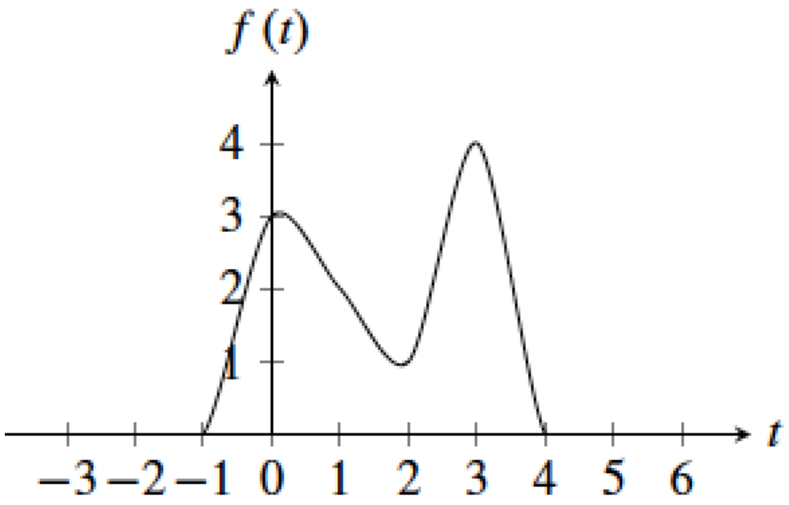
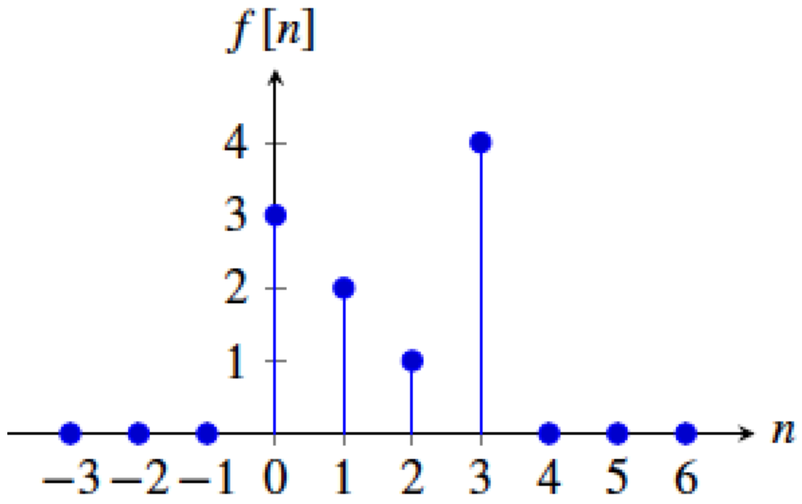
Lecture 4
Signal Processing

Bill Freeman, Antonio Torralba, Phillip Isola

6.003 Signals and systems



Time continuous signal



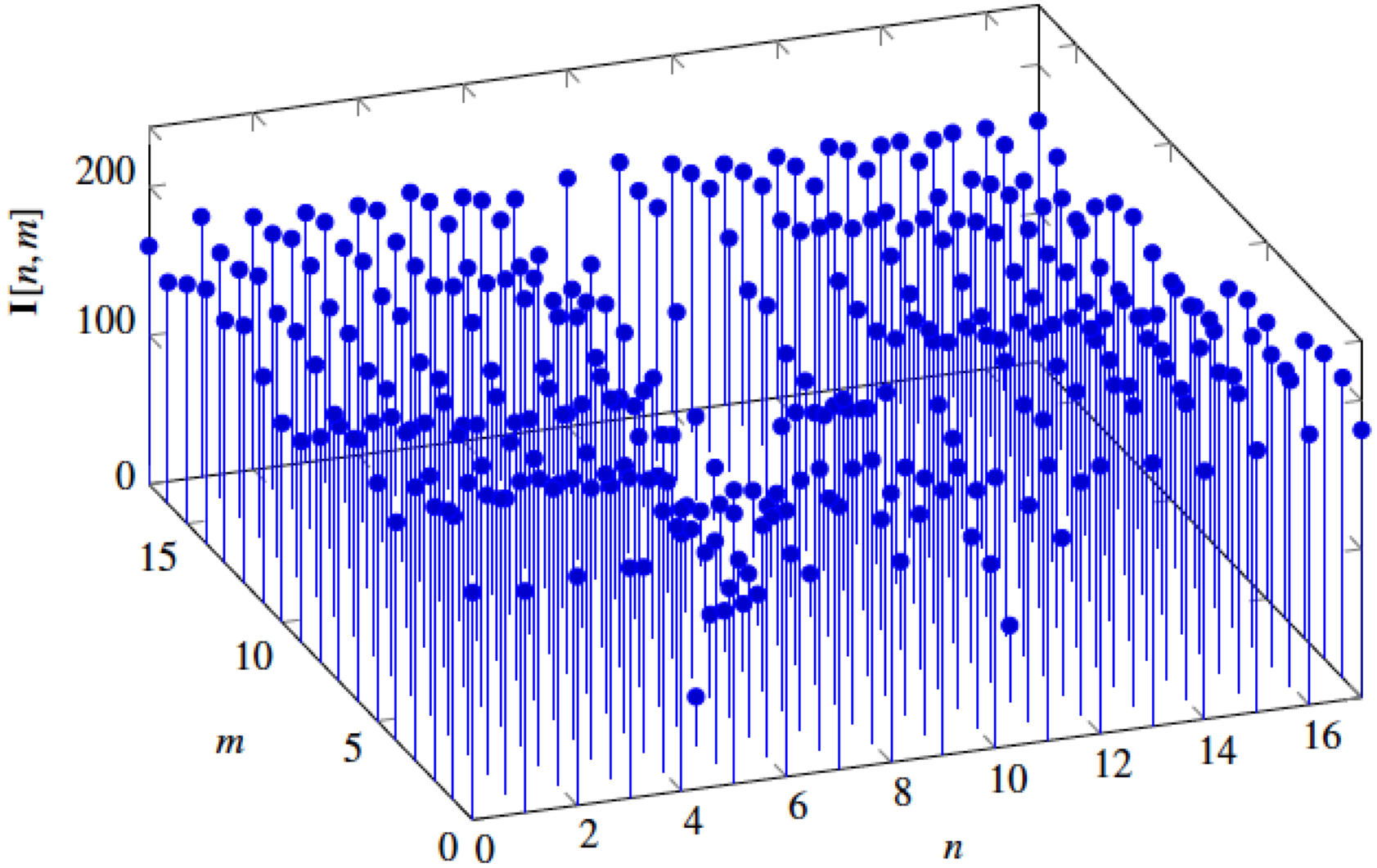
Time discrete signal

Review class notes!

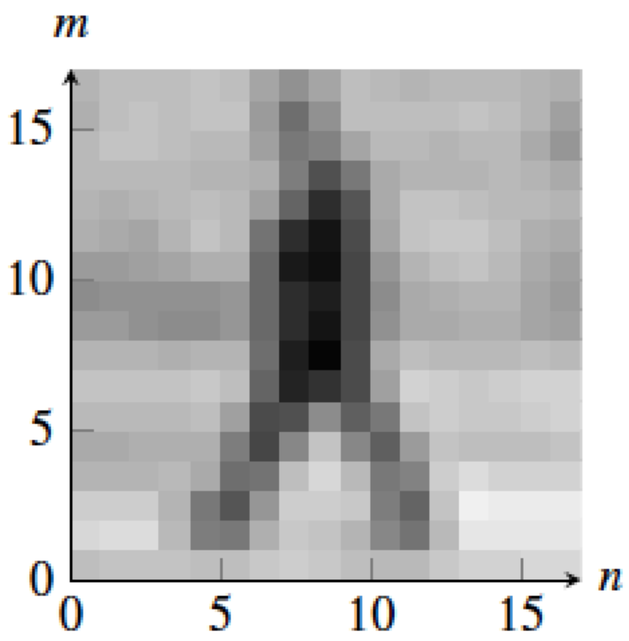
A 2D discrete signal

$$
\mathbf{I} = \begin{bmatrix}
160 & 175 & 171 & 168 & 168 & 172 & 164 & 158 & 167 & 173 & 167 & 163 & 162 & 164 & 160 & 159 & 163 & 162 \\
149 & 164 & 172 & 175 & 178 & 179 & 176 & 118 & 97 & 168 & 175 & 171 & 169 & 175 & 176 & 177 & 165 & 152 \\
161 & 166 & 182 & 171 & 170 & 177 & 175 & 116 & 109 & 169 & 177 & 173 & 168 & 175 & 175 & 159 & 153 & 123 \\
171 & 174 & 177 & 175 & 167 & 161 & 157 & 138 & 103 & 112 & 157 & 164 & 159 & 160 & 165 & 169 & 148 & 144 \\
163 & 163 & 162 & 165 & 167 & 164 & 178 & 167 & 77 & 55 & 134 & 170 & 167 & 162 & 164 & 175 & 168 & 160 \\
173 & 164 & 158 & 165 & 180 & 180 & 150 & 89 & 61 & 34 & 137 & 186 & 186 & 182 & 175 & 165 & 160 & 164 \\
152 & 155 & 146 & 147 & 169 & 180 & 163 & 51 & 24 & 32 & 119 & 163 & 175 & 182 & 181 & 162 & 148 & 153 \\
134 & 135 & 147 & 149 & 150 & 147 & 148 & 62 & 36 & 46 & 114 & 157 & 163 & 167 & 169 & 163 & 146 & 147 \\
135 & 132 & 131 & 125 & 115 & 129 & 132 & 74 & 54 & 41 & 104 & 156 & 152 & 156 & 164 & 156 & 141 & 144 \\
151 & 155 & 151 & 145 & 144 & 149 & 143 & 71 & 31 & 29 & 129 & 164 & 157 & 155 & 159 & 158 & 156 & 148 \\
172 & 174 & 178 & 177 & 177 & 181 & 174 & 54 & 21 & 29 & 136 & 190 & 180 & 179 & 176 & 184 & 187 & 182 \\
177 & 178 & 176 & 173 & 174 & 180 & 150 & 27 & 101 & 94 & 74 & 189 & 188 & 186 & 183 & 186 & 188 & 187 \\
160 & 160 & 163 & 163 & 161 & 167 & 100 & 45 & 169 & 166 & 59 & 136 & 184 & 176 & 175 & 177 & 185 & 186 \\
147 & 150 & 153 & 155 & 160 & 155 & 56 & 111 & 182 & 180 & 104 & 84 & 168 & 172 & 171 & 164 & 168 & 167 \\
184 & 182 & 178 & 175 & 179 & 133 & 86 & 191 & 201 & 204 & 191 & 79 & 172 & 220 & 217 & 205 & 209 & 200 \\
184 & 187 & 192 & 182 & 124 & 32 & 109 & 168 & 171 & 167 & 163 & 51 & 105 & 203 & 209 & 203 & 210 & 205 \\
191 & 198 & 203 & 197 & 175 & 149 & 169 & 189 & 190 & 173 & 160 & 145 & 156 & 202 & 199 & 201 & 205 & 202 \\
153 & 149 & 153 & 155 & 173 & 182 & 179 & 177 & 182 & 177 & 182 & 185 & 179 & 177 & 167 & 176 & 182 & 180
\end{bmatrix}
$$

A 2D discrete signal

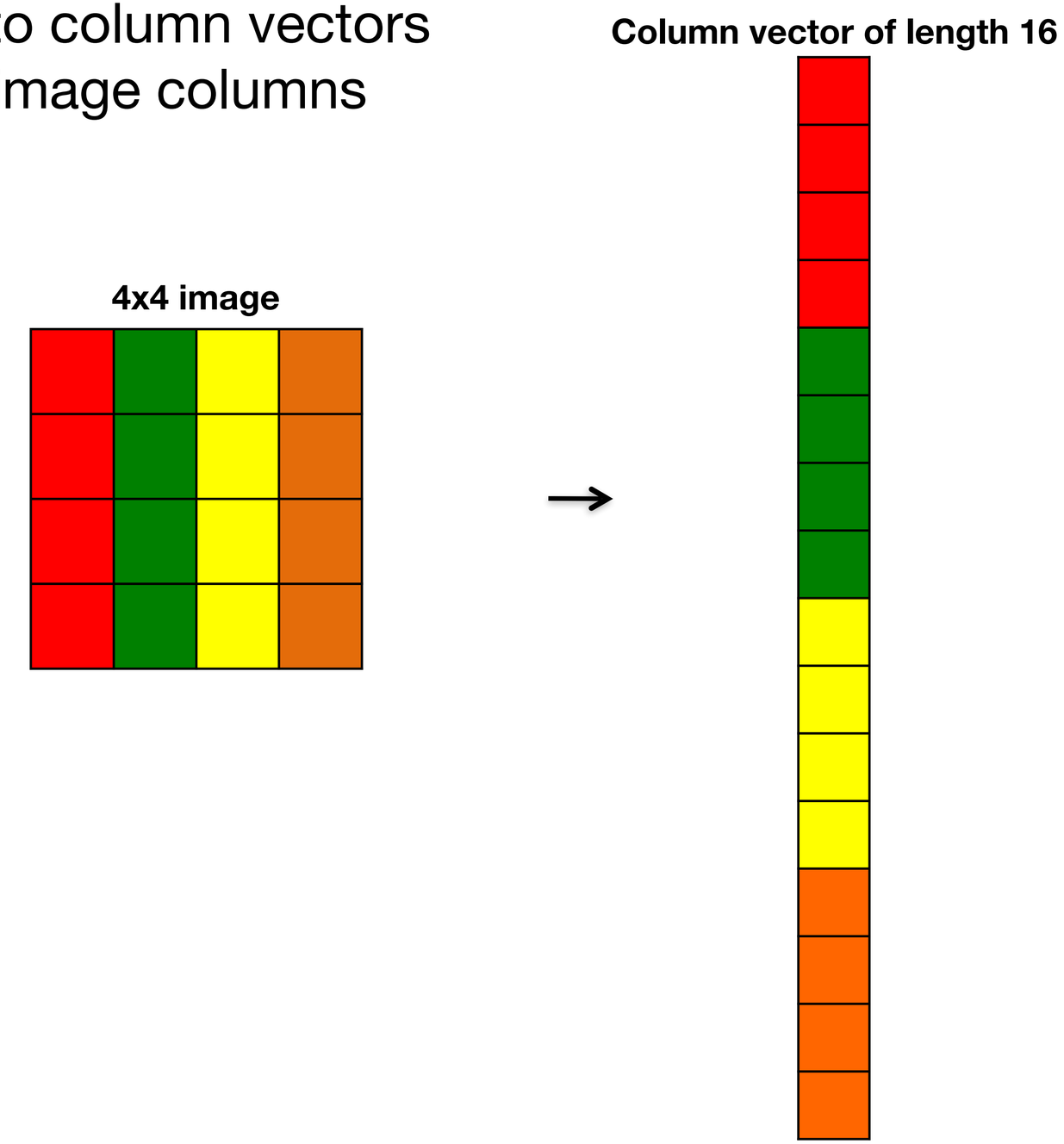


A 2D discrete signal



A tiny person of 18 x 18 pixels

Images are turned into column vectors by concatenating all image columns



Signal / image space

Scalar product between two signals f, g :

$$\langle f, g \rangle = \sum_{n=0}^{N-1} f[n]\, g^*[n] = f^T g^*$$

L2 norm of f :

$$E_f = \|f\|^2 = \langle f, f \rangle = \sum_{n=0}^{N-1} |f[n]|^2 = f^T f^*$$

Distance between two signals f, g :

$$d_{f,g}^2 = \|f - g\|^2 = \sum_{n=0}^{N-1} |f[n] - g[n]|^2 = E_f + E_g - 2\langle f, g \rangle$$

Filtering

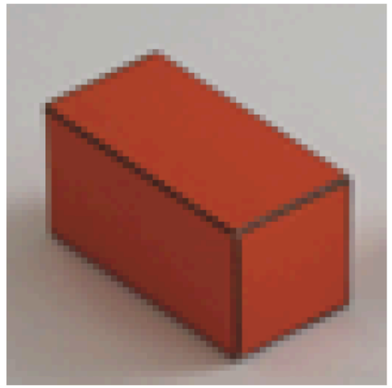
Input image

$$g[n,m] \rightarrow \boxed{\mathrm{H}} \rightarrow f[n,m]$$

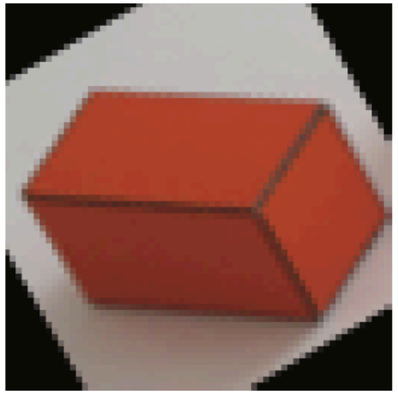
Output image

We want to remove unwanted sources of variation, and keep the information relevant for whatever task we need to solve

Linear filtering



$$g[n,m] \rightarrow \boxed{\mathrm{H}} \rightarrow f[n,m]$$

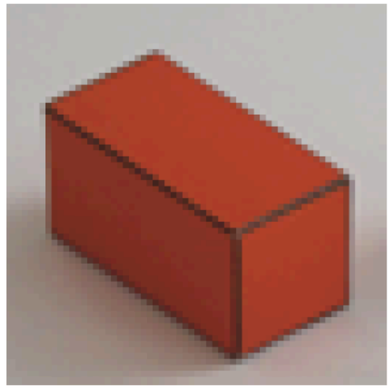


For a filter, H, to be linear, it has to verify:

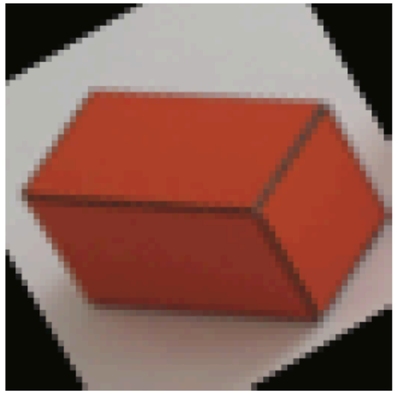
H(a [m,n] + b [m,n]) = H(a [m,n]) + H(b [m,n])

H(C a [m,n]) = C H(a [m,n])

Linear filtering



$$g[n,m] \rightarrow \boxed{\mathrm{H}} \rightarrow f[n,m]$$



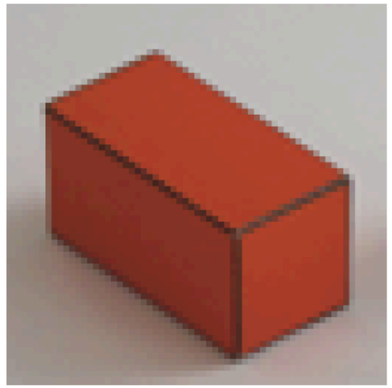
A linear filter in its most general form can be written as (for a 1D signal of length N):

$$f[n] = \sum_{k=0}^{N-1} h[n,k]\, g[k]$$

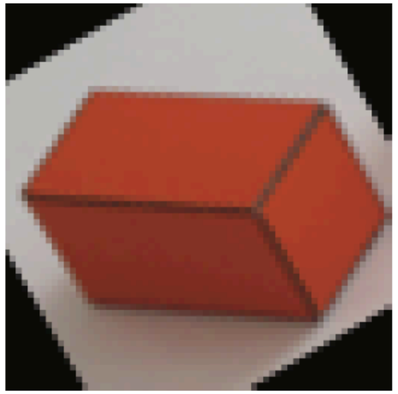
It is useful to write this a matrix and vector multiplications:

$$\begin{bmatrix} f[0] \\ f[1] \\ \vdots \\ f[M-1] \end{bmatrix} = \begin{bmatrix} h[0,0] & h[0,1] & \dots & h[0,N-1] \\ h[1,0] & h[1,1] & \dots & h[1,N-1] \\ \vdots & \vdots & \vdots & \vdots \\ h[M-1,0] & h[M-1,1] & \dots & h[M-1,N-1] \end{bmatrix} \begin{bmatrix} g[0] \\ g[1] \\ \vdots \\ g[N-1] \end{bmatrix}$$

Linear filtering



$$g[n,m] \rightarrow \boxed{\mathrm{H}} \rightarrow f[n,m]$$



A linear filter in its most general form can be written as (for a 2D image of size NxM):

$$f[n,m] = \sum_{k,\, l=0}^{N-1,\, M-1} h[n,m,k,l]\, g[k,l]$$

Which can also be written in matrix form as in the 1D case:

We need translation invariance

Picture by Fredo Durand

A translation invariant filter

Example: The output for the sample n is twice the value of the input at that same time minus the sum of the two previous time steps

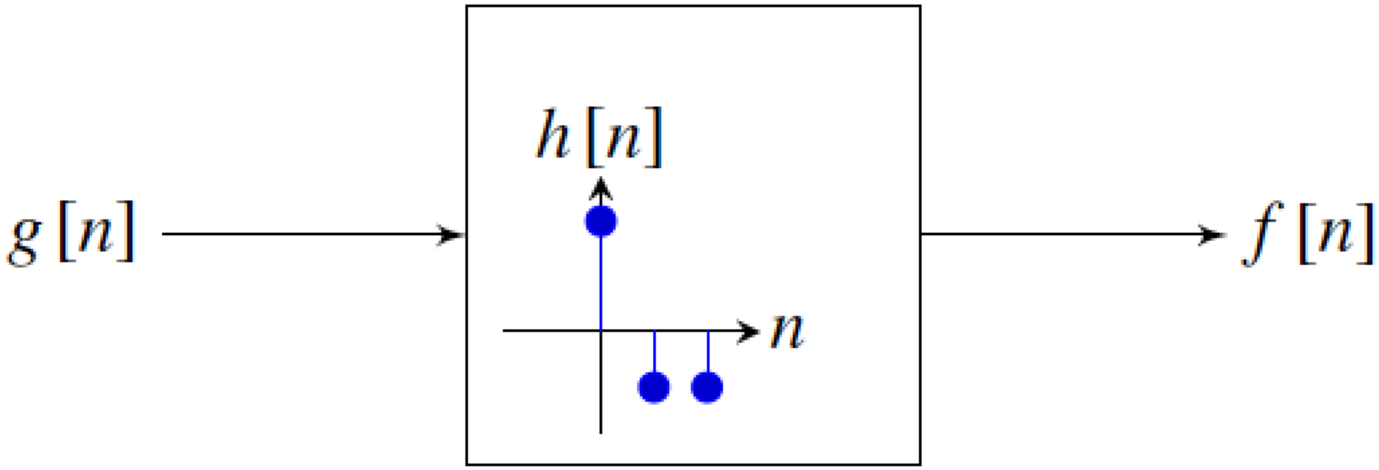
$$f[0] = 2g[0] - g[-1] - g[-2]$$
$$f[1] = 2g[1] - g[0] - g[-1]$$
$$f[2] = 2g[2] - g[1] - g[0]$$
$$\ldots$$
$$f[n] = 2g[n] - g[n-1] - g[n-2]$$

A filter is linear translation invariant (LTI) if it is linear and when we translate the input signal by m samples, the output is also translated by m samples.

Convolution

$$f[n] = h \circ g = \sum_{k=0}^{N-1} h[n-k]\, g[k]$$

For the previous example: h = [2, -1, -1]



Properties of the convolution

Commutative

$$h[n] \circ g[n] = g[n] \circ h[n]$$

Associative

$$h[n] \circ g[n] \circ q[n] = h[n] \circ (g[n] \circ q[n]) = (h[n] \circ g[n]) \circ q[n]$$

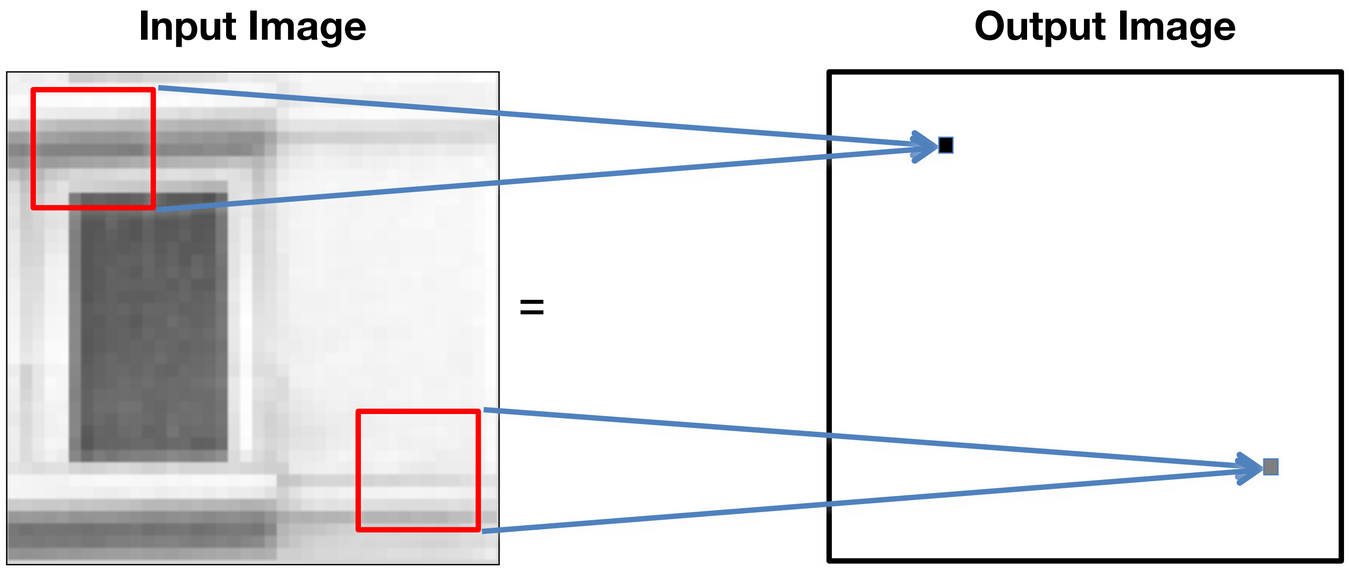
Distributive with respect to the sum

$$h[n] \circ (f[n] + g[n]) = h[n] \circ f[n] + h[n] \circ g[n]$$

Shift property

$$f[n - n_0] = h[n] \circ g[n - n_0] = h[n - n_0] \circ g[n]$$

A translation invariant filter



The same weighting occurs within each window

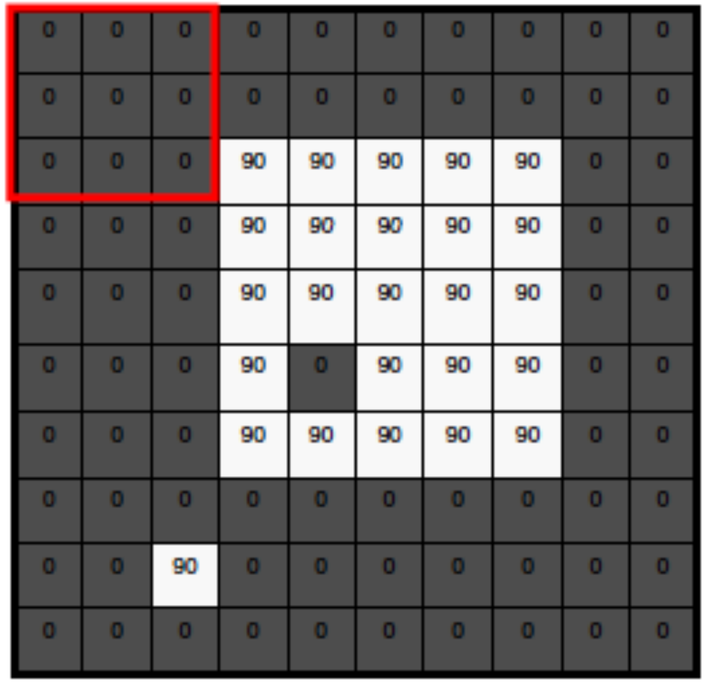
2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

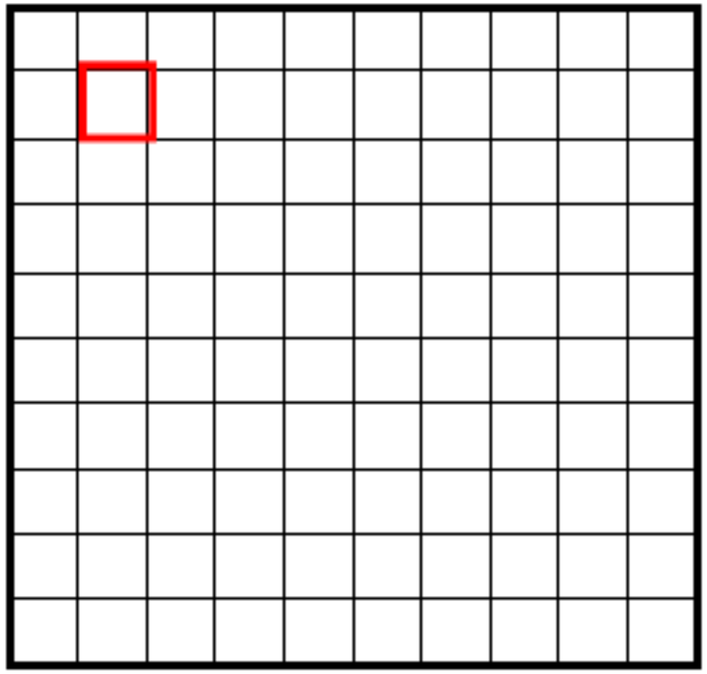
Convolution weights

2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



Input image



Convolution output

2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Input image

	0								

Convolution output

2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Input image

	0	10							

Convolution output

2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Input image

	0	10	20						

Convolution output

2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Input image

	0	10	20	30					

Convolution output

2D convolution

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Input image

	0	10	20	30	30				

Convolution output

What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

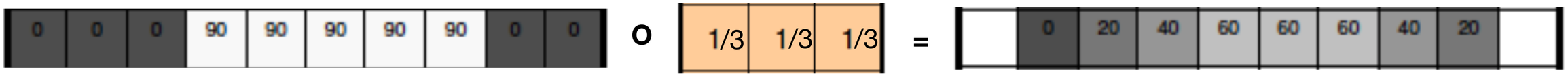
Input image

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Convolution output

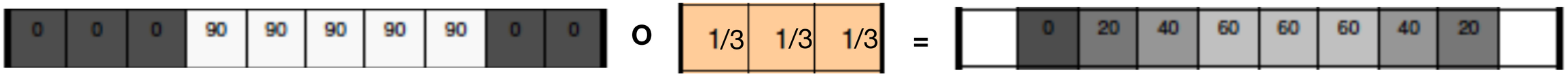
Convolution

In the 1D case, it helps to make explicit the structure of the matrix:



Convolution

In the 1D case, it helps to make explicit the structure of the matrix:

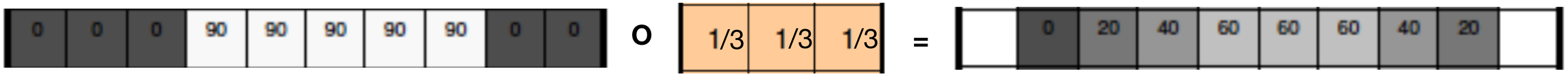


In the 1D case, it helps to make explicit the structure of the matrix:

$$\begin{bmatrix} f[1] \\ f[2] \\ \\ \dots \\ f[8] \end{bmatrix} = \begin{bmatrix} & & & & & & & & & \\ & & & & & & & & & \\ & & & & & & & & & \\ & & & & & & & & & \\ & & & & & & & & & \end{bmatrix} \begin{bmatrix} g[0] \\ g[1] \\ \\ \dots \\ g[9] \end{bmatrix}$$

Convolution

In the 1D case, it helps to make explicit the structure of the matrix:

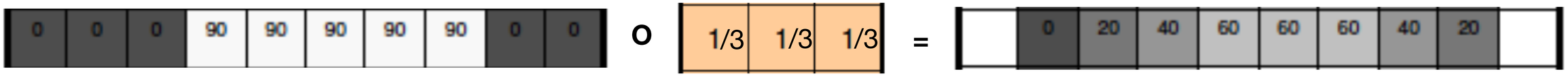


In the 1D case, it helps to make explicit the structure of the matrix:

$$\begin{bmatrix} f[1] \\ f[2] \\ \\ \dots \\ f[8] \end{bmatrix} = \begin{bmatrix} 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & & & \\ & & & & & & & & & \\ & & & & & & & & & \\ & & & & & & & & & \\ & & & & & & & & & \end{bmatrix} \begin{bmatrix} g[0] \\ g[1] \\ \\ \dots \\ g[9] \end{bmatrix}$$

Convolution

In the 1D case, it helps to make explicit the structure of the matrix:

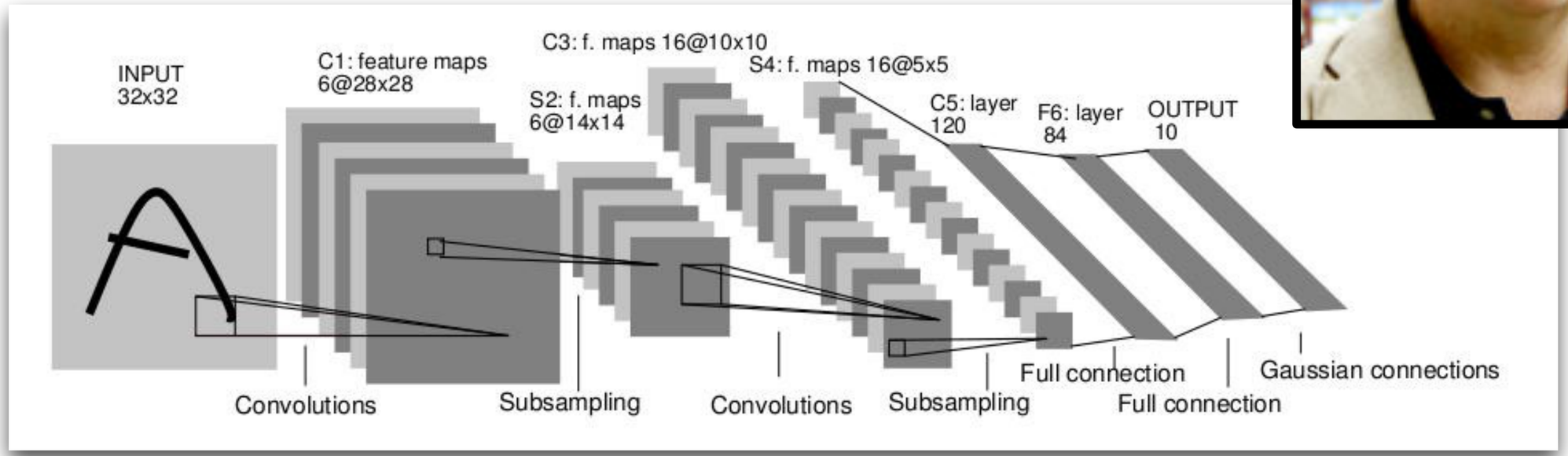


In the 1D case, it helps to make explicit the structure of the matrix:

$$
\begin{bmatrix} f[1] \\ f[2] \\ \\ \dots \\ f[8] \end{bmatrix}
=
\begin{bmatrix}
1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1/3 & 1/3 & 1/3
\end{bmatrix}
\begin{bmatrix} g[0] \\ g[1] \\ \\ \dots \\ g[9] \end{bmatrix}
$$

Convolutional Neural Networks

- Neural network with specialized connectivity structure



LeCun et al. 1989

Rectangular filter

g[m,n]

$\otimes$

h[m,n]

=

f[m,n]

Rectangular filter

g[m,n]

$\otimes$

h[m,n]

=

f[m,n]

Rectangular filter

g[m,n]

$\otimes$

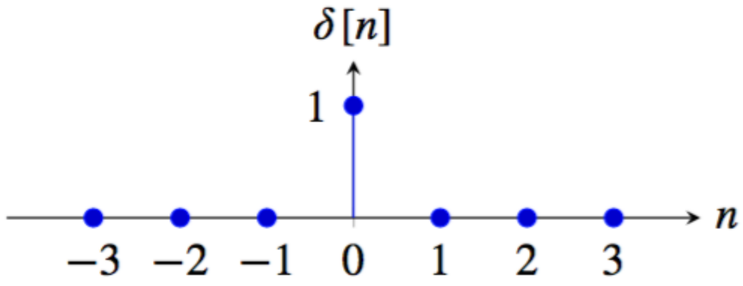
h[m,n]

=

f[m,n]

Important signals

The impulse



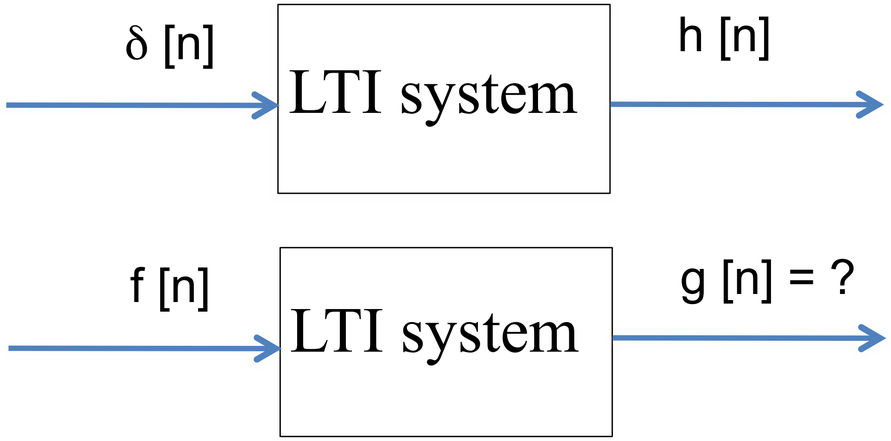
The result of convolving a signal g[n] with the impulse signal is the same signal:

$$f[n] = \delta \circ g = \sum_k \delta[n-k]\, g[k] = g[n]$$

Convolving a signal f with a translated impulse $\delta\,[n - n_0]$ results in a translated signal:

$$f[n - n_0] = \delta[n - n_0] \circ f[n]$$

Why the impulse is so important



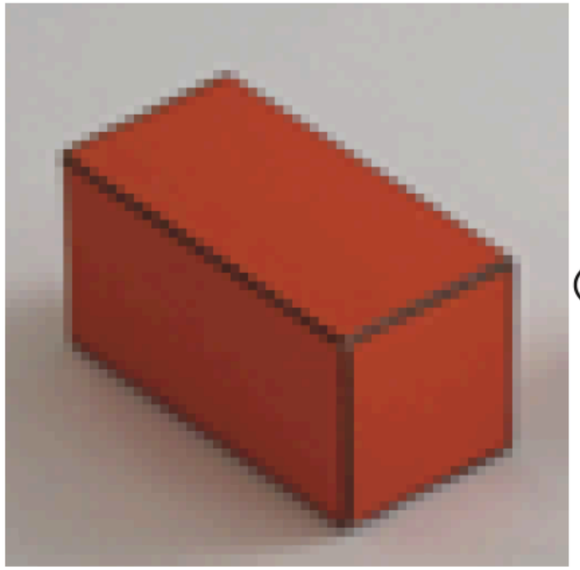
$$f[n] = \sum_k f[k]\delta[n-k]$$

Write the input signal as a sum of impulses

$$g[n] = \sum_k f[k]h[n-k] = f \circ h = h \circ f$$

Then the output of an LTI system is the corresponding sum of impulse responses

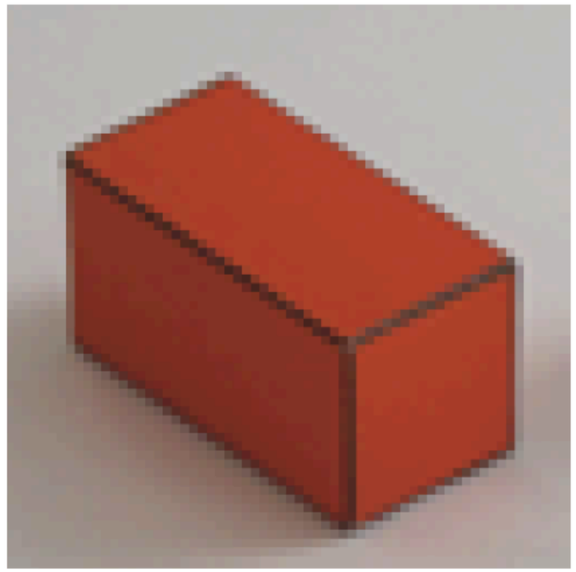
Examples



$\circ$

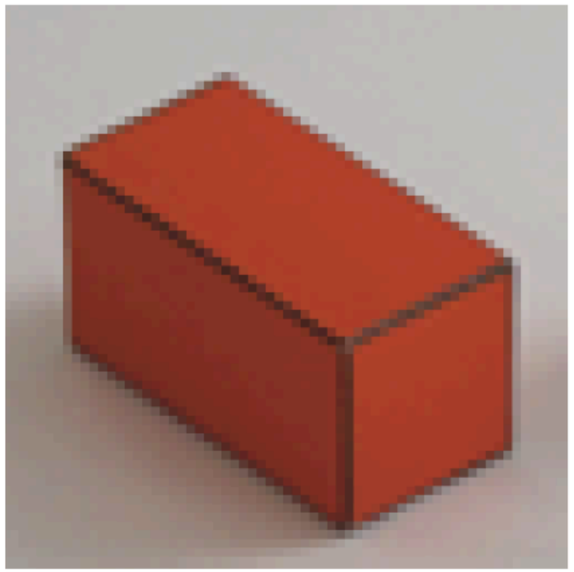
0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

