フーリエ変換を介した
フィルタリング

フーリエ変換を介すると

- 利点が多い
  - 直交変換をすると
    - 重要な成分と不要な成分に分かれることが多い.
  - 不要な成分を捨てることで
    - データ圧縮できる
    - 計算を省ける
  - 計算の効率化と高速化
    - フィルタリングで生じる重複計算を省ける
  - 線形代数の直交化と関係し
    - 逆変換（逆行列）が容易に求まる状態になる（かもしれない）

フィルタリング

- 周辺画素を用いた計算
  - 周辺画素に対して,
    用意したフィルタ係数（重み）を乗算し,
    出力を得る.

- 周波数的に解釈すると
  - フィルタとは,
    画像の持つ周波数成分にうち,
    特定の帯域を通過させるもの.

フィルタリング　（数式的）

- 画像空間

相関演算（所謂フィルタリング）

$$J(x,y) = \sum_{u,v} K(u,v) I(x+u, y+v)$$
$$:= (K * I)(x,y)$$

畳み込み演算

$$J(x,y) = \sum_{u,v} K(u,v) I(x-u, y-v)$$
$$:= (K \otimes I)(x,y)$$

- 周波数空間

$$\mathcal{F}(J)(\xi,\eta) = \overline{\mathcal{F}(K)(\xi,\eta)} \circ \mathcal{F}(I)(\xi,\eta)$$

共役複素数

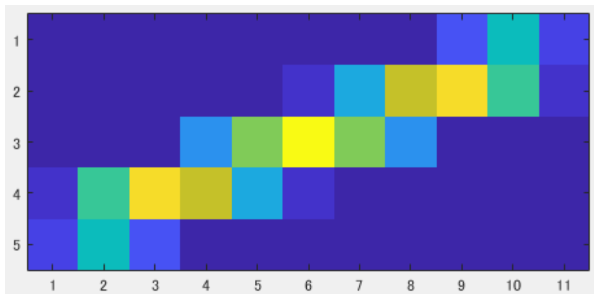
$$\mathcal{F}(J)(\xi,\eta) = \mathcal{F}(K)(\xi,\eta) \circ \mathcal{F}(I)(\xi,\eta)$$

習うより慣れよ　（まずは畳み込み）

```
% 画像をぼかす
% まずは，画像空間で，畳み込みを用いて
I = imread( '../images/Mandrill.png' );
I = im2double( I );

% ぼけ方（点拡がり関数 PSF）
len = 10;
ang = 20;
K = fspecial( 'motion', len, ang );

% ぼかす（畳み込み）
J = imfilter( I, K, 'conv', 'replicate' );

figure(1), imshow( [I, J] );
figure(2), imagesc( K );  axis image;
```

