Lab 6: Spatial Filters

Shido Nakajima

03/10/2023

03/10/2023

OBJECTIVE: The objective of this lab exercise was to learn and understand the basis and application of spatial filters utilizing kernels. Additionally, the function of zero padding, as well as different values for the kernels were observed to further understand their effects.

METHODS: Part 1 was simply a tutorial.

For part 2, few codes were copied from the lab to achieve the 6 images of different filtering methods of imfilter() function. Then for the question in section b regarding normalizing the filter by dividing by 45, the number 45 came from the sum of all of the elements of the 3x3 filter (1+2+3+4+5+6+7+8+9 = 45). Correlation and Convolution options were compared on the same figure. While the values are different among the two, I was unable to observe significant differences between the two methods. For section c, same and full options of imfilter() function were compared, where I was again unable to recognize significant difference.

For part 3, 3 different options for fspecial() function were compared, where, unlike the previous sections, I was able to recognize obvious differences between the four.

For part 4, fspecial() function was used to achieve the three kernels, which I decided to use an averaging box kernel. Then, for zero padding, instead of creating the zero borders and adding them onto the image, I made a zero image of the size of the padded image, which I replaced the center part with the original image. Then, using for-loops for 3x3, 5x5, and 7x7 kernels, I applied each kernels onto each of the images. For the neutral bone x-ray image, I used 3x3 and 5x5 kernels for comparison. For the bone with noise 1 image, I used 3x3 and 7x7 kernels for comparison. For the bone with noise 2 image, I used 5x5 and 7x7 kernels for comparison. For section 4.2 and 4.3, I copied the same methods with only altering the values of the kernel. Instead of using fspecial() function, I decided to hard code the values.

RESULTS: In part 4, images with different sized kernels applied are placed on the same figure side-by-side for easier comparison. It can be observed that larger kernels would lead to more blurred image, which is very visible with the images with noise introduced. Additionally, since larger kernels require larger paddings, larger kernel images have very obvious band of dark around the border.

In section 4.2, the kernel made the image much darker compared to the original and blurring the image in general. It made the image very difficult to distinguish, especially with the noise 1 introduced. In section 4.3, the kernel made the image intensities slightly more towards the gray while smoothening the image overall. This filter would be useful in smoothening images with too high contrast.

CONCLUSION: These exercises allowed me to understand the basis of special filtering using kernels and zero padding. While the differences and benefits of some of the filtering methods may not be obvious, in applications that require smoothening or noise reduction, special filtering is extremely useful. This was especially noticeable in exercise 4.1 with noise 1, where the original image was relatively difficult to see the fracture.

APPENDIX:

Contents


```
clear;clc;close all;
```

2 spcial fintering with the imfilter MATLAB function

```
h = fspecial('average',5);

I = imread('Brain_MRI.tif');
figure, imshow(I)
title('Original Image')

J = imcomplement(I);
figure, imshow(J)
title('Image Complement')

MRI1 = imfilter(J,h);
figure, imshow(MRI1)
title('Zero-Padded')

MRI2 = imfilter(J,h,'replicate');
figure, imshow(MRI2)
title('Replicate');

MRI3 = imfilter(J,h,'symmetric');
figure, imshow(MRI3)
title('Symmetric')

MRI4 = imfilter(J,h,'circular');
figure, imshow(MRI4)
title('Circular')

% the number 45 comes from the sum of the filter values,
% 1+2+3+4+5+6+7+8+9 = 45

I = imread('Brain_MRI_noise.tif');
J = imcomplement(I);
h = [1 2 3 4 5; 6 7 8 9 10; 11 12 13 14 15];
h = h/sum(h(:));
MRI_corr = imfilter(J,h,'corr');
MRI_conv = imfilter(J,h,'conv');
figure, subplot(121), imshow(MRI_corr), title('corr')
subplot(122), imshow(MRI_conv), title('conv')

h = ones(3,3)/9;
MRI_same = imfilter(J,h,'same');
MRI_full = imfilter(J,h,'full');
figure, subplot(121), imshow(MRI_same), title('same')
subplot(122), imshow(MRI_full),title('full')
```