$=$



The impulse signal

Examples

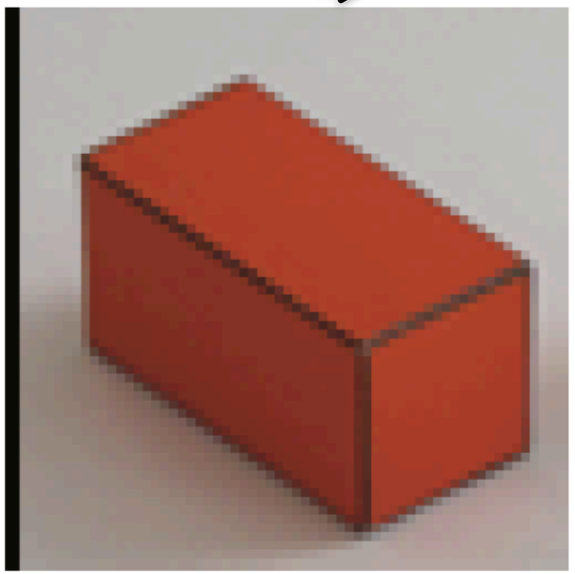


$\circ$

0	0	0	0	0
0	0	0	0	0
0	0	0	0	1
0	0	0	0	0
0	0	0	0	0

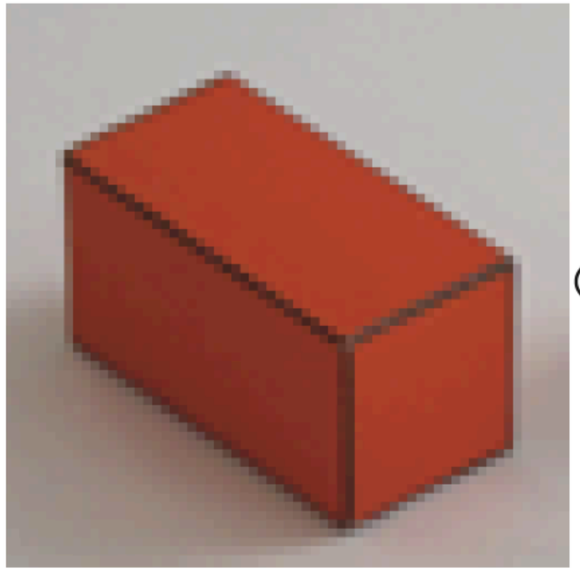
$=$

2 pixels



(using zero padding)

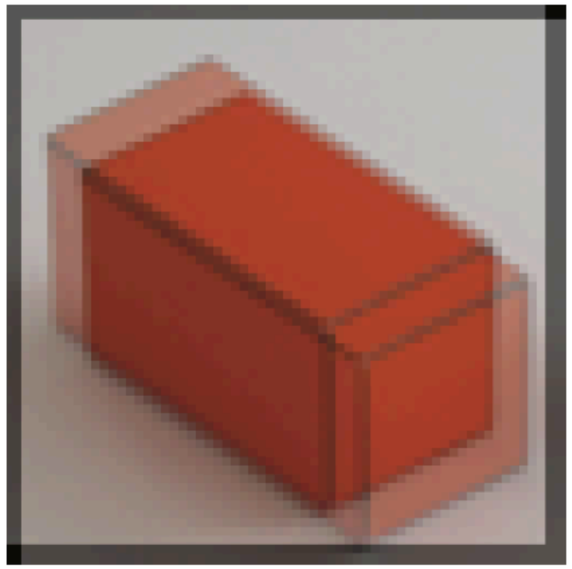
Examples



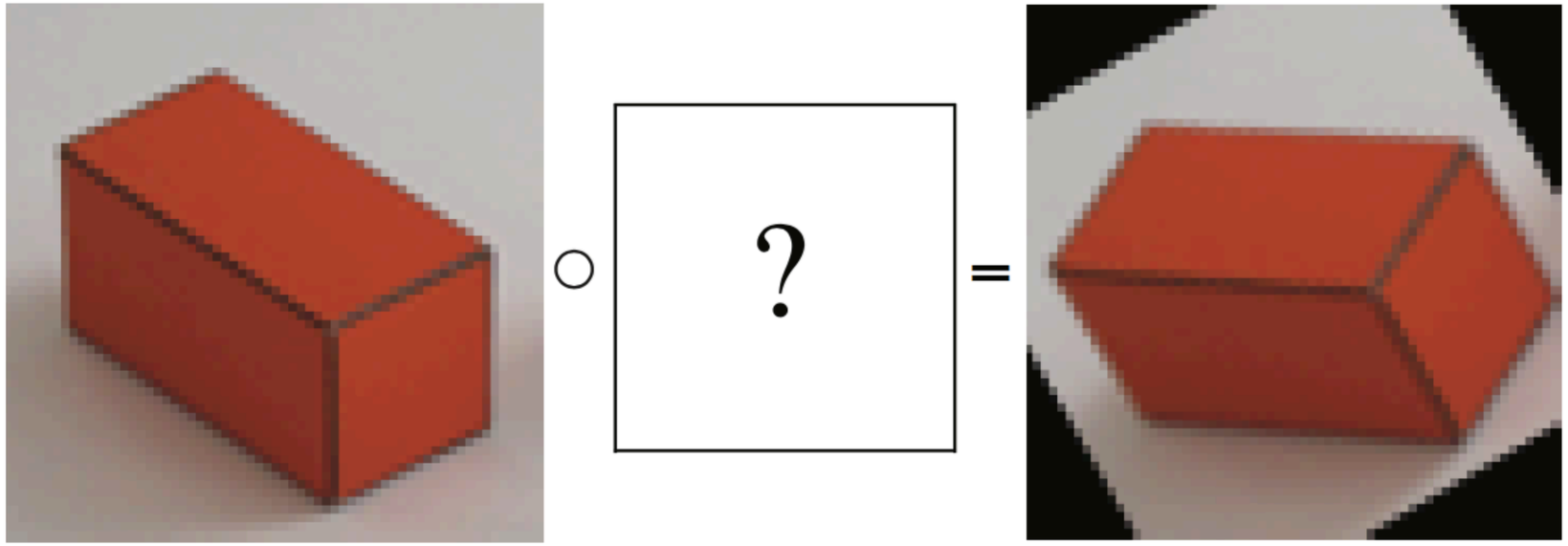
$\bigcirc$

.5	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	.5

$=$



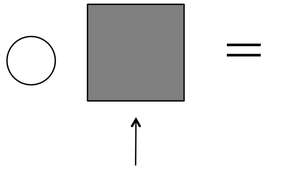
Examples

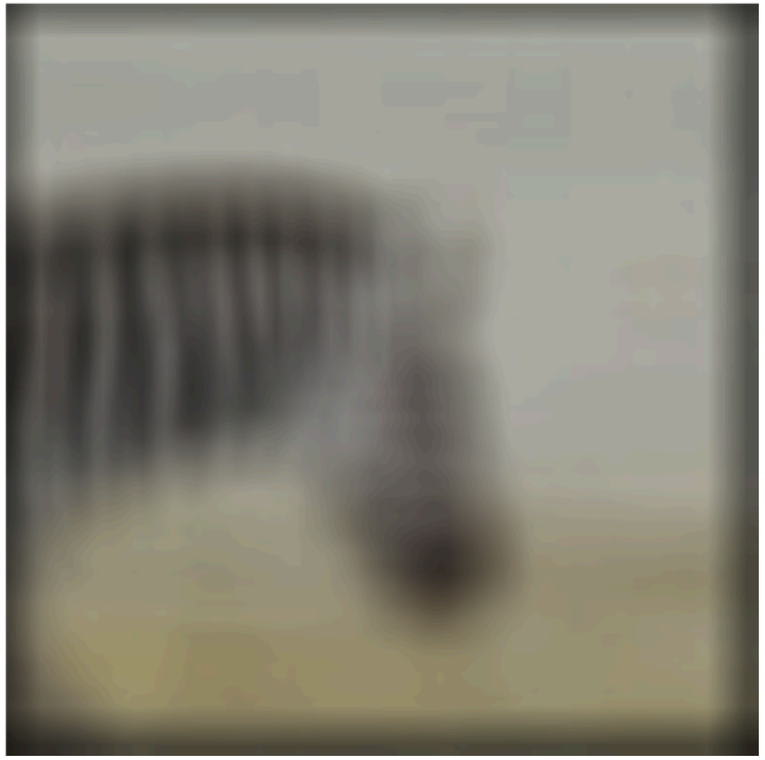


Handling boundaries

Handling boundaries

Zero padding





Handling boundaries

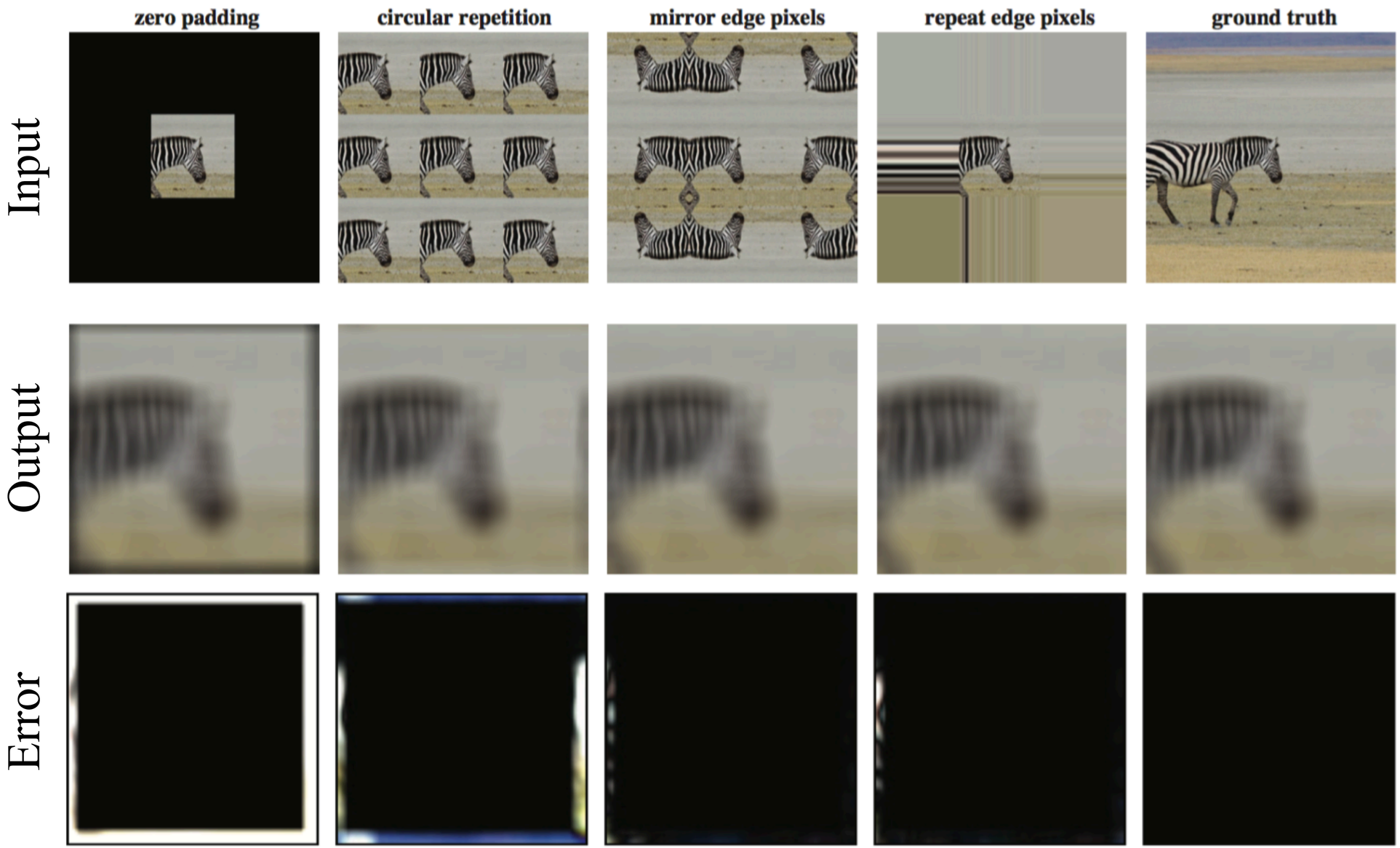
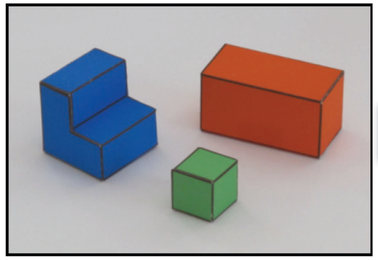
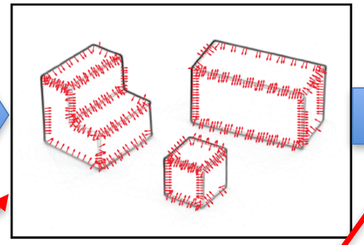


Image transformations

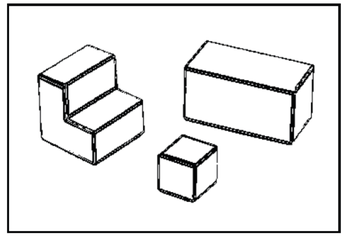
Input image



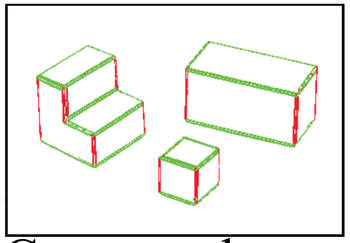
Edge normals



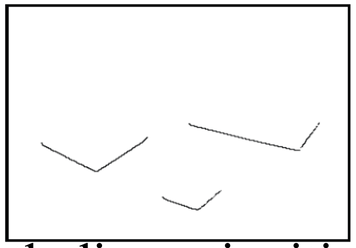
Edge strength



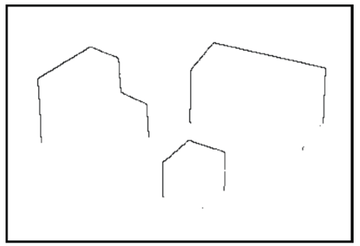
3D orientation

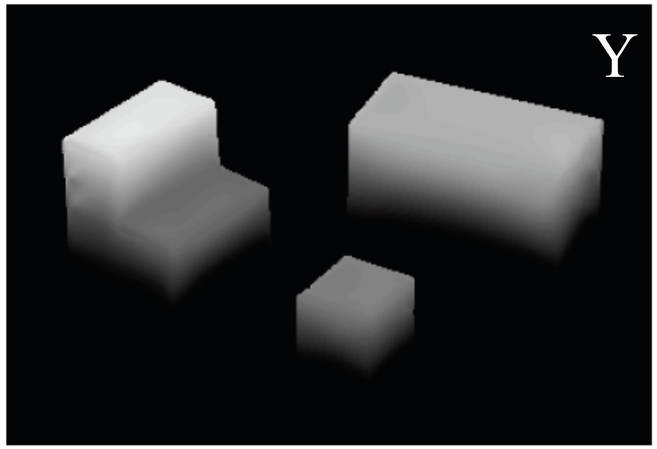


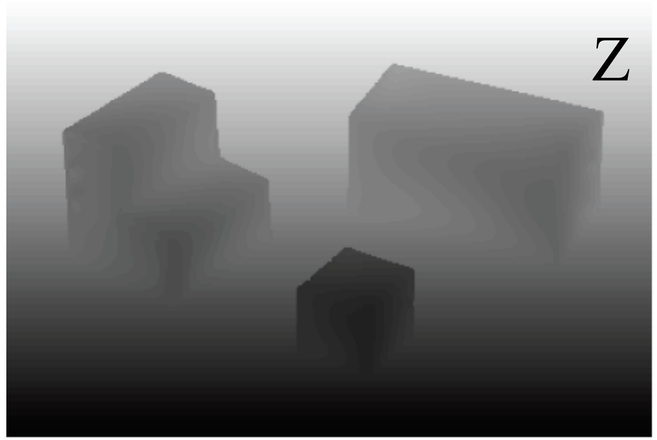
Contact edges



Depth discontinuities







1. From pixels to edges

2. From edges to geometric primitives

Linear image transformations

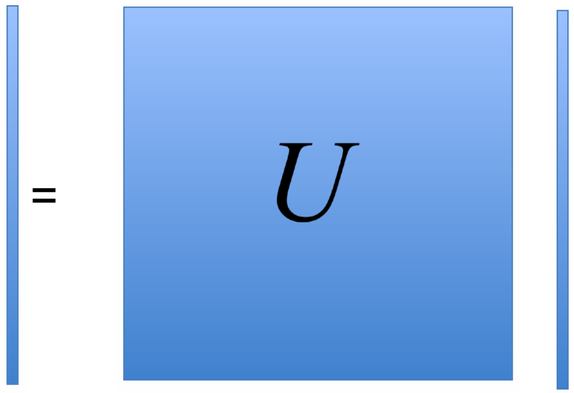
In analyzing images, it's often useful to make a change of basis.

Transformed image

$$\vec{F} = U\vec{f}$$

Vectorized image

Fourier transform, or
Wavelet transform, or
Steerable pyramid transform



The Discrete Fourier transform

Discrete Fourier Transform (DFT) transforms a signal *f* [*n*] into *F* [*u*] as:

$$F[u] = \sum_{n=0}^{N-1} f[n] \exp\left(-2\pi j \frac{un}{N}\right)$$

The inverse of the DFT is:

$$f[n] = \frac{1}{N} \sum_{u=0}^{N-1} F[u] \exp\left(2\pi j \frac{un}{N}\right)$$

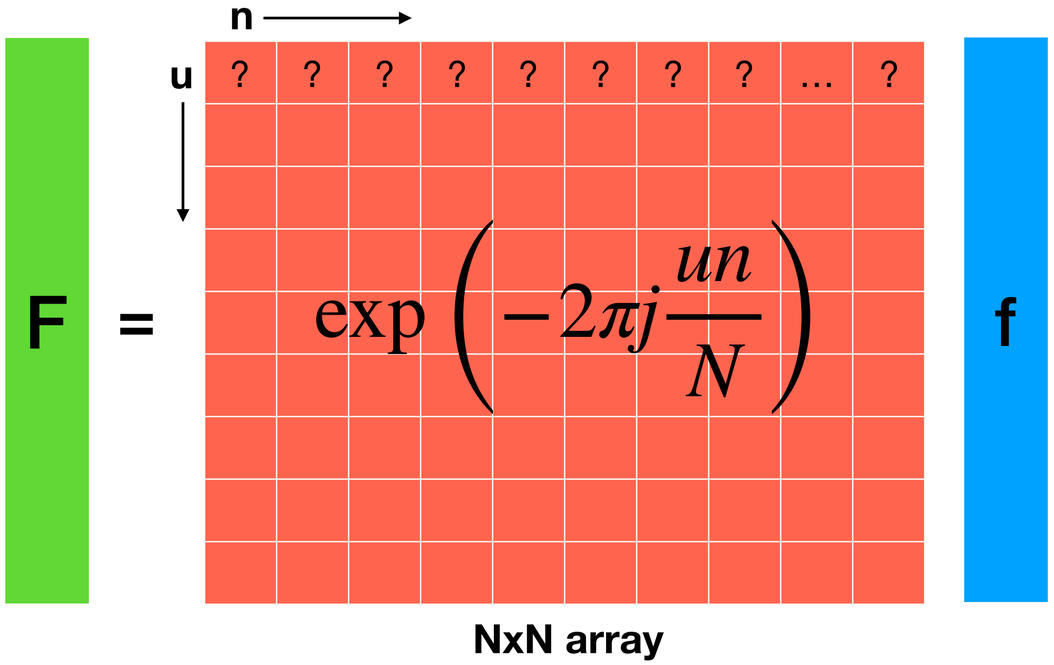
The signal *f* [*n*] is a weighted linear combination of complex exponentials with weights *F* [*u*]

The Discrete Fourier transform

Discrete Fourier Transform (DFT) transforms a signal *f* [*n*] into *F* [*u*] as:

$$F[u] = \sum_{n=0}^{N-1} f[n] \exp\left(-2\pi j \frac{un}{N}\right)$$

Discrete Fourier Transform (DFT) is a linear operator. Therefore, we can write:



Lets visualize the transform coefficients

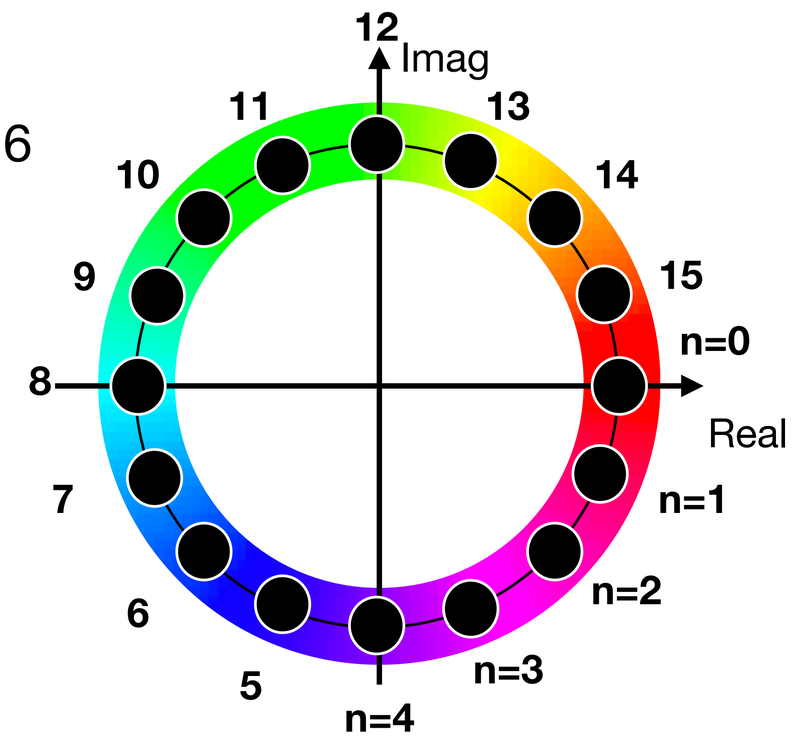
Visualizing the Fourier transform

$$F[u] = \sum_{n=0}^{N-1} f[n] \exp\left(-2\pi j \frac{un}{N}\right)$$

$$\exp(\alpha j) = \cos(\alpha) + j \sin(\alpha)$$

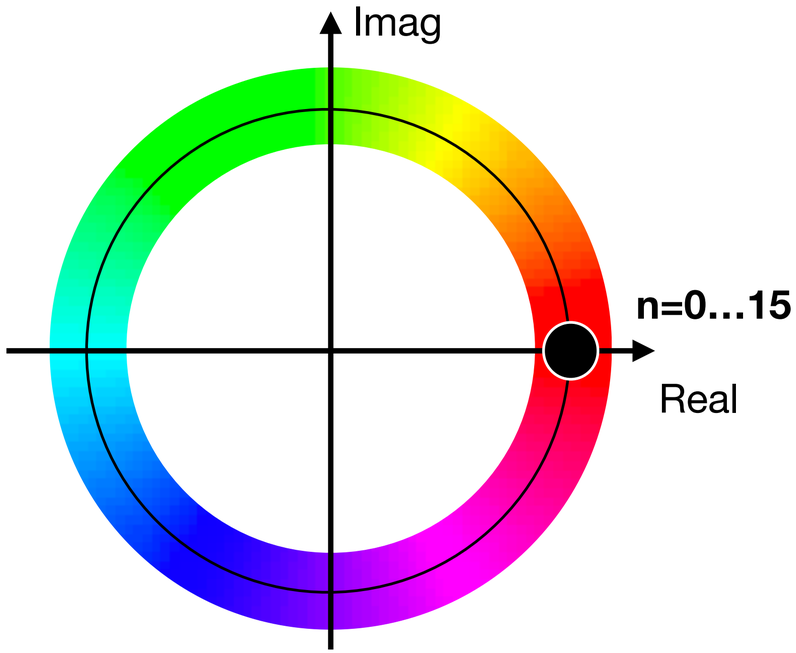
$$\cos\left(2\pi \frac{un}{N}\right) - j \sin\left(2\pi \frac{un}{N}\right)$$

