A Brief Introduction on Matlab Functions Related to Image Processing

Overview

Getting Help

2D Matrix / Image

- **Coordinate system**
- **Display**
- **Storing images**

2D Functions

Discrete Fourier Transform

- **1D DFT**
- **fftshift**
- **2D DFT: zero-filling and shifting the image**

Getting Help in Matlab

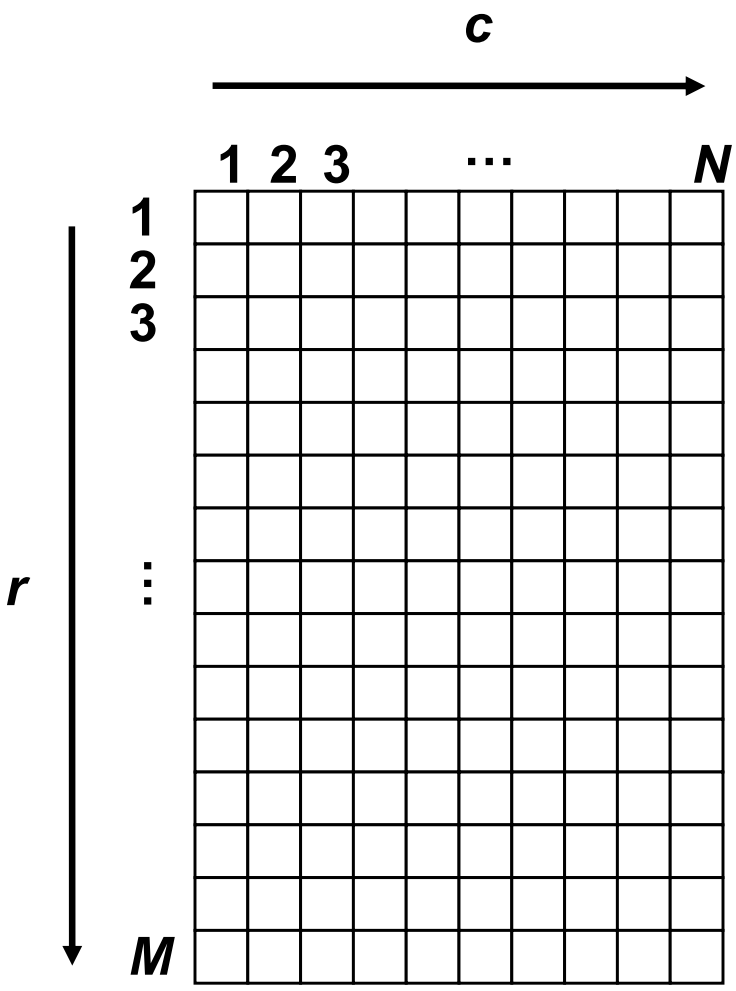
Online Help

- **Help Browser (? In toolbar or >>helpbrowser)**
- **Help Functions (>>help *functionname*)**
- **Matlab website: www.mathworks.com**
- **Demos**

Documentation

- **Online as pdf files**

2D Matrix



2D matrix has *M* rows and *N* columns

Example

- **>>m=zeros(400,200);**
- **>>m(1:50,1:100) = 1;**

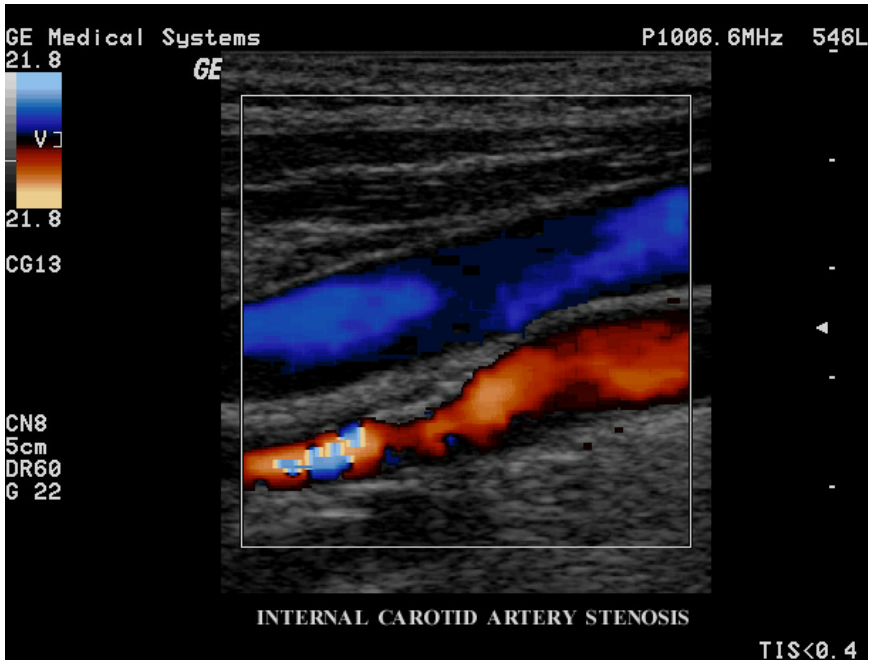
Note:

- **Index 1 to M, not 0 to M-1**
- **Origin of coordinate system is in upper left corner**
- **Row index increases from top to bottom**
- **Coordinate system is rotated in respect to 'standard' x-y coordinate system**

Image Types

Medical images are mostly represented as grayscale images

- **Human visual system resolves highest spatial resolution for grayscale contrast**
- **Color is sometimes used to superimpose information on anatomical data**

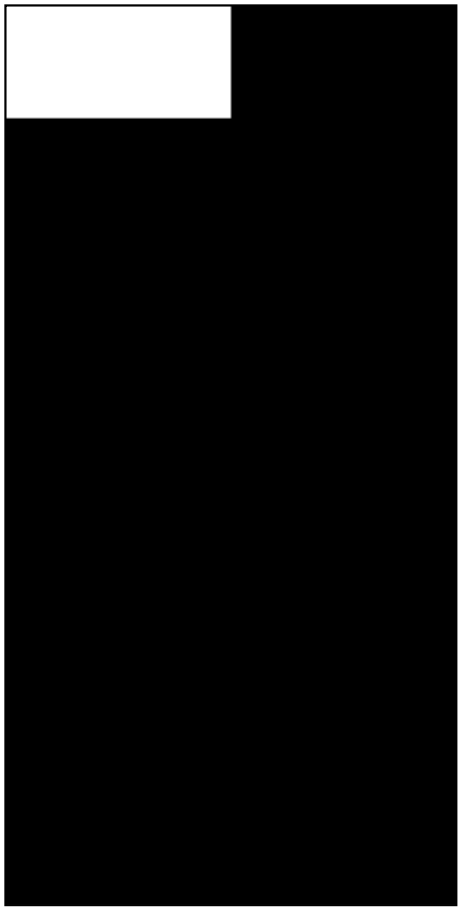


Save Image Matrix

MATLAB binary format

- **>>save example**

  -> writes out example.mat



Standard image format (bmp,tiff,jpeg,png,hdf,pcx,xwd)

- **>>imwrite(m,'example.tif','tif')**

  -> writes out example.tif
- **Warning: imwrite expects data in range [0…1]**
- **If data range outside this [0…1], use mat2gray**
- **>>m_2 = mat2gray(m);**
- **>>imwrite(m_2,'example.tif','tif')**

Save Figure (incl. labels etc)

Save in Matlab Figure Format:

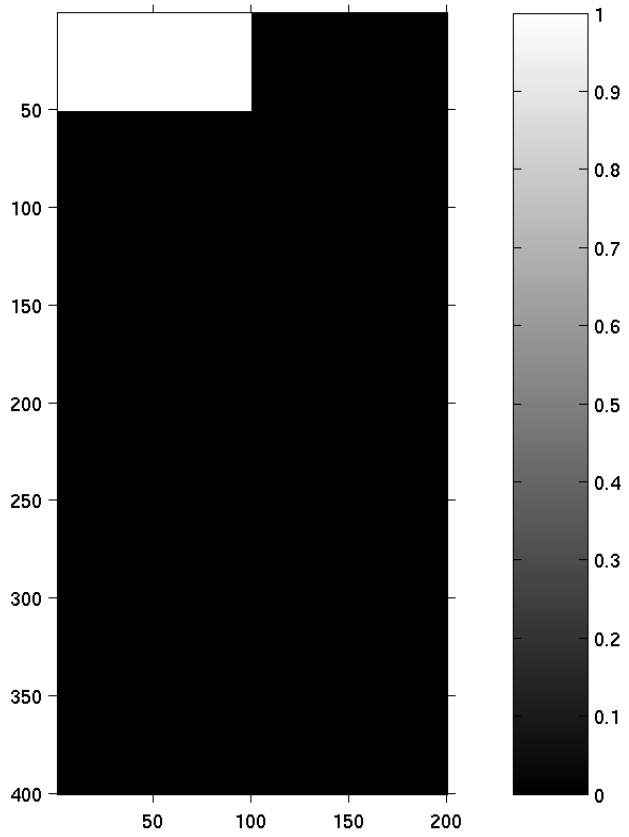
- **>File>Save As> in Figure Window**

  -> writes out example.fig

Standard image formats

- **>File>Export in Figure Window**
  - **E.g. jpg, tif, png, eps, …**
- **Alternatively, use print, e.g.**

  >>print –deps2 example.eps



Loading 2D Data and Images

Load matrix from MATLAB binary format

>>load example (loads example.mat)

Load matrix from standard image format

>>m_in = imread('example.tif','tif');

To check on an image: >>imfinfo('example.tif','tif')

Load figure from Matlab Figure Format (*.fig):

- **>File>Open> 'example.fig' in Figure Window**

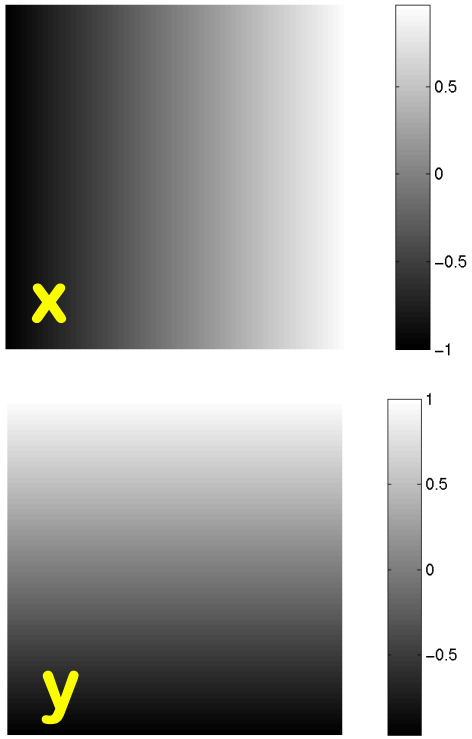
->Check loaded matrices

- **>>whos**

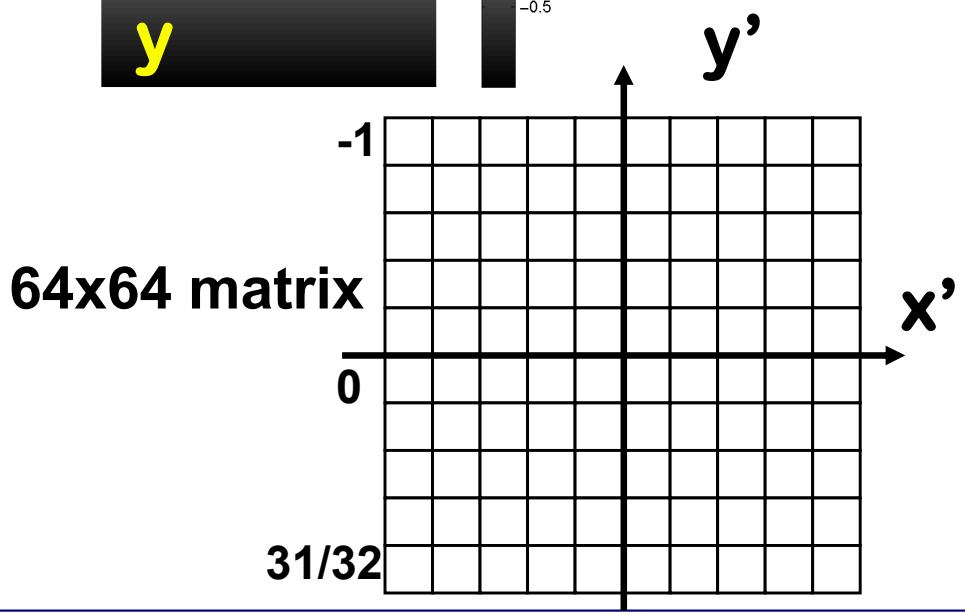
Some Common Image Formats

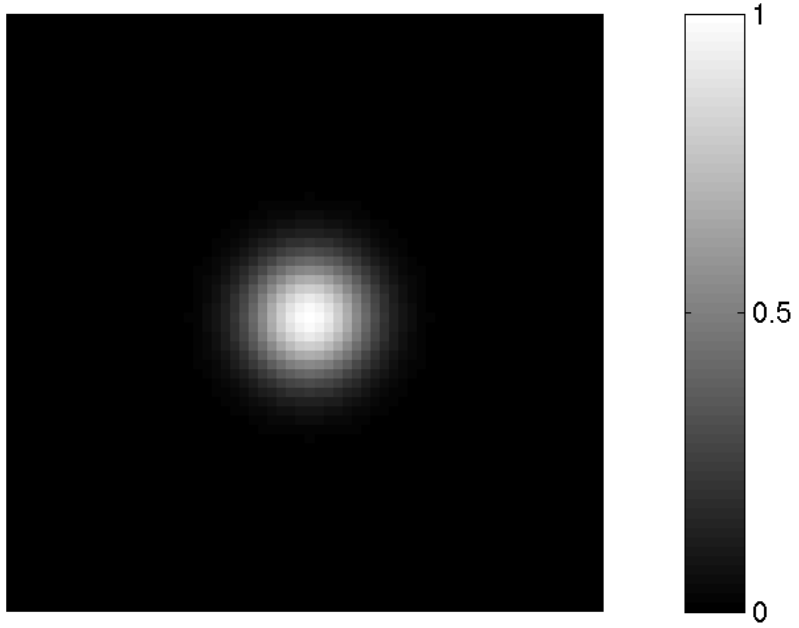
- **TIF (TIFF)**
  - **Very common, compatible, all purpose image format**
  - **Allows for lossless compression**
- **JPEG**
  - **Allows for lossy compression (small file size)**
  - **Very common for internet**
- **PNG (portable network graphics)**
  - **Good for saving MATLAB images for importing into Microsoft documents such as Word**
- **Dicom**
  - **THE medical imaging format**
  - **Lots of header information (patient name & ID, imaging parameters, exam date, …) can be stored**
  - **Allows for lossy and lossless compression**
  - **Matlab function 'dicomread', 'dicomwrite'**
- **EPS (Encapsulated Postscript)**
  - **Graphics format rather than an image format**
  - **Great for best quality print results**
  - **Large file size**

