Transform Methods & Signal Processing

lecture 04

Matthew Roughan
<matthew.roughan@adelaide.edu.au>

Discipline of Applied Mathematics
School of Mathematical Sciences
University of Adelaide

October 26, 2009

Transforms in 2D

Until now we have only considered 1D functions as possible inputs, but there are many applications where we want to consider functions with 2 (or more) independent variables, e.g., surfaces, and the Fourier transform naturally generalizes to this case.

Transforms in 2D

We want a separable basis.

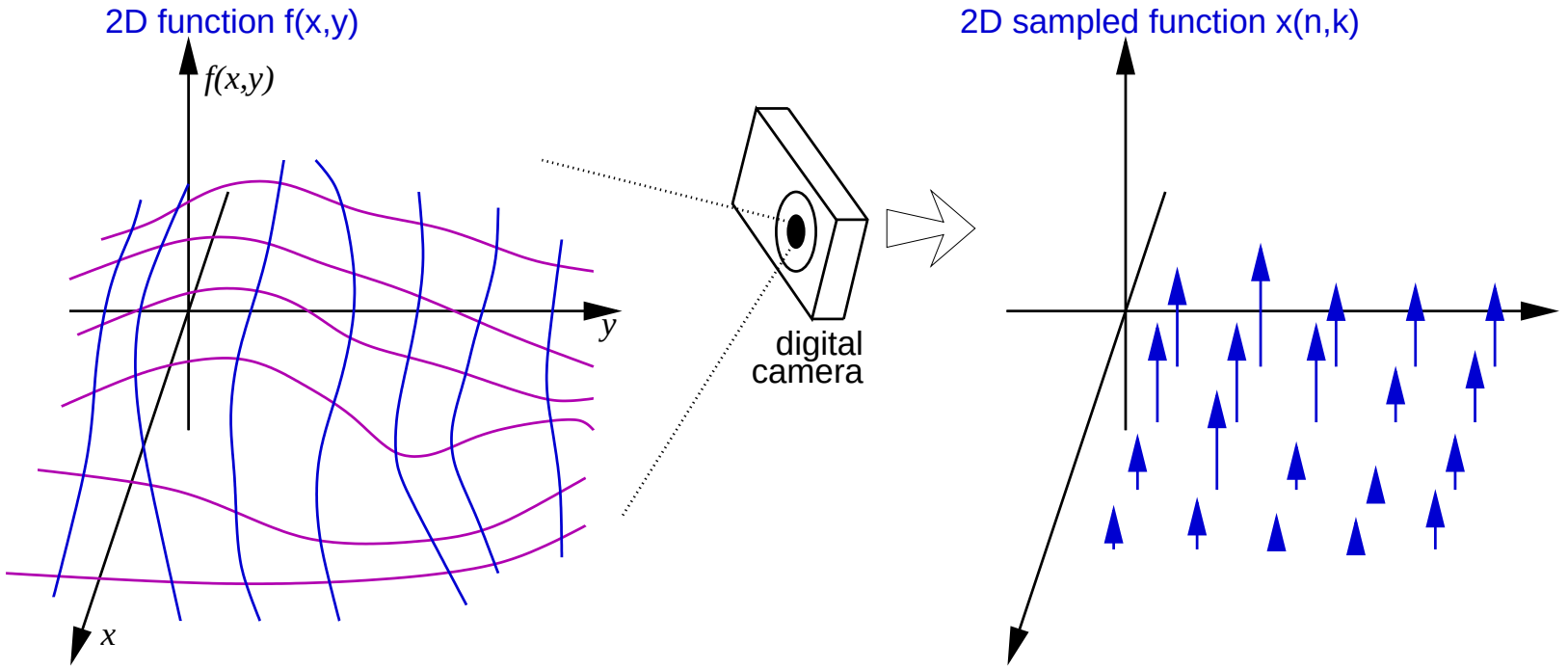
- basis "vectors" are the product of basis vectors from two component subspaces. e.g. if the function can be written as $f(x,y) = g(x)h(y)$, we should be able to write its FT as $F(s,t) = G(s)H(t)$.
- makes computation easier
- makes most of the math easier (same methods used for proofs)