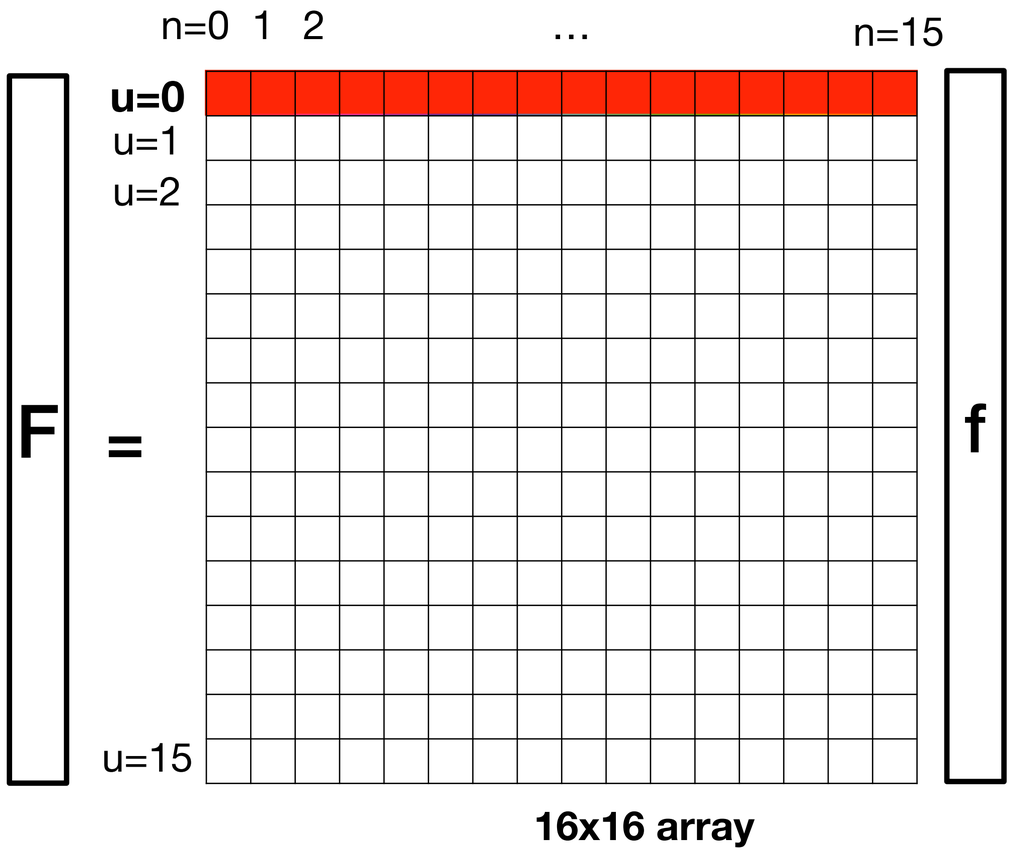
For:
u=1
N=16



Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$ For N=16

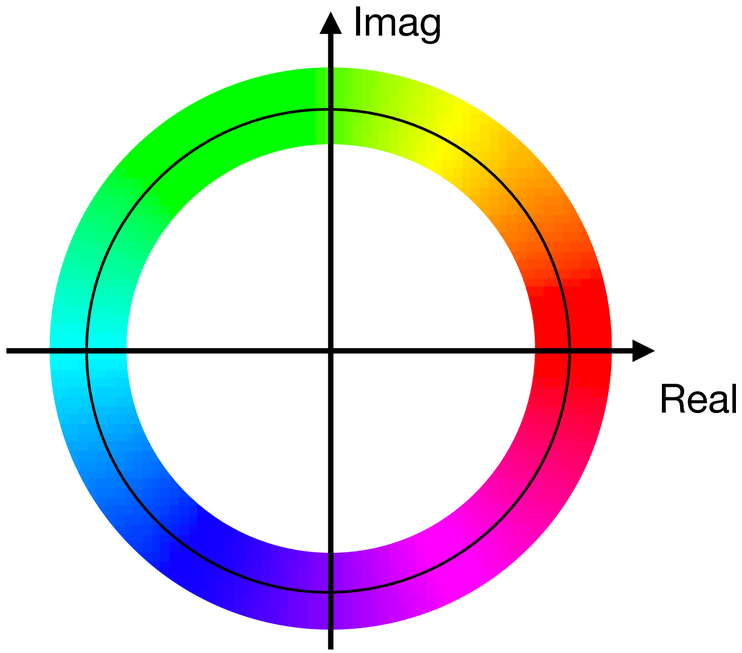


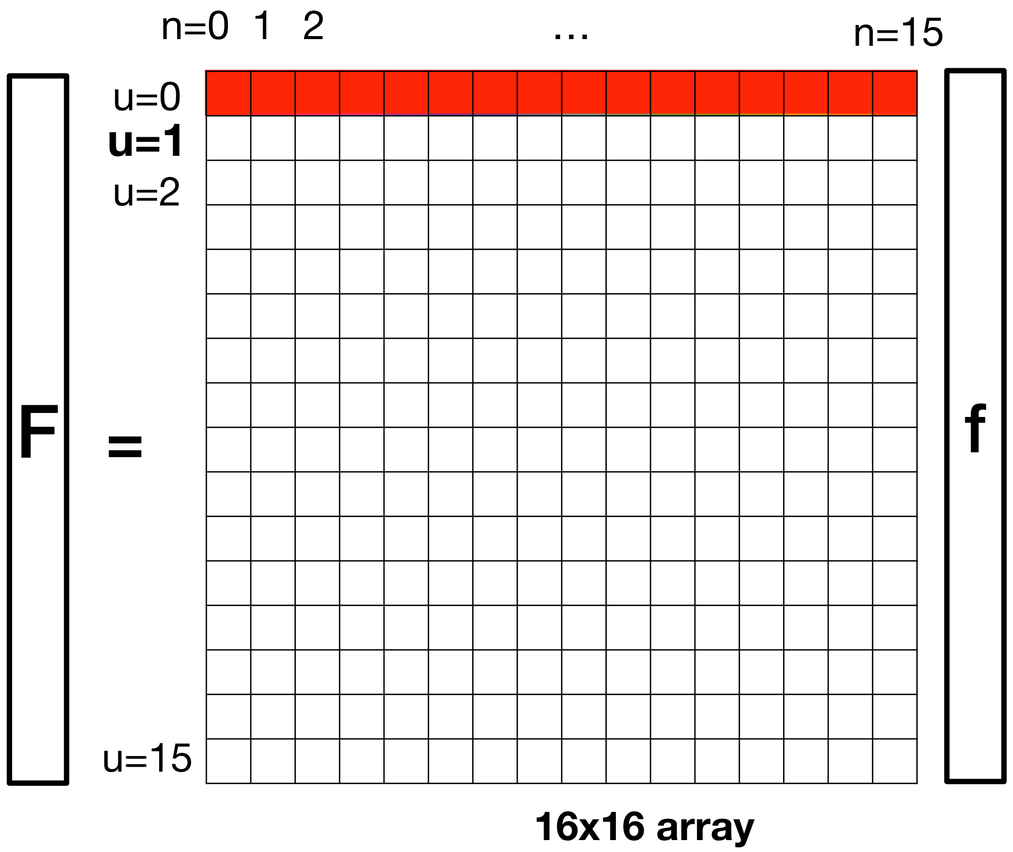


Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$

For N=16

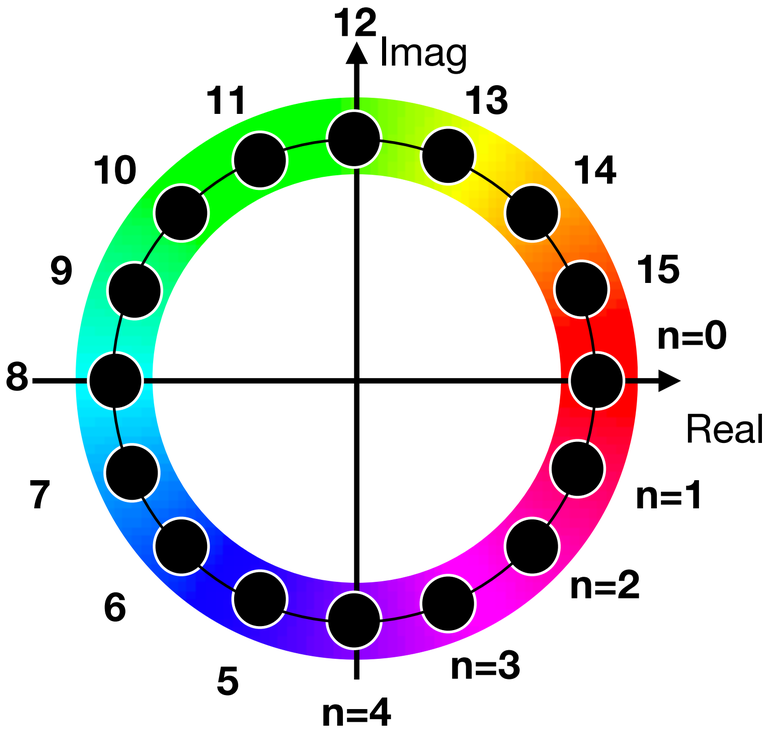


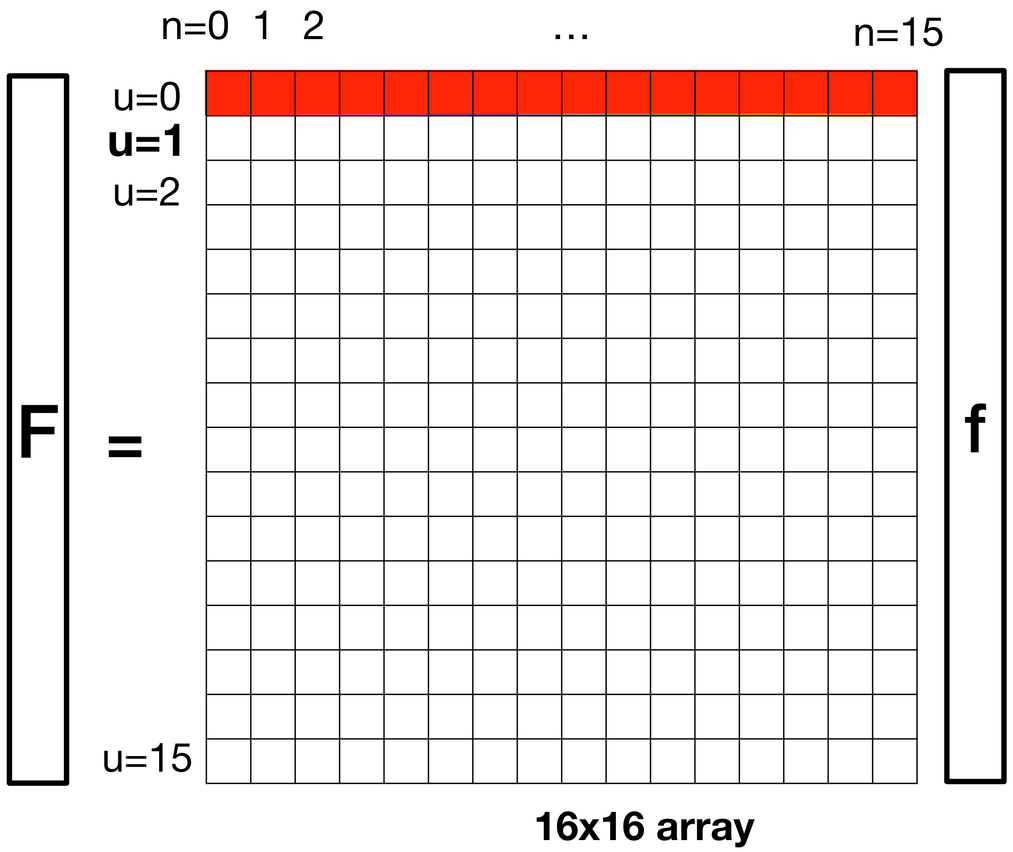


Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$

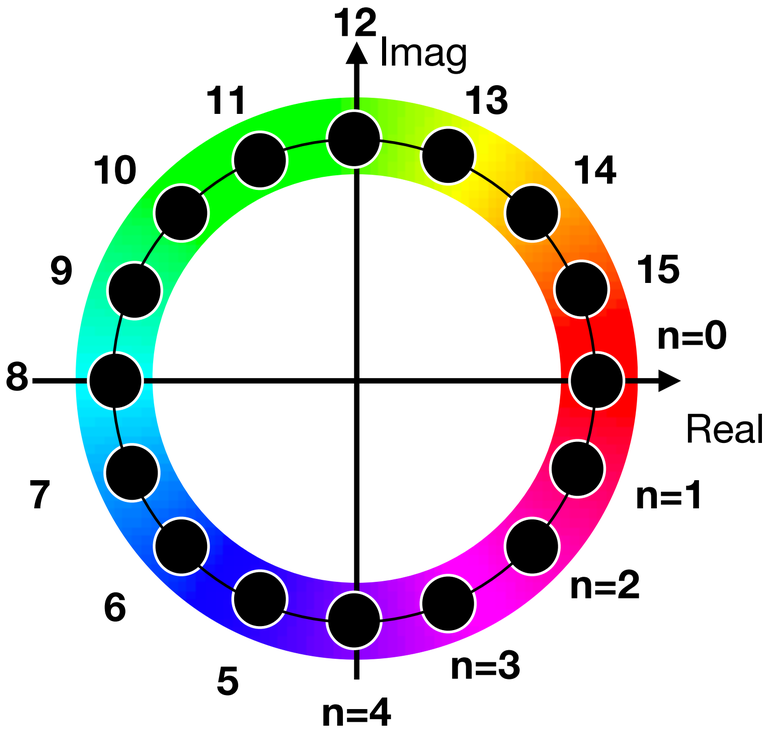
For N=16

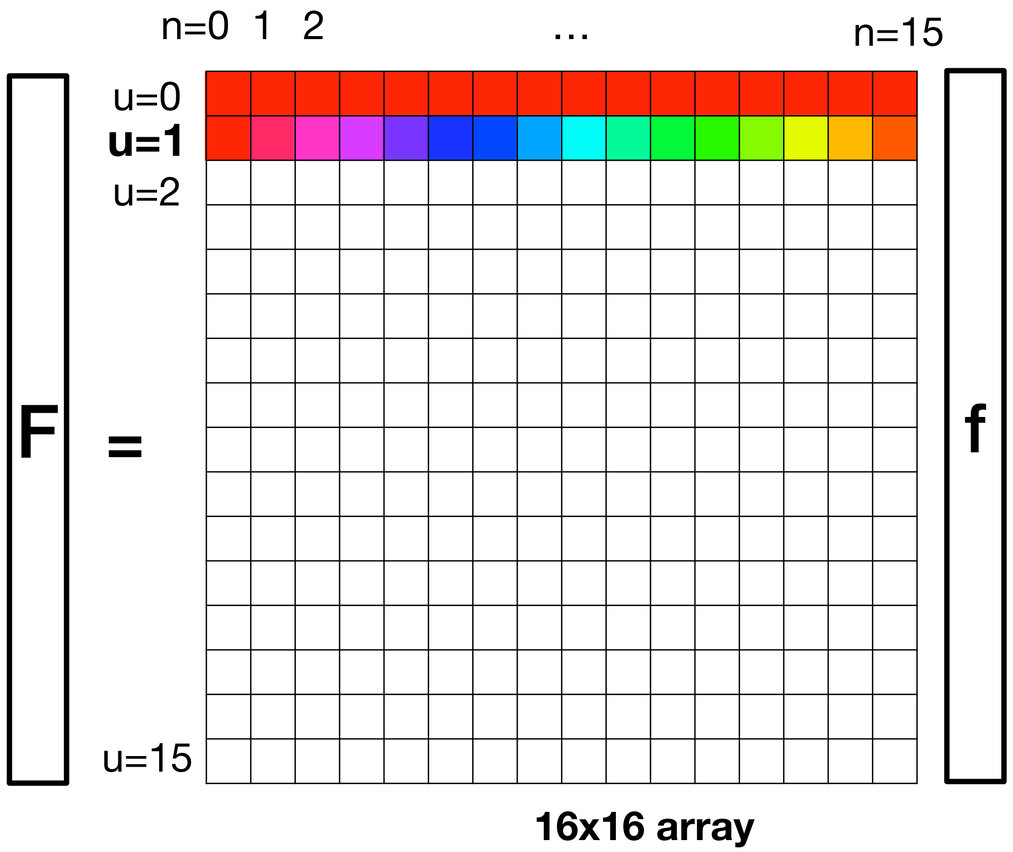




Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$ For N=16

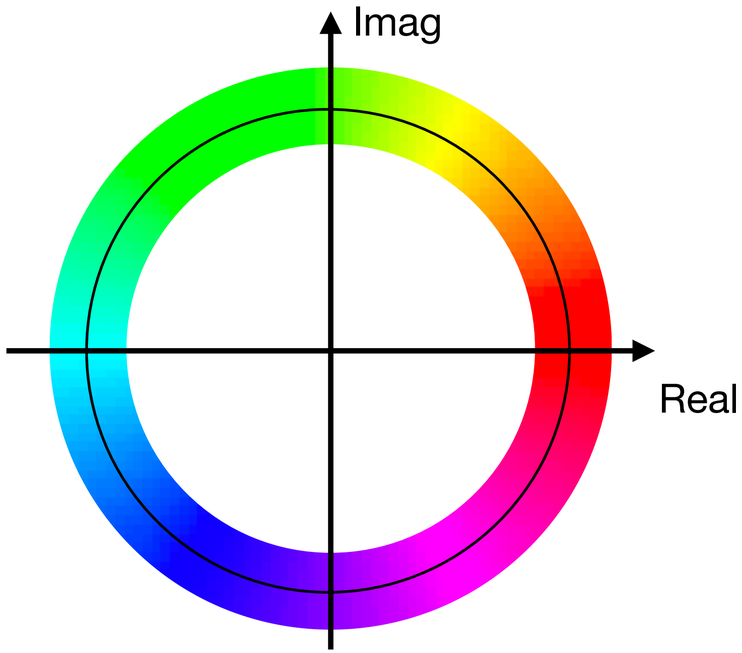


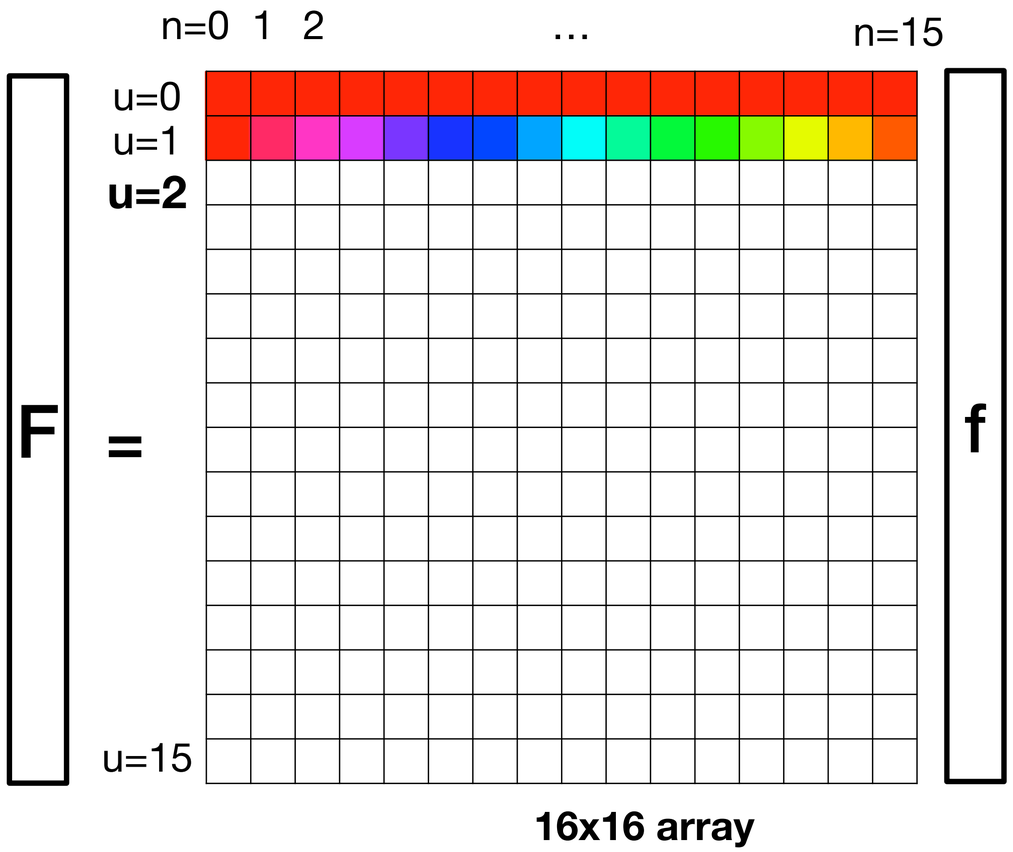


Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$

For N=16

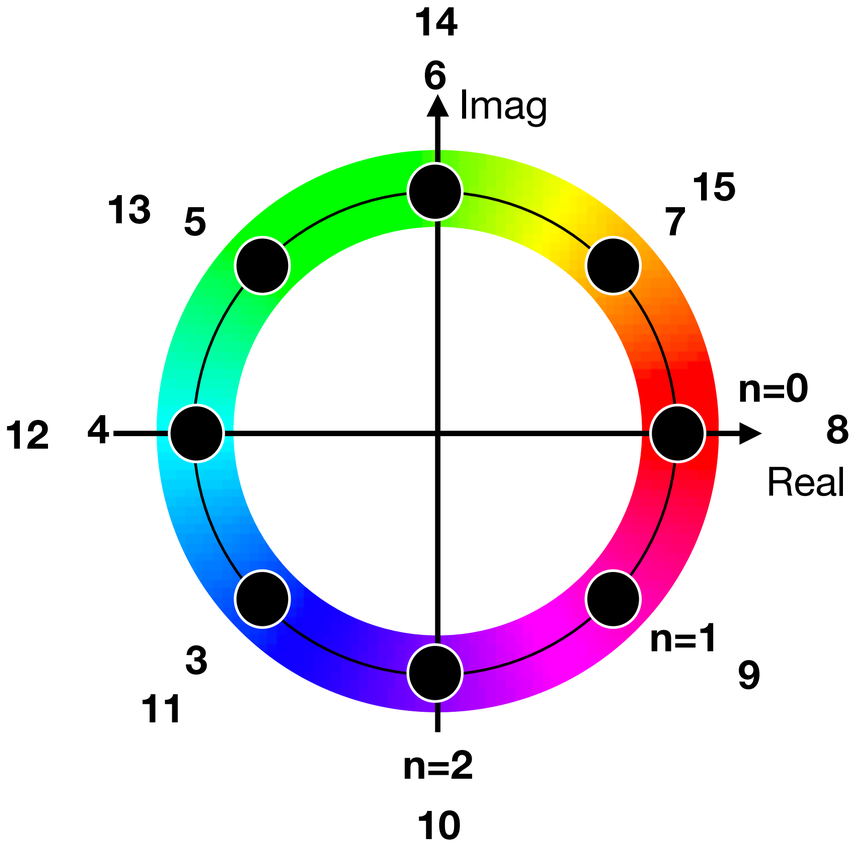


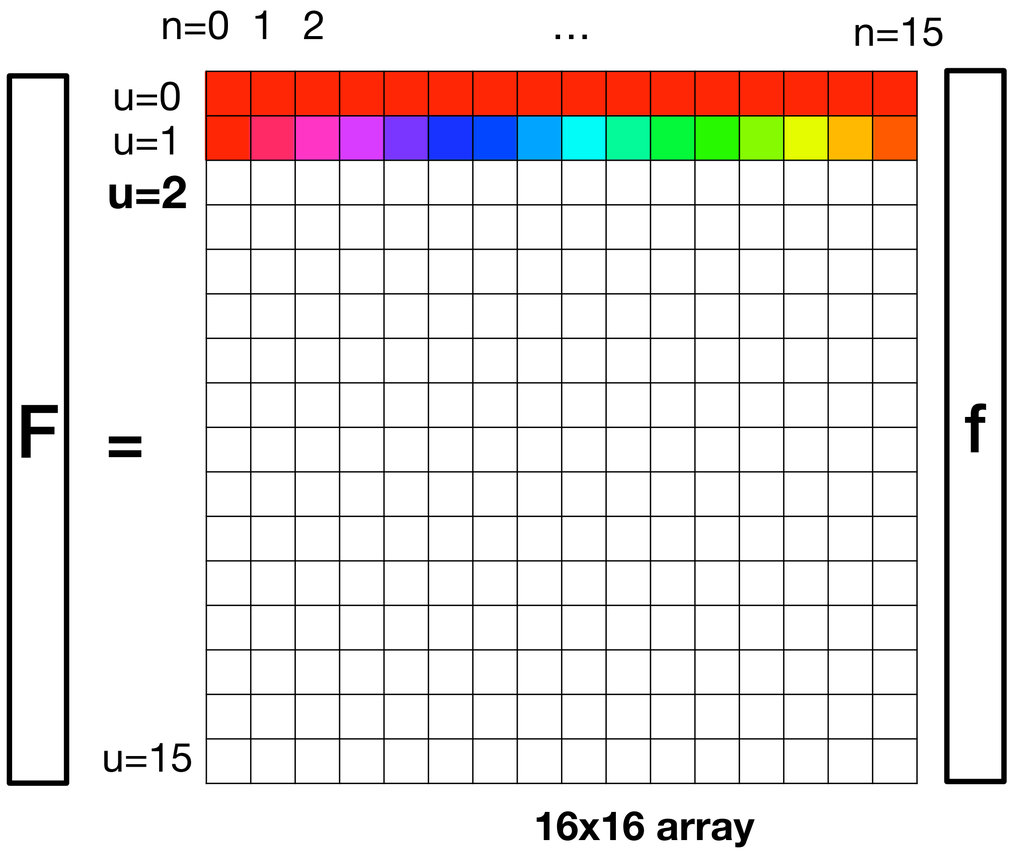


Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$

For N=16

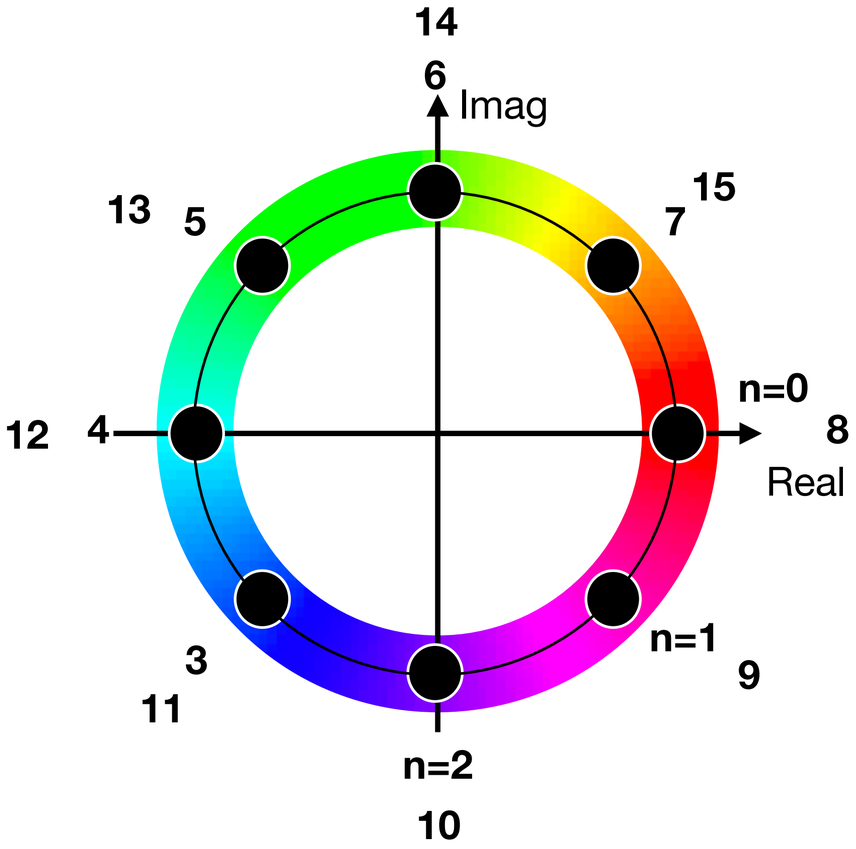


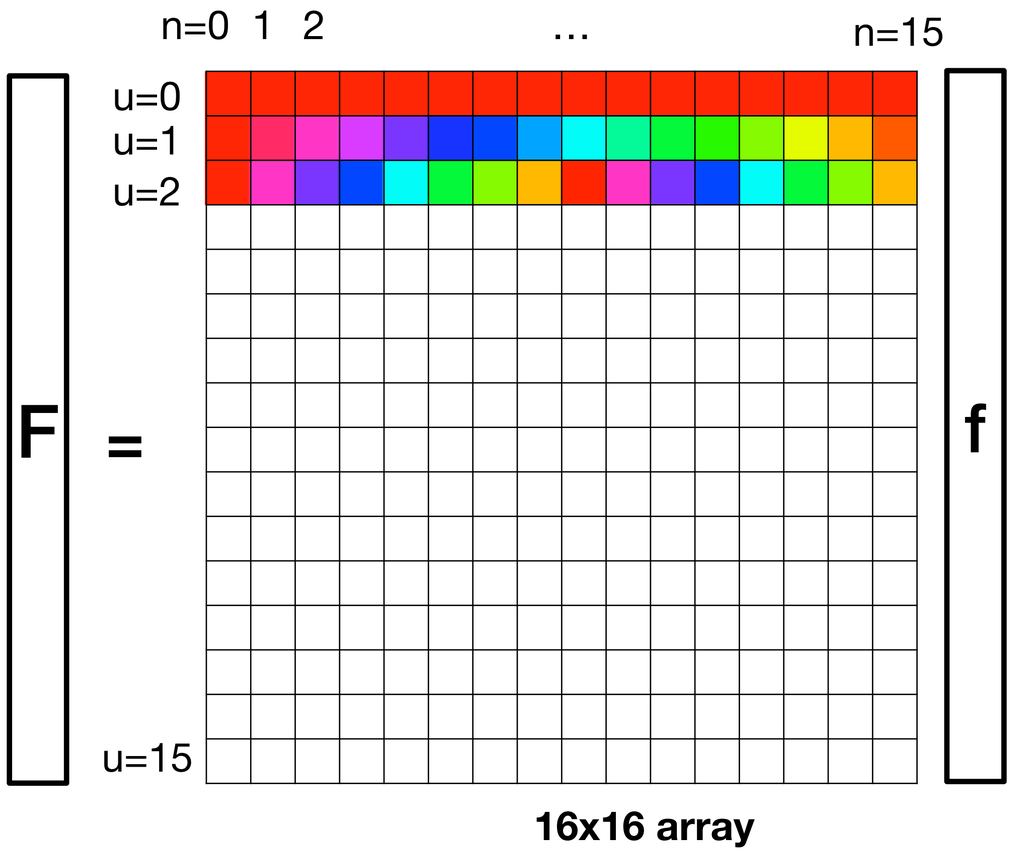


Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$

For N=16

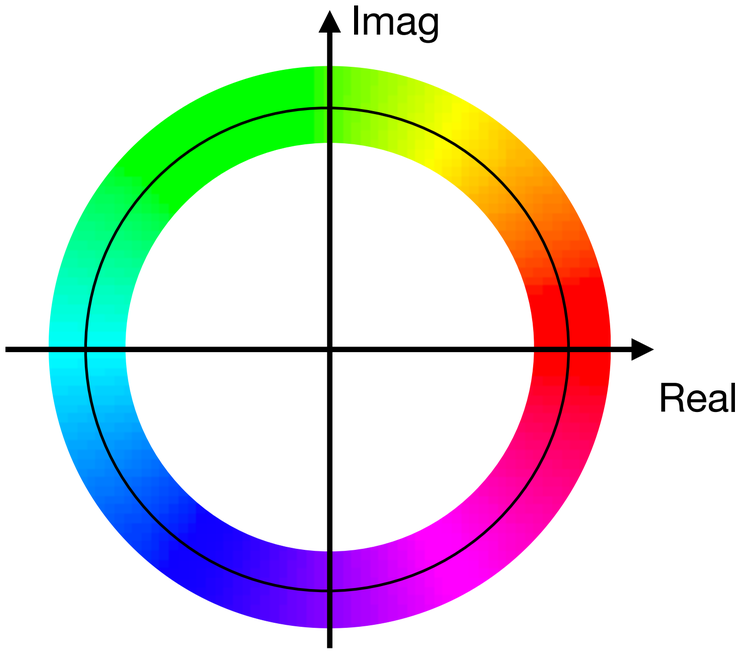


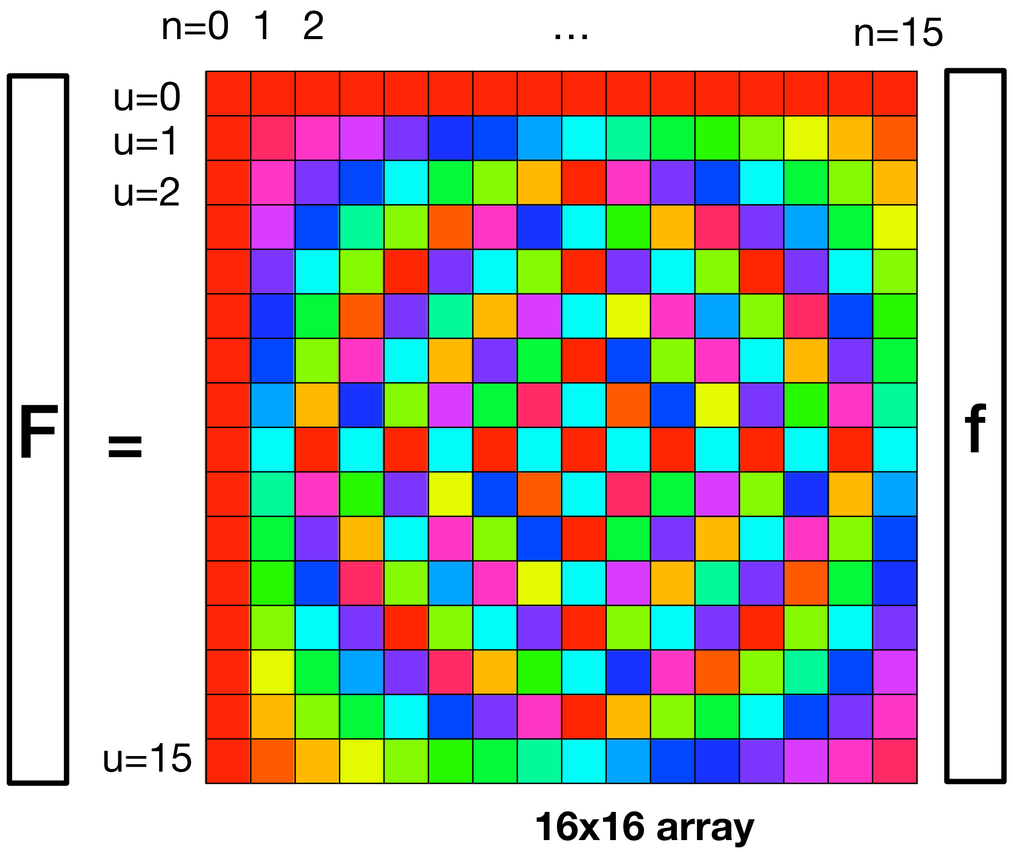


Visualizing the transform coefficients

$$\exp\left(-2\pi j\frac{un}{N}\right)$$

For N=16

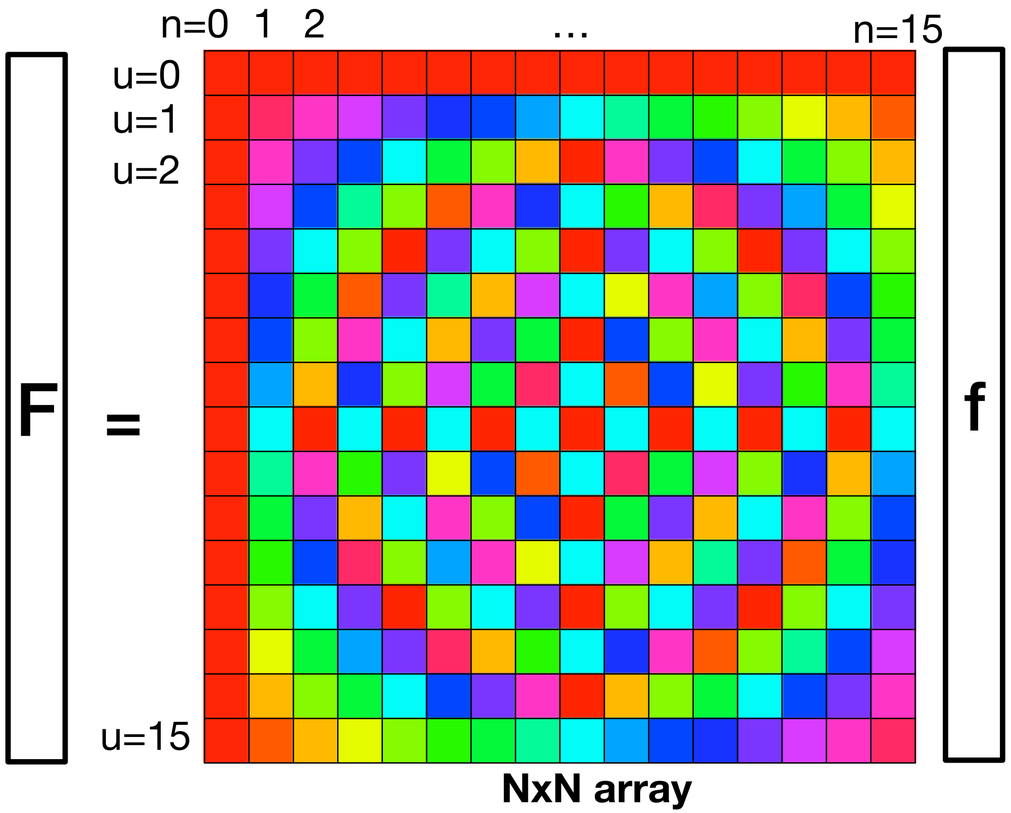




The inverse of the Discrete Fourier transform

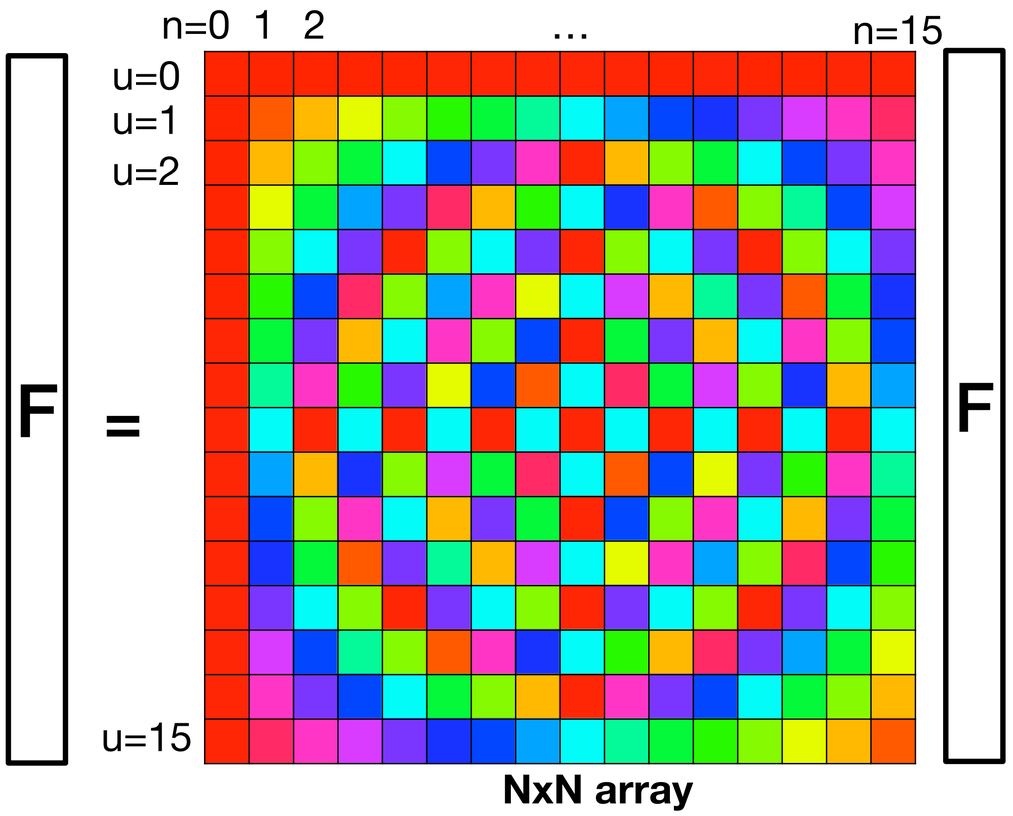
Discrete Fourier Transform (DFT):

$$F[u] = \sum_{n=0}^{N-1} f[n] \exp\left(-2\pi j \frac{un}{N}\right)$$



Its inverse:

$$f[n] = \frac{1}{N} \sum_{u=0}^{N-1} F[u] \exp\left(2\pi j \frac{un}{N}\right)$$



For images, the 2D DFT

1D Discrete Fourier Transform (DFT) transforms a signal *f* [*n*] into *F* [*u*] as:

$$F[u] = \sum_{n=0}^{N-1} f[n] \exp\left(-2\pi j \frac{un}{N}\right)$$

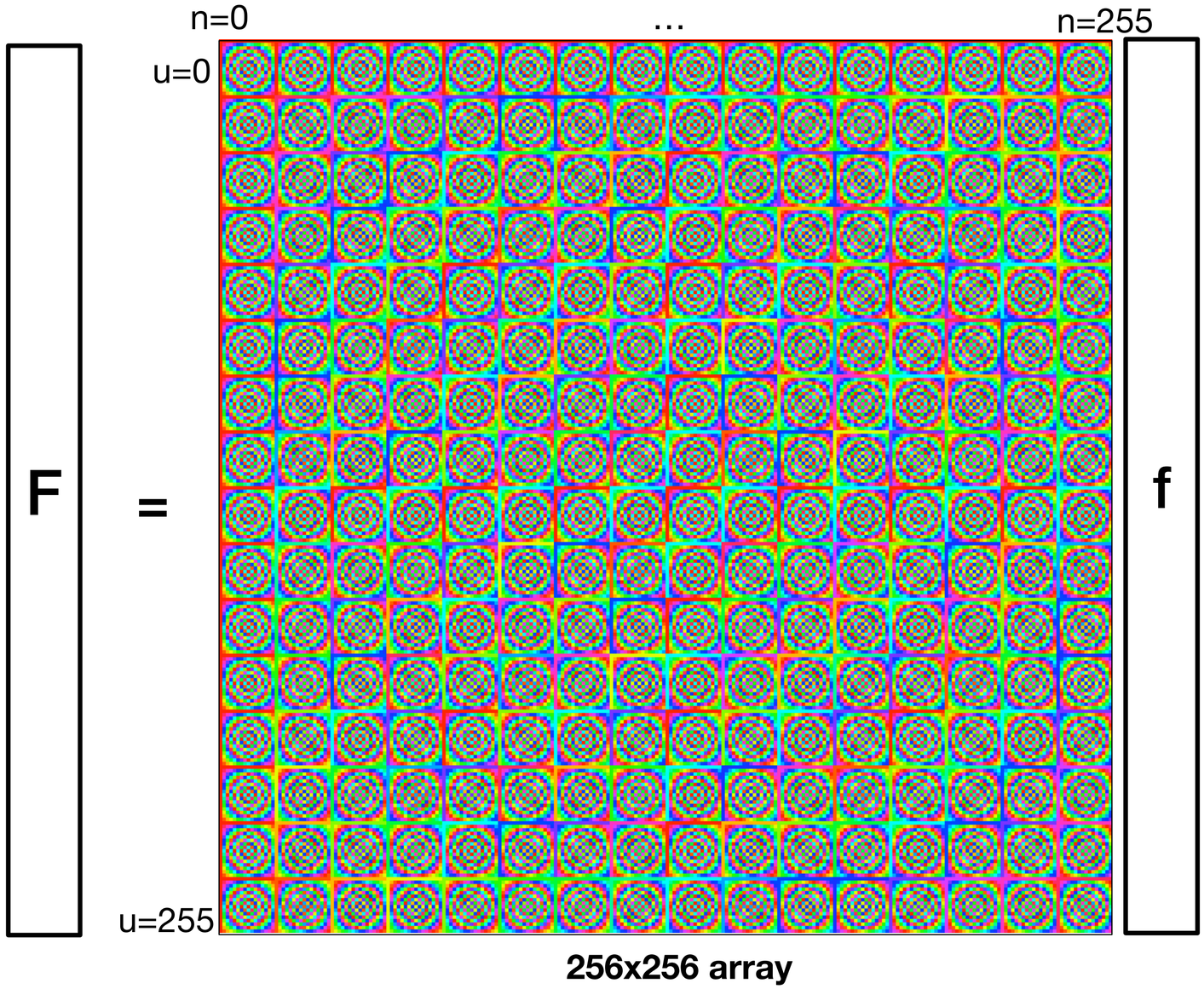
2D Discrete Fourier Transform (DFT) transforms an image *f* [*n,m*] into *F* [*u,v*] as:

$$F[u,v] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

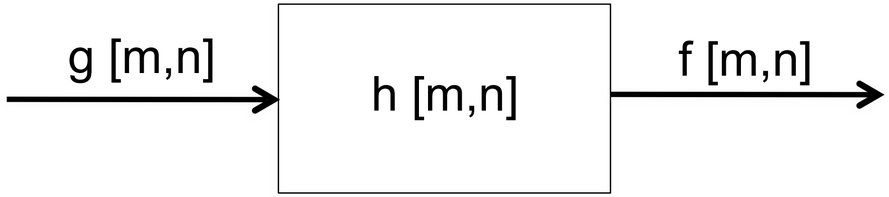
Visualizing the 2D DFT coefficients

$$F[u,v] = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j\left(\frac{un}{N}+\frac{vm}{M}\right)\right)$$

For N=M=16



A remarkable property of Fourier transform



In the spatial domain, the output of f is the convolution:

$$f[m,n] = h \circ g = \sum_{k,l} h[m-k, n-l]\, g[k,l]$$

In the frequency domain:

$$F[u,v] = G[u,v]\, H[u,v]$$

Terminology:

Impulse response: h [m,n]

Transfer function: $H[u,v]$

Dual convolution property

The Fourier transform of the convolution is the product of Fourier transforms

$$f[m,n] = h \circ g \quad \longleftrightarrow \quad F[u,v] = G[u,v]\,H[u,v]$$

The Fourier transform of the product is the convolution of Fourier transforms

$$f[n,m] = g[n,m]\,h[n,m] \quad \longleftrightarrow \quad F[u,v] = \frac{1}{NM} G[u,v] \circ H[u,v]$$

Visualizing the image Fourier transform

$$F[u,v] = \sum_{n=0}^{N-1}\sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j\left(\frac{un}{N}+\frac{vm}{M}\right)\right)$$

The values of $F[u,v]$ are complex.

Using the real and imaginary components:

$$F[u,v] = Re\{F[u,v]\} + j\,Imag\{F[u,v]\}$$

Or using a polar decomposition:

$$F[u,v] = A[u,v]\exp(j\theta[u,v])$$

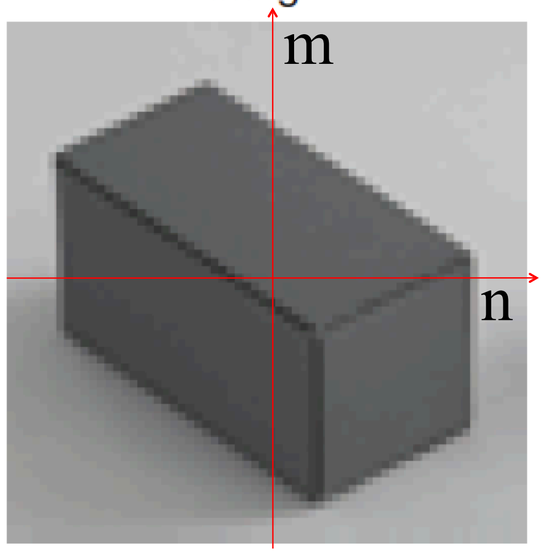
Amplitude

Phase

Visualizing the image Fourier transform

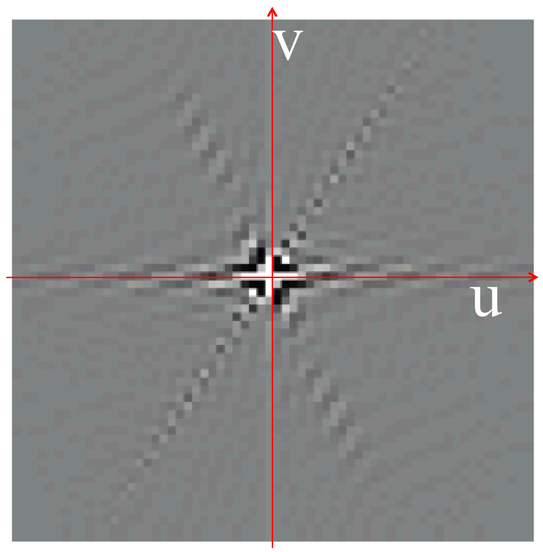
$f[n,m]$

image

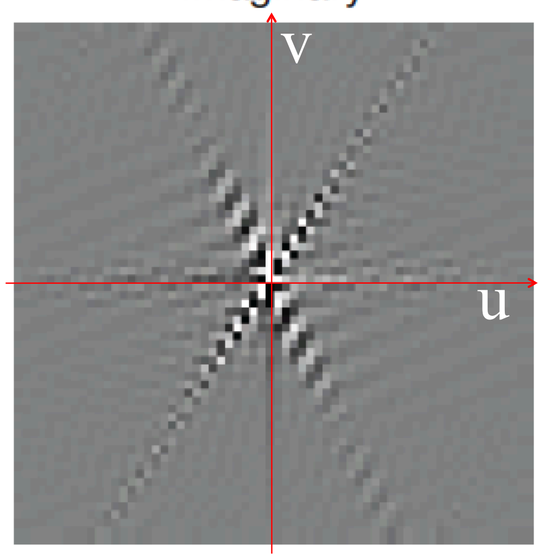


$F[u,v]$

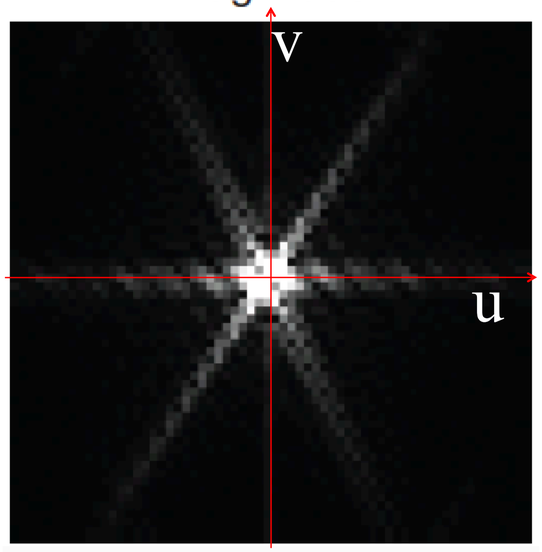
real



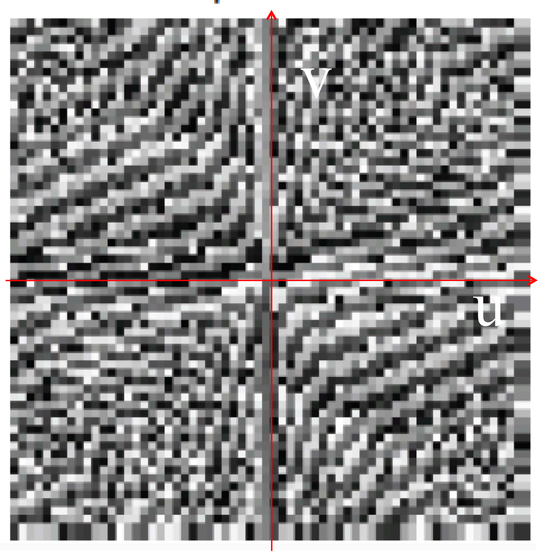
imaginary



magnitude

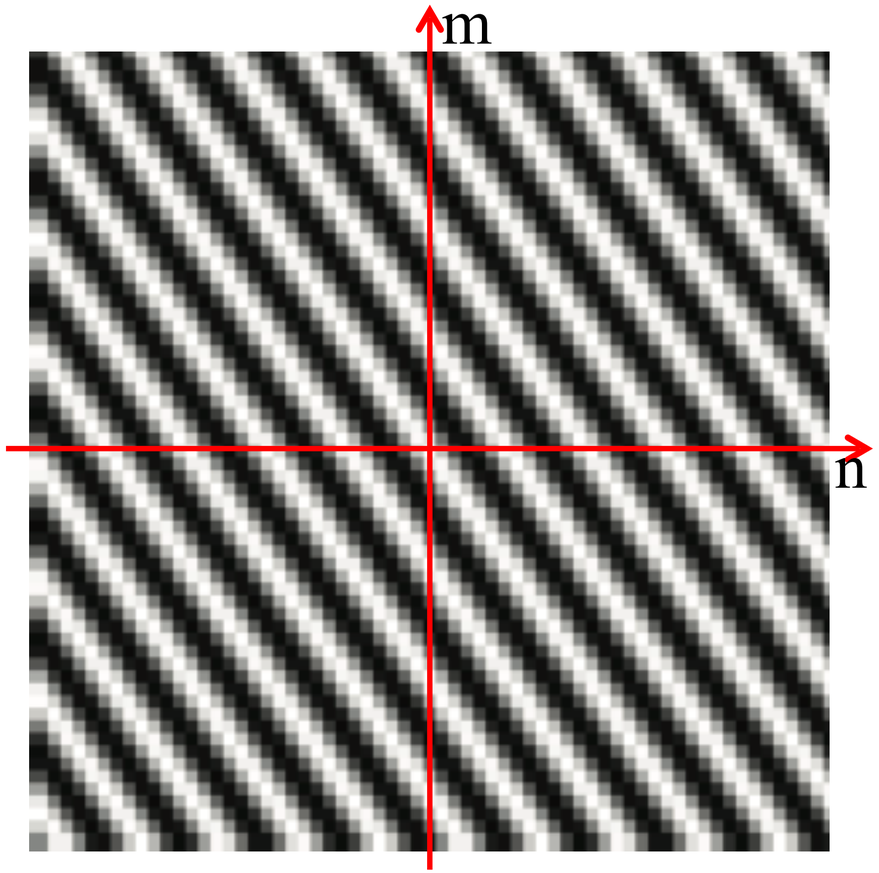


phase



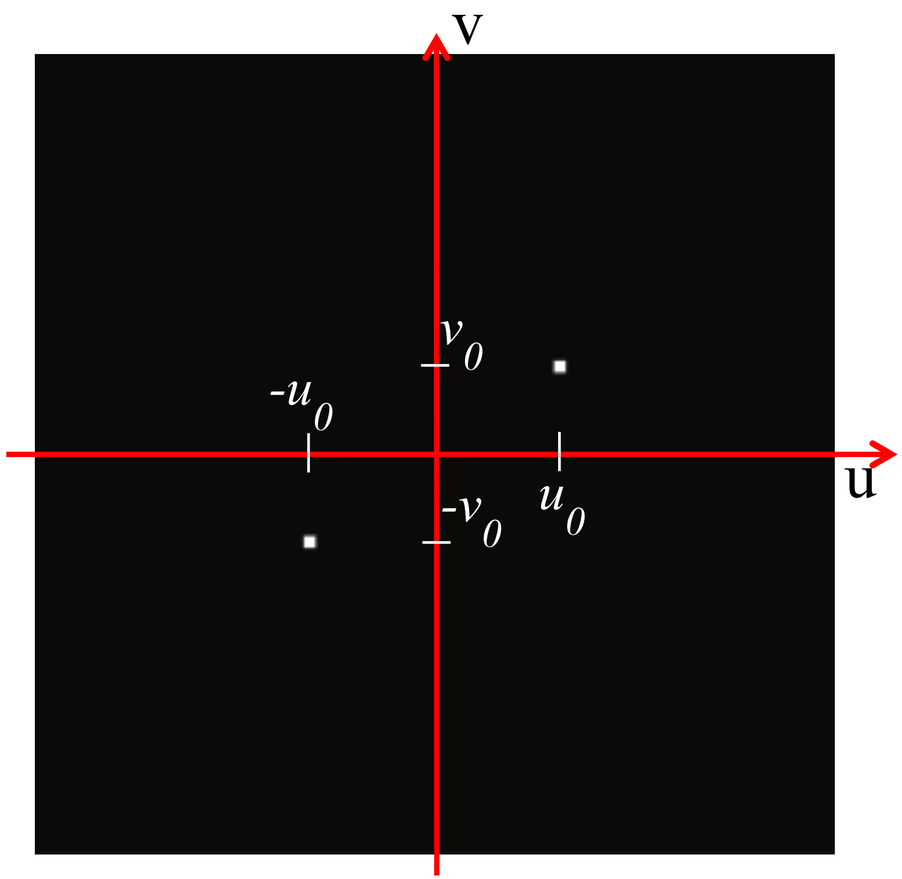
Simple Fourier transforms

Image



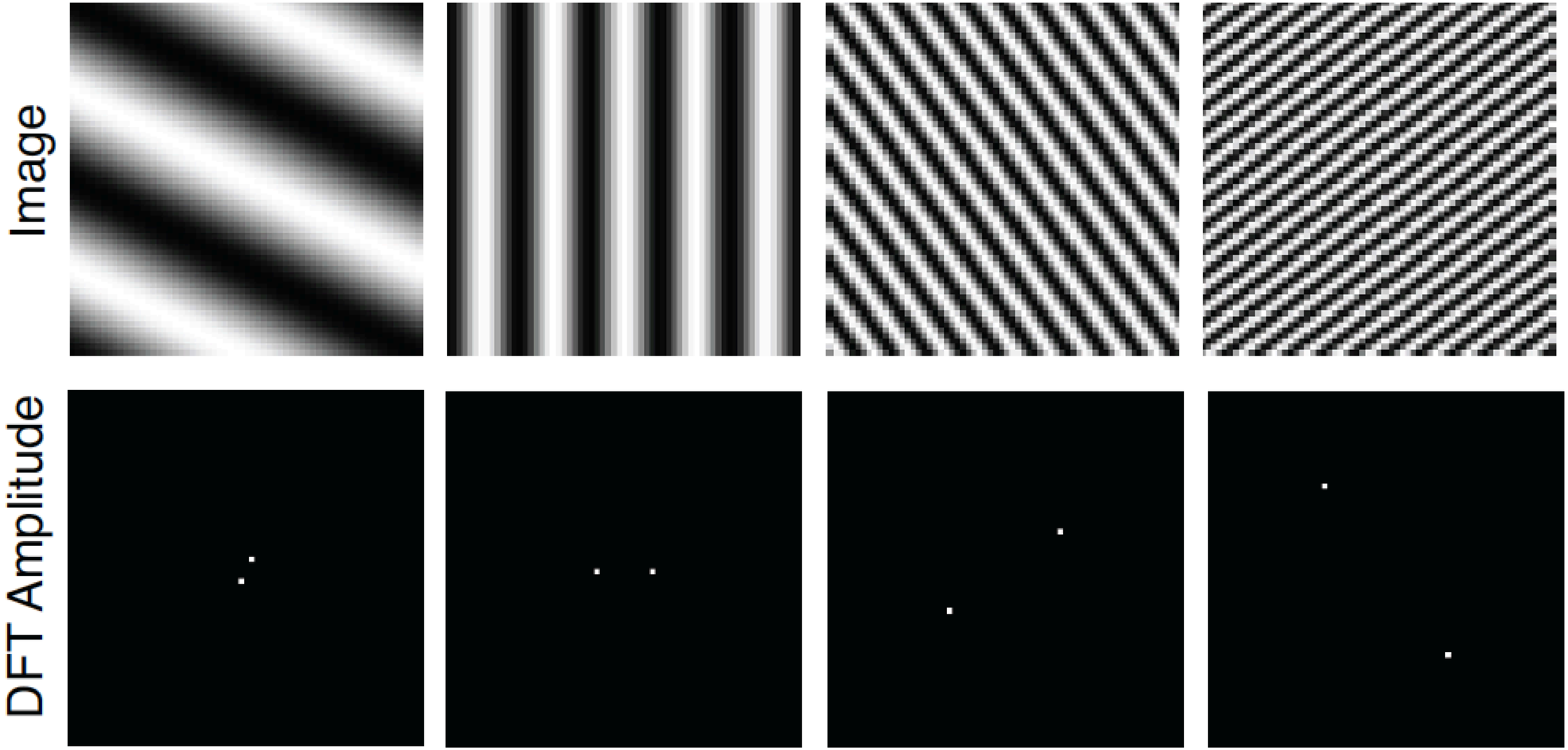
64x64 pixels

DFT (amplitude)



$$\cos\left(2\pi\left(\frac{u_0\,n}{N}+\frac{v_0\,m}{M}\right)\right) \longleftrightarrow \frac{1}{2}\left(\delta\left[u-u_0, v-v_0\right]+\delta\left[u+u_0, v+v_0\right]\right)$$

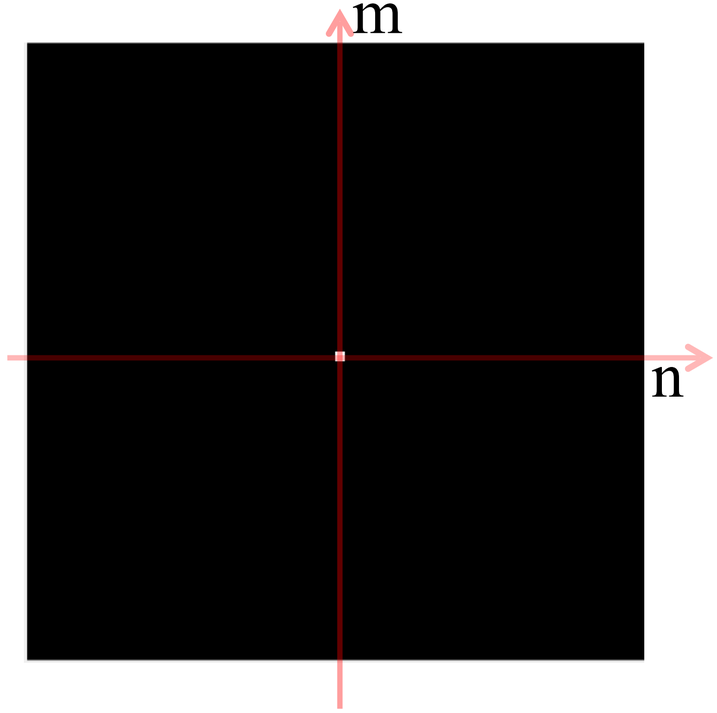
Simple Fourier transforms



Images are 64x64 pixels. The wave is a cosine, therefore DFT phase is zero.

