$\begin{pmatrix} y[1] \\ y[2] \\ \vdots \\ y[k] \\ \vdots \\ y[N] \end{pmatrix} = \begin{pmatrix} F_N^{0\cdot 0} & F_N^{1\cdot 0} & \cdots & F_N^{(n-1)\cdot 0} & \cdots & F_N^{(N-1)\cdot 0} \\ F_N^{0\cdot 1} & F_N^{1\cdot 1} & \cdots & F_N^{(n-1)\cdot 1} & \cdots & F_N^{(N-1)\cdot 1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ F_N^{0\cdot (k-1)} & F_N^{1\cdot (k-1)} & \cdots & F_N^{(n-1)(k-1)} & \cdots & F_N^{(N-1)(k-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ F_N^{0\cdot (N-1)} & F_N^{1\cdot (N-1)} & \cdots & F_N^{(n-1)(N-1)} & \cdots & F_N^{(N-1)(N-1)} \end{pmatrix} \begin{pmatrix} x[1] \\ x[2] \\ \vdots \\ x[n] \\ \vdots \\ x[N] \end{pmatrix}$$

$$\mathbf{y} = \mathcal{F}(\mathbf{x}) = \mathbf{F}\mathbf{x}$$

行列計算としての逆フーリエ変換（１／２）

- 離散逆フーリエ変換

虚数の符号が反転する

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} y(k) e^{i2\pi \frac{k}{N} n}$$

入力信号に対して,
基本となる係数 W を
－n k 乗した値を乗じている
ともみなせる

$$x(n) = \frac{1}{N}\sum_{k=0}^{N} y(k) (F_N)^{-nk} \quad F_N = \exp(-i2\pi/N)$$

x と y を配列とみなして
インデックスを k = 1
から開始する場合
$x(0) \to x[1],\ \ y(0) \to y[1]$

$$x[n] = \frac{1}{N}\sum_{k=1}^{N} y[k] F_N^{-(n-1)(k-1)}$$

行列計算としての逆フーリエ変換（２／２）

- 逆フーリエ変換行列

$$\begin{pmatrix} x[1] \\ x[2] \\ \vdots \\ x[n] \\ \vdots \\ x[N] \end{pmatrix} = \frac{1}{N} \begin{pmatrix} F_N^{-0\cdot 0} & F_N^{-1\cdot 0} & \cdots & F_N^{-(n-1)\cdot 0} & \cdots & F_N^{-(N-1)\cdot 0} \\ F_N^{-0\cdot 1} & F_N^{-1\cdot 1} & \cdots & F_N^{-(n-1)\cdot 1} & \cdots & F_N^{-(N-1)\cdot 1} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ F_N^{-0\cdot(k-1)} & F_N^{-1\cdot(k-1)} & \cdots & F_N^{-(n-1)(k-1)} & \cdots & F_N^{-(N-1)(k-1)} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ F_N^{-0\cdot(N-1)} & F_N^{-1\cdot(N-1)} & \cdots & F_N^{-(n-1)(N-1)} & \cdots & F_N^{-(N-1)(N-1)} \end{pmatrix} \begin{pmatrix} y[1] \\ y[2] \\ \vdots \\ y[k] \\ \vdots \\ y[N] \end{pmatrix}$$

$$\mathbf{x} = \mathcal{F}(\mathbf{y}) = \mathbf{F}^{-1}\mathbf{y}$$

補足：
フーリエ変換行列と逆フーリエ変換行列の関係

- 共に対称行列
  - 対角線を基準として，対称位置に同じ係数が現れる．

$$
\mathbf{F}_v = \begin{pmatrix}
F_N^{0\cdot 0} & F_N^{1\cdot 0} & {}^{2\cdot 0}_{N} & \cdots & F_N^{(N-1)\cdot 0} \\
F_N^{0\cdot 1} & F_N^{1\cdot 1} & F_N^{2\cdot 1} & \cdots & F_N^{(N-1)\cdot 1} \\
F_N^{0\cdot 2} & F_N^{1\cdot 2} & F_N^{2\cdot 2} & \cdots & F_N^{(N-1)\cdot 2} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
F_N^{0\cdot (N-1)} & F_N^{1\cdot (N-1)} & F_N^{2\cdot (N-1)} & \cdots & F_N^{(N-1)(N-1)}
\end{pmatrix}
$$

- ユニタリ行列（複素数での直交行列）
  - 逆行列が，複素共役転置で表される．
  - 対称行列であるので，**複素数の符号を反転したもの**．

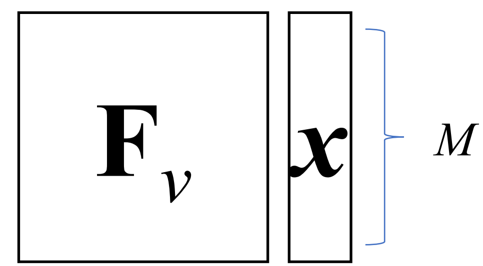
逆行列が簡単に求まる特殊な行列であり，逆変換が計算しやすいことを意味する．

$$
\mathbf{F}^{-1} = \frac{1}{N}\mathbf{F}^H = \frac{1}{N}\overline{\mathbf{F}}^\top = \frac{1}{N}\overline{\mathbf{F}}
$$