Original Image

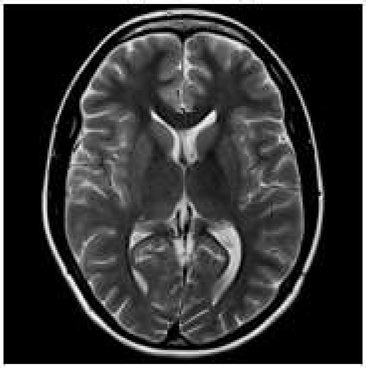
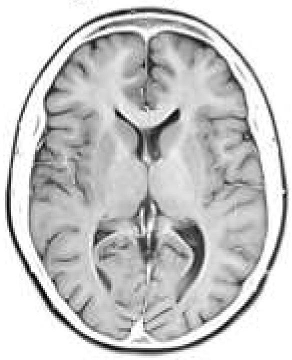
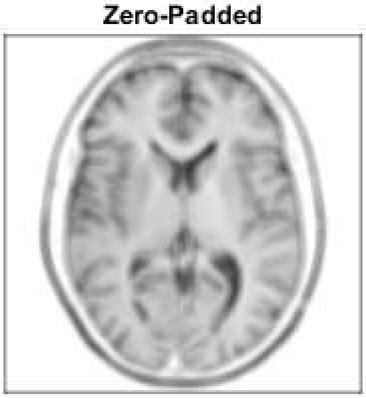


Image Complement



Zero-Padded



Replicate

Symmetric

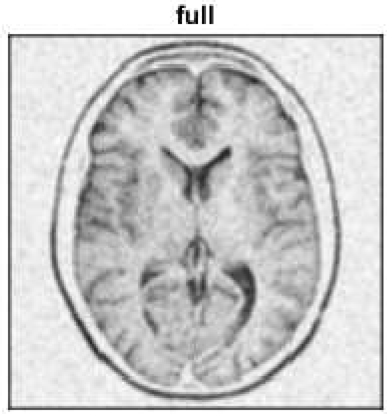
Circular

corr

conv

same

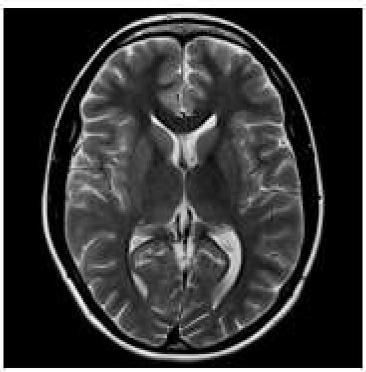
full

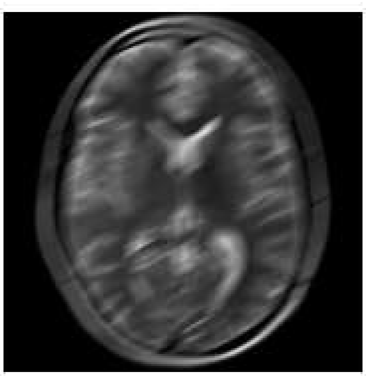


3 predefined filters

Original image

```
I = imread('Brain_MRI.tif');
figure, imshow(I)

% Image motion blur
h = fspecial('motion',10,25);
MRI_motion = imfilter(I,h);
figure, imshow(MRI_motion)

% Image sharpening
h = fspecial('unsharp');
MRI_sharp = imfilter(I,h);
figure, imshow(MRI_sharp)

% Edge detection
h = fspecial('sobel');
MRI_sobel = imfilter(I,h);
figure, imshow(MRI_sobel)
```

