دانشکده روانشناسی و علوم تربیتی

پژوهشگاه دانش‌های بنیادی IPM

MATLAB for Brain and Cognitive Psychology (Generating Stimuli)

Presented by:

Ehsan Rezayat, Ph.D.

Faculty of Psychology and Education, University of Tehran.

Institute for Research in Fundamental Sciences (IPM), School of Cognitive Sciences,

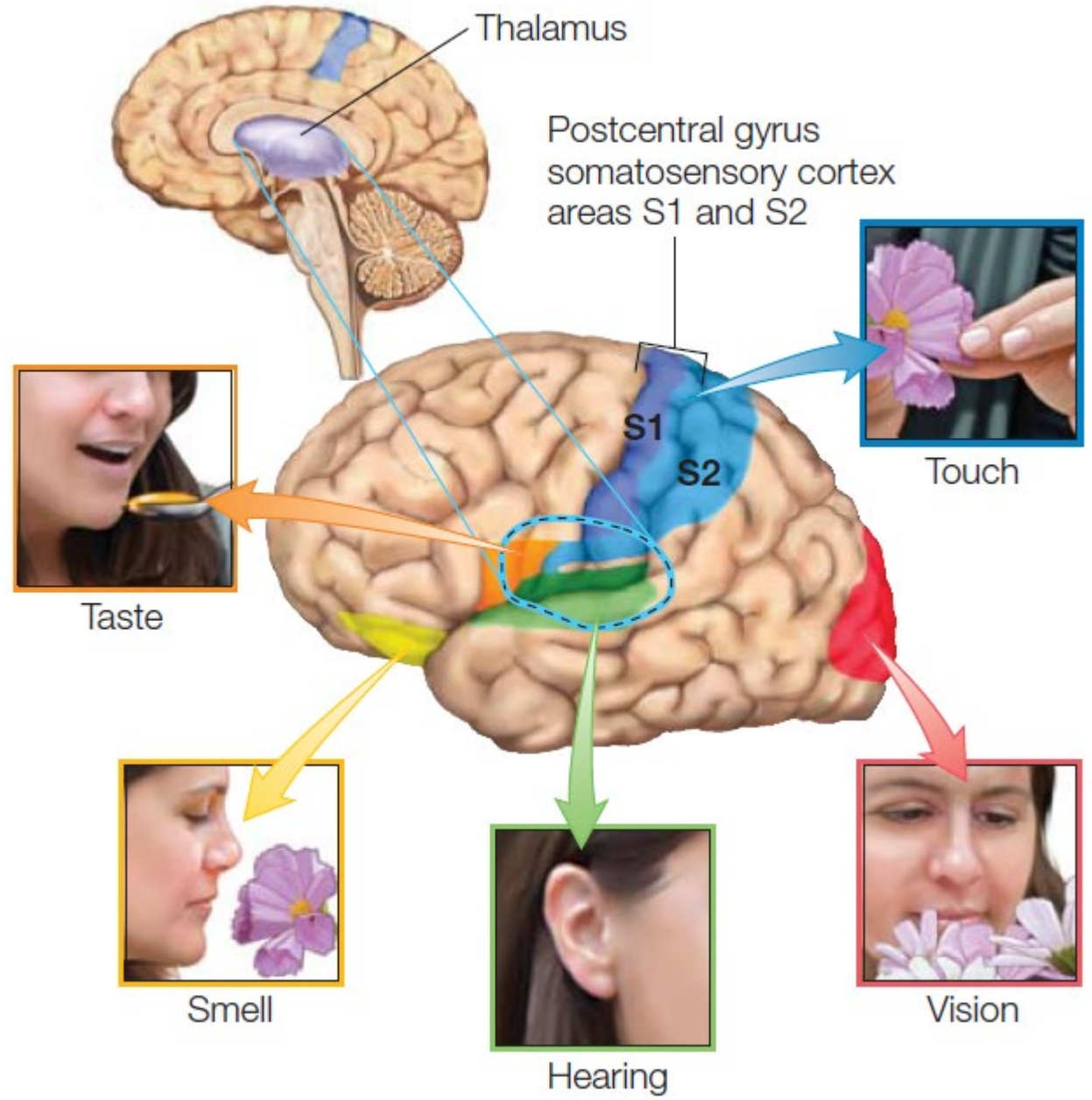
emails: rezayat@ut.ac.ir, rezayat@ipm.ir, erezayat.er@gmail.com

Today: Steps in the Psychophysics Lab

- Generating Stimuli
- Stimulus presentation
- Visual Display
- Response collection

Different senses

- Auditory
- Visual
- Somatosensory
- Olfactory
- Gustatory



ادارک

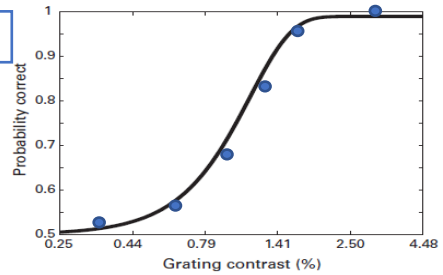
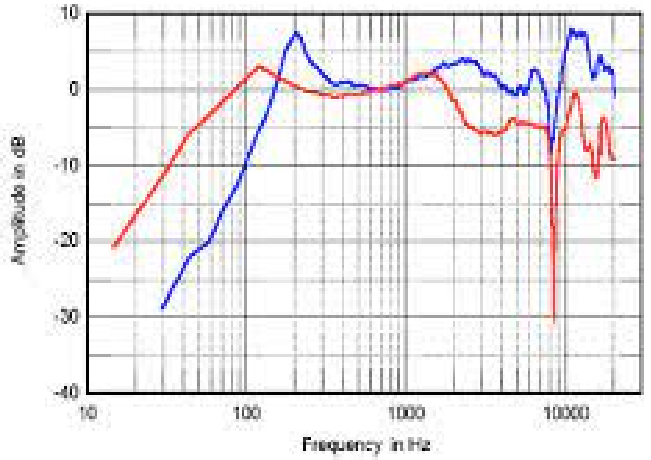


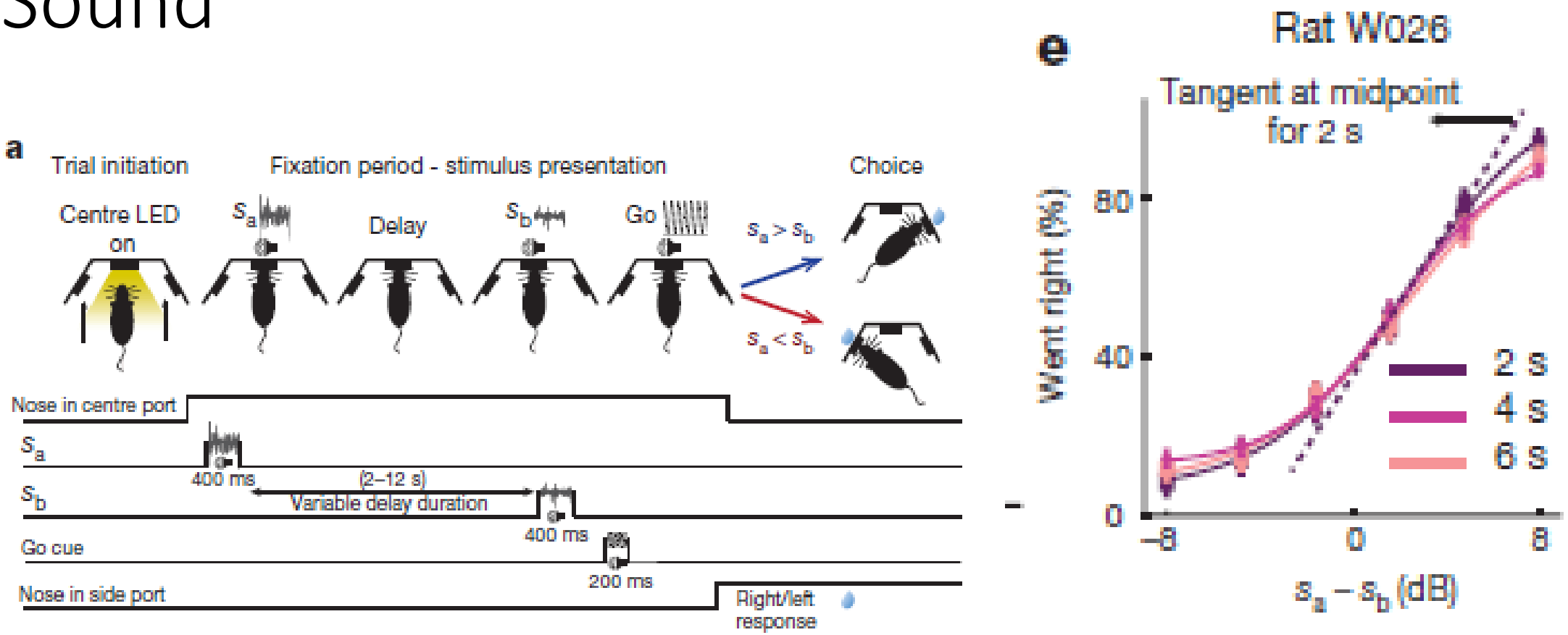
Figure 2.2
Psychometric function for detecting sine-wave gratings of different contrasts.