補足：　なぜフーリエ変換を行列演算で表さない？

- フーリエ変換を行列で表すと
線形代数との関係が顕著になるが・・・
  - 線形代数の恩恵を受けられる

- なぜ使わない？
  - 行列サイズが大きくなる傾向があり，
計算時に行列（配列）を
用意したくない．



サイズ M のデータに対して，行列サイズは $M\times M$ と膨大になる

- 計算と実装は別
  - 計算の理解には線形代数を用いるが，
実際の計算には行列演算を用いない．

行ベクトルのフーリエ変換（1／3）

- ```
  X = magic(3);  % 何らかの行列を生成（ここでは魔法陣行列）
  ```

```
% ある1列の行ベクトルのみに対して，フーリエ変換
x = X(:,1)
M = length( x );

% フーリエ変換（組み込み関数版）
fft(x)

% フーリエ変換行列
k = 0:M-1;  m = 0:M-1;
Fv = exp( (complex(0,-2*pi)/M) * k' * m );

y = Fv*x
```

$$F_M^{km} = \exp(-i2\pi/M)^{(k-1)(m-1)}$$
$$= \exp(-i2\pi\frac{(k-1)}{M}(m-1))$$

行列の縦,横のサイズを
M, N と表したいので,
N の代わりに M を用いた
```

行ベクトルのフーリエ変換（２／３）

- ```
  % 逆フーリエ変換
  ifft(y)

  % 行列バージョン
  1/M * conj(Fv) * y

  % 逆行列との比較
  inv( Fv )
  1/M * conj(Fv)

  % フーリエ変換行列と逆フーリエ変換の乗算
  1/M * Fv' * Fv
  % 単位行列になる
  ```
  ```

行ベクトルのフーリエ変換（３／３）

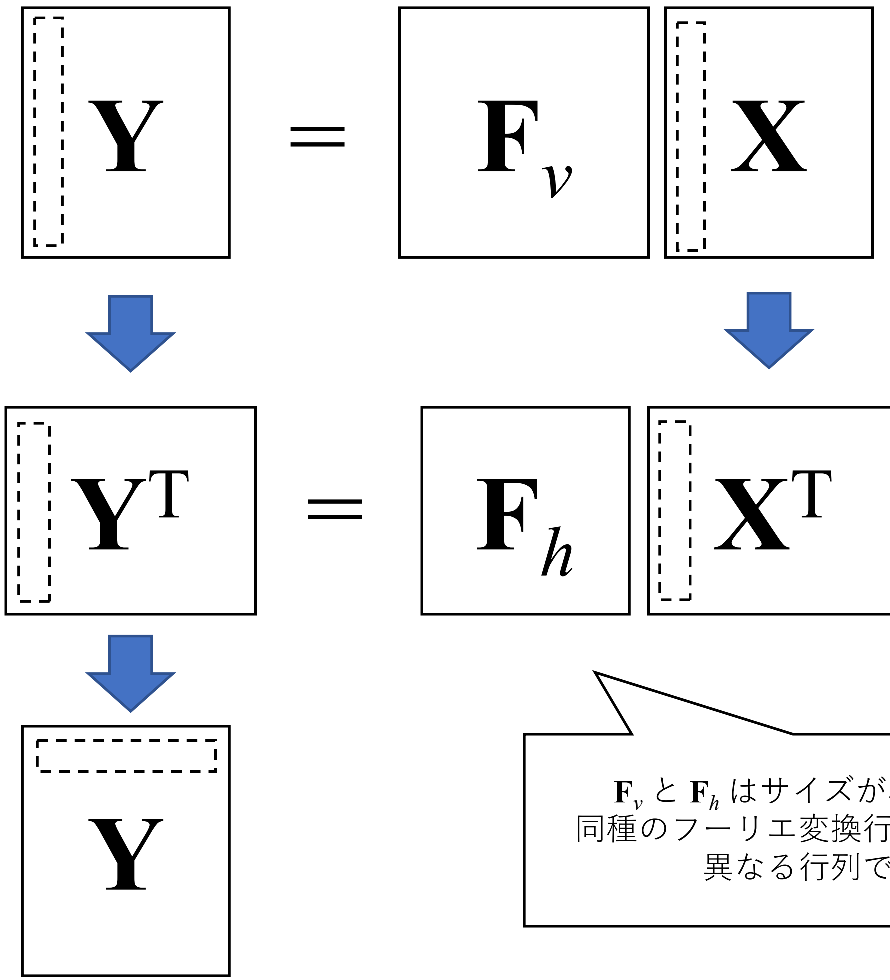
- ```
  % 全ての行に対してフーリエ変換
  fft(X,[],1)

  % フーリエ変換行列を用いるバージョン
  Y = Fv*X

  % 逆フーリエ変換
  1/M * conj(Fv) * Y
  ```

[] は引数を指定しない
という意味
引数のパタンでモードを変えている
1 は垂直方向を表す．水平方向の場合は 2

$$\boldsymbol{y} = \mathbf{F}_v \boldsymbol{x}$$

一行を計算

$$\mathbf{Y} = \mathbf{F}_v \mathbf{X}$$

まとめて計算
  ```

列ベクトルのフーリエ変換（1／2）

- 転置を考える

縦方向への
フーリエ変換

行列を
転置して計算し

計算結果をもと
に戻せば

横方向に
フーリエ変換が
行われる



$\mathbf{F}_v$ と $\mathbf{F}_h$ はサイズが異なるため，
同種のフーリエ変換行列ではあるが，
異なる行列である．

列ベクトルのフーリエ変換（２／２）

- フーリエ変換行列を転置

$$\boxed{\mathbf{Y}^{\mathrm{T}}} = \boxed{\mathbf{F}_h} \boxed{\mathbf{X}^{\mathrm{T}}}$$

$$\boxed{\mathbf{Y}} = \boxed{\mathbf{X}} \boxed{\mathbf{F}_h{}^{\mathrm{T}}}$$

$$(\mathbf{Y}^\top)^\top = (\mathbf{F}_h \mathbf{X}^\top)^\top$$

$$\mathbf{Y} = \mathbf{X}\mathbf{F}_h^\top$$

逆順になりつつ
転置がかかる

F は虚数を含む．虚数行列の転置には，
転置 と 共役複素転置（転置 + 虚数の符号を反転）がある．
ここでは単なる転置となる．

列ベクトルのフーリエ変換

