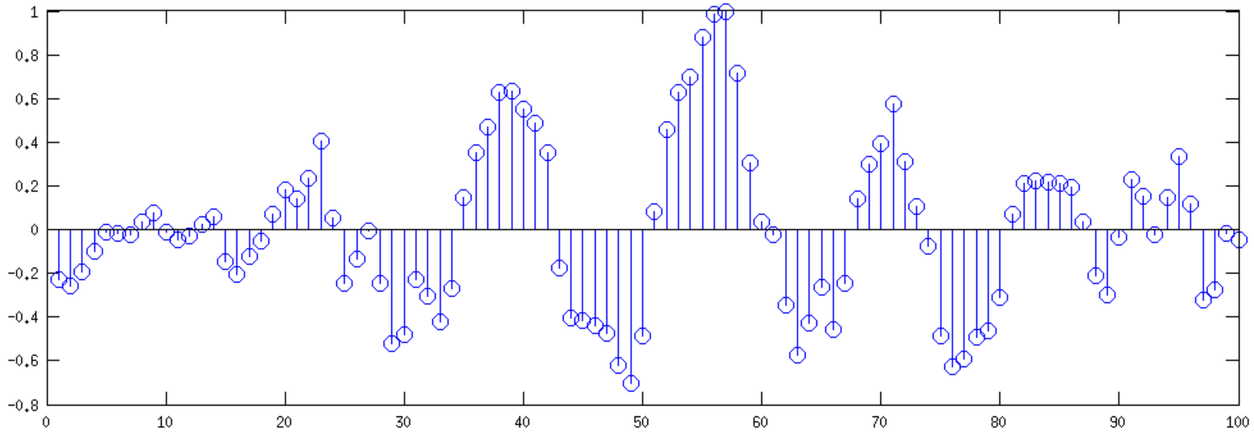
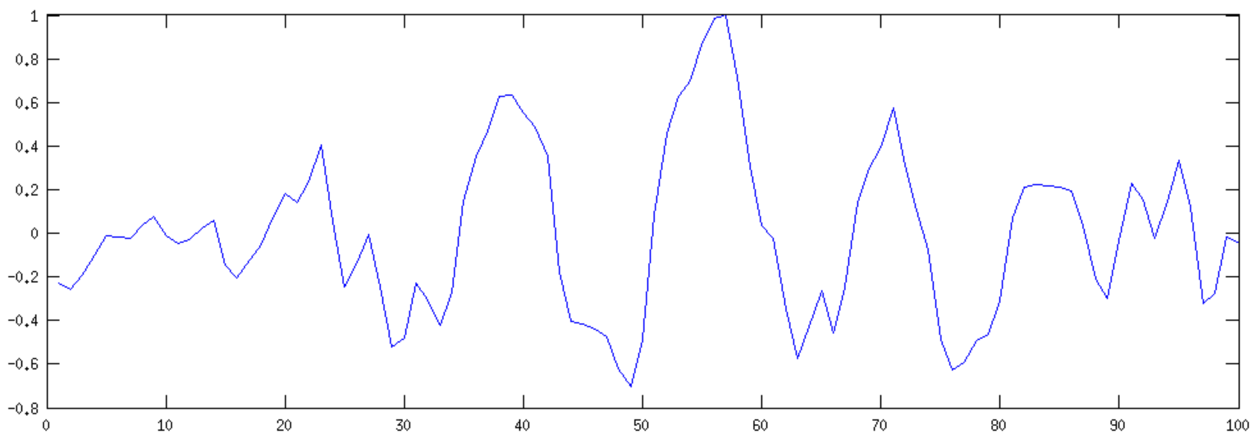
# Image processing

Honza Černocký, ÚPGM

# 1D signal

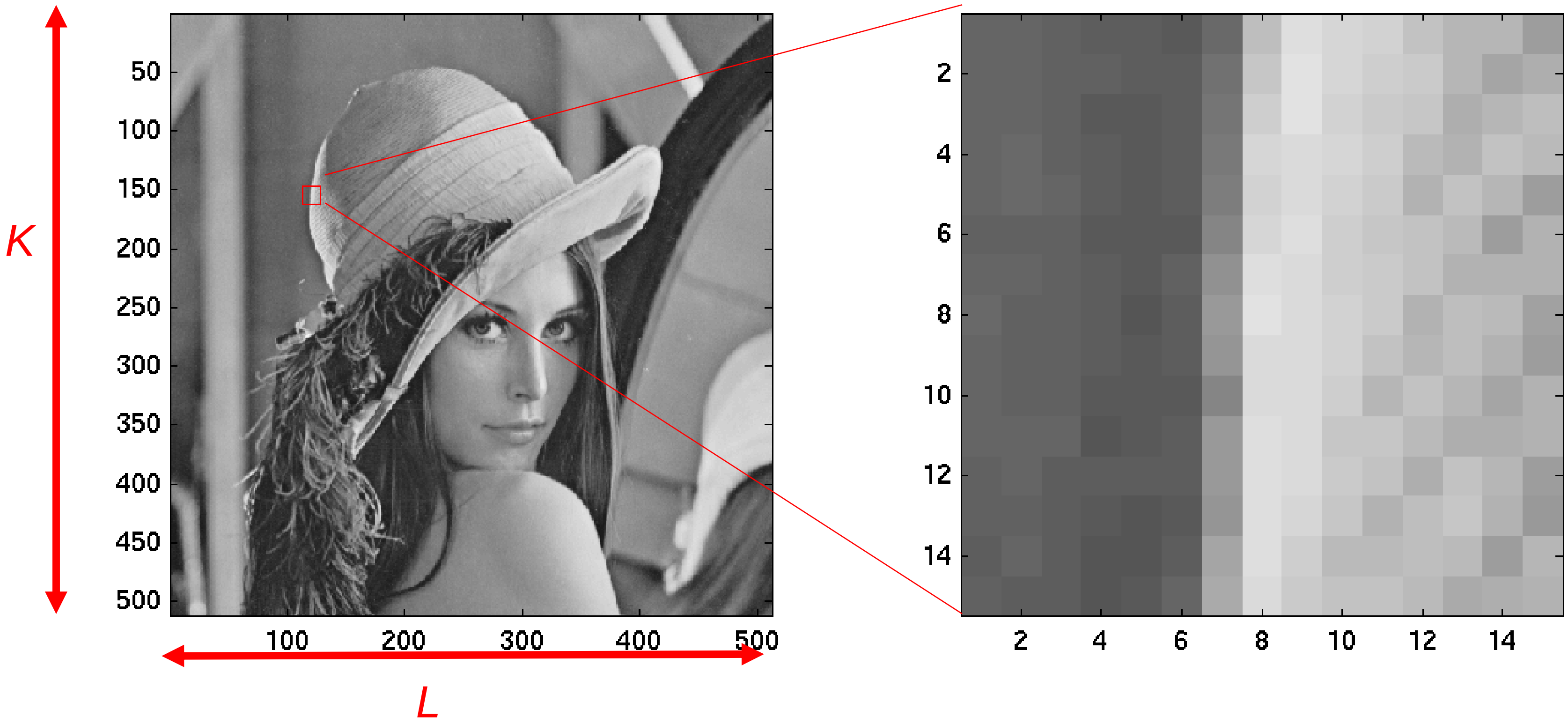

1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
10
20
30
40
50
60
70
80
90
100


1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
10
20
30
40
50
60
70
80
90
100

# Images

- 2D – grayscale photo
- Multiple 2D – color photo
- 3D – medical imaging, vector graphics, point-clouds (depth information, Kinect)
- Video – time is another dimension

# Image



K
L
50
100
150
200
250
300
350
400
450
500
100
200
300
400
500
2
4
6
8
10
12
14
2
4
6
8
10
12
14

# Mathematically

$x[k, l]$

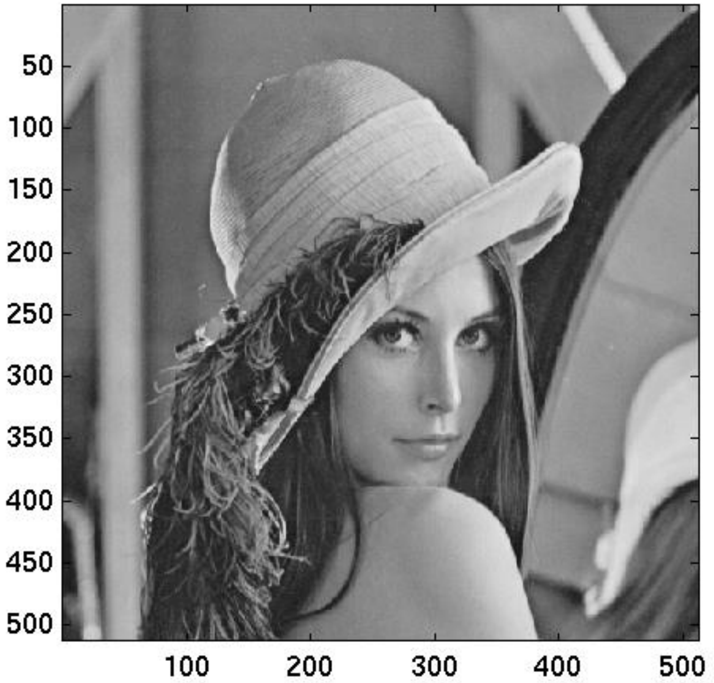
$$x[k,l] = \begin{bmatrix} x[0,0] & x[0,1] & \cdots & x[0,L-1] \\ x[1,0] & x[1,1] & \cdots & x[1,L-1] \\ \vdots & & & \vdots \\ x[K-1,0] & x[K-1,1] & \cdots & x[K-1,L-1] \end{bmatrix}$$

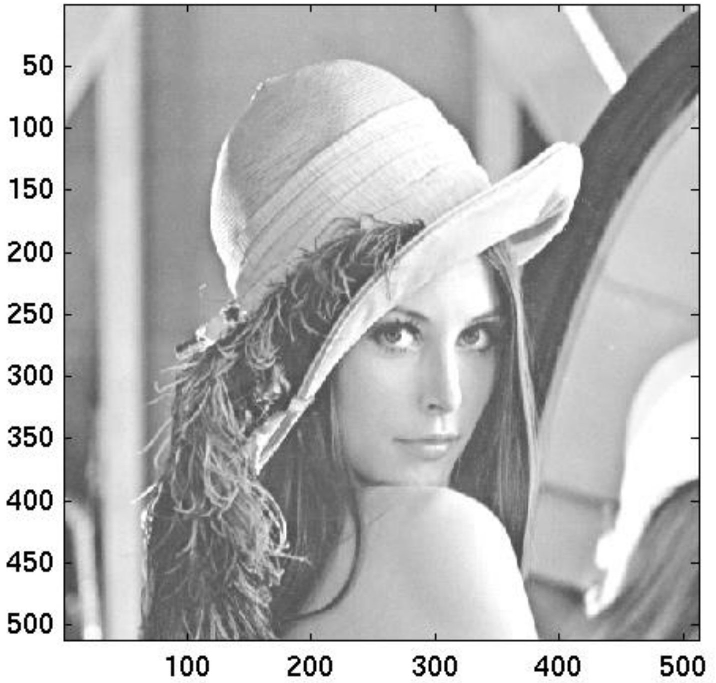
# Representation of pixels

- For storage – quantization
- 8 (normal) / 16 (HDR) bits
- 0…255 etc
- For computing [0…1]
- **floats**

# Operations on pixels – more brightness

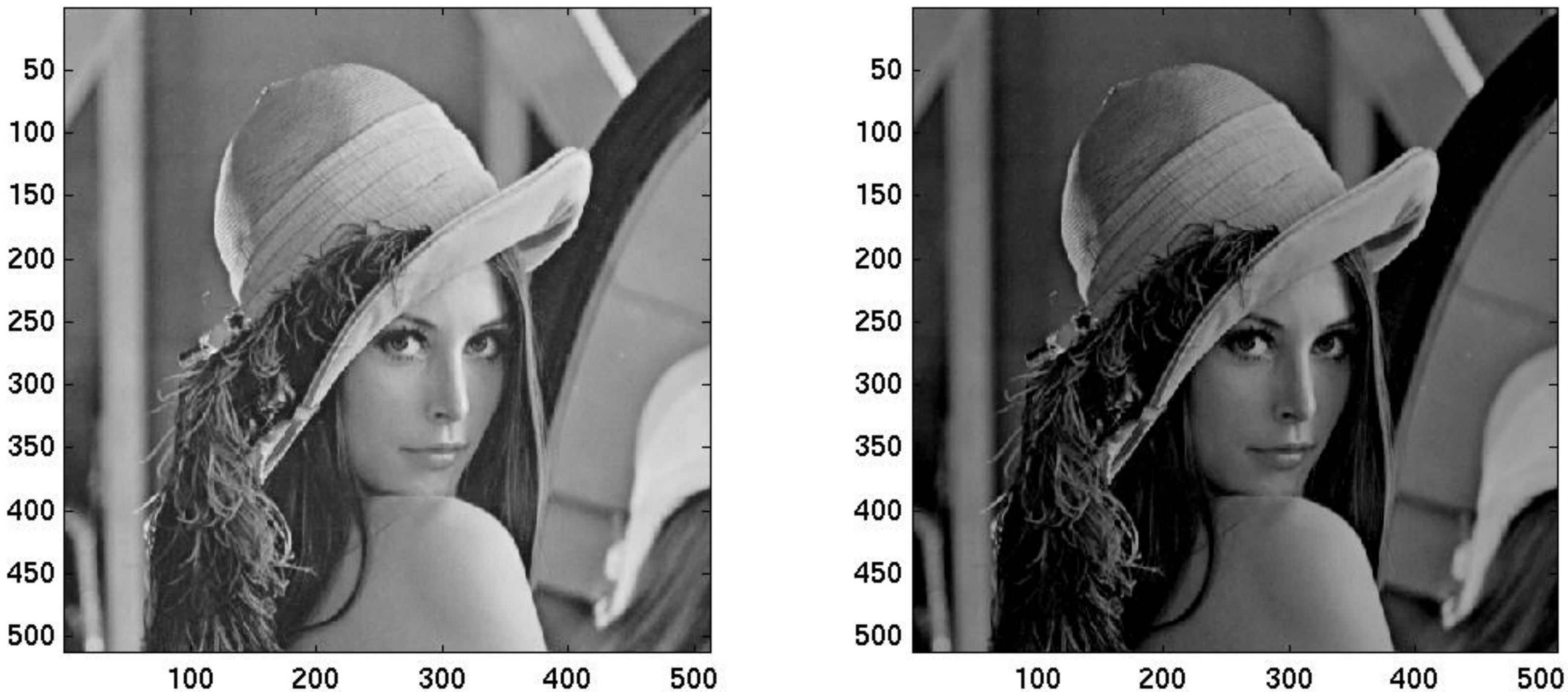
$$y[k,l] = x[k,l] + const.$$



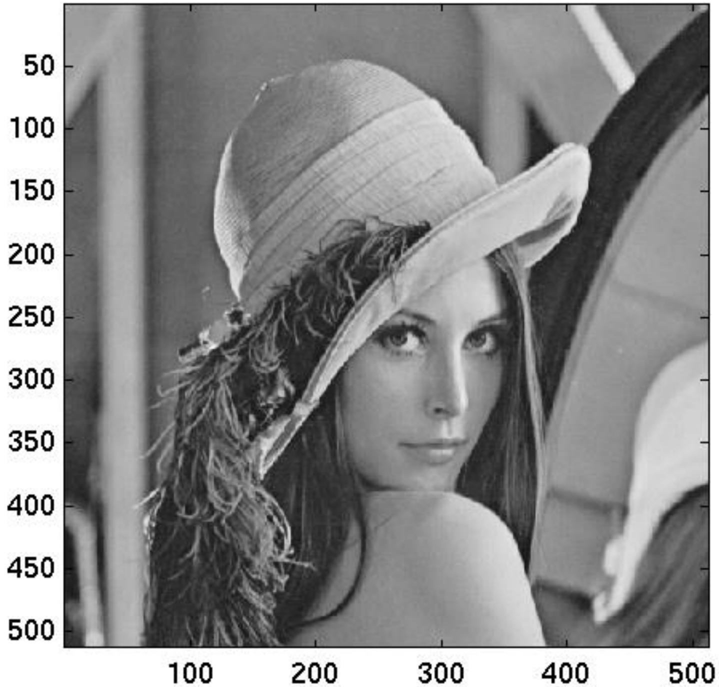
# Operations on pixels – less brightness

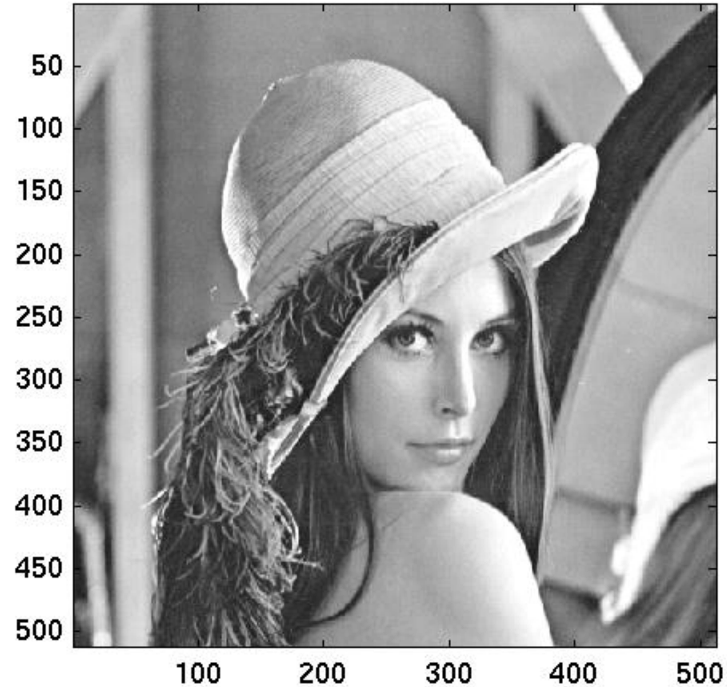
$$y[k,l] = x[k,l] + const.$$



# More contrast

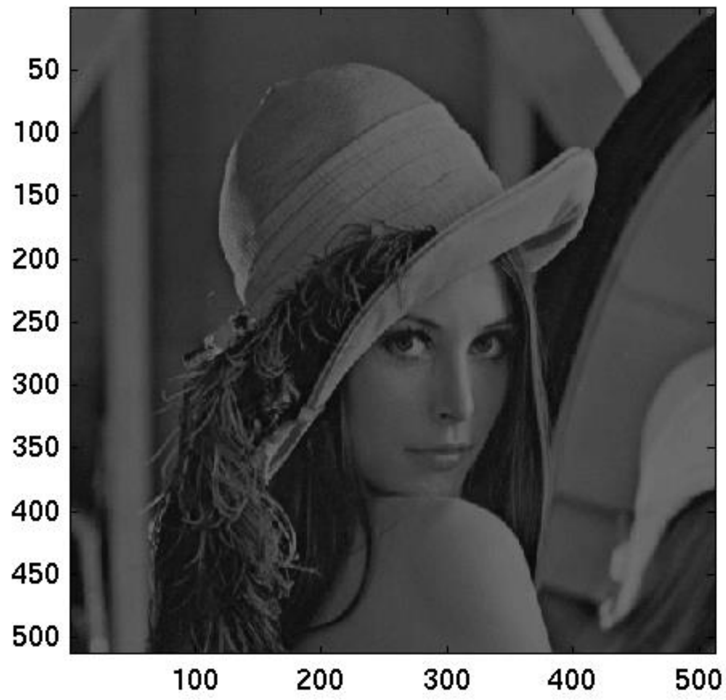
$$y[k,l] = x[k,l] \times const.$$



# Less contrast

$$y[k,l] = x[k,l] \times const.$$

# What about bad values ?

$x[k,l] < 0$ or $x[k,l] > 1$

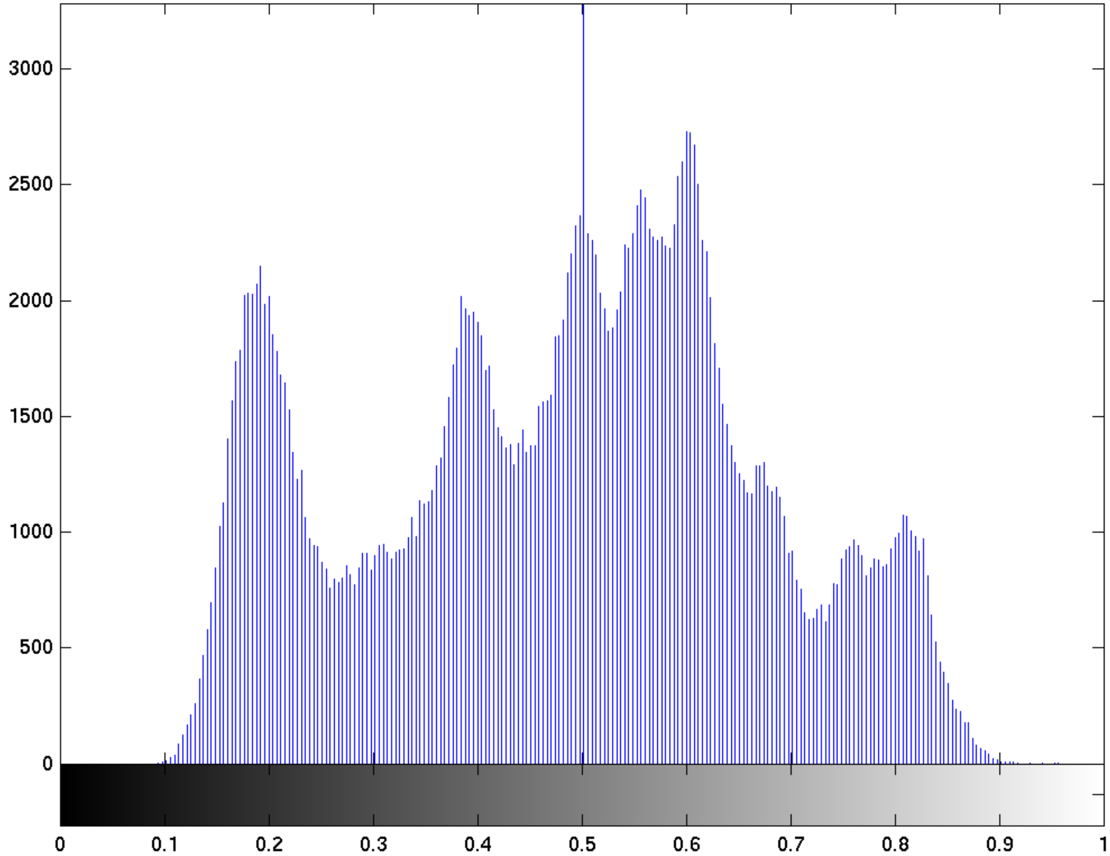
- Let them be …
- Clip them:

$$y[k,l] = 0, \quad \text{if} \;\; y[k,l] < 0$$

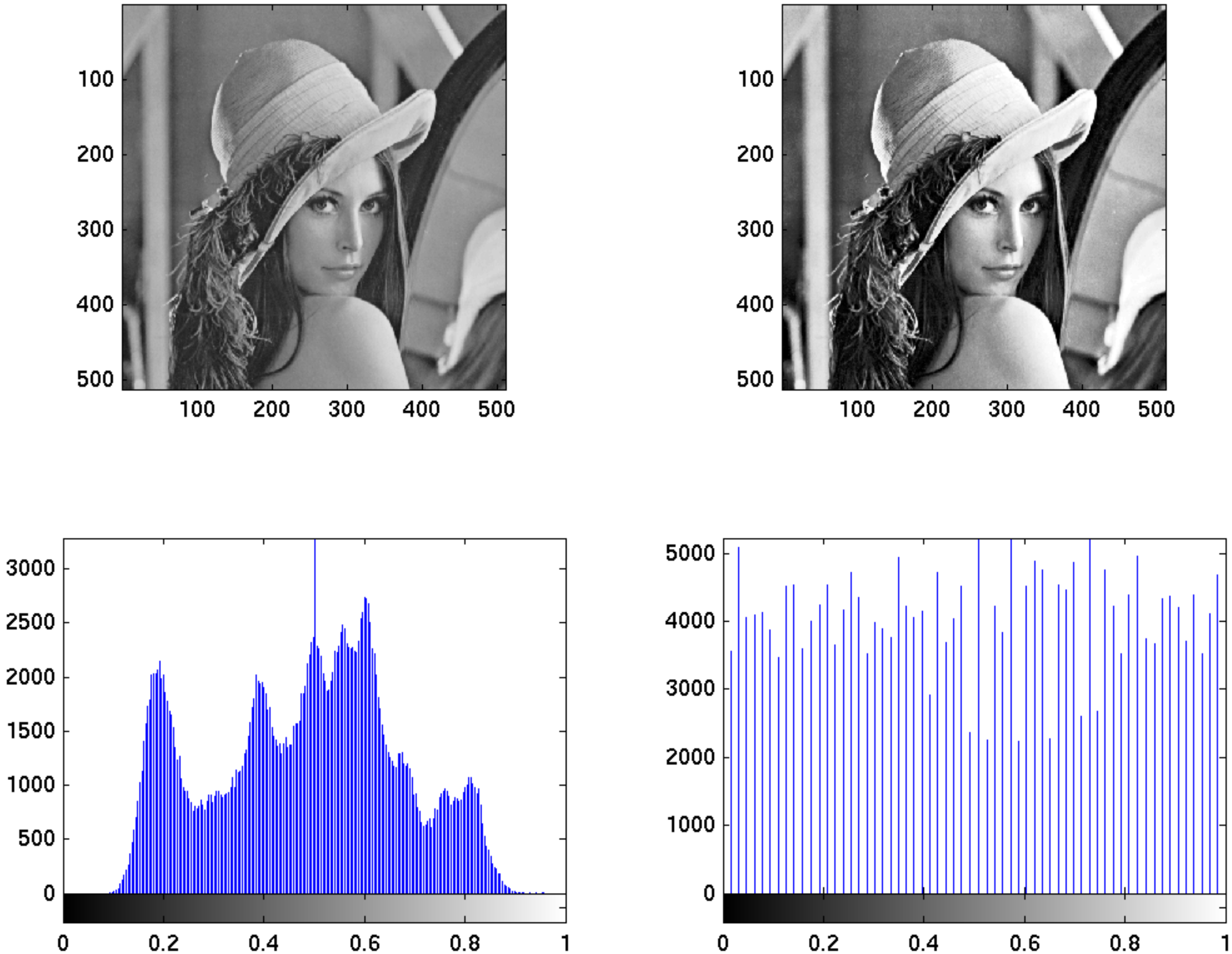
$$y[k,l] = 1, \quad \text{if} \;\; y[k,l] > 1$$