فیزیک

- Auditory
- Visual
- Olfactory
- Somatosensory
- Gustatory

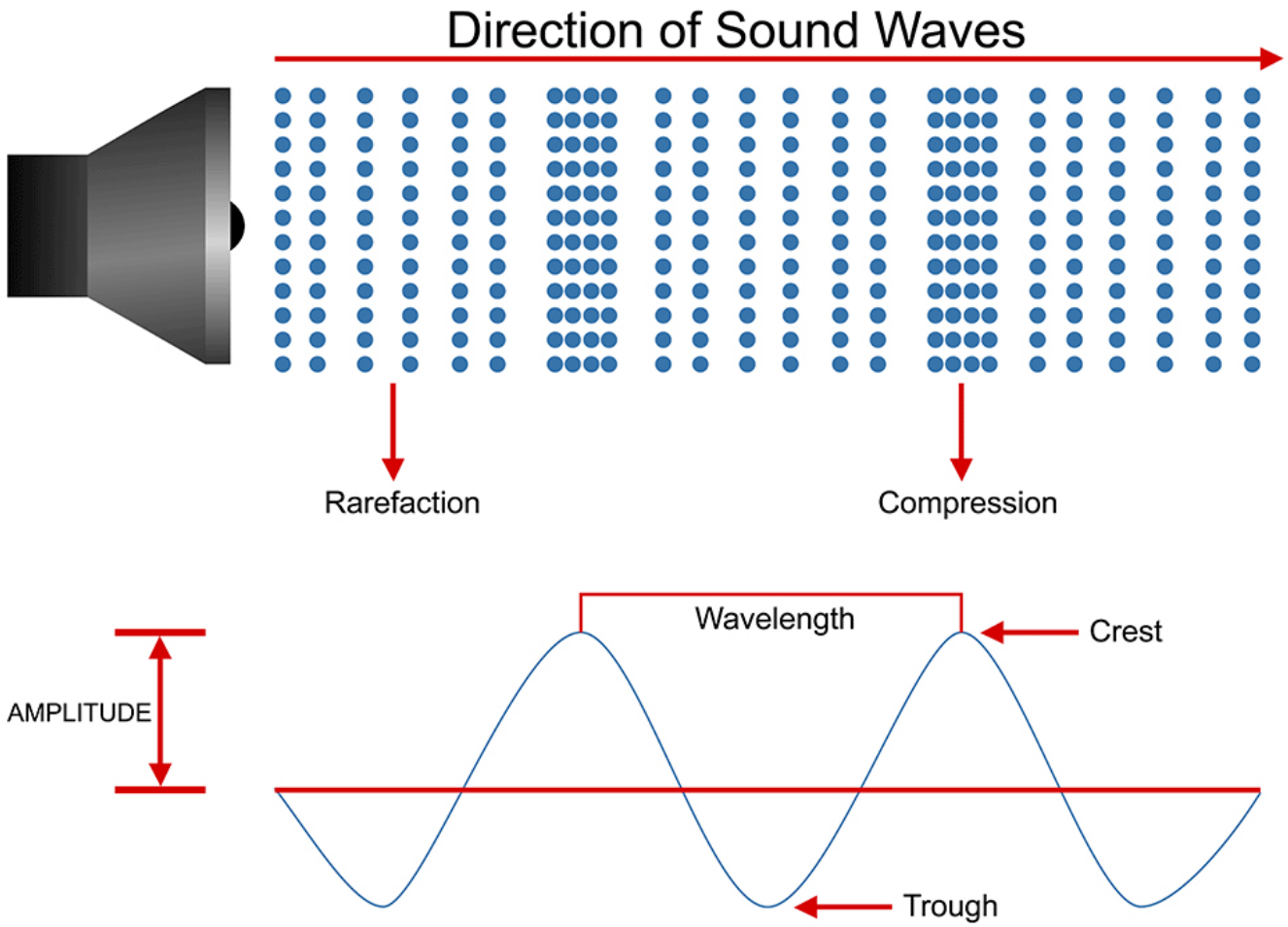


Sound



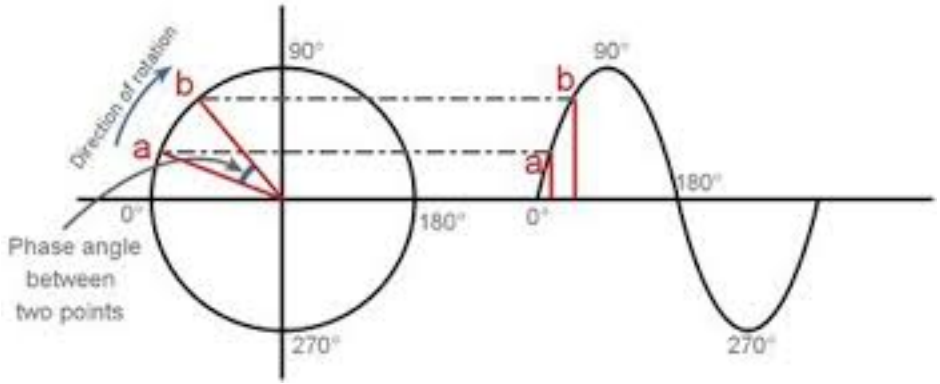
Akrami, A., Kopec, C. D., Diamond, M. E., & Brody, C. D. (2018). Posterior parietal cortex represents sensory history and mediates its effects on behaviour. *Nature*, *554*(7692), 368-372.

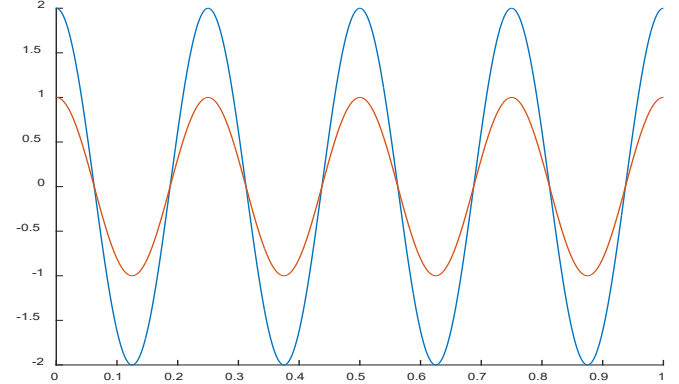
Sound Wave

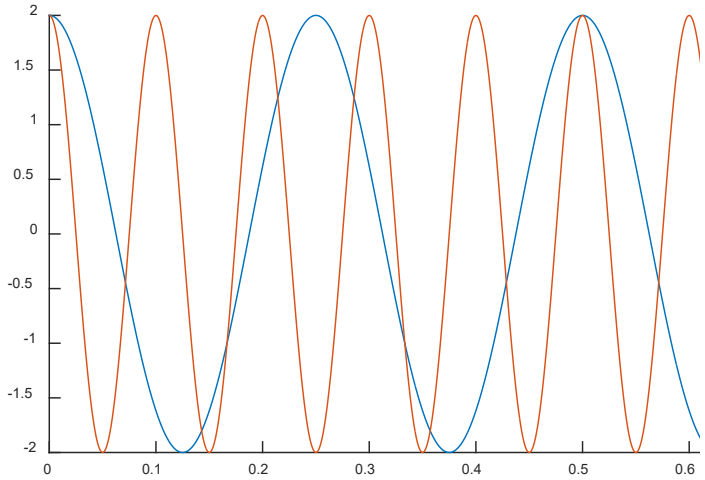

Direction of Sound Waves
Rarefaction
Compression
AMPLITUDE
Wavelength
Crest
Trough