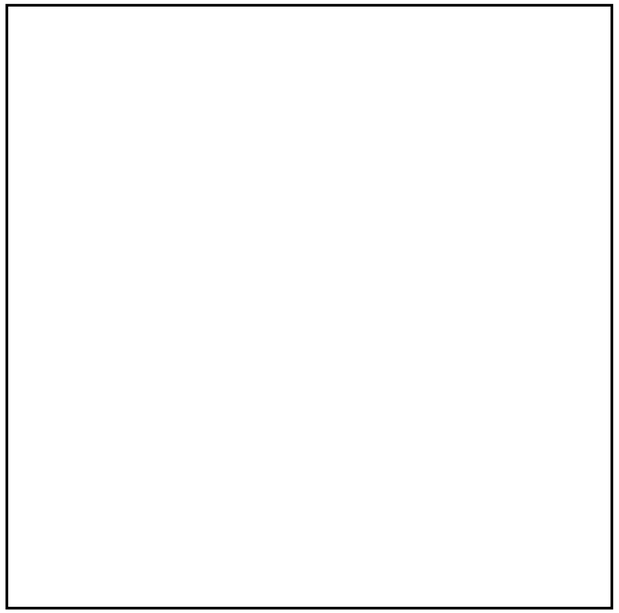
Some important Fourier transforms

Image



64x64 pixels

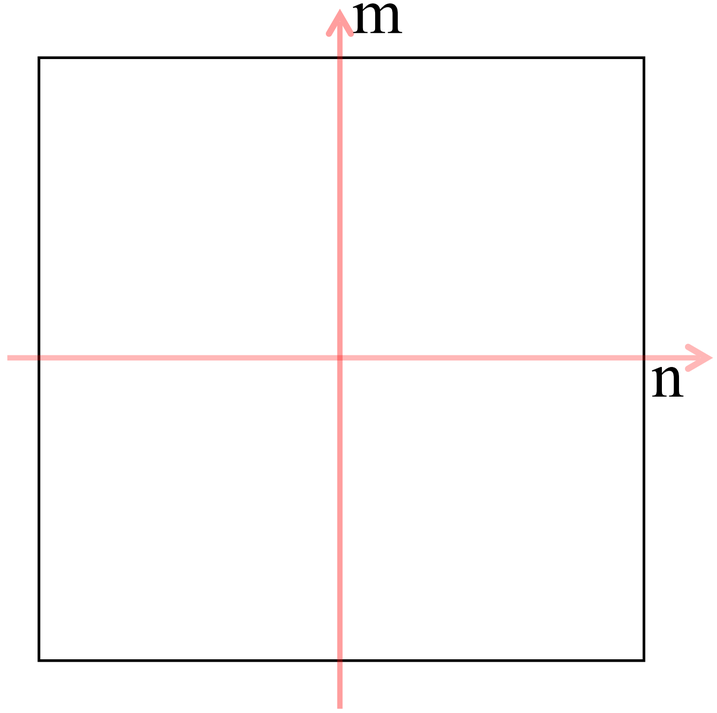
Magnitude DFT



Phase DFT

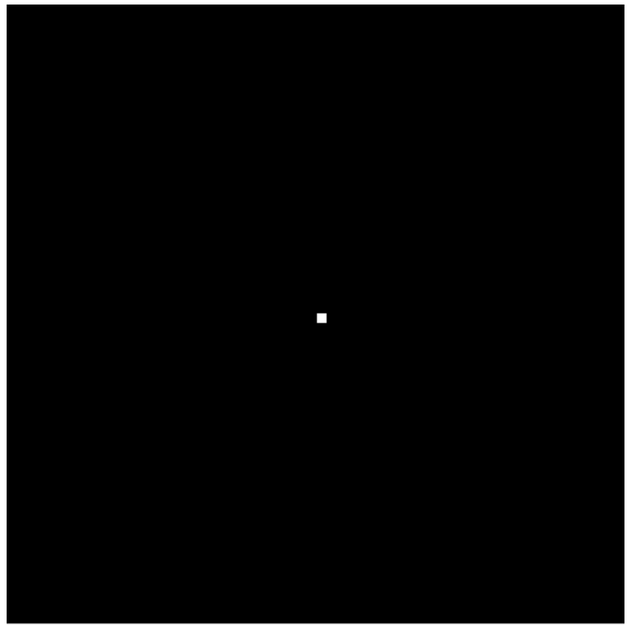
Some important Fourier transforms

Image



64x64 pixels

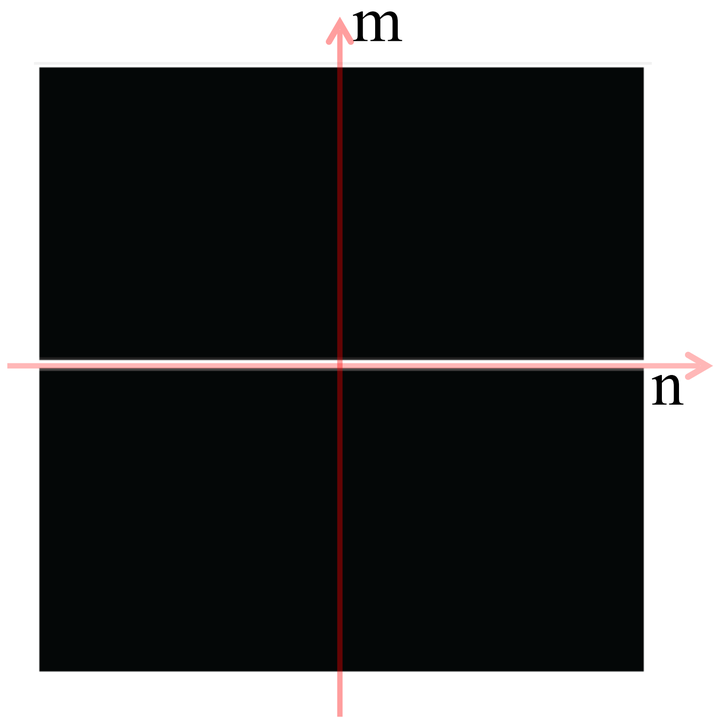
Magnitude DFT



Phase DFT

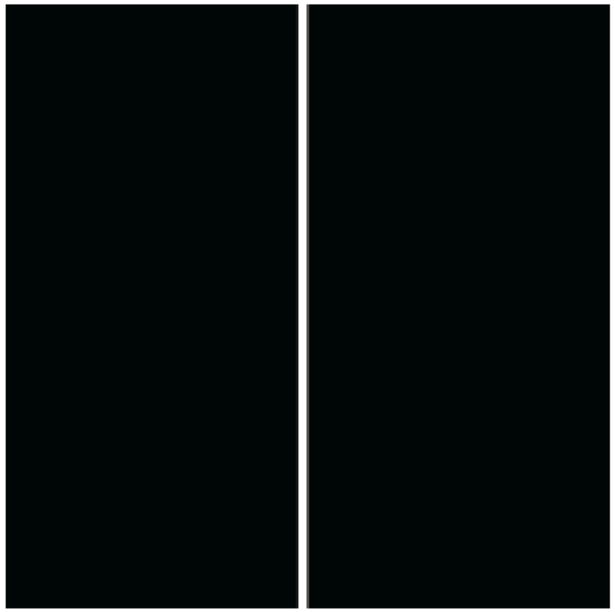
Some important Fourier transforms

Image



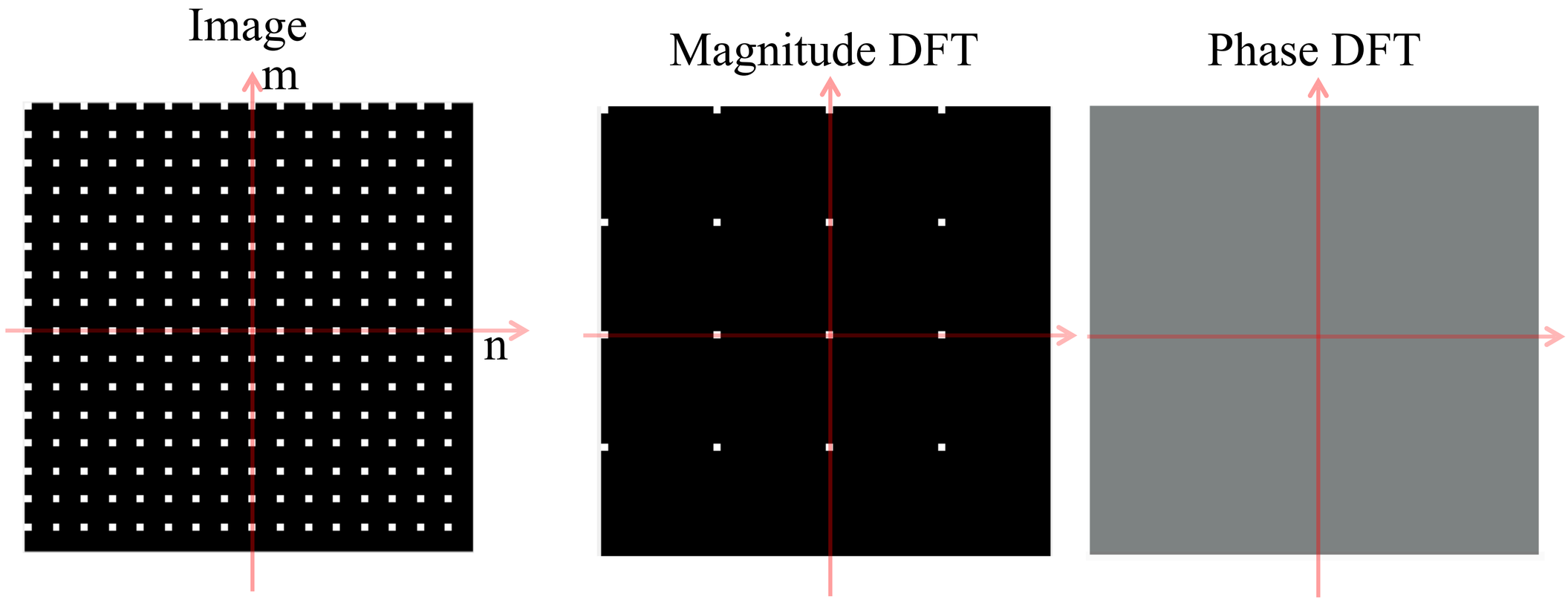
64x64 pixels

Magnitude DFT



Phase DFT

Some important Fourier transforms



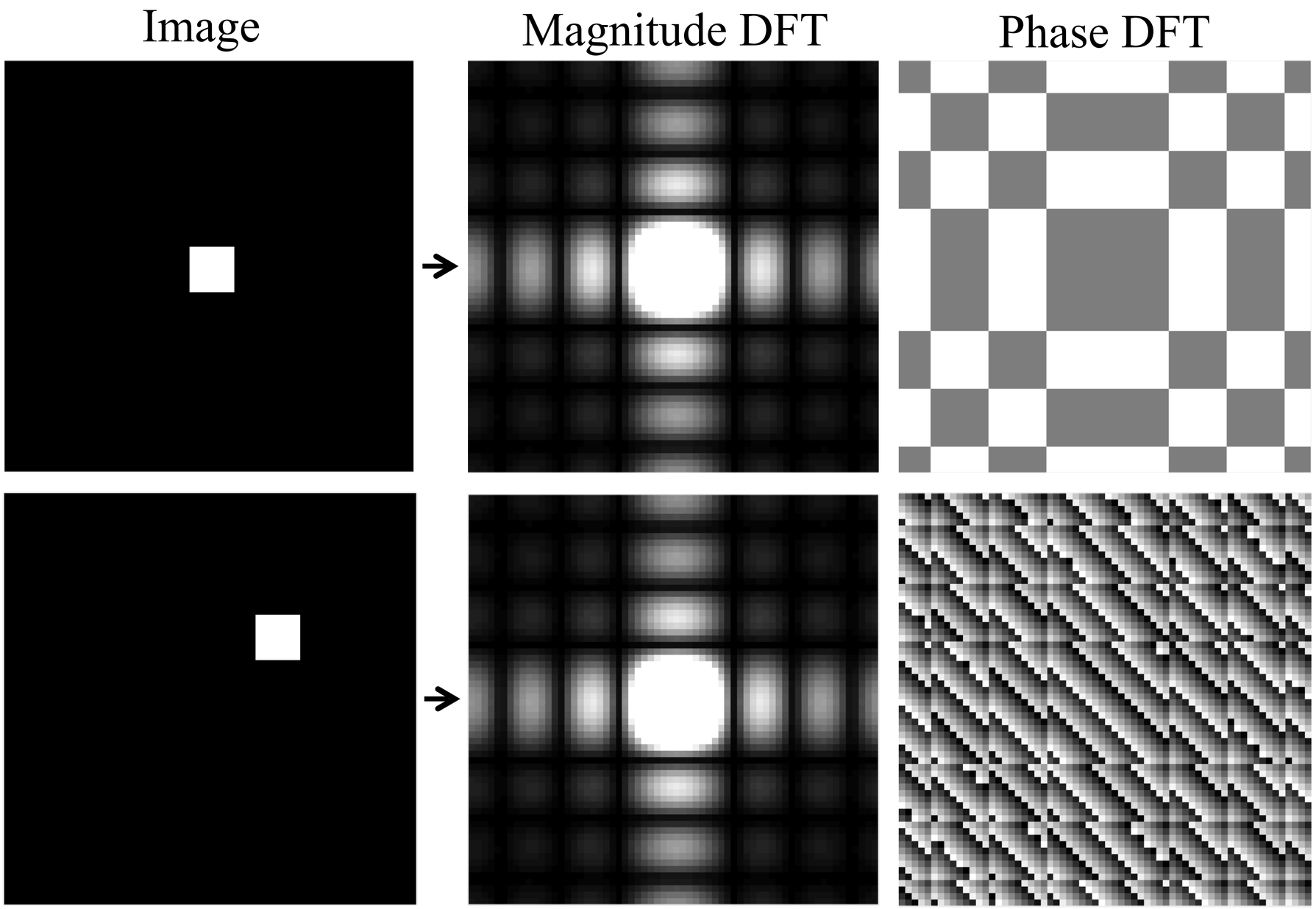
64x64 pixels

when M and N are divisible by k

$$\delta_k[n,m] = \sum_{s=0}^{N/k-1} \sum_{r=0}^{M/k-1} \delta[n - sk, m - rk] \quad \longleftrightarrow \quad \Delta_k[u,v] = \frac{NM}{k^2} \sum_{s=0}^{k-1} \sum_{r=0}^{k-1} \delta\left[u - s\frac{N}{k}, v - r\frac{M}{k}\right]$$

delta train, delta comb, impulse train

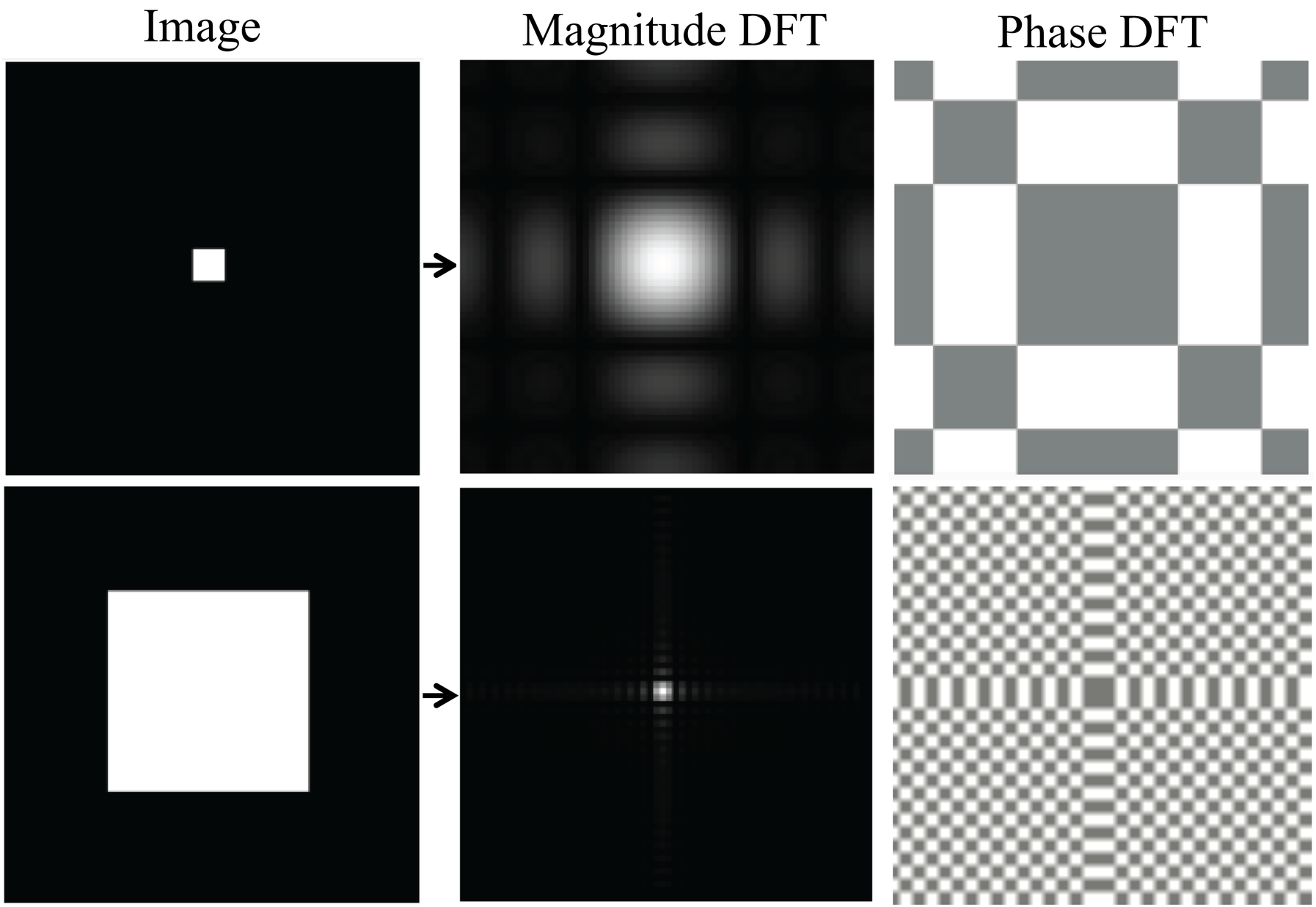
Some important Fourier transforms



Translation

Shifts of an image only produce changes on the phase of the DFT.

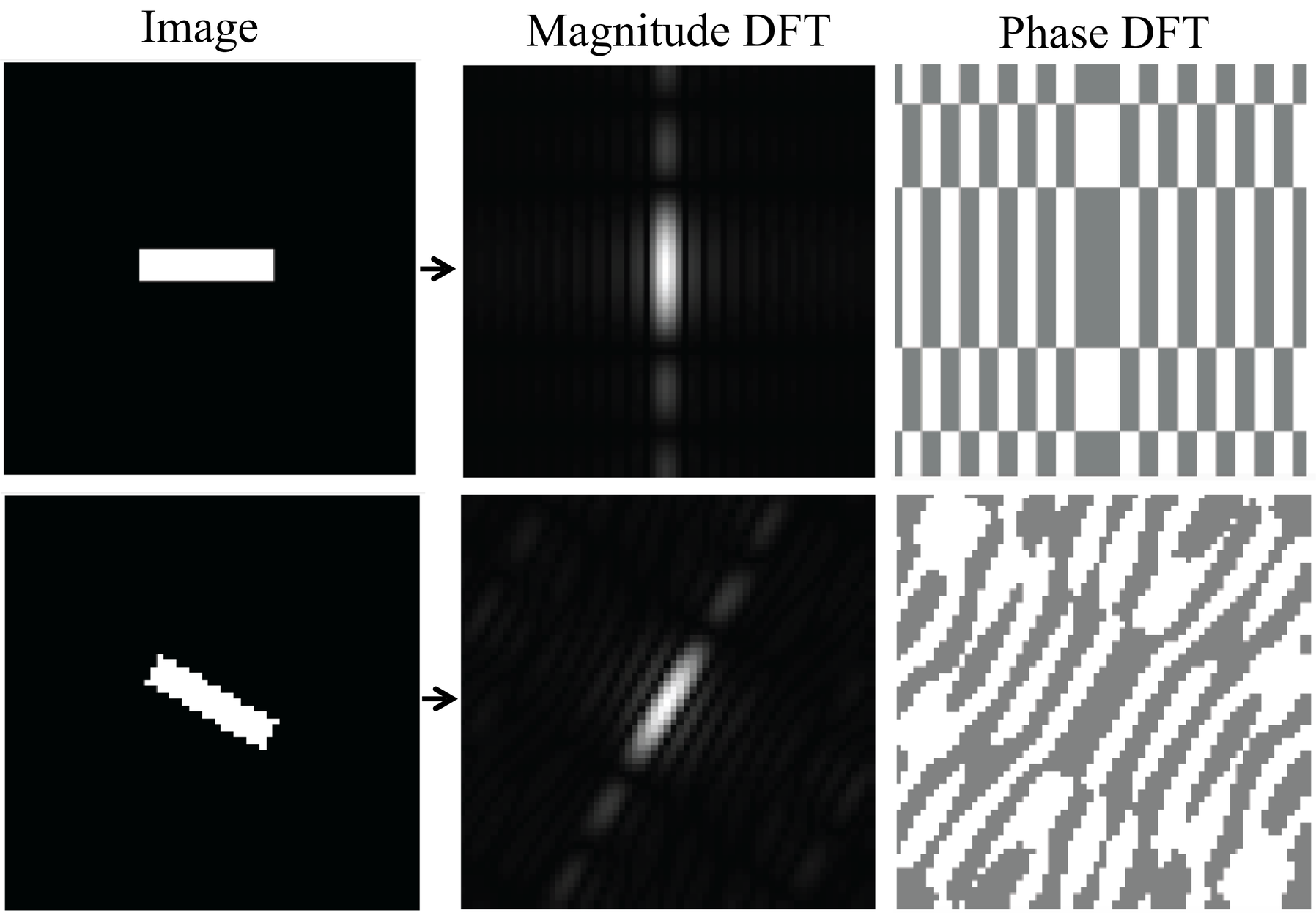
Some important Fourier transforms



Scale

Small image details produce content in high spatial frequencies

Some important Fourier transforms



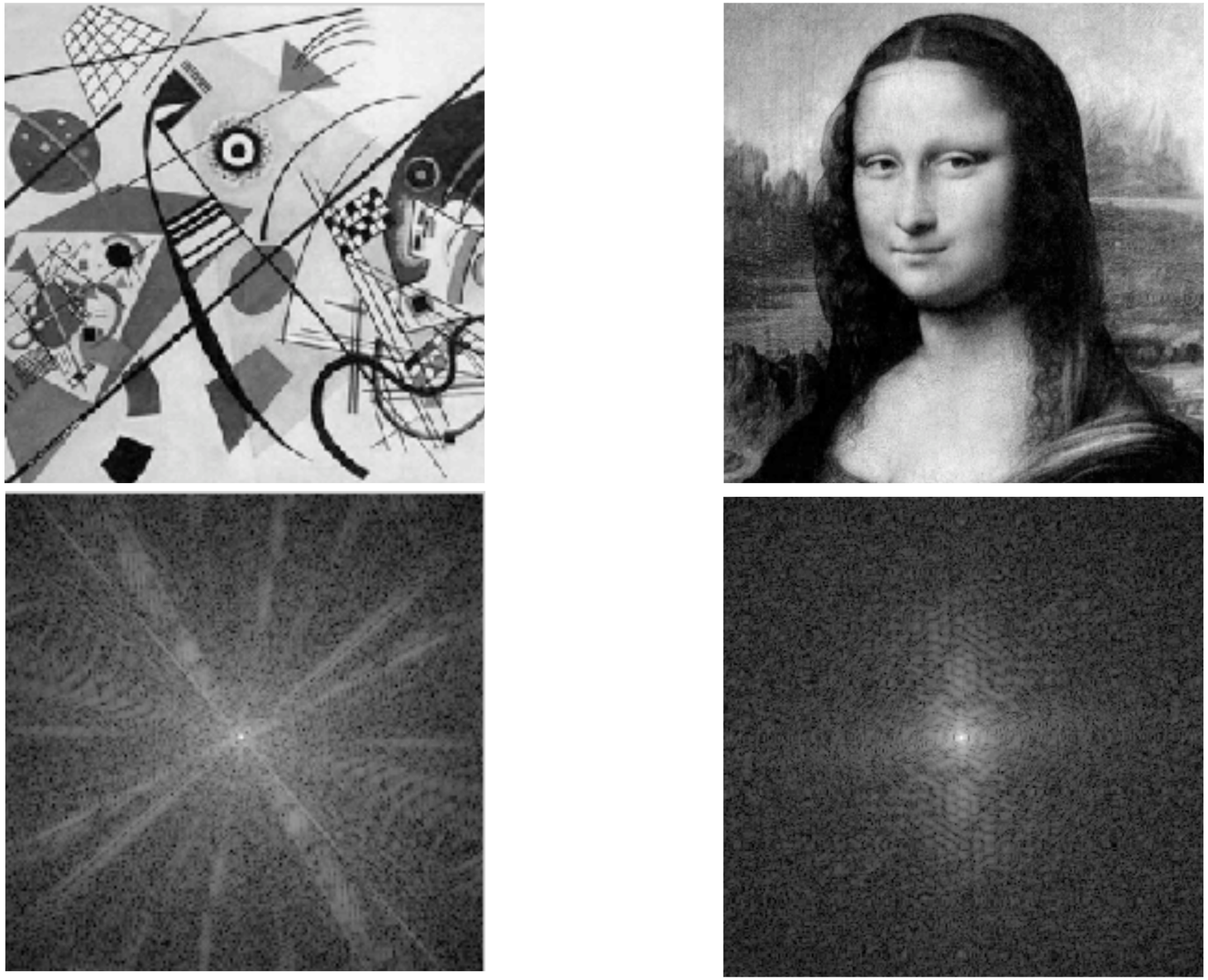
Orientation

A line transforms to a line oriented perpendicularly to the first.

The Fourier Transform of some important images

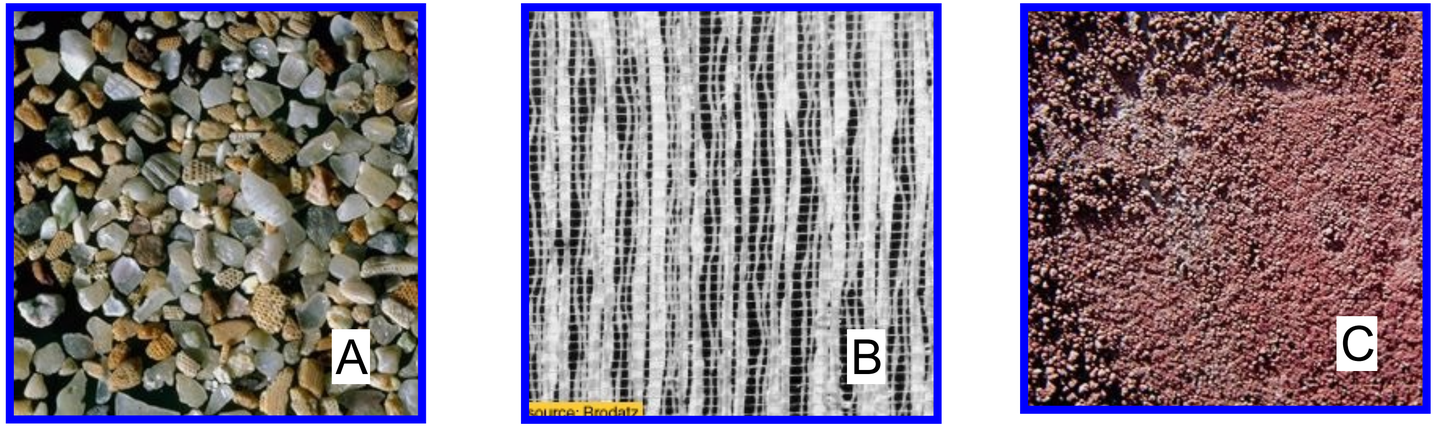
Image

Log(1+Magnitude FT)

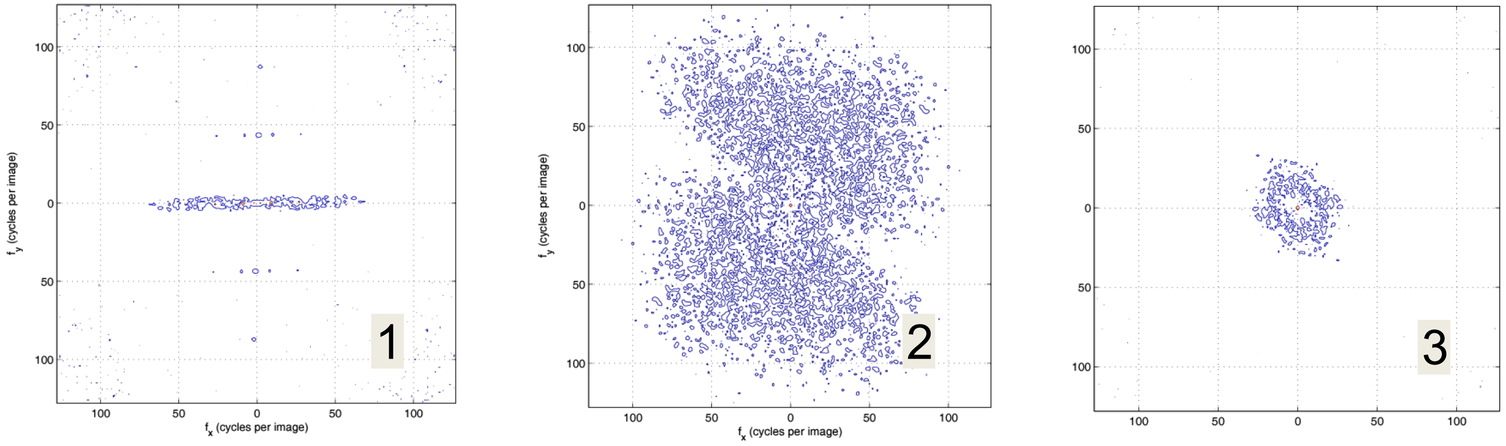


The DFT Game: find the right pairs

Images



DFT magnitude

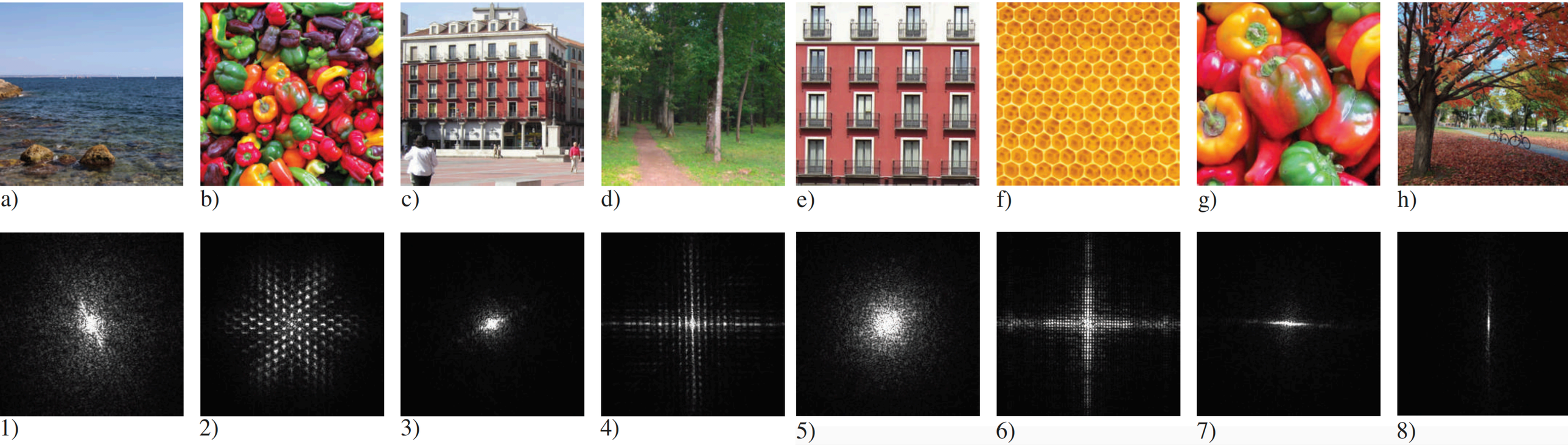


fx(cycles/image pixel size)

fx(cycles/image pixel size)

fx(cycles/image pixel size)

The DFT Game: find the right pairs



(Solution in the class notes)

The inverse Discrete Fourier transform

2D Discrete Fourier Transform (DFT) transforms an image *f* [*n*,*m*] into *F* [*u*,*v*] as:

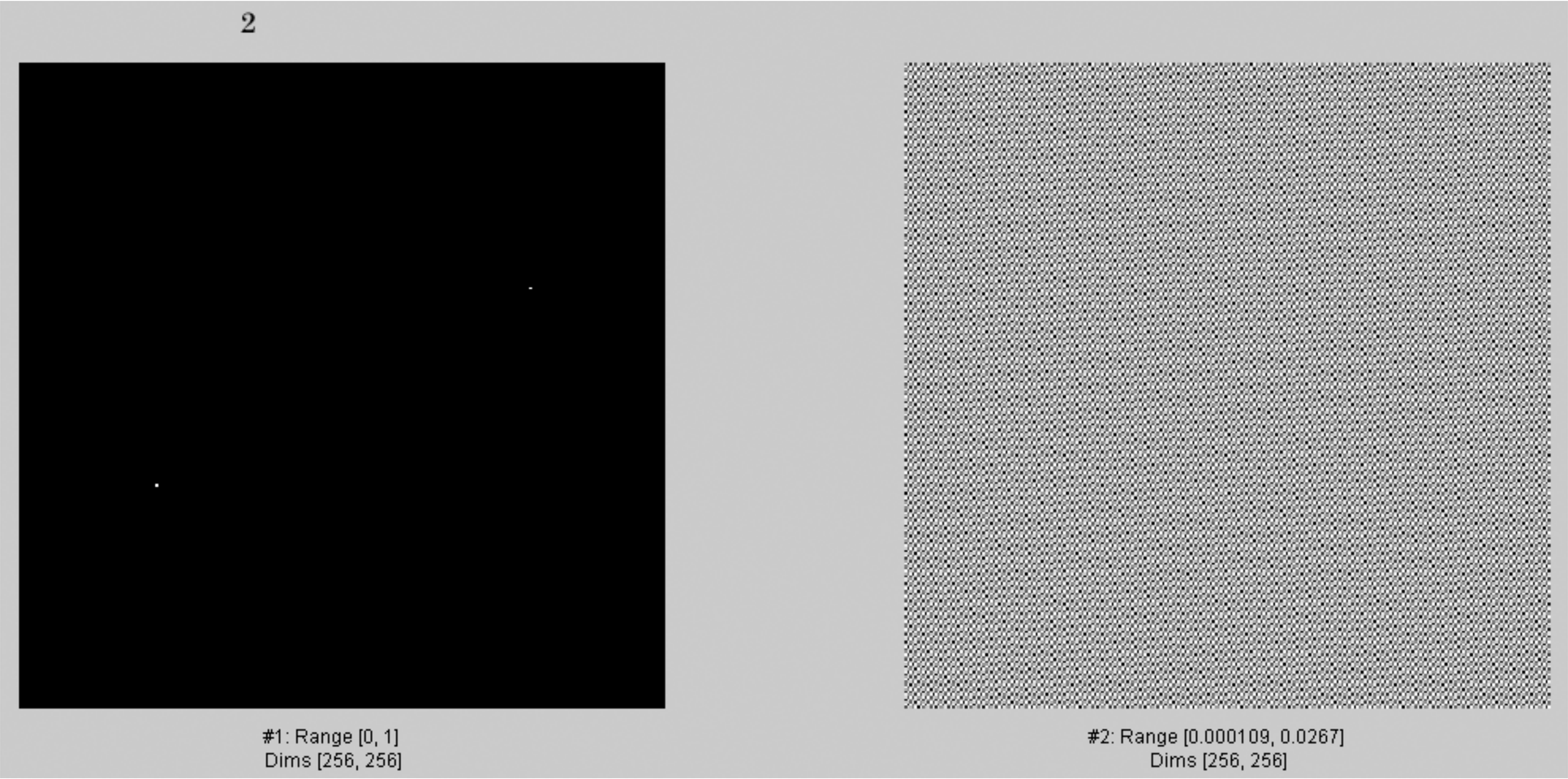
$$F[u,v] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

The inverse of the 2D DFT is:

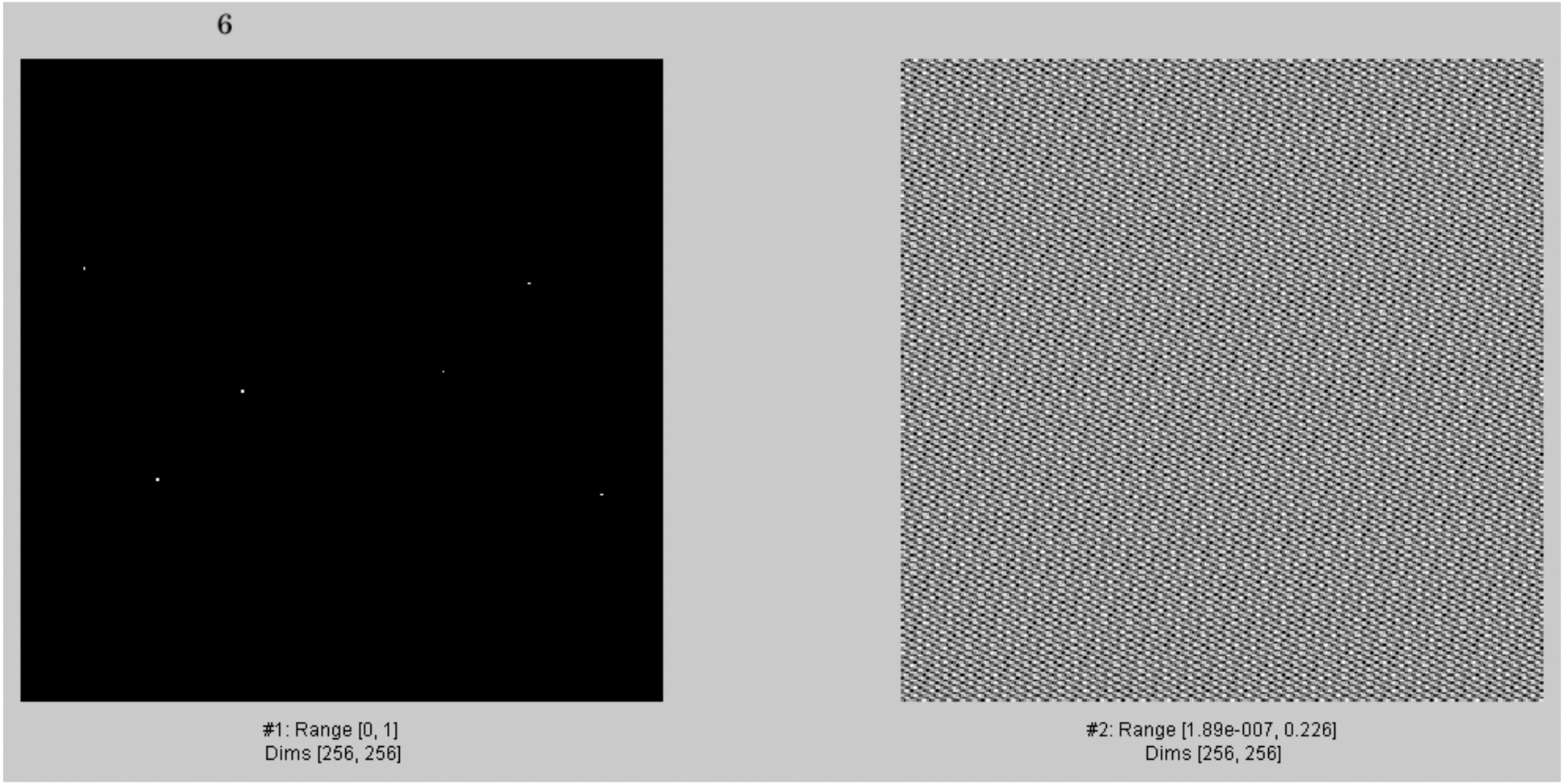
$$f[n,m] = \frac{1}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} F[u,v] \exp\left(+2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

How does summing waves ends up giving back a picture?