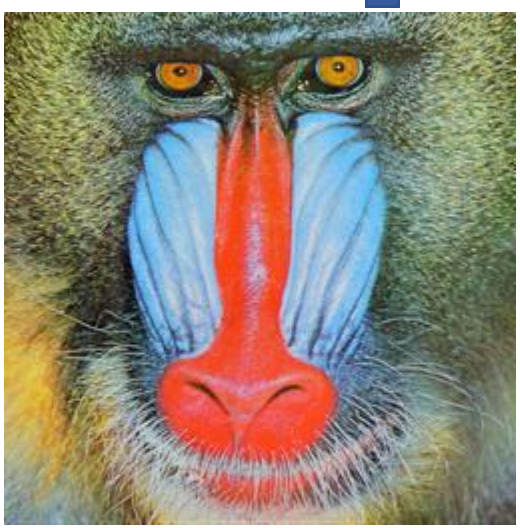
処理結果

PSF

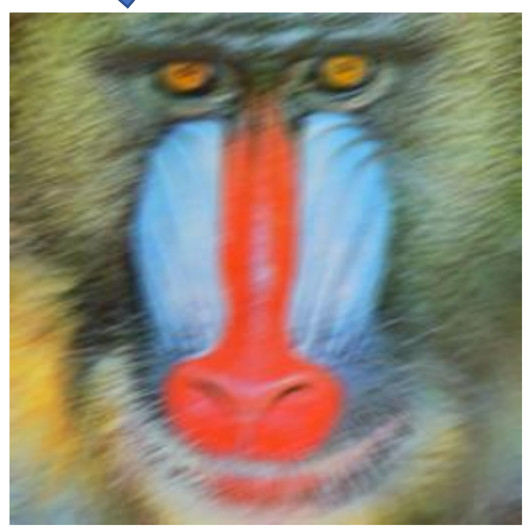


「畳み込み」は
ある点の強度値を
PSF の形状に広げる

畳み込み

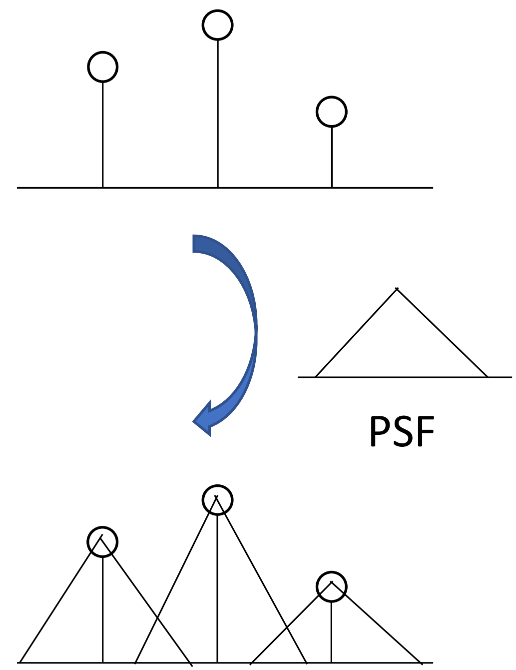


処理前の画像

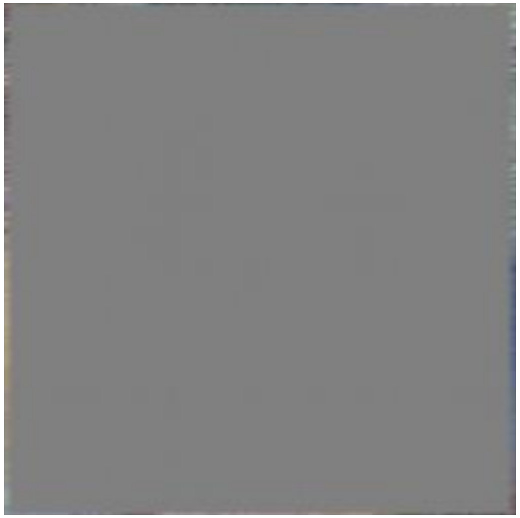
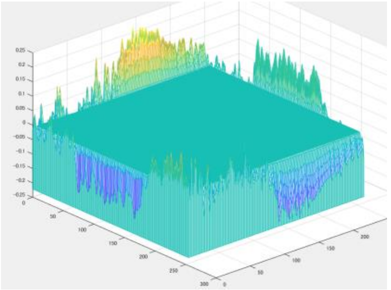
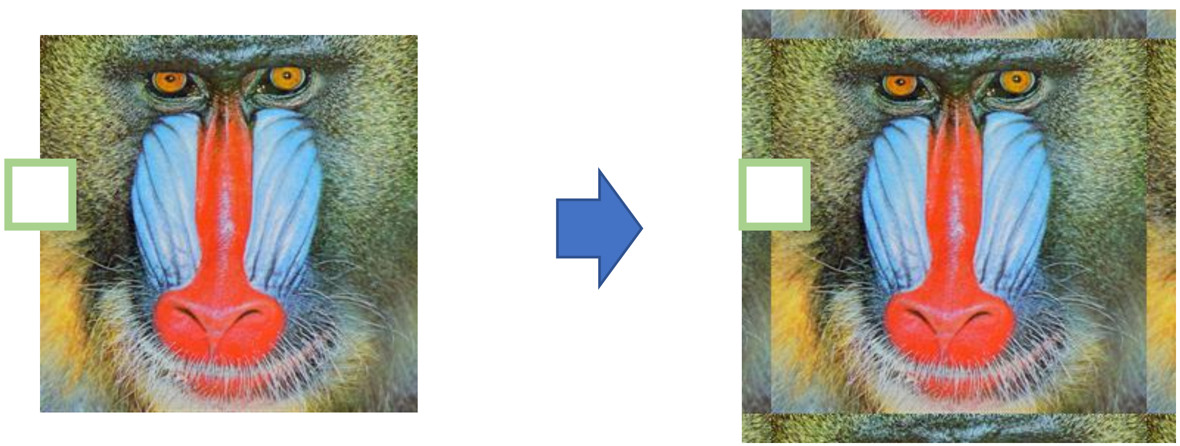


ぼけ画像

１次元の場合



フーリエ変換で一致するか？

- ```
  % フーリエ変換
  FI = fft2( I );

  % 画像と同サイズになるように調整した後,
  % フーリエ変換
  [sy,sx,sc] = size( I );
  FK = psf2otf( K, [sy,sx] );

  % ぼかす（畳み込み）
  FJ = FI .* FK;

  % 逆フーリエ変換
  J_ft = real( ifft2( FJ ) );

  figure(3), imshow( [J, J_ft, J-J_ft+0.5] );
  ```
  ```

処理結果


画像空間での
畳み込みの結果


フーリエ空間を介した
計算の結果


誤差画像

画像端で誤差は生じるが
それ以外ではほぼ同値


（赤色の）誤差を縦軸に示したもの

完全に一致させるには？

- 画像端処理を巡回拡張にする.
  - 画像が巡回していると仮定して処理する.

- % フーリエ変換
```
J = imfilter( I, K, 'conv', 'circular' );
```



習うより慣れよ　（次は相関演算）

