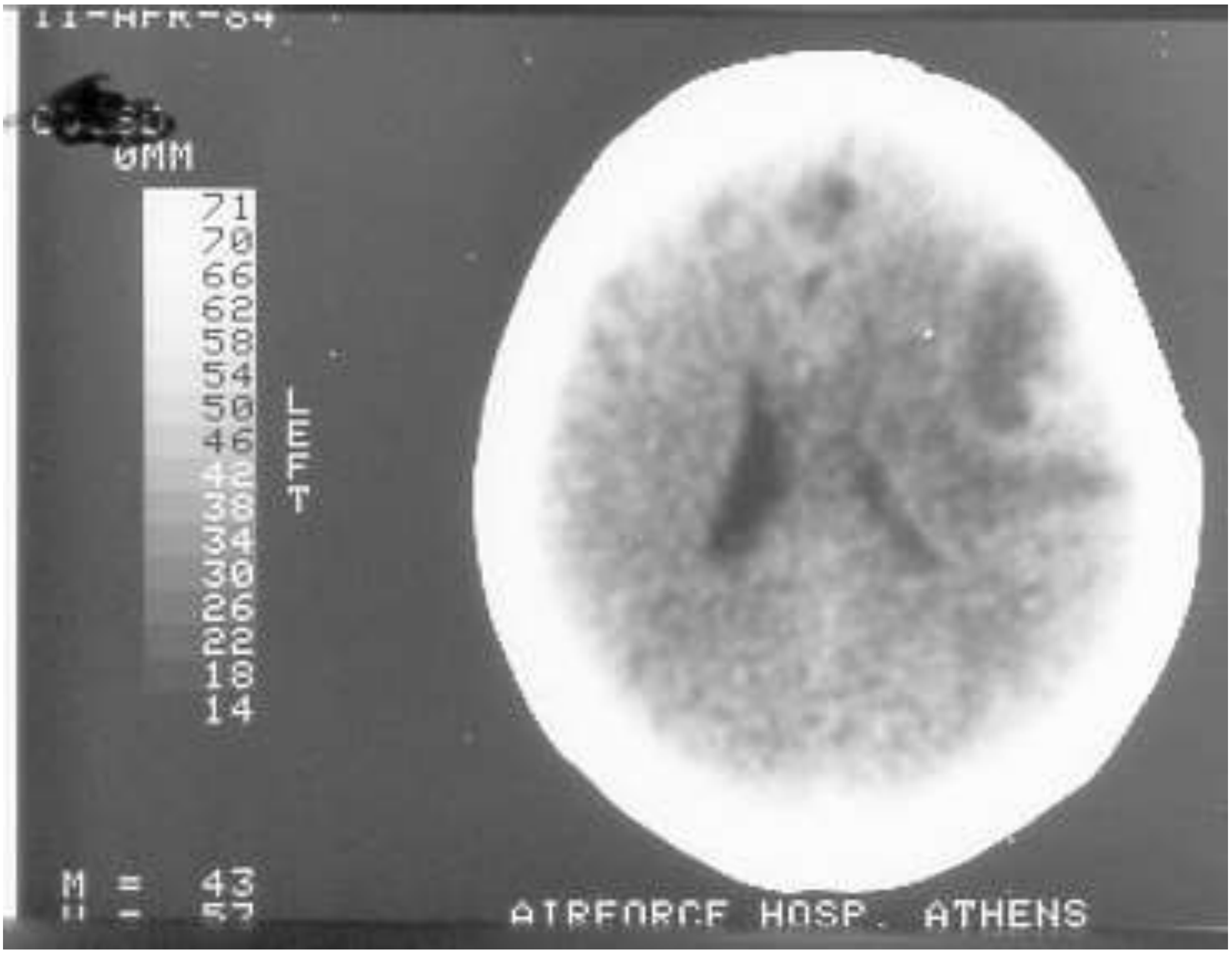
Image matrix manipulation for decreasing image noise and blur.

Image matrix manipulation techniques can be applied in either the spatial or the frequency domain, since the convolution process can be carried out in either domain. For this reason image enhancement will examined in the present lectures in both domains.



Time Domain Image Matrix Manipulation Techniques..

These techniques attempt to reduce one of the sources of image degradation, described by the general image degradation model and given by equation:

$$x^*(n_1,n_2) = x(n_1,n_2) * h(n_1,n_2) + d(n_1,n_2)$$

1/ The convolution term $x(n_1,n_2)*h(n_1,n_2)$ usually suppresses the medium to high frequency image content with result the blurring of the image boundaries and edges and considerable loss of image detail ($h(n_1,n_2)$ is the same as the Point Spread Function or PSF of the system).