```
% ある1行の行ベクトルのみに対して，フーリエ変換
x = X(1,:)
N = length( x );

% フーリエ変換（組み込み関数版）
fft(x)

% フーリエ変換行列
k = 0:N-1;  n = 0:N-1;
Fh = exp( (complex(0,-2*pi)/N) * k' *n );

x * Fh.'

% 全ての列に対するフーリエ変換
fft( X, [], 2 )

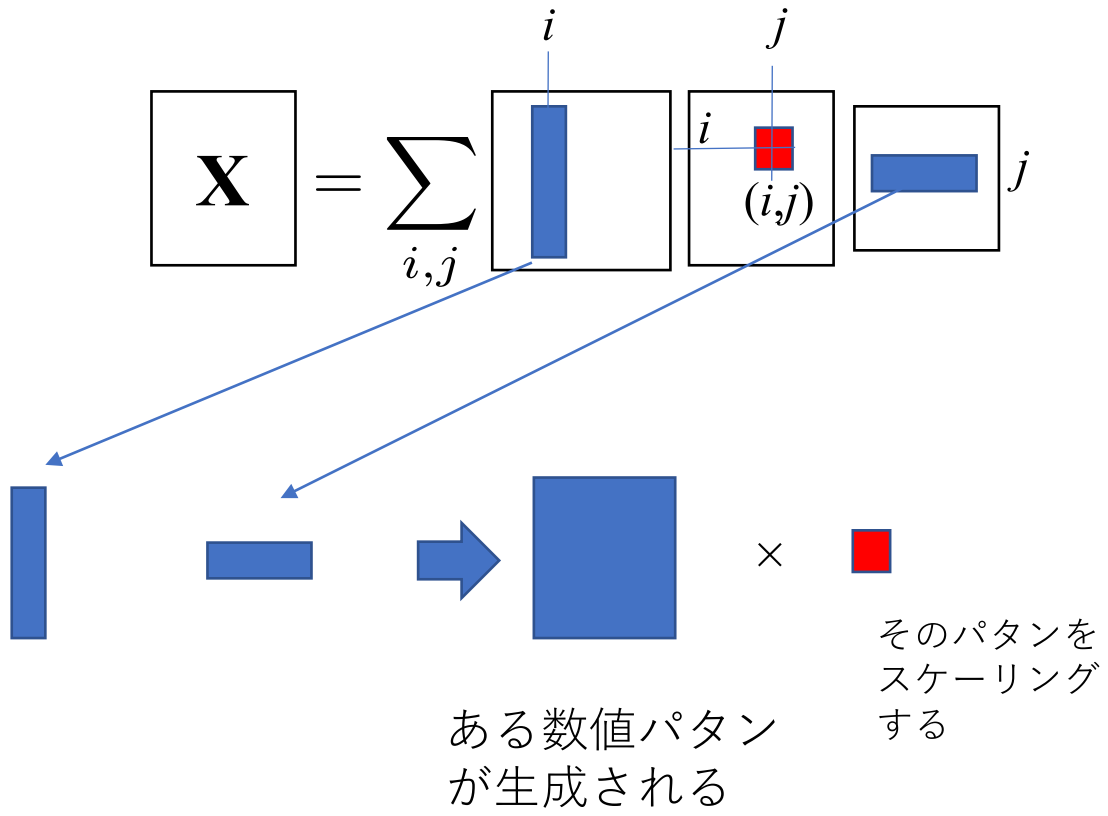
X * Fh.'
```

2次元フーリエ変換を行列で表すと

- 各行に対する フーリエ変換 と
  各列に対する フーリエ変換 で表される
  - 画像に対するフーリエ変換とは,
    画像を行列とみなして,
    両側からフーリエ変換行列を乗ずる変換とみなせる

フーリエ変換

$$y(k,l) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} x(m,n) e^{-i2\pi \frac{k}{M} m} e^{-i2\pi \frac{l}{N} n}$$

$$\boxed{\mathbf{Y}} = \boxed{\mathbf{F}_v} \boxed{\mathbf{X}} \boxed{\mathbf{F}_h^\top}$$

逆フーリエ変換

$$x(m,n) = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} y(k,l) e^{i2\pi \frac{k}{M} m} e^{i2\pi \frac{l}{N} n}$$

$$\boxed{\mathbf{X}} = \underbrace{\boxed{\mathbf{F}_v^{-1}}}_{\frac{1}{M}\overline{\mathbf{F}}_v^\top} \boxed{\mathbf{Y}} \underbrace{\boxed{\mathbf{F}_h^{-\top}}}_{\frac{1}{N}\overline{\mathbf{F}}_h}$$

二次元フーリエ変換

```
% フーリエ変換

% 組み込み関数版
fft2( X )

% 行列計算バージョン
Y = Fv * X * Fh.'