Fourier transform in 2D

$$F(s,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(sx+ty)}\,dx\,dy$$

2D signals = images

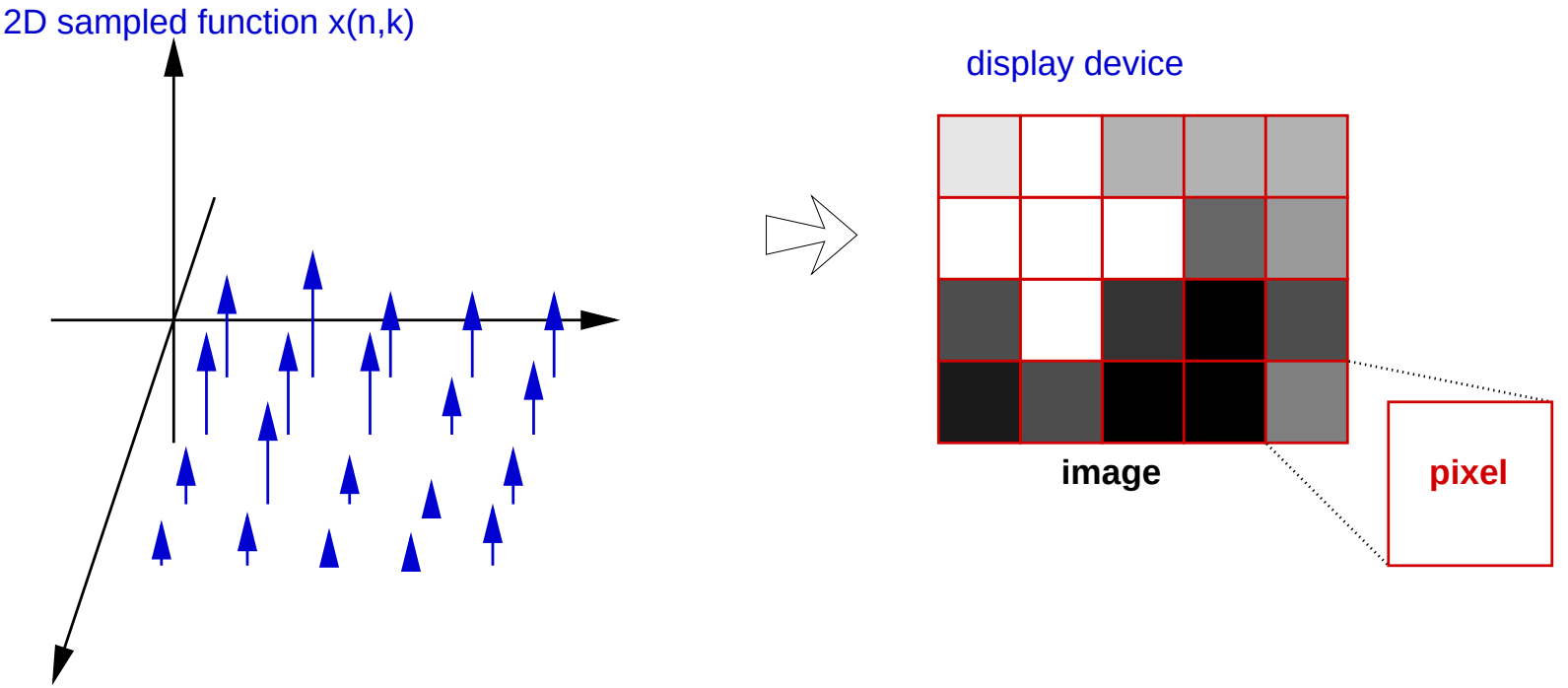
- 2D signal processing is used for images
- images are sampled signals in 2D