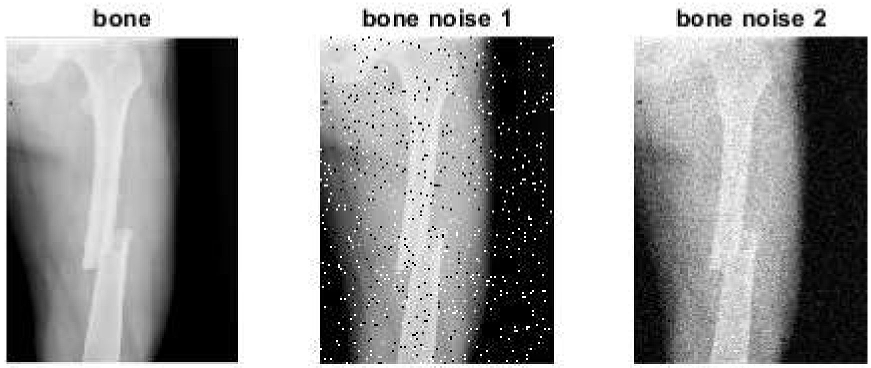


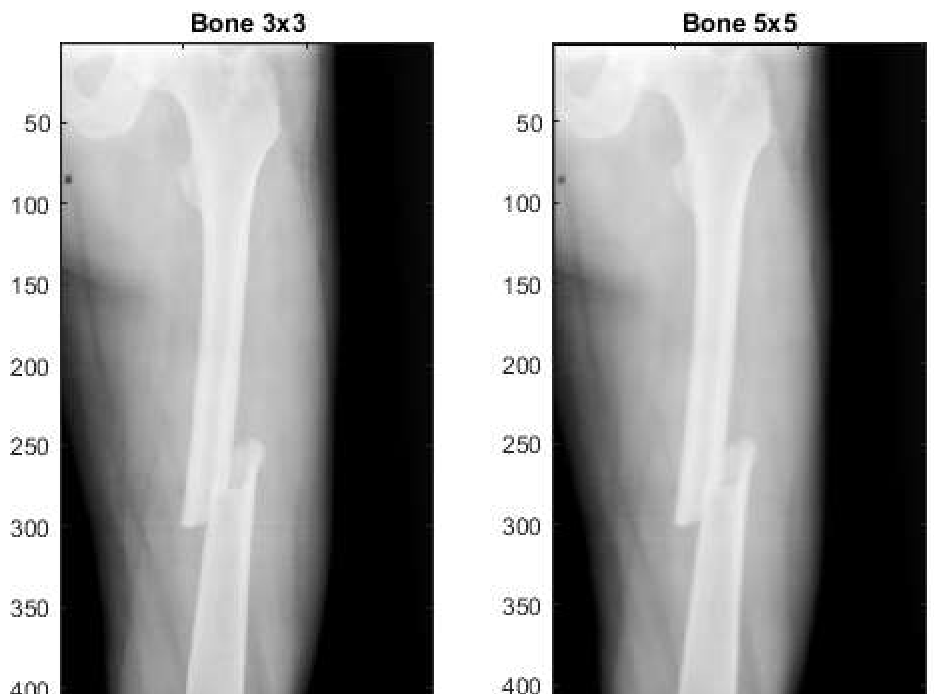


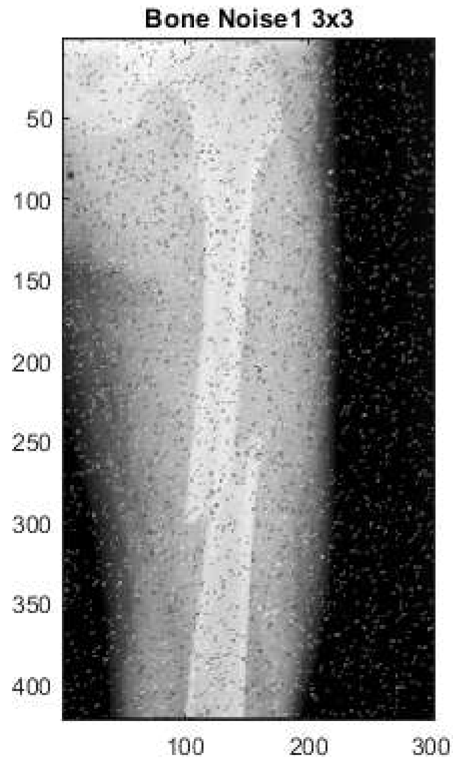
4.1

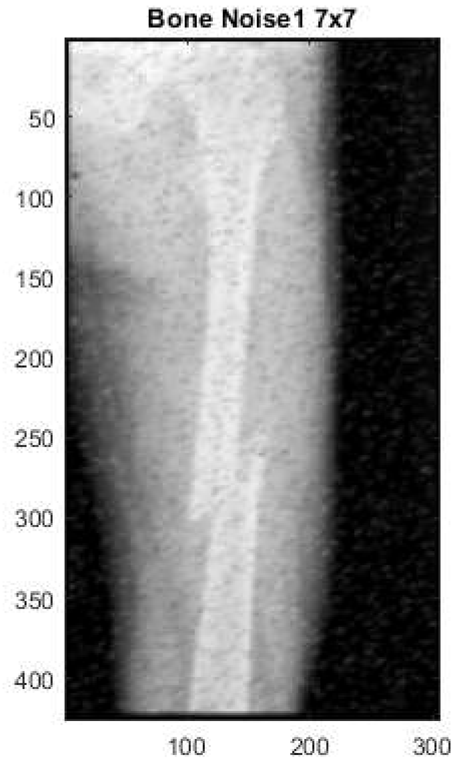
```
h3 = fspecial('average',3);
h5 = fspecial('average',5);
h7 = fspecial('average',7);

IB = imread('Xray_Bone.tif');
IBN1 = imread('Xray_Bone_Noise1.tif');
IBN2 = imread('Xray_Bone_Noise2.tif');

[m,n] = size(IB);

% pad for 3x3 kernel
IB3 = zeros(m+2,n+2);
IB3(2:m+1,2:n+1) = IB;

IBN13 = zeros(m+2,n+2);
IBN13(2:m+1,2:n+1) = IBN1;

% pad for 5x5 kernel
IB5 = zeros(m+4,n+4);
IB5(3:m+2,3:n+2) = IB;

IBN25 = zeros(m+4,n+4);
```

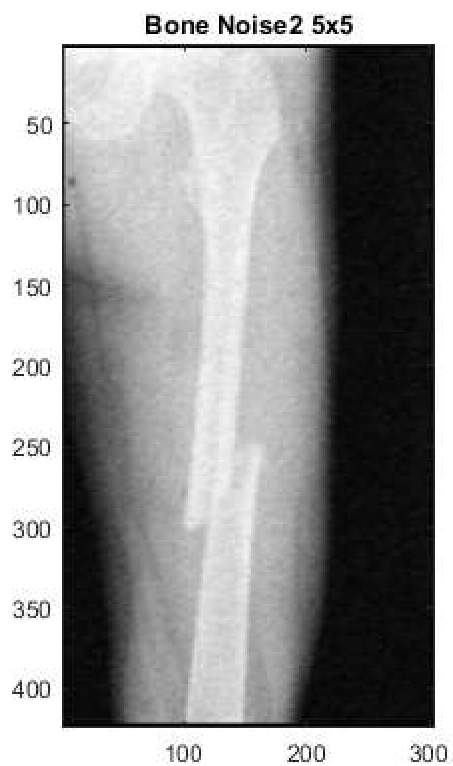
```
IBN25(3:m+2,3:n+2) = IBN2;

% pad for 7x7 kernel
IBN17 = zeros(m+6,n+6);
IBN17(4:m+3,4:n+3) = IBN1;

IBN27 = zeros(m+6,n+6);
IBN27(4:m+3,4:n+3) = IBN2;

% apply filter for 3x3 kernels
[m,n] = size(IB3);
for M=2:m-1
    for N=2:n-1
        B0 = double(IB3(M-1:M+1,N-1:N+1)).*h3;
        IB3(M,N) = mean(B0,'all');
        B0 = double(IBN13(M-1:M+1,N-1:N+1)).*h3;
        IBN13(M,N) = mean(B0,'all');
    end
end

% apply filter for 5x5 kernels