2D Functions



- **Create a matrix that evaluates 2D Gaussian: exp(-p/2(x²+y²)/s²)**
  - **>>ind = [-32:1:31] / 32;**
  - **>>[x,y] = meshgrid(ind,-1*ind);**
  - **>>z = exp(-pi/2*(x.^2+y.^2)/(.25.^2));**
  - **>>imshow(z)**
  - **>>colorbar**





Discrete Fourier Transform in 1-D Revisited

The *discrete Fourier transform* (DFT) is defined as

$$F_n = \sum_{k=0}^{N-1} f_k e^{-\frac{j2\pi nk}{N}}, \mathrm{n} = 0,1,2,...\mathrm{N}-1$$

$\mathrm{n} = 0$ corresponding to the DC component (spatial frenquency is zero)

$\mathrm{n} = 1,..., \mathrm{N}/2 - 1$ are corresponding to the positive frenquencies $0 < \mathrm{u} < \mathrm{u_c}$

$\mathrm{n} = \mathrm{N}/2, ..., \mathrm{N} - 1$ are corresponding to the negative frenquencies $-\mathrm{u_c} < \mathrm{u} < 0$

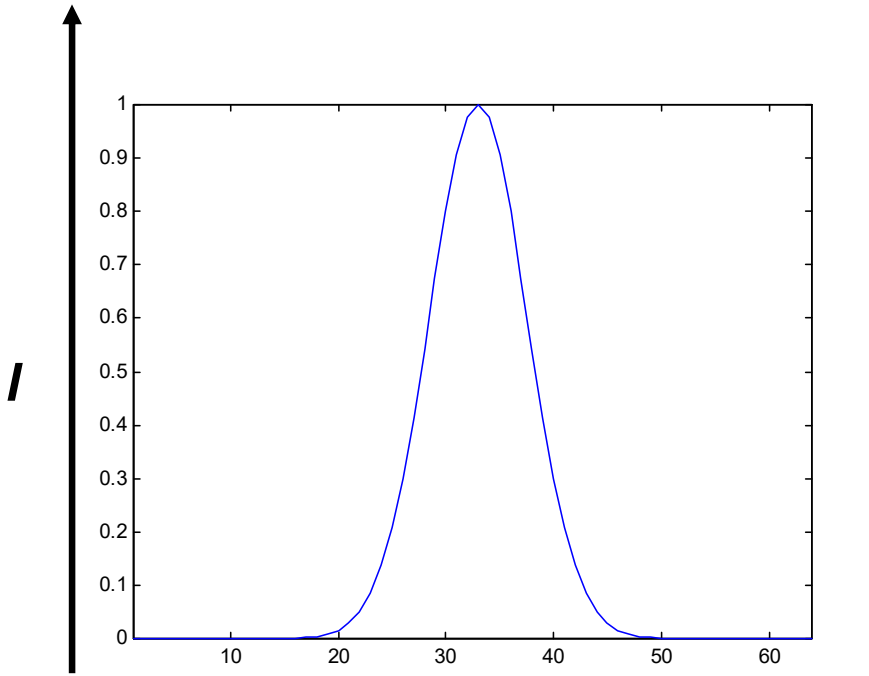
And the *inverse DFT* is defined as

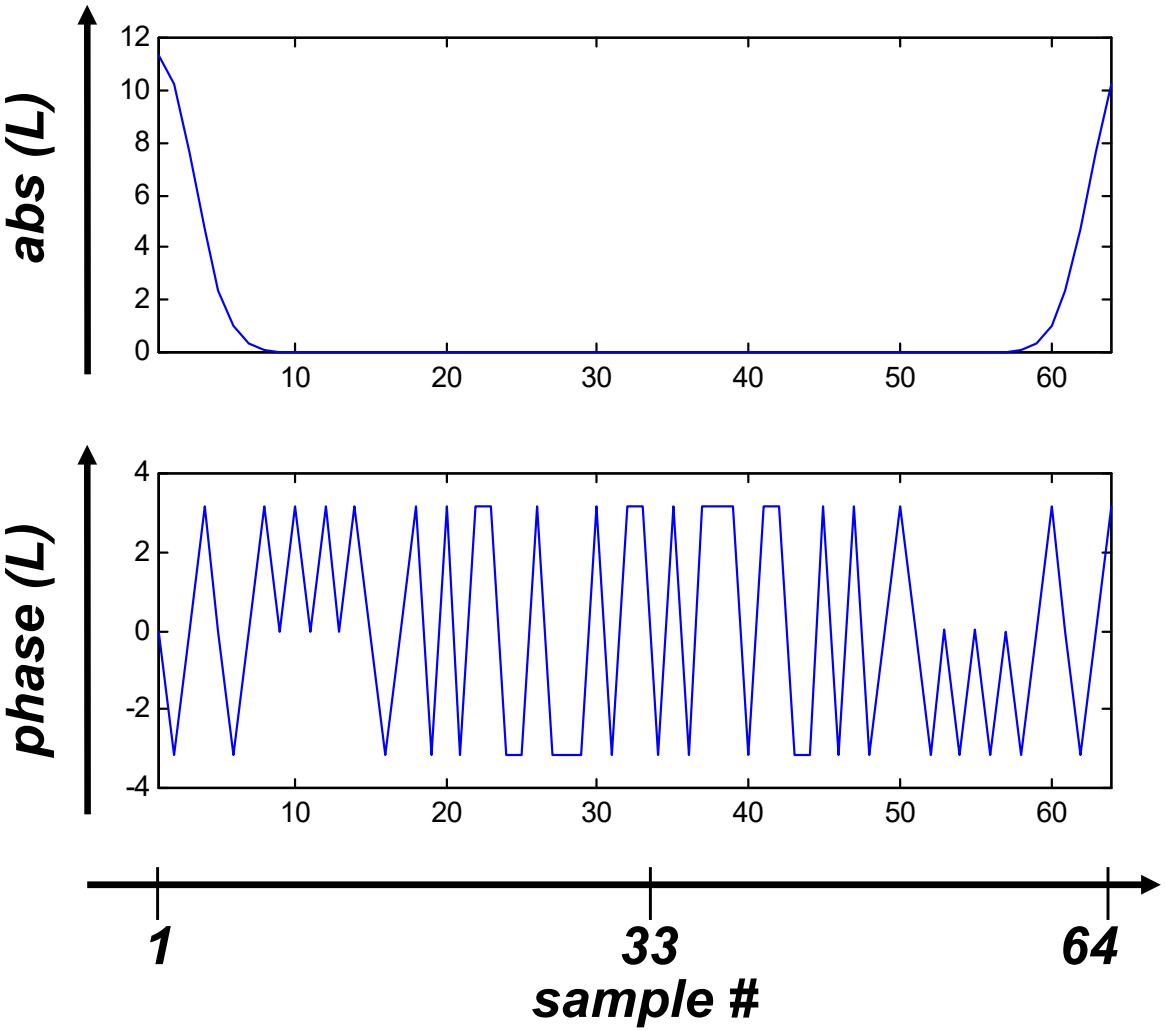
$$f_k = \frac{1}{N} \sum_{n=0}^{N-1} F_n e^{-\frac{j2\pi nk}{N}}, \mathrm{k} = 0,1,2,...\mathrm{N}-1$$