% 逆フーリエ変換

% 組み込み関数版
ifft2( Y )

% 行列計算バージョン
(1/M)*conj(Fv).' * Y * (1/N)*conj(Fh)
```

３つに分解された行列が意味するもの

行列の直交分解（１／２）

- $\mathbf{X}$ という行列を,
  $\mathbf{U}, \mathbf{Y}, \mathbf{V}$ という３つの行列に分解しているともみなせる.

$$\boxed{\mathbf{X}} = \boxed{\mathbf{F}_v^{-1}}\,\boxed{\mathbf{Y}}\,\boxed{\mathbf{F}_h^{-\top}}$$

$$\boxed{\mathbf{X}} = \boxed{\mathbf{U}}\,\boxed{\mathbf{Y}}\,\boxed{\mathbf{V}^{\mathrm{T}}}$$

両側に
直交行列

行列の直交分解（１／２）

- 両側を直交行列で挟まれる場合,
  次のようにも計算ができる.

$$\mathbf{X} = \sum_i \sum_j y(i,j)\mathbf{u}_i \mathbf{v}_j^\top$$



分解された行列の合成

- ```
  U = inv(Fv);
  V = inv(Fh);

  X = zeros( size( Y ) );

  for j = 1:N
    for i = 1:M

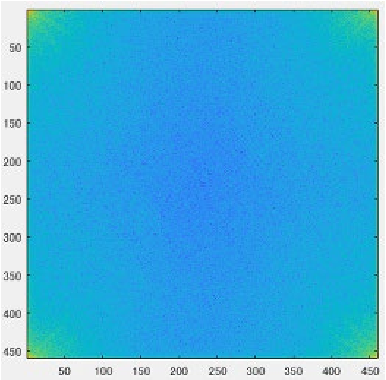
      ui  = U(:,i);
      vj  = V(:,j);
      yij = Y(i,j);

      X = X + yij * ui * vj.';
    end
  end

  X % 結果を表示
  ```
  ```

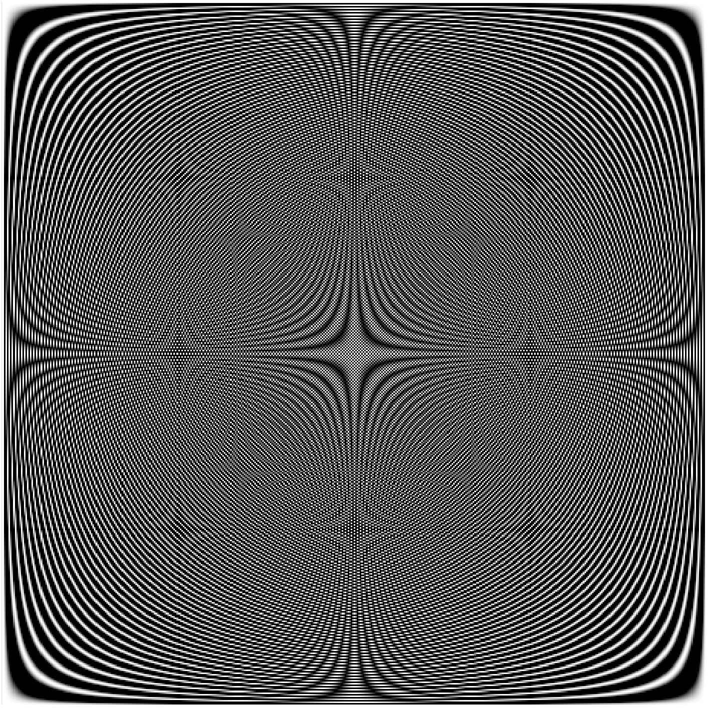
実際の画像データを用いて

```
X = imread('../images/face8.jpg' );
X = im2double( X );
X = rgb2gray( X );
figure(1), imshow( X );

[M,N] = size( X );

% フーリエ変換行列の作成
DFTmtx = @(M) ...
  exp( complex(0,-2*pi)/M * (0:M-1)' * (0:M-1) );

Fv = DFTmtx( M );
Fh = DFTmtx( N );

% フーリエ変換
Y = Fv*X*Fh.';
figure(2), imagesc( log( abs(Y) + 0.01 ) );
```

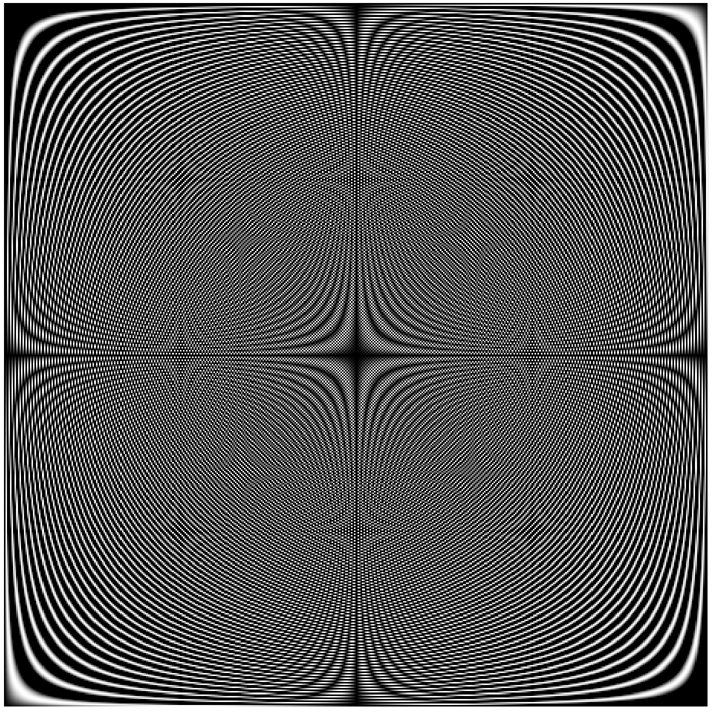


$\mathbf{F}_v$ と $\mathbf{F}_h{}^{\mathrm{T}}$ はどのような行列か？

-
```
figure(3), imshow( [real( Fv ), imag( Fv )] );
figure(4), imshow( [real( Fh.'), imag( Fh.')] );
```