[m,n] = size(IB5);
for M=3:m-2
    for N=3:n-2
        B0 = double(IB5(M-2:M+2,N-2:N+2)).*h5;
        IB5(M,N) = mean(B0,'all');
        B0 = double(IBN25(M-2:M+2,N-2:N+2)).*h5;
        IBN25(M,N) = mean(B0,'all');
    end
end

% apply filter for 7x7 kernels
[m,n] = size(IBN17);
for M=4:m-3
    for N=4:n-3
        B0 = double(IBN17(M-3:M+3,N-3:N+3)).*h7;
        IBN17(M,N) = mean(B0,'all');
        B0 = double(IBN27(M-3:M+3,N-3:N+3)).*h7;
        IBN27(M,N) = mean(B0,'all');
    end
end

figure,subplot(131),imshow(IB),title('bone')
subplot(132),imshow(IBN1),title('bone noise 1')
subplot(133),imshow(IBN2),title('bone noise 2')

figure,subplot(121),imagesc(IB3),colormap('gray'),title('Bone 3x3')
subplot(122),imagesc(IB5),colormap('gray'),title('Bone 5x5')

figure,subplot(121),imagesc(IBN13),colormap('gray'),title('Bone Noise1 3x3')
subplot(122),imagesc(IBN17),colormap('gray'),title('Bone Noise1 7x7')

figure,subplot(121),imagesc(IBN25),colormap('gray'),title('Bone Noise2 5x5')
subplot(122),imagesc(IBN27),colormap('gray'),title('Bone Noise2 7x7')
```