# Use statistics of values

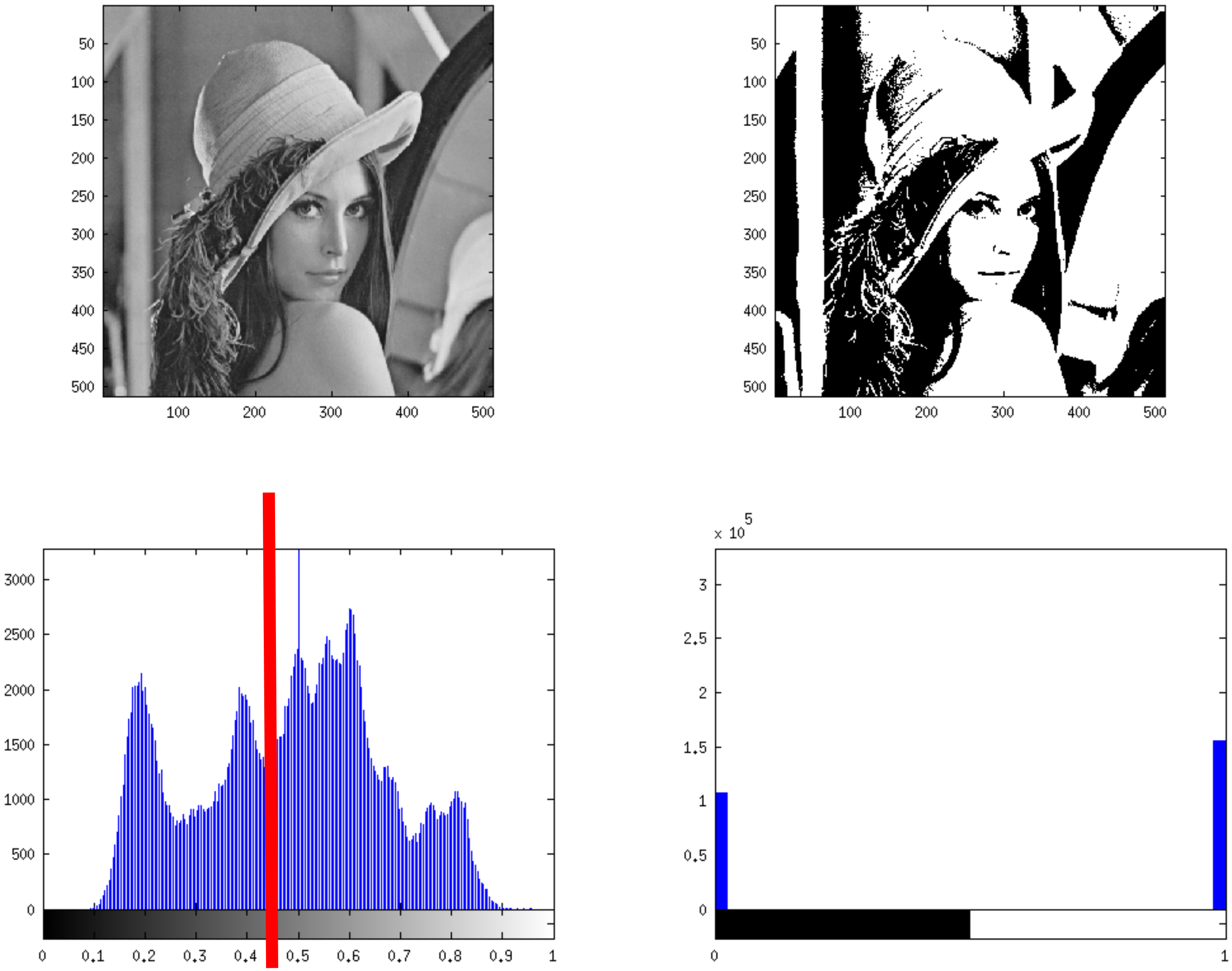
- Histogram



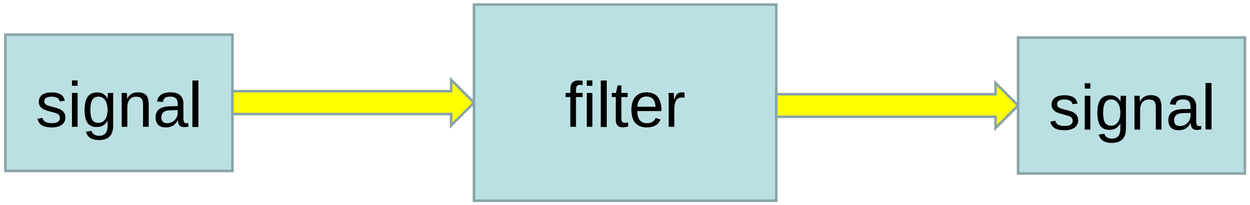
# Histogram equalization



# Thresholding



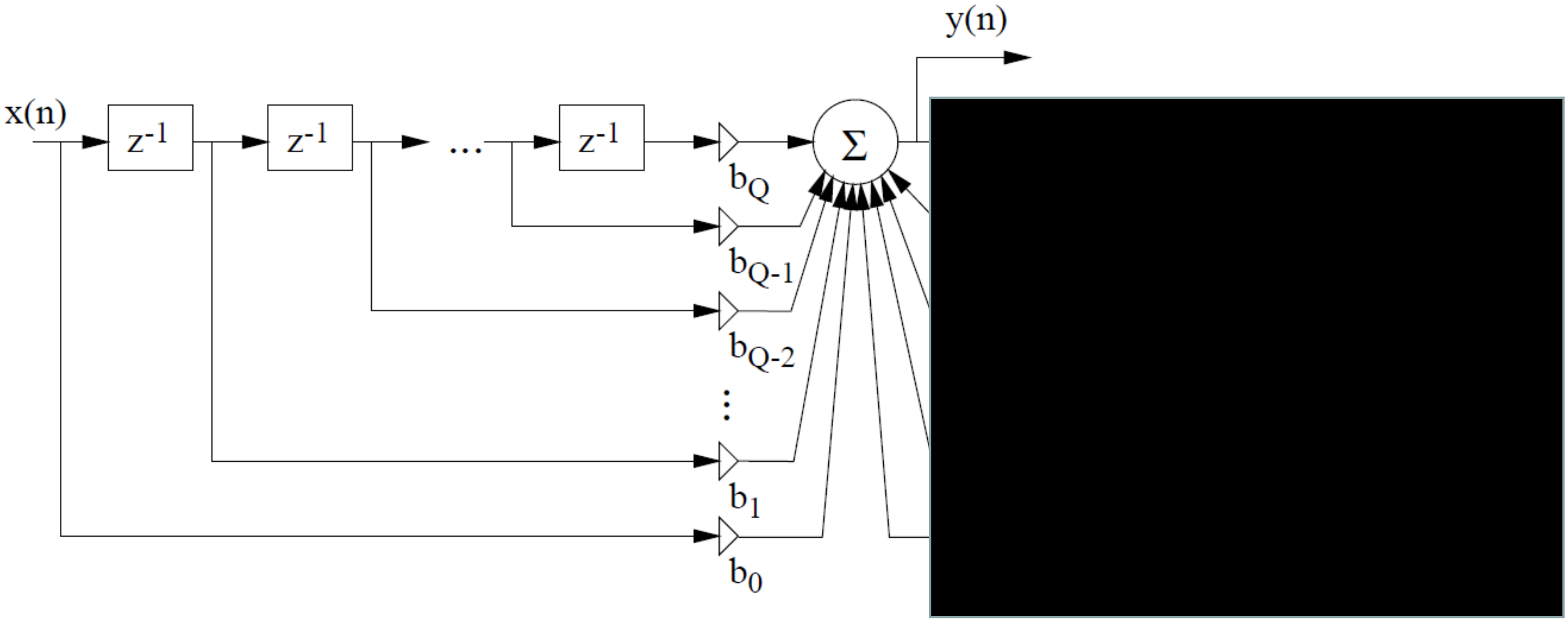
# Filtering




Task ?

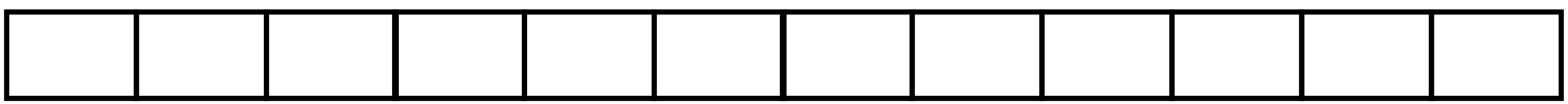
Obtain a **new signal** with desired properties.

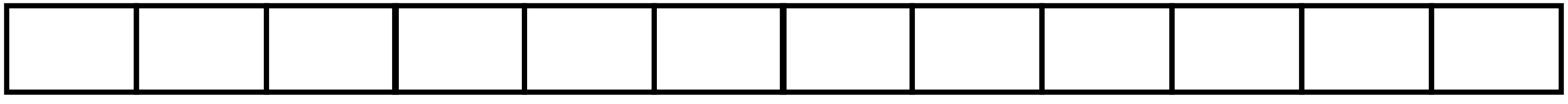
# Filtering – reminder of 1D



# Filtering – reminder of 1D

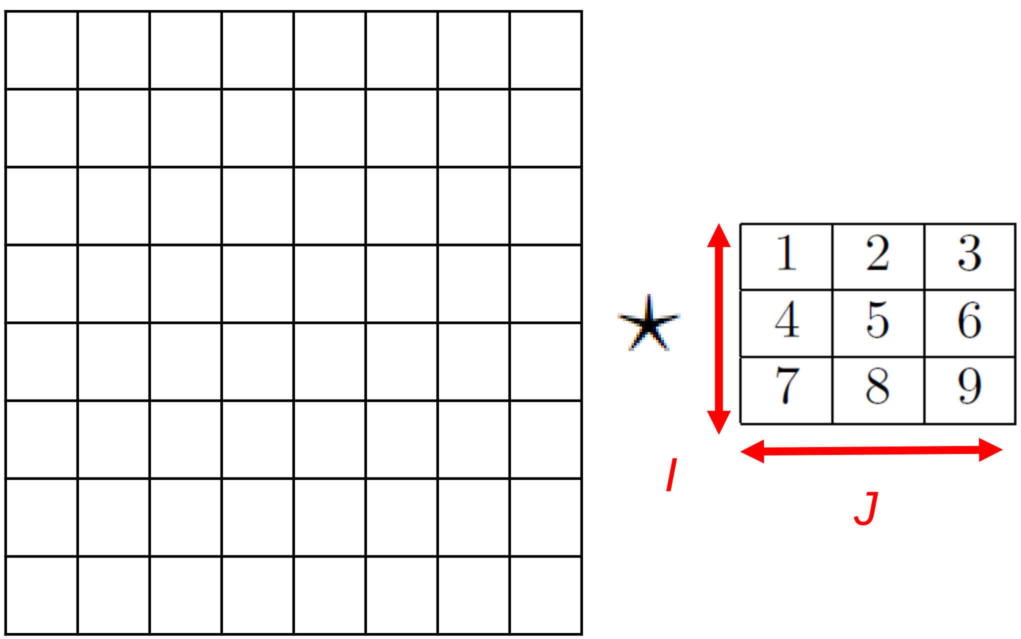
$$y[n] = x[n] \star h[n] = \sum_{k=0}^{Q} h[k]x[n-k]$$



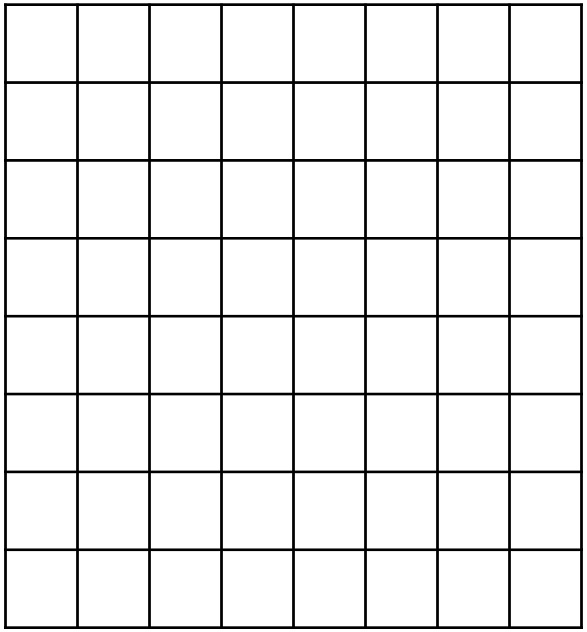


# 2D filter

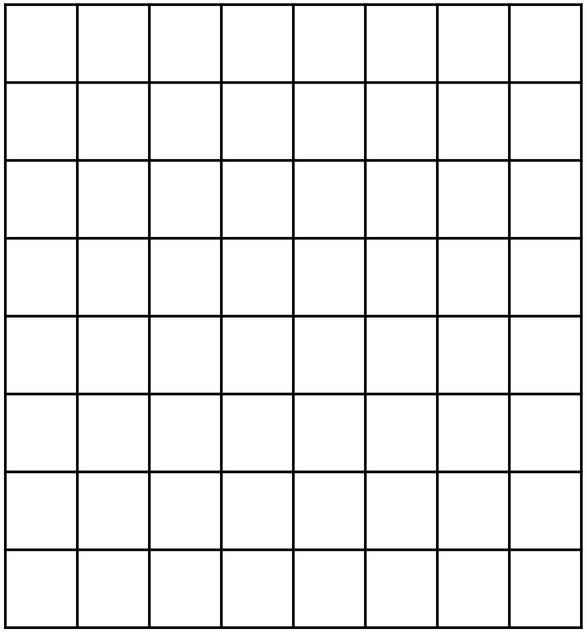
$$y[k,l] = x[k,l] \star h[k,l] = \sum_{m=-\frac{I-1}{2}}^{\frac{I-1}{2}} \sum_{n=-\frac{J-1}{2}}^{\frac{J-1}{2}} h[m,n]x[k-m,l-n]$$



# Filtering …



| | | |
|---|---|---|
| 9 | 8 | 7 |
| 6 | 5 | 4 |
| 3 | 2 | 1 |



Edge ? Insert zeros for example …

# Coefficients *h[k,l]*

**Terminology**

- Coefficients
- Mask
- Convolution kernel …

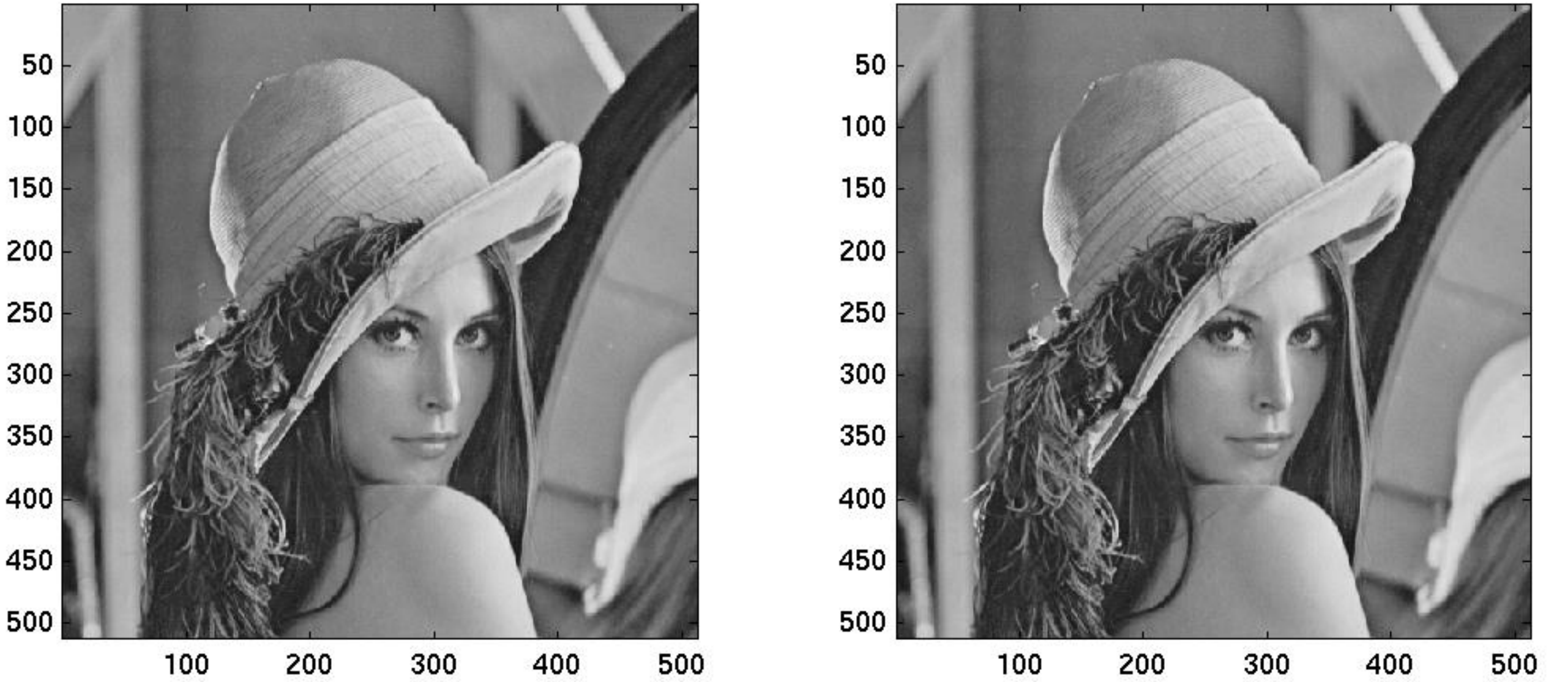
**We want them**

- to have some sense …
- not to change the dynamics of the signal

$$\sum_k \sum_l |h[k,l]| = 1$$

# Wire

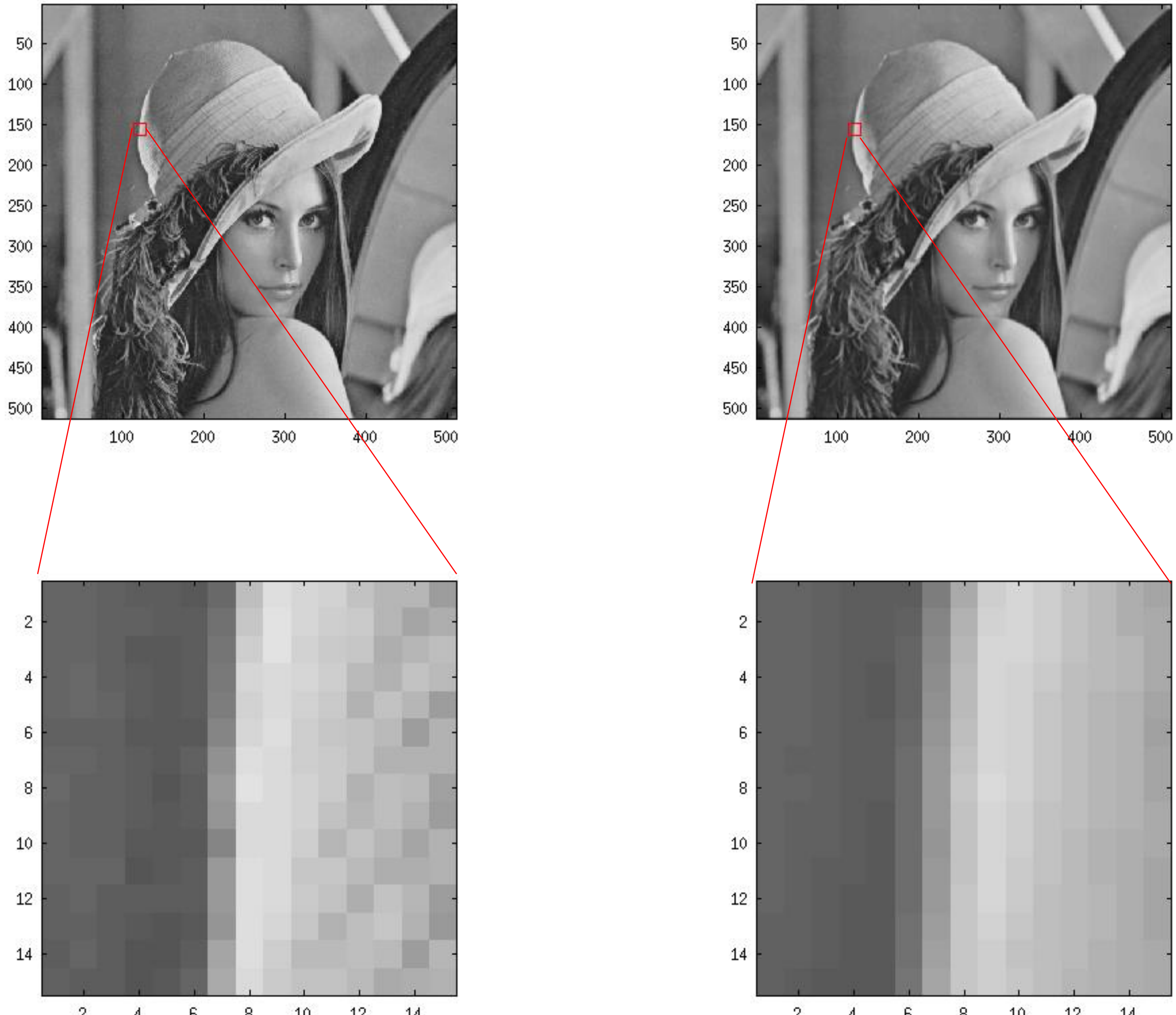
$$h[k,l] = [1]$$



50
100
150
200
250
300
350
400
450
500
100
200
300
400
500
50
100
150
200
250
300
350
400
450
500
100
200
300
400
500