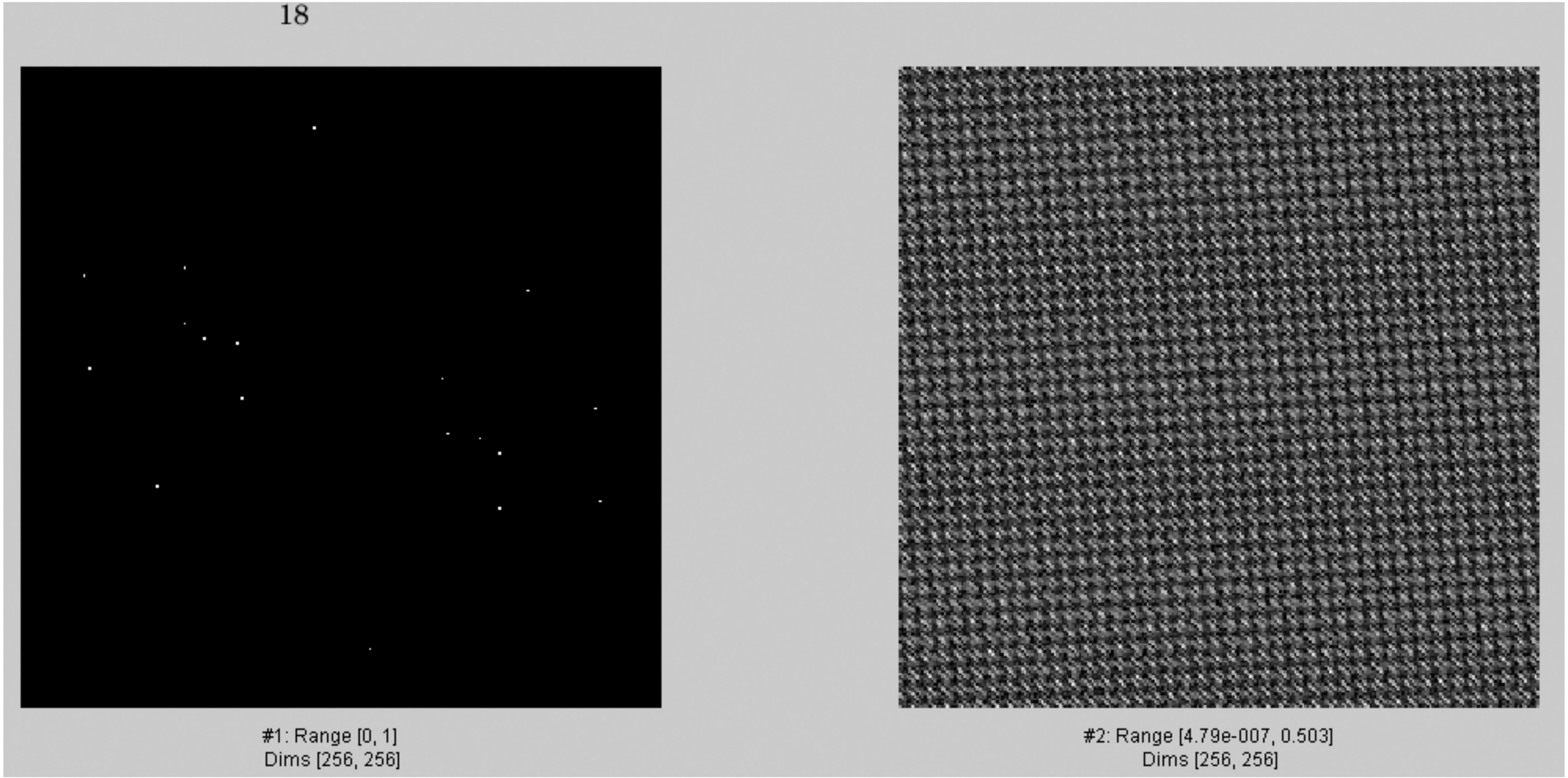
2



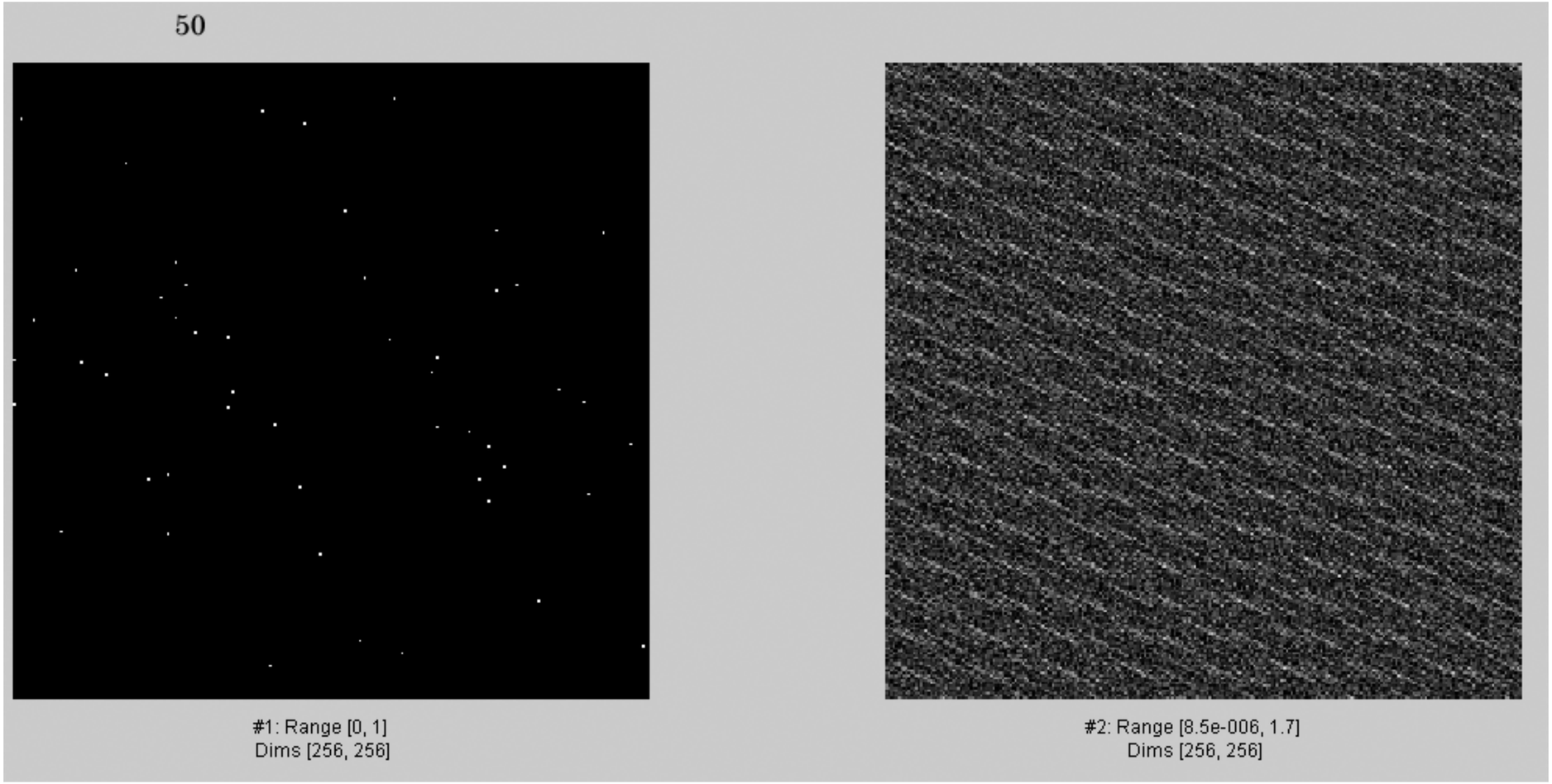
6



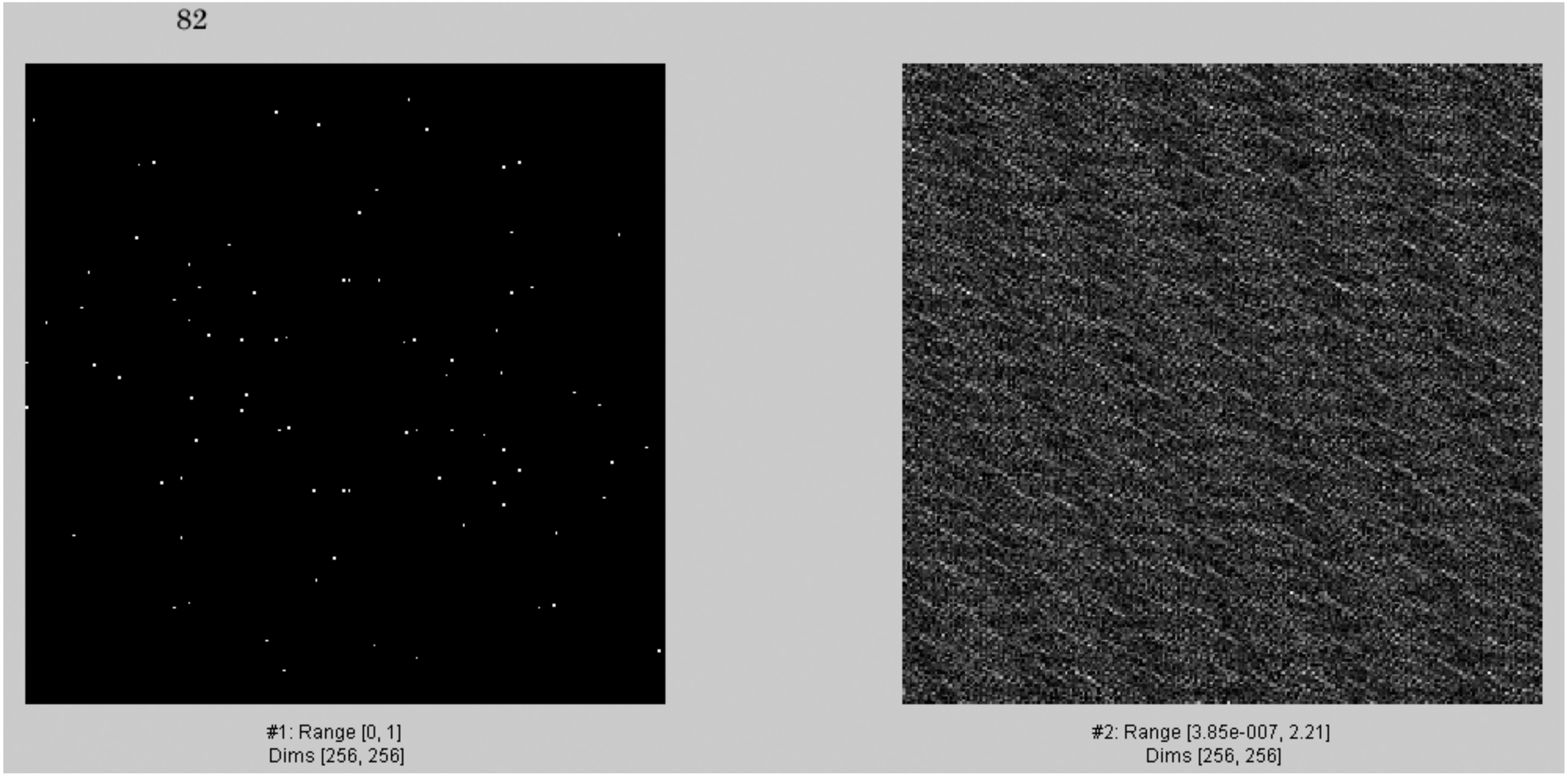
18



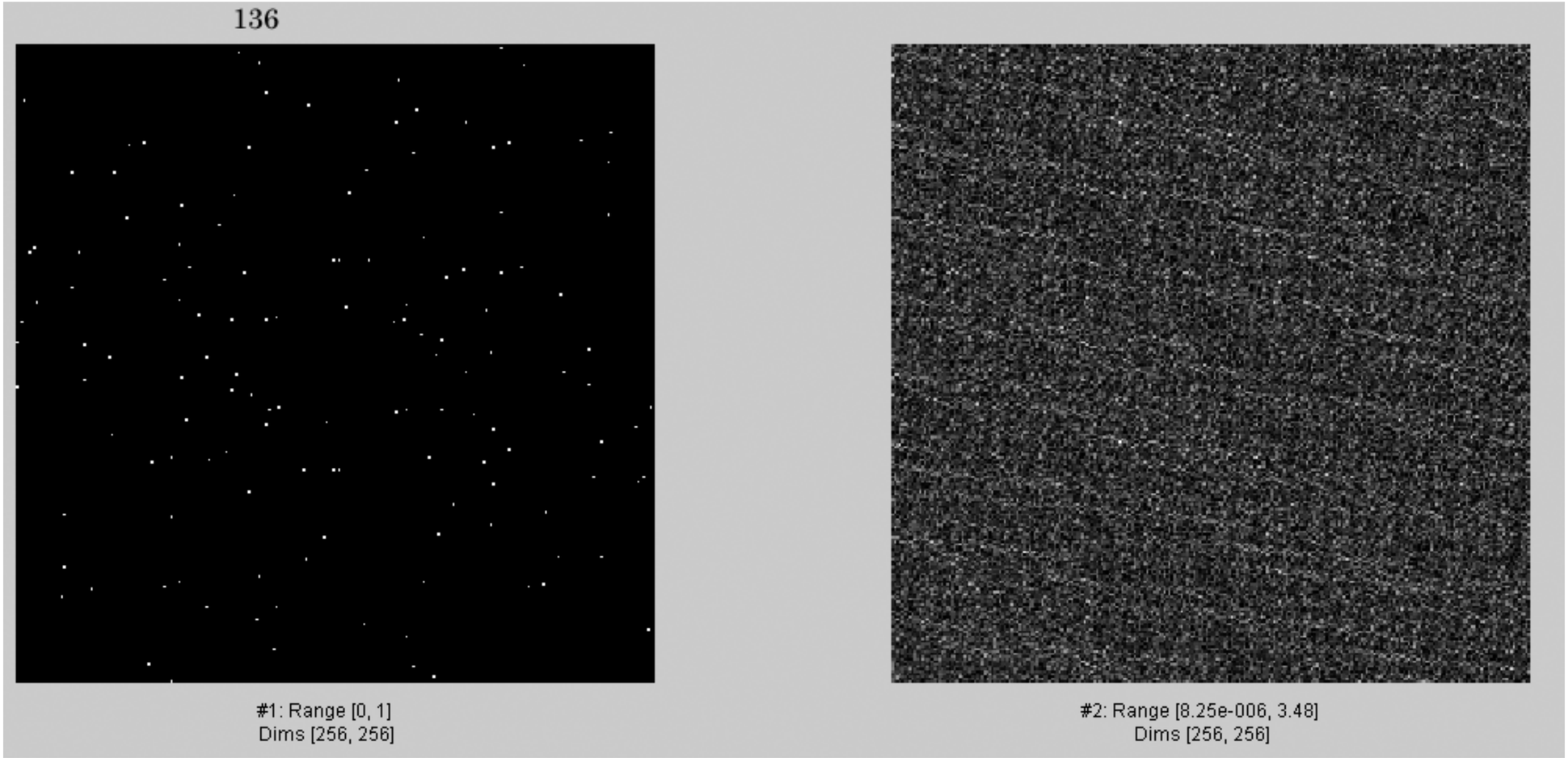
50



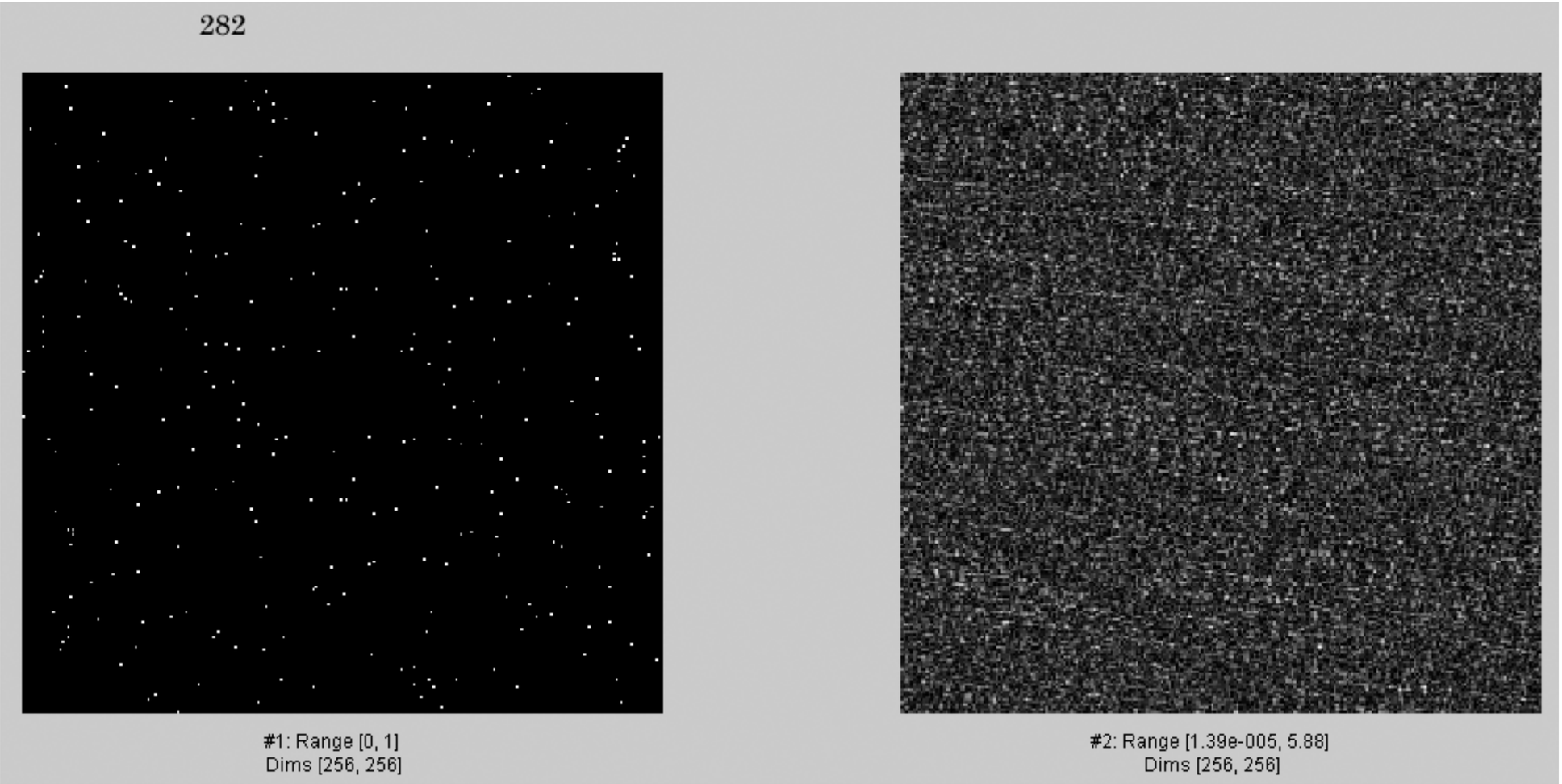
82



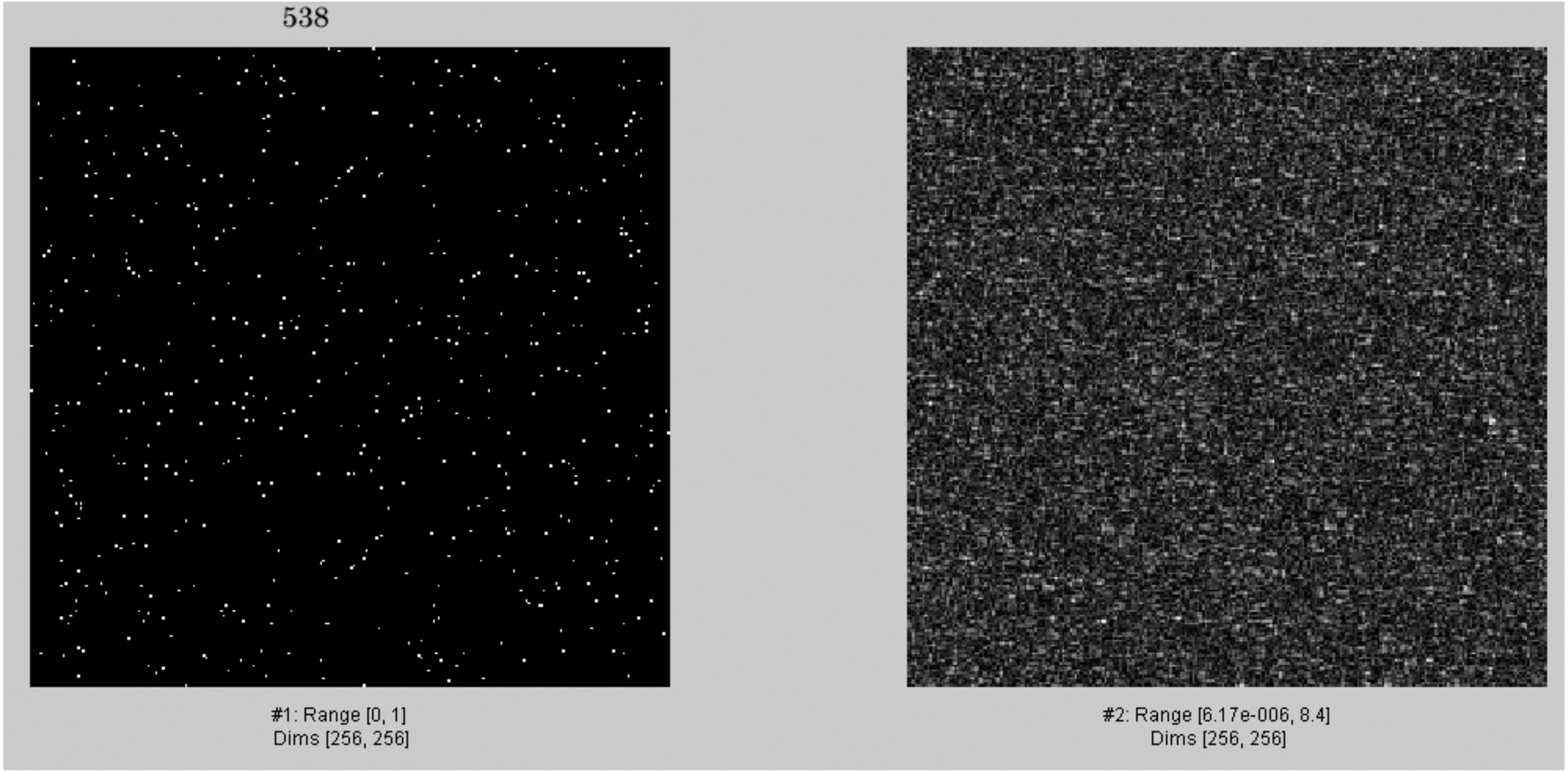
136



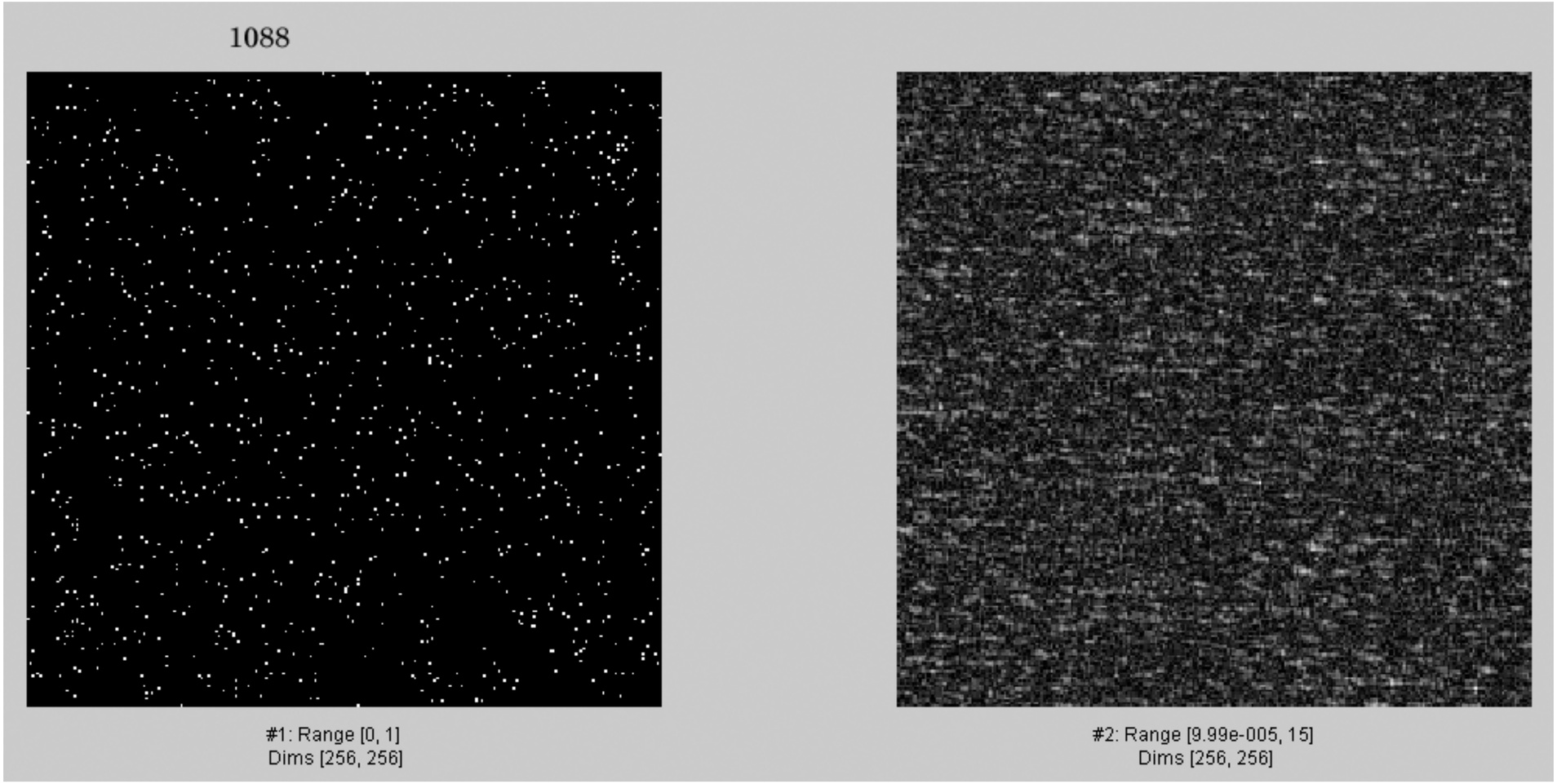
282



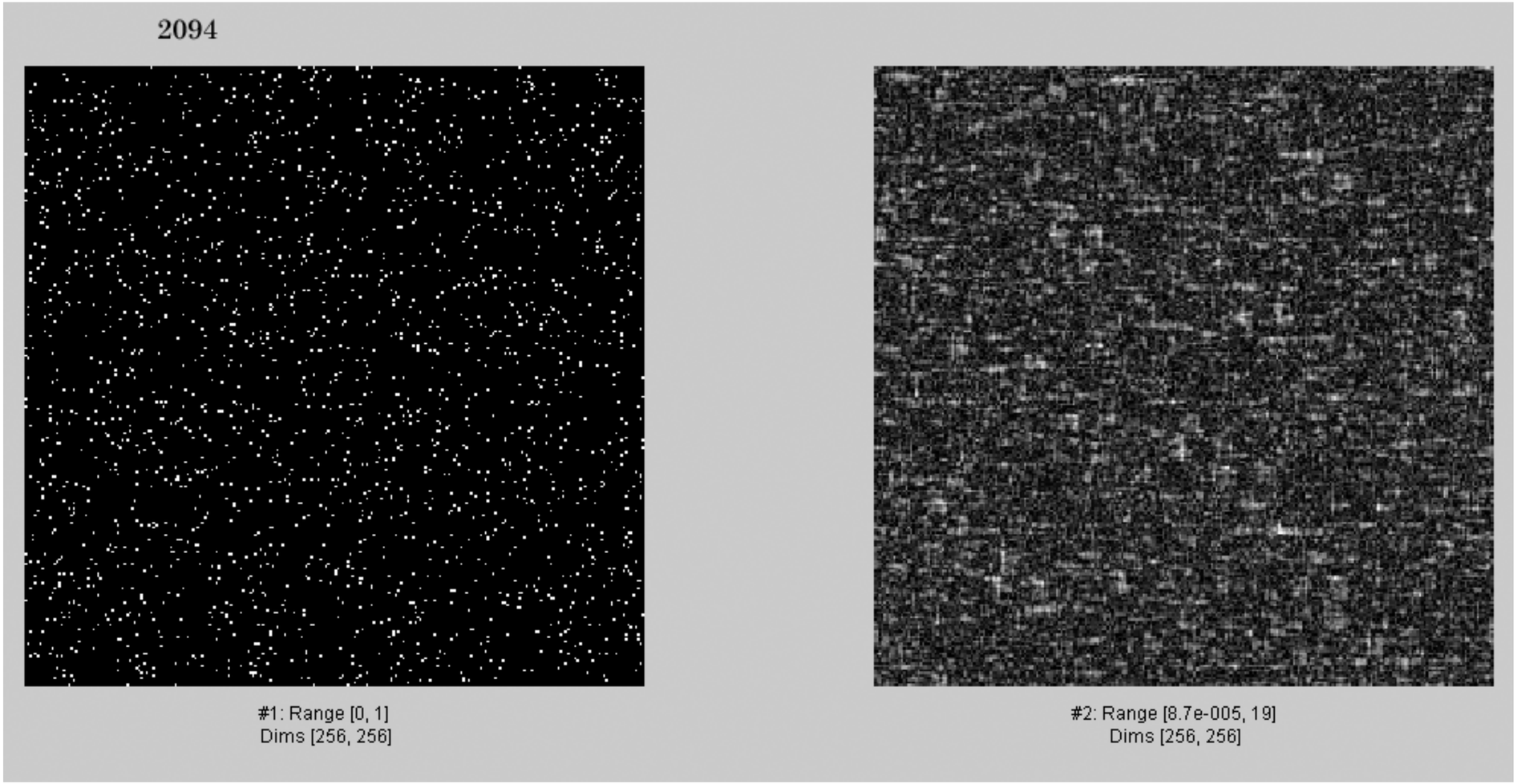
538



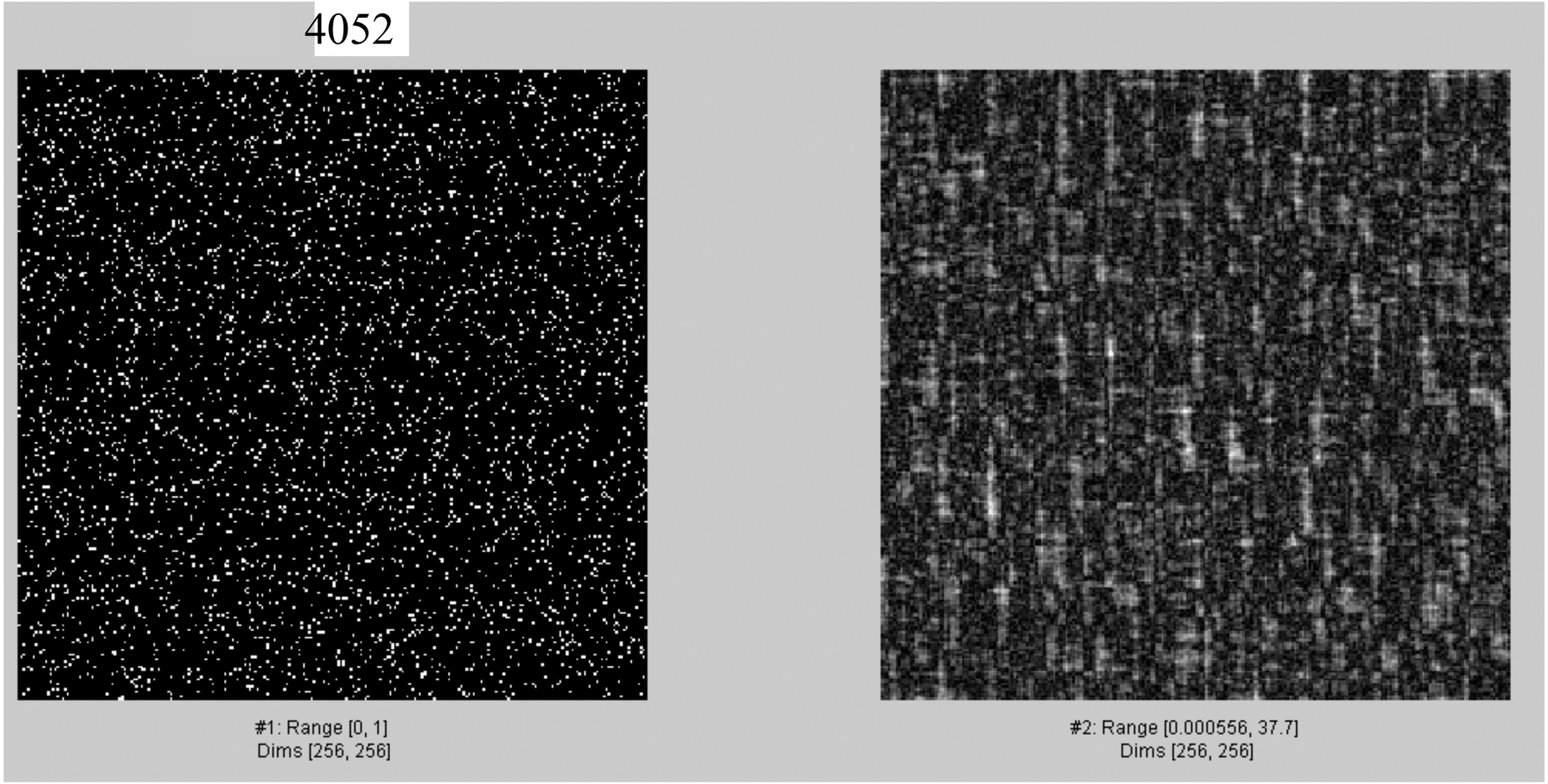
1088



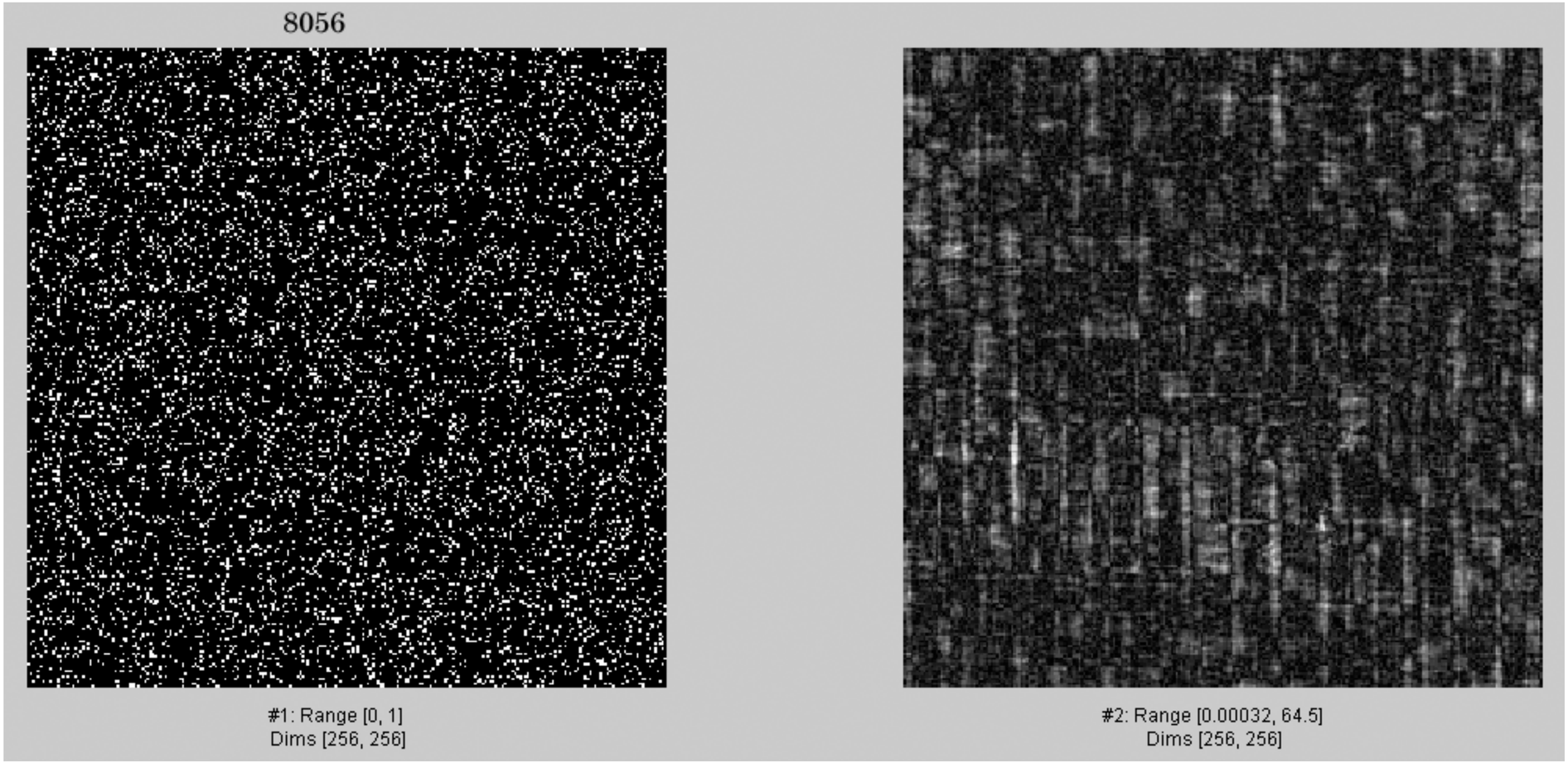
2094



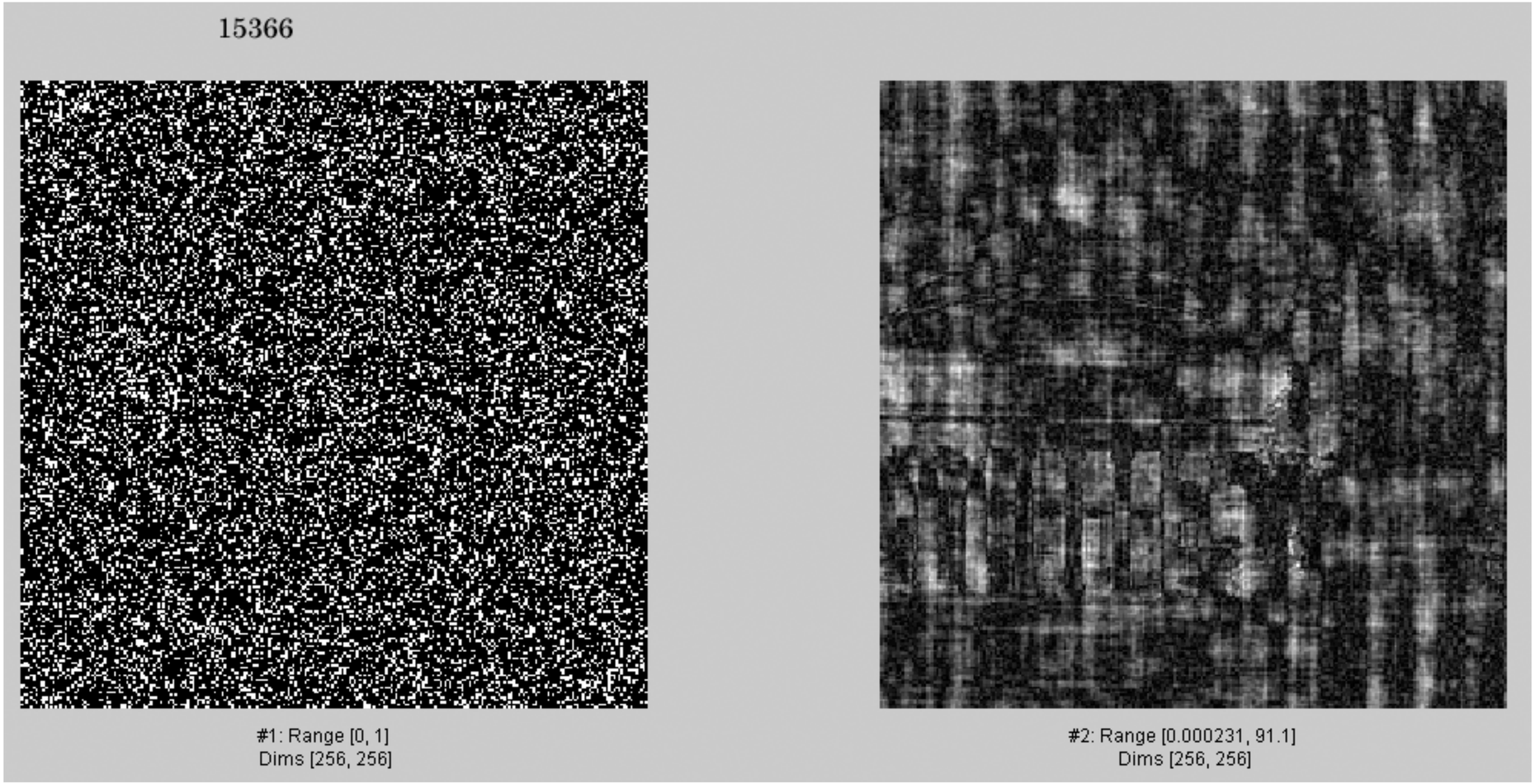
4052.



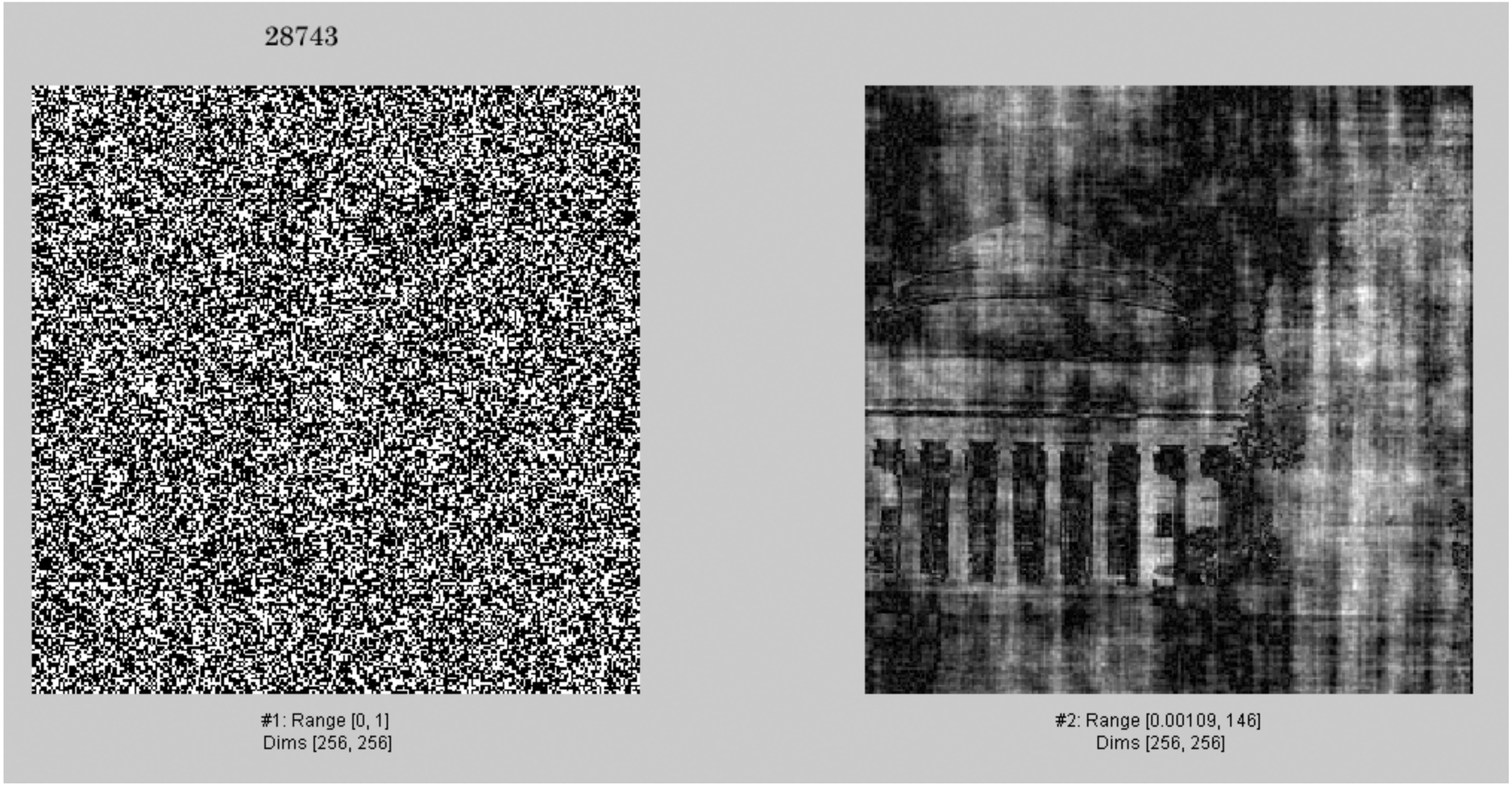
8056.



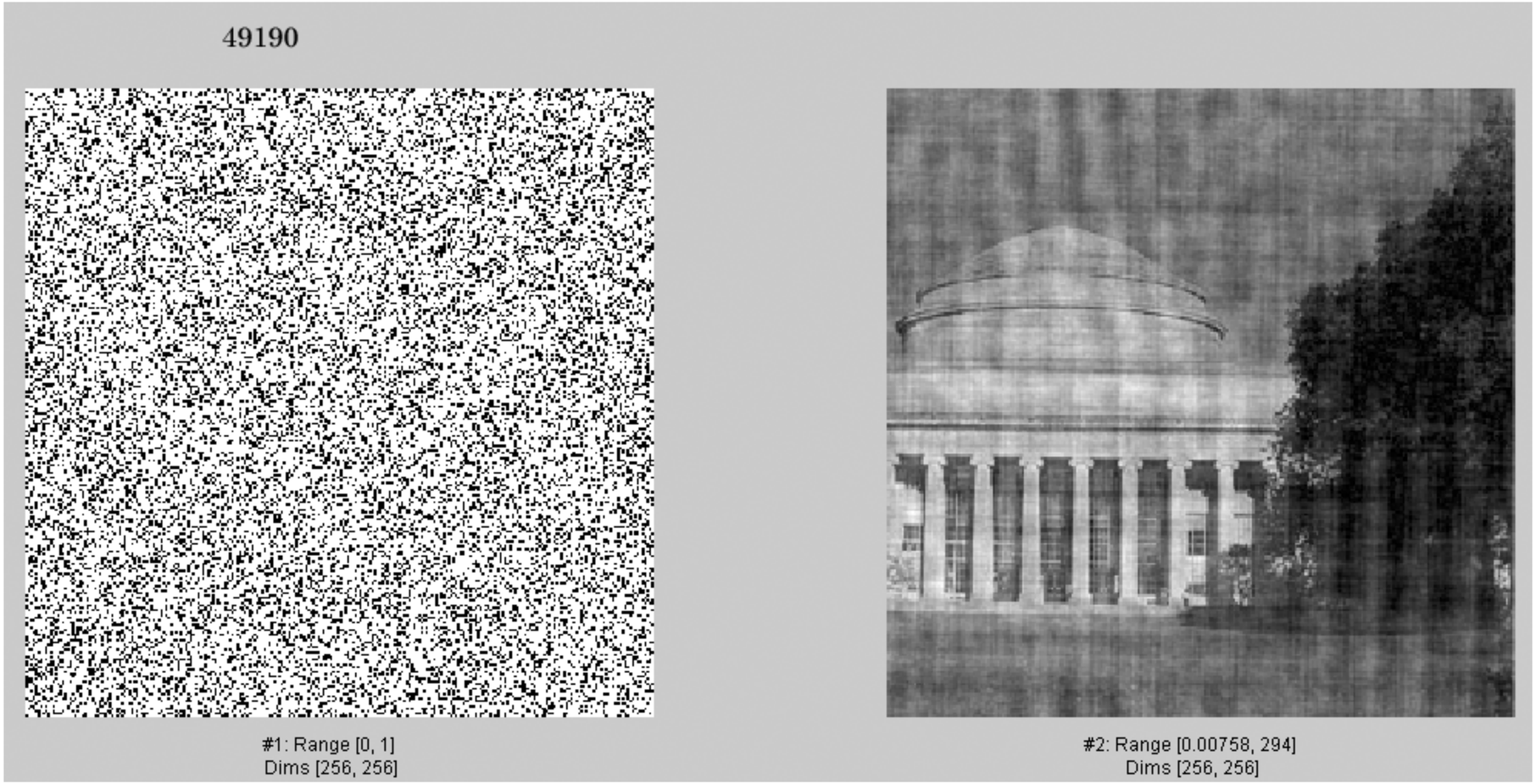
15366



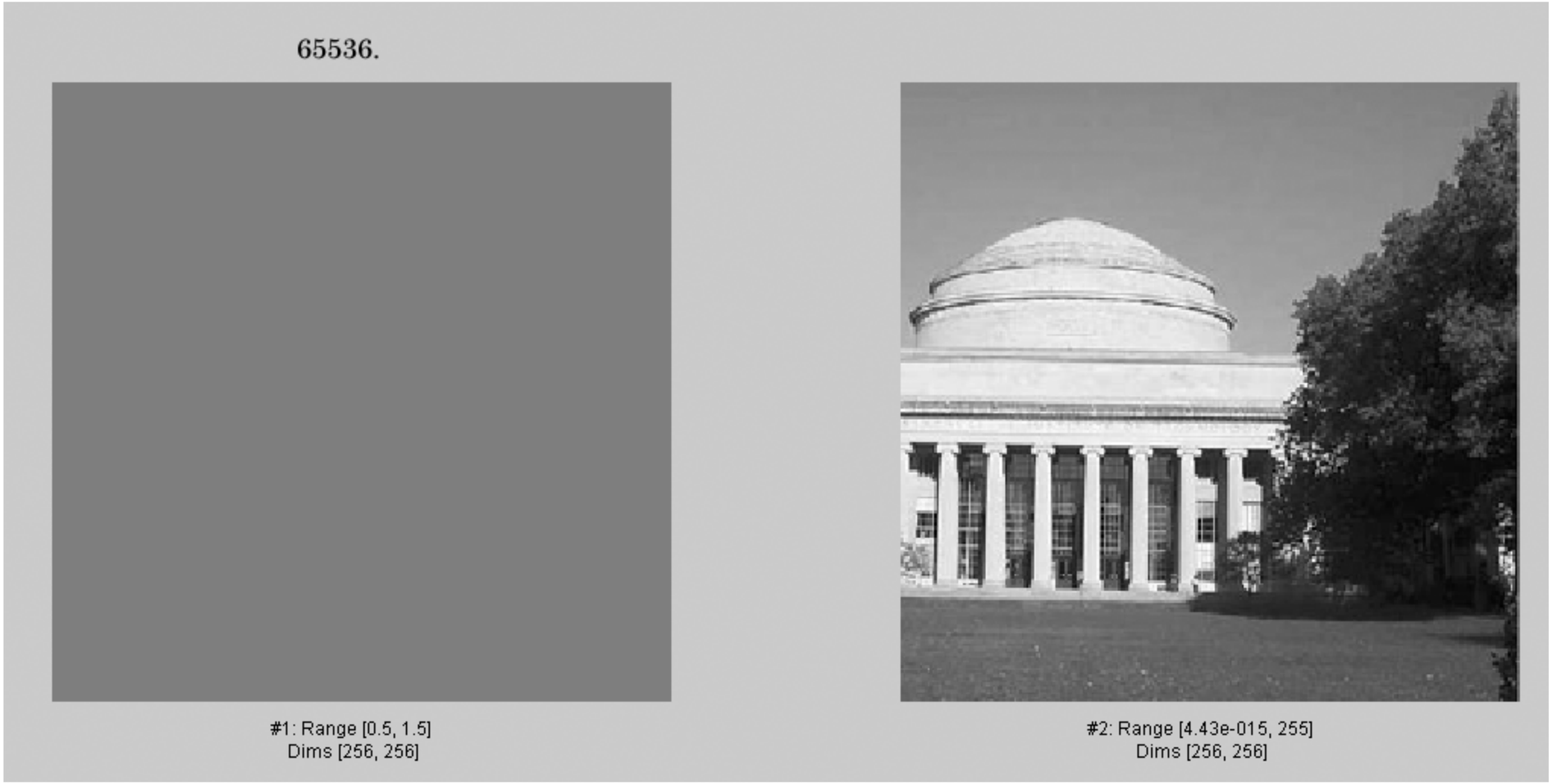
28743



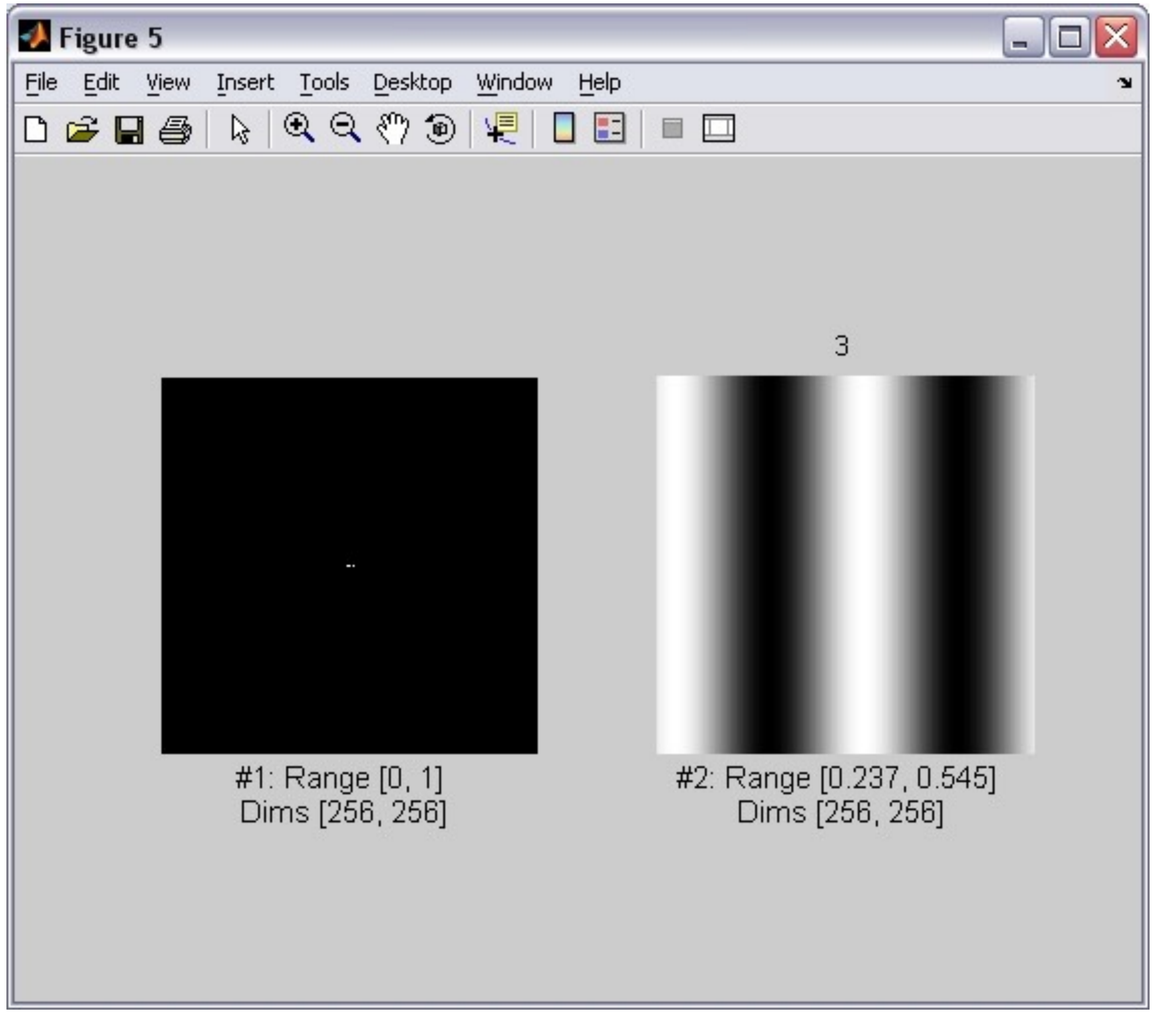
49190.



65536.


65536.
#1: Range [0.5, 1.5]
Dims [256, 256]
#2: Range [4.43e-015, 255]
Dims [256, 256]

3



Now, an analogous sequence of images, but selecting Fourier components in descending order of magnitude.