2/The noise degradation $d(n_1,n_2)$ is of statistical nature and to a good approximation it is considered as additive and signal independent (although signal dependency may sometimes be severe). Noise resides mostly in the high frequency spectrum of the medical image and when its magnitude is larger than the difference in intensity of two neighboring areas (e.g. metastatic area against normal tissue in liver) then distinguishing those two areas is difficult.

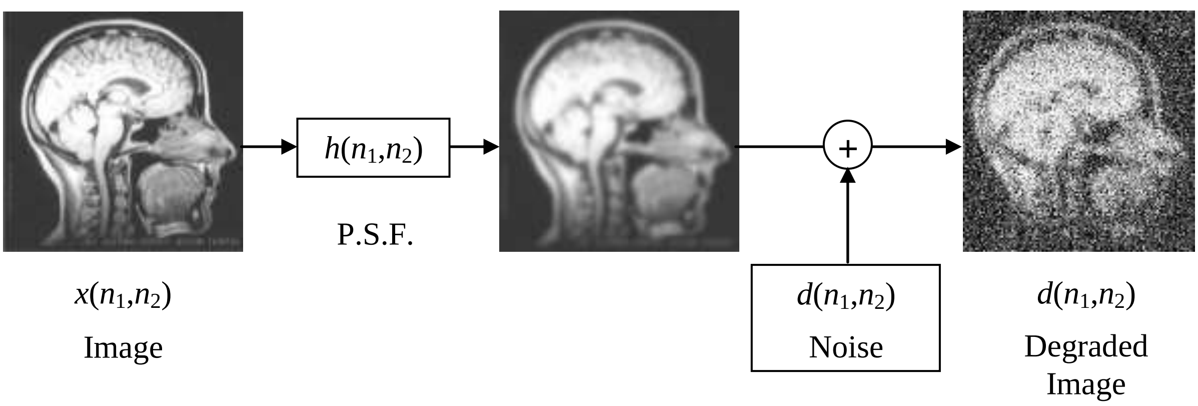
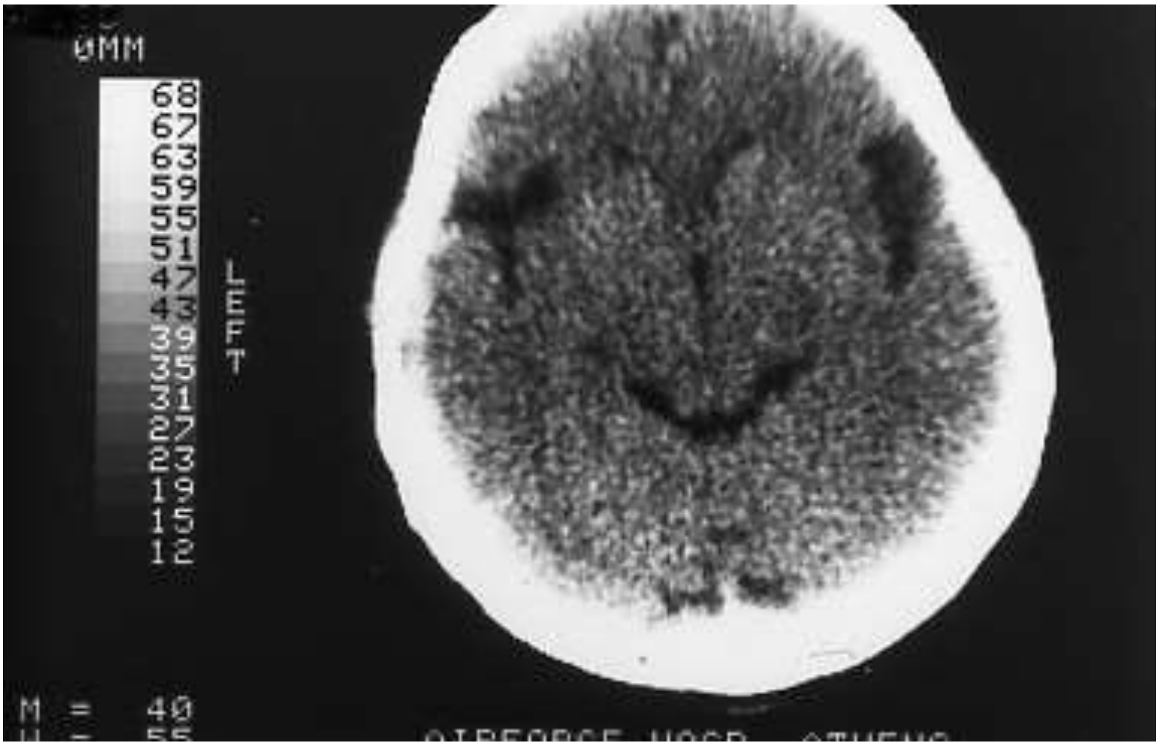


Image Smoothing.