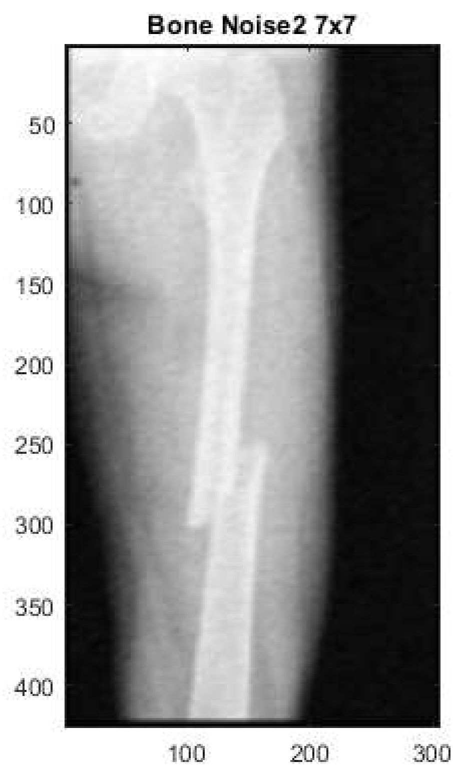

bone
bone noise 1
bone noise 2


Bone 3x3
Bone 5x5
50
100
150
200
250
300
350
400
100
200
300





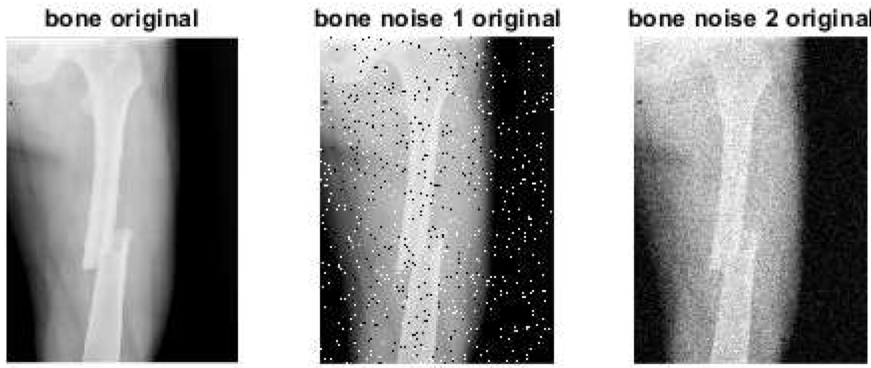


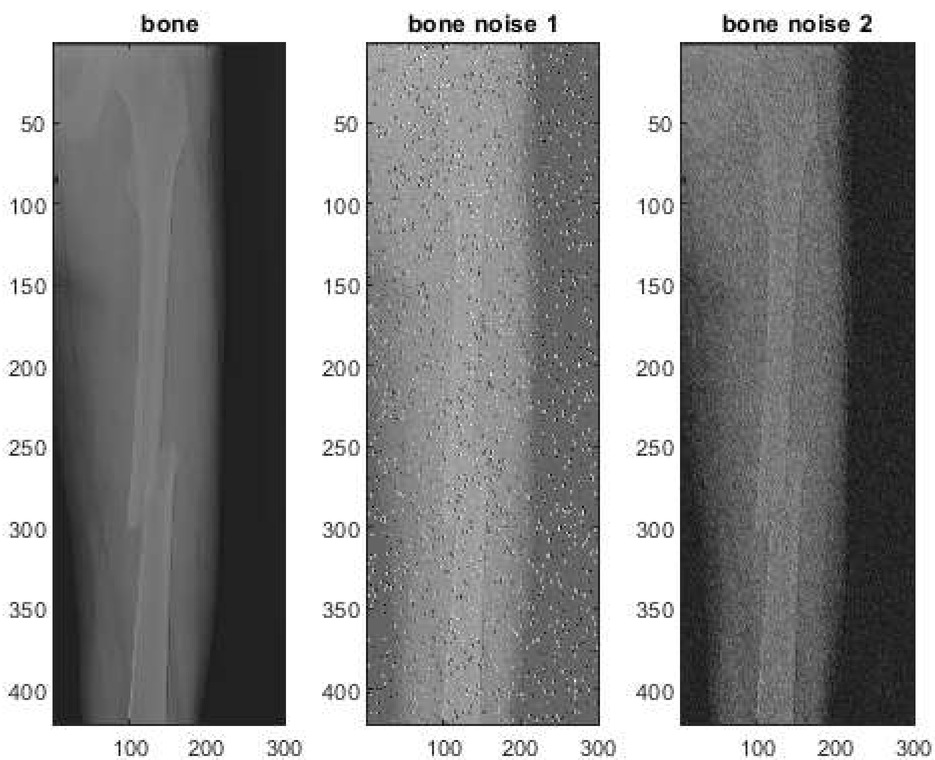


4.2

```
h3 = [-1 0 1;-1 0 1;-1 0 1];

% pad for 3x3 kernel
```

```
[m,n] = size(IB);

IB3 = zeros(m+2,n+2);
IB3(2:m+1,2:n+1) = IB;

IBN13 = zeros(m+2,n+2);
IBN13(2:m+1,2:n+1) = IBN1;

IBN23 = zeros(m+2,n+2);
IBN23(2:m+1,2:n+1) = IBN2;

% apply filter for 3x3 kernels
[m,n] = size(IB3);
for M=2:m-1
    for N=2:n-1
        B0 = double(IB3(M-1:M+1,N-1:N+1)).*h3;
        IB3(M,N) = mean(B0,'all');
        B0 = double(IBN13(M-1:M+1,N-1:N+1)).*h3;
        IBN13(M,N) = mean(B0,'all');
        B0 = double(IBN23(M-1:M+1,N-1:N+1)).*h3;
        IBN23(M,N) = mean(B0,'all');
    end
end

figure,subplot(131),imshow(IB),title('bone original')
subplot(132),imshow(IBN1),title('bone noise 1 original')
subplot(133),imshow(IBN2),title('bone noise 2 original')

figure,subplot(131),imagesc(IB3),colormap('gray'),title('bone')
subplot(132),imagesc(IBN13),colormap('gray'),title('bone noise 1')
subplot(133),imagesc(IBN23),colormap('gray'),title('bone noise 2')
```