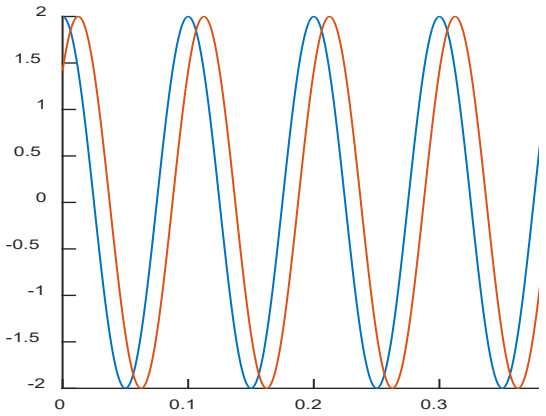
Sound wave

- $v(t) = \mathrm{A}\sin(2 * \pi * f * t + \varphi)$

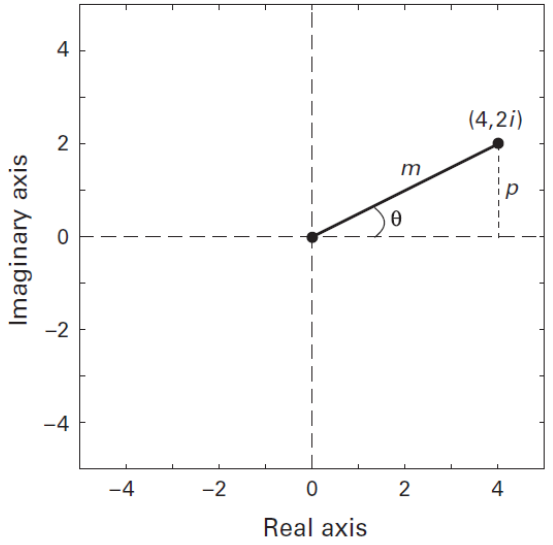




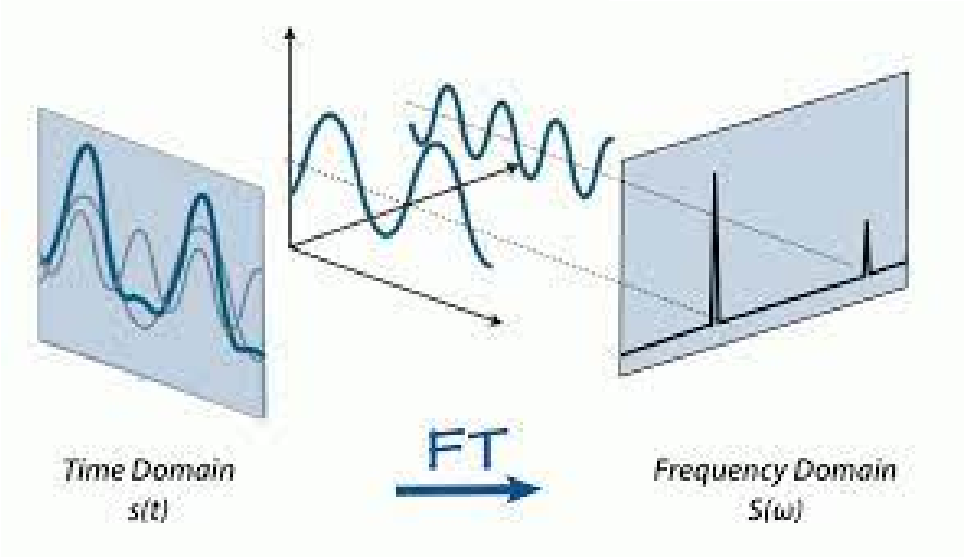




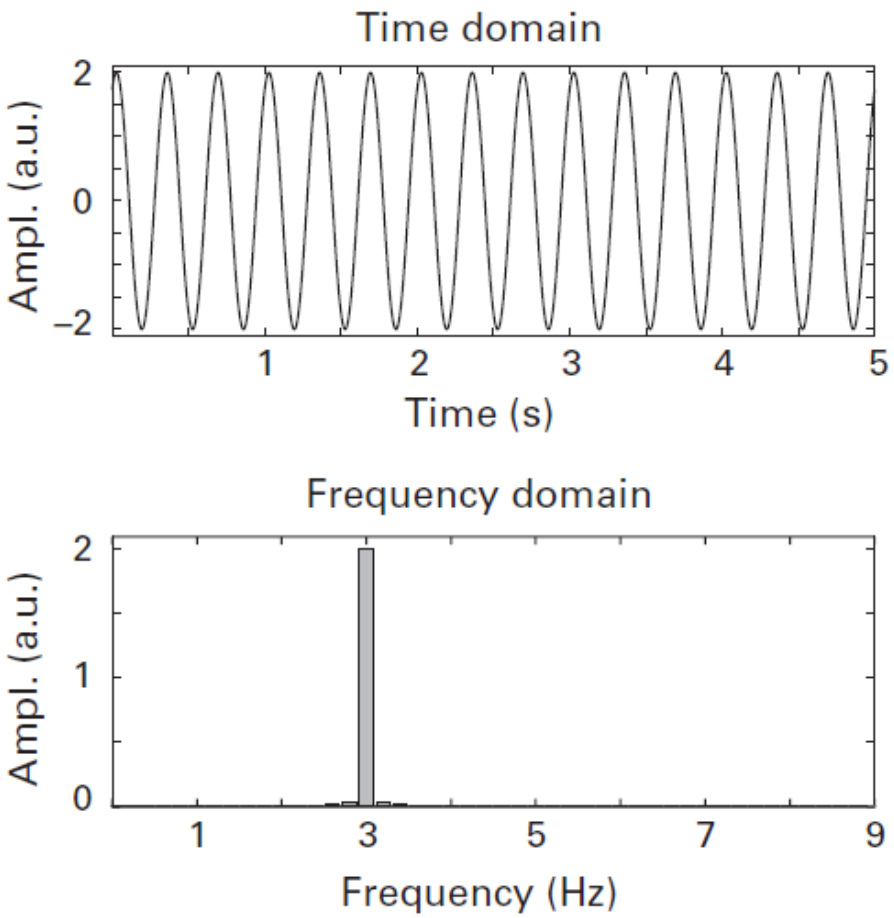
Fourier transform

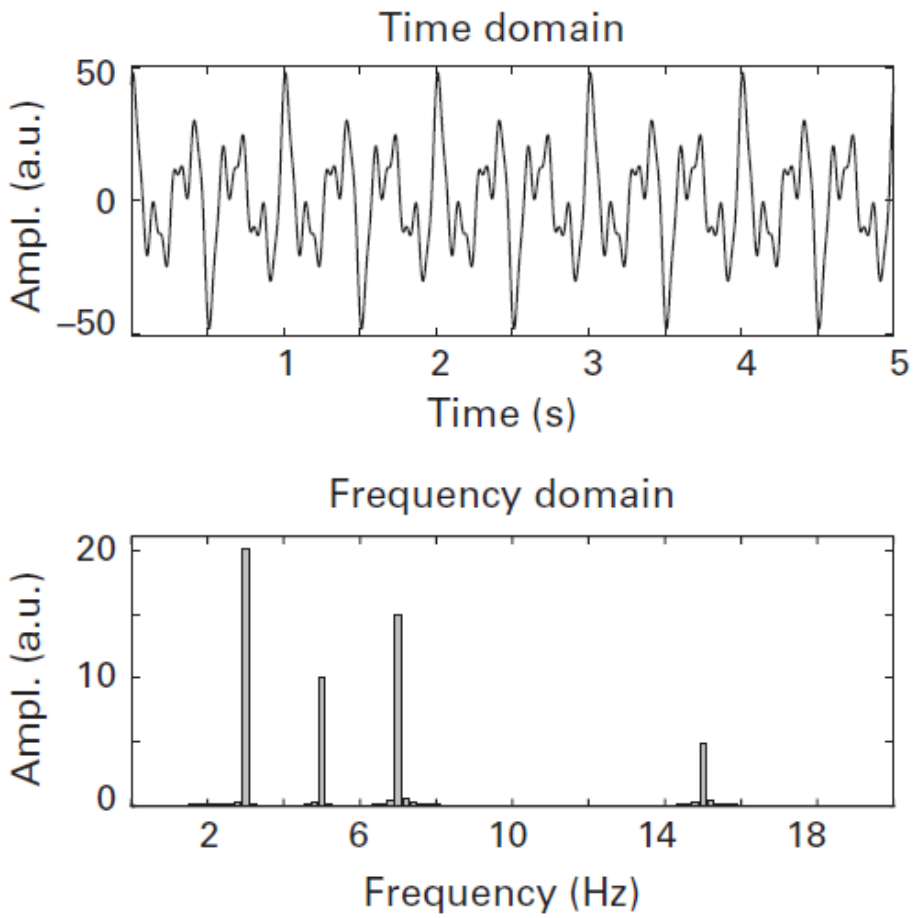


$$
\begin{aligned}
F'(\omega) &= \int_{-\infty}^{+\infty} f(t - t_0)e^{-i\omega t}dt \\
&= \int_{-\infty}^{+\infty} f(\tau)e^{-i\omega(\tau+t_0)}d\tau \\
&= e^{-i\omega t_0}\int_{-\infty}^{+\infty} f(\tau)e^{-i\omega\tau}d\tau
\end{aligned}
\qquad (2.9)
$$

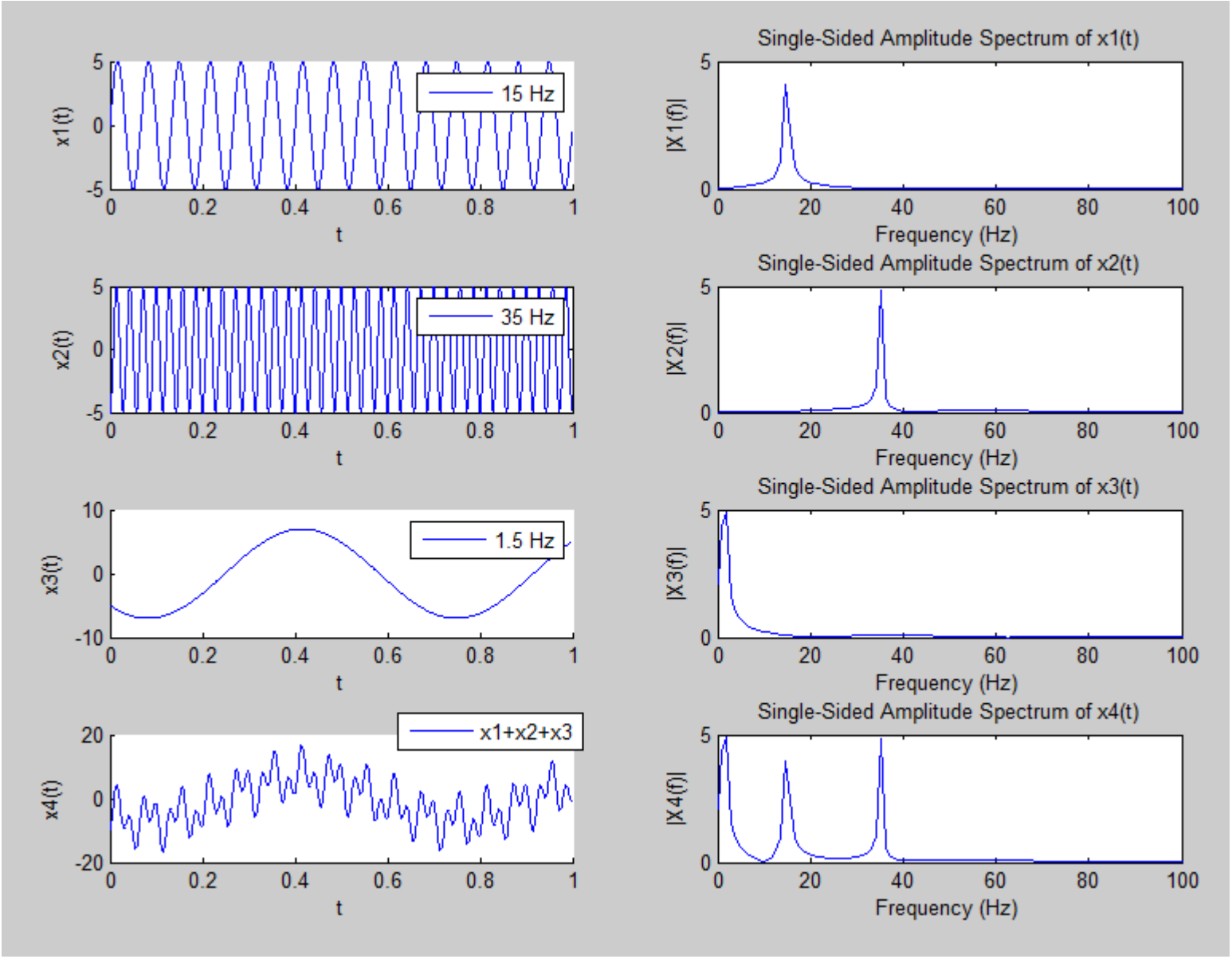


Fourier transform



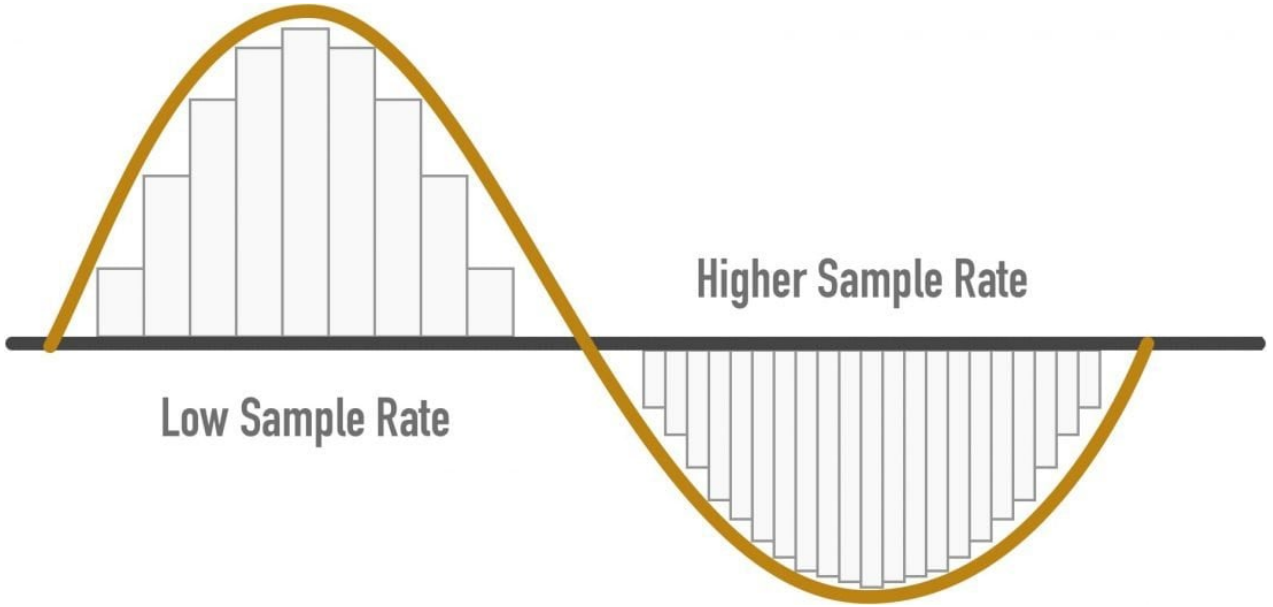


Time domain


15 Hz
35 Hz
1.5 Hz
x1+x2+x3
x1(t)
x2(t)
x3(t)
x4(t)
t
Single-Sided Amplitude Spectrum of x1(t)
Single-Sided Amplitude Spectrum of x2(t)
Single-Sided Amplitude Spectrum of x3(t)
Single-Sided Amplitude Spectrum of x4(t)
|X1(f)|
|X2(f)|
|X3(f)|
|X4(f)|
Frequency (Hz)