- ```
  % 微分フィルタ（Sobel）
  K = (1/8)*[-1 0 1; -2 0 2; -1 0 1];

  % フィルタリング
  J = imfilter( I, K, 'corr', 'replicate' );

  % フーリエ変換バージョン
  [sy,sx,sc] = size( I );
  FI = fft2( I );
  FK = psf2otf( K, [sy,sx] );

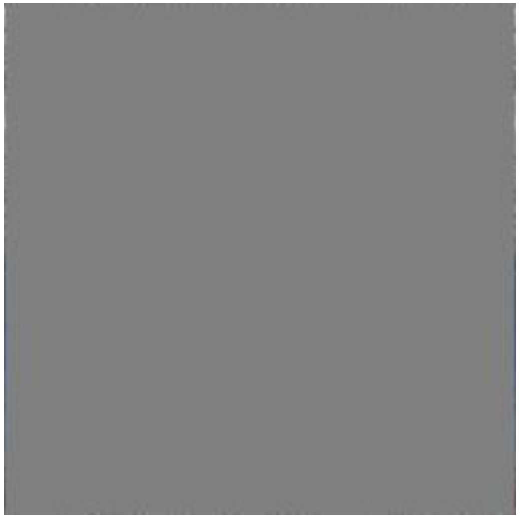
  FJ = FI .* conj( FK );
  J_ft = real( ifft2( FJ ) );