- same issues for sampling/quantization as we have for 1D signals

Displaying 2D signals

- 2D signal processing is used for images
- images are sampled signals in 2D



The DFT in 2D

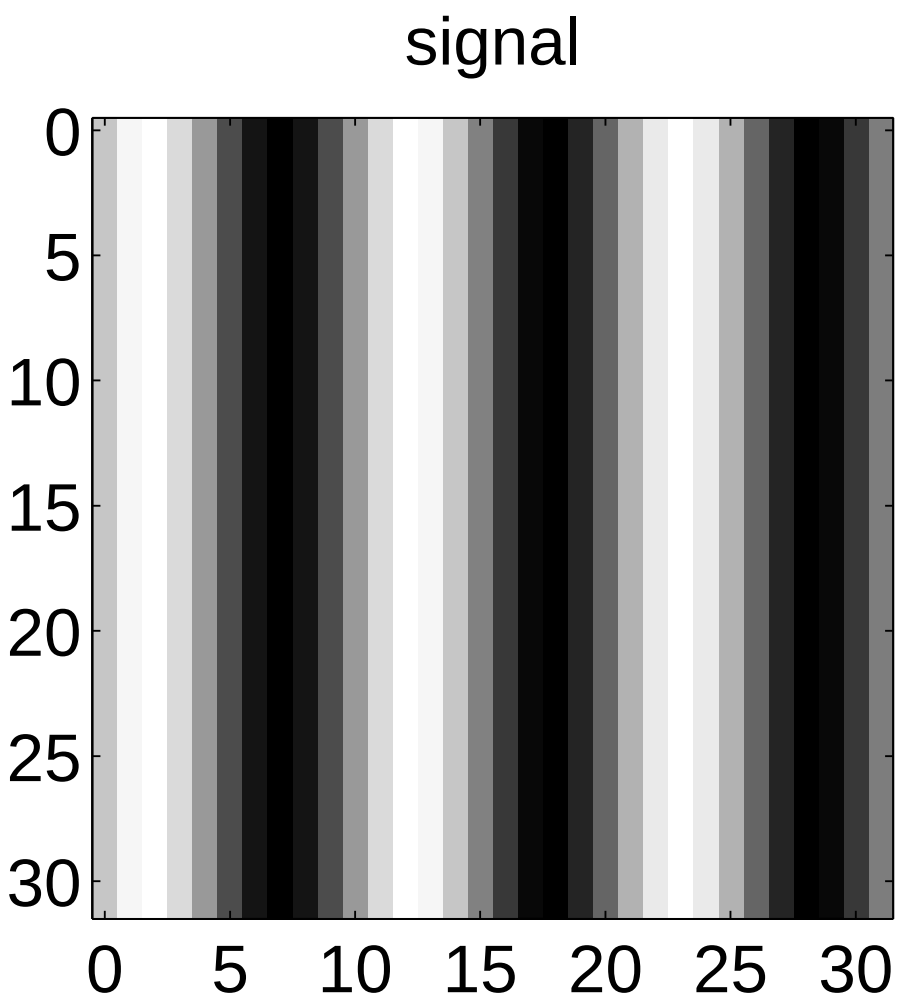
DFT

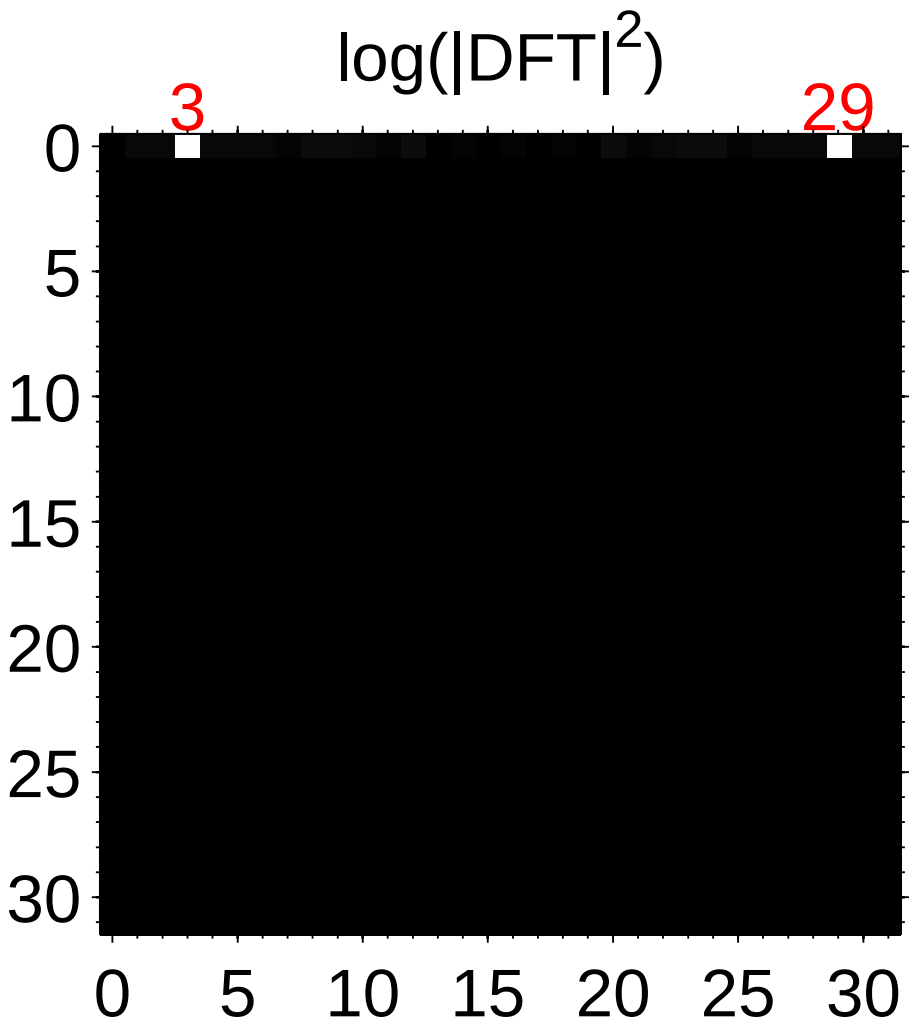
$$X(k_1,k_2) = \sum_{n_1=0}^{N_1-1}\sum_{n_2=0}^{N_2-1} x(n_1,n_2)e^{-i2\pi k_1 n_1/N_1}e^{-i2\pi k_2 n_2/N_2},$$

- To compute it efficiently:
  1. compute 1D FFT along the rows
  2. then do a 1D FFT along the columns
- Called **row-column** algorithm
  - note that the order could change.
- naturally generalizes to higher dimensions

Examples (i)

