

Sample 11-5

画像ノイズ除去

事前分布

画像処理特論

村松 正吾

動作確認: MATLAB R2023a

Image denoising

Prior distribution

Advanced Topics in Image Processing

Shogo MURAMATSU

Verified: MATLAB R2023a

準備

(Preparation)

```
clear
close all
import msip.download_img
msip.download_img
```

```
kodim01.png already exists in ./data/
kodim02.png already exists in ./data/
kodim03.png already exists in ./data/
kodim04.png already exists in ./data/
kodim05.png already exists in ./data/
kodim06.png already exists in ./data/
kodim07.png already exists in ./data/
kodim08.png already exists in ./data/
kodim09.png already exists in ./data/
kodim10.png already exists in ./data/
kodim11.png already exists in ./data/
kodim12.png already exists in ./data/
kodim13.png already exists in ./data/
kodim14.png already exists in ./data/
kodim15.png already exists in ./data/
kodim16.png already exists in ./data/
kodim17.png already exists in ./data/
kodim18.png already exists in ./data/
kodim19.png already exists in ./data/
kodim20.png already exists in ./data/
kodim21.png already exists in ./data/
kodim22.png already exists in ./data/
kodim23.png already exists in ./data/
kodim24.png already exists in ./data/
See Kodak Lossless True Color Image Suite
```

パラメータ設定

(Parameter settings)

- sgm: ノイズ標準偏差 σ_w (Standard deviation of noise)
- nlevels: ウェーブレット段数 (Wavelet levels)

```
% Parameter settings
nlevels = 3;
```

画像の読込

(Read image)

```
img = "kodim23";
u = im2double(imread("./data/" + img + ".png"));
if size(u,3) == 3
    u = rgb2gray(u);
end
```

分析処理

(Analysis process)

直交ウェーブレット変換 Symlet を利用. (Uses Symlet, which is an orthogonal wavelet transform.)

```
% Preperation of filters for wavelets
iswtb = license('checkout','wavelet_toolbox');
if iswtb % Functions in Wavelet Toolbox are used
    dwtmode('per')
    wname = "sym4";
    [h0,h1,f0,f1] = wfilters(wname);
    %save(['./data/' char(wname) '.mat'],'h0','h1','f0','f1')
else
    import msip.ezwavedec2
    import msip.ezwaverec2
    S = load('./data/sym4.mat');
    h0 = S.h0;
    h1 = S.h1;
    f0 = S.f0;
    f1 = S.f1;
    clear H F
    % Analysis bivariate filters
    H.h00 = h0(:)*h0(:).';
    H.h01 = h0(:)*h1(:).';
    H.h10 = h1(:)*h0(:).';
    H.h11 = h1(:)*h1(:).';
    % Synthesis bivariate filters
    F.f00 = f0(:)*f0(:).';
    F.f01 = f0(:)*f1(:).';
    F.f10 = f1(:)*f0(:).';
```

```
F.f11 = f1(:)*f1(:).';
end
```

```
*****
** DWT 拡張モード: 周期化 **
*****
```

分析処理 (Analysis process)

```
if iswtb
    [coefs,scales] = wavedec2(u,nlevels,h0,h1);
    % Reconstruction to check PR
    r = waverec2(coefs,scales,f0,f1);
else
    [coefs,scales] = ezwavedec2(u,nlevels,H);
    % Reconstruction to check PR%
    r = ezwaverec2(coefs,scales,F);
end
assert(norm(u-r,"fro")^2/numel(u)<1e-18,'Perfect reconstruction is violated.')
```

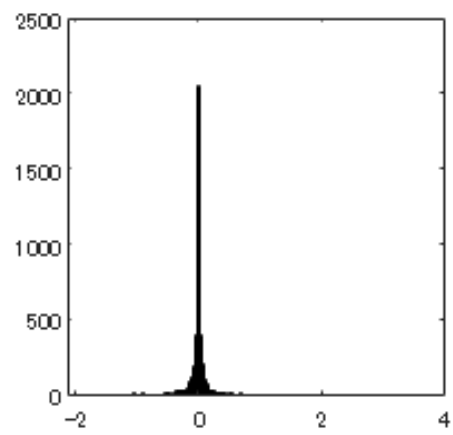
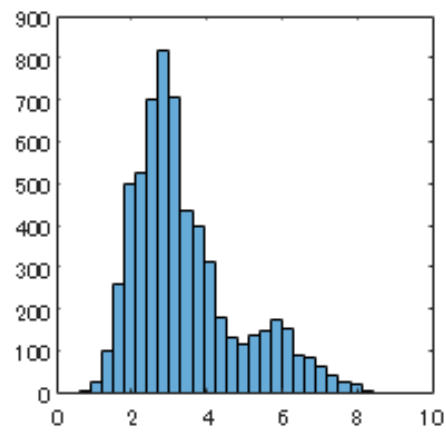
変換係数の抽出 (Extraction of coefficients)