  figure(4), imshow( [J, J_ft, J-J_ft]+0.5 );
  ```
  ```

処理結果

画像空間での
相関演算の結果

フーリエ空間を介した
計算の結果



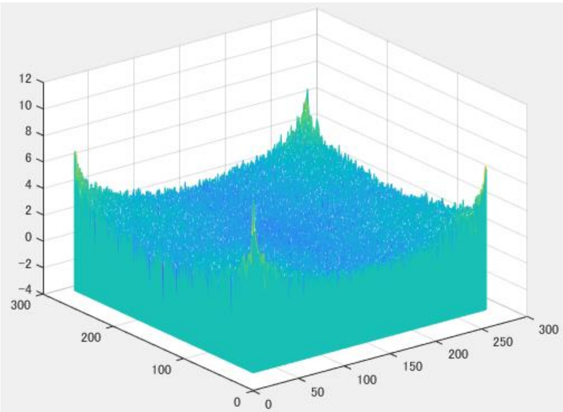
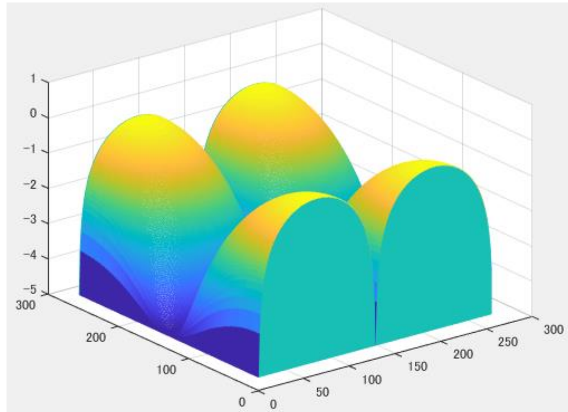
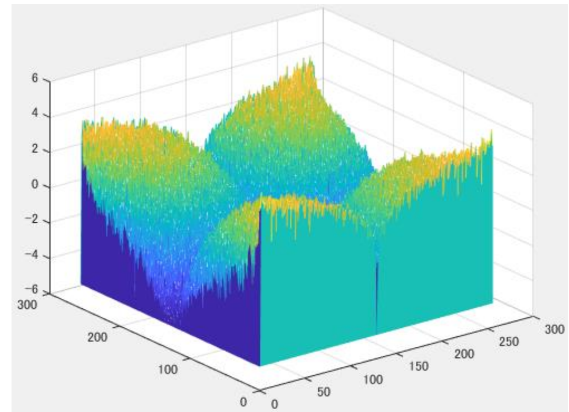