$$x(n,k) = \sin(2\pi \mathbf{3} k/N)$$

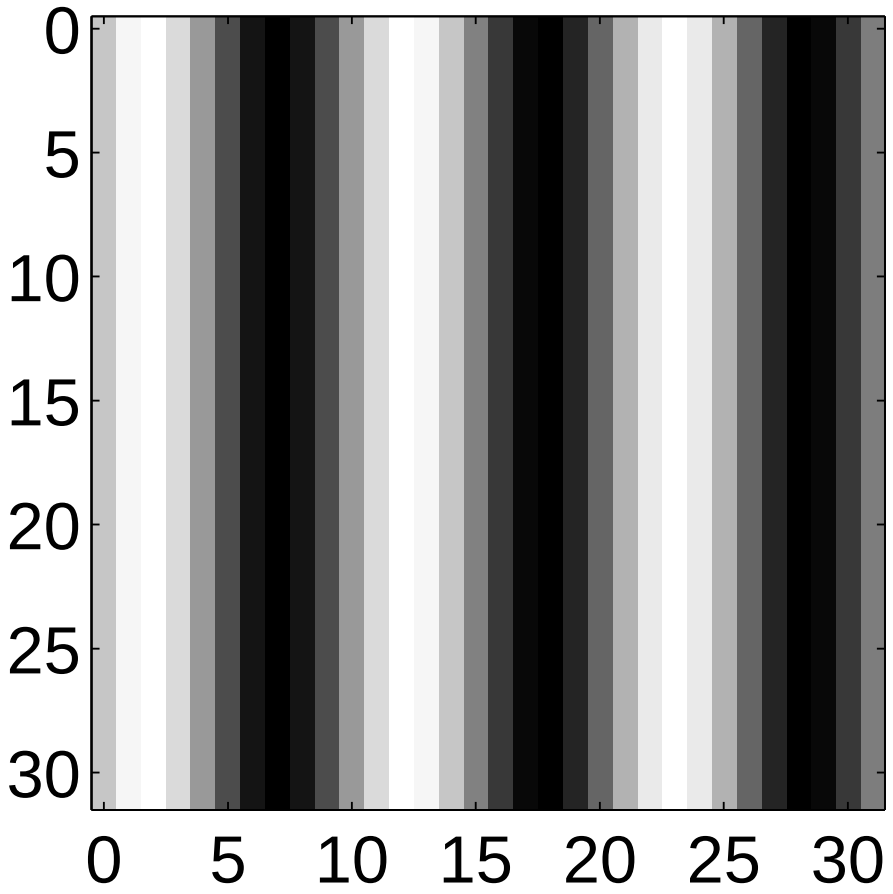




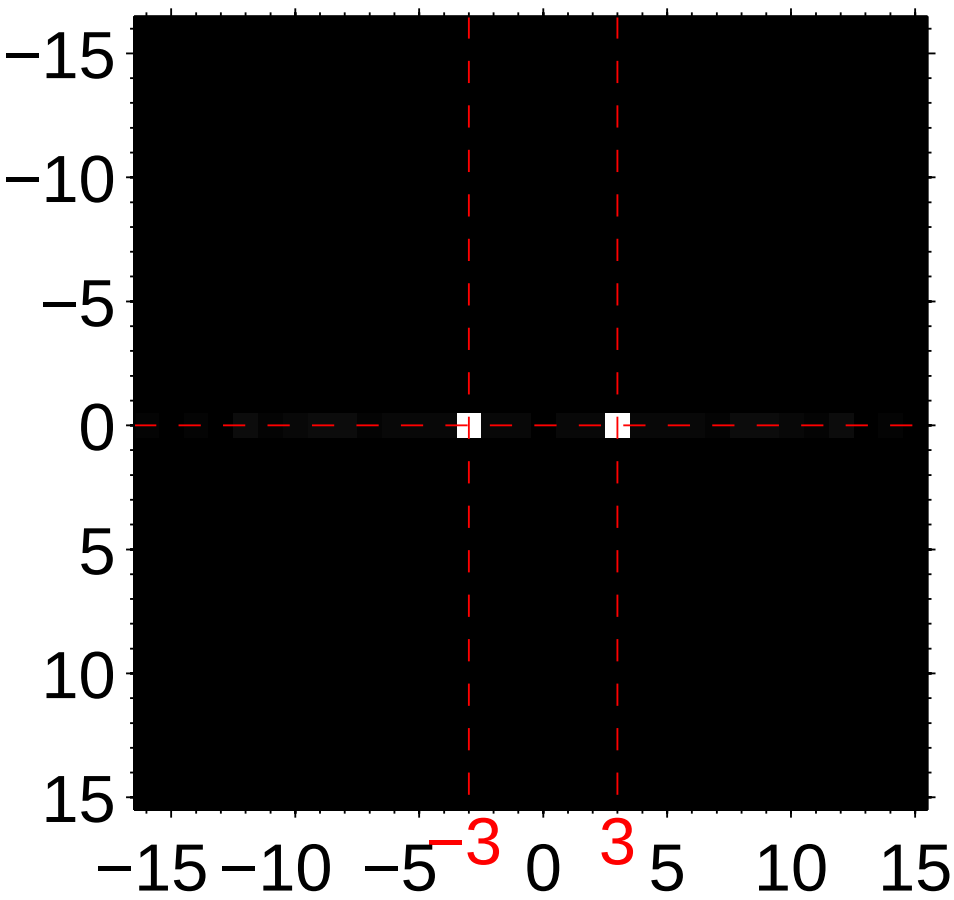