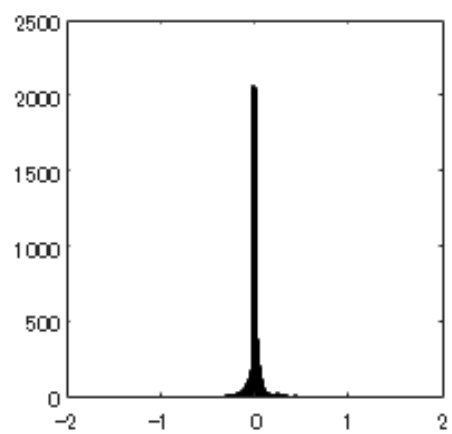
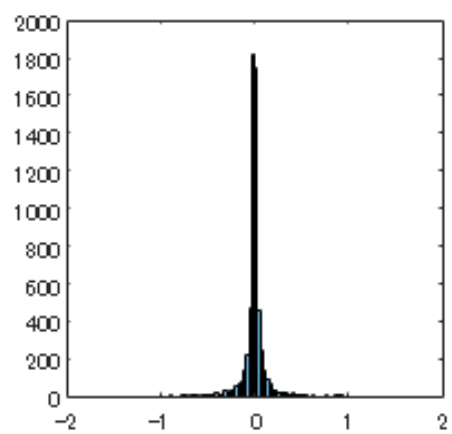
```
s = extractcoefs(coefs,scales);
```

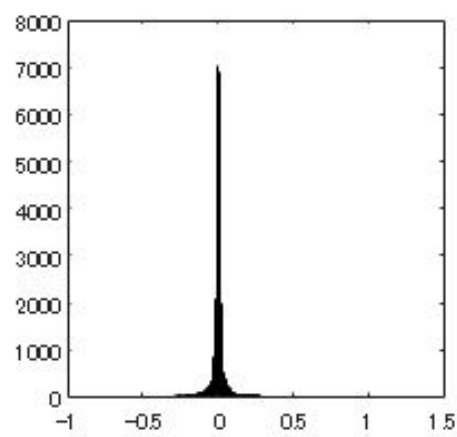
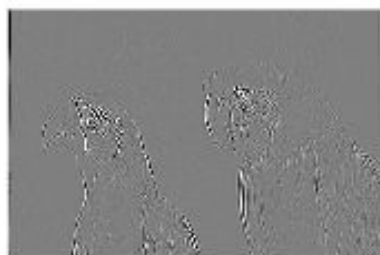
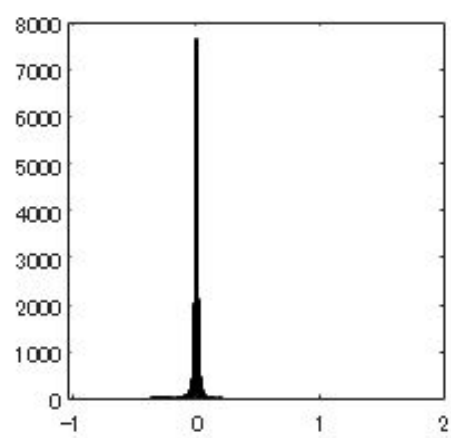
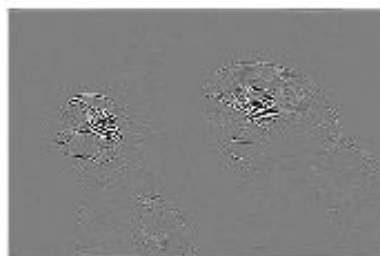
変換係数の分布