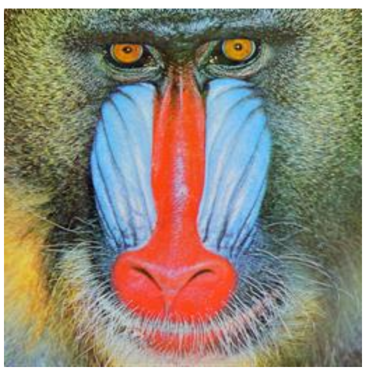
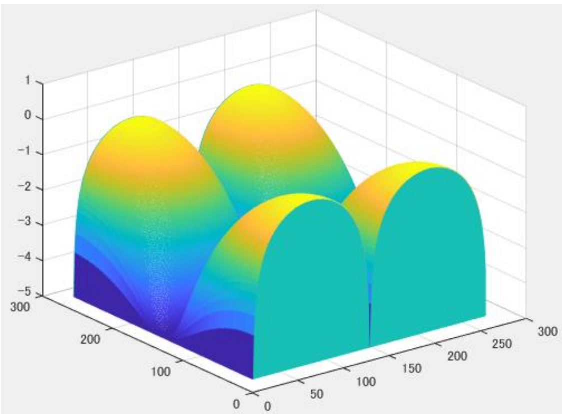
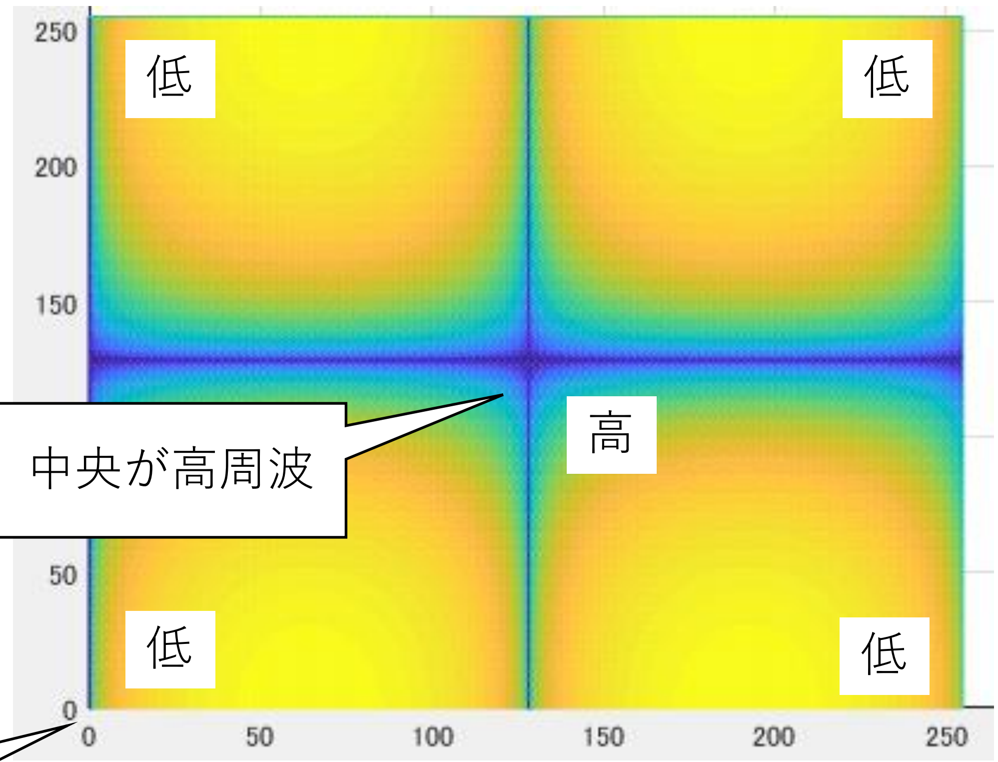
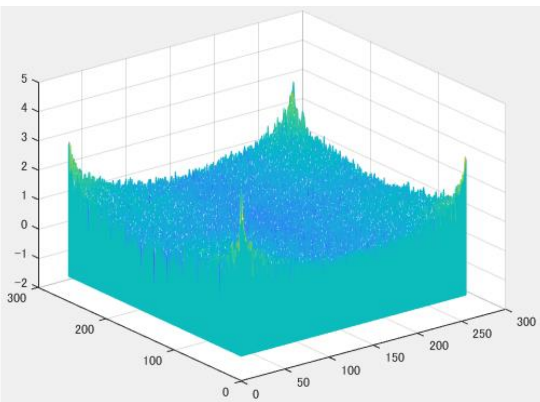
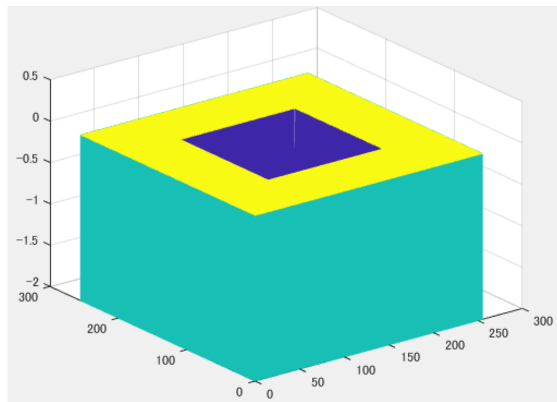
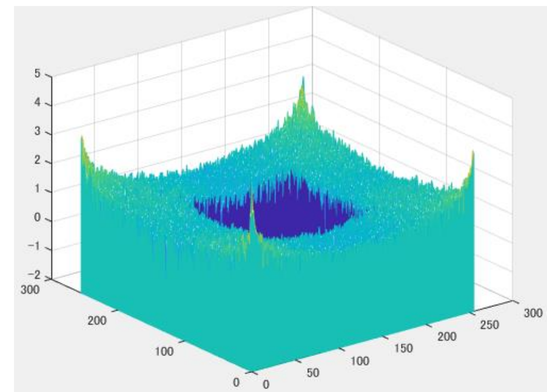
誤差画像