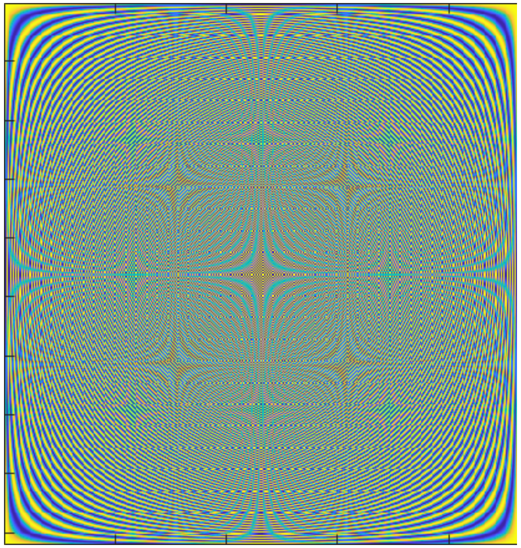
$\mathbf{F}_v$ の実数部分



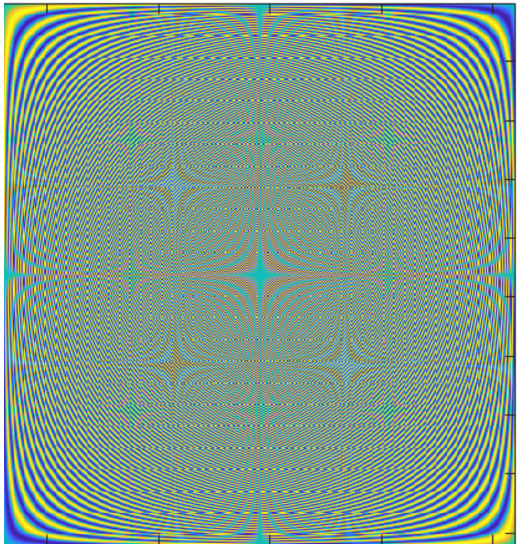
$\mathbf{F}_v$ の虚数部分

$\mathbf{U}$ と $\mathbf{V}^T$ はどのような行列か？

- ```
  % 逆フーリエ変換行列
  U = 1/M * conj( Fv );
  V = 1/N * conj( Fh );

  % さほど変わらない
  figure(5), imagesc( [real( U ), imag( U )] );
  figure(6), imagesc( [real( V.'), imag( V.')] );
  ```