FT Conventions - 1D

1D Fourier Transform

- **>>l = z(33,:);**
- **>>L = fft(l);**



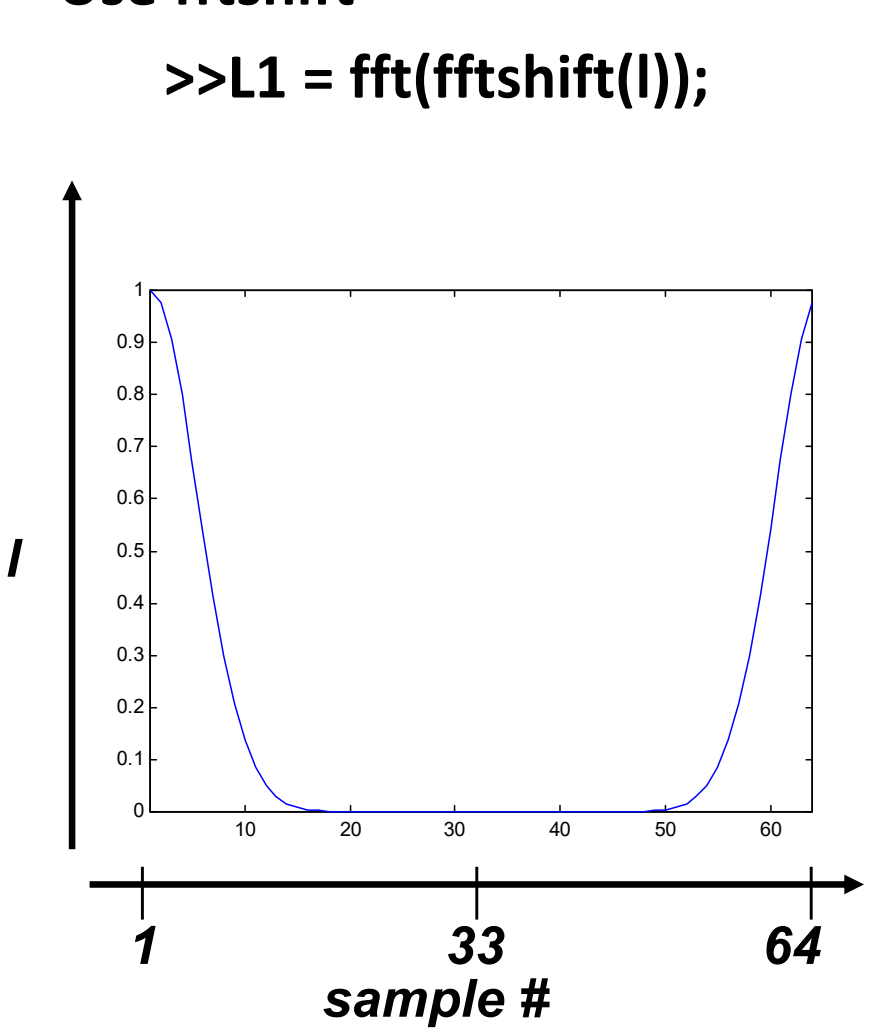


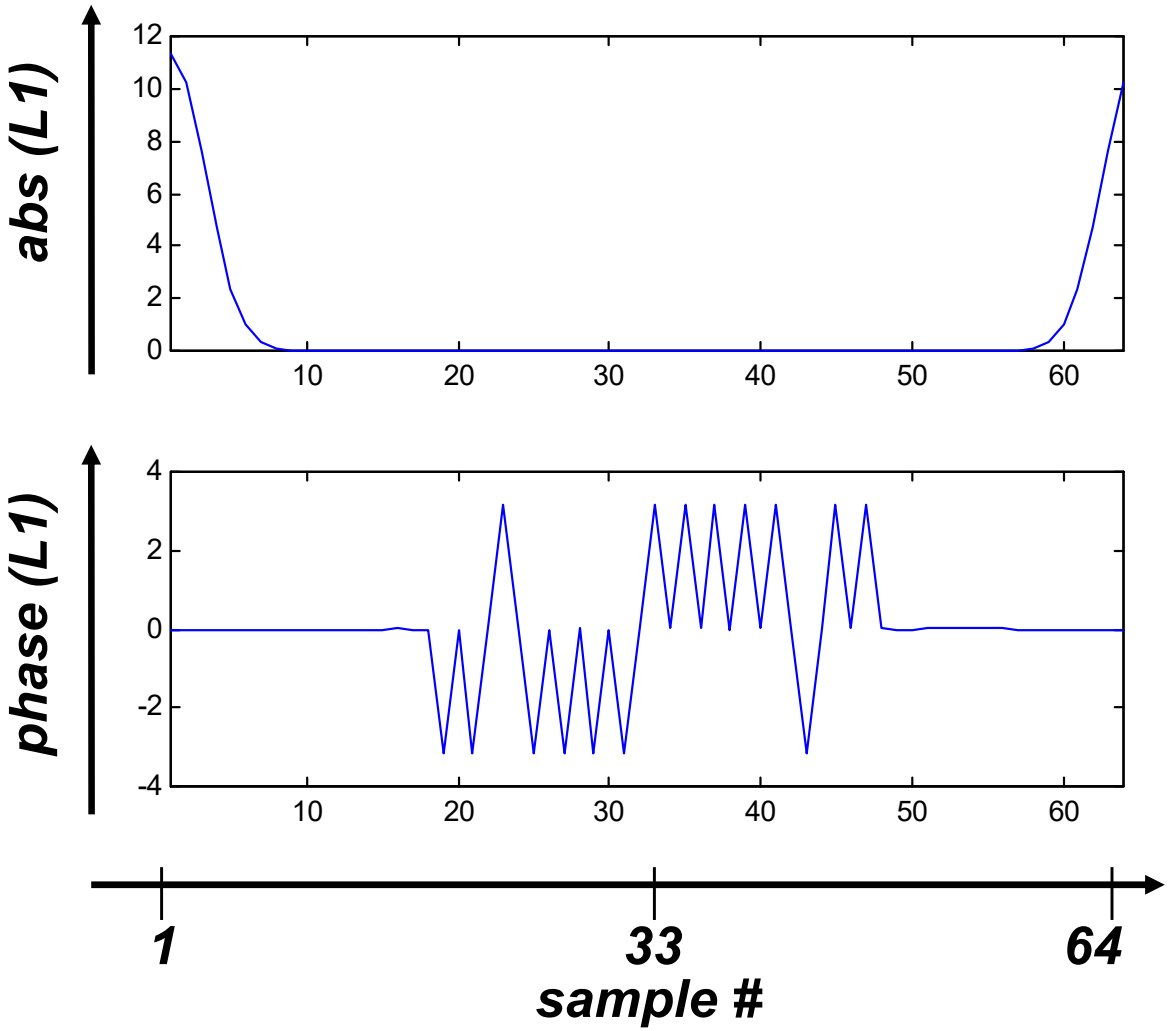
0 ***x* [Matlab]** x_{max} ***0*** $f_N/2$ f_N