画像端で誤差は生じるが
それ以外ではほぼ同値

どのようなものを計算している？

$$\mathcal{F}(J) = \overline{\mathcal{F}(K)} \circ \mathcal{F}(I)$$

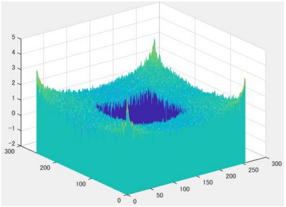
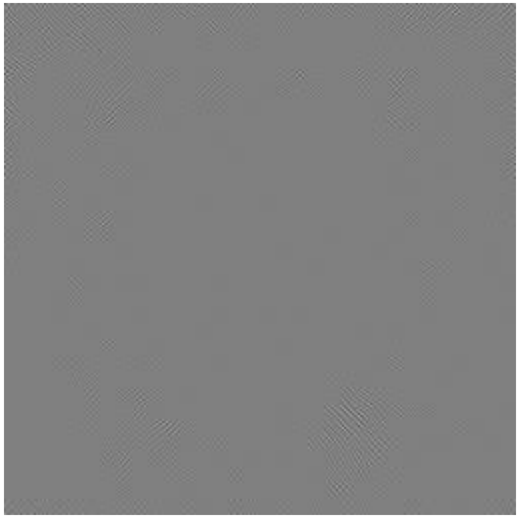
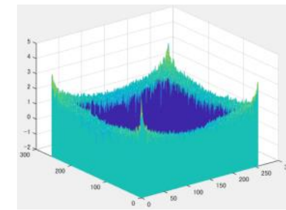
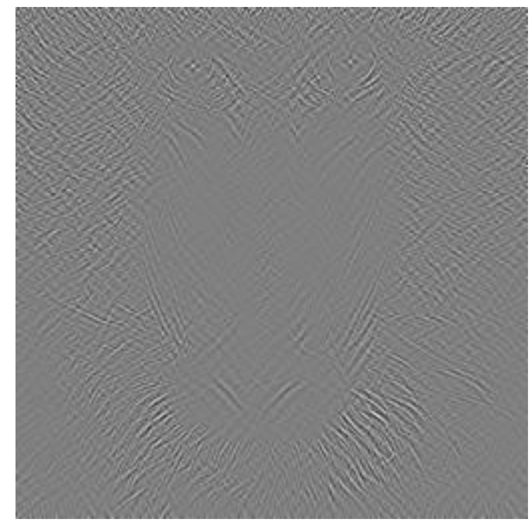
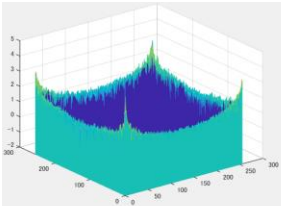
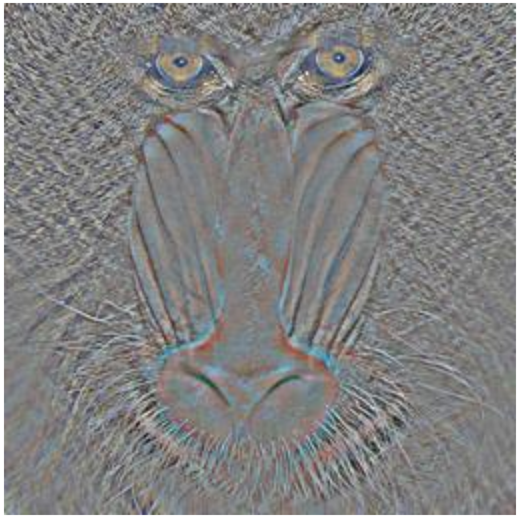
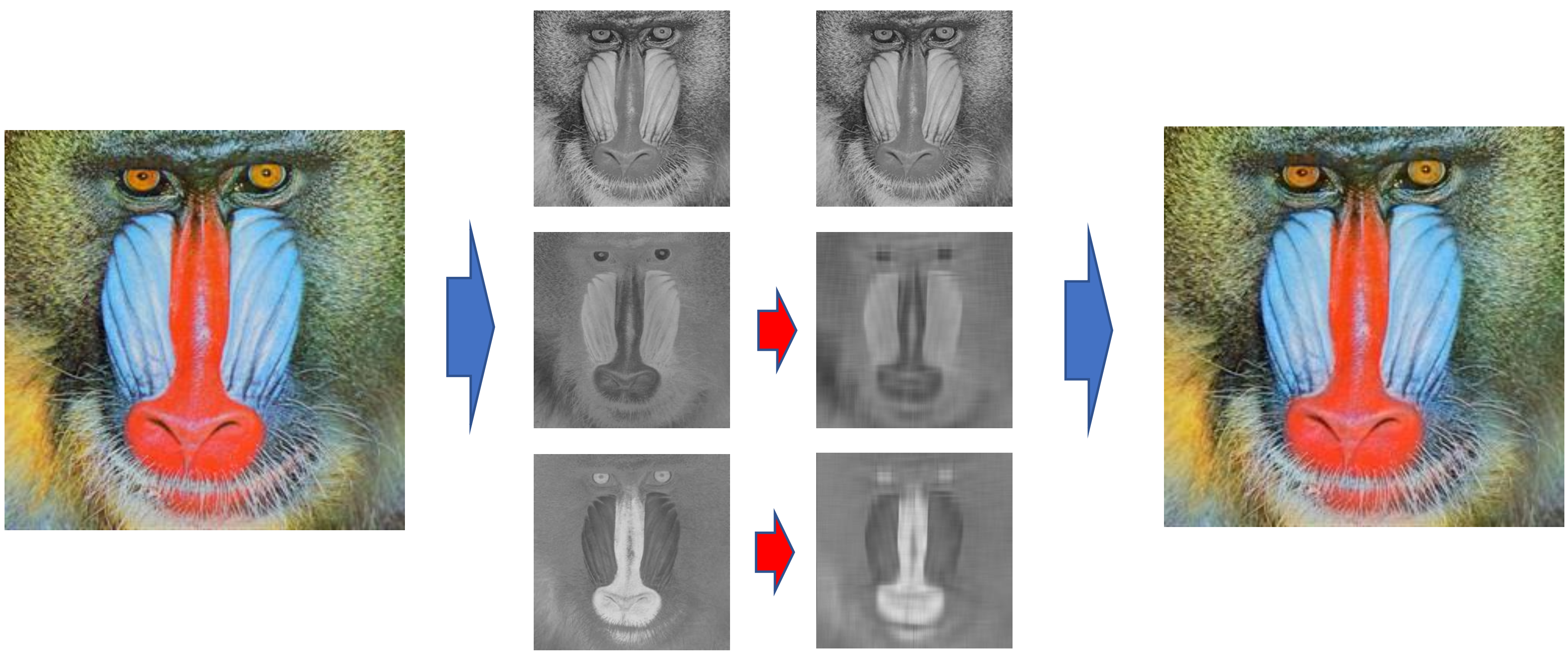
- ```
  % 振幅スペクトルを表示
  c = 1; % 赤色成分のみ
  FI_abs = abs( FI(:,:,c) );
  FJ_abs = abs( FJ(:,:,c) );
  FK_abs = abs( conj(FK) );

  DB = @(X) log10( X + 0.01 ); % 対数表示

  figure(5),  meshz( DB( FI_abs ) );
  figure(6),  meshz( DB( FK_abs ) );
  figure(7),  meshz( DB( FJ_abs ) );
  ```

微小値 0.01 の足しこみは
log(0) = $-\infty$ の防止用
eps(1) を足しても良い
  ```

処理結果



画像

○



フィルタ
（伝達関数）

=



処理画像



*

-1	0	1
-2	0	2
-1	0	1

=

伝達関数の見方





中央が高周波

4 隅が低周波

上から見た図

ただし，信号処理の教科書では，中央が低周波となるように表示することが多い

データ圧縮

人間が知覚できないデータの削除

周波数成分をどこまで削れるか？

- 高周波数帯域を遮断するフィルタを施すと処理画像はどうなるか？



画像

○



フィルタ

=



処理画像

逆フーリエ変換

?

プログラム

```
rate = 0.5;
dy = round(sy/2*rate); % 低周波数をどの程度残すか（幅）
dx = round(sx/2*rate);

FK_cut = zeros(sy,sx);
FK_cut( 1:dy, : ) = 1;  FK_cut( end-dy+1:end, : ) = 1;
FK_cut( :, 1:dx ) = 1;  FK_cut( :, end-dx+1:end ) = 1;