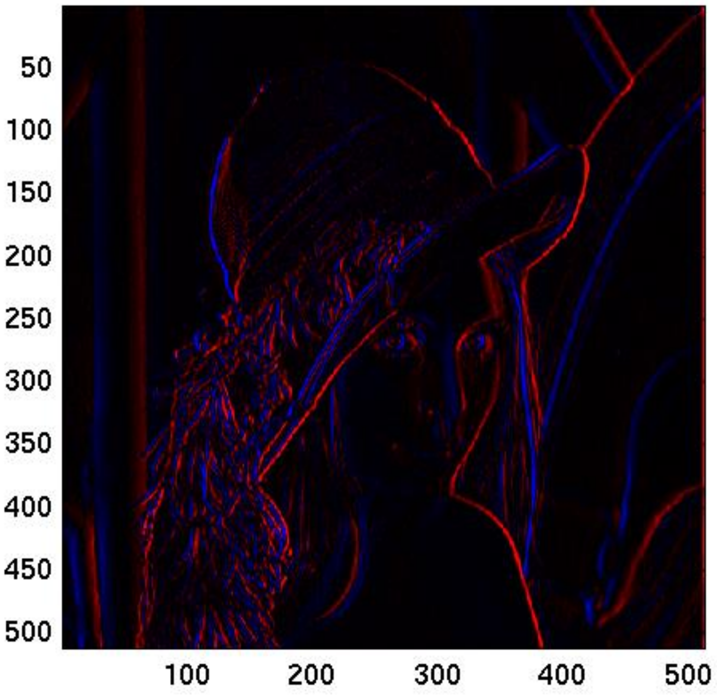
# Smoothing

$$h[k,l] = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



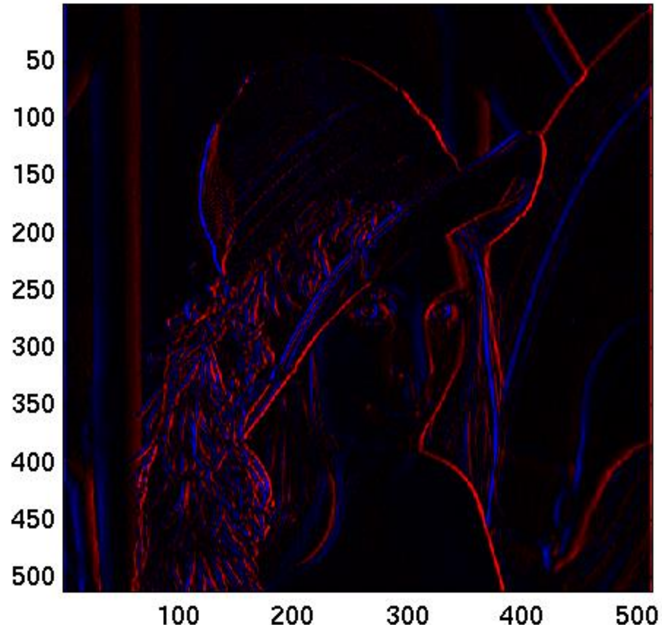
# Vertical edge detector

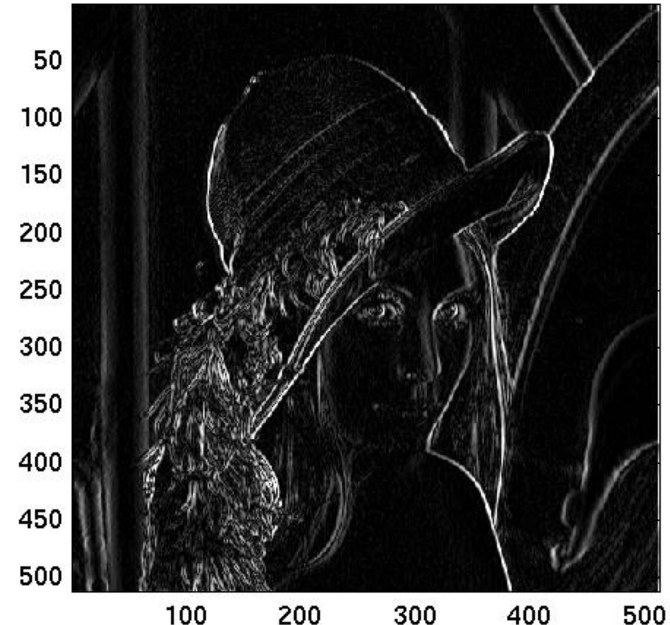
$$h_v[k,l] = \frac{1}{8}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

# What to do with negative samples?

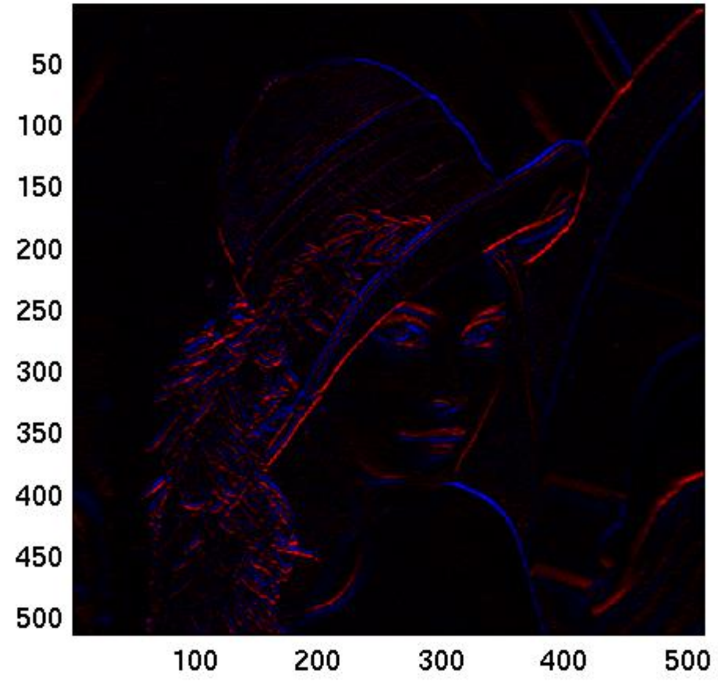
- Let them as they are for further computation…
- Or convert to [0…1] for visualization – for example take absolute value



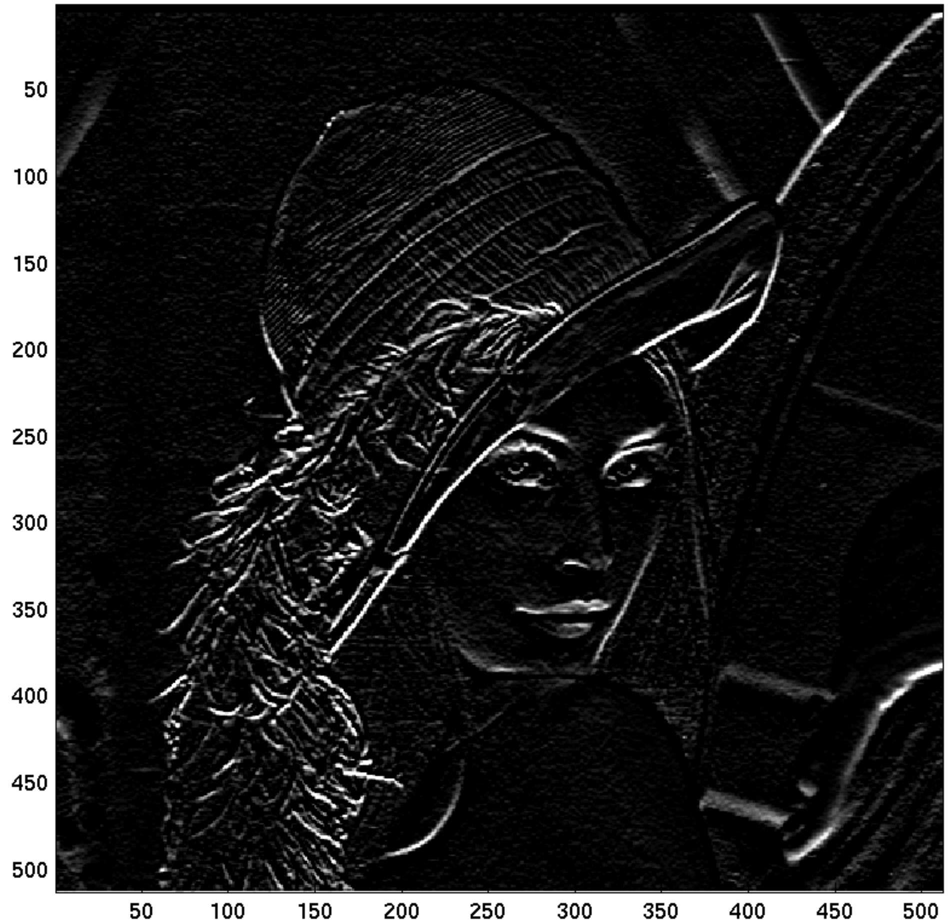
# Horizontal edge detector

$$h_h[k,l] = \frac{1}{8}\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

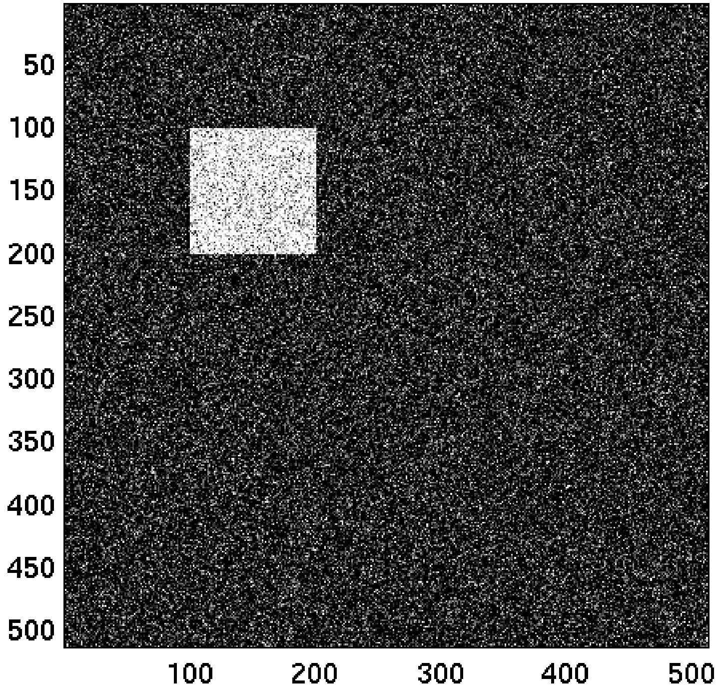
# Both together …

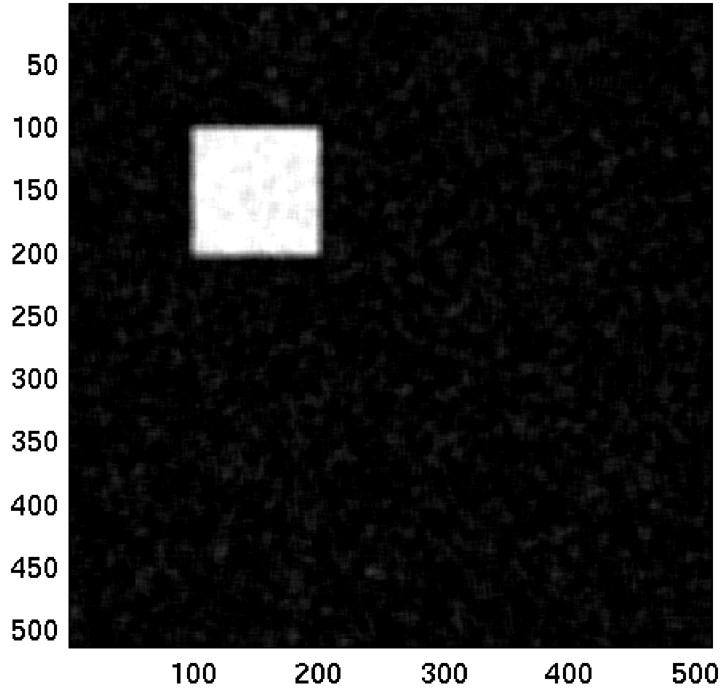
$$y[k,l] = |y_v[k,l]| + |y_h[k,l]|$$



# Denoising…

- Mask 9x9, values 1/81 … **zoom noise_low_pass.png**

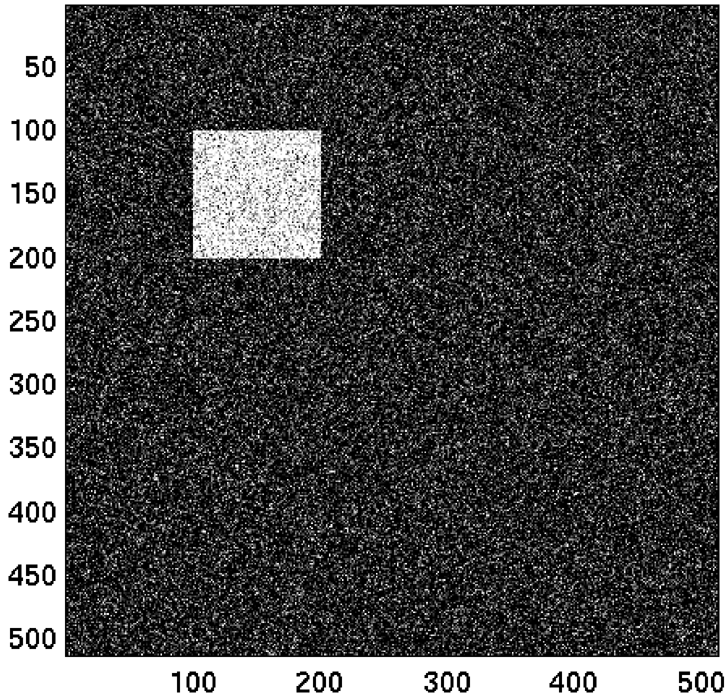


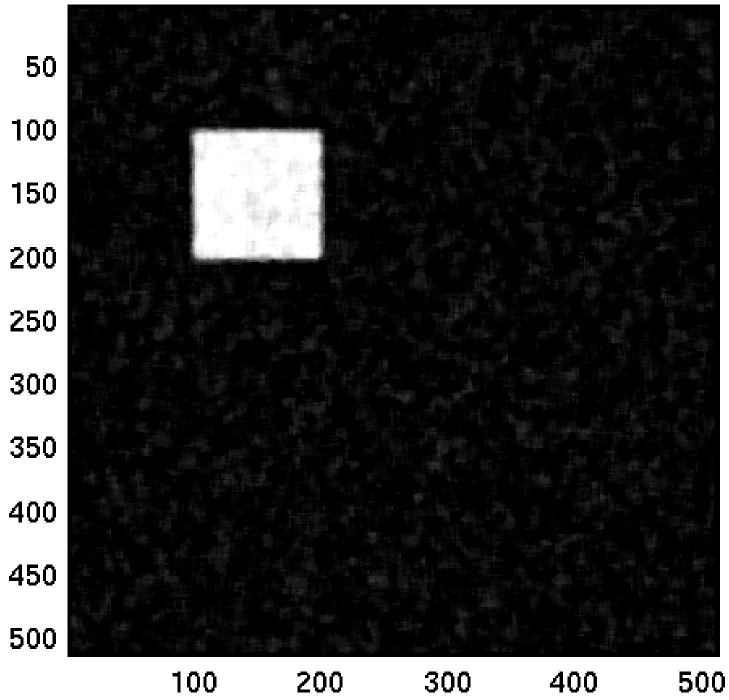



# Denoising II – median filter

$$y[k,l] = \text{median}_{k=-\frac{I-1}{2}\dots\frac{I-1}{2},\quad l=-\frac{J-1}{2}\dots\frac{J-1}{2}} x[k,l]$$

- **zoom noise_median.png**



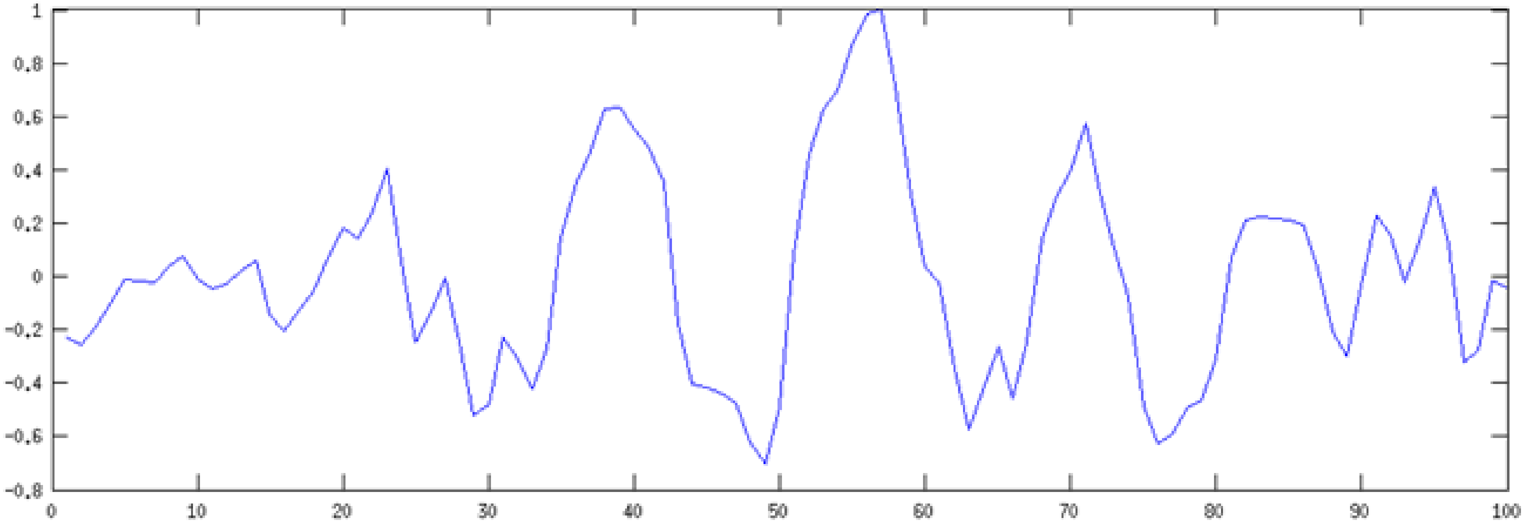
# Spectral analysis

- Task:

  **Determine, what is in the signal on different frequencies**

- Why ?
  - Visualization,
  - Feature extraction (think of Facebook)
  - Filtering (conversion to spectral domain and multiplication therein can be more efficient)
  - Coding (think of JPEG)

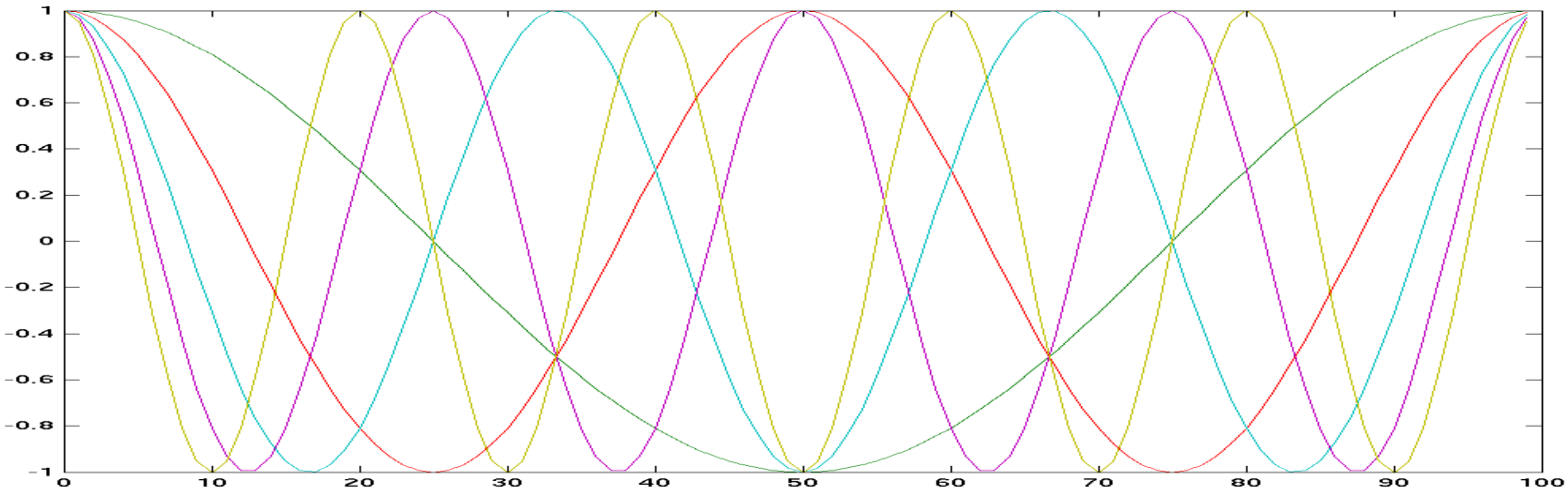
# Reminder – 1D




- What is in ?