5

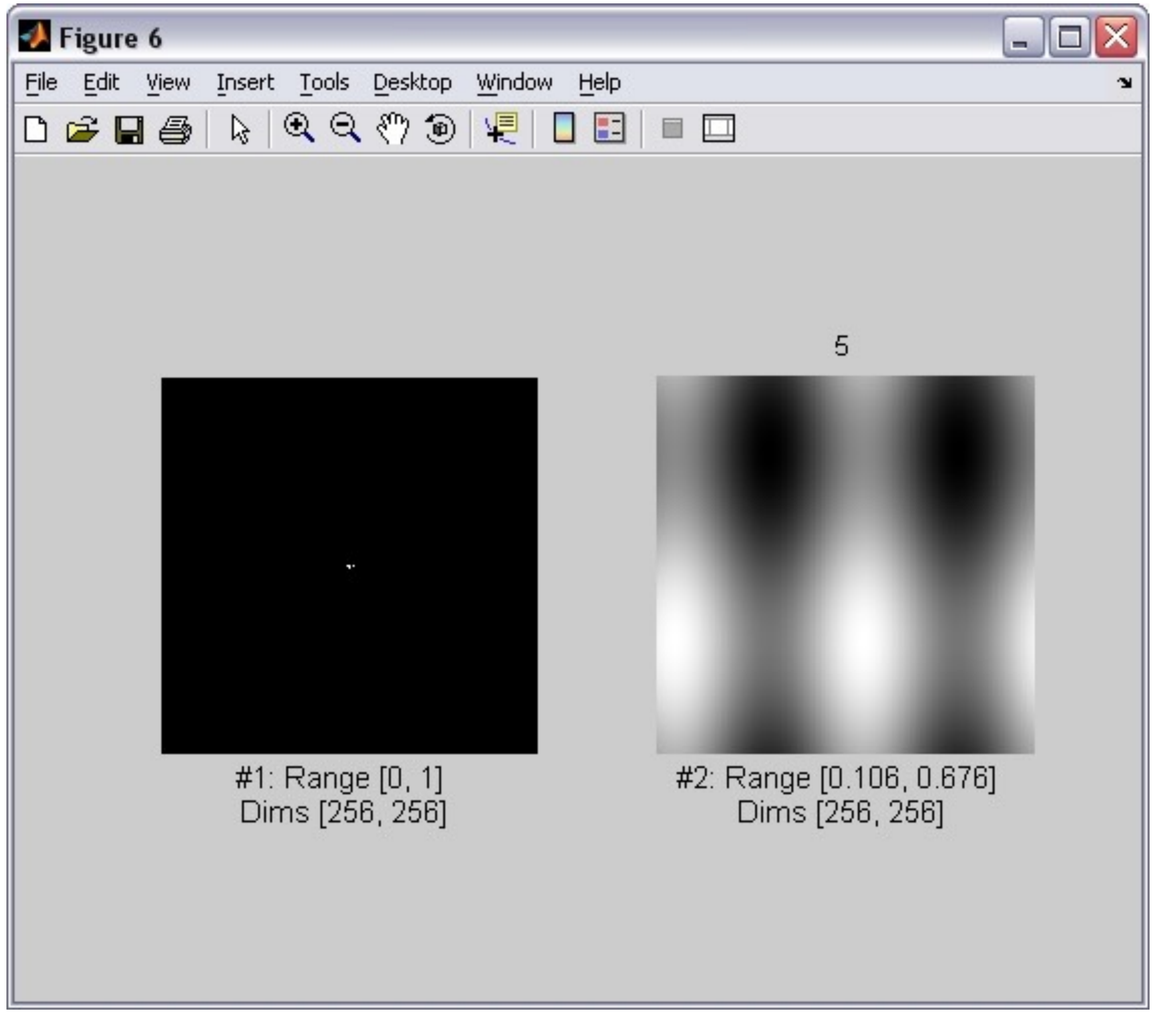


Figure 6
File Edit View Insert Tools Desktop Window Help
5
#1: Range [0, 1]
Dims [256, 256]
#2: Range [0.106, 0.676]
Dims [256, 256]

9

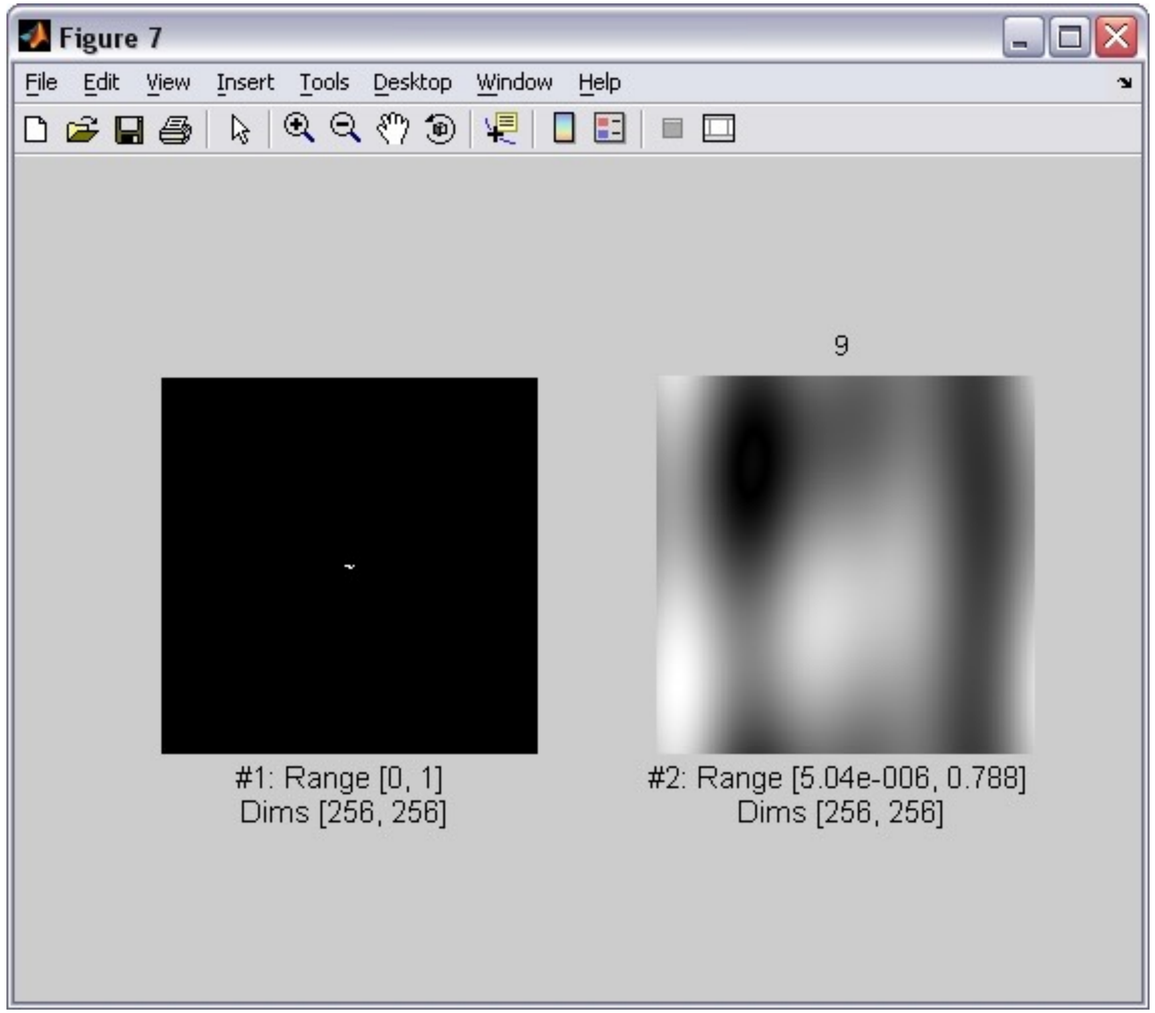


Figure 7
File Edit View Insert Tools Desktop Window Help
9
#1: Range [0, 1]
Dims [256, 256]
#2: Range [5.04e-006, 0.788]
Dims [256, 256]

17

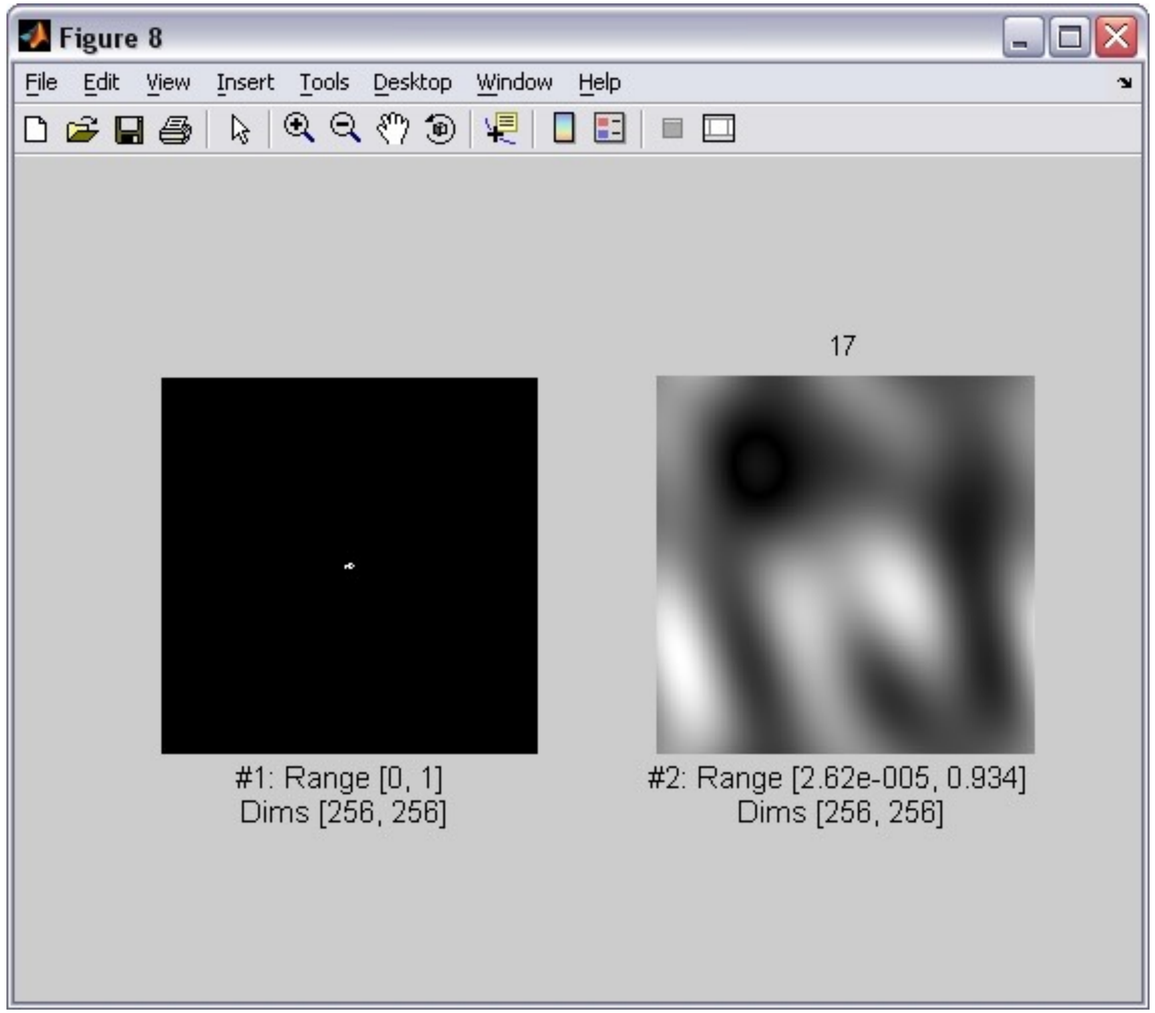


Figure 8
File Edit View Insert Tools Desktop Window Help
17
#1: Range [0, 1]
Dims [256, 256]
#2: Range [2.62e-005, 0.934]
Dims [256, 256]

33

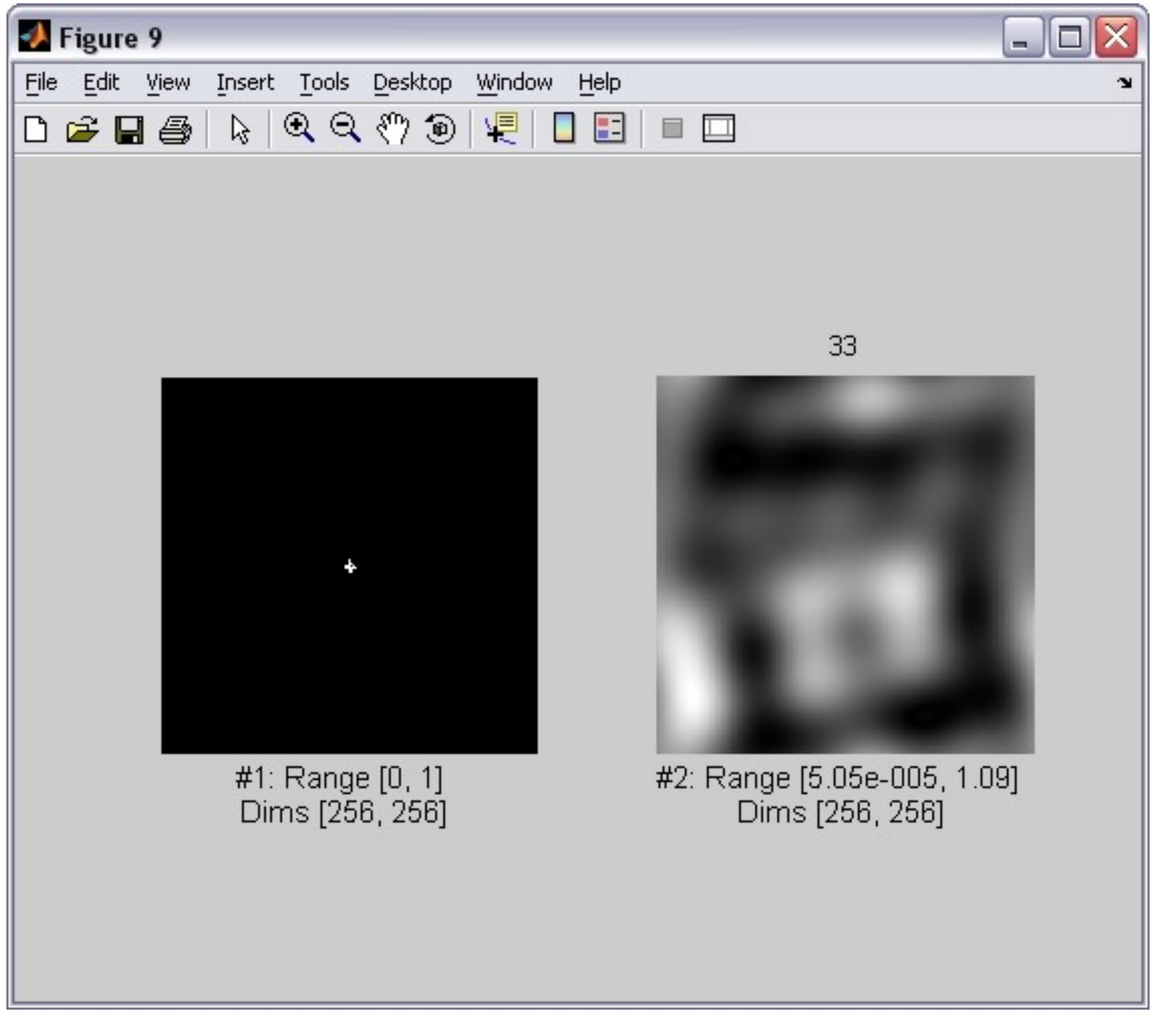


Figure 9
File Edit View Insert Tools Desktop Window Help
33
#1: Range [0, 1]
Dims [256, 256]
#2: Range [5.05e-005, 1.09]
Dims [256, 256]

65

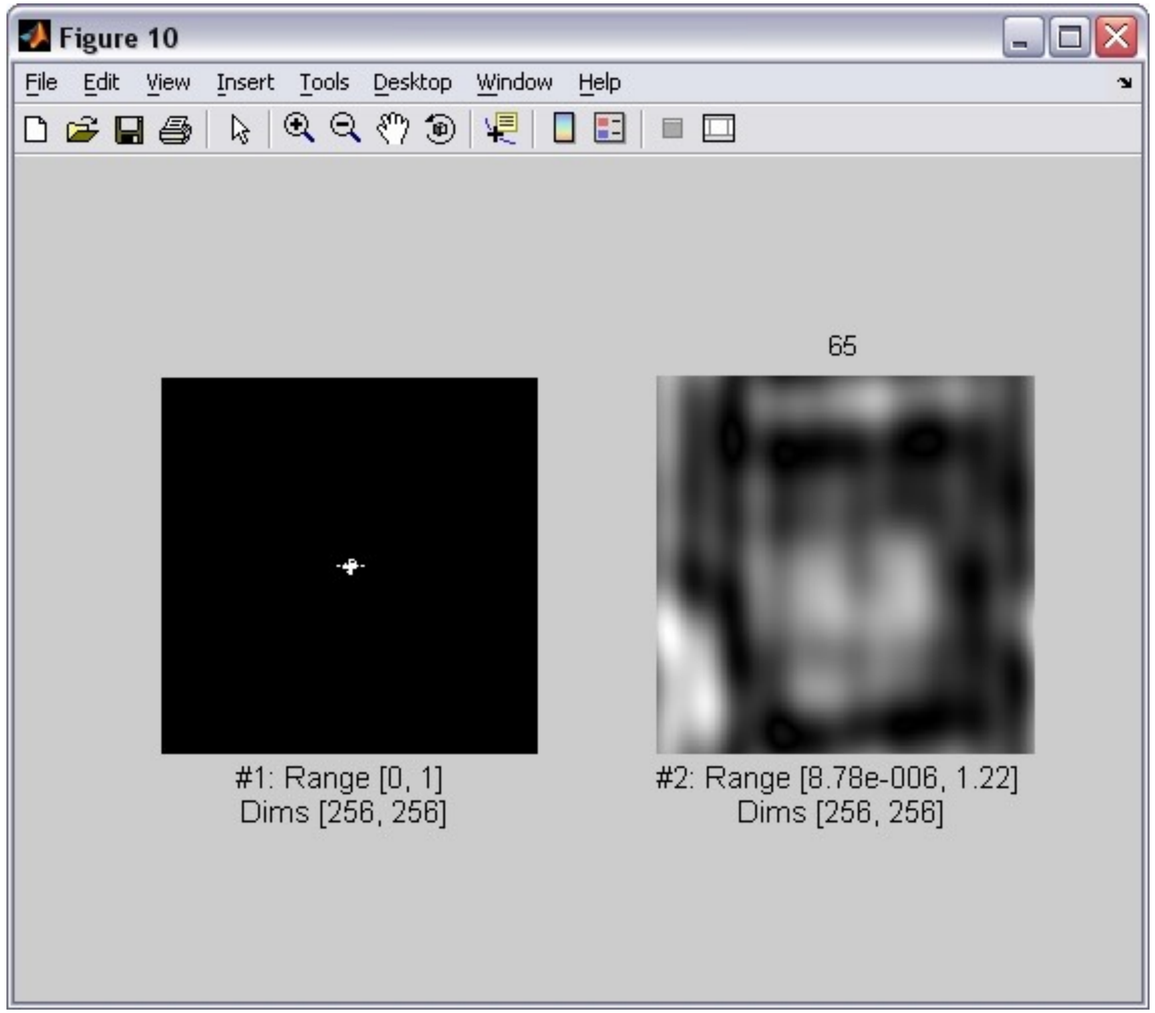


Figure 10
File Edit View Insert Tools Desktop Window Help
65
#1: Range [0, 1]
Dims [256, 256]
#2: Range [8.78e-006, 1.22]
Dims [256, 256]