% 作成したフィルタの表示
figure(8), meshz( DB( FK_cut ) );

% フィルタリング
FI_cut = FI .* FK_cut;
figure(9), meshz( DB( abs( FI_cut(:,:,c) ) ) );

% 逆フーリエ変換
I_cut = real( ifft2( FI_cut ) );
figure(10), imshow( [I, I_cut, I-I_cut+0.5] );
```

処理結果（１／３）

- 周波数領域の $(1/2)^2$ を除去したもの
  - 一見ではほぼ同じ，比べれば分かる



原画像

処理結果



誤差

処理結果（２／３）

- 周波数領域の $(4/5)^2$ を除去したもの
  - 模様が荒くなったのが分かる



原画像

処理結果



誤差

処理結果（３／３）

- 周波数領域の $(19/20)^2$ を除去したもの
  - 流石に劣化したのが分かる



原画像

処理結果



誤差

色変換との組み合わせ

- 輝度の変化は知覚しやすいが,
  色の変化は知覚しにくい
  - YCbCr 色変換後,
    フーリエ変換を行い，CbとCr を削っても
    知覚されにくい



プログラム

- ```
  % 色変換
  Iycc = rgb2ycbcr( I );

  % フィルタ（Cb,Cr のみカット）
  FK_ycc = cat( 3, ones(sy,sx), FK_cut, FK_cut );

  % フィルタリング
  FIycc      = fft2( Iycc );
  FIycc_cut = FK_ycc .* FIycc;
  Iycc_cut  = real( ifft2( FIycc_cut ) );

  % 色変換
  I_cut = ycbcr2rgb( Iycc_cut );

  figure(11), imshow( [I, I_cut, I-I_cut+0.5] );
  figure(12), imshow( Iycc(:,:) );
  figure(13), imshow( Iycc_cut(:,:) );
  ```
  ```

逆畳み込み
画像復元

畳み込み結果を，畳み込み前の状態に戻す

フィルタ結果を戻す

- フィルタ処理で完全に除去されていなければもとに戻せる可能性がある

畳み込み演算の順方向計算

$$J(x, y) = \sum_{u,v} K(u, v) I(x-u, y-v)$$
$$:= (K \otimes I)(x, y)$$

K が既知であれば,

$$J \to I$$

を求めることは可能.

ただし，畳み込みのような式ではかけない.

フーリエ変換時の式に着目

- 掛け算　の反対は　割り算

畳み込み演算の順方向計算

$$J(x,y) = \sum_{u,v} K(u,v) I(x-u, y-v)$$
$$:= (K \otimes I)(x,y)$$

$$\mathcal{F}(J)(\xi,\eta) = \mathcal{F}(K)(\xi,\eta) \circ \mathcal{F}(I)(\xi,\eta)$$

$$\mathcal{F}(I)(\xi,\eta) = \frac{\mathcal{F}(J)(\xi,\eta)}{\mathcal{F}(K)(\xi,\eta)}$$

$$I(x,y) = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(J)(\xi,\eta)}{\mathcal{F}(K)(\xi,\eta)}\right)$$

0 割防止用の微修正

- 0 が分母に現れると，**1/0 = 無限大**となり，
  その影響が逆フーリエ変換時に，画像全体に広がる．

  これを避ける

$$I(x, y) = \mathcal{F}^{-1}\left(\frac{\mathcal{F}(J)(\xi, \eta)}{\mathcal{F}(K)(\xi, \eta)}\right)$$

共役複素数を
分子と分母にかける
分母は必ず 0 以上になる

$$I(x, y) = \mathcal{F}^{-1}\left(\frac{\overline{\mathcal{F}(K)(\xi, \eta)} \circ \mathcal{F}(J)(\xi, \eta)}{\overline{\mathcal{F}(K)(\xi, \eta)} \circ \mathcal{F}(K)(\xi, \eta) + \varepsilon}\right)$$

0 割を防止するため，
微小値 ε を加える

プログラム

```
K = fspecial( 'motion', 10, 30 );
J = imfilter( I, K, 'conv', 'replicate' );

% deconvolution in FFTed domain
FJ = fft2( J );
FK = psf2otf( K, [sy,sx] );

FI_deconv = (conj(FK).*FJ) ./ (conj(FK).*FK + 0.001);
I_deconv = real( ifft2( FI_deconv ) );


figure(14), imshow( [J, I_deconv, I] );
```