Image smoothing concerns image processing techniques used for suppressing noise, which for various reasons (e.g. sampling, transmission, statistical nature of radiation) resides on the image. Usually, medical image noise is statistical and occupies mainly the image high frequency band. Thus, techniques for image smoothing are in effect high frequencies suppression techniques or low pass filtering techniques.



1-d case:

Let discrete time signal: x(n)={5,9,7,12,6,15,8} and lets apply on the signal the operation :

$$y(n) = \frac{x(n) + x(n-1)}{2} \qquad (1)$$

Result: y(n)={5, 7, 8, 9.5, 9,10.5,11.5}

It's obvious the y(n) is a smoothed version of x(n).

Suppose that this smoothing operation is satisfactory for removing statistical noise from the signal. One way to generalize such an operation is to attempt obtain the same result but through the 1-d convolution:

$$y(n) = \sum_{k=0}^{k=L-1} x(n-k) \cdot h(k) = x(n)h(0) + x(n-1)h(1) \qquad (2)$$

Where, L=2 since operation (1) concerns 2 points only.

Rewriting (1)

$$y(n) = \frac{1}{2}x(n) + \frac{1}{2}x(n-1) \qquad (3)$$

And comparing (2) and (3) we get:

$$h(0) = \frac{1}{2}, h(1) = \frac{1}{2} \qquad \textbf{or} \qquad h(n) = \left\{\frac{1}{2}, \frac{1}{2}\right\}$$

Thus:

$$x(n) \longrightarrow \boxed{h(n)} \longrightarrow y(n)$$

$$h(n) = \left\{\frac{1}{2}, \frac{1}{2}\right\}$$

and h(n) is a Low-pass filter

Similarly $h(n) = \left\{\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right\}$ **corresponding to** $y(n) = \sum_{-1}^{1} \frac{x(n-k)}{3}$

And generalizing $h(n) = \left\{\frac{1}{L}, \frac{1}{L}, \frac{1}{L}, \ldots, \frac{1}{L}\right\}$ **for** $y(n) = \sum_{-\frac{L-1}{2}}^{+\frac{L-1}{2}} \frac{x(n-k)}{L}$, L: odd

Image Smoothing:

2-D Convolution:

$$y(n_1,n_2) = x(n_1,n_2)*h_f(n_1,n_2) = \sum_{l=-\frac{M-1}{2}}^{l=\frac{M-1}{2}} \sum_{k=-\frac{M-1}{2}}^{k=\frac{M-1}{2}} x(n_1-l,n_2-k)h_f(l,k)$$

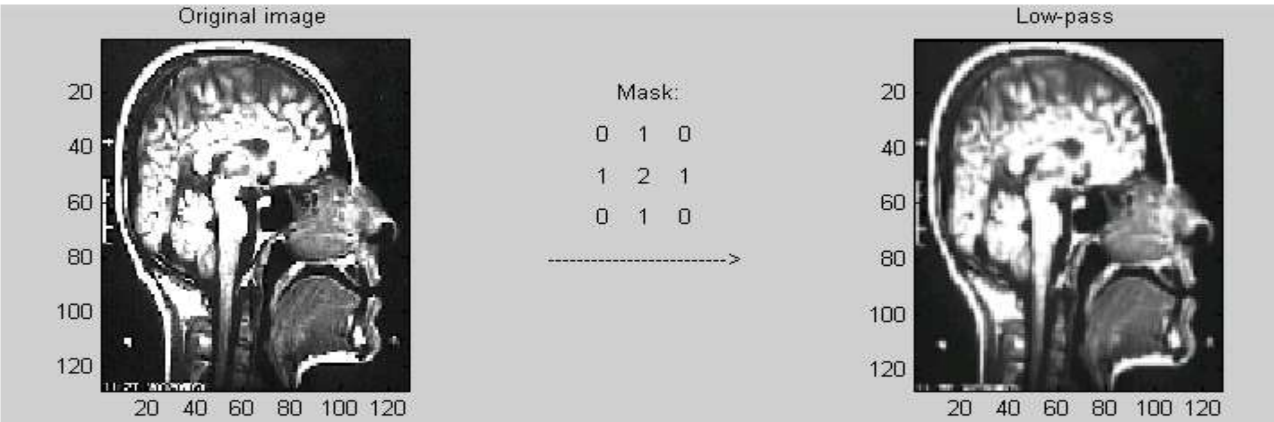
Where $$h_f(l,k) = \frac{1}{\sum h(l,k)} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ . & . & . & . & . \\ . & . & . & . & . \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