129

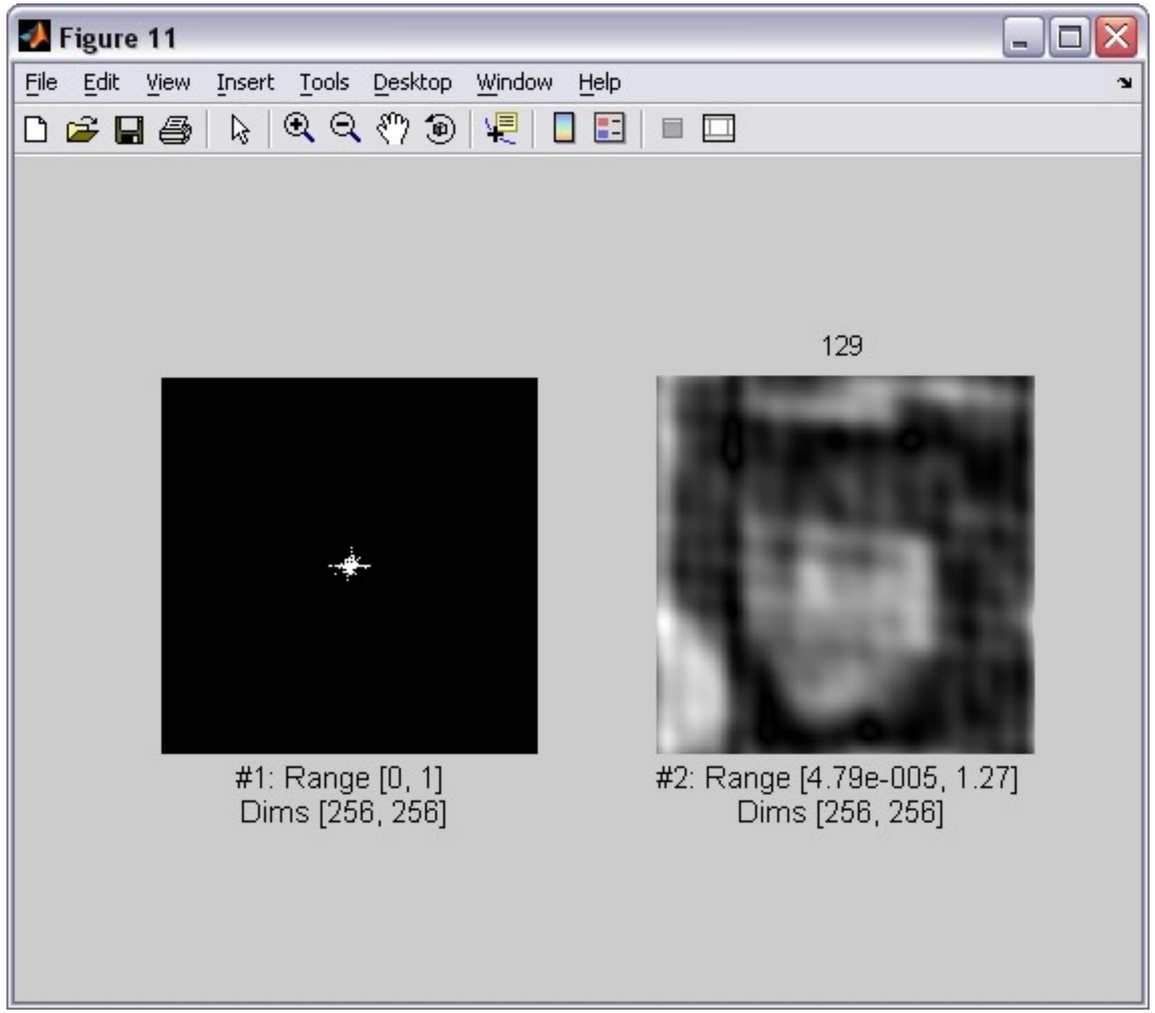


Figure 11
File Edit View Insert Tools Desktop Window Help
129
#1: Range [0, 1]
Dims [256, 256]
#2: Range [4.79e-005, 1.27]
Dims [256, 256]

257

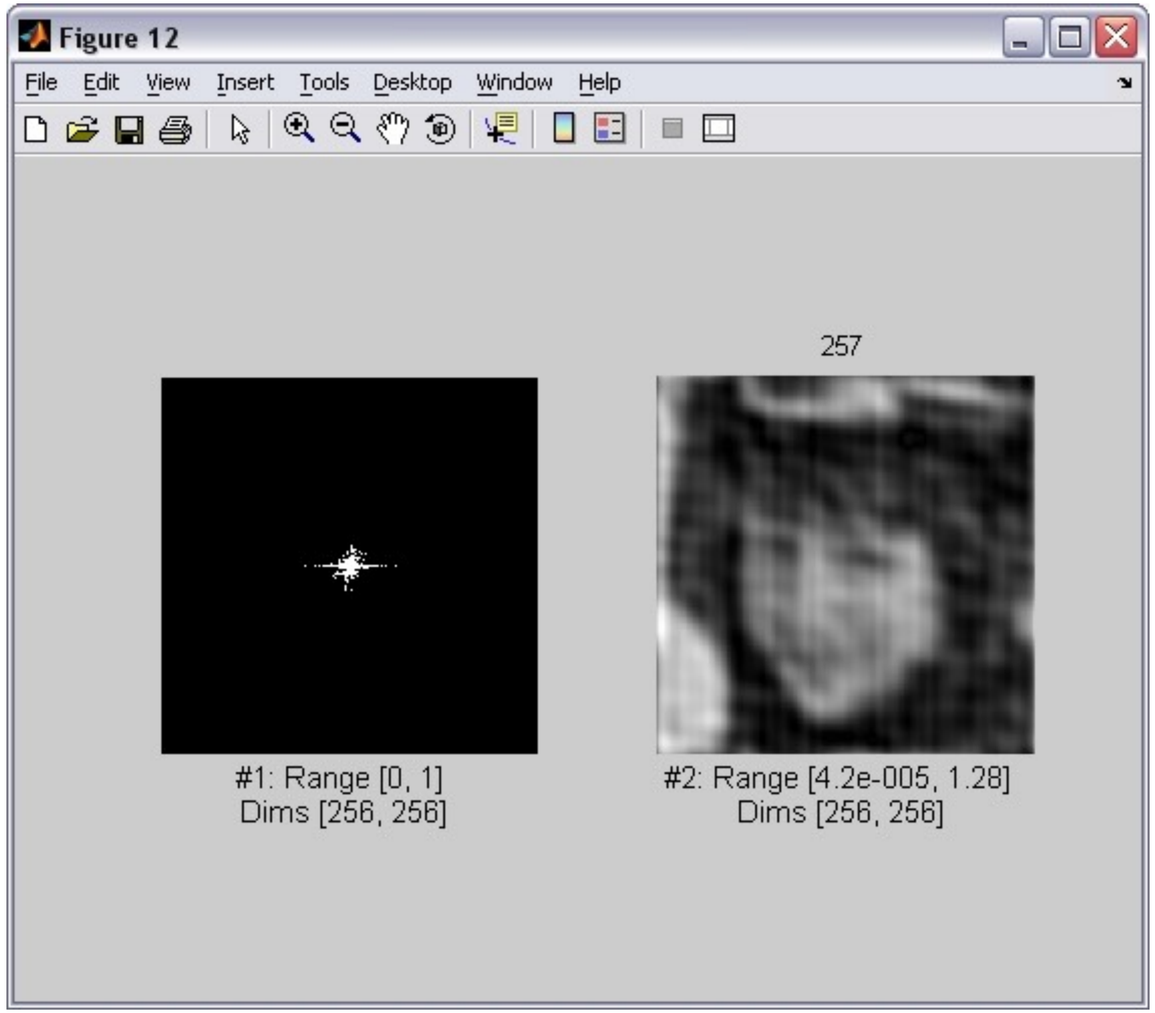


Figure 12
File Edit View Insert Tools Desktop Window Help
257
#1: Range [0, 1]
Dims [256, 256]
#2: Range [4.2e-005, 1.28]
Dims [256, 256]

513

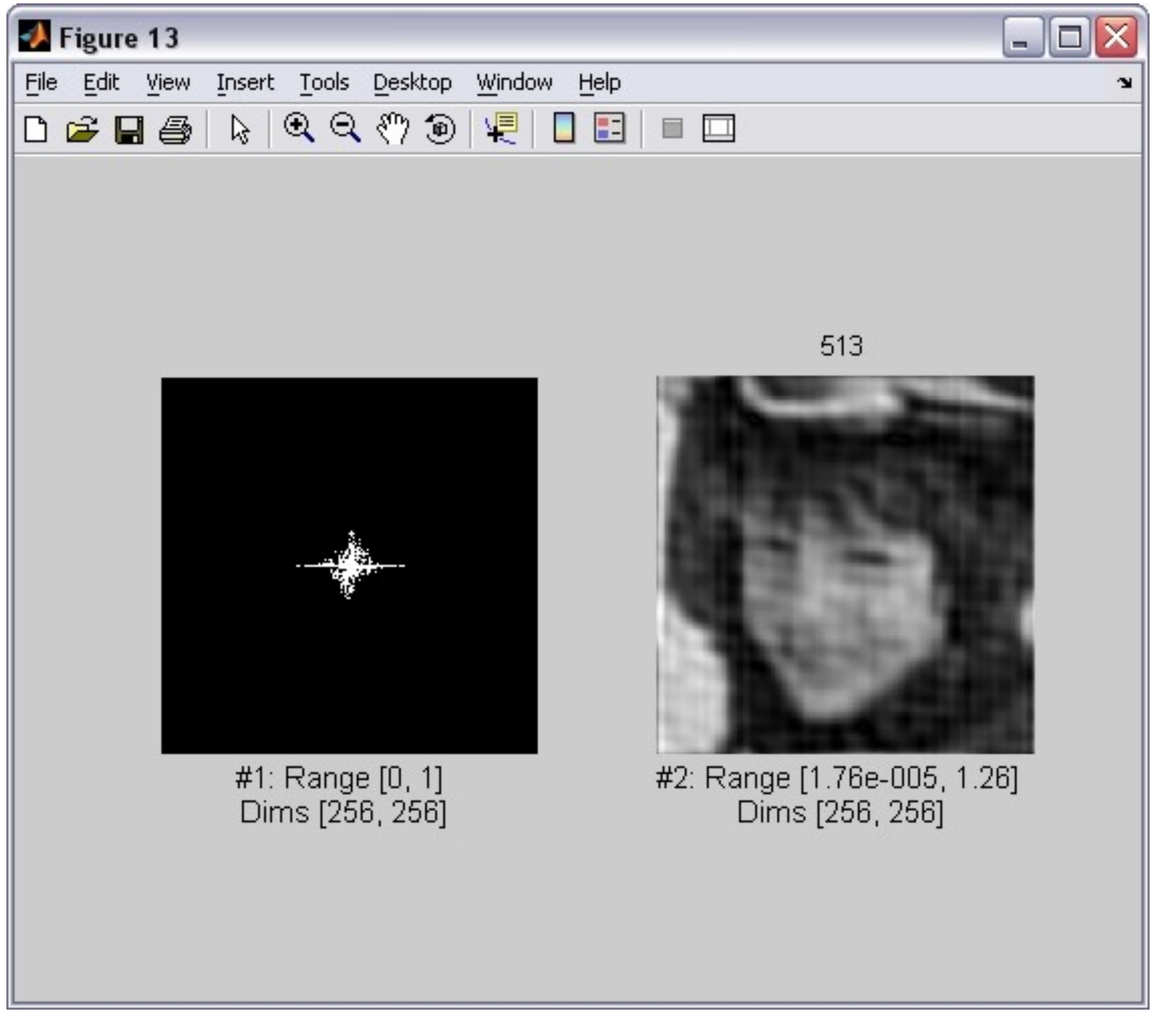


Figure 13
File Edit View Insert Tools Desktop Window Help
513
#1: Range [0, 1]
Dims [256, 256]
#2: Range [1.76e-005, 1.26]
Dims [256, 256]

1025

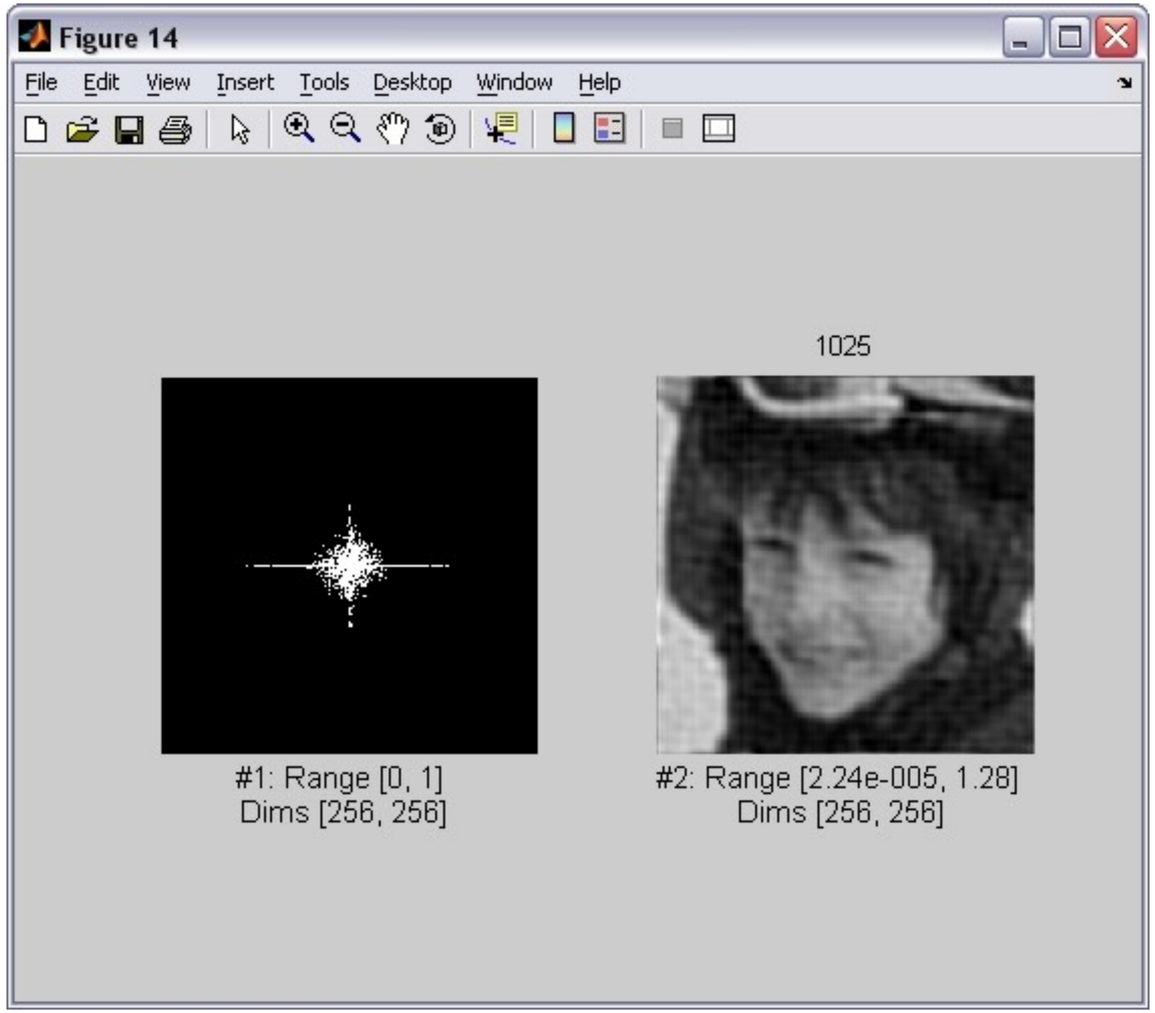


Figure 14
File Edit View Insert Tools Desktop Window Help
1025
#1: Range [0, 1]
Dims [256, 256]
#2: Range [2.24e-005, 1.28]
Dims [256, 256]

2049

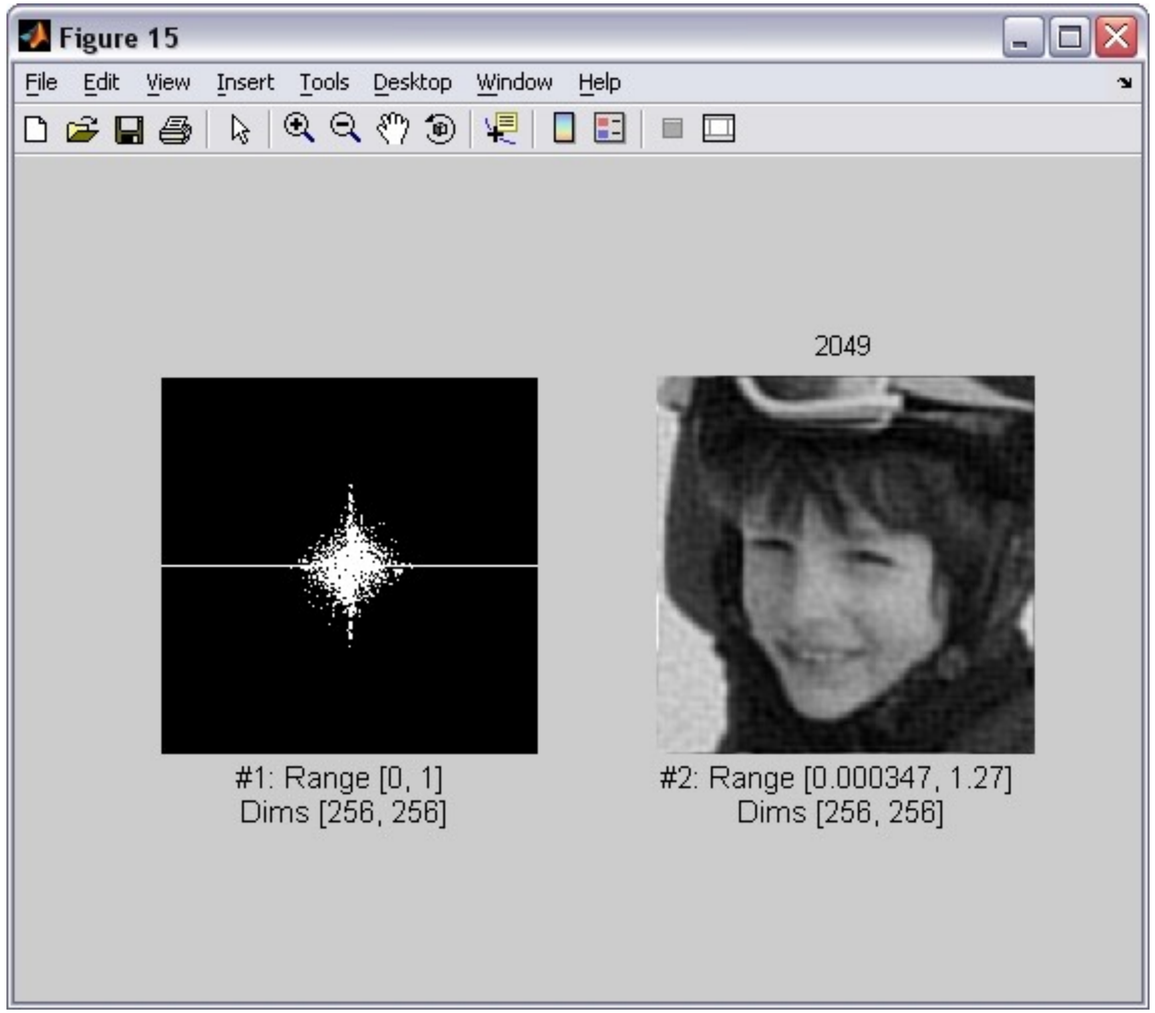


Figure 15
File Edit View Insert Tools Desktop Window Help
2049
#1: Range [0, 1]
Dims [256, 256]
#2: Range [0.000347, 1.27]
Dims [256, 256]

4097

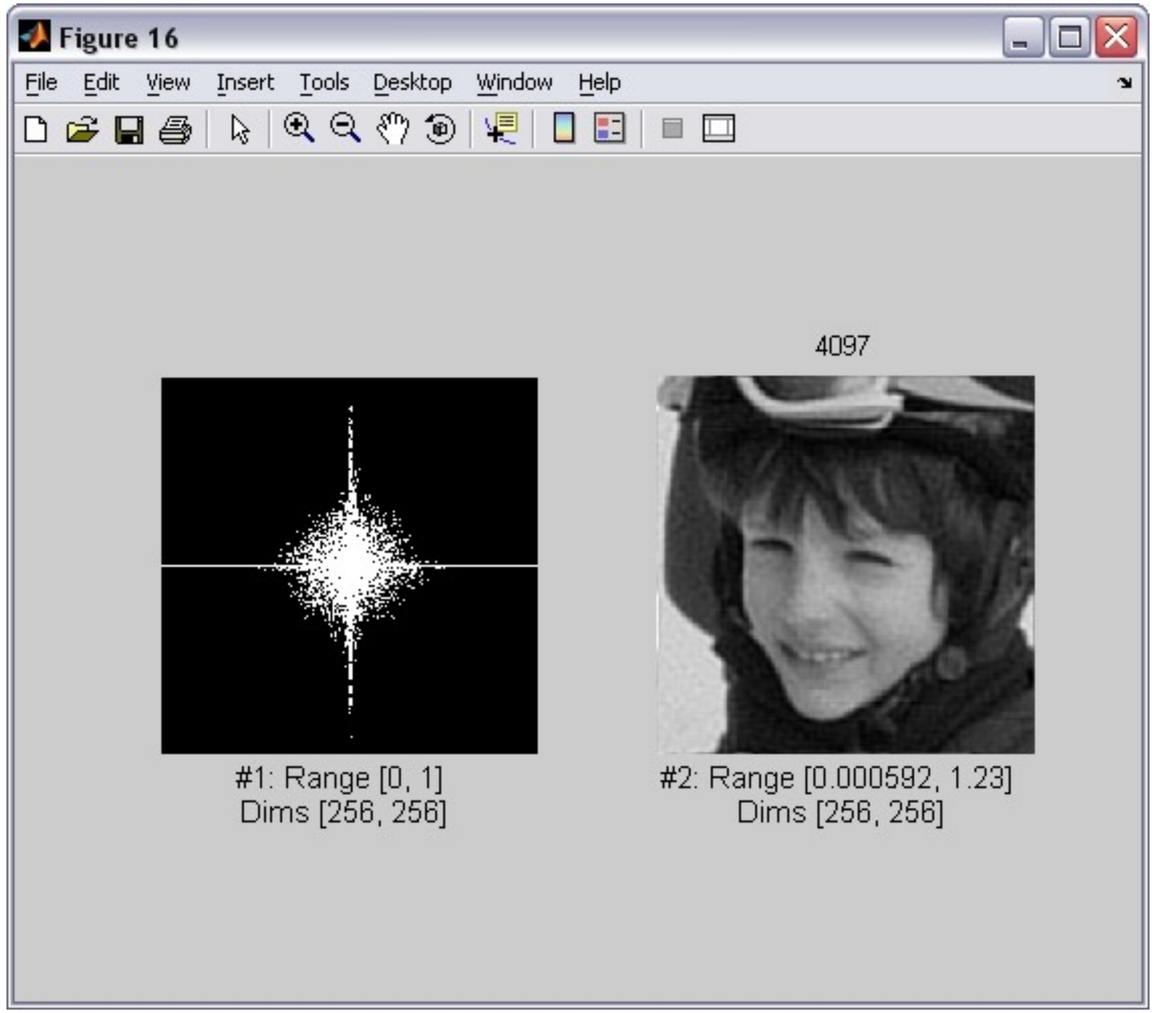


Figure 16
File Edit View Insert Tools Desktop Window Help
4097
#1: Range [0, 1]
Dims [256, 256]
#2: Range [0.000592, 1.23]
Dims [256, 256]

8193

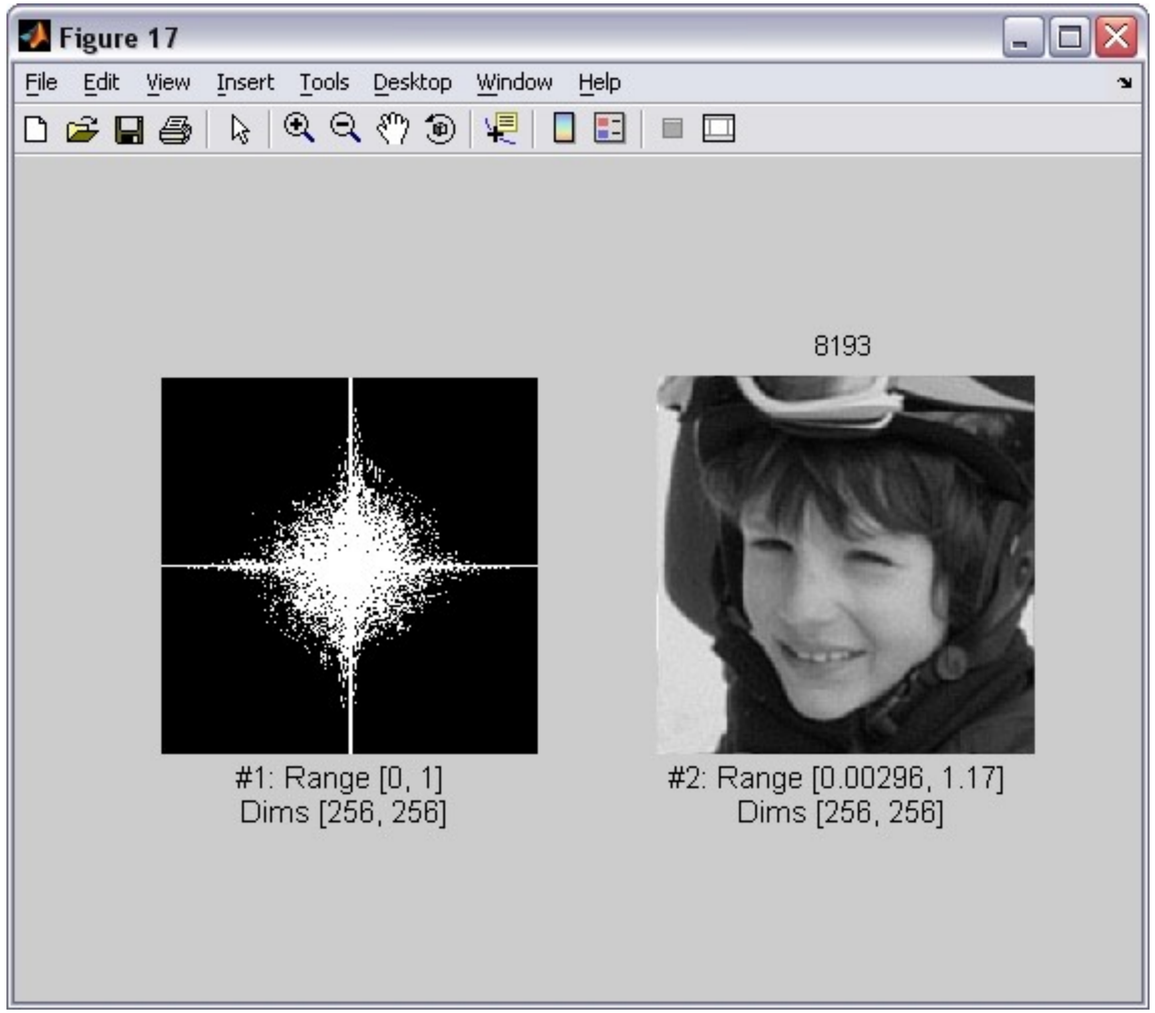


Figure 17
File Edit View Insert Tools Desktop Window Help
8193
#1: Range [0, 1]
Dims [256, 256]
#2: Range [0.00296, 1.17]
Dims [256, 256]

16385

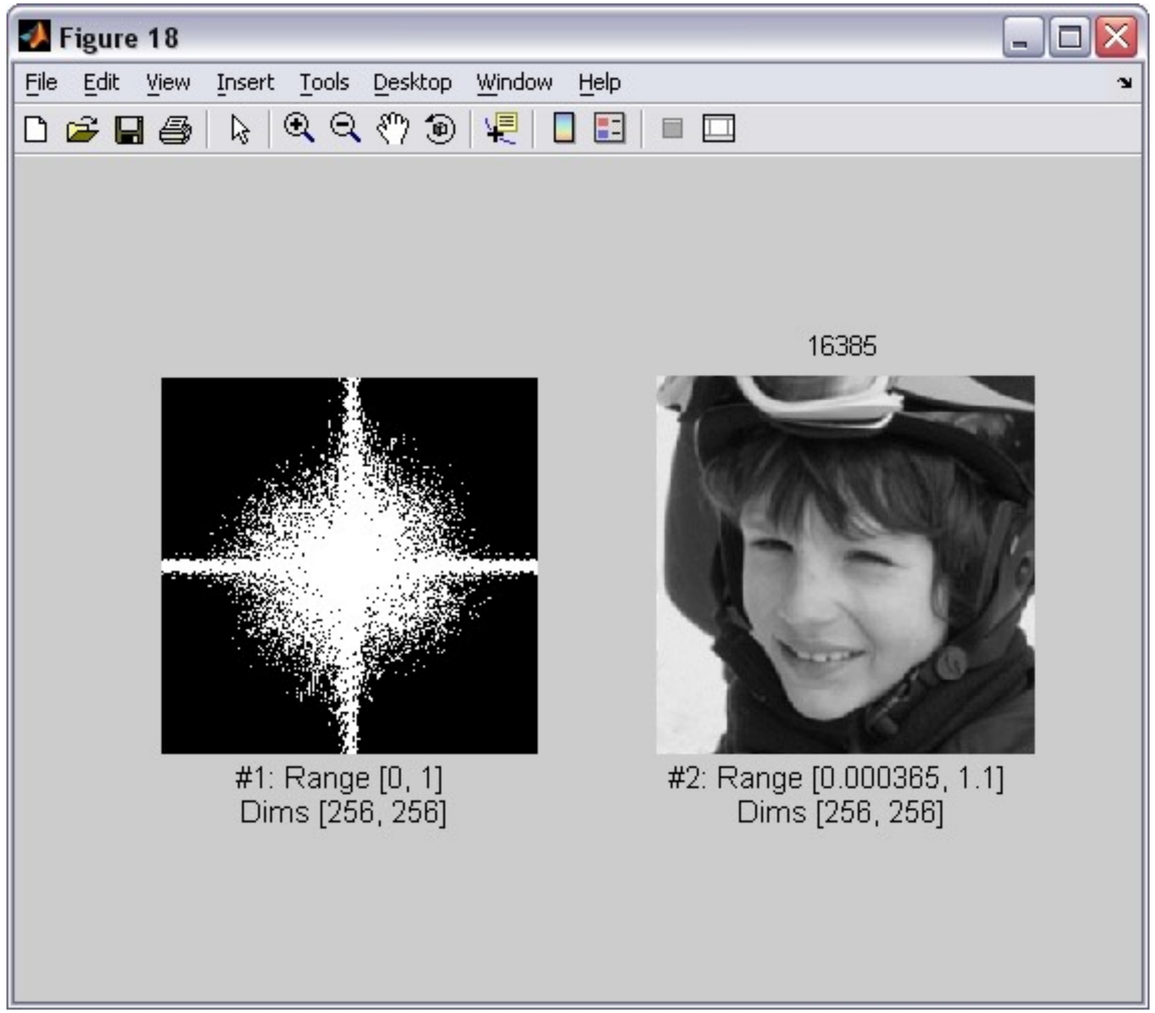


Figure 18
File Edit View Insert Tools Desktop Window Help
16385
#1: Range [0, 1]
Dims [256, 256]
#2: Range [0.000365, 1.1]
Dims [256, 256]

32769

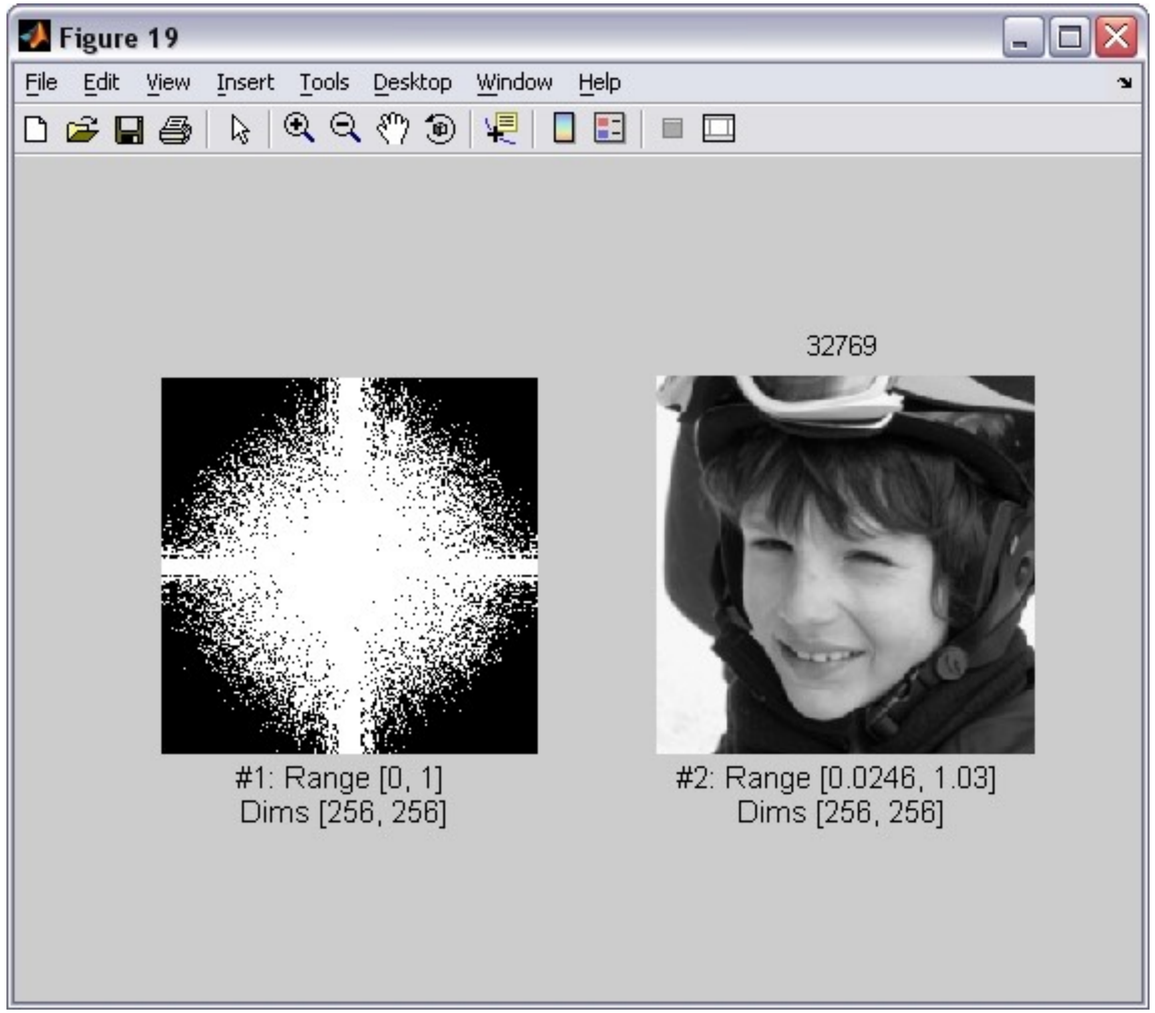


Figure 19
File Edit View Insert Tools Desktop Window Help
32769
#1: Range [0, 1]
Dims [256, 256]
#2: Range [0.0246, 1.03]
Dims [256, 256]

65536

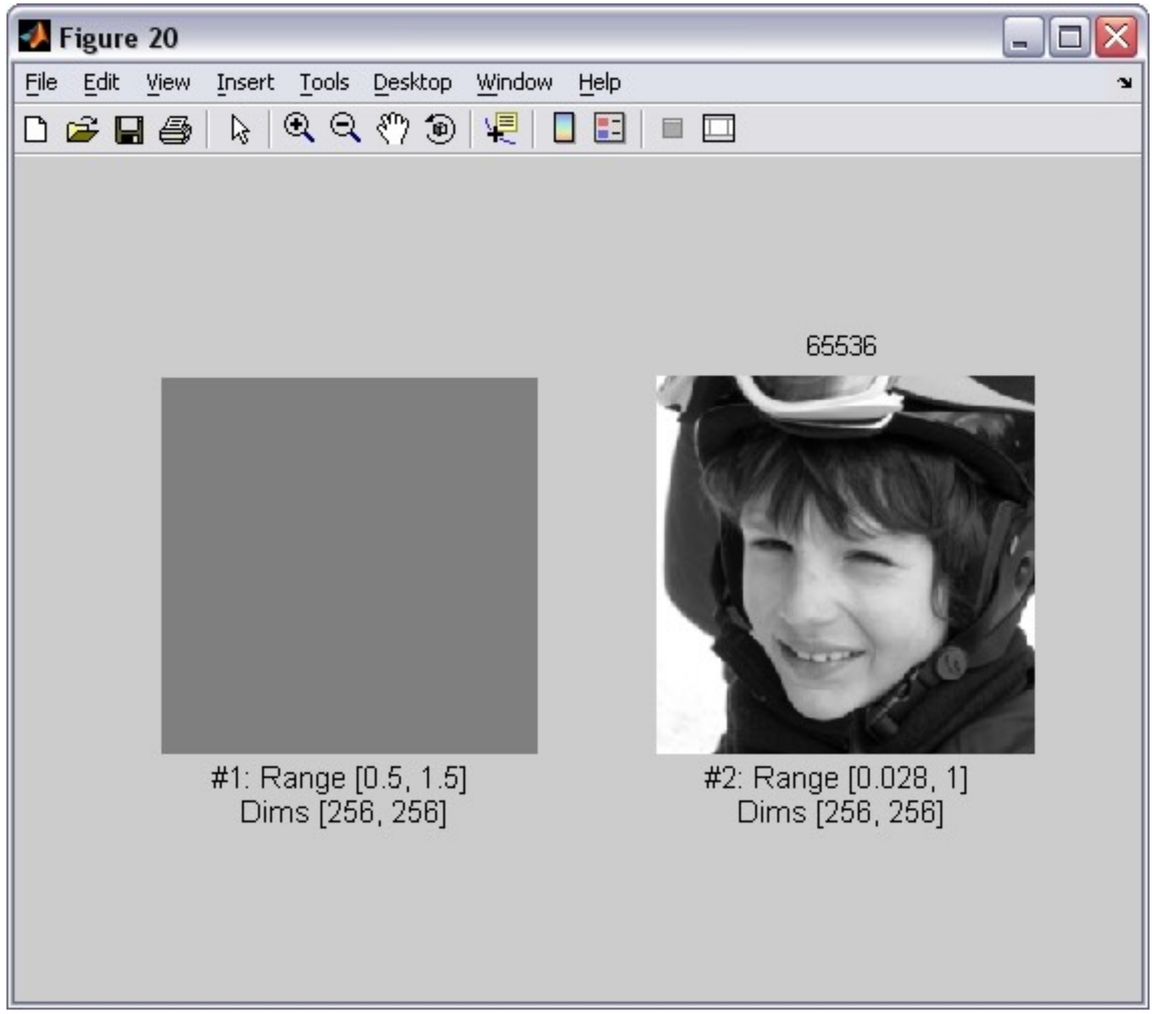
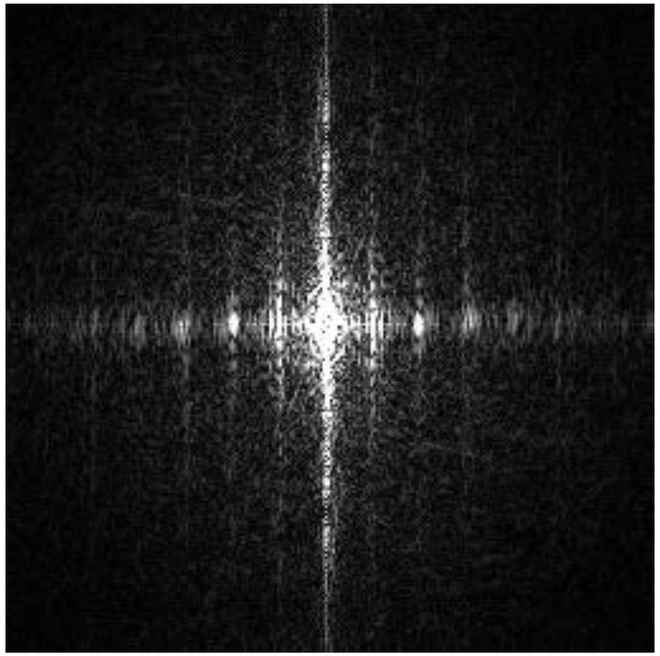
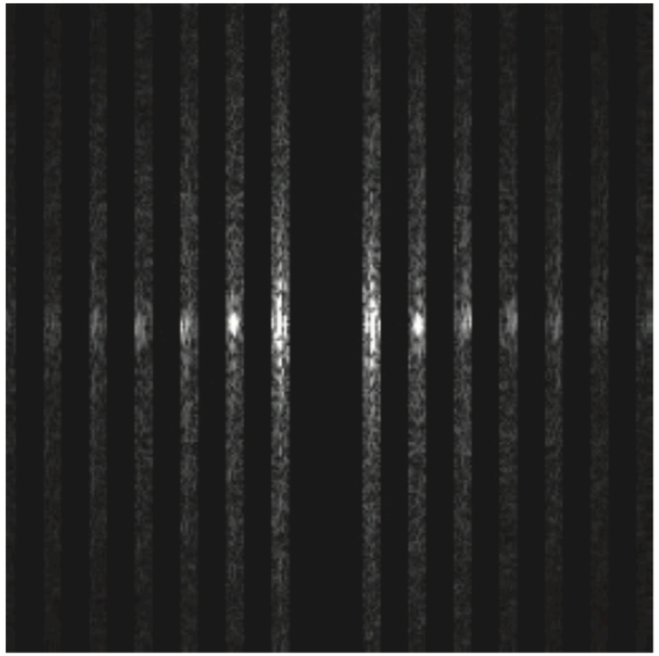