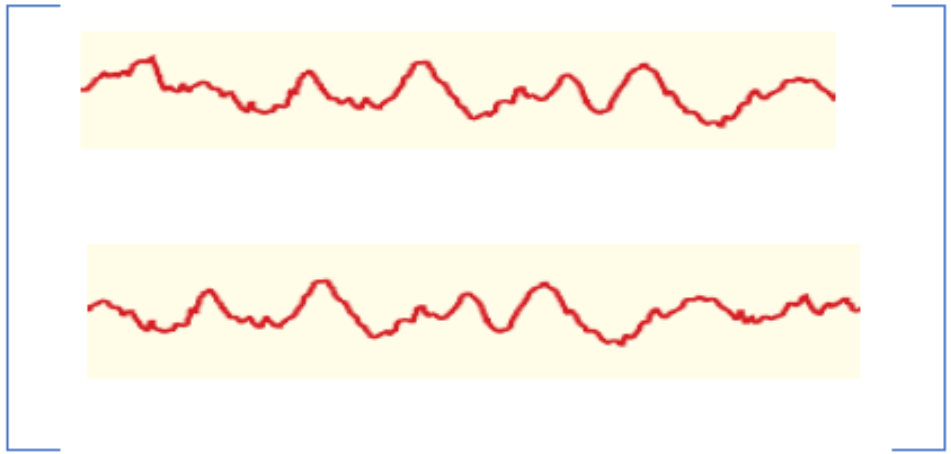
Sound

- Creating sounds to play
- Playing existing sound files



left

right



Sound data

- Sound data should be in the form of a matrix where each row is one sound channel.
- Samples in the vector should range from -1 to 1, where 0 is silent.
- You can create a sound by generating data for a matrix on your own, or you can read in from a wav file

Reading from .au files

Y = auread(AUFILE)

[Y, freq] = auread(AUFILE)

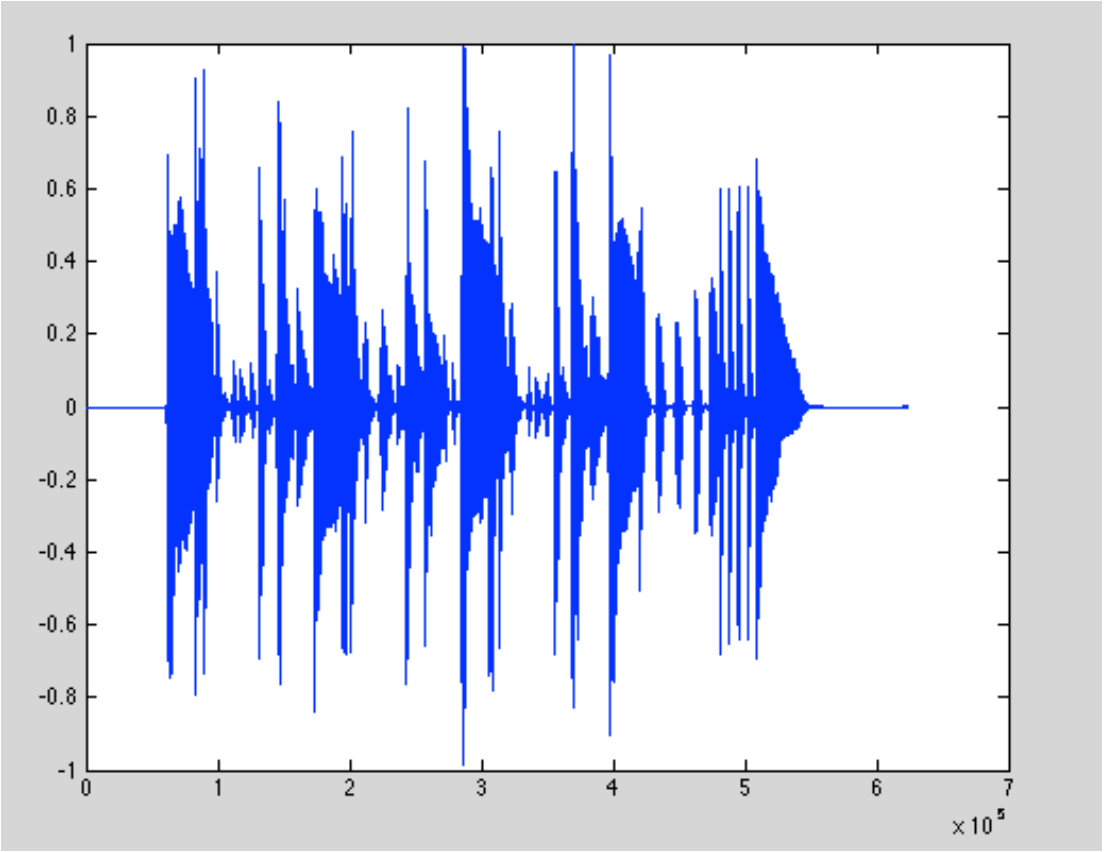
[Y, freq] = psychwavread(AUFILE);

Reading from audiofiles

New Matlab command available in versions 2012b and later, will read many audio formats including WAV, FLAC, MP3, MPEG-4, OGG

[Y, freq] = audioread()

Reading in sounds



Preparing sound data for playing

```
>> whos funkData
 Name                Size                Bytes  Class     Attributes

  funkData       624000x1             4992000  double

>> funkData = funkData'
>> funkData = [funkData; funkData];
>> whos funkData
 Name           Size                   Bytes  Class     Attributes

  funkData       2x624000             9984000  double
```

change column to row

duplicate to make two rows for stereo

Creating sound stimuli (white noise)

- Length of vector is sampling frequency * duration (we want sfreq samples per second for X seconds)

```
>> samplingFreq = 48000;
>> duration = 5;
>> whitenoise = rand(1,(samplingFreq * duration));
>> whos whitenoise
Name            Size                  Bytes  Class     Attributes

  whitenoise      1x240000            1920000  double
```

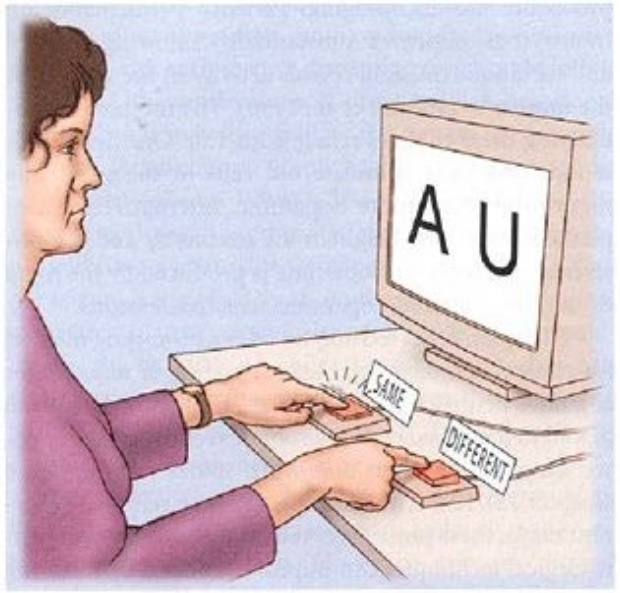
Creating sounds

- MakeBeep() will create a pure tone
- Y = MakeBeep(freq, duration, samplingrate)

```
>> beep1000 = MakeBeep(1000,5,48000);
>> sound(beep1000,48000);
>> beep500 = MakeBeep(500,5,48000);
>> sound(beep500,48000);
```

matlab's built-in sound function, not PTB's

- Auditory
- Visual
- Olfactory
- Somatosensory
- Gustatory



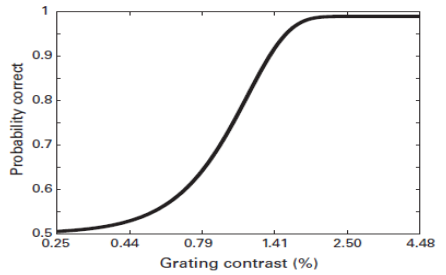
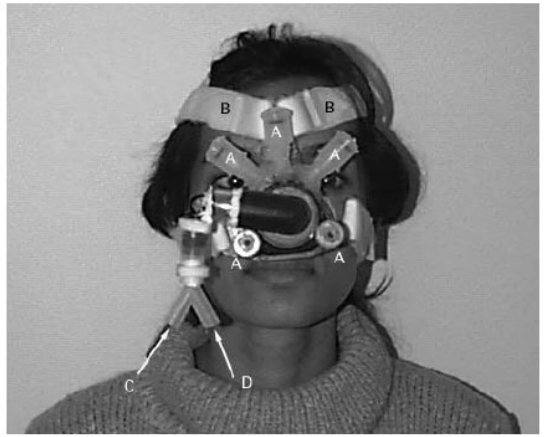


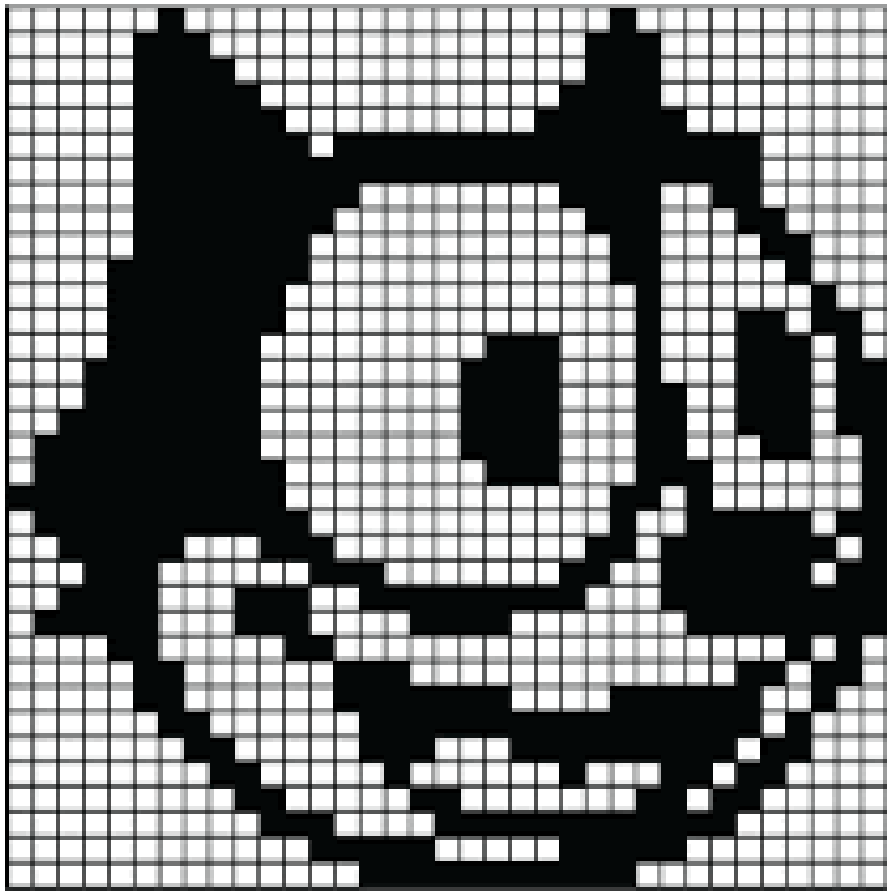
Figure 2.2
Psychometric function for detecting sine-wave gratings of different contrasts.