1D FT - fftshift

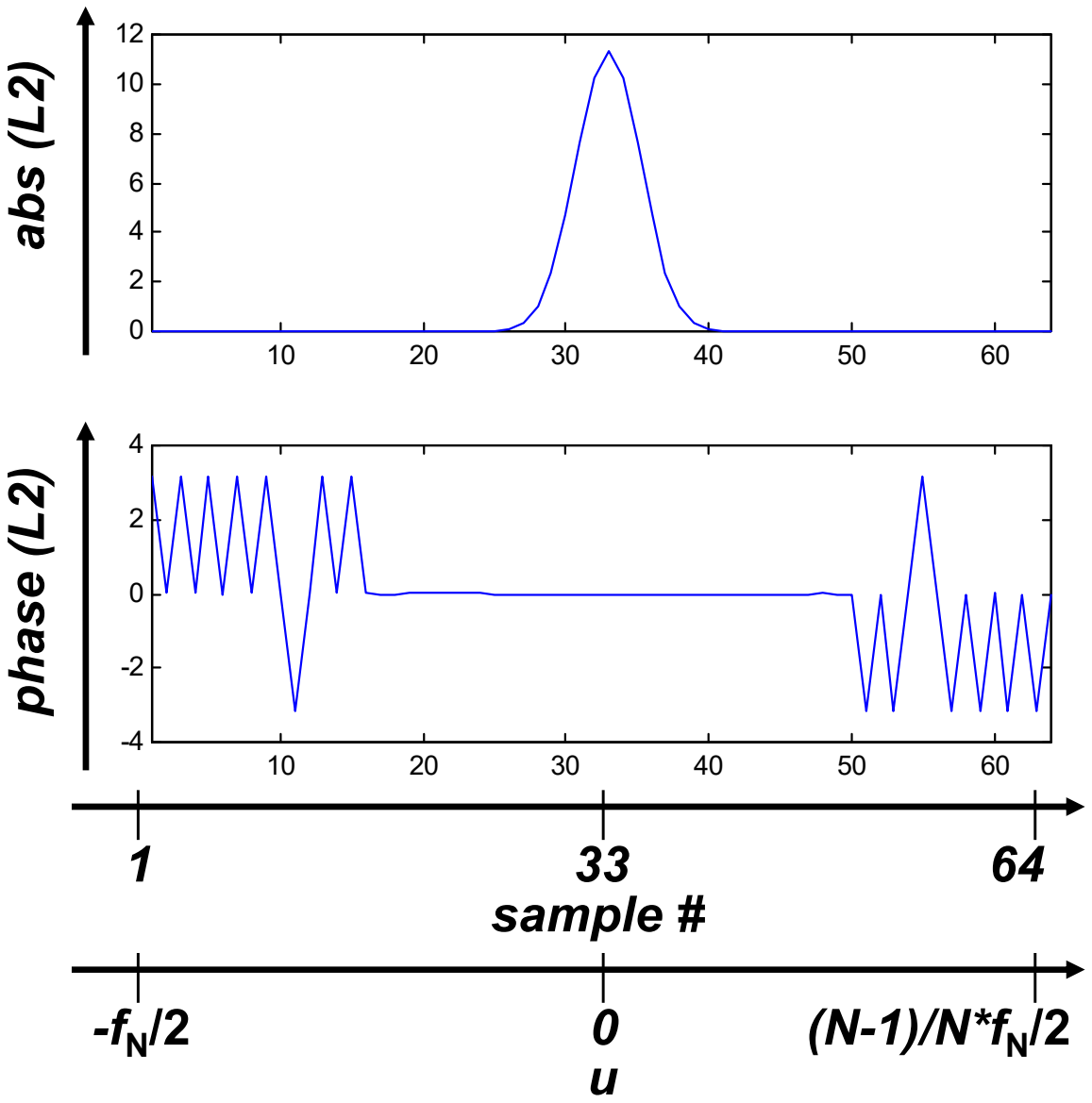
Use fftshift

>>L1 = fft(fftshift(I));





1D FT - 2xfftshift

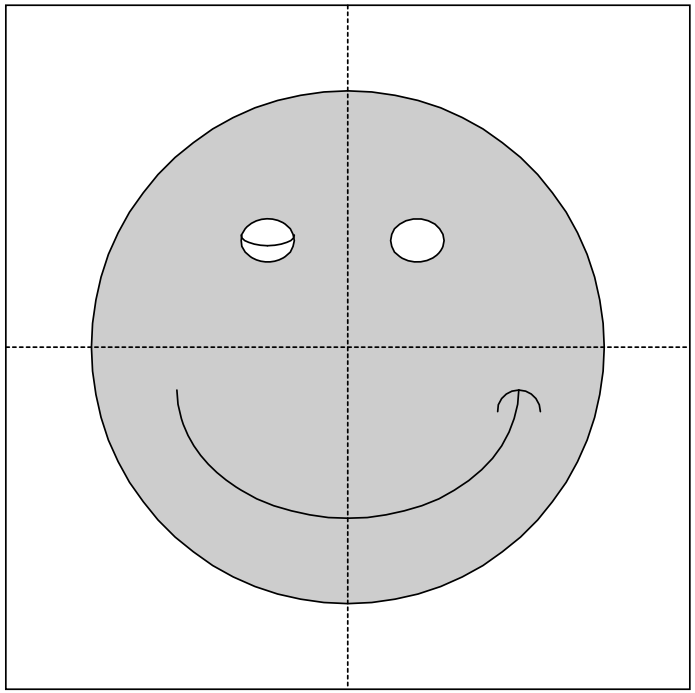


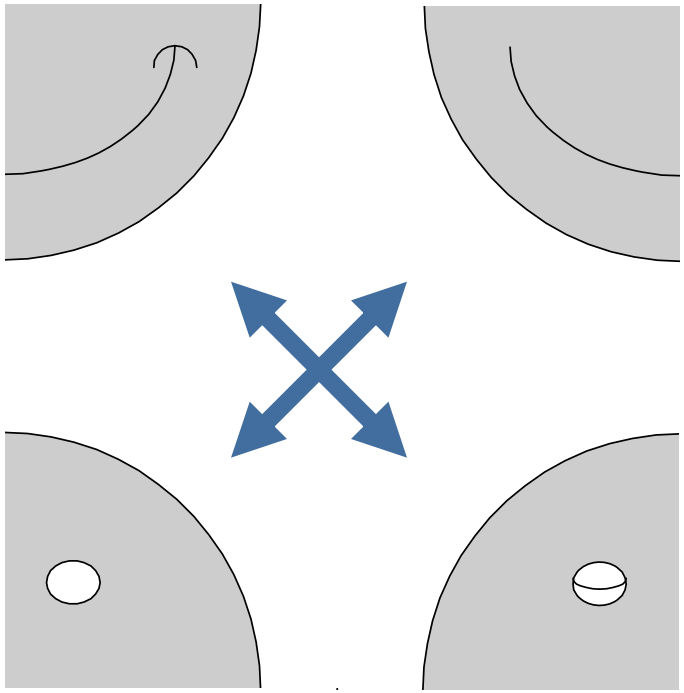
Center Fourier Domain: >>L2 = fftshift(fft(fftshift(I)));