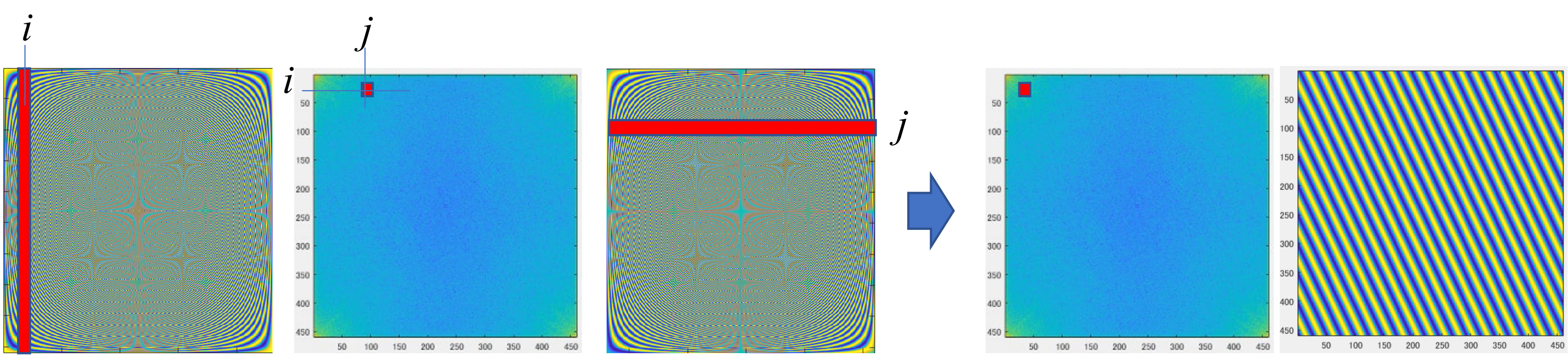


$\mathbf{U}$ の実数部分



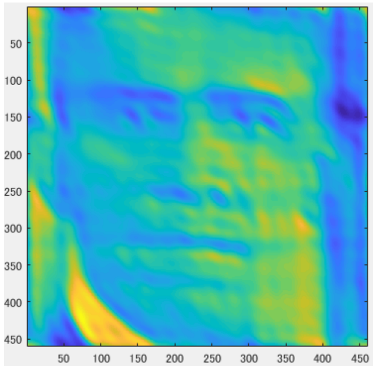
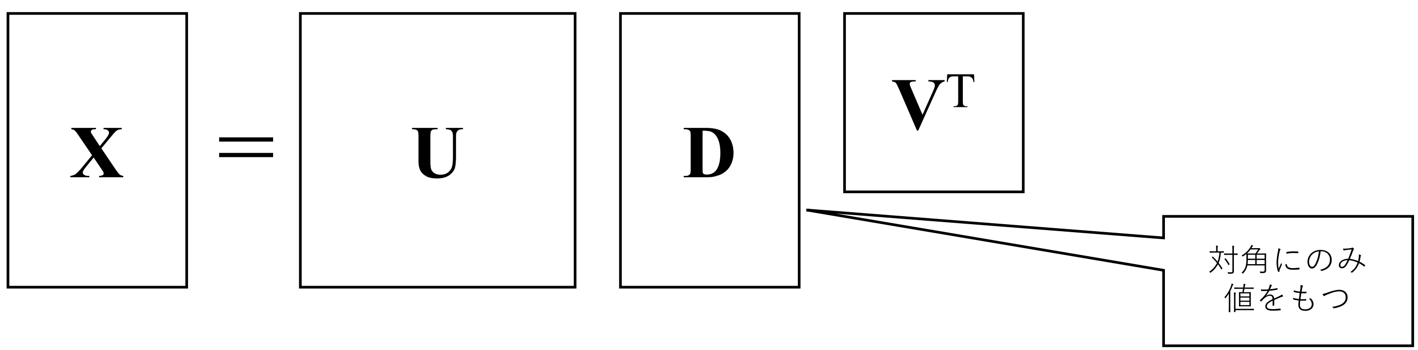
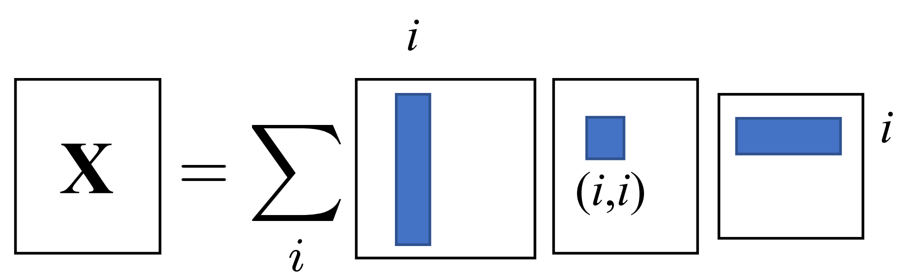
$\mathbf{U}$ の虚数部分
  ```

分解された行列の合成

```
• i = 10;  j = 20;  % の計算を行ってみる
  ui  = U(:,i);   vj  = V(:,j);    yij = Y(i,j);

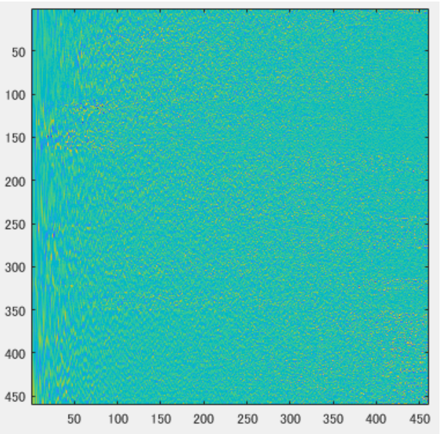
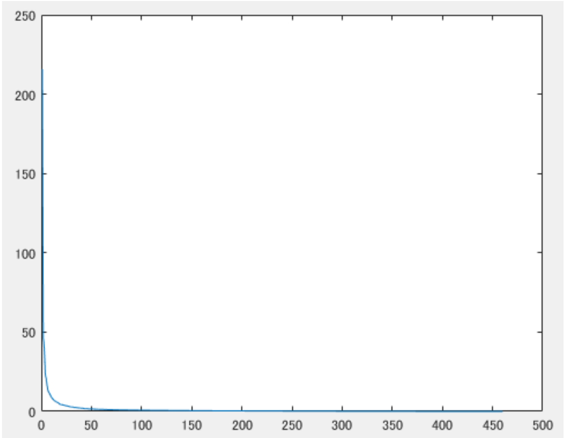
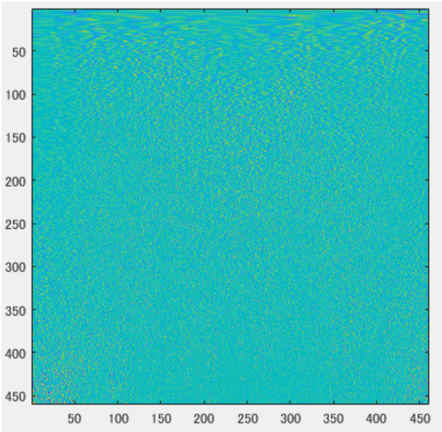
  uvt = ui * vj.';  % パタン
  Z = yij * uvt;

  figure(7), imagesc( [real(uvt), imag(uvt)] );
  figure(8), imagesc( real(Z) );  axis image;
```



赤い部分を取り出して計算すると

あるパタンが
生成される

For 文で累積和を計算してみる

```
Z = zeros( size(X) );
for j = 1:20
  for i = 1:20
    ui  = U(:,i);
    vj  = V(:,j);
    yij = Y(i,j);

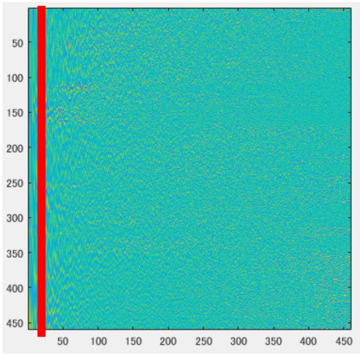
    uvt = ui * vj.';  % パタン
    Z = Z + yij * uvt;

% 高速にパタンが変わり，明滅するので，注視しすぎないよう
%figure(7), imagesc( [real(uvt), imag(uvt)] );
figure(8), imagesc( real(Z) );  axis image;
    drawnow;