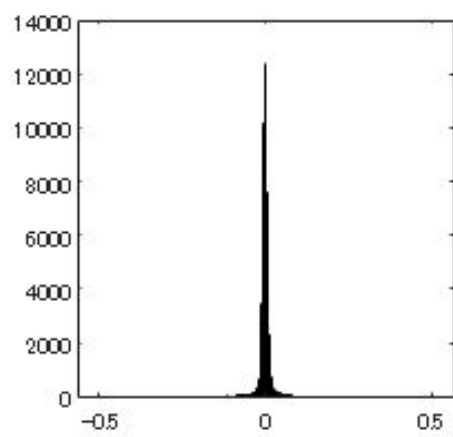
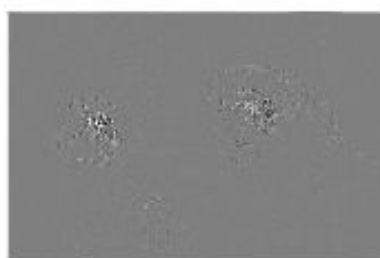
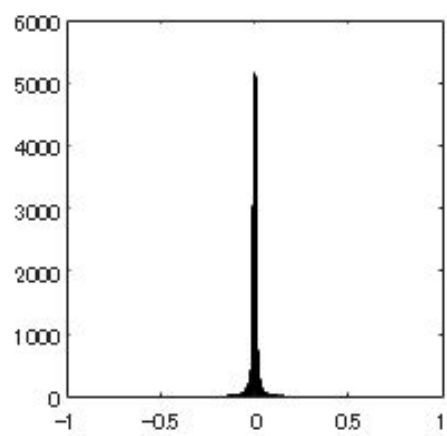
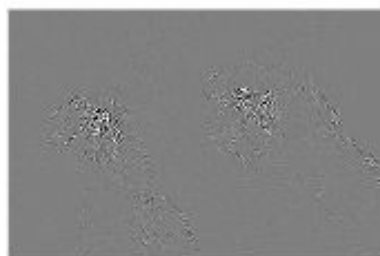
(Distribution of Coefs.)

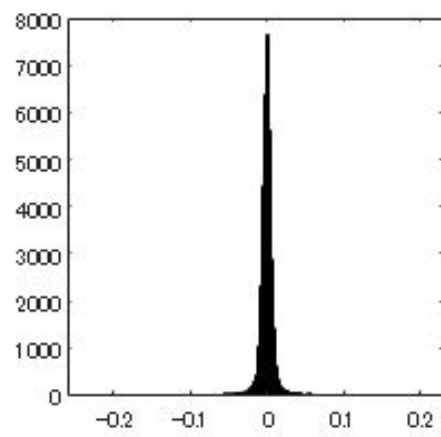
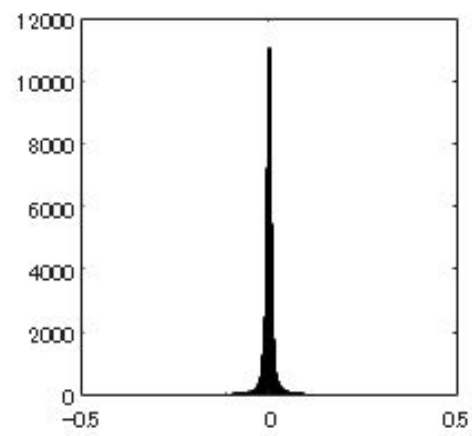
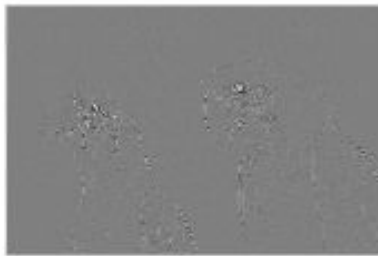
```
nchs = length(s);
for ich = 1:nchs
    figure
    subplot(1,2,1)
    if ich == 1
        imshow(s{ich}*pow2(-nlevels))
    else
        imshow(s{ich}+.5)
    end
    subplot(1,2,2)
    histogram(s{ich}(:))
    axis square
    drawnow
end
```