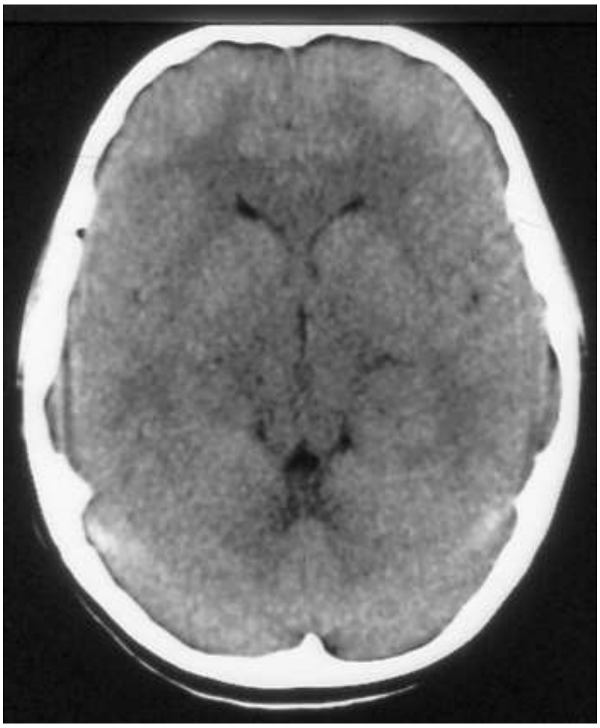
Image Smoothing:

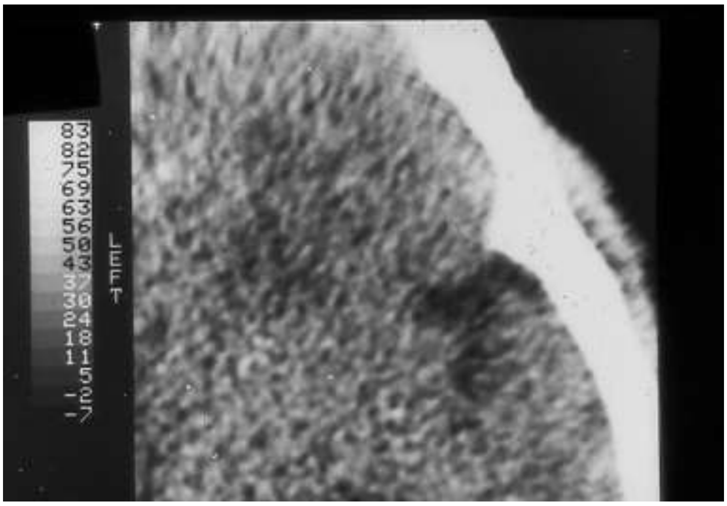
For M=3

$$h_f(l,k) = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = SM1, \quad h_f(l,k) = \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix} = SM2,$$

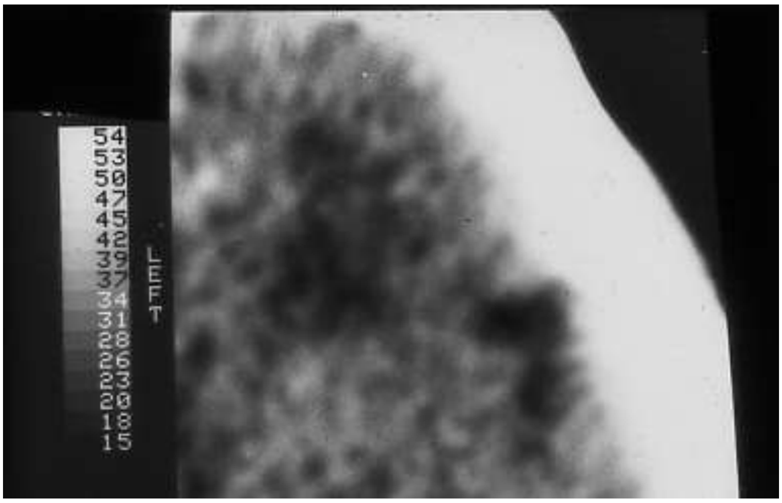
$$h_f(l,k) = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = SM3, \quad h_f(l,k) = \frac{1}{5}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{bmatrix} = SM4$$







83
82
75
69
63
56
50
43
37
30
24
18
11
5
LEFT



54
53
50
47
45
42
39
37
34
31
28
26
23
20
18
15
LEFT

Example:

Let image matrix

25	10	17	12
30	31	12	9
31	12	26	22
8	9	17	12

And filter mask

$$h_f(l,k) = \frac{1}{X}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

1/X???? →X=16

2/ Frame:

25	10	17	12
30	a	b	9
31	c	d	22
8	9	17	12

3/ a=(25x1+10x2+17x1+30x2+31x4+12x2+31x1+12x2+26x1)/16=22

b=…..=17, c=…..=19, d=….=18

if values a, b, c, or d <0 then =0

Thus

25	10	17	12
30	31	12	9
31	12	26	22
8	9	17	12

$\xrightarrow{\text{SM}}$

25	10	17	12
30	22	17	9
31	19	18	22
8	9	17	12

$mean_A$=(31+12+12+26)/4=20.25, $sigma_A$= 9.74

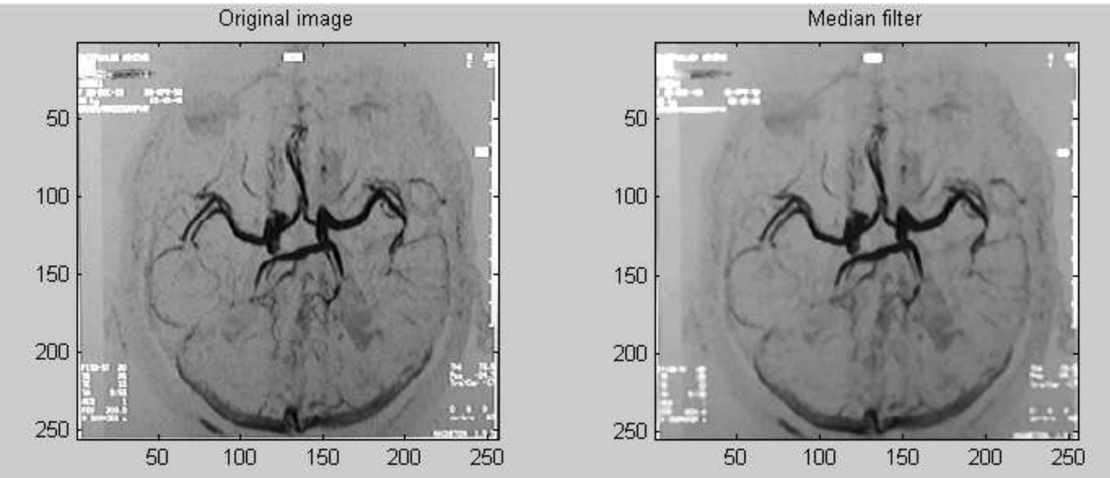
$mean_E$=(22+17+19+18)/4=19, $sigma_E$ = 2.16

noise diff: 100x(sigmaA-sigmaE)/sigmaA=77.83% noise reduction

Median Filter.

The idea behind this smoothing technique is that we replace each image value x by the median of the values surrounding that point, the value of the point itself included. Thus for 3x3 neighborhood the value x is replaced by the fifth (median) largest value. This implies that at each image matrix point a value sorting algorithm has to be applied, which increases the algorithms computational demands.