fftshift in 2D

Use fftshift for 2D functions

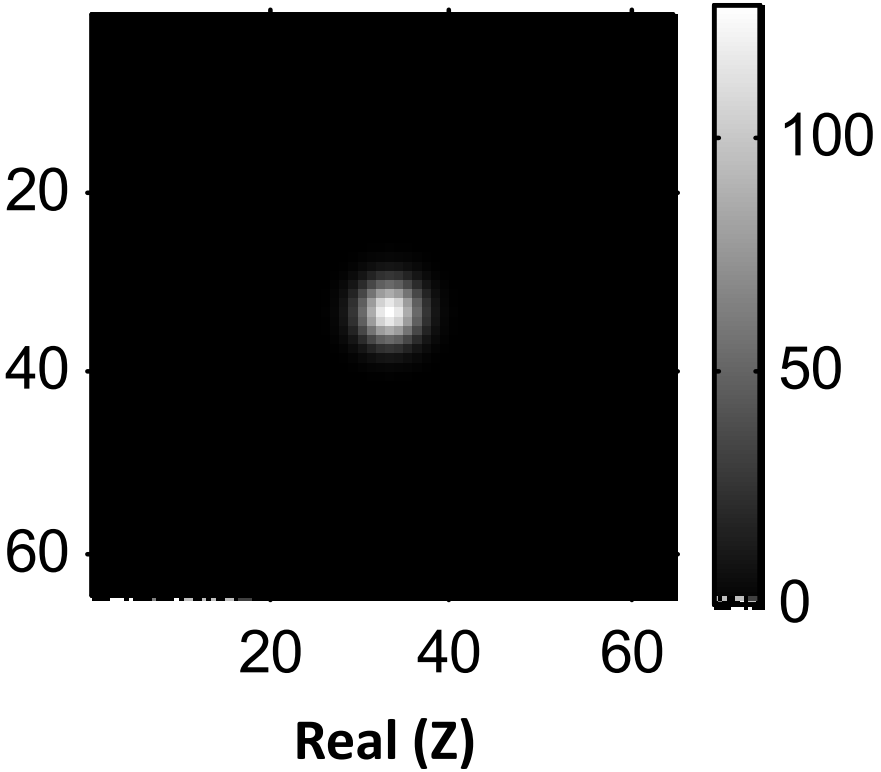
- **>>smiley2 = fftshift(smiley);**





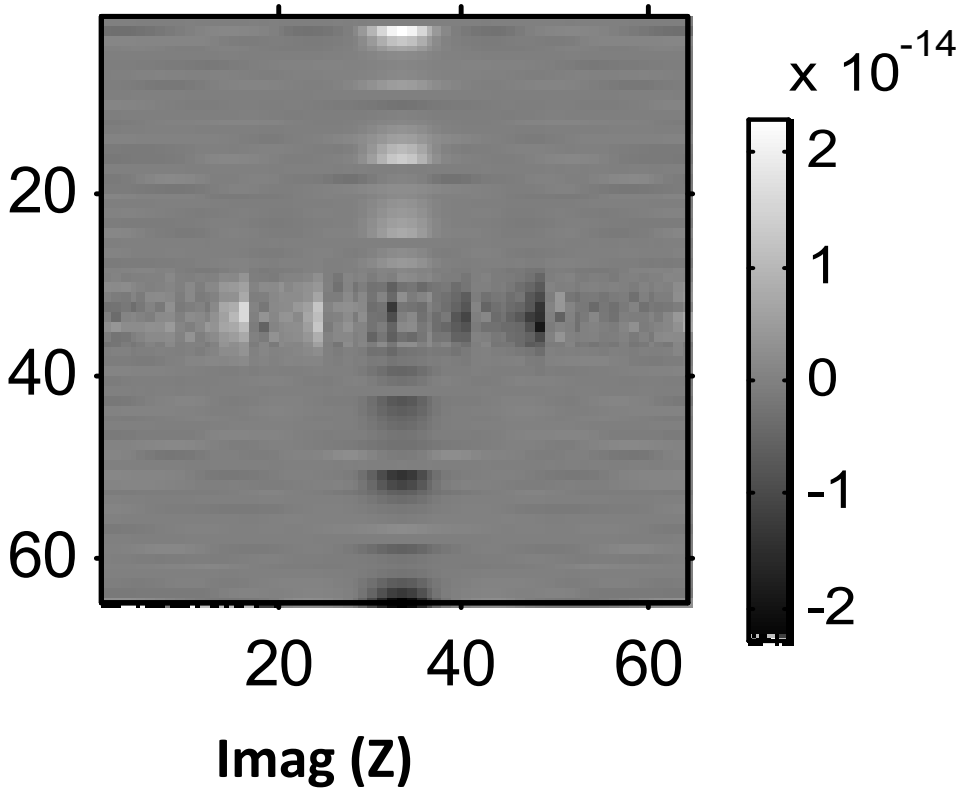
2D Discrete Fourier Transform

- **>>Z=fftshift(fft2(fftshift(z)));**
- **>>whos**
- **>>figure**
- **>>imshow(real(Z))**
- **>>imshow(real(Z),[])**
- **>>colorbar**



Real (Z)

- **>>figure**
- **>>imshow(imag(Z),[])**
- **>>colorbar**

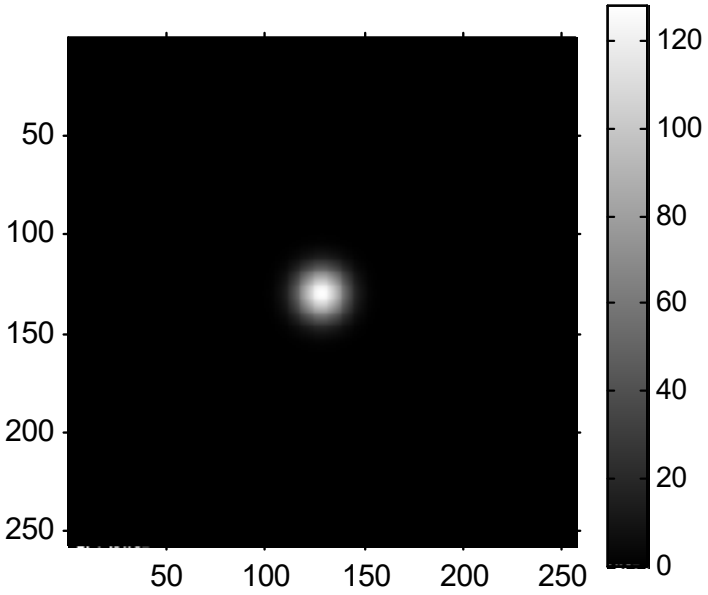


Imag (Z)

2D DFT - Zero Filling

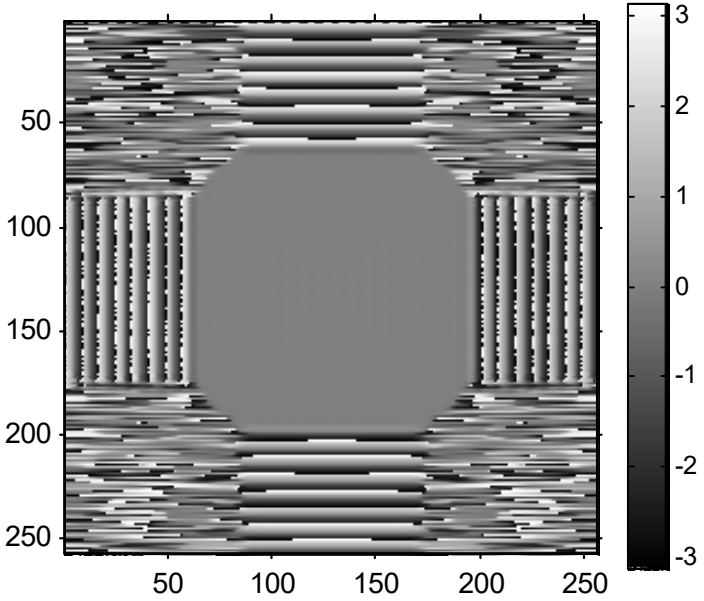
Interpolating Fourier space by zerofilling in image space