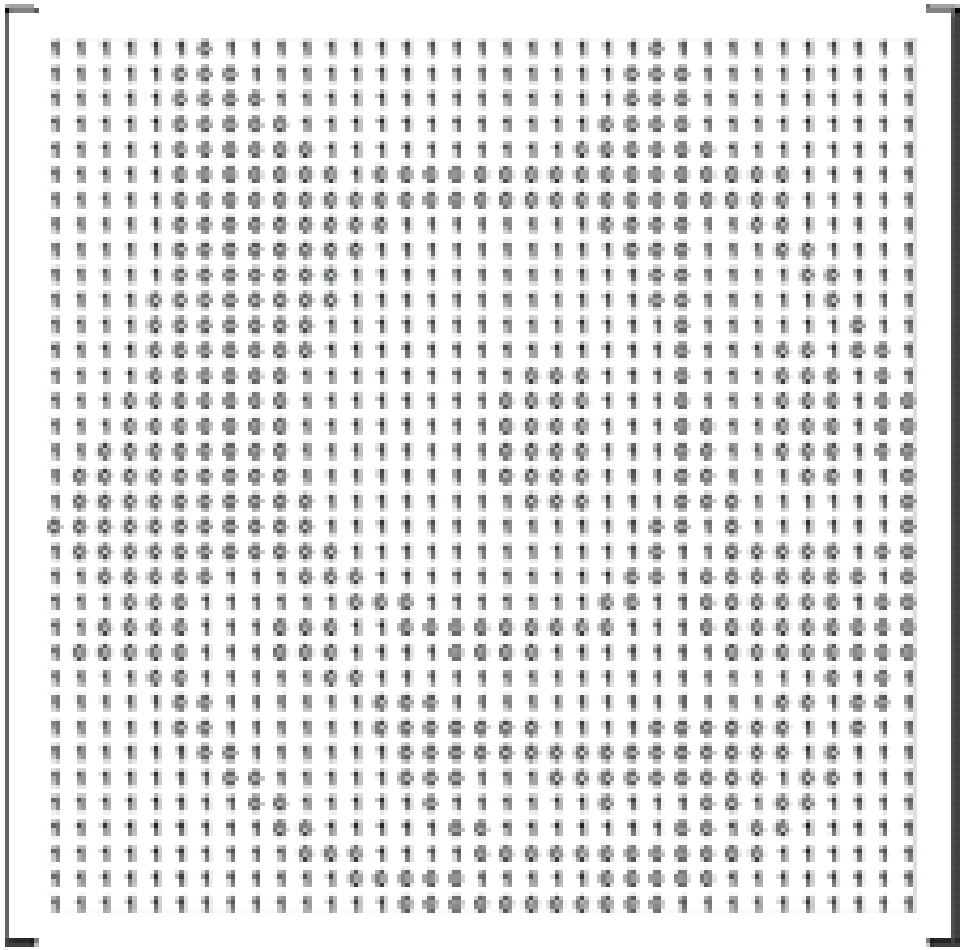


a

Images are matrix in MATLAB

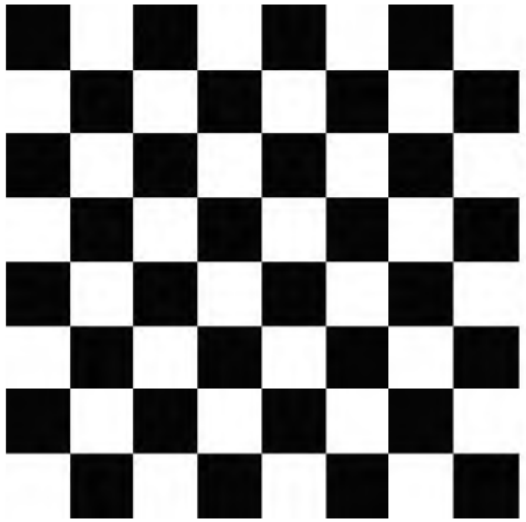
- Gray Image





Example

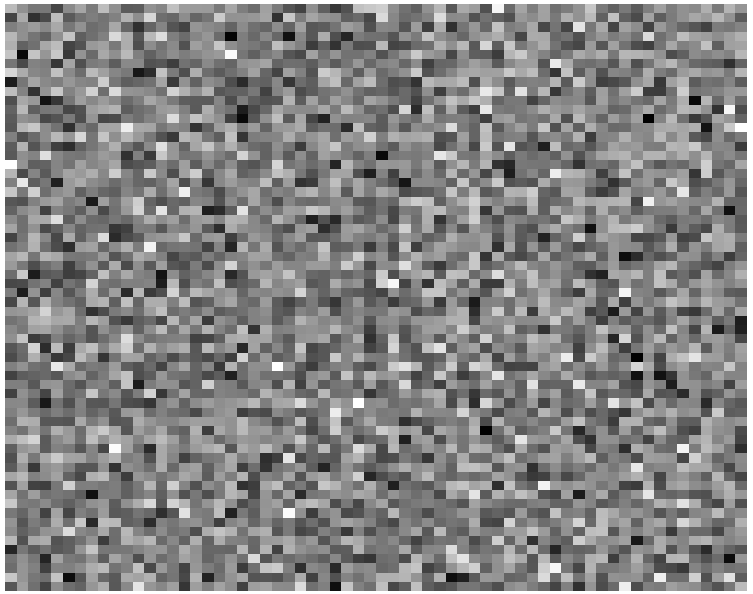
```
%%% Program Checkboard1.m
M = [1 2 1 2 1 2 1 2
     2 1 2 1 2 1 2 1
     1 2 1 2 1 2 1 2
     2 1 2 1 2 1 2 1
     1 2 1 2 1 2 1 2
     2 1 2 1 2 1 2 1
     1 2 1 2 1 2 1 2
     2 1 2 1 2 1 2 1];
```



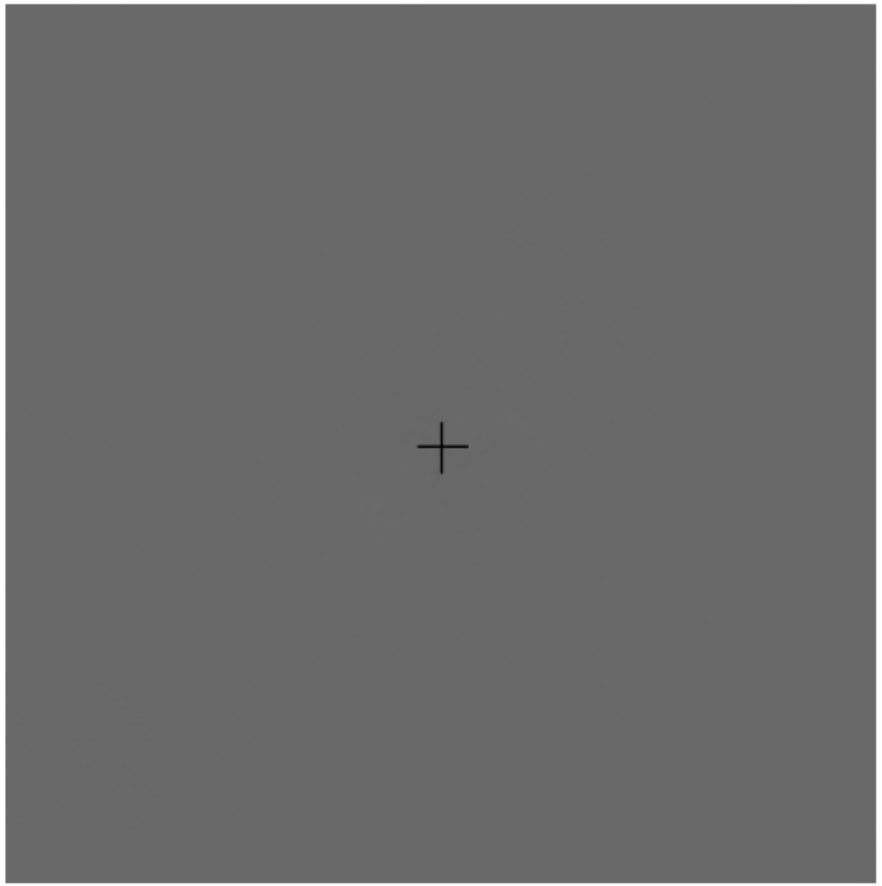
```
%%% Program Checkboard2.m
for i = 1:8
    for j = 1:8
        M(i,  j) = mod(i + j,  2) + 1;
    end
end
```