$$c = \sum_{n=0}^{N-1} x[n]a[n]$$

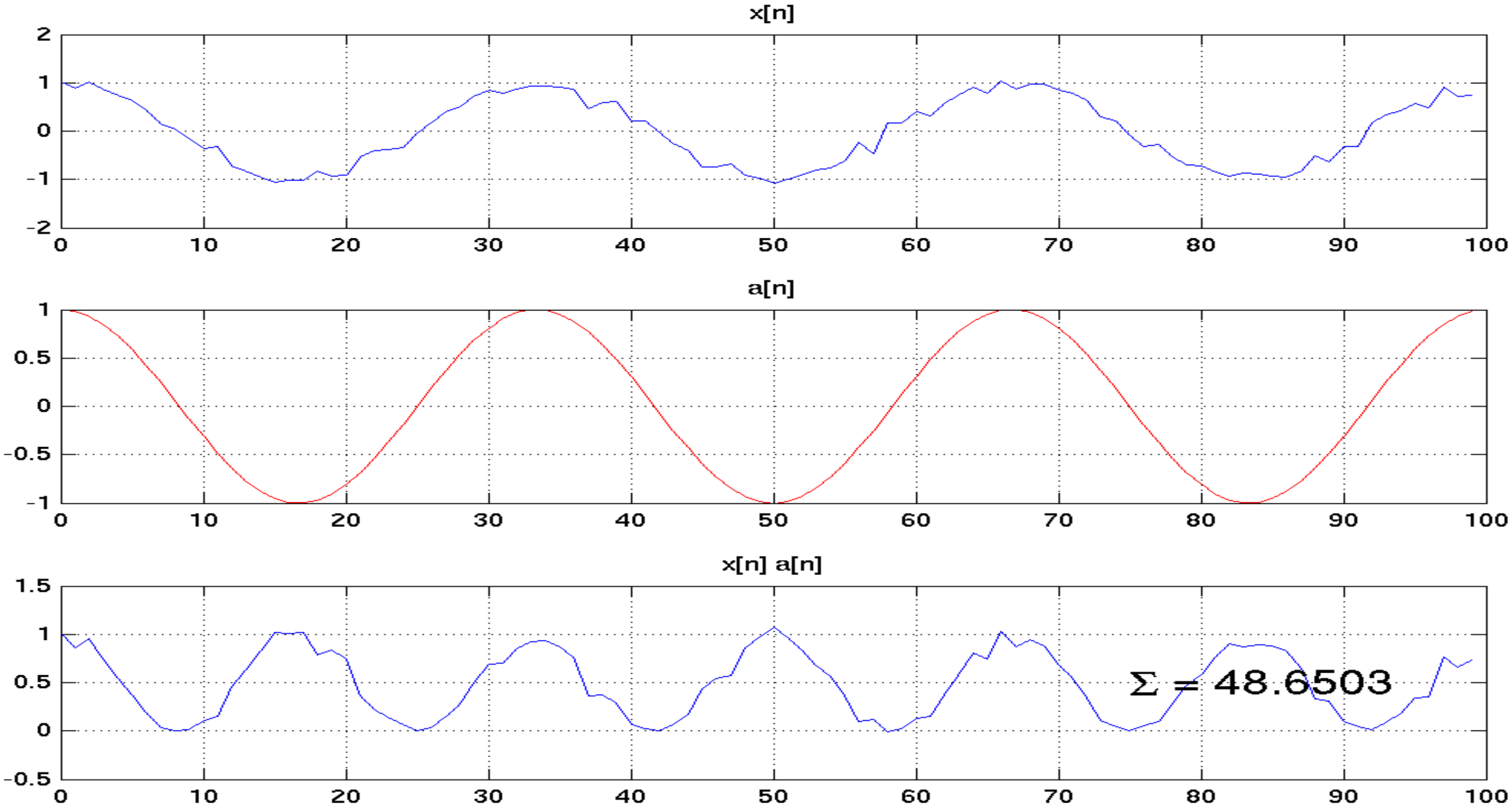
# Reminder – 1D cosine bases




$$\cos(2\pi \frac{k}{N} n)$$

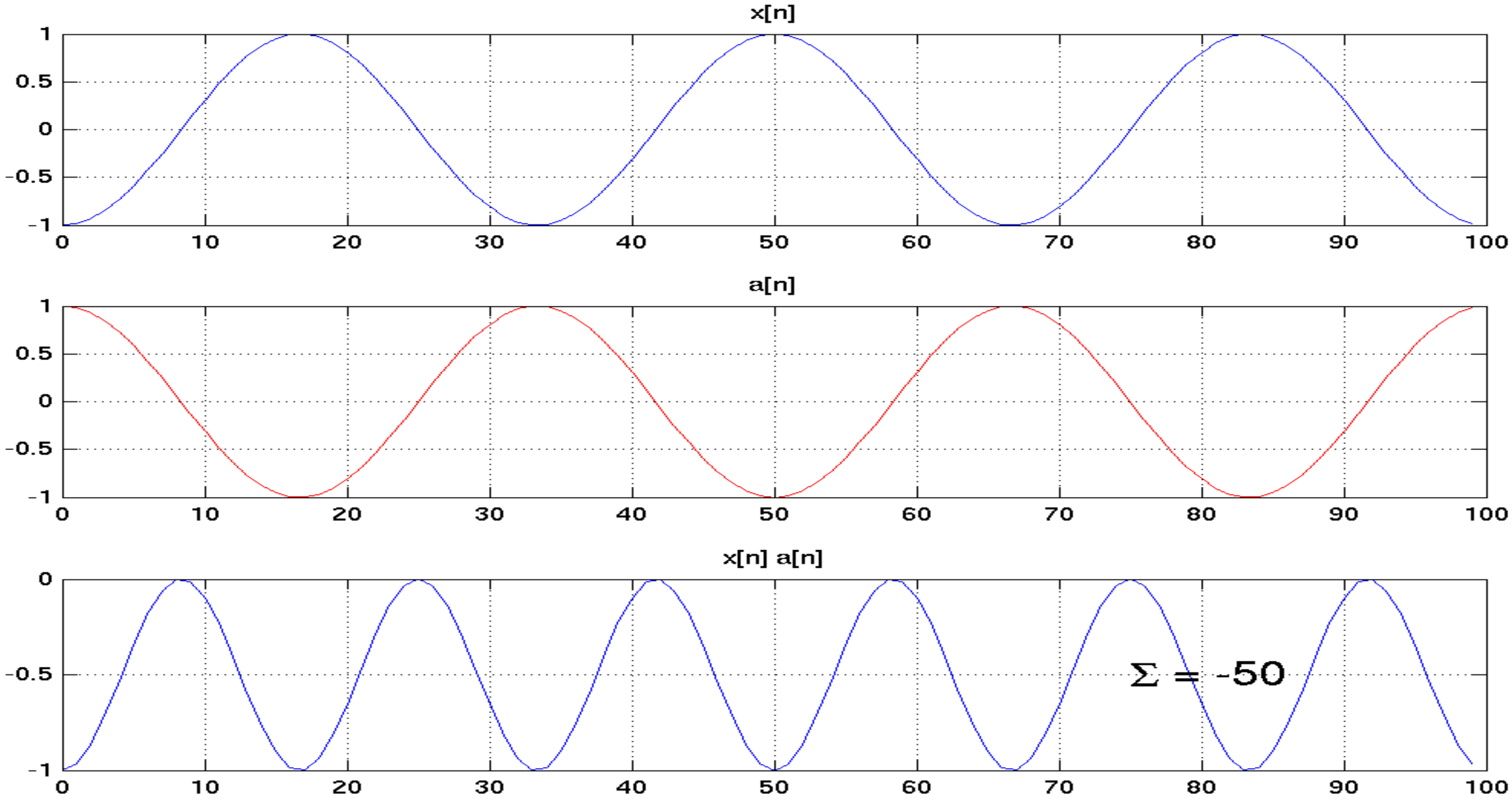
Problem with the phase – the signal “wave” begins not necessarily at zero …

# OK ...

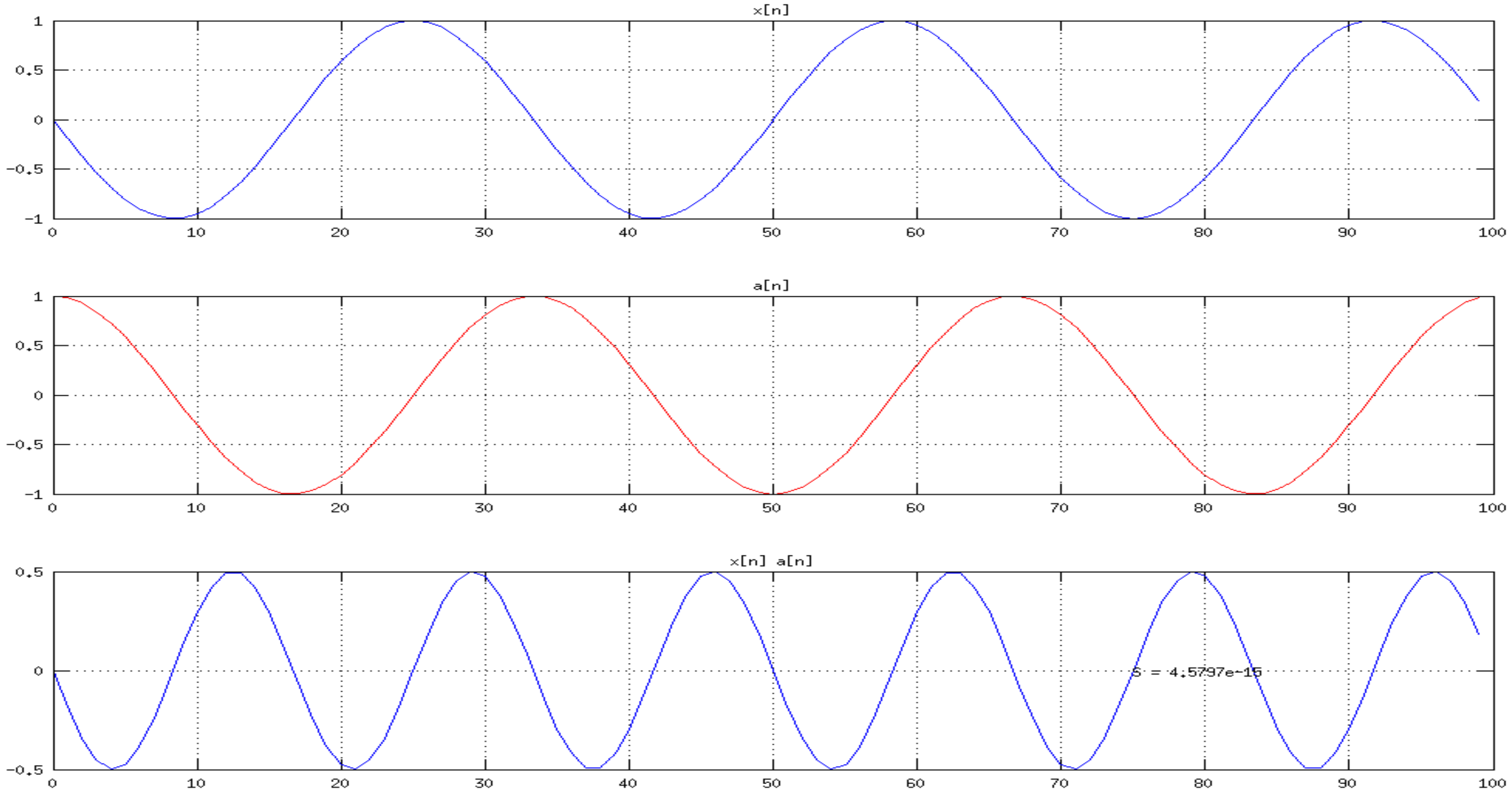


x[n]
2
1
0
-1
-2
0
10
20
30
40
50
60
70
80
90
100
a[n]
1
0.5
0
-0.5
-1
0
10
20
30
40
50
60
70
80
90
100
x[n] a[n]
1.5
1
0.5
0
-0.5
0
10
20
30
40
50
60
70
80
90
100
Σ = 48.6503

# Also OK …

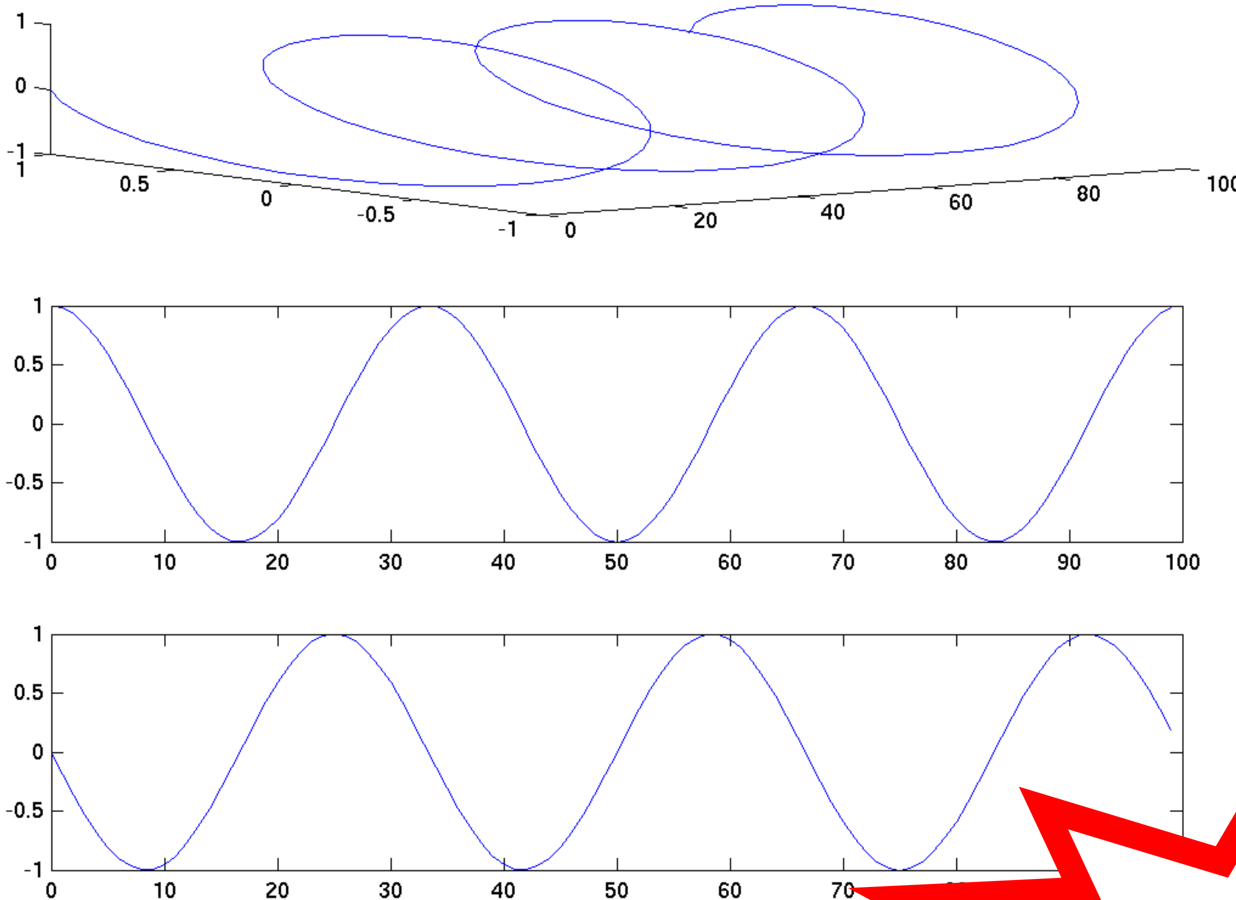


# Bad bad bad ☹



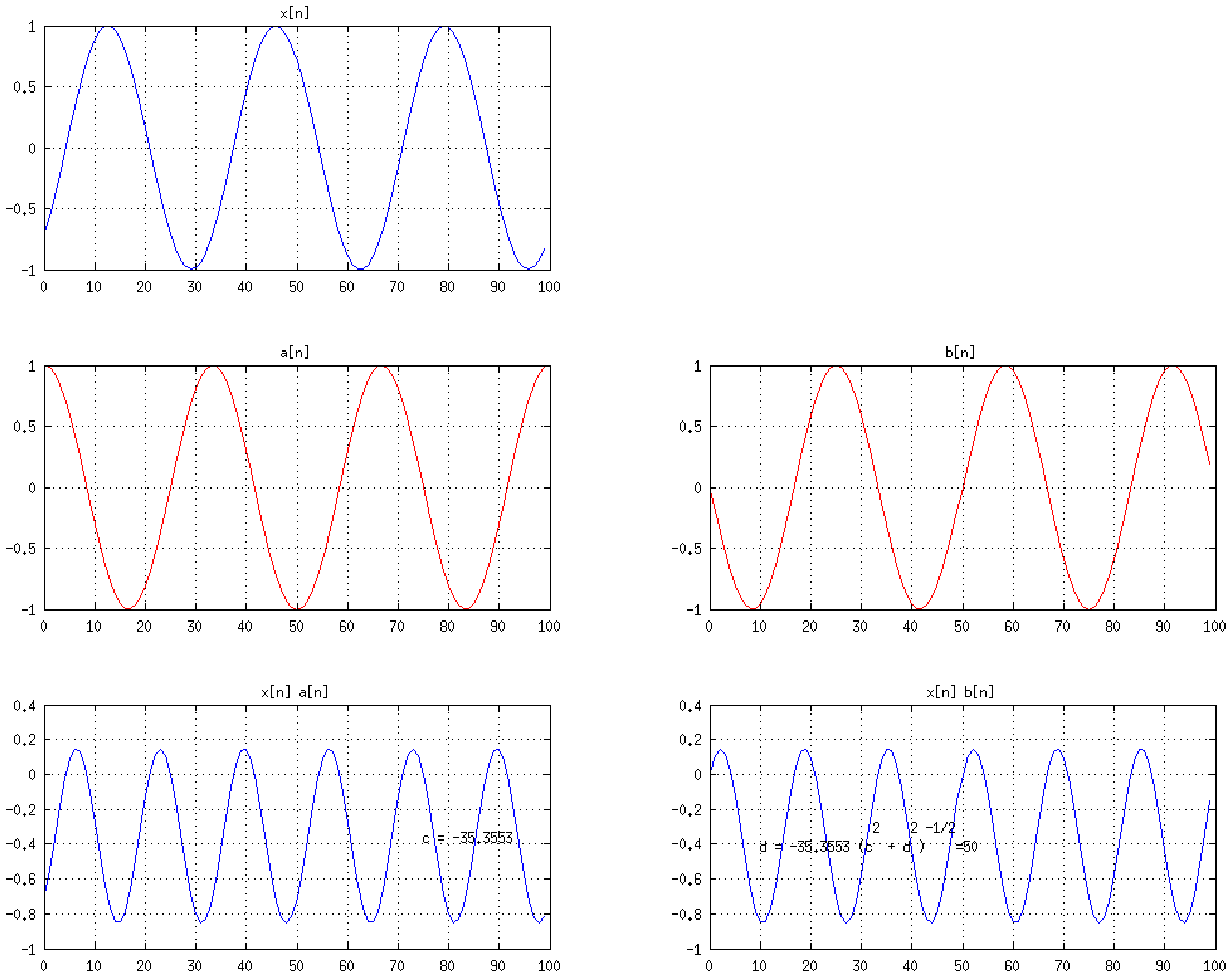

- How come zero when $\sin(x) = \cos(x - \frac{\pi}{2})$ ?

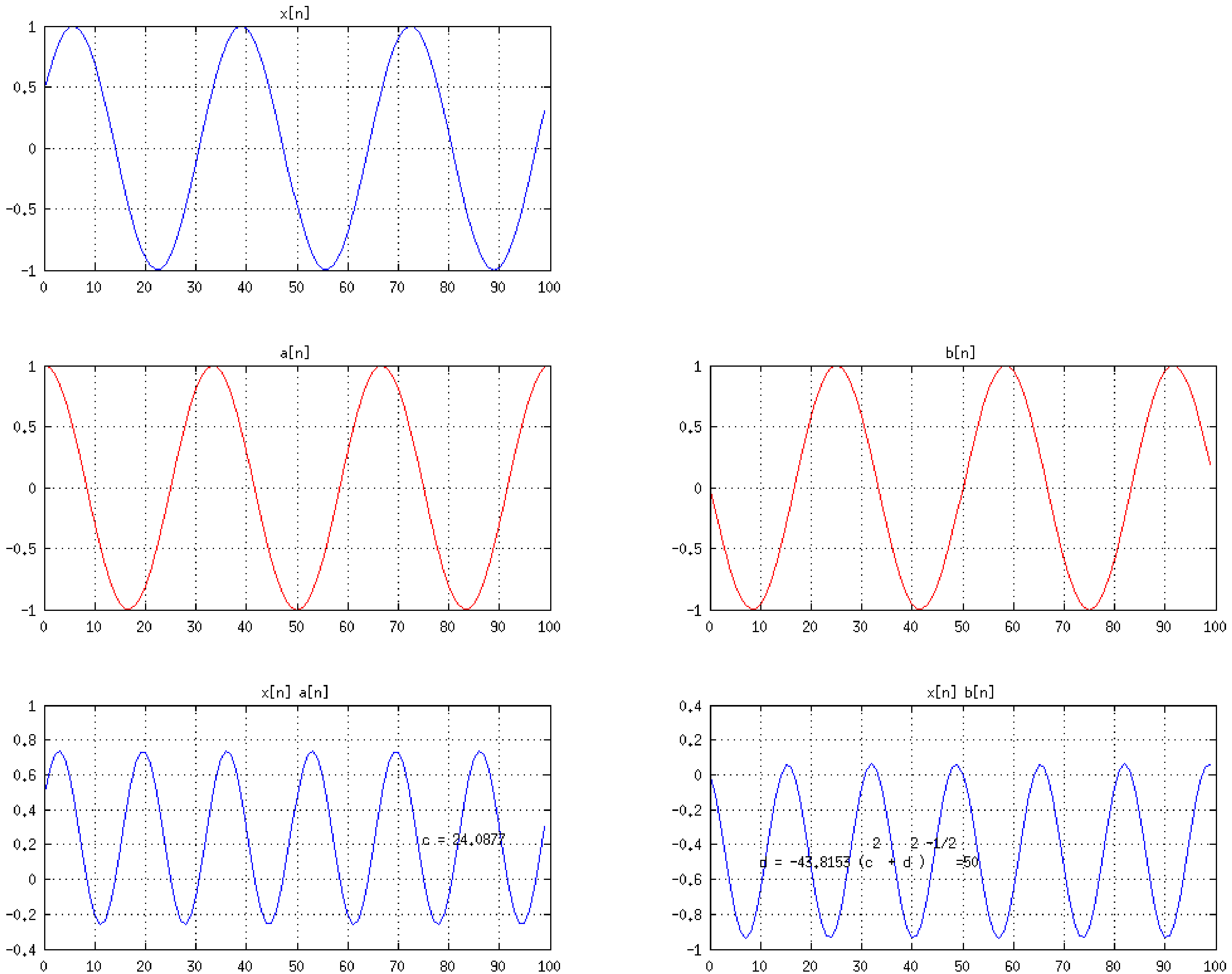
# Complex exponentials




$$e^{jx} = \cos(x) + j\sin(x)$$

$$e^{-j2\pi\frac{k}{N}n} = \cos(2\pi\frac{k}{N}n) - j\sin(2\pi\frac{k}{N}n)$$


x[n]
a[n]
b[n]
x[n] a[n]
x[n] b[n]
c = -35.3553
d = -35.3553 (c^2 + d^2)^-1/2 =50


x[n]
a[n]
b[n]
x[n] a[n]
c = 24.0877
x[n] b[n]
d = -43.8153 (c^2 + d^2)^-1/2 = 50