- **>>z_zf = zeros(256);**
- **>>z_zf(97:160,97:160) = z;**
- **>>Z_ZF = fftshift(fft2(fftshift(z_zf)));**
- **>>figure**
- **>>imshow(abs(Z_ZF),[])**
- **>>colorbar**



Magnitude (Z_ZF)

- **>>figure**
- **>>imshow(angle(Z_ZF),[])**
- **>>colorbar**

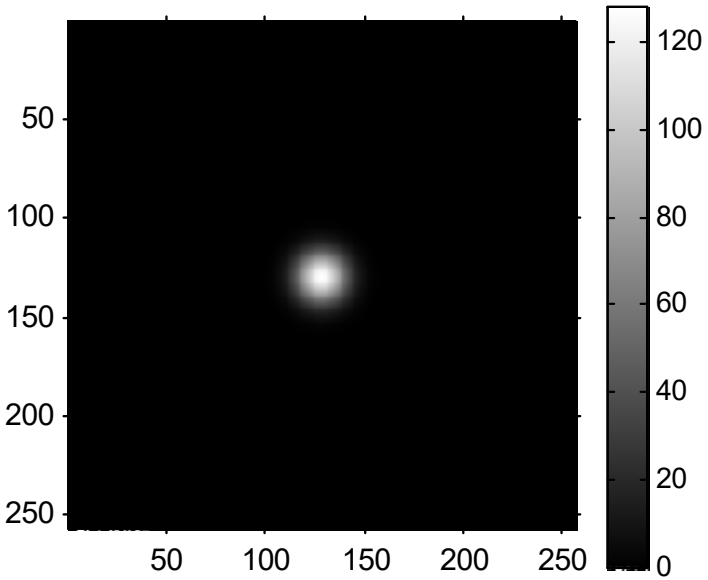


Phase (Z_ZF)

2D DFT - Voxel Shifts

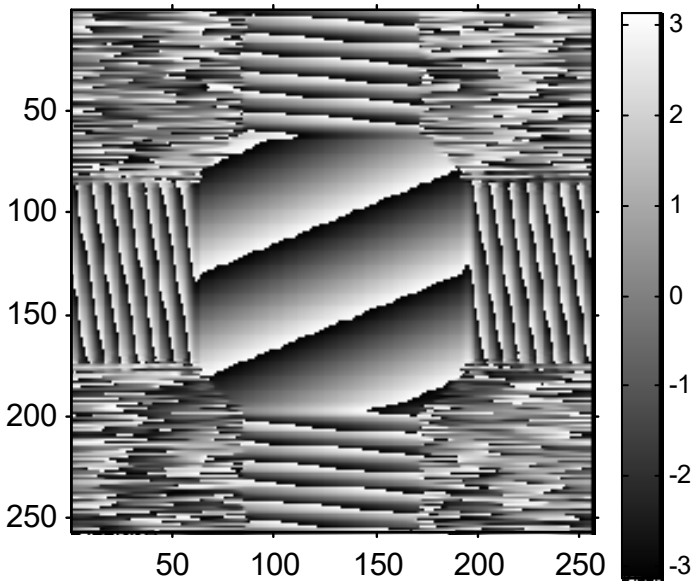
Shift image: 5 voxels up and 2 voxels left

- **>>z_s = zeros(256);**
- **>>z_s(92:155,95:158) = z;**
- **>>Z_S = fftshift(fft2(fftshift(z_s)));**
- **>>figure**
- **>>imshow(abs(Z_S),[])**
- **>>colorbar**



Magnitude (Z_S)

- **>>figure**
- **>>imshow(angle(Z_S),[])**
- **>>colorbar**



Phase (Z_S)

Review of functions …

Input / Output

- **>>imread**
- **>>imwrite**
- **>>imfinfo**

Image Display

- **>>imshow**

Others

- **>>axis**
- **>>colorbar**
- **>>colormap**
- **>>meshgrid**
- **>>fft, fft2**
- **>>fftshift**

More useful functions …

Image Display

- **>>title**
- **>>xlabel, ylabel**
- **>>subimage**
- **>>zoom**
- **>>mesh**

Others

- **>>imresize**
- **>>imrotate**
- **>>imhist**
- **>>conv2**
- **>>radon**
- **>>roipoly**

Demos in Image Processing Toolbox

- **>>help imdemos**

Matlab Examples

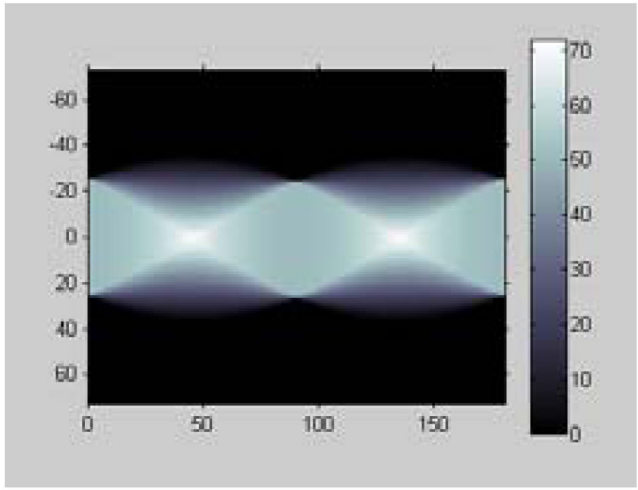
P = phantom('Modified Shepp-Logan',200); imshow(P)

Shepp-Logan phantom is a standard head CT simulation.

Ref: Matlab Help Guide

Matlab Examples

R = radon(I,theta)
If you omit theta, it defaults to 0:179.

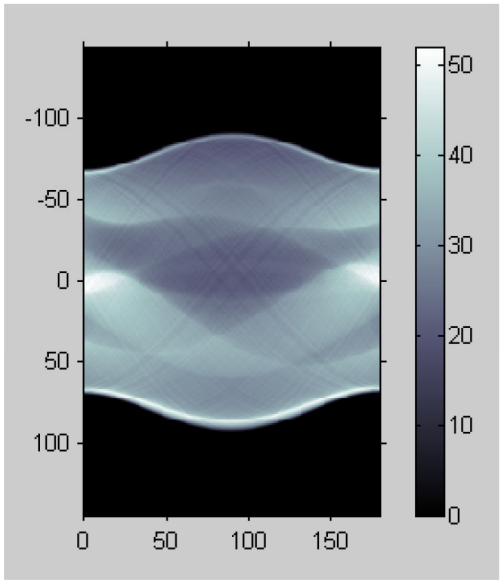


Imaging a square

```
iptsetpref('ImshowAxesVisible','on')
I = zeros(100,100);
I(25:75,25:75) = 1;
theta = 0:180;
[RI,xp] = radon(I,theta);
imshow(theta,xp,RI,[],'notruesize'), colormap(bone), colorbar
```

Imaging the Modified Shepp-Logan phantom

```
P = phantom('Modified Shepp-Logan',200);  imshow(P)
[RP,xp] = radon(P,theta);
imshow(theta,xp,RP,[],'notruesize'), colormap(bone), colorbar
```