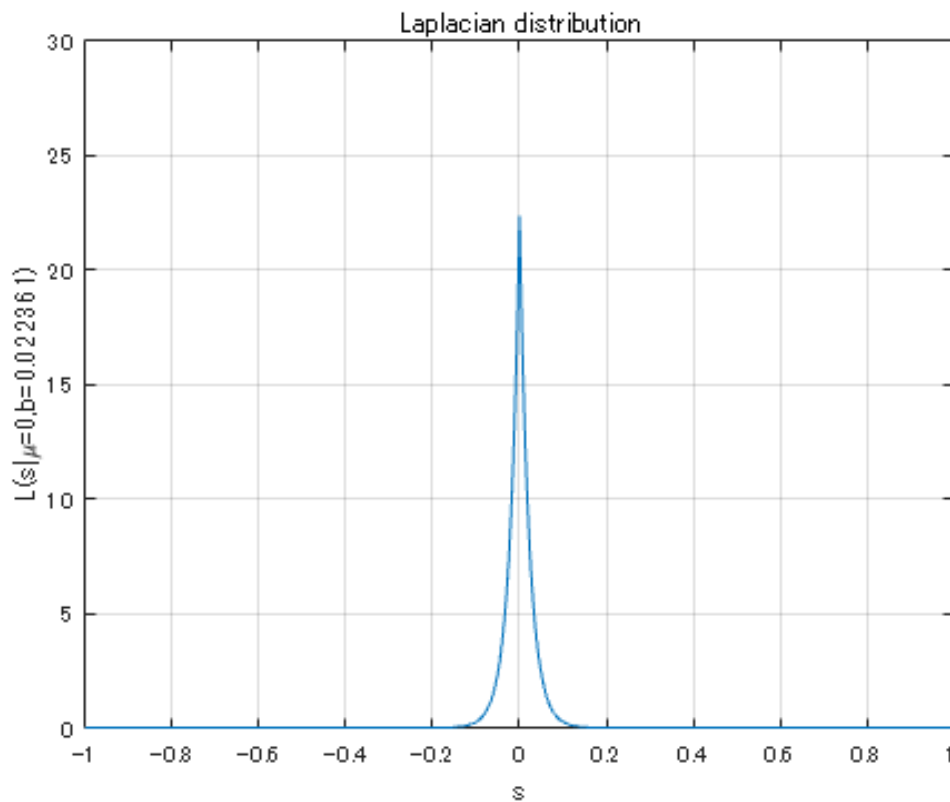
ラプラス分布

(Laplace distribution)

変換係数はラプラス分布に従う乱数と仮定. (Coefs. are assumed to be random numbers drawn from a Laplace distribution.)

- $s \sim \text{Lap}(s|\mu = \mathbf{0}, b)$
- $\text{Lap}(s|\mu = \mathbf{0}, b) = \frac{1}{2b} \exp\left(-\frac{\|s - \mu\|_1}{b}\right)$

```
% Laplacian parameters
mu = 0;
sgm2 = 0.001;
b = sqrt(sgm2/2);
% Laplacian distribution
spdf = @(x) 1/(2*b)*exp(-abs(x-mu)/b);
figure
h = fplot(spdf);
xlabel('s')
ylabel(['L(s|\mu=0,b=' num2str(b) ')'])
title('Laplacian distribution')
grid on
axis([-1 1 0 30])
```



観測画像

(Observation image)

パラメータ設定 (Parameter settings)

```
sgmuint8 = 30;  
sgm = sgmuint8/255;
```

ノイズ付加 (Add noise)

```
v = imnoise(u, 'gaussian', 0, sgm^2);
```

分析処理 (Analysis process)

```
if iswtb  
    [coefs,scales] = wavedec2(v,nlevels,h0,h1);  
else  
    [coefs,scales] = ezwavedec2(v,nlevels,H);  
end
```

ウェーブレット縮退処理

(Wavelet shrinkage)

問題設定 (Problem settings)

- $\hat{\mathbf{s}} = \arg \min_{\mathbf{s}} \frac{1}{2} \|\mathbf{v} - \mathbf{D}\mathbf{s}\|_2^2 + \lambda \|\mathbf{s}\|_1$
- $\mathbf{D}\mathbf{D}^T = \mathbf{D}^T\mathbf{D} = \mathbf{I}$ (Orthonormal)

パラメータ設定 (Parameter settings)

```
isbayesshrink = false;  
if ~isbayesshrink  
    isaprxleft = true;  
    lambda = 10^-0.6  
end
```

```
lambda = 0.2512
```

ソフト閾値処理 (Soft-thresholding)

$$\hat{\mathbf{s}} = \mathcal{T}_{\lambda}(\mathbf{s}) = \text{sign}(\mathbf{s}) \odot \max(\text{abs}(\mathbf{s}) - \lambda \mathbf{1}, \mathbf{0})$$

% サブバンド適応ソフト縮退処理

```
if isbayesshrink  
    import msip.bayesshrink  
    coefs = bayesshrink(coefs,scales);  
else  
    if isaprxleft  
        mask = ones(size(coefs));  
        mask(1:prod(scales(1,:))) = 0;
```

```

        lambda = lambda * mask;
    end
    softshrink = @(x) sign(x).*max(abs(x)-lambda,0);
    coefs = softshrink(coefs);
end

```

合成处理 (Synthesis process)

```

if iswtb
    r = waverec2(coefs,scales,f0,f1);
else
    r = ezwaverec2(coefs,scales,F);
end

```

画像表示

(Image show)

```

figure
imshow(u)
title('Original image u')

```

Original image u



```

figure
imshow(v)