# 1D ultimate result - DFT

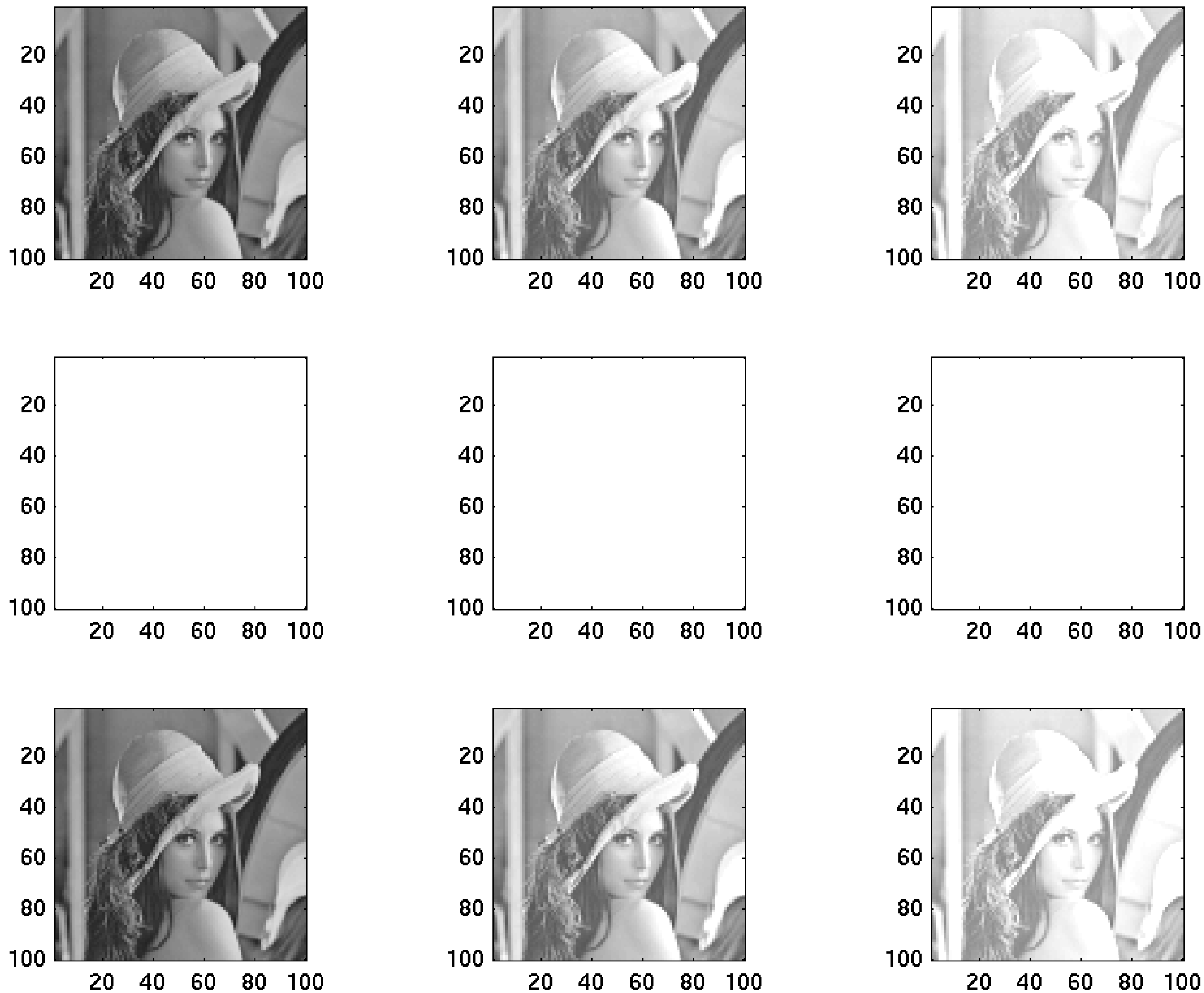
$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi \frac{k}{N} n}, \quad k = 0 \ldots N-1$$

# Now finally 2D

- Correlation = determination of similarity = projection to bases
- **We have seen this, it's boring, it's all the time the same … YES IT IS !**

$$c = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x[k,l] a[k,l]$$

# Analyzing signal = d.c. $a[k,l] = 1$

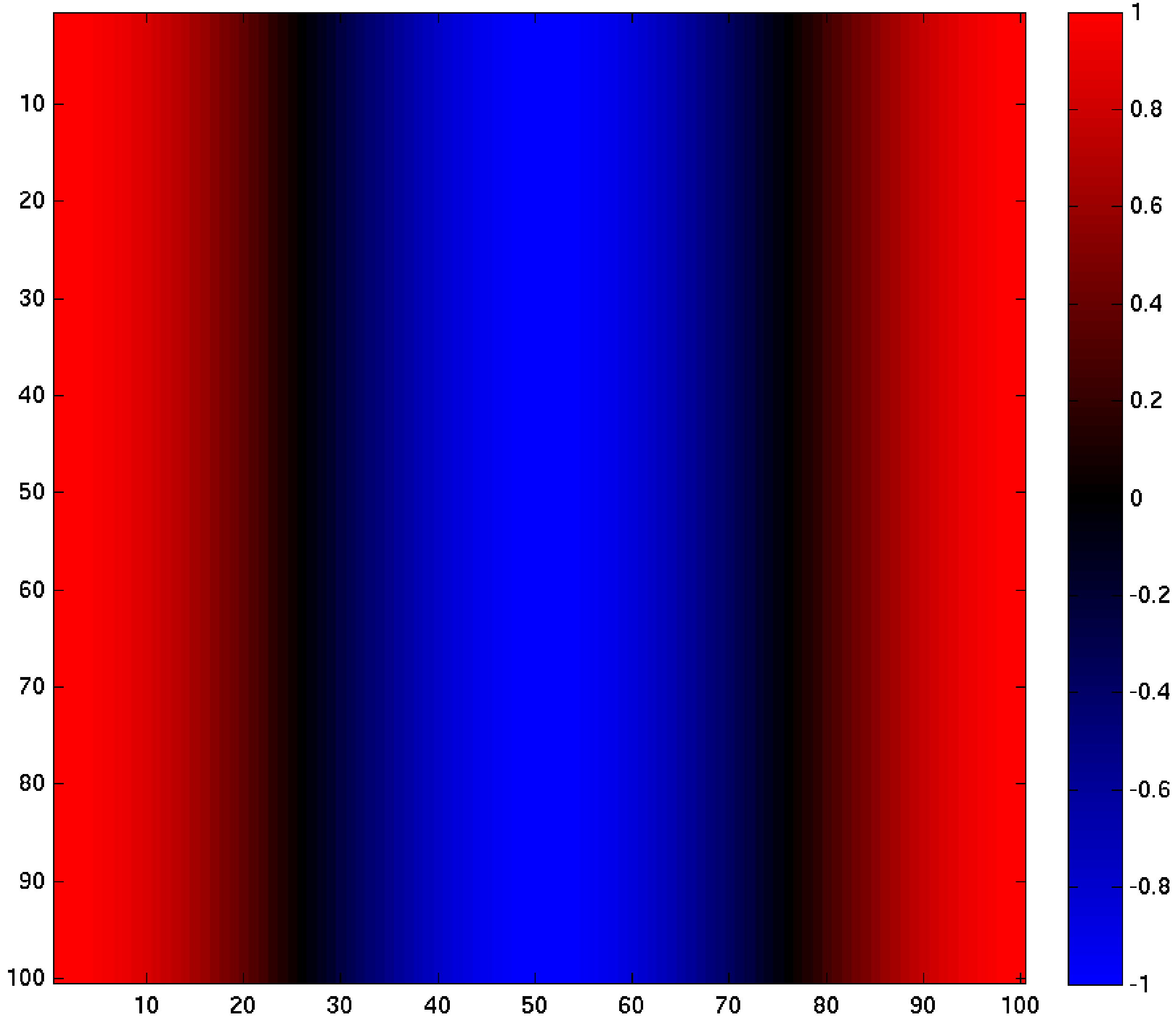



c=4.8287e+03 c=6.8287e+03 c=8.8287e+03

# Analyzzing signal – horizontal cosine

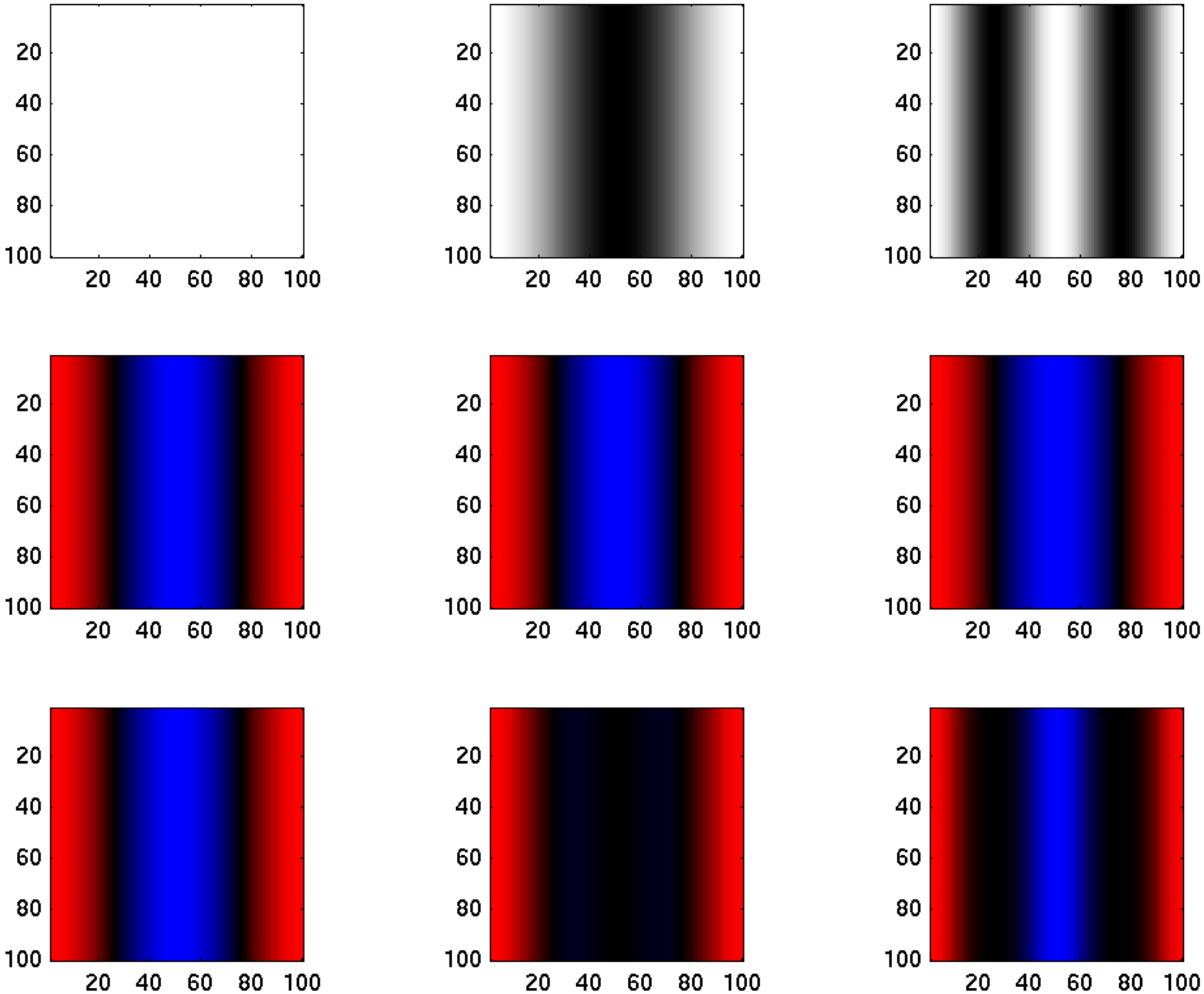
$$a[k, l] = \cos(2\pi \frac{1}{100} l)$$

- Changes only in one direction
- We’ll need to visualize negative values too.




Geeks: how to do this in Matlab ??

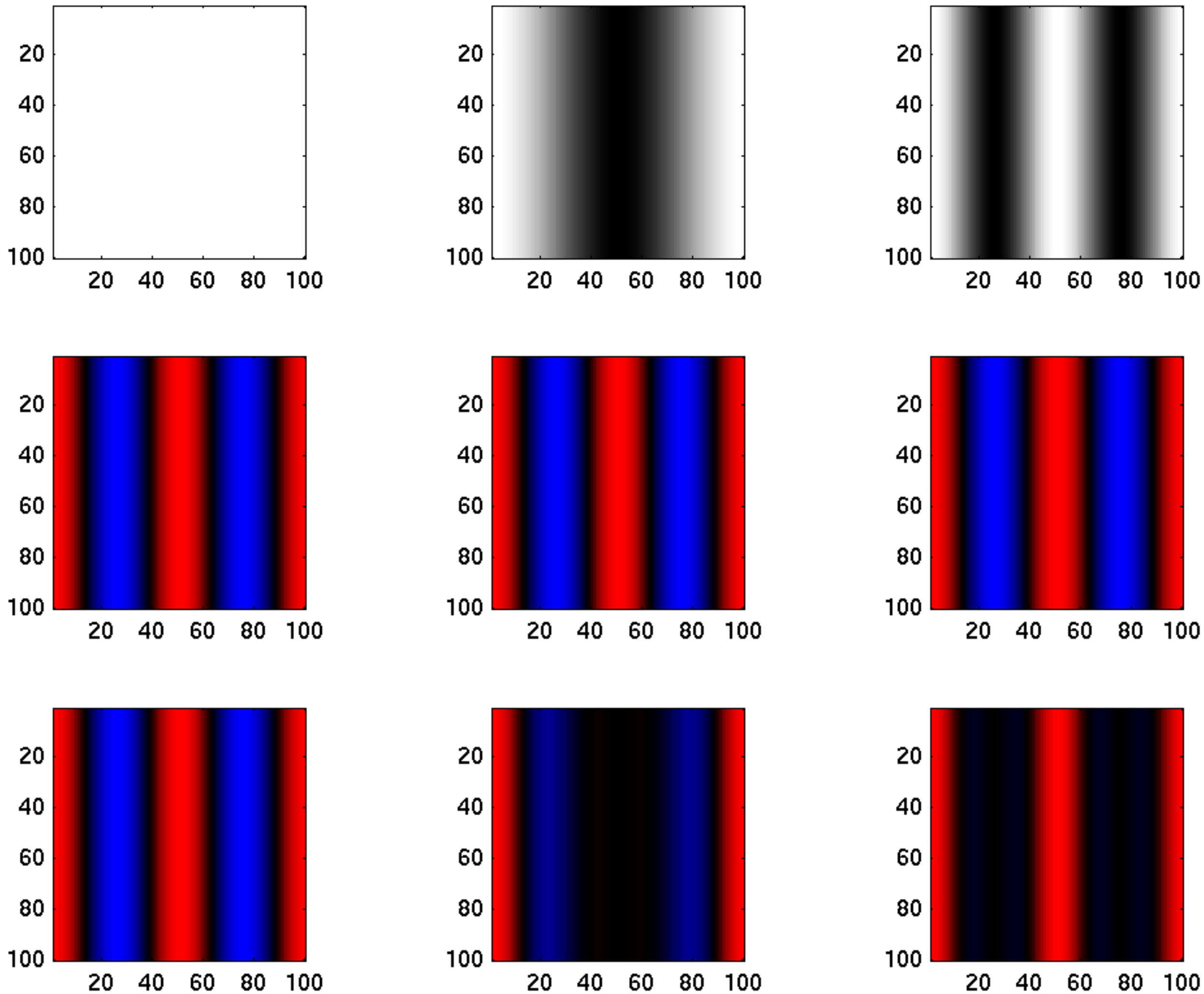
# Analysis for $a[k,l] = \cos(2\pi \frac{1}{100} l)$




c=4.5475e-13 c=2500 c=3.4106e-13

# 2 x faster horizontal cos

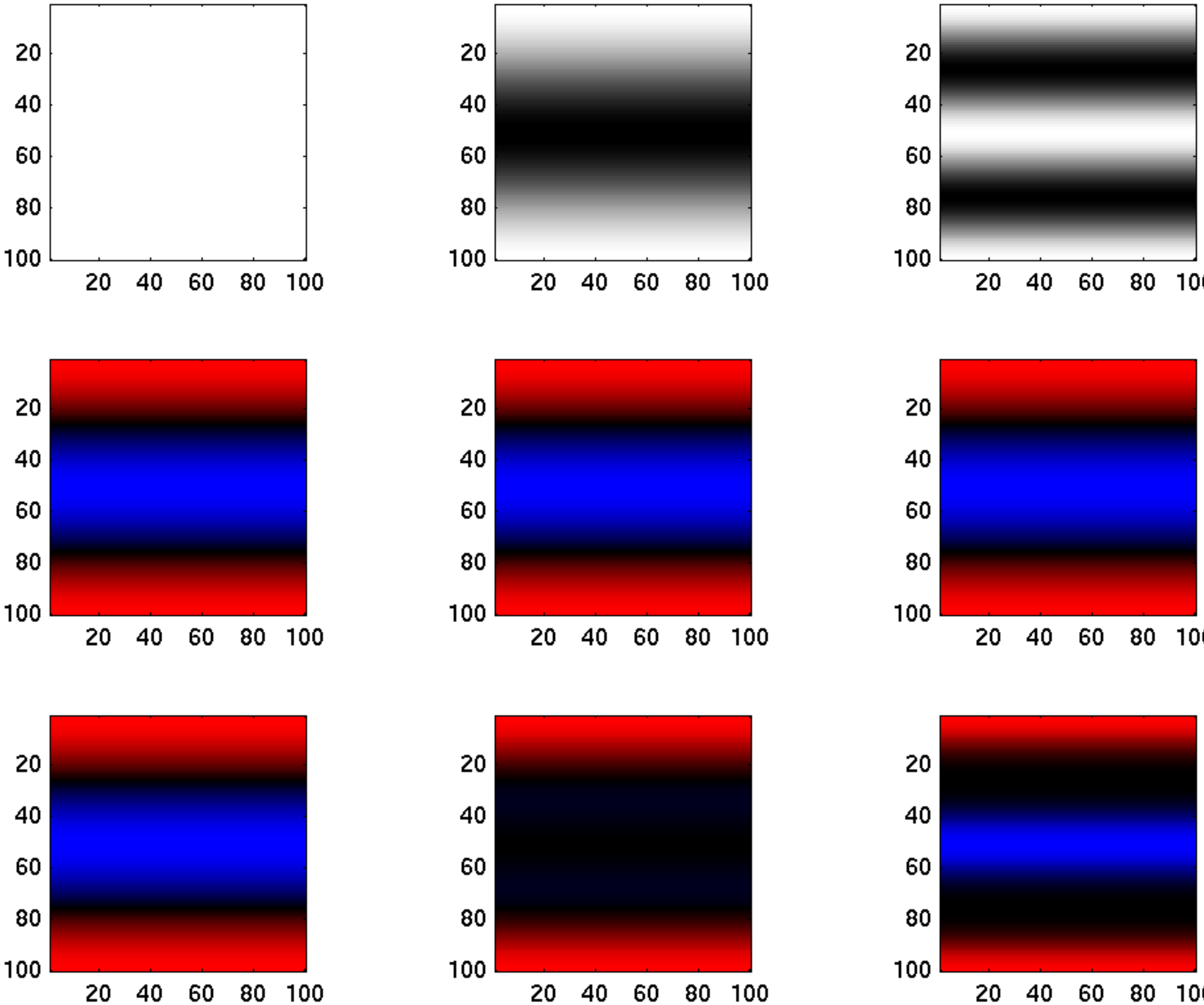
$$a[k,l] = \cos(2\pi \frac{2}{100} l)$$




c=2.9843e-13 c=1.5064e-12 c=2.5000e+03

# Vertical cos

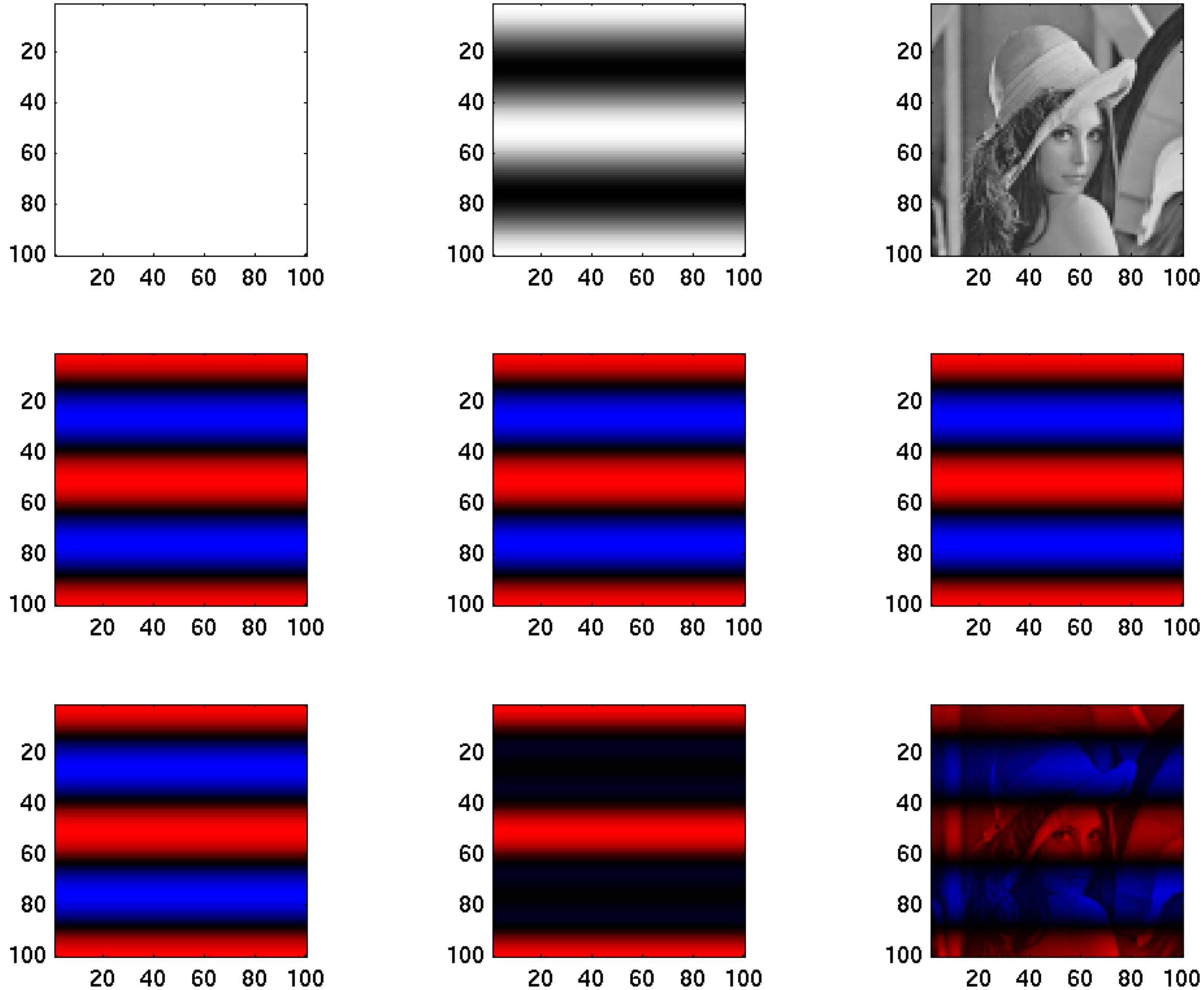
$$a[k,l] = \cos(2\pi \frac{1}{100} k)$$




c= -7.7716e-14 c=2.5000e+03 c=6.9944e-13

# 2 x faster vertical cos

$$a[k,l] = \cos(2\pi \frac{2}{100} k)$$

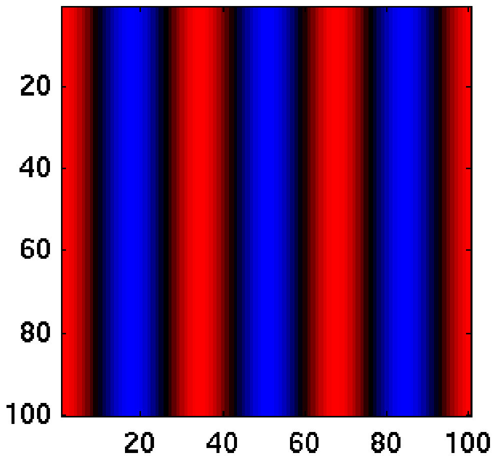


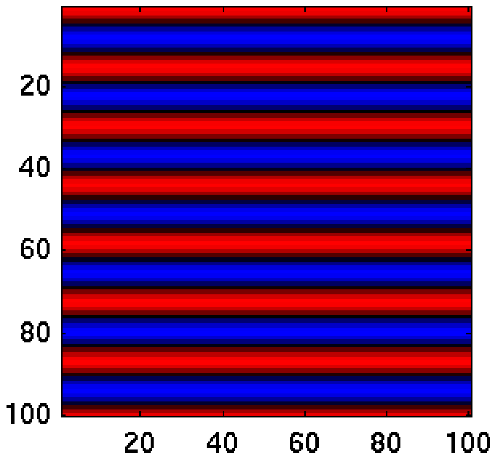

c= 4.9960e-13 c=2.5000e+03 c=-65.4694

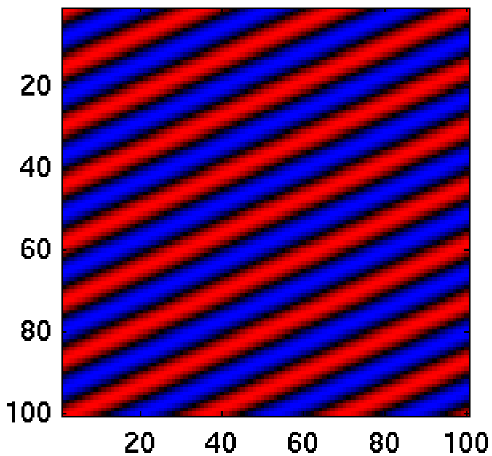
# Mix of both directions …

$$a[k,l] = \cos(2\pi \frac{3}{100} l) \qquad a[k,l] = \cos(2\pi \frac{7}{100} k)$$

$$a[k,l] = \cos\left[2\pi(\frac{7}{100}k + \frac{3}{100}l)\right]$$



# Generalization

$$X[m,n] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x[k,l] \cos\left[2\pi(\frac{m}{K}k + \frac{n}{L}l)\right]$$