処理結果

- 完全に戻すことは難しいが，
  ある程度，もとに戻ることが分かる．

ぼけ画像

逆畳み込み結果

原画像（正解画像）

補足：縞模様はなんなのか？

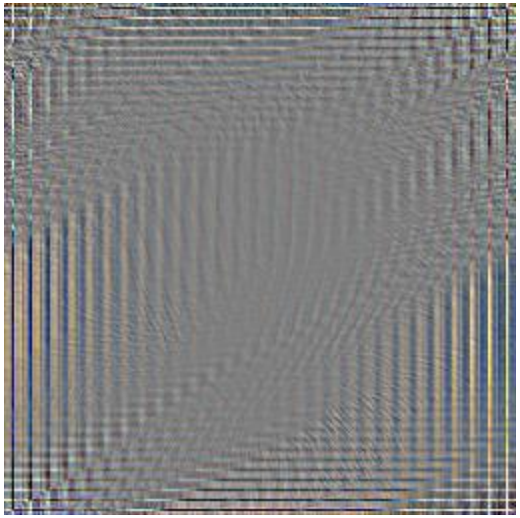
- リンギングと呼ばれる人工的な模様 (artifact)

逆畳み込み結果

−

原画像（正解画像）

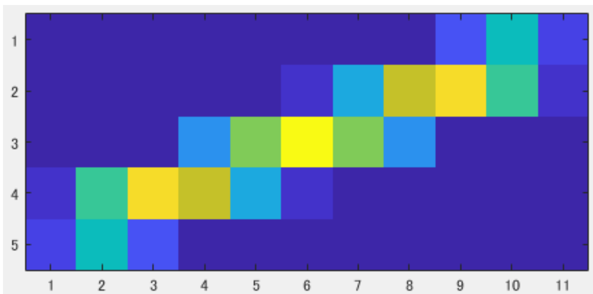
＝



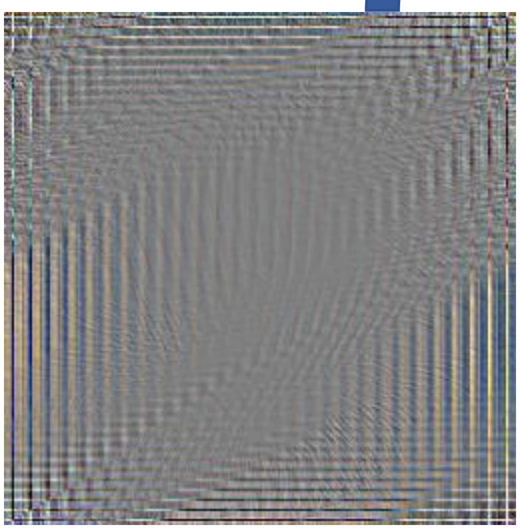
リンギング

補足： PSF を畳み込むと 0 になる

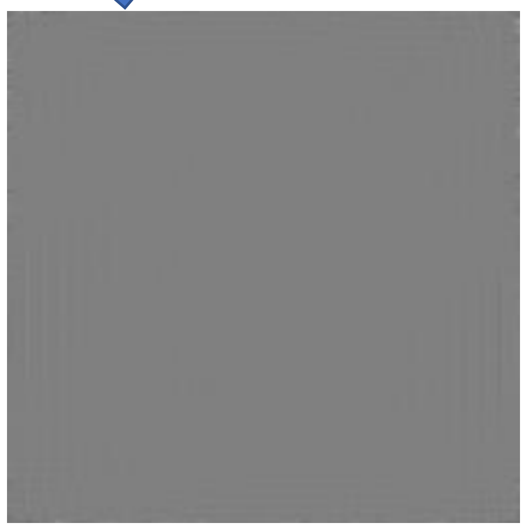
PSF



畳み込み



リンギング



ぼけ画像

$$R = I - \hat{I}$$

$$R \otimes K = 0$$

フーリエ変換

$$\mathcal{F}(R)(\xi, \eta) \circ \mathcal{F}(K)(\xi, \eta) = 0$$

K が 0 となる箇所に R は値を持ち,
K が値を持つ箇所で R は 0 となる

プログラム

```
R = I - I_deconv;
figure(15), imshow( R + 0.5 );

R_conv = imfilter( R, K, 'conv', 'replicate' );
figure(16), imshow( R_conv + 0.5 );

s = 5;
J_r = imfilter( I + s*R, K, 'conv', 'replicate' );
figure(17), imshow( [J, J_r] );
```

なぜ起こる？　防げる？

- PSF のフーリエ変換時に，分母に 0 が現れるのが原因
- この計算式だけでは防げない．
  鮮鋭な画像の特徴を表す別の式を加えれば，低減できる

$$R \otimes K = 0$$

となるリンギングがある場合

$$B = (I + sR) \otimes K$$

s 倍したリンギングを加えた
画像を畳み込むと
同じ処理結果が得られる

$$B = I \otimes K + s\underbrace{(R \otimes K)}_{0}$$

方程式の解が一意に定まらない
状態を意味する