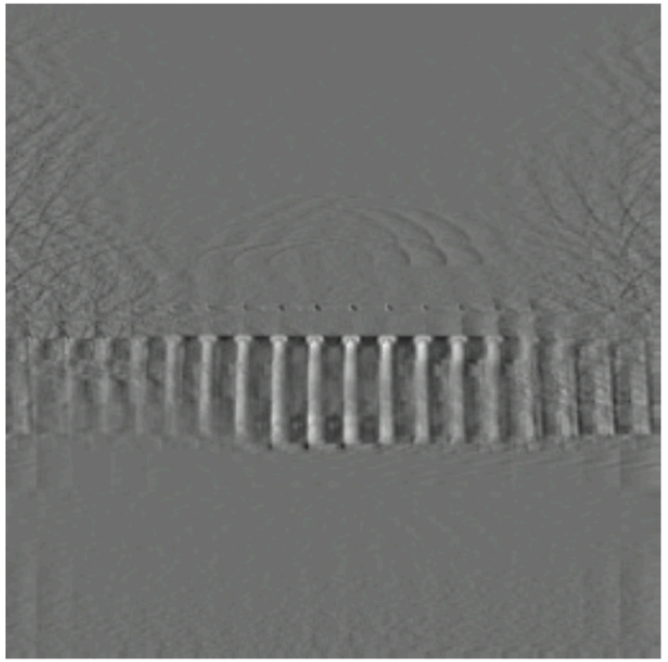
Figure 20
File Edit View Insert Tools Desktop Window Help
65536
#1: Range [0.5, 1.5]
Dims [256, 256]
#2: Range [0.028, 1]
Dims [256, 256]

DFT →

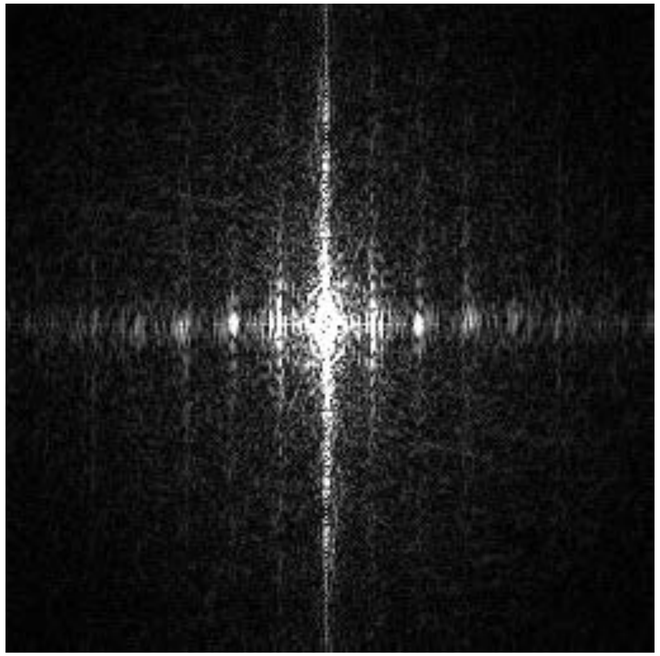


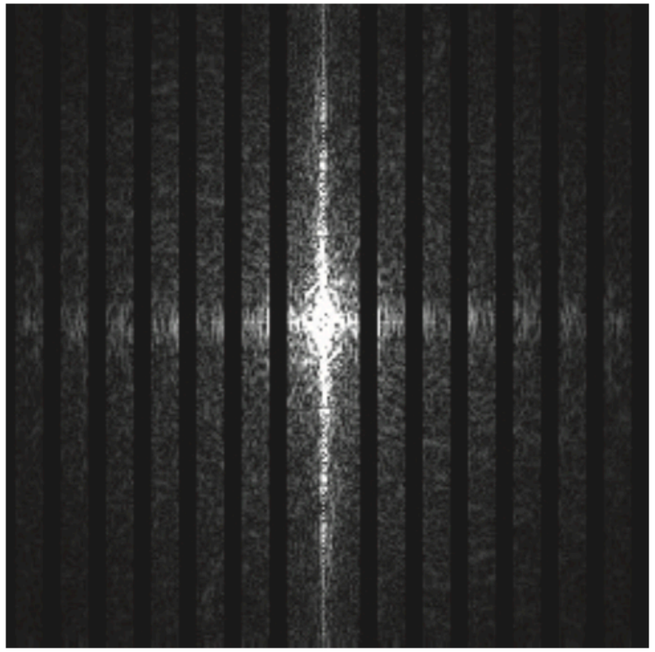


DFT^{-1} →



DFT →



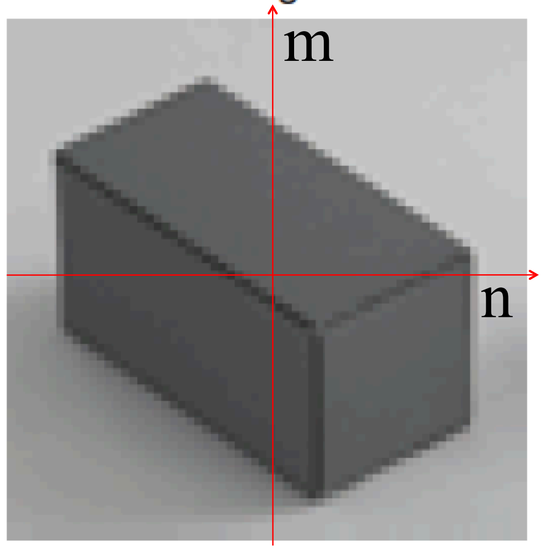


DFT^{-1} →

Visualizing the image Fourier transform

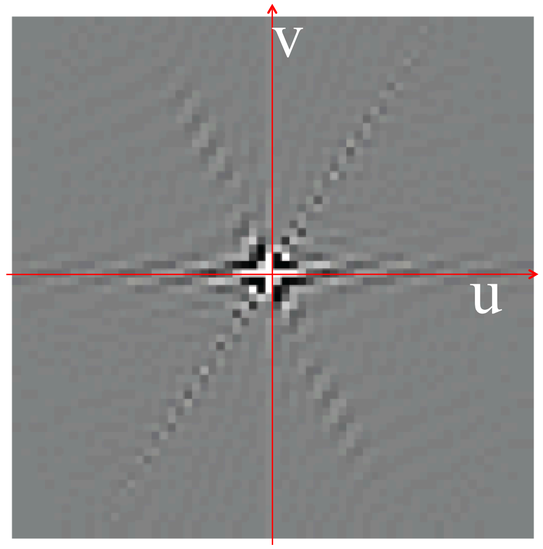
$f[n,m]$

image

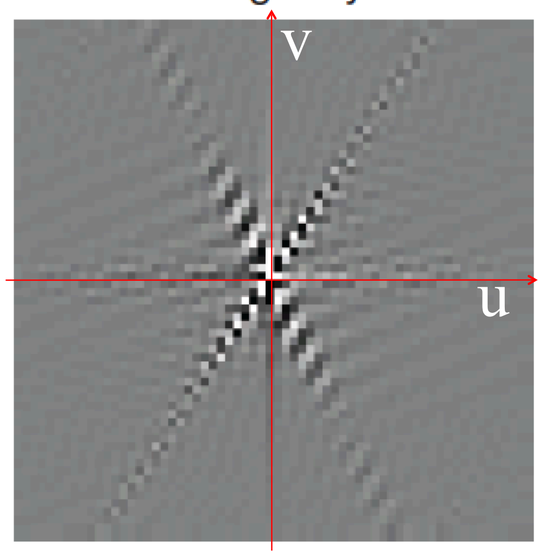


$F[u,v]$

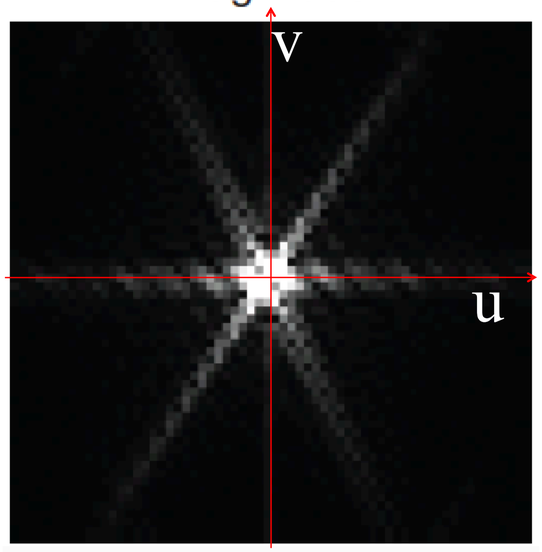
real



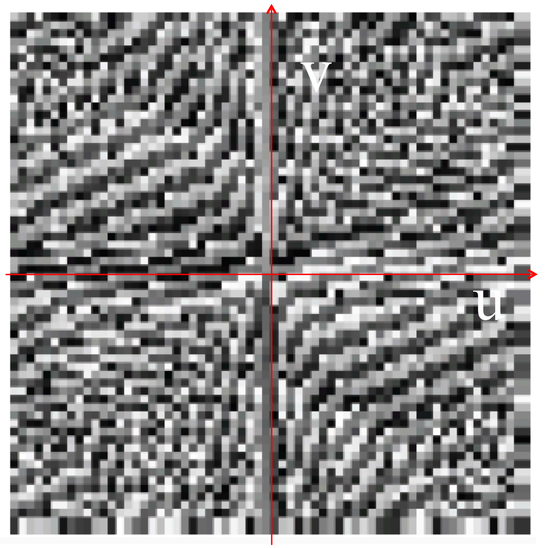
imaginary



magnitude

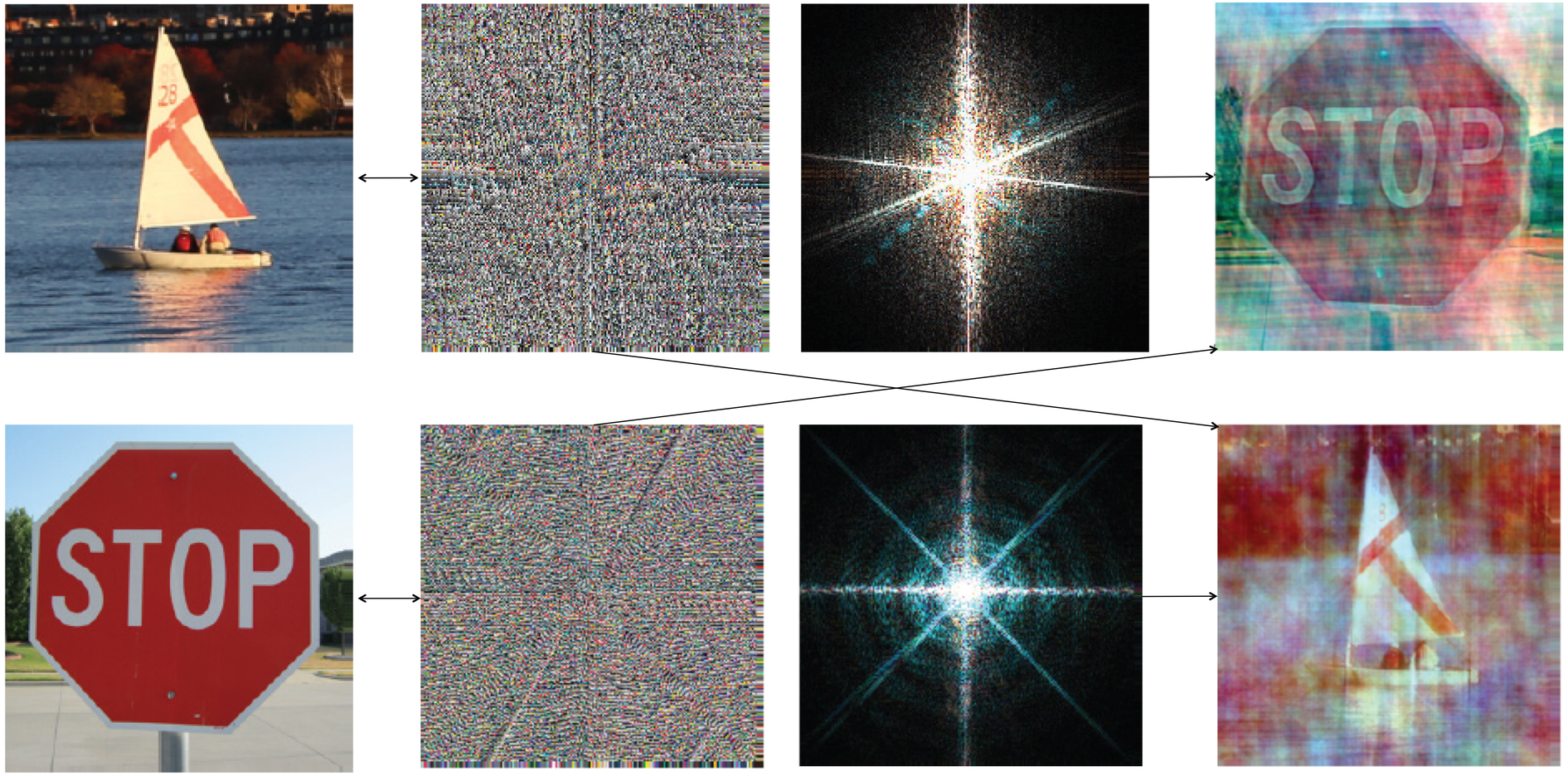


phase



Phase and Magnitude

$$F[u,v] = A[u,v]\exp(j\theta[u,v])$$

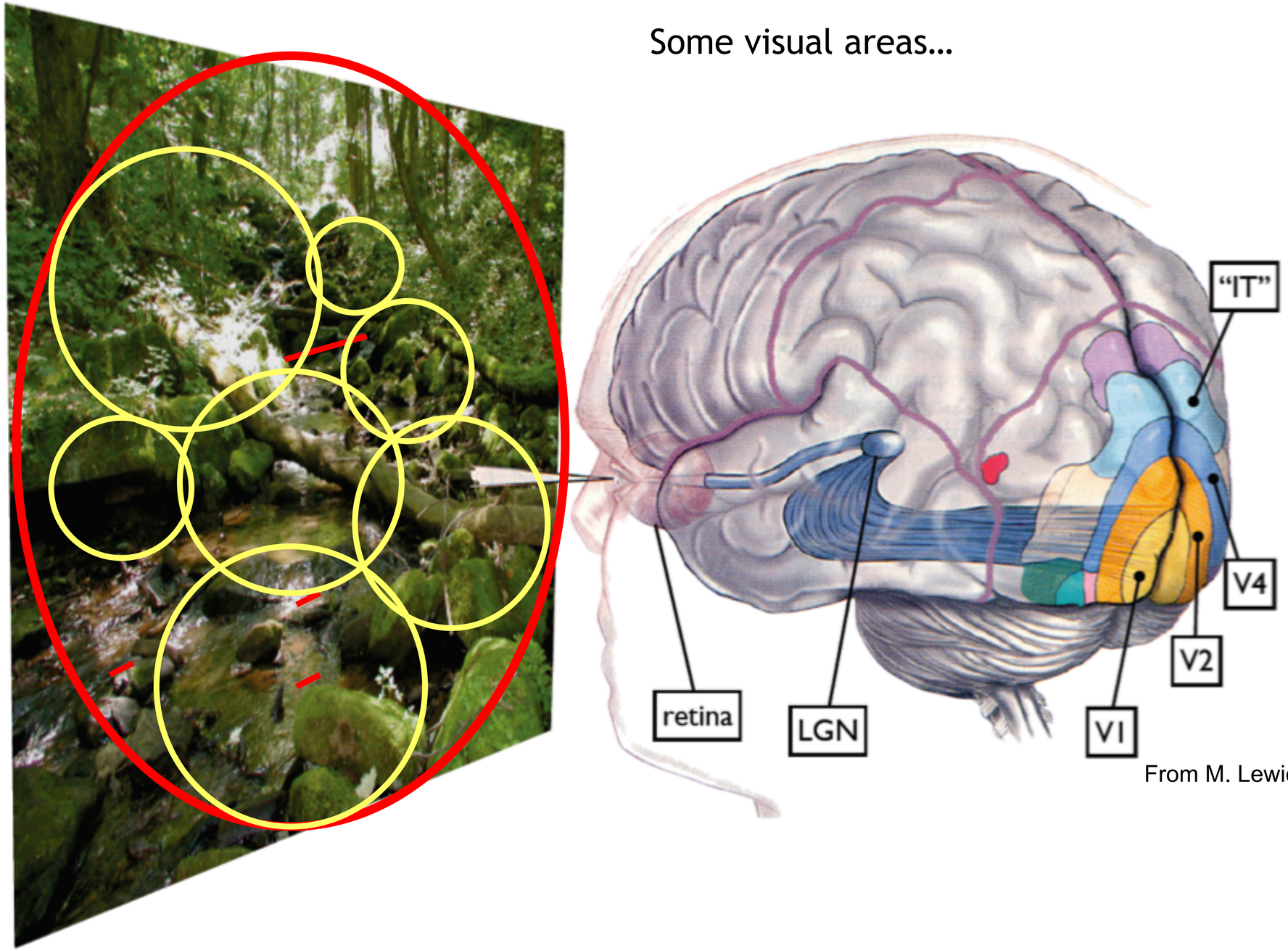


Each color channel is processed in the same way.

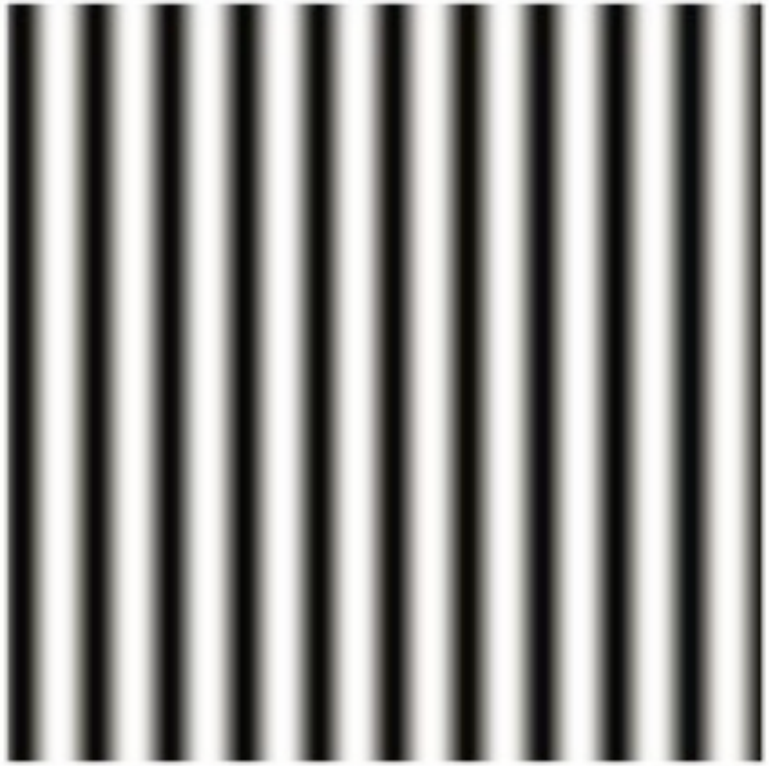
Phase and Magnitude

- Curious fact
  - all natural images have about the same magnitude transform
  - hence, phase seems to matter, but magnitude largely doesn't

Some visual areas…



From M. Lewicky



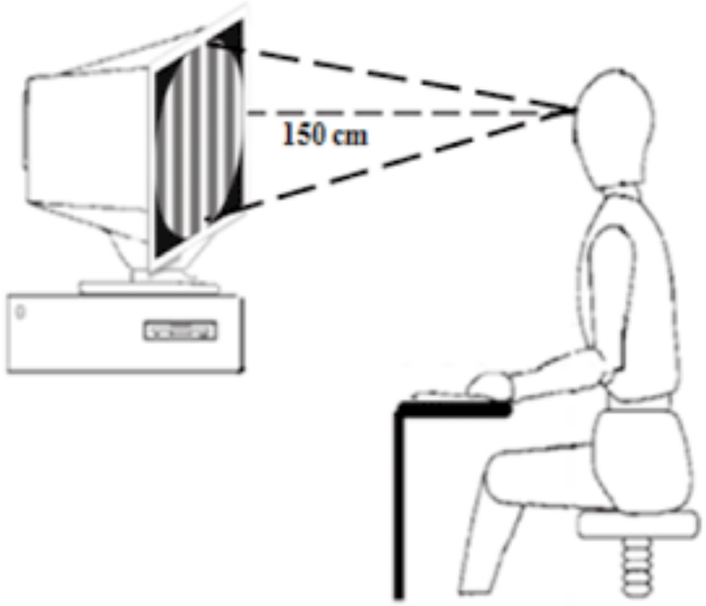


Figure 1. Stimulus presentation scheme. The stimuli were originally calibrated to be seen at a distance of 150 cm in a 19" display.

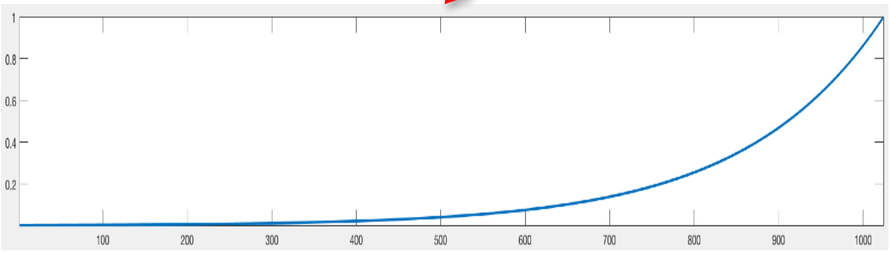
Campbell & Robson chart

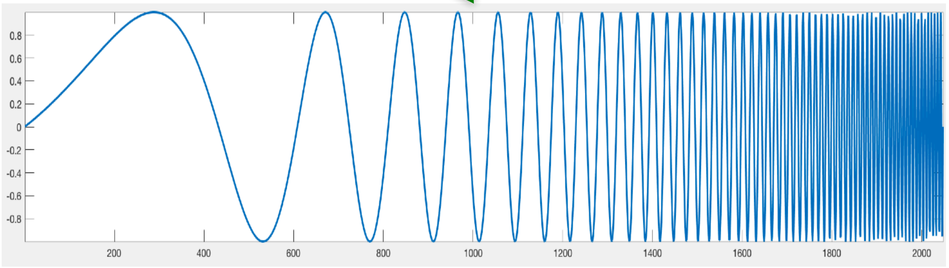
Let's define the following image:

$$\mathbf{I}[n,m] = \boxed{A[n]}\,\boxed{\sin(2\pi f[m]\,m/M)}$$

With:

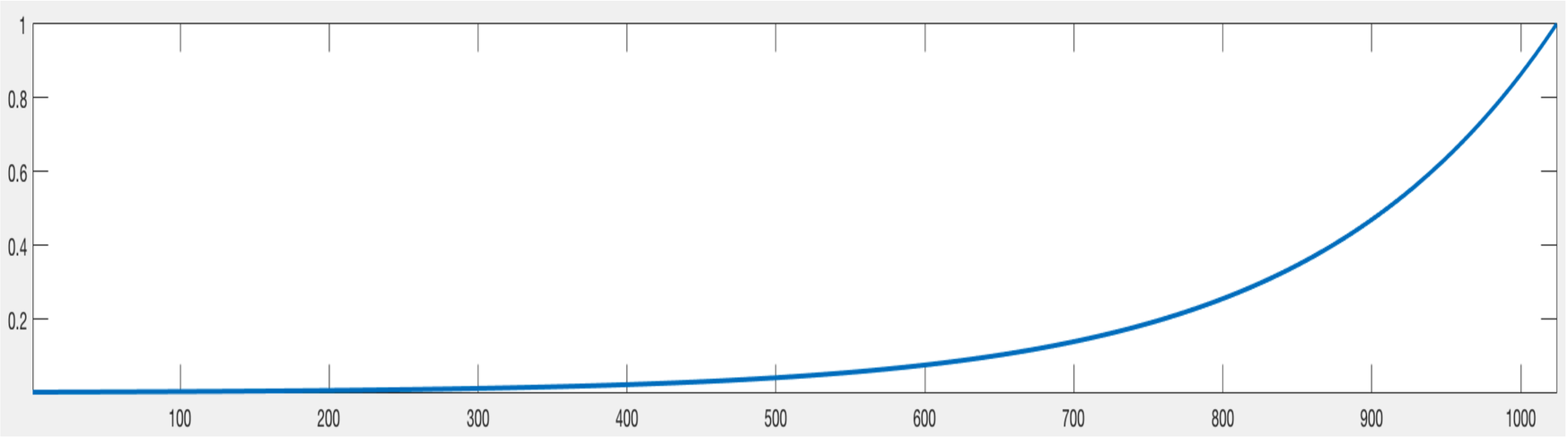
$$A[n] = A_{min}\left(\frac{A_{max}}{A_{min}}\right)^{n/N} \qquad f[m] = f_{min}\left(\frac{f_{max}}{f_{min}}\right)^{m/M}$$

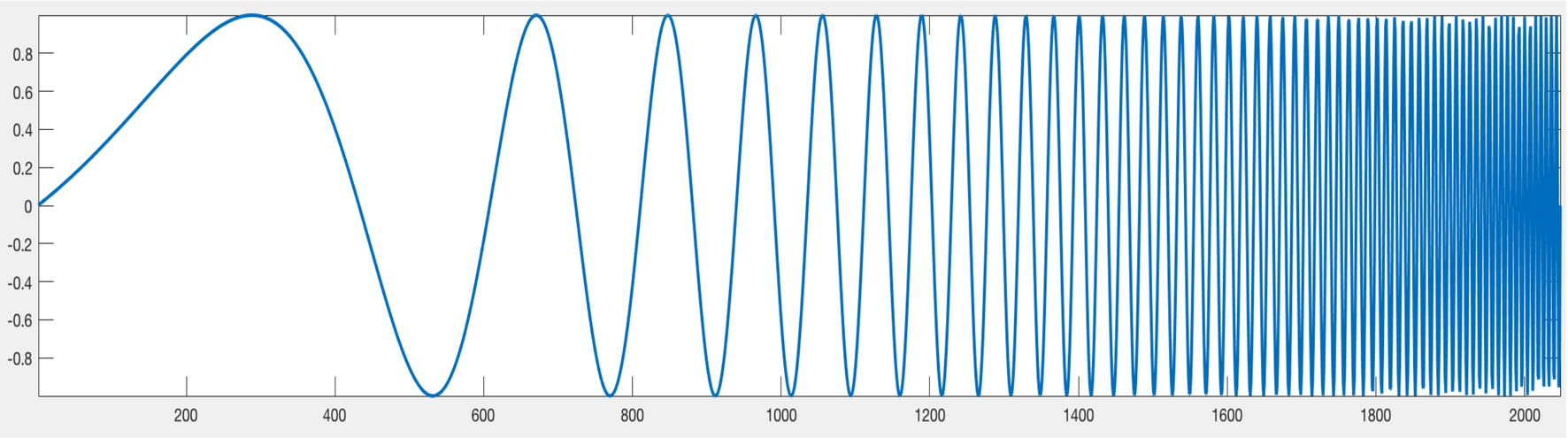




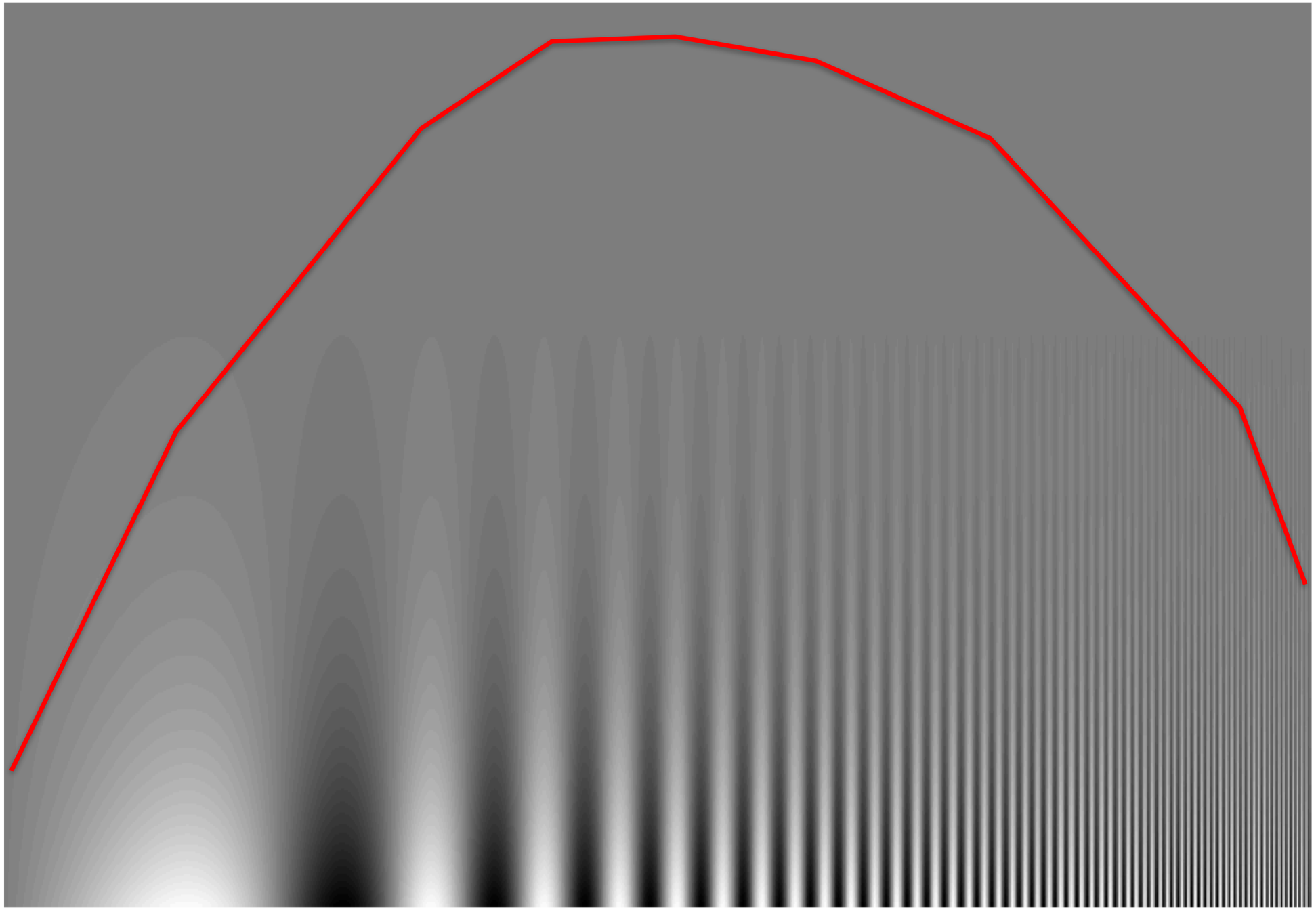
What do you think you should see when looking at this image?

$$\mathbf{I}[n,m] = A[n]\sin(2\pi f[m]\, m/M)$$





$$\mathbf{I}[n,m] = A[n]\sin(2\pi f[m]\, m/M)$$

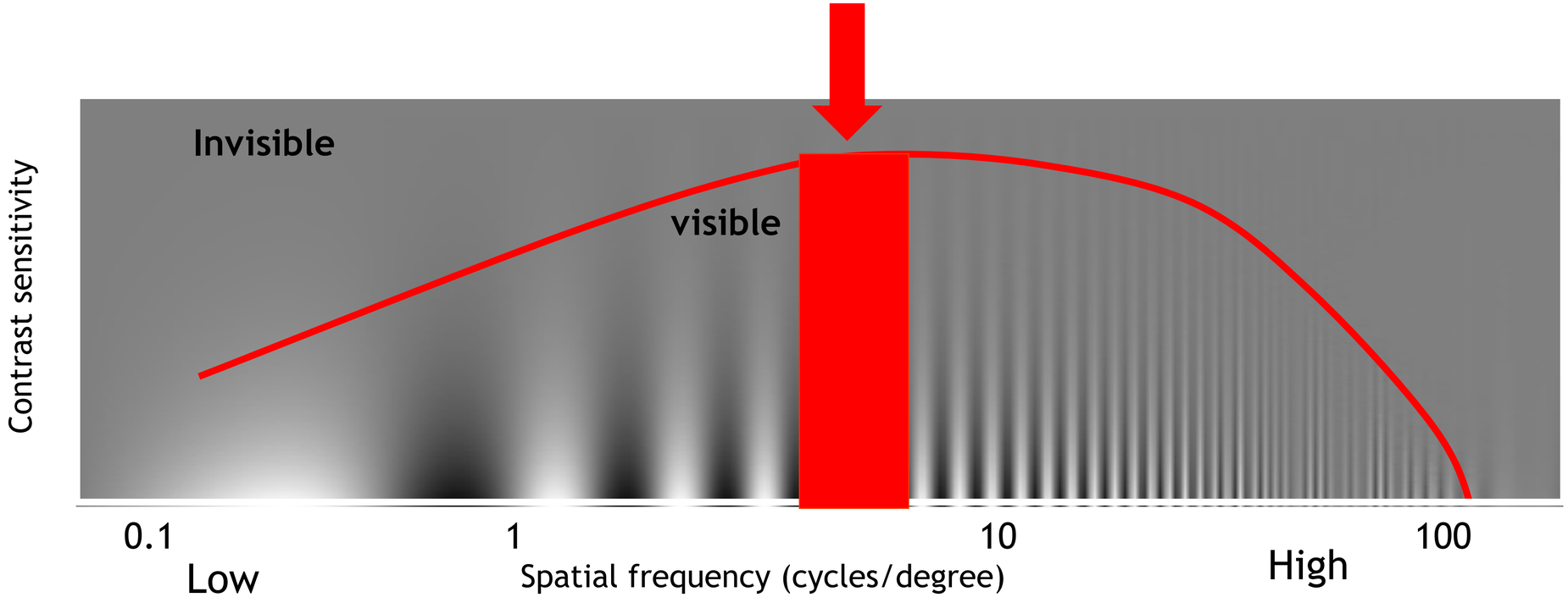


Contrast Sensitivity Function

Blackmore & Campbell (1969)

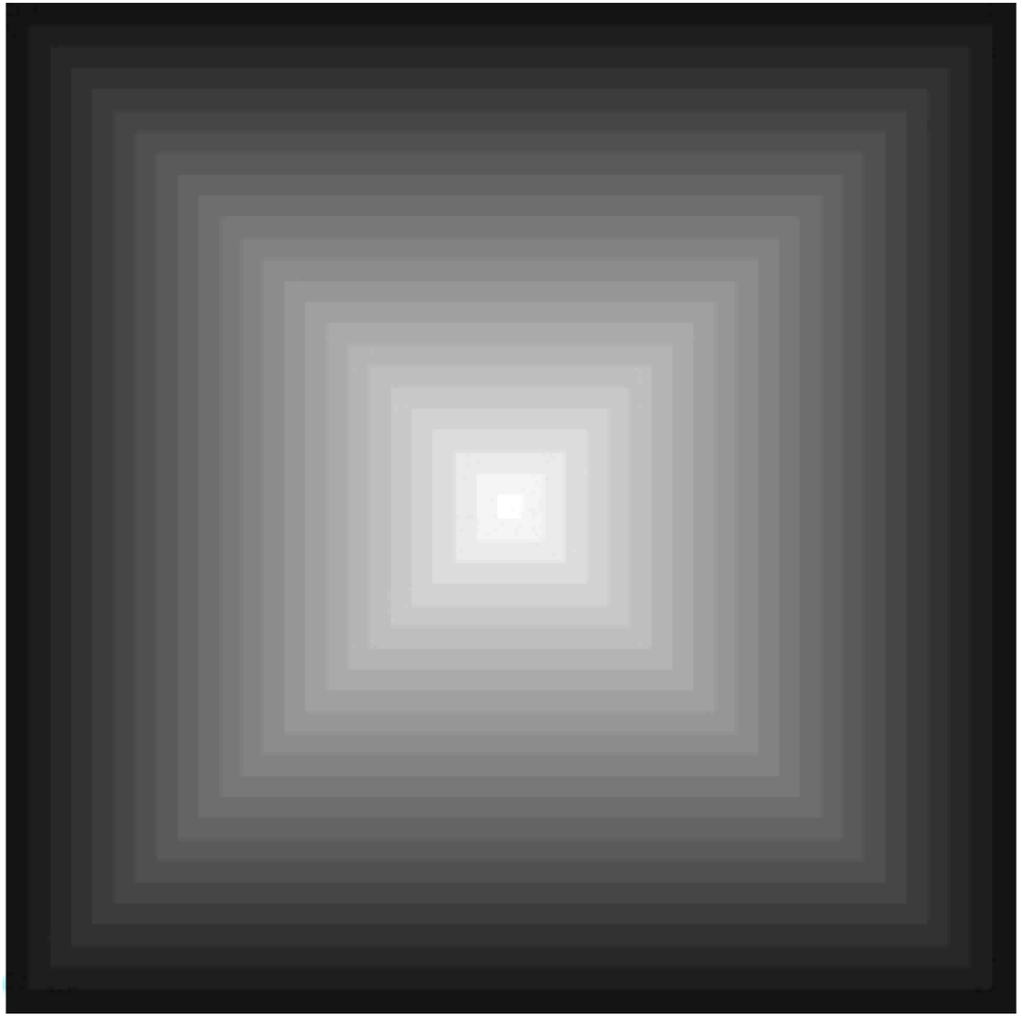
Maximum sensitivity

~ **6** cycles / degree of visual angle

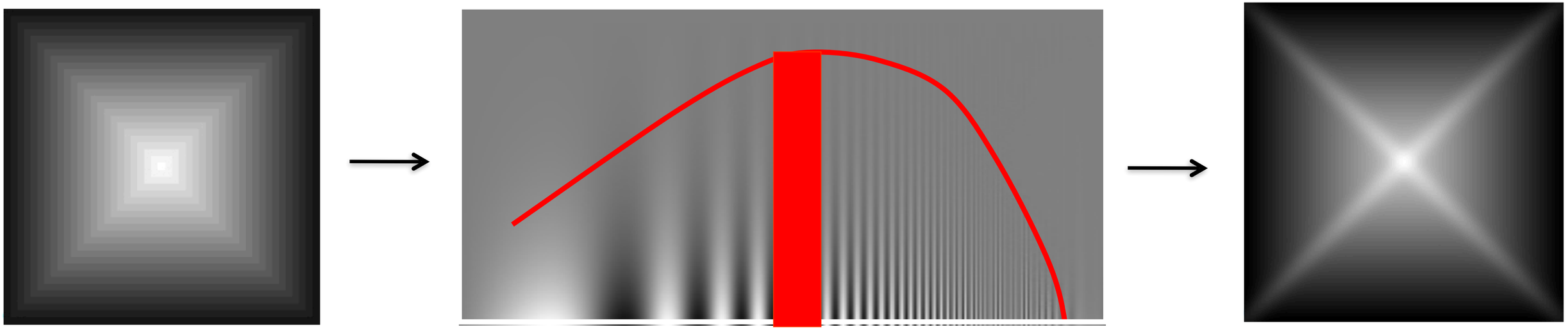


Things that are very close and/or large are hard to see

Things far away are hard to see



Vasarely visual illusion



Horizontal section