Ref: Matlab Help Guide

Matlab Examples

I = iradon(P,theta)
I = iradon(P,theta,interp,filter,d,n)

interp specifies the type of interpolation to use in the backprojection. The available options are listed in order of increasing accuracy and computational complexity:
'nearest' - nearest neighbor interpolation
'linear' - linear interpolation (default)
'spline' - spline interpolation

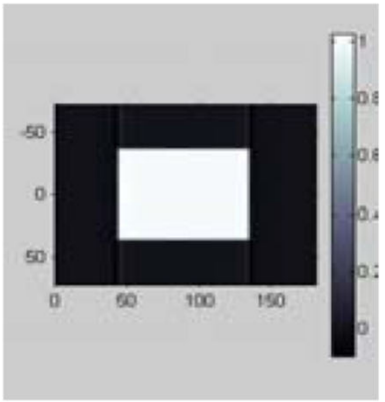
filter specifies the filter to use for frequency domain filtering. filter is a string that specifies any of the following standard filters:
'Ram-Lak' - The cropped Ram-Lak or ramp filter (default). The frequency response of this filter is | f |. Because this filter is sensitive to noise in the projections, one of the filters listed below may be preferable. These filters multiply the Ram-Lak filter by a window that de-emphasizes high frequencies.
'Shepp-Logan' - The Shepp-Logan filter multiplies the Ram-Lak filter by a sinc function.
'Cosine' - The cosine filter multiplies the Ram-Lak filter by a cosine function.
'Hamming' - The Hamming filter multiplies the Ram-Lak filter by a Hamming window.
'Hann' - The Hann filter multiplies the Ram-Lak filter by a Hann window.

d is a scalar in the range (0,1] that modifies the filter by rescaling its frequency axis. The default is 1. If d is less than 1, the filter is compressed to fit into the frequency range [0,d], in normalized frequencies; all frequencies above d are set to 0.

n is a scalar that specifies the number of rows and columns in the reconstructed image. If n is not specified, the size is determined from the length of the projections.
n = 2*floor(size(P,1)/(2*sqrt(2)))

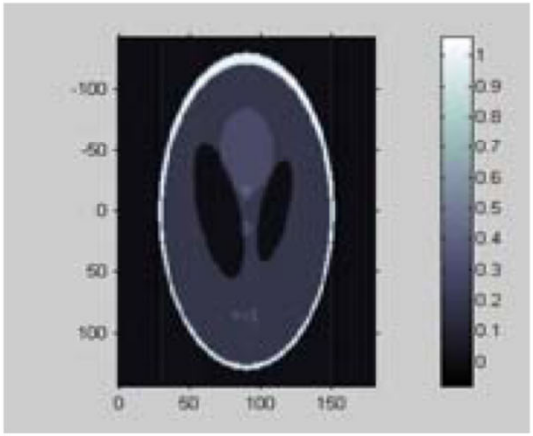
Matlab Examples

```
I = iradon(P,theta)
I = iradon(P,theta,interp,filter,d,n)
```

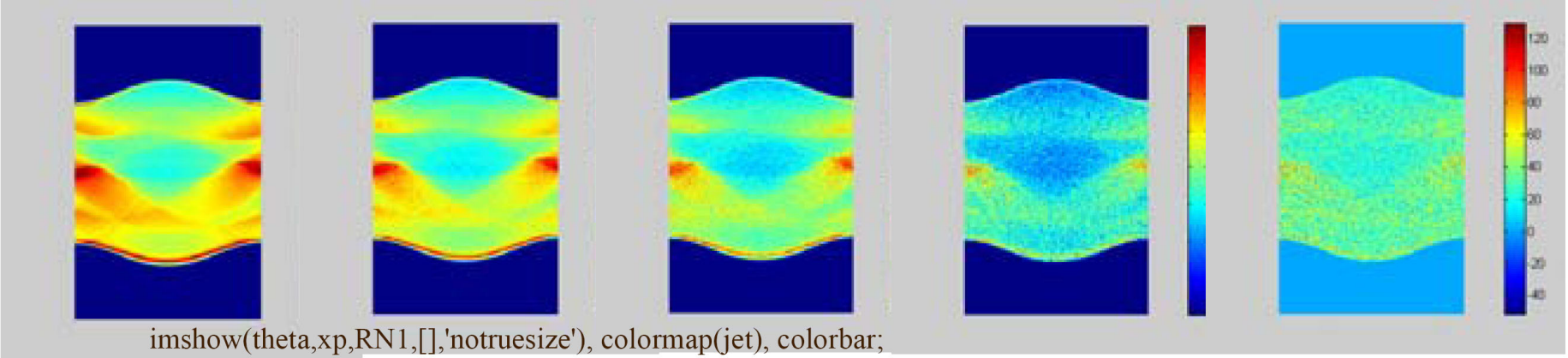



```
KI = iradon(PI,theta)
imshow(theta,xp,KI,[],'notruesize'), colormap(bone), colorbar
```

```
KP = iradon(P,theta)
imshow(theta,xp,KP,[],'notruesize'), colormap(bone), colorbar
```

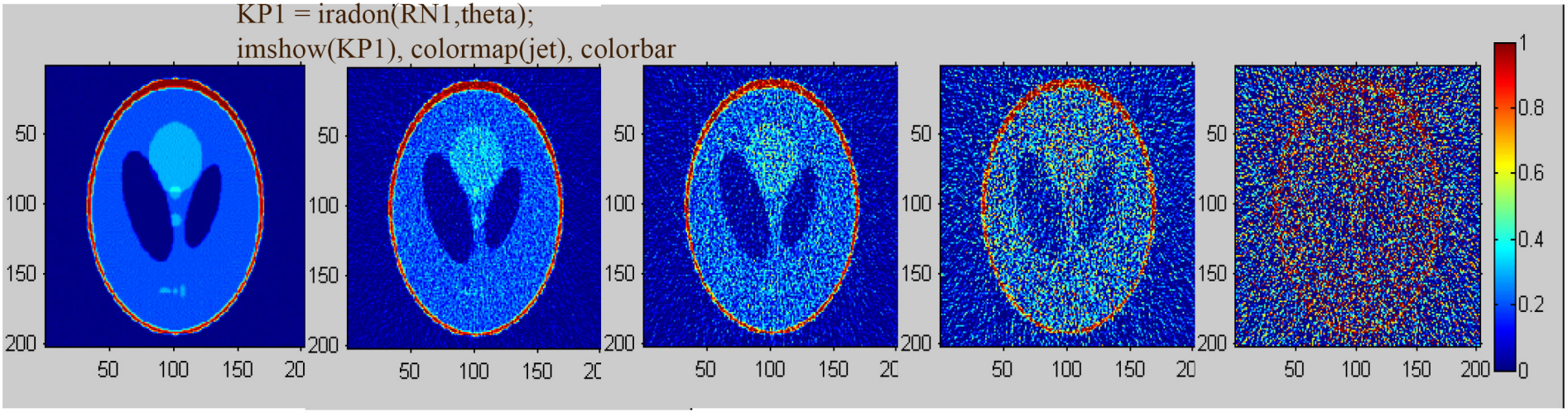


Matlab Examples



imshow(theta,xp,RN1,[],'notruesize'), colormap(jet), colorbar;

KP1 = iradon(RN1,theta);
imshow(KP1), colormap(jet), colorbar



sd=0 sd=0.05 sd=0.10 sd=0.20 sd=0.50