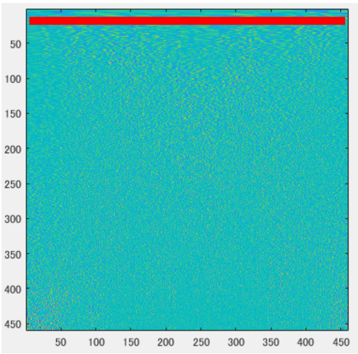
  end
end
```



他の類似な
行列の分解法として

特異値分解

行列の特異値分解

- 特異値分解 (SVD: singular value decomposition)
- 行列を **U**, **D**, **V**$^{\mathrm{T}}$ という３つの行列に分解する
  - **D** は対角行列,　　対角に特異値が並ぶ
  - **U** と **V**は直交行列,　　各行に特異ベクトルが並ぶ



$$\mathbf{X} = \sum_i d(i,i)\mathbf{u}_i \mathbf{v}_i^\top$$



画像データ行列に対する SVD

- ```
  [U,D,V] = svd( X );

  figure(9),  imagesc( U ); axis image;
  figure(10), imagesc( V' ); axis image;
  figure(11), plot( diag( D ) );
  ```



U



D の対角値



$\mathbf{V}^{\mathrm{T}}$
  ```

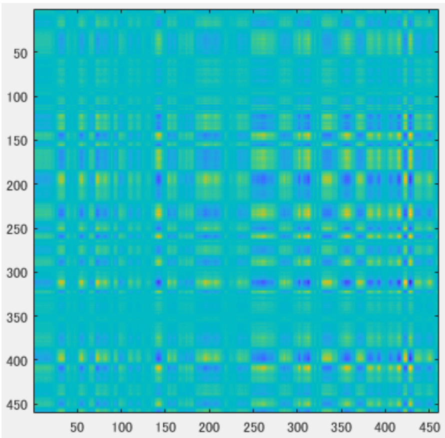
```
Z = zeros( size(X) );

for i = 1:20
  ui = U(:,i);
  vi = V(:,i);
  dii = D(i,i);

  uvt = ui * vi';
  figure(12), imagesc( uvt );

  Z = Z + dii * uvt;
  figure(13), imagesc( Z );

  pause( 1.0 );
end
```



U



$\mathbf{V}^{\mathrm{T}}$



処理結果

５個

１０個

２０個

５０個

特異値・特異ベクトルの個数

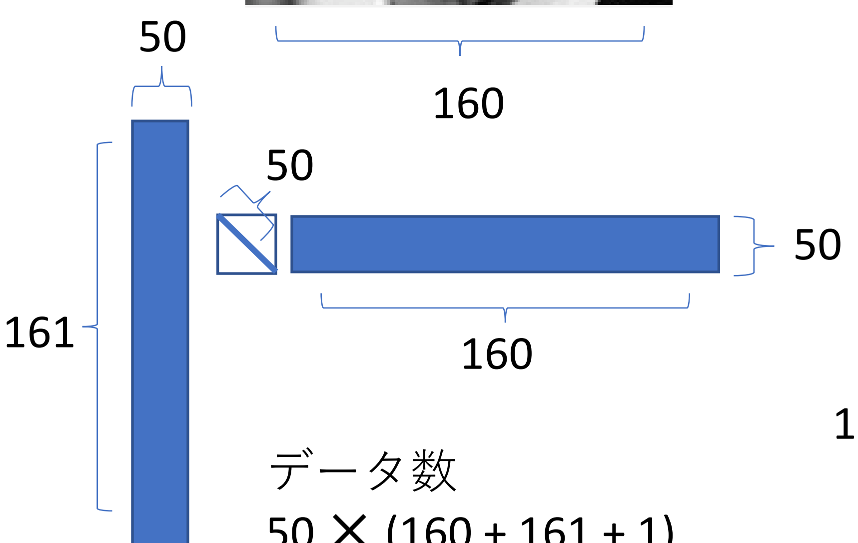
補足：　SVD vs DFT　（１／３）

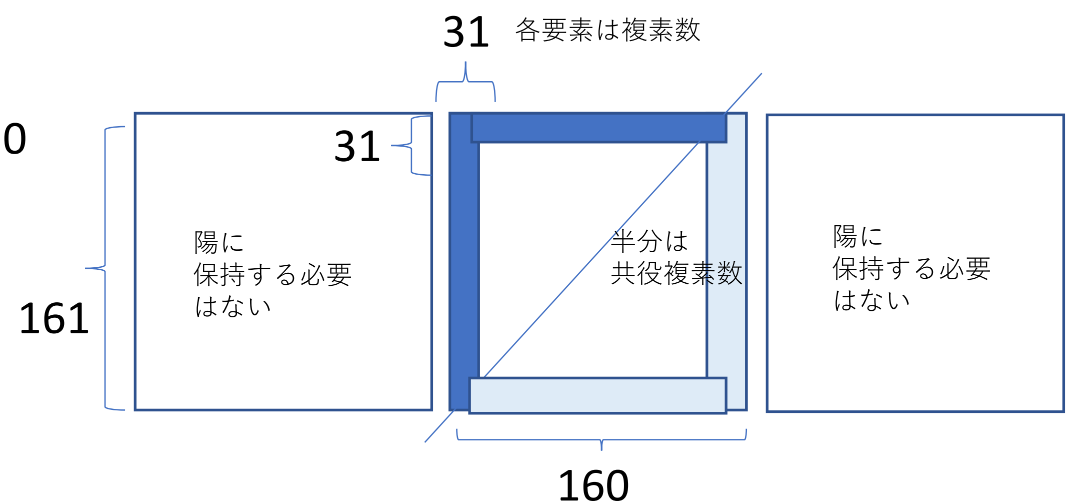
- SVD（特異値分解）のほうが
  少ないパタンでも復元能力が高い

- なぜ使わない？
  - 左右の行列 U と V を（一部ではあるが）
    陽に保持しておく必要がある.
  - 計算量が DFT よりも多い.
  - 低いデータ量のときは，DFT の方が見た目が良い.