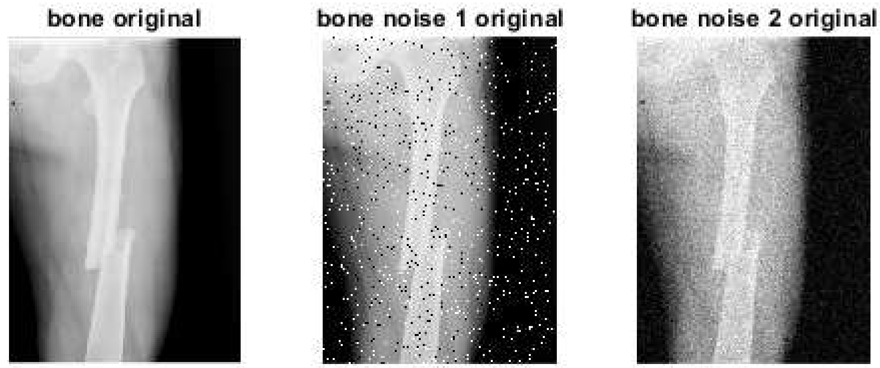


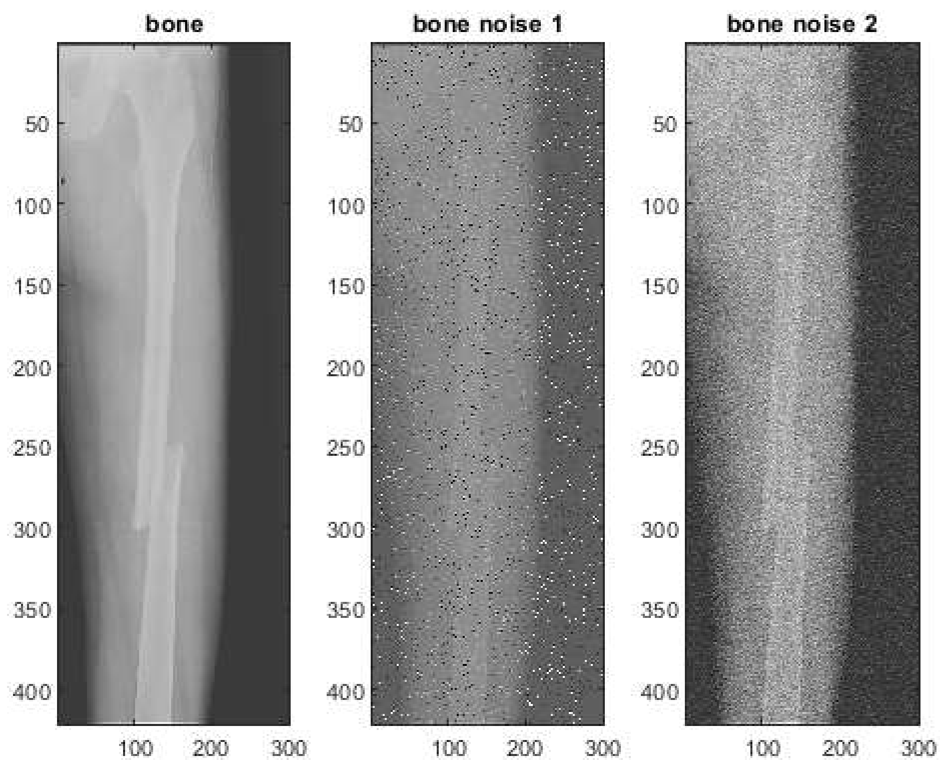


4.3

```
h3 = [0 -1 0;-1 4 -1;0 -1 0];

% pad for 3x3 kernel
```

```matlab
[m,n] = size(IB);

IB3 = zeros(m+2,n+2);
IB3(2:m+1,2:n+1) = IB;

IBN13 = zeros(m+2,n+2);
IBN13(2:m+1,2:n+1) = IBN1;

IBN23 = zeros(m+2,n+2);
IBN23(2:m+1,2:n+1) = IBN2;

% apply filter for 3x3 kernels
[m,n] = size(IB3);
for M=2:m-1
    for N=2:n-1
        B0 = double(IB3(M-1:M+1,N-1:N+1)).*h3;
        IB3(M,N) = mean(B0,'all');
        B0 = double(IBN13(M-1:M+1,N-1:N+1)).*h3;
        IBN13(M,N) = mean(B0,'all');
        B0 = double(IBN23(M-1:M+1,N-1:N+1)).*h3;
        IBN23(M,N) = mean(B0,'all');
    end
end

figure,subplot(131),imshow(IB),title('bone original')
subplot(132),imshow(IBN1),title('bone noise 1 original')
subplot(133),imshow(IBN2),title('bone noise 2 original')

figure,subplot(131),imagesc(IB3),colormap('gray'),title('bone')
subplot(132),imagesc(IBN13),colormap('gray'),title('bone noise 1')
subplot(133),imagesc(IBN23),colormap('gray'),title('bone noise 2')
```


bone original
bone noise 1 original
bone noise 2 original


bone
bone noise 1
bone noise 2
50
100
150
200
250
300
350
400
100
200
300