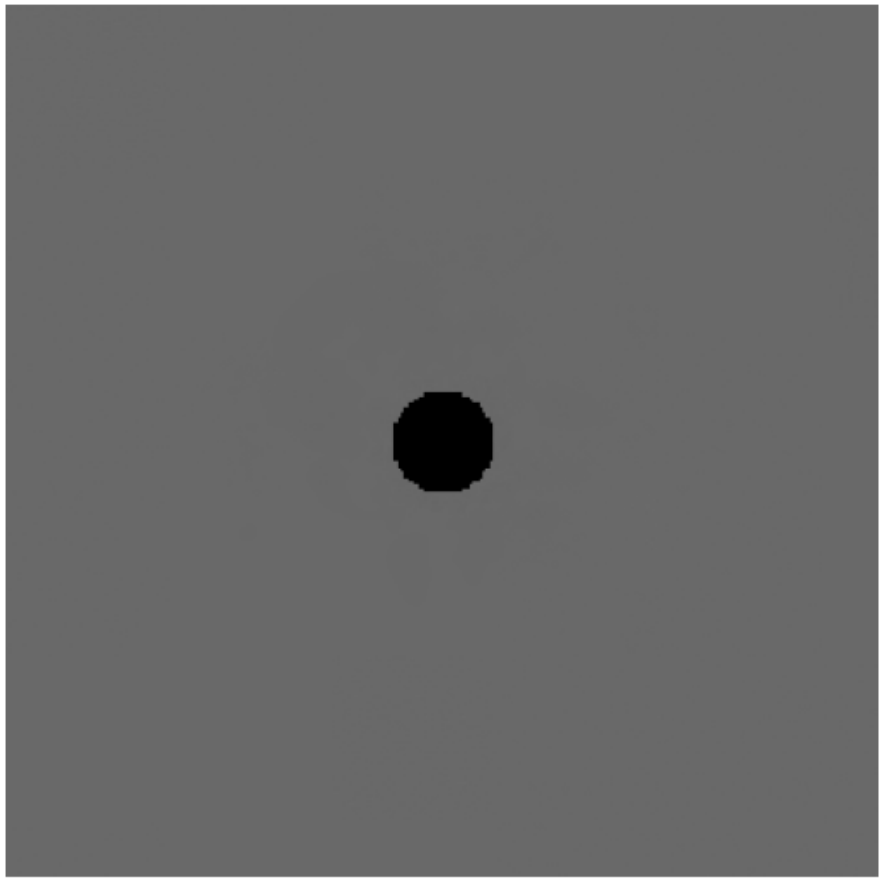
```
%%% Program Checkboard3.m
[x, y] = meshgrid(-4:3,  4:-1:-3);
M = mod(x + y,  2) + 1;
```

White noise image



Fixation Cross or dot





Image

- Image read
- Image format
- Image size
- Image crop
- Image location
- Merging two image
- Noise masking

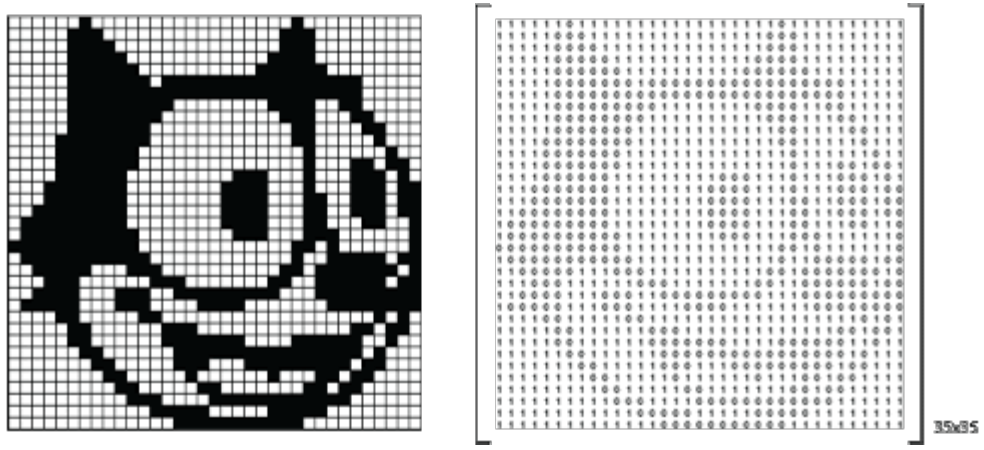
Image read

- A=imread(fileName)

```
%%% read Image
img = imread('milad.jpg');
showImage(img, 'grayscale');
```

Image is a matrix

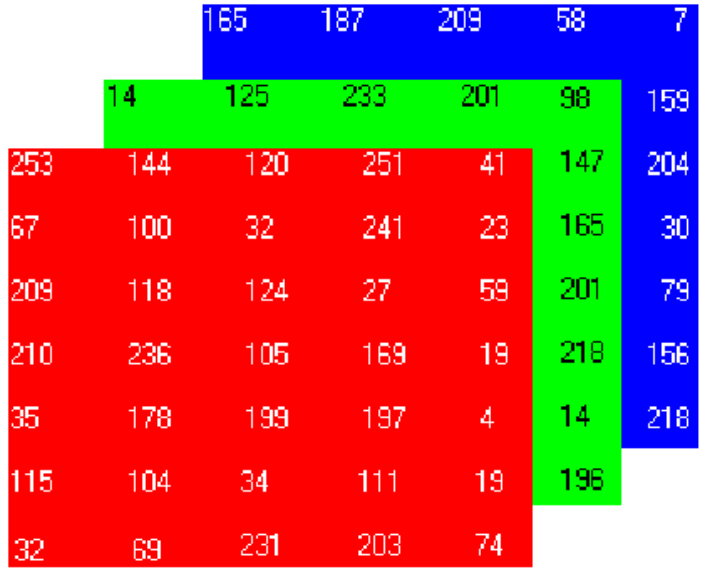
- Gray Image



- Color Image

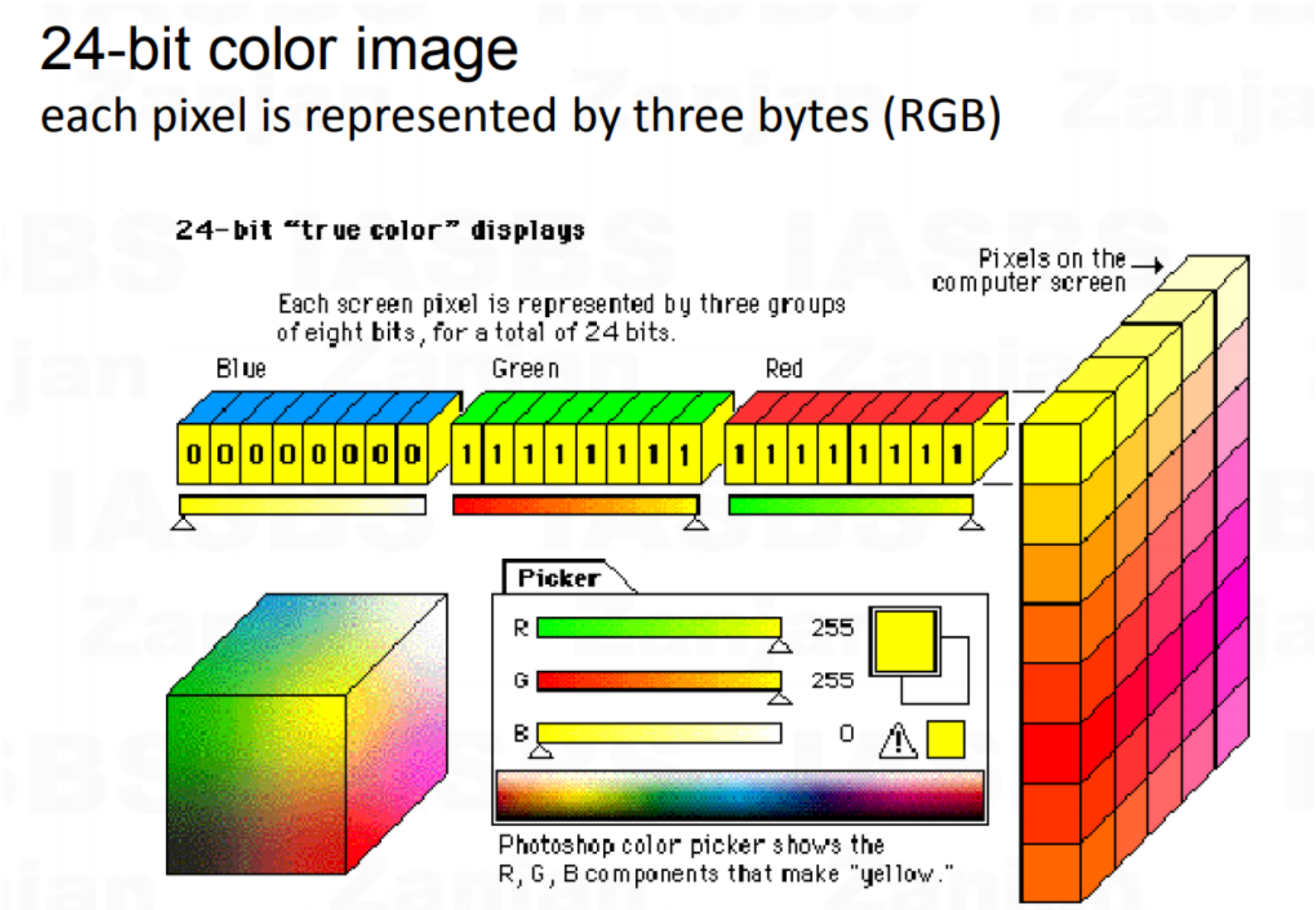
Bit-Depths Converted To Potential Gray Tones and Colors

Bits Per Color	Log Formula (power of 2)	Monochrome Grayscale Values	Potential R,G,B Color Values
1-bit	2^1 =	2	8
2-bit	2^2 =	4	64
3-bit	2^3 =	8	512
4-bit	2^4 =	16	4096
6-bit	2^6 =	64	262144
8-bit	2^8 =	256	16.77 Million
10-bit	2^{10} =	1024	1.07 Billion
11-bit	2^{11} =	2048	8.59 Billion
12-bit	2^{12} =	4096	68.72 Billion