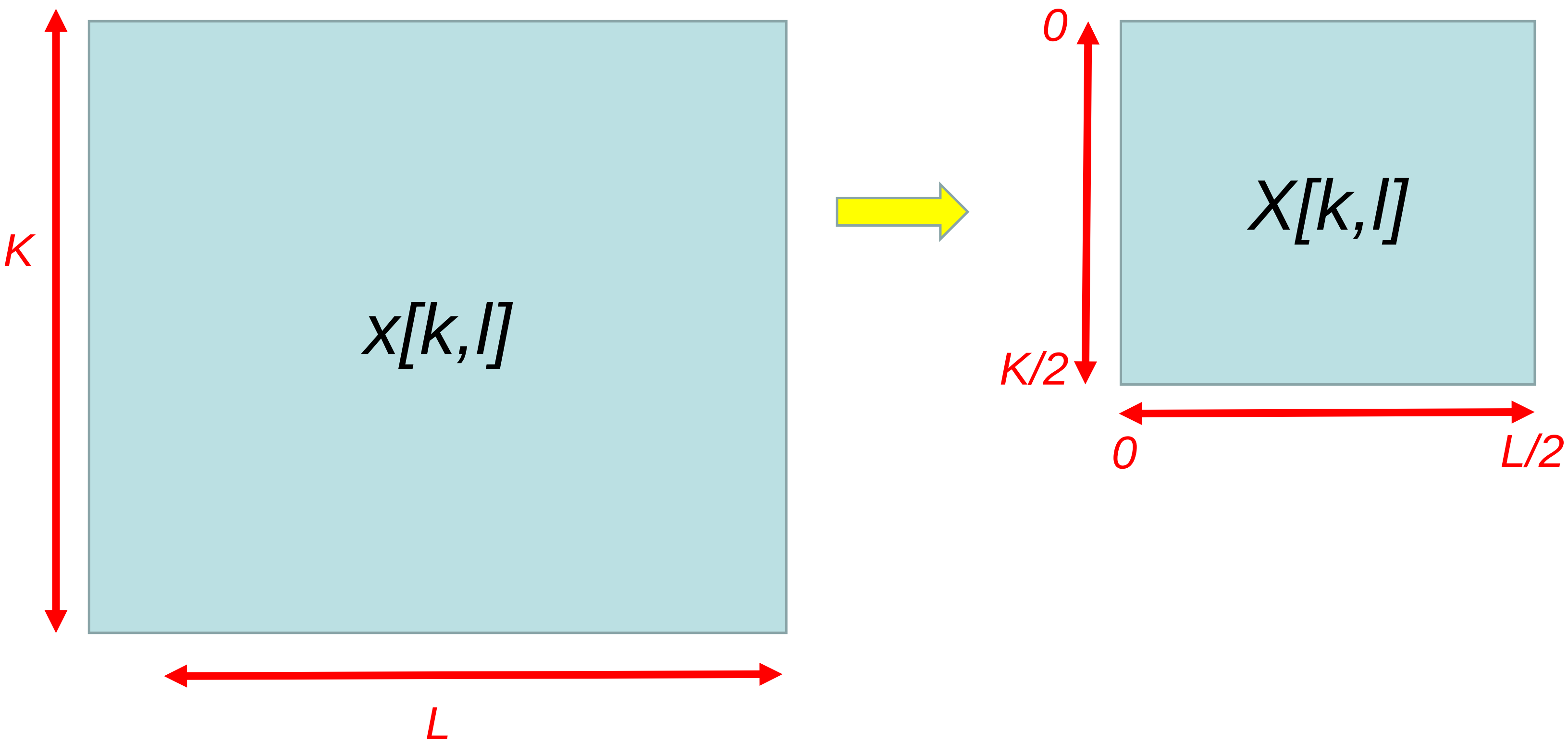
- *m/K* – horizontal frequency
- *n/L* – vertical frequency

# Reasonable ranges of frequencies ?

- *m/K* and *n/L*

  0 … ½ => OK

  > ½ => hm…



# More on frequencies …

- 1D

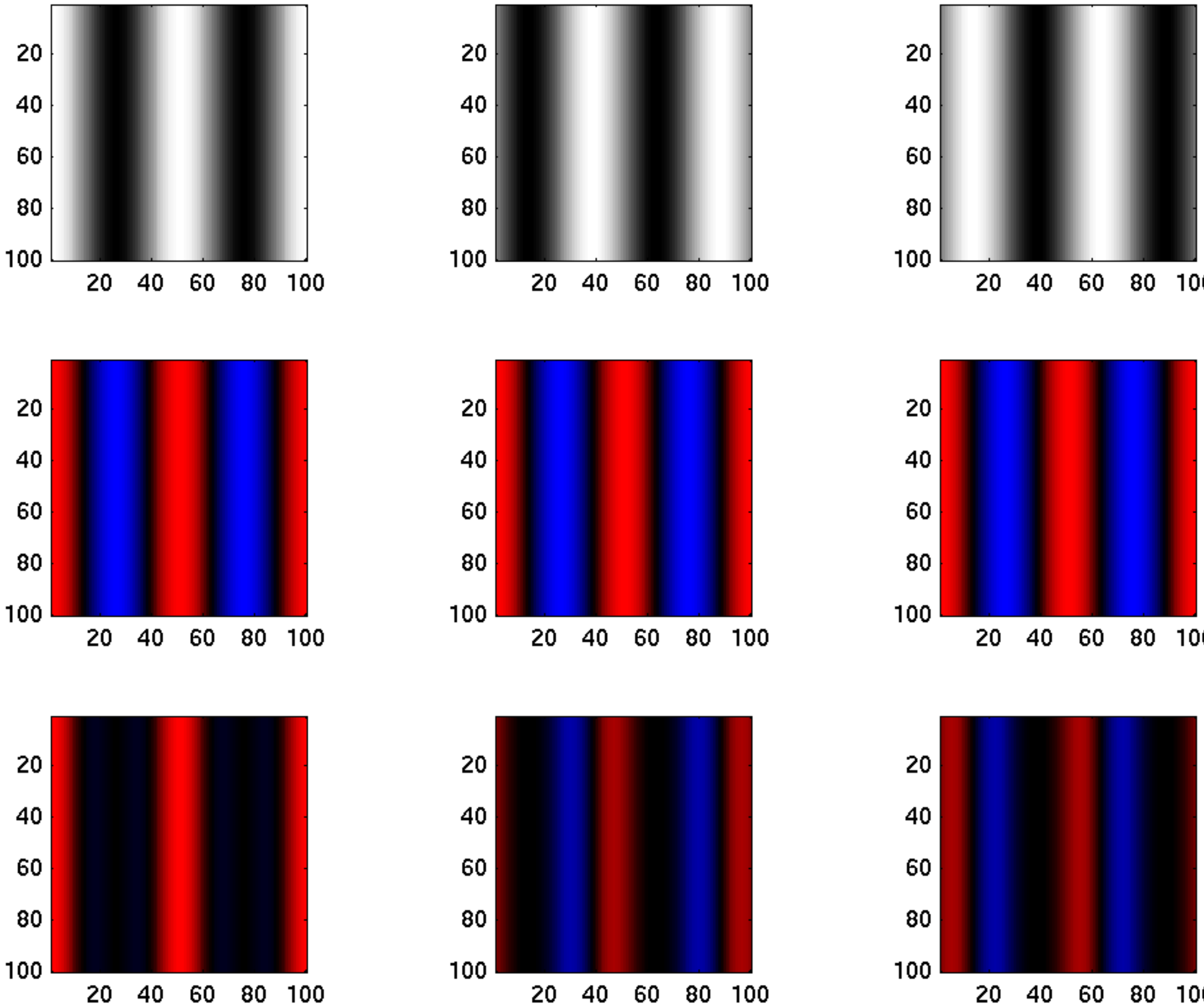
$$Hz = \frac{1}{s} \qquad F_s = \frac{\#samples}{s} \qquad f_{norm} = \frac{f_{skut}}{F_s} \qquad f_{skut} = \frac{k}{N} F_s$$

- 2D

$$dpi = \frac{1}{inch} \qquad F_s = \frac{\#pixels}{inch} \qquad f_{norm} = \frac{f_{skut}}{F_s}$$

$$f_{skut,vert} = \frac{m}{K} F_s, \quad f_{skut,horiz} = \frac{n}{L} F_s$$

# Phase – problem again ☹




c=2.5000e+03 c=9.9476e-13 c=-5.6843e-14

# Solution – complex exponentials (again)

$$X[m,n] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x[k,l] e^{-j\left[2\pi\left(\frac{m}{K}k + \frac{n}{L}l\right)\right]}$$

Reminder what is what :

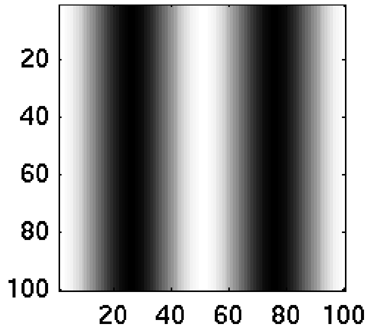
- *k,l* indices of pixels (input)
- *m,n* indices of frequencies (result)
- *m/K* normalized vertical frequency
- *n/L* normalized horizontal frequency

# Imagining it I.

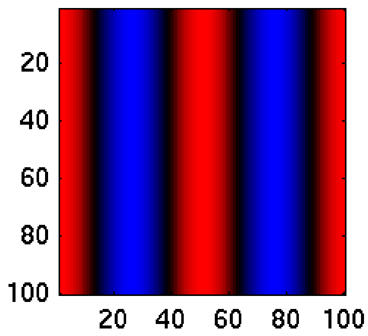
$$X[0,1] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x[k,l] e^{-j\left[2\pi\left(\frac{0}{K}k + \frac{1}{L}l\right)\right]} = \ldots$$

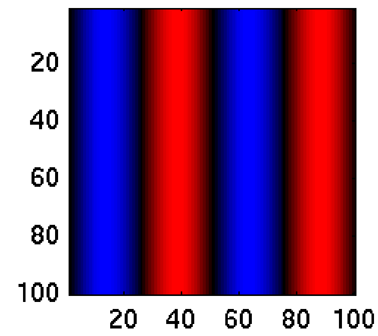
# Imagining it II.

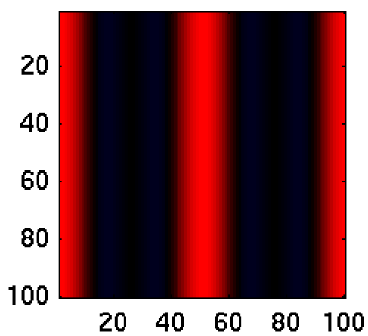
$$X[3,0] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x[k,l] e^{-j\left[2\pi\left(\frac{3}{K}k + \frac{0}{L}l\right)\right]} = \dots$$

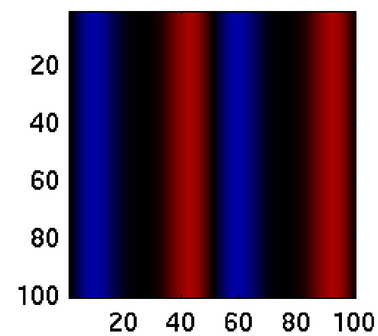



# Correlation with *x[n]*



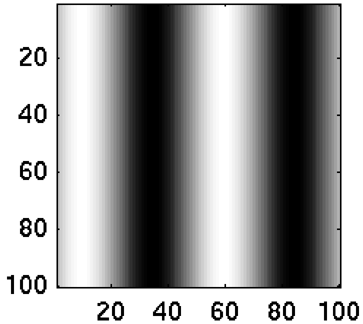



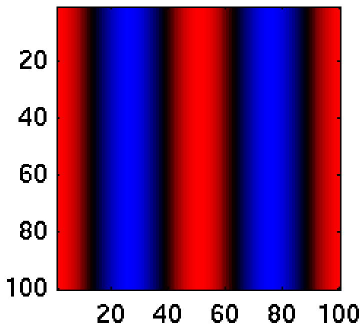

Re(c) = 2500

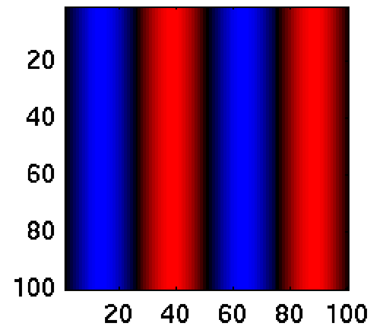
Im(c) = 0

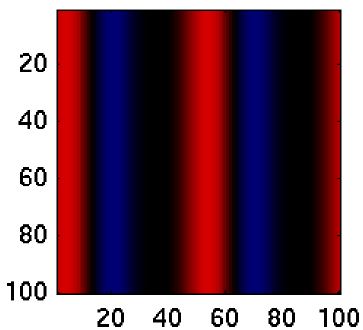
**|c| = 2500**

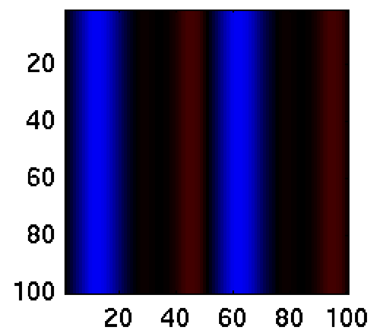
# Correlation with *x[n]*



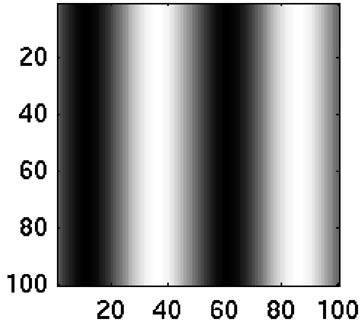



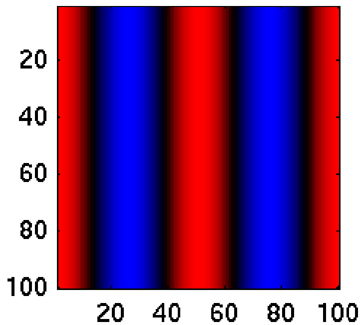

Re(c) = 1250

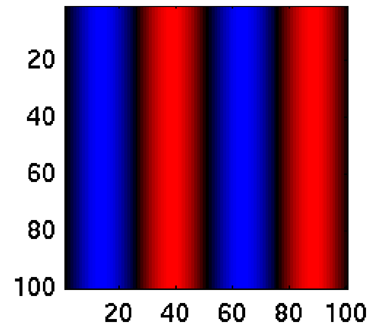
Im(c) = -2165

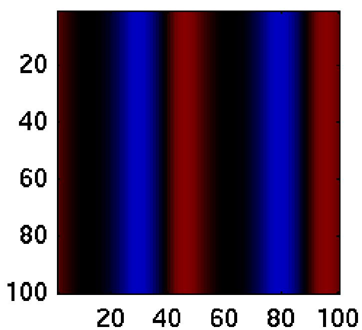
**|c| = 2500**

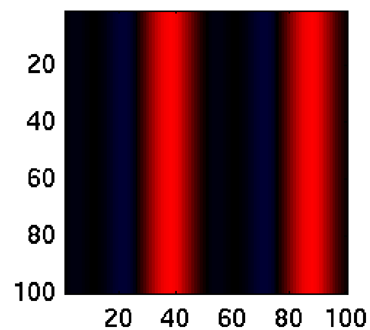
# Correlation with *x[n]*



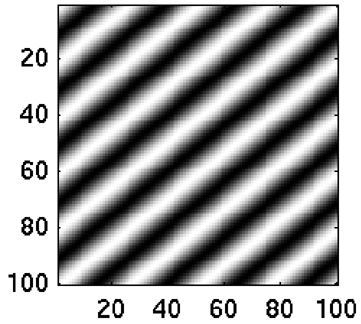



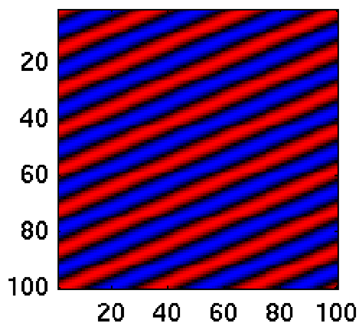

Re(c) = -772

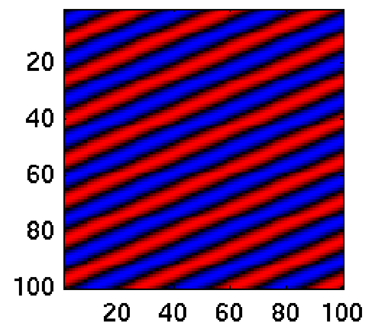
Im(c) = 2378

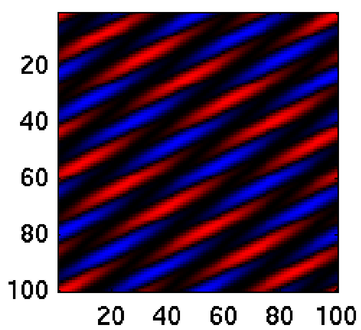
**|c| = 2500**

# Change in both directions…*X[7,3]*

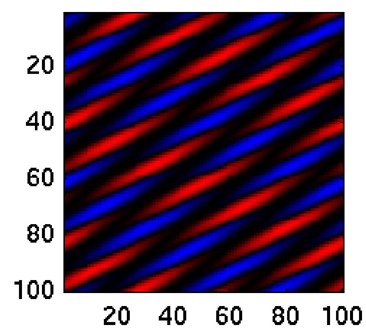


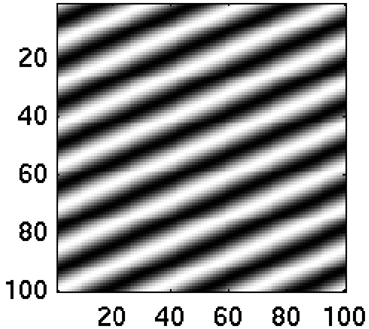

Re(c) = 0

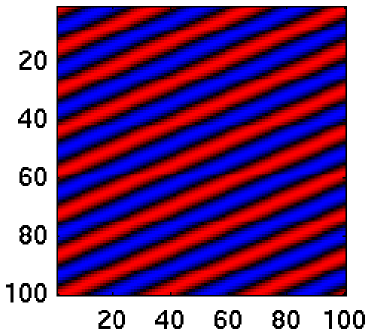


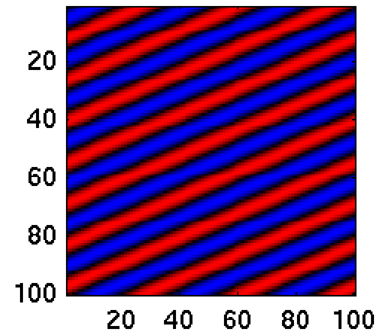

Im(c) = 0

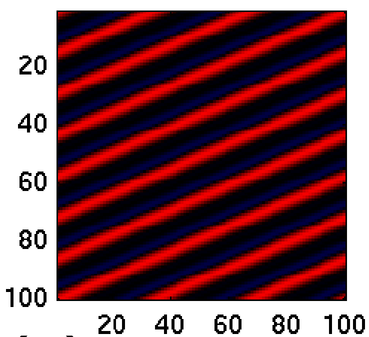
**|c| = 0 … not similar**

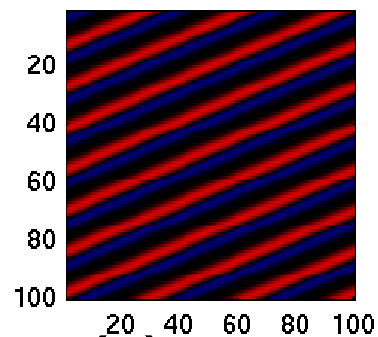
# Change in both directions…*X[7,3]*




Re(c) = 2165

Im(c) = 1250

**|c| = 2500 … similar**

# 2D DFT using 2 x 1D DFT

$$X[m,n] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} x[k,l] e^{-j2\pi\left(\frac{mk}{K}+\frac{nl}{L}\right)} =$$