$$\begin{matrix} 27 & 22 & 20 \\ 23 & 28 & 20 \\ 21 & 24 & 24 \end{matrix} \Rightarrow 20 \quad 20 \quad 21 \quad 22 \quad \underline{\underline{23}} \quad 24 \quad 24 \quad 27 \quad 28$$



```
%Program 4 to read, plot, and convolve *.dat image
function []=Program_4_txt ()
clc;echo off;close all;
A=[30,31,12, 9,
    17,12,25,10,
    12, 8,17, 9,
    31,12,26,22];
sm1  = [1,1,1;1,1,1;1,1,1];%Low pass or smoothing

A=double(A);B=A;C=A;D=A;
disp ('A');disp((A));
image_depth=31;tones=8;

value =1;
switch value
case 1
    B=convolve (A,sm1);
%   B=conv2(A,sm1,'same');
case 2
    B=median_filter(A);
case 3
    B=sharp(A,0.1); %%  if A(i,j)> local_mean  then
B(i,j)=A(i,j)+A(i,j)*threshold; (let threshold=0.1)
end;
C=ampl_fft2(A);
D=ampl_fft2(B);
B=Normalize_Matrix(B,image_depth);C=Normalize_Matrix(C,image_depth);D=Norma
lize_Matrix(D,image_depth);
disp ('B');disp(round(B));disp ('Amplitute A');disp(round(C));disp
('Amplitude B');disp(round(D));

%==========================================================================
====
function [B]= convolve (A,sm)
%--------------------------------------
%convolution
x=size(A,1);y=size(A,2);
B=A;%important for frame
% Initialize
mask=sm;
mask=double(mask);
sum_mask=sum(sum(mask));if sum_mask<=0 sum_mask=1;end;
%filter image
for i=2:x-1,
    for j=2:y-1,
        value=0;icount=0;
        for ii=i-1:i+1,
            icount=icount+1; jcount=0;
            for jj=j-1:j+1,
                jcount=jcount+1;
                value=value+A(ii,jj)*mask(icount,jcount);
            end;
        end;
        if (value<0) value=0;end;
        B(i,j)= value/sum_mask;
    end;
end;
%==========================================================================
=====
function [C]=ampl_fft2(A)
x=size(A,1);y=size(A,2);
```

```
%/*--------   2D - FFT  ---------*/
for i=1:x;
    for j=1:y;
        if ( rem((i+j),2) == 1) A(i,j) = -A(i,j);
        else A(i,j) =A(i,j);
        end;%if
    end;%j
end;%i
C=fft2(A);
C=round(10.0 * log(abs(C)+1));
%===========================================
function [C]=Normalize_Matrix(A,tones)
max_A=max(max(A));
min_A=min(min(A));min_A=0;
C=(A-min_A)*(tones)/(max_A-min_A);%back to 0-(tones)

%============================================================================
LabWork 4:
Form all the Laplacian and High Emphasis masks as well as the median and
the sharp filter functions and include them in the same program with the
smoothing masks. Incorporate a simple switch-case code for choosing a
filtering function. Make sure that there is the correct  indication of the
processing operation underneath each processed image (e.g. smoothing, high-
emphasis etc.).
Thus, construct and submit by 26.3.2013 an overall program in Matlab with
all the filtering functions for image processing in the spatial domain. The
program should work on actual images (e.g. bmp images).
```

```
%Graphics version of Program 4
% Convolution in time domain & spectra
function []=Program_4_gr()
clc;echo off;close all;

A=imread('Images\Pelvis.bmp');%Pelvis.bmp
A=double(A);
x=size(A,1);y=size(A,2);
sprintf('x= %f  y=%f',x,y)
Normalize_Matrix(A,255);
sm1  = [1,1,1;1,1,1;1,1,1];%Low pass or smoothing

tic
B=A;C=A;D=A;

value =1;
switch value
    case 1
        B=convolve (A,sm1);
    case 2
        B=median_filter(A);
    case 3
        B=sharp(A,0.1);
end;

C=ampl_fft2(A);
D=ampl_fft2(B);
image_depth=255;
B=Normalize_Matrix(B,image_depth);C=Normalize_Matrix(C,image_depth);D=Norma
lize_Matrix(D,image_depth);
%==============PLOT IMAGES======================
colormap('gray');
subplot(2,2,1);imagesc(A);xlabel('Original Image');
axis equal;axis([1 size(A,2) 1 size(A,1)]);
subplot(2,2,2);imagesc(B);xlabel('Processed Image');
axis equal;axis([1 size(B,2) 1 size(B,1)]);

subplot(2,2,3);imagesc(C);xlabel('Original Image Spectrum');
axis equal;axis([1 size(C,2) 1 size(C,1)]);

subplot(2,2,4);imagesc(D);xlabel('Processed Image Spectrum');
axis equal;axis([1 size(D,2) 1 size(D,1)]);
toc


%==========================================================================
function [B]= convolve (A,sm)
B=A;
%Put your code here
%===========================================================================
function [C]=ampl_fft2(A)
C=A;
%Put your code here
%=========================================
function [C]=Normalize_Matrix(A,tones)
C=A;
%Put your code here
%===========================================================================
```

Image Sharpening

Image sharpening refers to image enhancement techniques employed to reduce blurring caused by the Point Spread Function (PSF) of the image formation process. The equation describing the image degradation process is:

$$y(n_1,n_2) = x(n_1,n_2) * h(n_1,n_2) + d(n_1,n_2)$$

where $x(n_1,n_2)$ is the original image,
$y(n_1,n_2)$ is the degraded image,
$h(n_1,n_2)$ is the impulse response or the PSF of the image formation system, and $d(n_1,n_2)$ is the image noise.

The result of image blurring is the loss of image detail and boundary clarity, that is the suppression of medium to high frequency image content. It is obvious that image de-blurring or sharpening is a high frequency 'strengthening' operation or high pass filtering. However, high pass filtering should be exercised with caution since image noise resides in the high frequencies of the image spectrum, where the noise magnitude may be comparable to image signal.