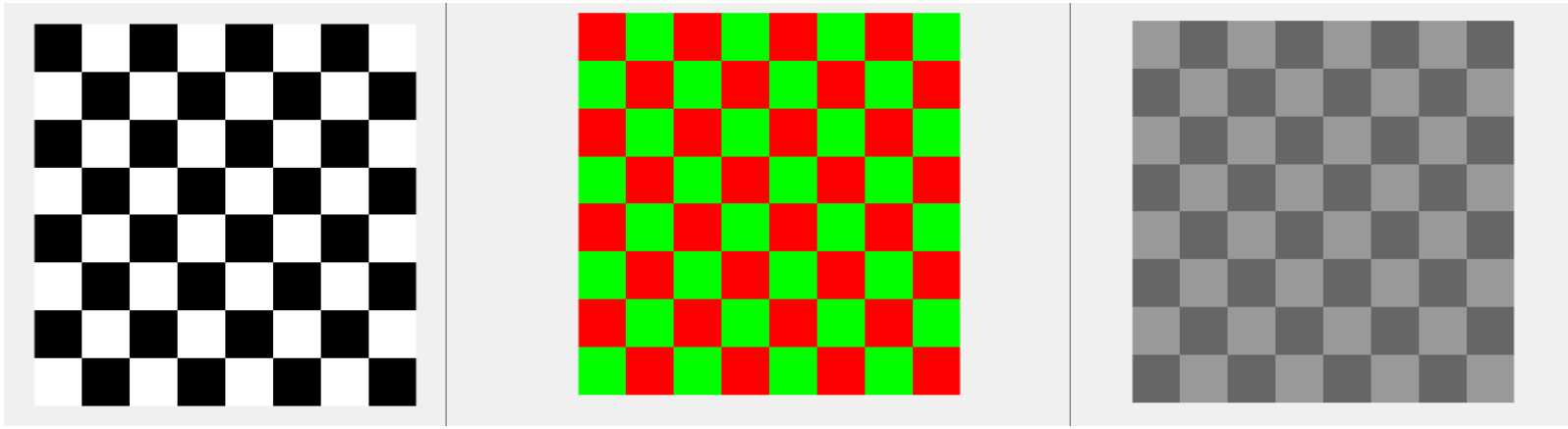


Color lookup table

- Colormap
- M*3 matrix show value for each data



Example



```
%%% Program Checkboard3.m
[x, y] = meshgrid(-4:3,  4:-1:-3);
M = mod(x + y,  2) + 1;
```

```
lut1 = [0 0 0;  1 1 1]; % define a two row lookup table for
colormap(lut1);         % the color translation of 1,2
```

```
lut2 = [1 0 0;  0 1 0]; % define a two row lookup table for
colormap(lut2);         % the color translation of 1,2
```

```
lut3 = [0.6 0.6 0.6; 0.4 0.4 0.4]; % define a two row lookup table for
colormap(lut3);         % the color translation of 1,2
```

How convert Color image to Grayscale

- rgb2gray ()

Grayscale = (R + G + B) / 3

Grayscale = 0.299R + 0.587G + 0.114B.

Image format

MATLAB supports the following graphics file formats, along with others:

- BMP (Microsoft® Windows® Bitmap)
- GIF (Graphics Interchange Files)
- HDF (Hierarchical Data Format)
- JPEG (Joint Photographic Experts Group)
- PCX (Paintbrush)
- PNG (Portable Network Graphics)
- TIFF (Tagged Image File Format)
- XWD (X Window Dump)

Image size

```
[rows, columns, numberOfColorChannels] = size(img);
```

Image resize

```
Image1 = imread('milad.jpg');
Image2 = imresize(Image1, 0.3);

imshow(Image2)
title('Resized Image')
```

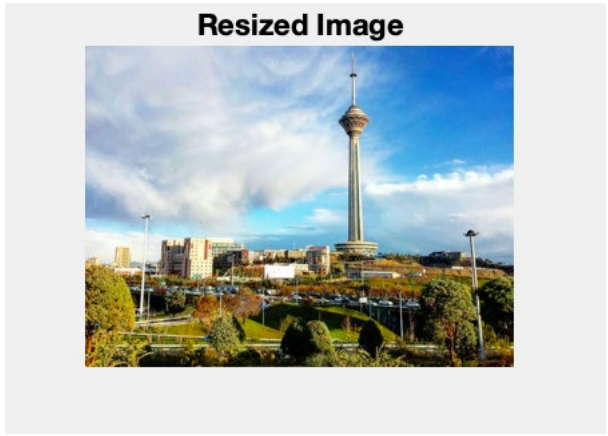


Image crop

- cropped=imcrop

Selected location to crop

```
%%% Image crop
IMG1 = imread('milad.jpg');
IMG2 = imcrop(IMG1,[300 400 500 600]);
subplot(1,2,1)
imshow(IMG1)
title('Original Image')
subplot(1,2,2)
imshow(IMG2)
title('Cropped Image')
```

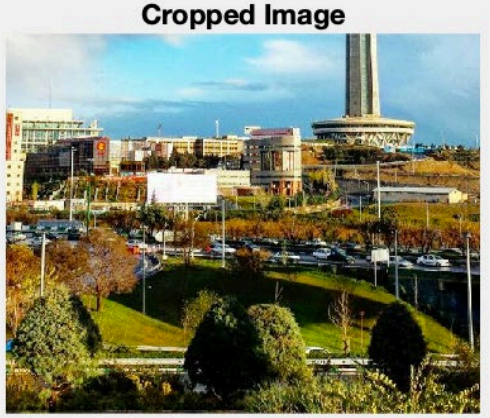


Image location

```
Image1 = imread('cameraman.tif');
Image2 = imread('moon.jpg');

% Display image 1 .
subplot(1, 2, 1);
imshow(Image1);
title('cameraman.tif', 'FontSize', fontSize, 'Interpreter', 'None');
% Set up figure properties:
% Enlarge figure to full screen.
set(gcf, 'Units', 'Normalized', 'OuterPosition', [0 0 1 1]);
% Display image 2.
subplot(1, 2, 2);
imshow(Image2);
title('moon.jpg', 'FontSize', fontSize, 'Interpreter', 'None');
```

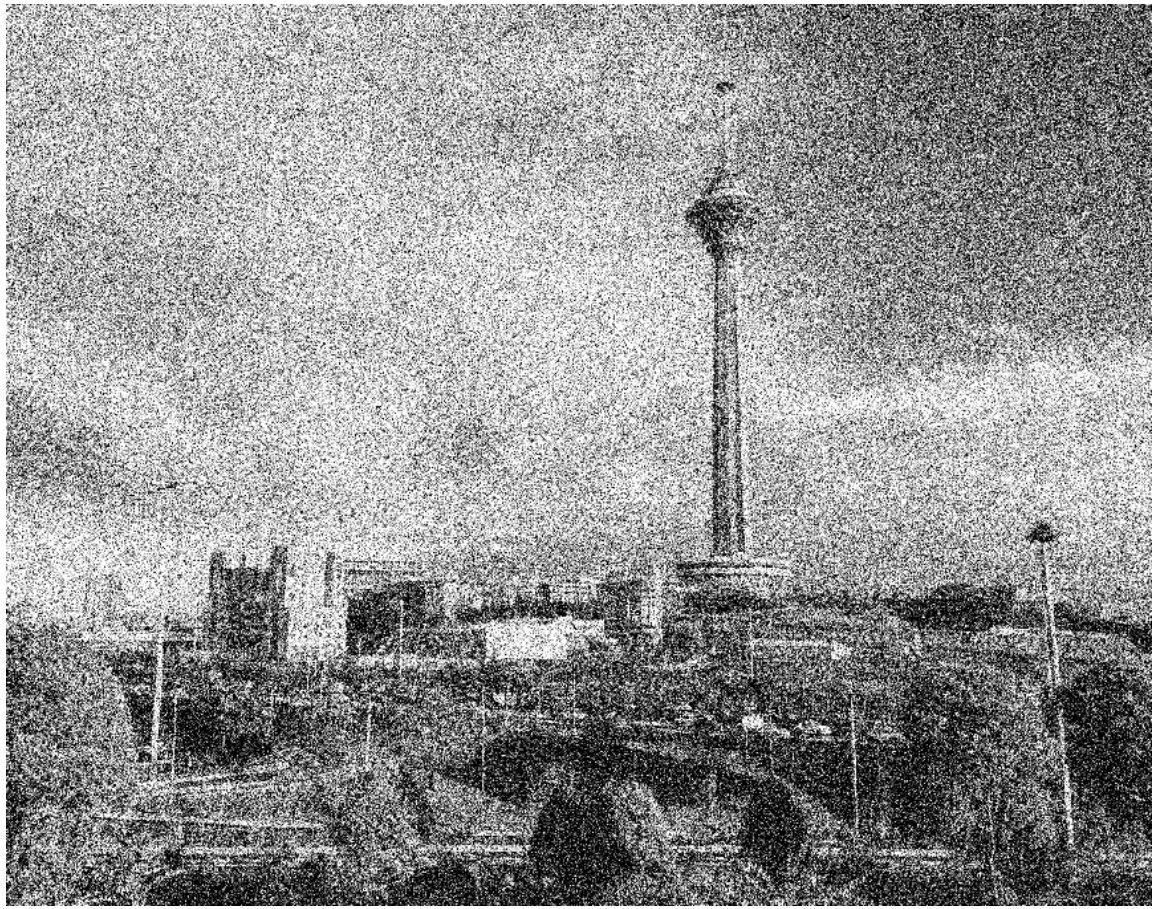
Set (set) the current figure (gcf) to have *'Units'* be *'normalized'* and the *'outerposition'*, i.e. the actual borders of the figure to be at the bottom left corner (0,0) and span the whole screen (1,1).

Merging two image

```
Image3 = imfuse(Image1,Image2,'blend','Scaling','joint');
```

creates a composite image from two images, Image1 and Image2. If Image1 and Image2 are different sizes, imfuse pads the smaller dimensions with zeros so that both images are the same size before creating the composite