```

```
title(sprintf('Noisy image v : PSNR = %5.2f [dB]',psnr(u,v)))
```

Noisy image v: PSNR = 18.72 [dB]



```
figure  
imshow(r)  
title(sprintf('Denoised image r : PSNR = %5.2f [dB]',psnr(u,r)))
```

Denoised image r:PSNR = 28.75 [dB]



ウェーブレット画像ノイズ除去関数

(Wavelet image denoising function)

参考資料 (Reference)

```
if iswtb
    help wthcoef2
    help wdenoise2
end
```

wthcoef2 - 2-D wavelet coefficient thresholding

This MATLAB function returns the horizontal, vertical, or diagonal coefficients obtained from the wavelet decomposition structure [C,S] by soft or hard thresholding defined in vectors N and T.

構文

```
NC = wthcoef2('type',C,S,N,T,SORH)
NC = wthcoef2('type',C,S,N)
NC = wthcoef2('a',C,S)
NC = wthcoef2('t',C,S,N,T,SORH)
```

入力引数

'type' - Type of coefficients
'h' | 'v' | 'd'
C - Wavelet decomposition vector
real-valued vector

S - Bookkeeping matrix
integer-valued matrix
N - Threshold vector
 $1 \leq N(i) \leq \text{size}(S,1)-2$
T - Threshold vector
nonnegative vectors
SORH - Soft or hard threshold
's' | 'h'

出力引数

NC - Wavelet coefficient threshold
real-valued vector

例

Calculate Coefficients Obtained From Wavelet Decomposition Structure

参考 wthcoef, wavedec2, wthresh

R2006a より前に Wavelet Toolbox で導入
wthcoef2 のドキュメンテーション

wdenoise2 - Wavelet image denoising

This MATLAB function denoises the grayscale or RGB image IM using an empirical Bayesian method.

構文

```
IMDEN = wdenoise2(IM)
IMDEN = wdenoise2(IM,LEVEL)
[IMDEN,DENOISEDCFS] = wdenoise2(____)

[IMDEN,DENOISEDCFS,ORIGCFS] = wdenoise2(____)
[IMDEN,DENOISEDCFS,ORIGCFS,S] = wdenoise2(____)
[IMDEN,DENOISEDCFS,ORIGCFS,S,SHIFTS] = wdenoise2(____)

[____] = wdenoise2(____,Name,Value)

wdenoise2(____)
```

入力引数

IM - Input image
real-valued 2-D matrix | real-valued 3-D array
LEVEL - Wavelet decomposition level
positive integer

名前と値の引数

Wavelet - Name of wavelet
'bior4.4' (既定の設定) | character vector | string scalar
DenoisingMethod - Denoising method
'Bayes' (既定の設定) | 'FDR' | 'Minimax' | 'SURE' |
'UniversalThreshold'
ThresholdRule - Threshold rule
'Hard' | 'Soft' | 'Mean' | 'Median'
NoiseEstimate - Method of estimating variance of noise
'LevelIndependent' (既定の設定) | 'LevelDependent'
NoiseDirection - Wavelet subbands
['h','v','d'] (既定の設定) | string vector | scalar string
CycleSpinning - Number of circular shifts
0 (既定の設定) | nonnegative integer
ColorSpace - Color space
'PCA' (既定の設定) | 'Original'

出力引数

IMDEN - Denoised image
real-valued matrix
DENOISEDCFS - Scaling and denoised wavelet coefficients

real-valued matrix
ORIGCFS - Scaling and wavelet coefficients
real-valued matrix
S - Bookkeeping matrix
integer-valued matrix
SHIFTS - Image shifts
integer-valued matrix

例

Denoise Grayscale Image Using Default Settings
Denoise Color Image Using Cycle Spinning
Denoise Image Using Specific Subband

参考 wdenoise, wavedec2

R2019a の Wavelet Toolbox で導入
wdenoise2 のドキュメンテーション

変換係数の抽出

(Extraction of Coefs.)

```
function s = extractcoefs(coefs,scales)
nscales = size(scales,1)-1;
s = cell(3*(nscales-1)+1,1);
sidx = 1;
ndims = scales(1,:);
eidx = sidx + prod(ndims) - 1;
s{1} = reshape(coefs(sidx:eidx),ndims);
sidx = eidx + 1;
ich = 2;
for iscale = 2:nscales
    ndims = scales(iscale,:);
    for iband = 1:3
        eidx = sidx + prod(ndims) - 1;
        s{ich} = reshape(coefs(sidx:eidx),ndims);
        sidx = eidx + 1;
        ich = ich + 1;
    end
end
end
```

© Copyright, Shogo MURAMATSU, All rights reserved.