Image Sharpening: 1D case

1/ Edge Enhancement: + $\text{if} + \rightarrow \text{LP, then} - \rightarrow \text{HP}$

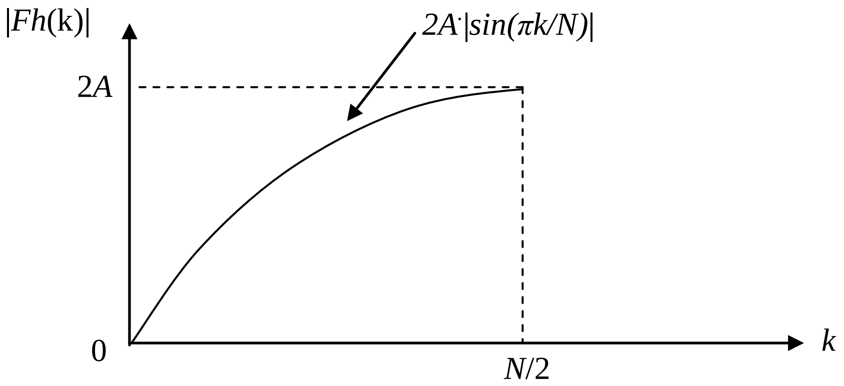
Let operation on x(n): $y(n) = x(n) - x(n-1)$ (1)

1-D Convolution: $y(n) = \sum_{k=0}^{L=1} x(n-k) \cdot h(k) = x(n)h(0) + x(n-1)h(1)$ (2)

From (1) and (2) : $y(n) = x(n) - x(n-1) = x(n) \times (1) + x(n-1) \times (-1)$

thus $h(n) = \{1, -1\}$ (3)

is the filter impulse response (differencing filter) with Spectral response:



Let a 3x3 ROI in the image matrix:

$$\begin{bmatrix} . & O_2 & . \\ O_3 & x & O_1 \\ . & O_4 & . \end{bmatrix}$$

Let Δ_i be the difference of point x from point O_i then (following direction from point O_i to x):

$$\left.\begin{aligned} \Delta_1 &= O_1 - x \\ \Delta_2 &= O_2 - x \\ \Delta_3 &= O_3 - x \\ \Delta_4 &= O_4 - x \end{aligned}\right\} + \Rightarrow y = \sum_{i=1}^{4} \Delta_i = \sum_{i=1}^{4} O_i - 4x$$

Reordering $y= (O_1\text{-}x)\text{-}(x\text{-}O_3)\text{+}(O_4\text{-}x)\text{-}(x\text{-}O_2) = \Delta_h^2 + \Delta_v^2$

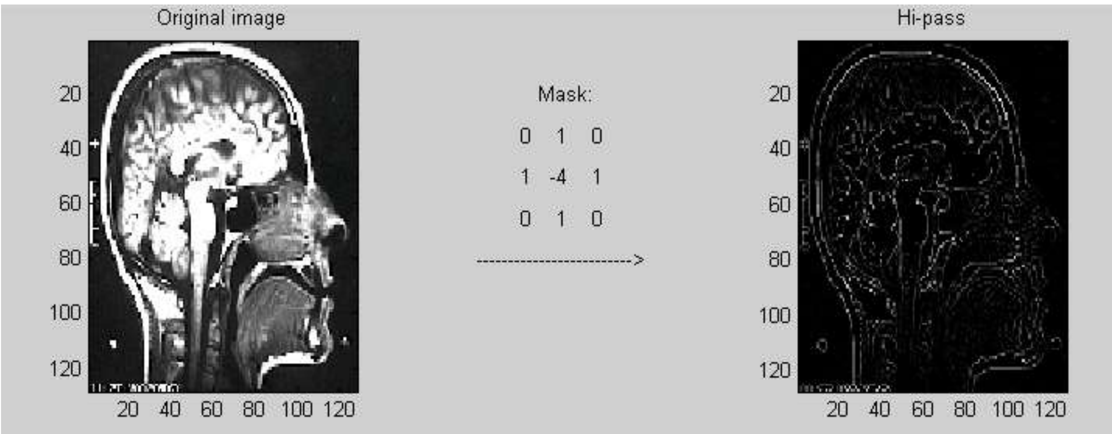
where, $\Delta_h^2 + \Delta_v^2$ is the horizontal and vertical second differences or the Laplacian Operators in the horizontal and vertical direction. This is the reason that the filter described above is also referred to as **the Laplacian filter.**

Thus the Laplacian masks

$$y(n_1,n_2) = x(n_1,n_2)*h_f(n_1,n_2) = \sum_{l=-1}^{l=1}\sum_{k=-1}^{k=1} x(n_1-l,n_2-k)h_f(l,k) \Rightarrow$$

$$h_f(l,k) = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} = \text{LM1}, \qquad h_f(l,k) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \text{LM2},$$

$$h_f(l,k) = \begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \text{LM3}, \qquad h_f(l,k) = \begin{bmatrix} -1 & 2 & -1 \\ 2 & -4 & 2 \\ -1 & 2 & -1 \end{bmatrix} = \text{LM4}$$



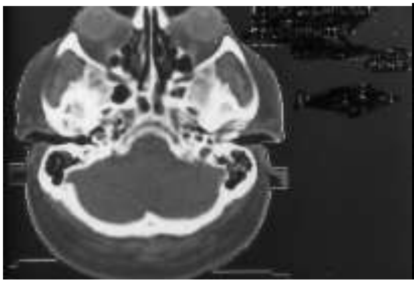
High emphasis filtering.