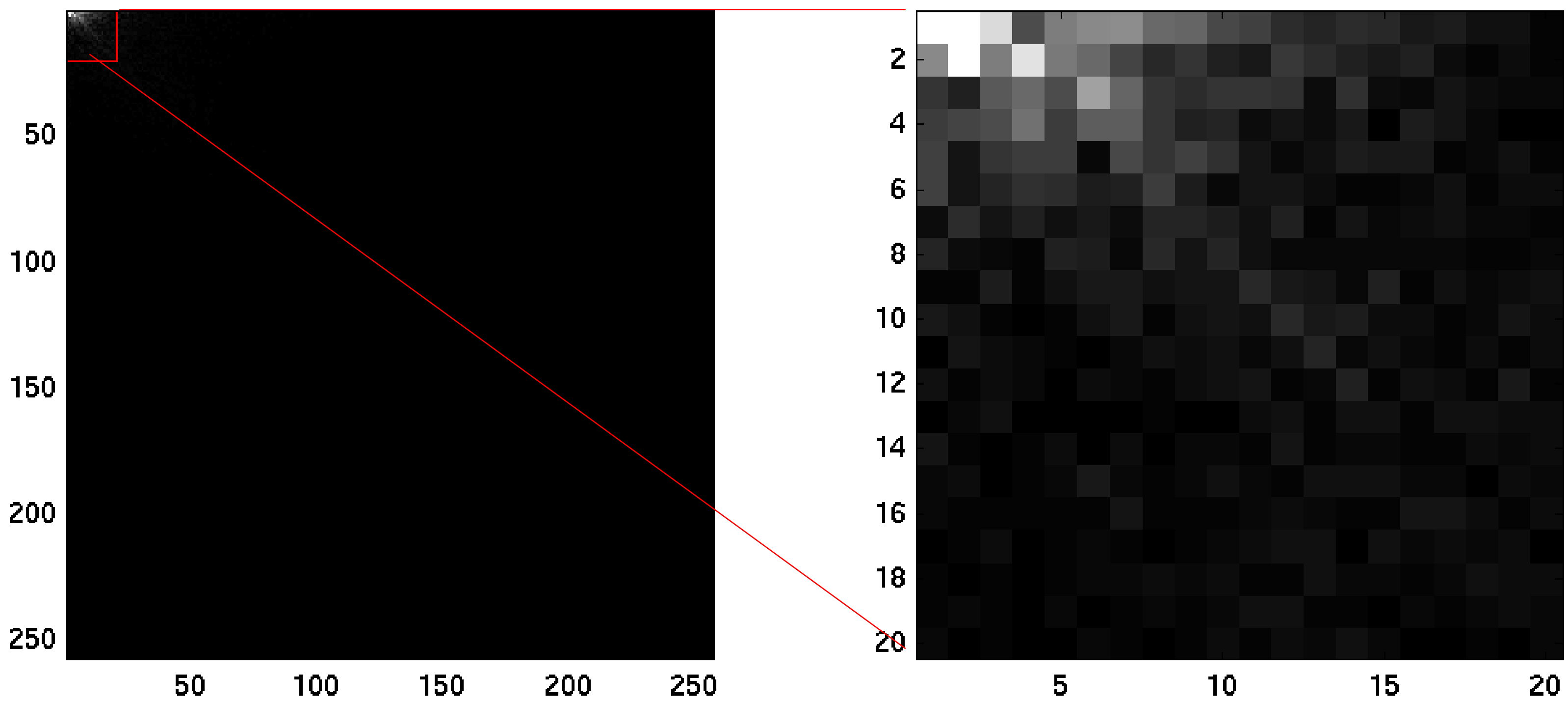
$$= \sum_{k=0}^{K-1} e^{-j2\pi\frac{mk}{M}} \sum_{l=0}^{L-1} x[k,l] e^{-j2\pi\frac{nl}{L}}, \quad \ldots \quad \text{or vice versa}$$

- so that

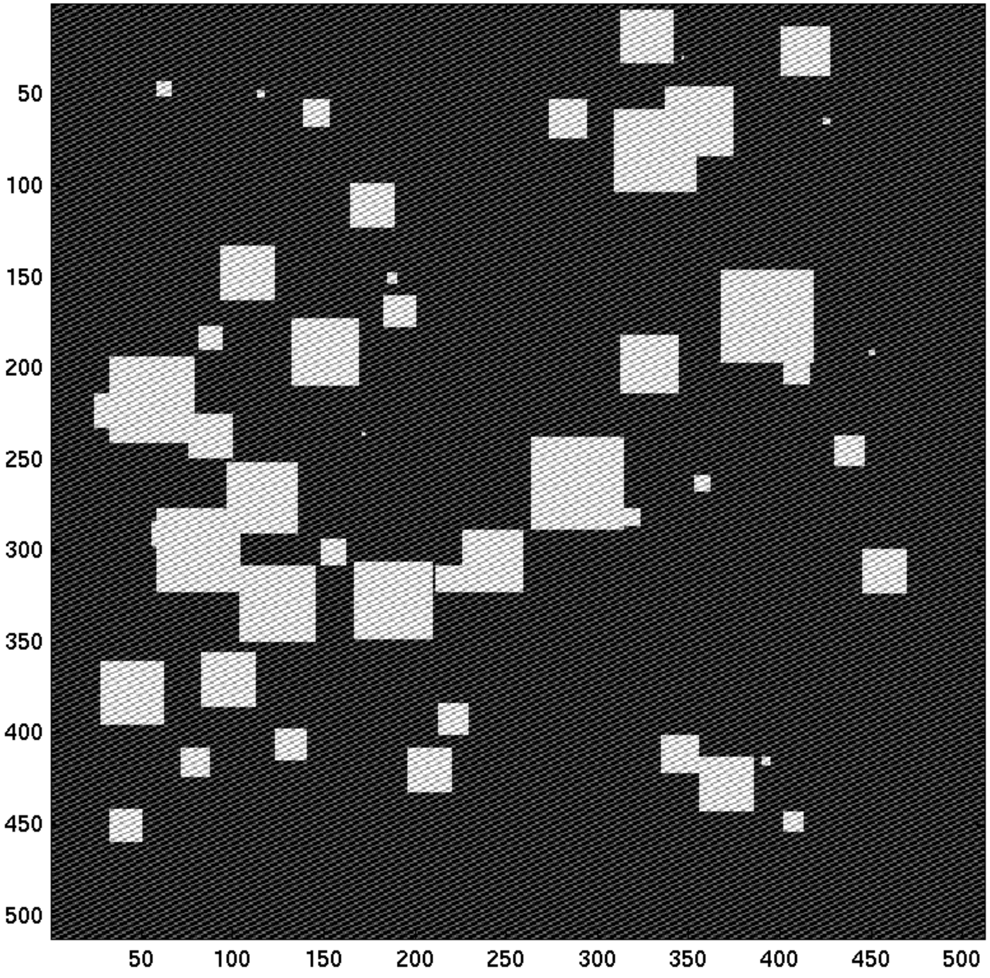
$$2DDFT\{x[k,l]\} =$$

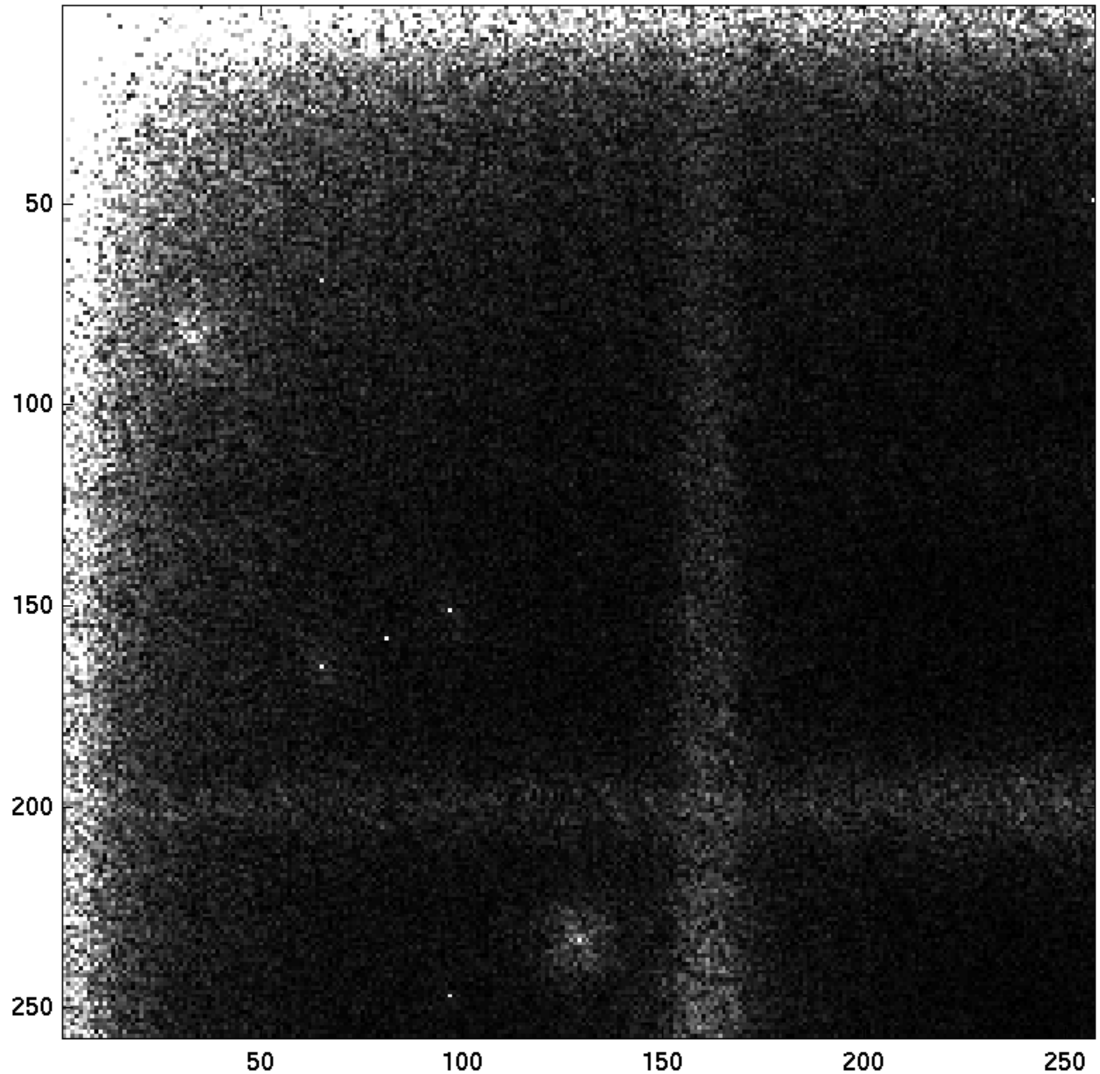
$$= 1DDFT_{columns}\{1DDFT_{rows} x[k,l]\} \quad \ldots \quad \text{or vice versa}$$

# 2D DFT for a real signal

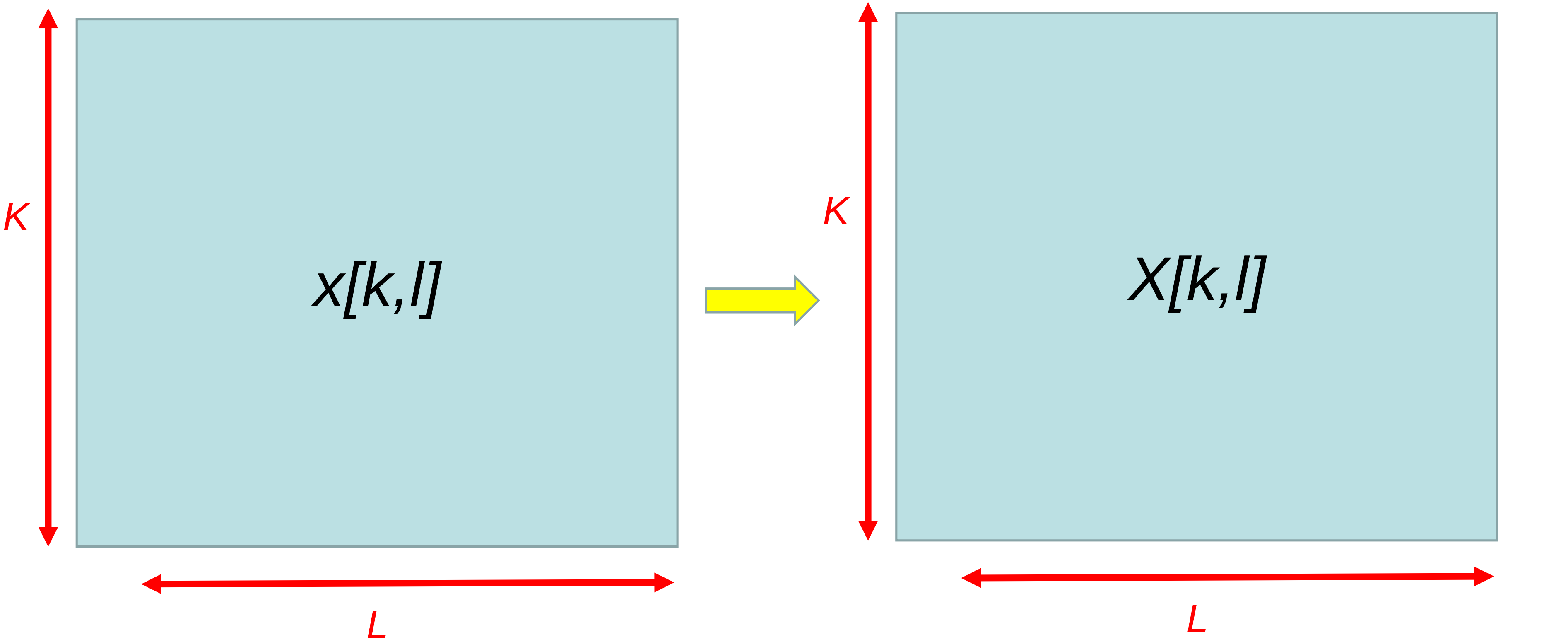

50
100
150
200
250
50
100
150
200
250
2
4
6
8
10
12
14
16
18
20
5
10
15
20

# Something with higher frequencies

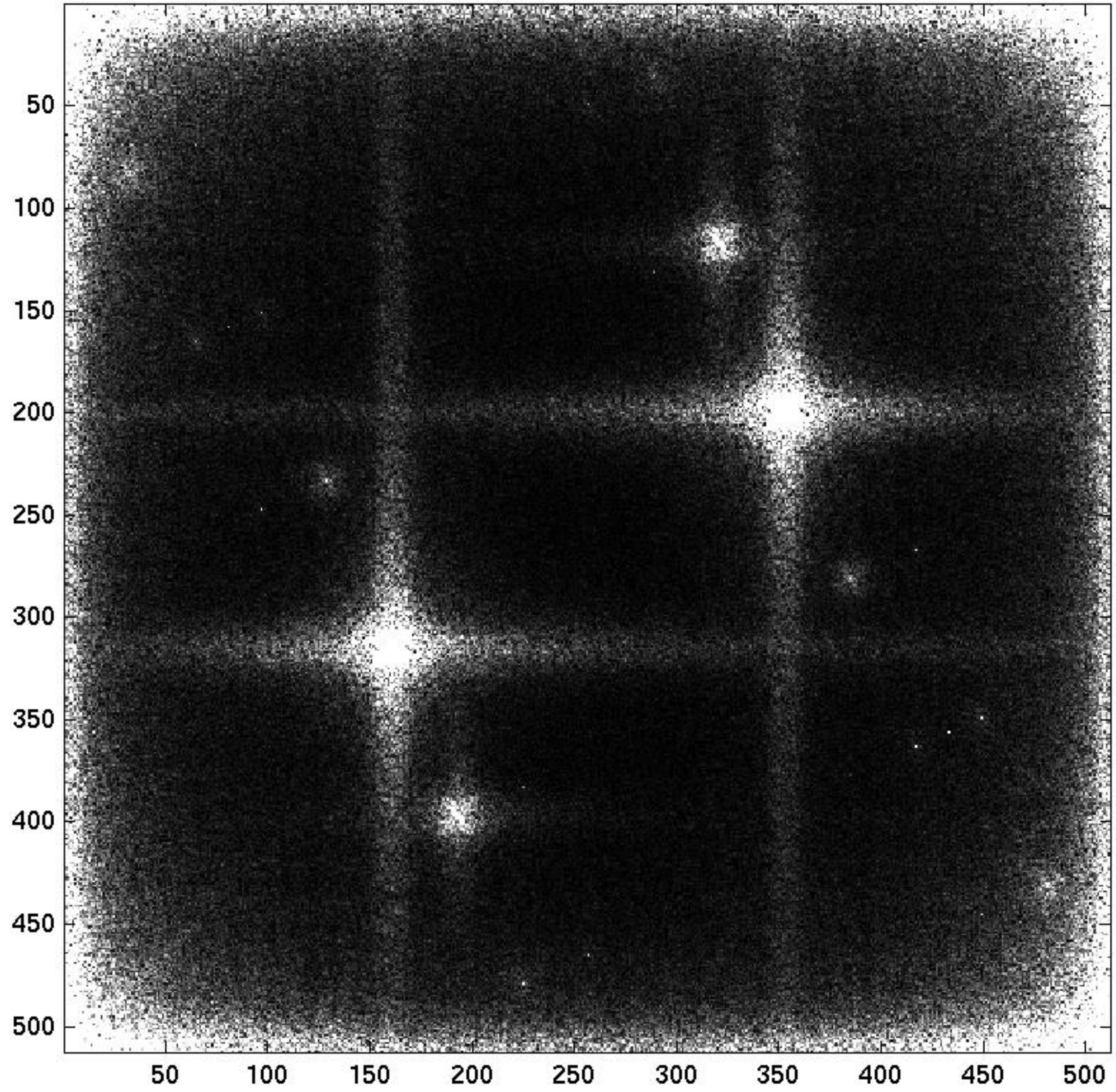



50
100
150
200
250
50
100
150
200
250

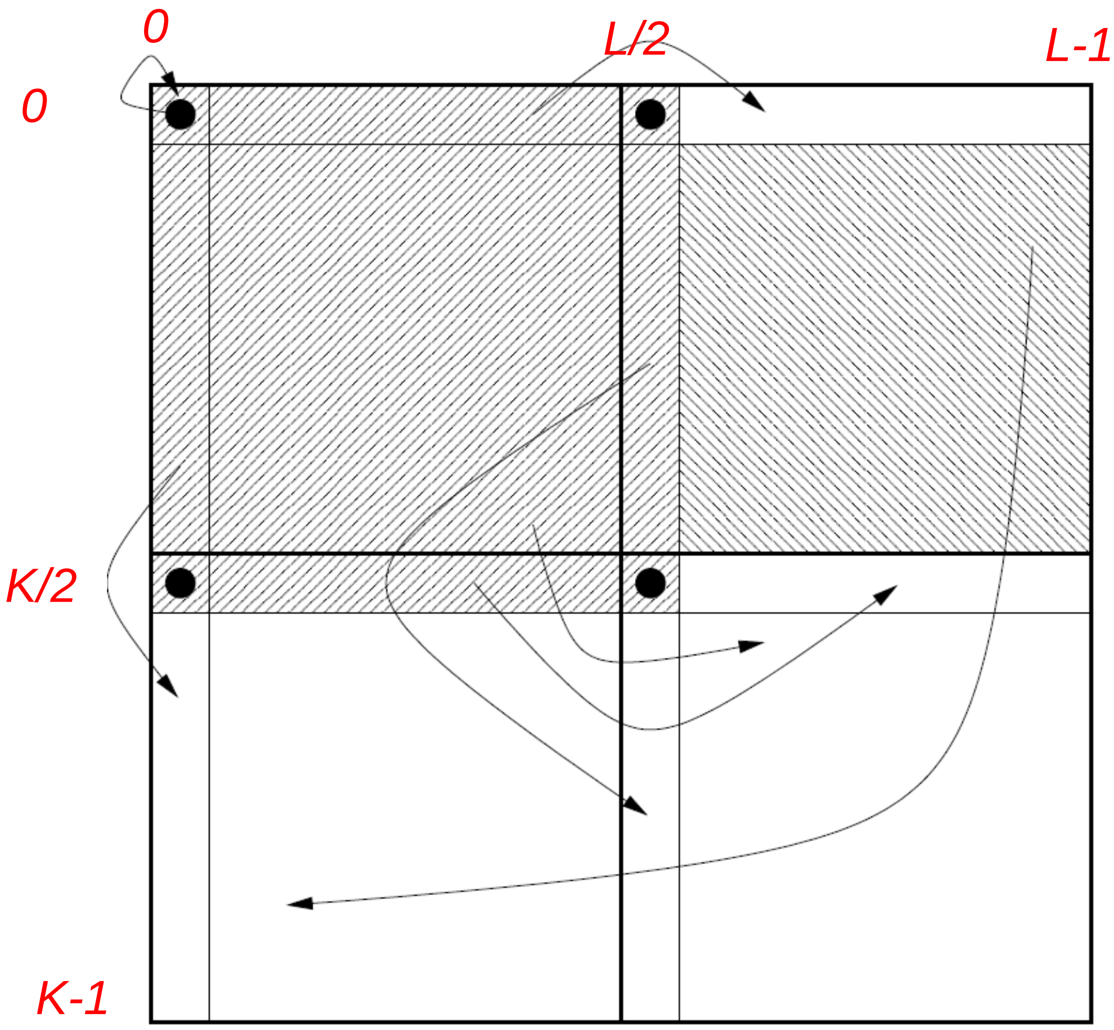
# DFT produces *K x L* points !