- DFT（フーリエ変換）の利点
  - U とV を sin と cos 関数から作り出せる.

補足：　SVD vs DFT　（２／３）





データ数

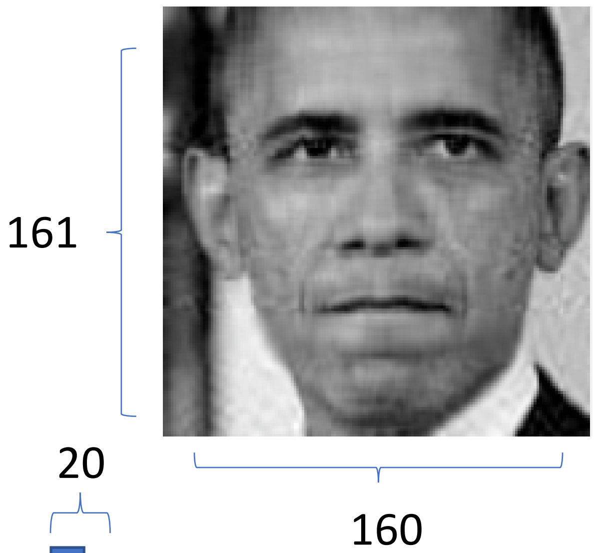
50 ✕ (160 + 161 + 1)

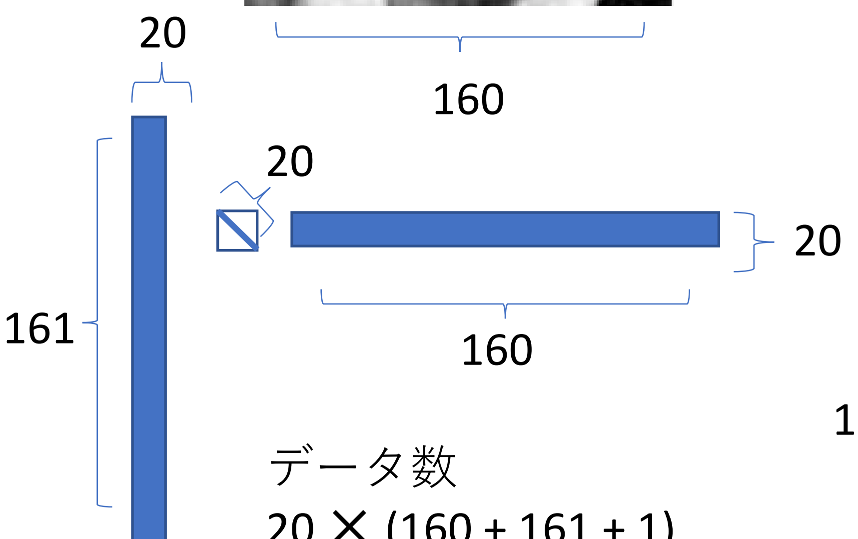
= 16,100

(161✕160 − (161−31✕2) ✕(160−31✕2)) ÷ 2✕2 =

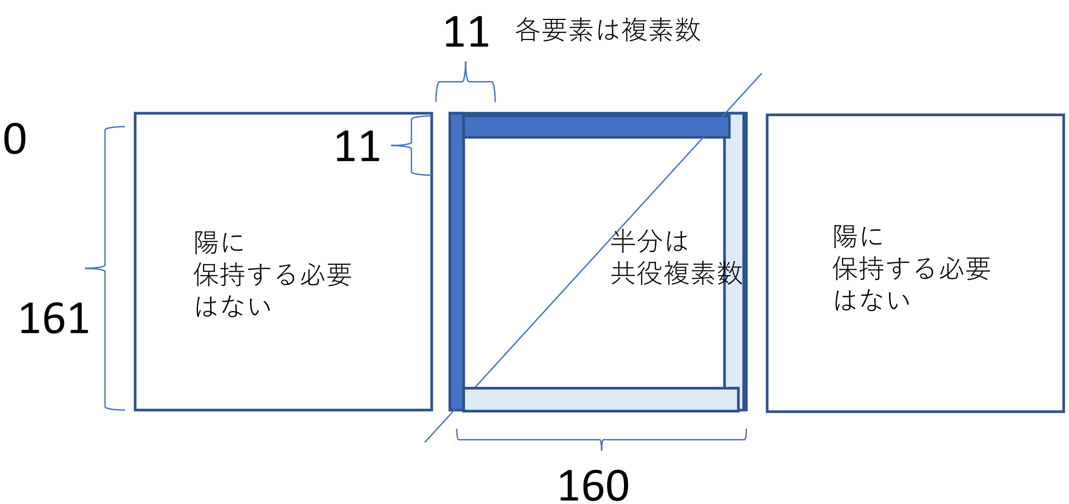
16156

補足：　SVD vs DFT　（３／３）





データ数
20 ✕ (160 + 161 + 1)
= 6440



(161✕160 – (161–11✕2) ✕(160–11✕2)) ÷ 2✕2 = 6716