Edge enhancement may be of value in many cases of image processing but equally important is image sharpening, that is retaining the low frequency image information and at the same time sharpening the edges and the overall image detail. This is achieved by the so-called high emphasis filter.

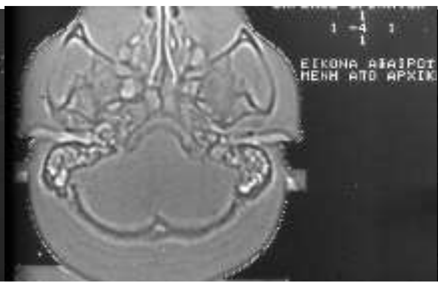
High Emphasis filter: 1D case

x(n)	0	2	4	6	8	10	10	10
D: x(n)-x(n-1)	0	2	2	2	2	**2**	0	0
L: x(n+1)+x(n-1) -2x(n)	0	0	0	0	0	**-2**	0	0
HE: x-L	0	2	4	6	8	**12**	10	10

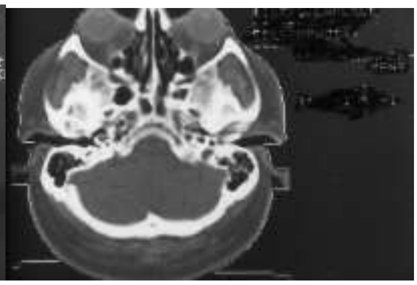
H. Freq. (edge)



A



Laplacian



HighEmphasis=X-L

High Emphasis filter: 2D case

Thus, since $HE = x - L$

or

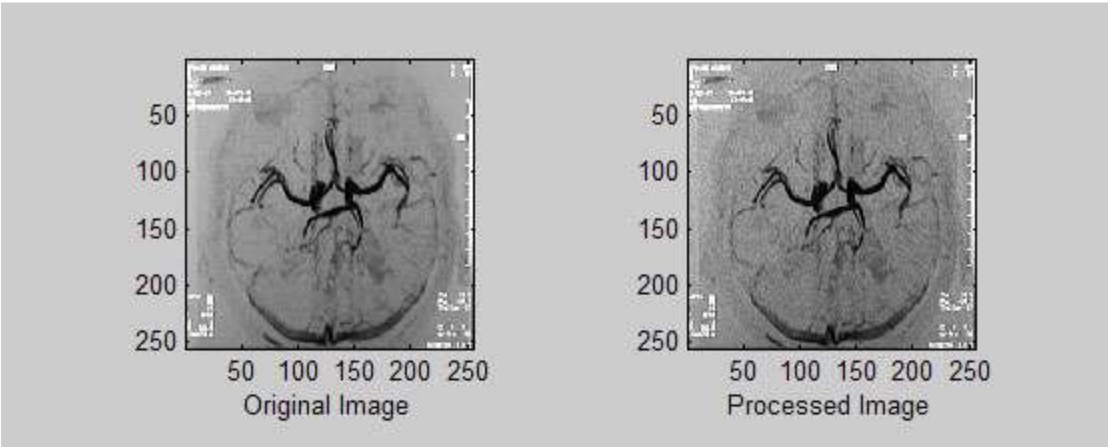
$$L = \left[\sum_{i=1}^{i=4} O_i - 4x\right]$$

Referring to 4 point Figure for the derivation of the Laplacian filter, the 2-d High Emphasis is given by:

$y = x-(O_1+ O_2+ O_3+ O_4 - 4x)= 5x-(O_1+ O_2+ O_3+ O_4)$

which, in correspondence with LM1, results in :

$h(m,l)=\begin{vmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{vmatrix}$ **HEM$_1$ (High Emphasis Mask 1)**



And in correspondence with LM:

$$h_f(l,k) = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} = \mathrm{HEM2},$$

$$h_f(l,k) = \begin{bmatrix} -1 & -2 & -1 \\ -2 & 13 & -2 \\ -1 & -2 & -1 \end{bmatrix} = \mathrm{HEM3}, \qquad h_f(l,k) = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 5 & -2 \\ 1 & -2 & 1 \end{bmatrix} = \mathrm{HEM4}$$