# The whole result for *k=0…K-1*, *l=0…L-1*



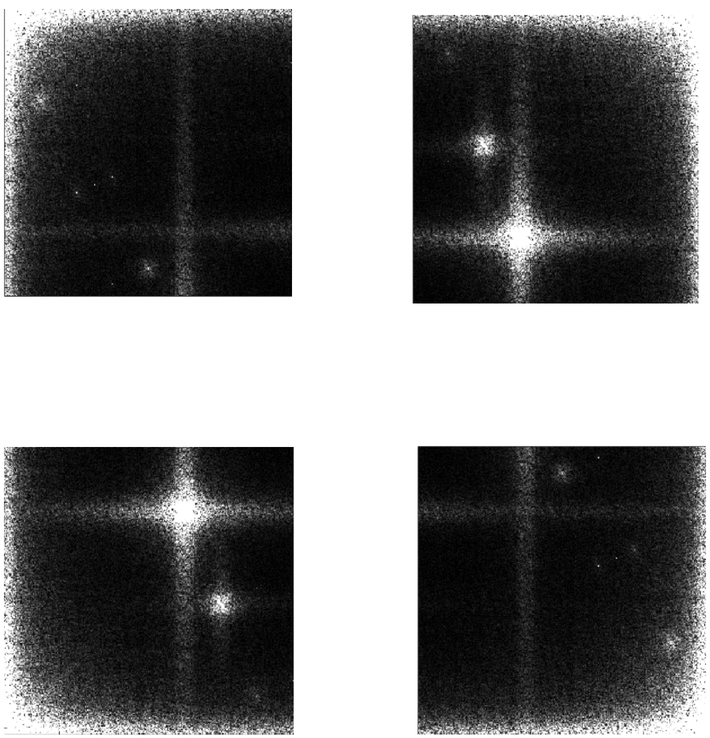
# Symmetry ? $X[m,n] = X^{\star}[K-m, L-n]$

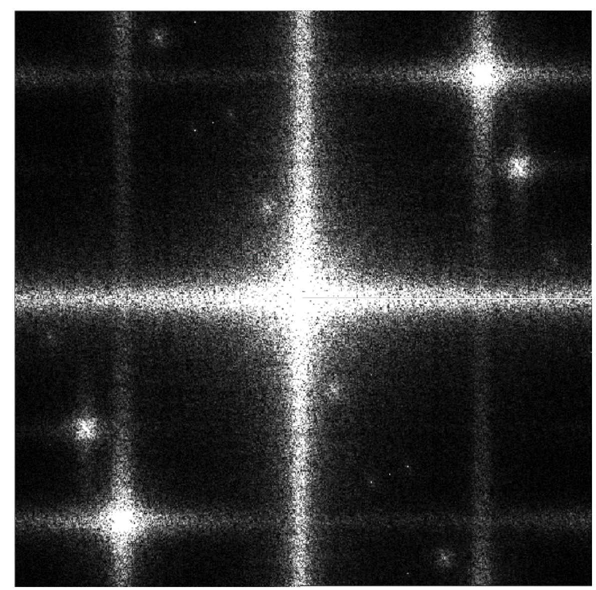



Source: http://www-personal.umich.edu/~hyunjinp/notes/n-dft.pdf

# Re-shuffling…

- Low frequencies to the center

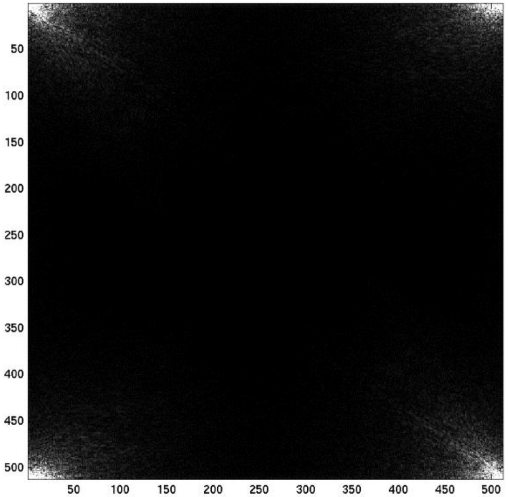




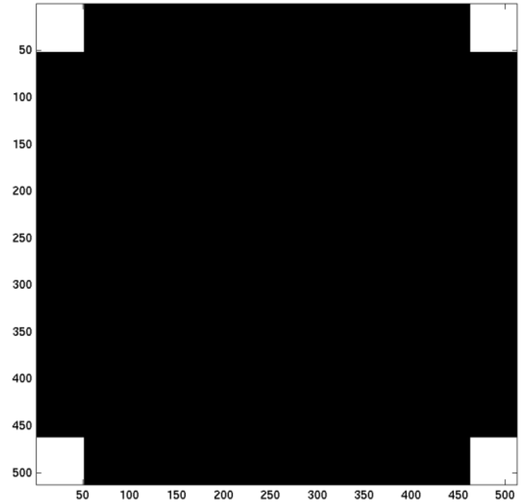
# Inverse 2D DFT

$$x[k,l] = \frac{1}{KL} \sum_{m=0}^{K-1} \sum_{n=0}^{L-1} X[m,n] e^{+j2\pi\left(\frac{mk}{K} + \frac{nl}{L}\right)}$$

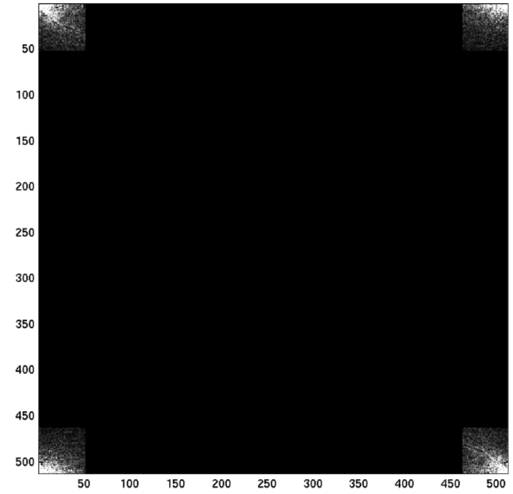
# Playing with frequencies

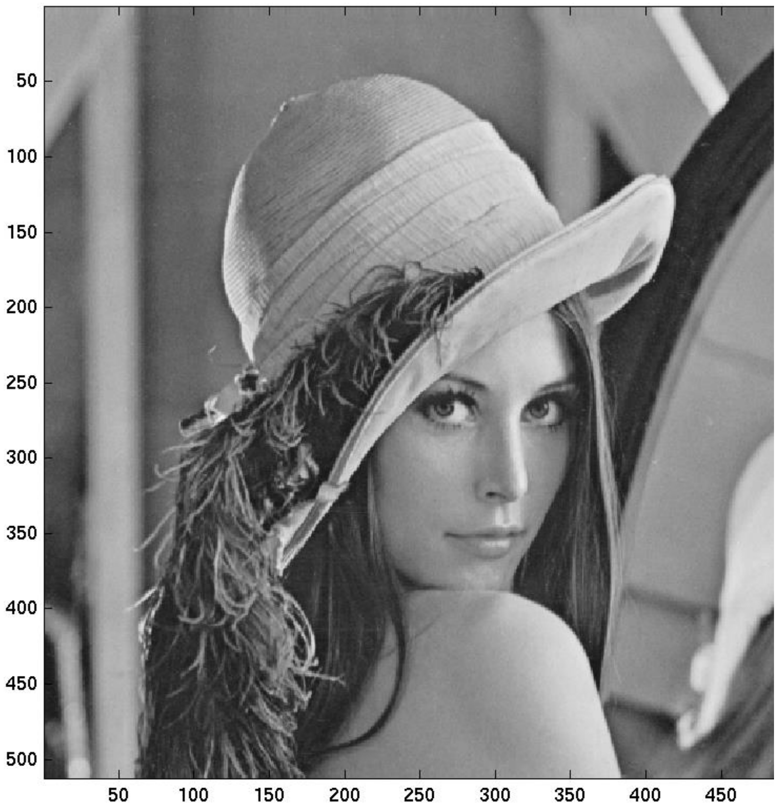


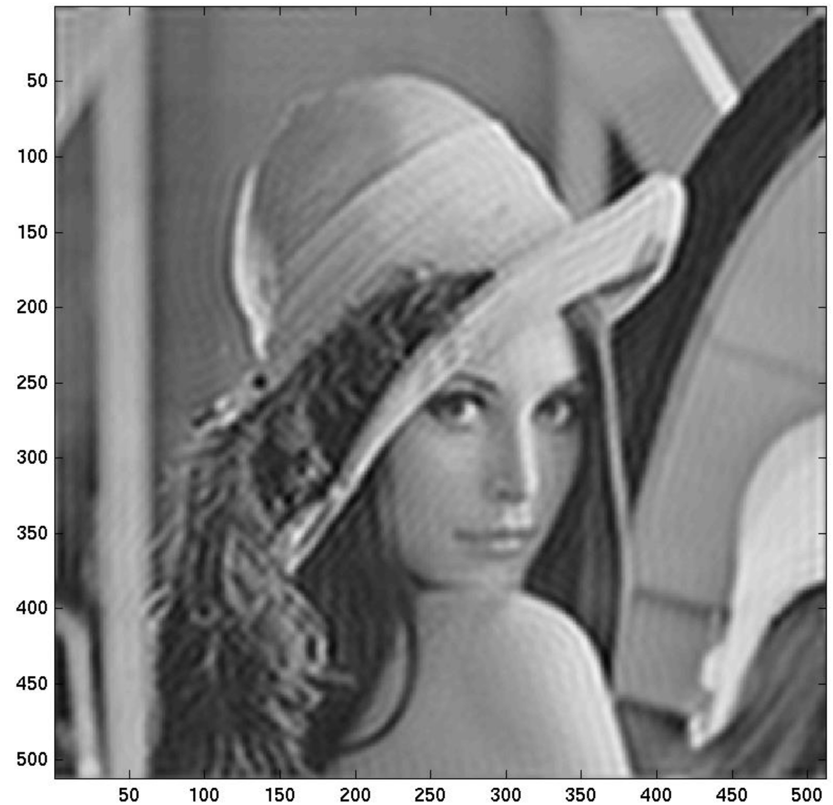

×




=




50
100
150
200
250
300
350
400
450
500
50 100 150 200 250 300 350 400 450


50
100
150
200
250
300
350
400
450
500
50 100 150 200 250 300 350 400 450 500

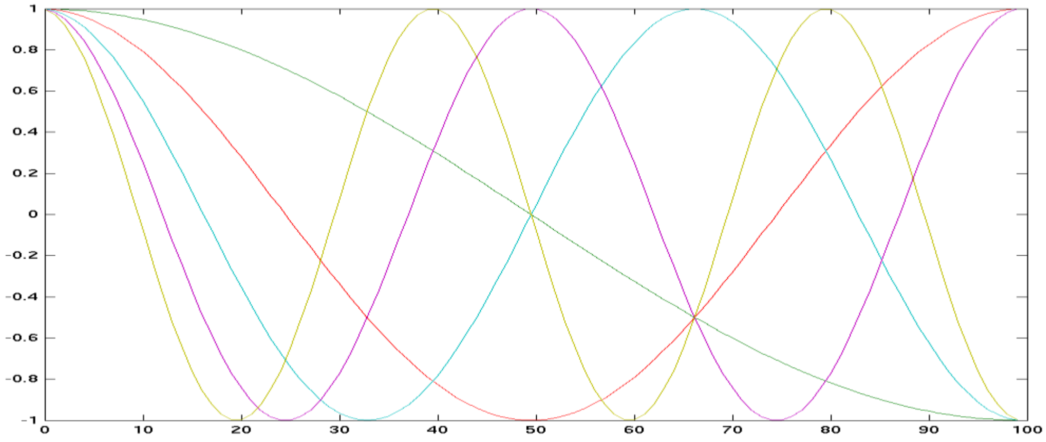
# DCT

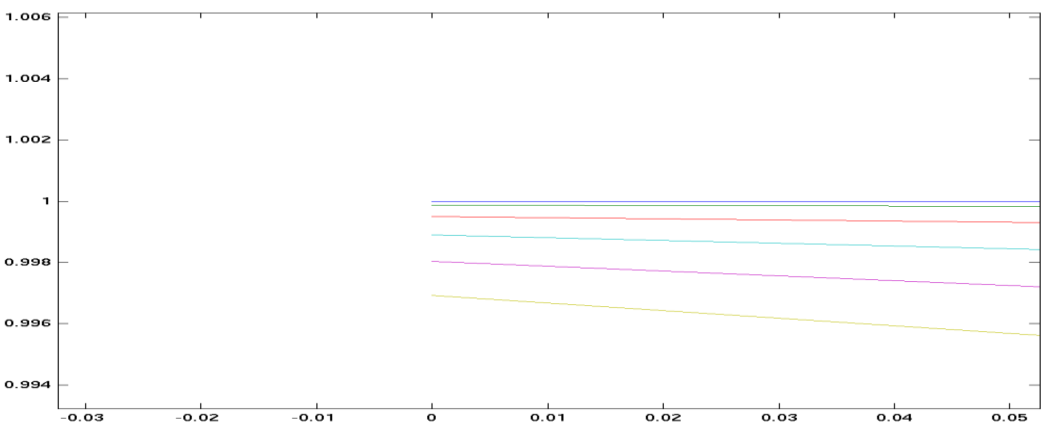
Why ?

- People don't like complex exponentials
- People don't like symmetries and "useless" values …
- For image *K x L,* we want *K x L* real values.

# 1D DCT

$$X[k] = \sum_{n=0}^{N-1} x[n] \cos\left[\frac{\pi}{n}(n+\frac{1}{2})k\right]$$




+ normalization of coefficients…

$$X[0] \ldots \times \sqrt{\frac{1}{N}}$$

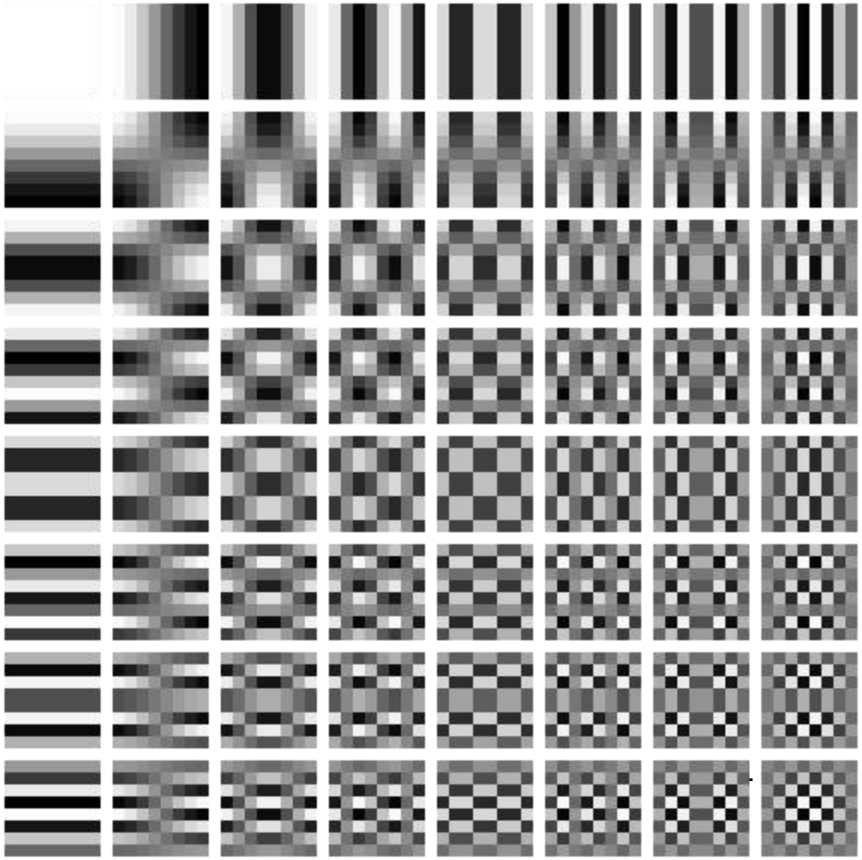
$$X[1 \ldots N-1] \ldots \times \sqrt{\frac{2}{N}}$$

# 2D DCT bases

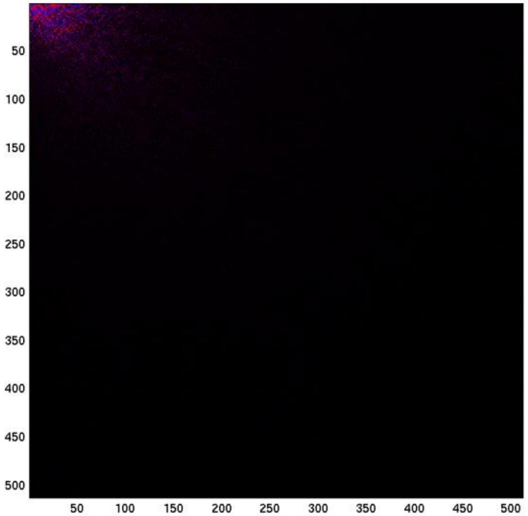
$$X_{k_1,k_2} = \sum_{n_1=0}^{N_1-1} \left( \sum_{n_2=0}^{N_2-1} x_{n_1,n_2} \cos\left[\frac{\pi}{N_2}\left(n_2+\frac{1}{2}\right)k_2\right]\right) \cos\left[\frac{\pi}{N_1}\left(n_1+\frac{1}{2}\right)k_1\right]$$

$$= \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x_{n_1,n_2} \cos\left[\frac{\pi}{N_1}\left(n_1+\frac{1}{2}\right)k_1\right] \cos\left[\frac{\pi}{N_2}\left(n_2+\frac{1}{2}\right)k_2\right].$$

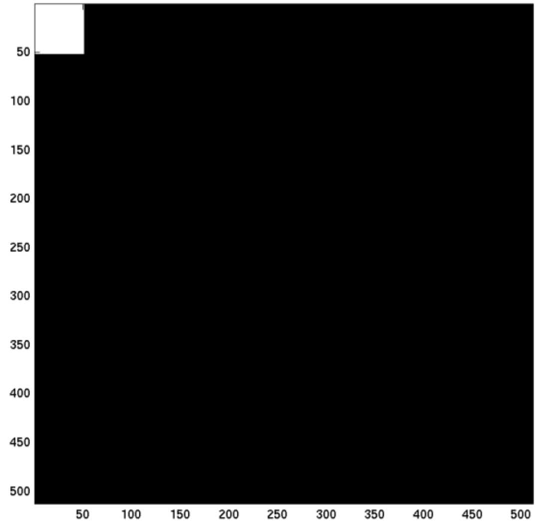
Source:
https://en.wikipedia.org/wiki/Discrete_cosine_transform#Multidimensional_DCTs



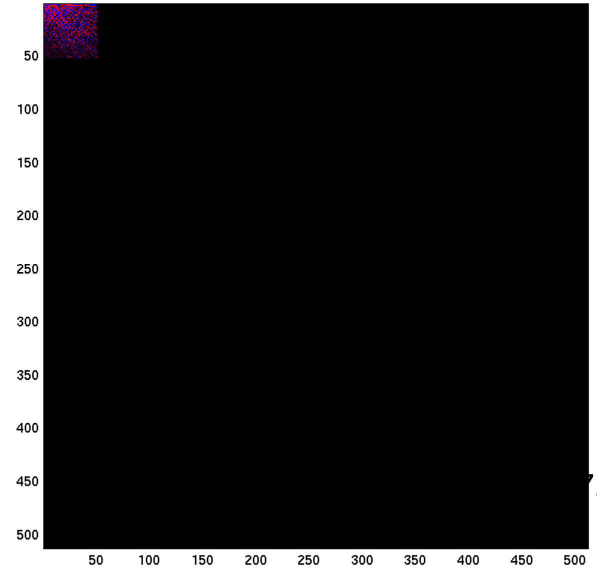
# Filtering in DCT



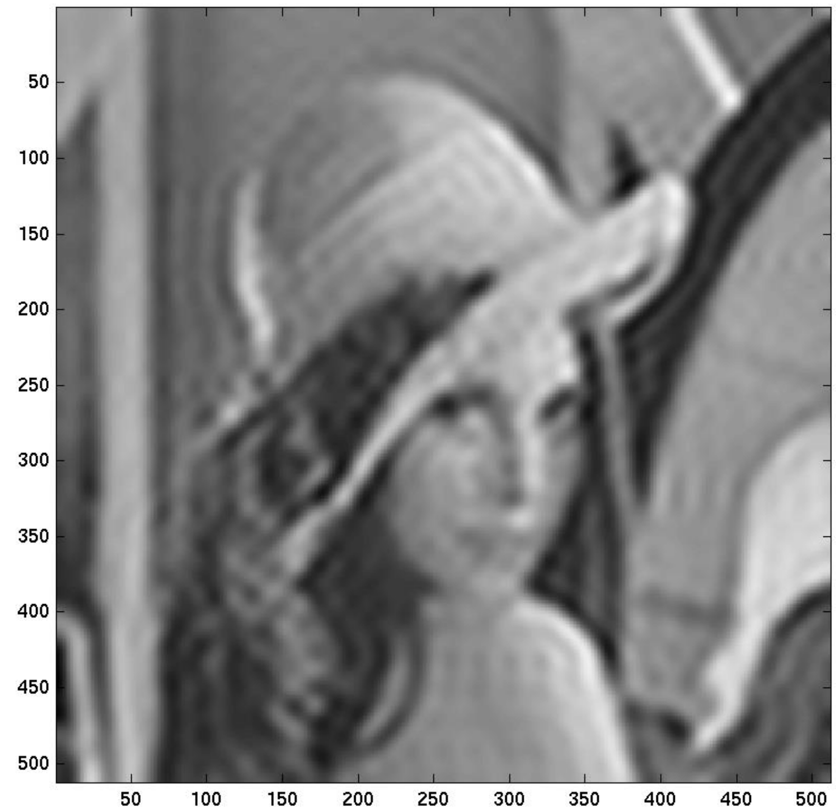

×



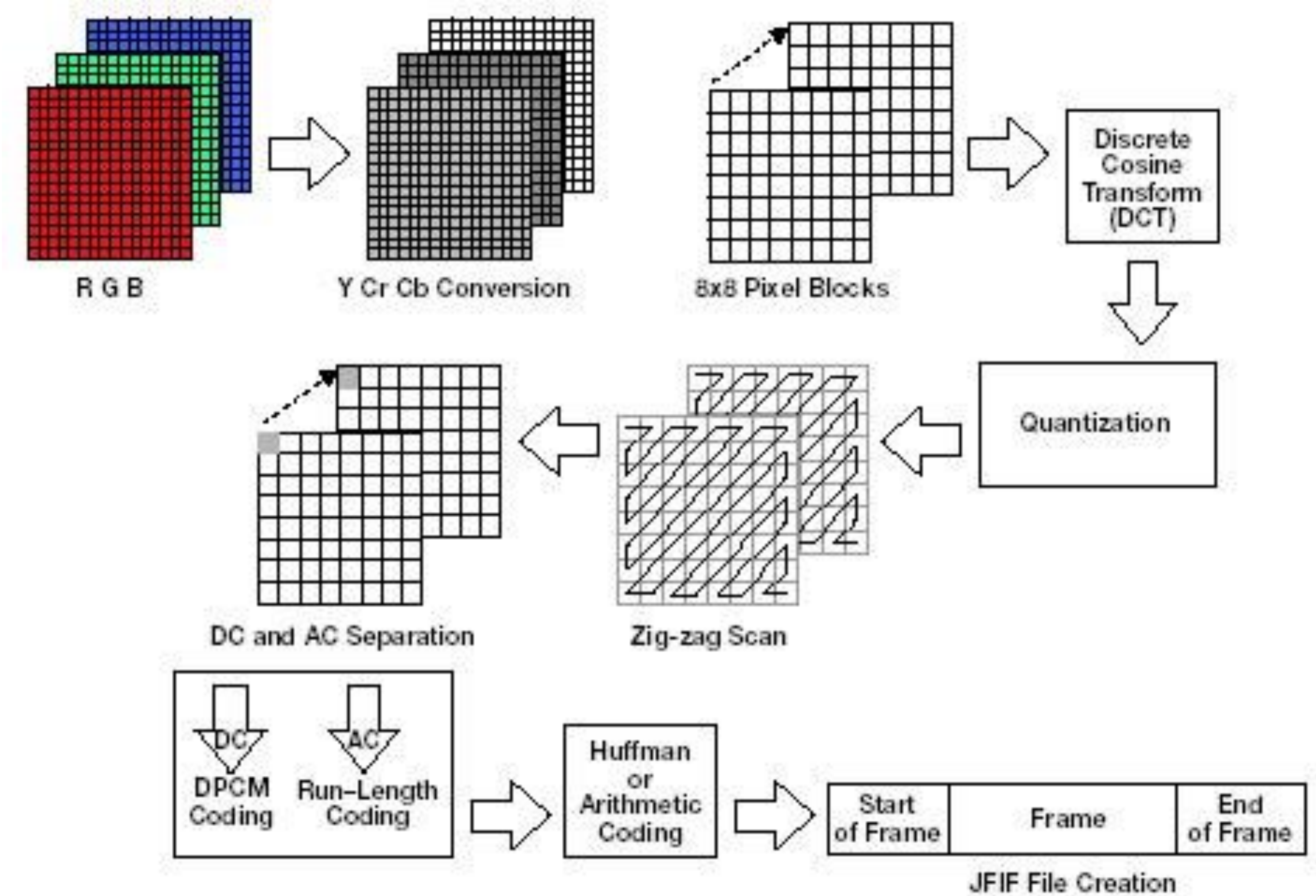

=




50
100
150
200
250
300
350
400
450
500
50 100 150 200 250 300 350 400 450 500


50
100
150
200
250
300
350
400
450
500
50 100 150 200 250 300 350 400 450 500

# JPEG




- Source: http://www.eetimes.com/document.asp?doc_id=1225736
- More in computer graphics courses and lab exercise.

# SUMMARY

- Image is a 2D signal
- Filtering
  - Convolution, analogy with 1D FIR filters
  - No feedback (IIR) in images
- Sweeping mask over the image, multiplying everything that is underneath with its coefficients, adding.

# SUMMARY II.

- Frequency analysis 2D
  - Similar to 1D – projection to bases
  - Image frequencies also have some sense.
  - Cos bases are a good exercise, but not enough.
- 2D DFT
  - Projection/similarity/correlation with complex exponentials
  - Coefficients are complex – magnitude and angle.
  - Lots of symmetries in the resulting DFT matrix
  - Can use spectrum for filtering

# SUMMARY III.

- DCT
  - 2x „slower“ bases than DFT
  - Slightly More complicated definition
  - Produces real coefficients, low frequencies (only!) at the beginning.
  - Use in JPEG

# TO BE DONE

- Determining frequency response of a 2D filter
- How is it exactly with the symmetries of 2D DFT ?
- Why are 1D and 2D DCT exactly defined and why this half-sample shift ?
- Colors (color models, etc).
- How does the face matching on Facebook work?
  - Hint: https://research.facebook.com/researchers/684639631606527/yann-lecun/

# The END