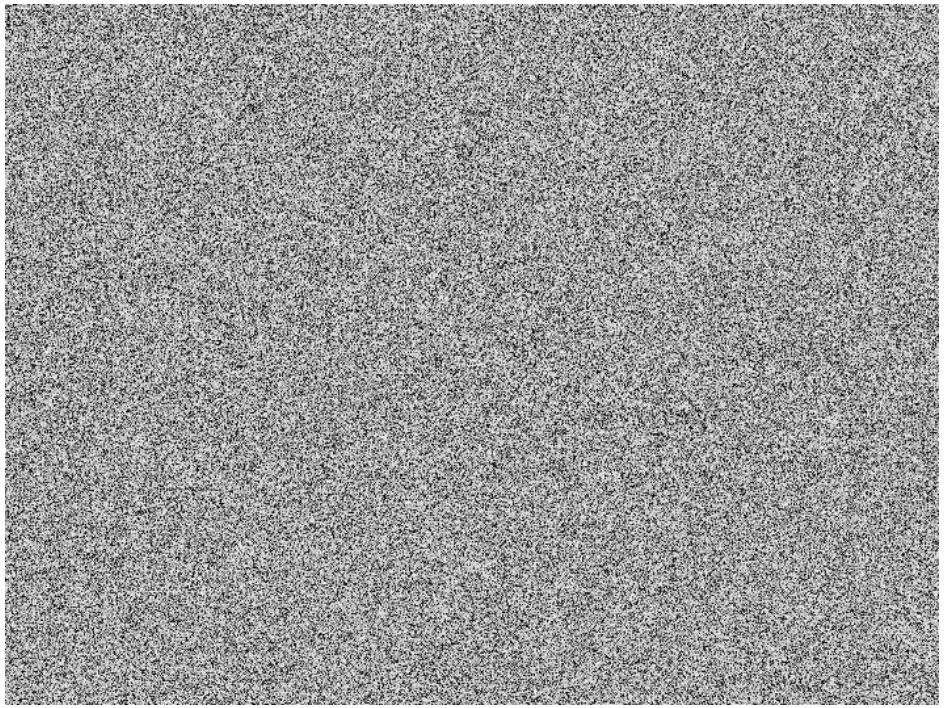
Noise masking



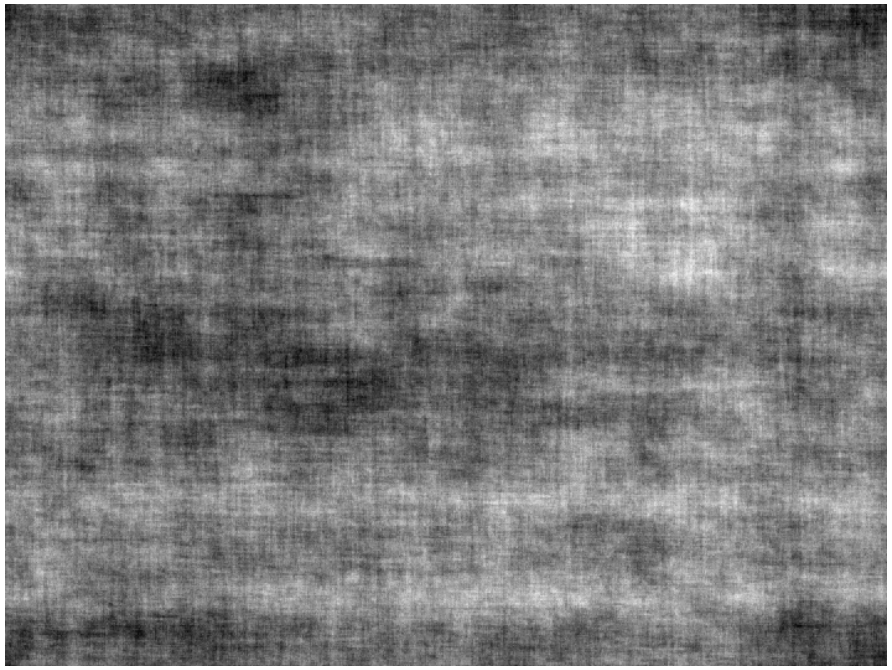
Making scrambled Images

- Pixel scrambled
- Phase scrambled
- Texture scrambled

Pixel scrambled



Phase scrambled



Texture scrambled

original

Scrambled

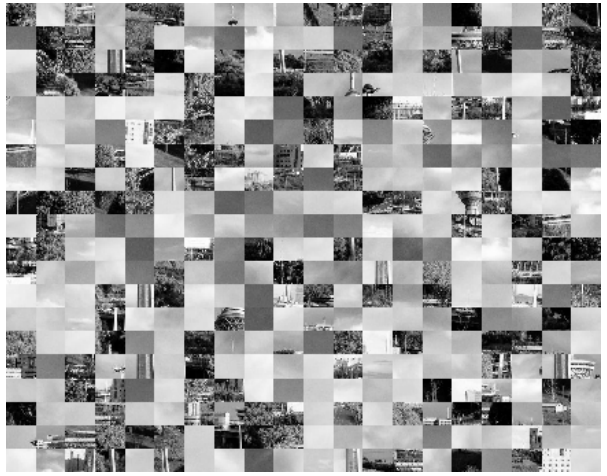
Divider=2

Scrambled

Divider=8

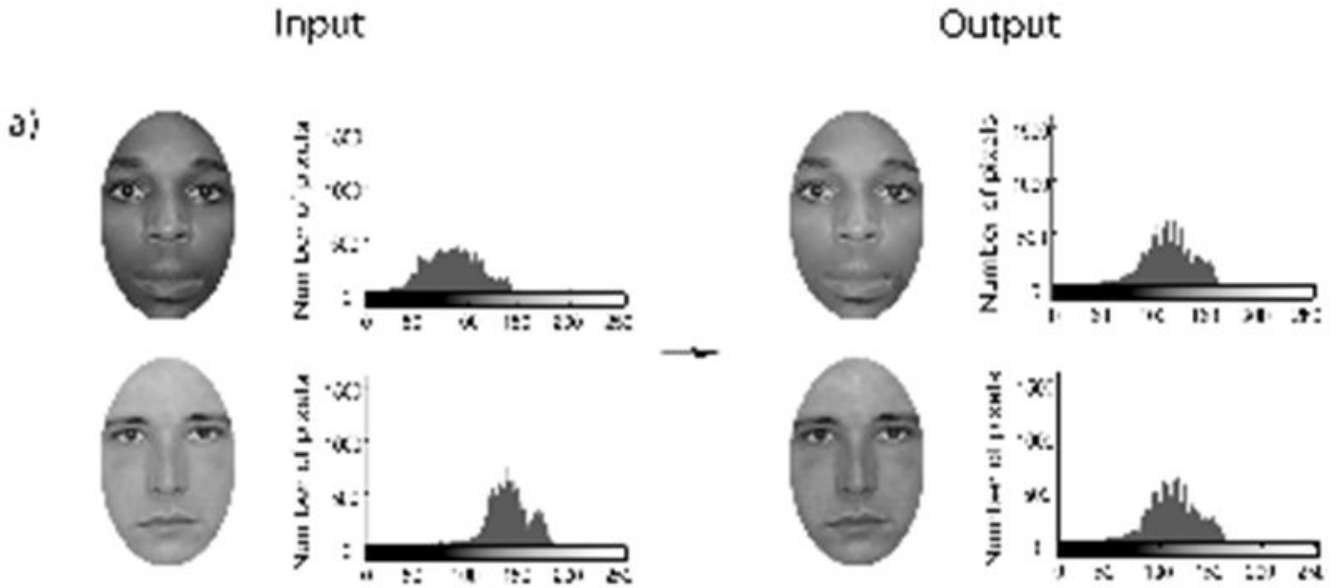
Scrambled

Divider=20



Shine toolbox

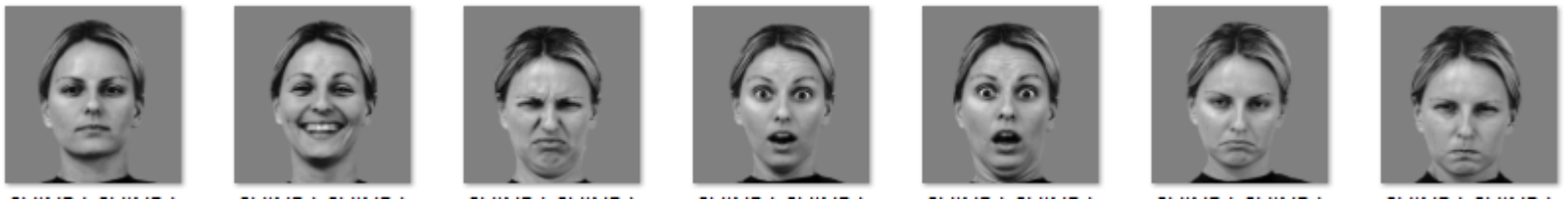
- https://link.springer.com/article/10.3758/BRM.42.3.671



Shine toolbox

match luminance

```
>> SHINE
SHINE options    [1=default, 2=custom]: 2
Matching mode    [1=luminance, 2=spatial frequency, 3=both]: 1
Luminance option [1=lumMatch, 2=histMatch]: 2
Optimize SSIM    [1=no, 2=yes]: 1
Matching region  [1=whole image, 2=foreground/background]: 1

Number of images: 2

Option:   histMatch on the whole images
Progress: histMatch successful

RMSE:     1.106239e+01
SSIM:     9.554870e-01
```



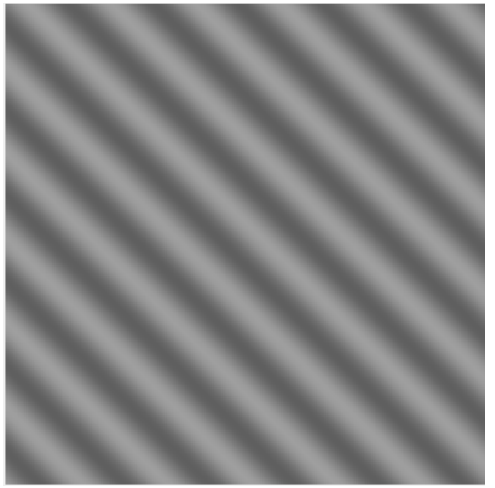
match spatial frequency

```
>> SHINE
SHINE options    [1=default, 2=custom]: 2
Matching mode    [1=luminance, 2=spatial frequency, 3=both]: 2
Spectrum options [1=sfMatch, 2=specMatch]: 1

Number of images: 14

Option:   sfMatch
Progress: sfMatch successful

RMSE:     1.354226e+01
SSIM:     9.618390e-01
```

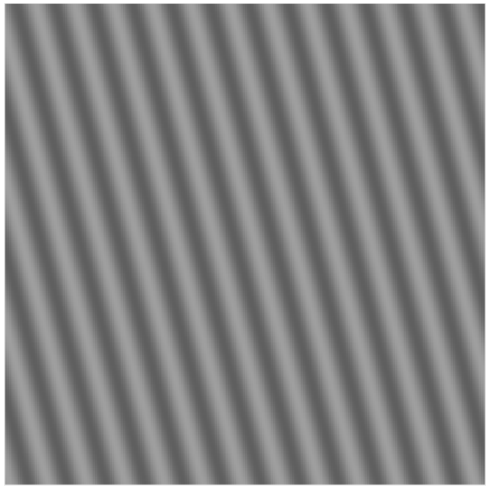
Spatial frequency

```
c = 0.25;                % contrast of the Gabor
f = 1/32;                % spatial frequency in 1/pixels
t = 35*pi/180;           % tilt of 35 degrees into radians
s = 24;                  % standard deviation of the spatial
                         % window of the Gabor
[x,  y] = meshgrid(-128:127, 128:-1:-127);
M1 = uint8(127*(1 + c*sin(2.0*pi*f*(y*sin(t) + x*cos(t)))));
        % uint8 converts the elements of the array into unsigned
        % 8-bit integers. Values outside this range are mapped
        % to 0 or 255.
showImage(M1,  'grayscale');
```

f = 1/32
t = 45*pi/180



f = 1/20
t = 15*pi/180



Spatial frequency

```
M2 = uint8(127*(1 + c*sin(2.0*pi*f*(y*sin(t) + x*cos(t))) ...
     .*exp(-(x.^2 + y.^2)/2/s^2)));
showImage(M2,   'grayscale');
```

Session #5 assignment

- Write a function named “yourInitials_session3()”

  1- The function should read a color image from your MATLAB current directory (copy an image before running there)

  2- convert color image to gray and preview both by subplot beside each other

  3- resize gray scale image to specific size (100*100) preview both by subplot beside each other.

  4- save you resized image in current directory

  5- add noise to image