Examples (i): `fftshift`

$$x(n,k) = \sin(2\pi \mathbf{3} k/N)$$

signal

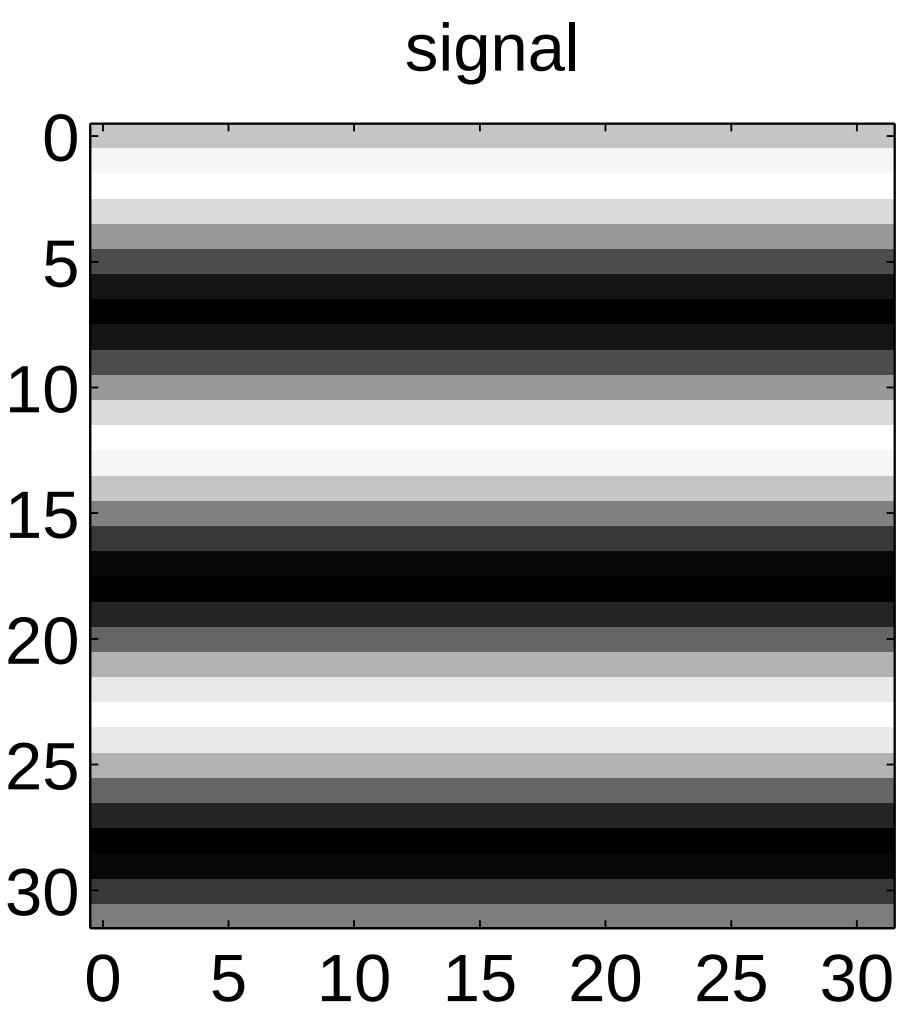


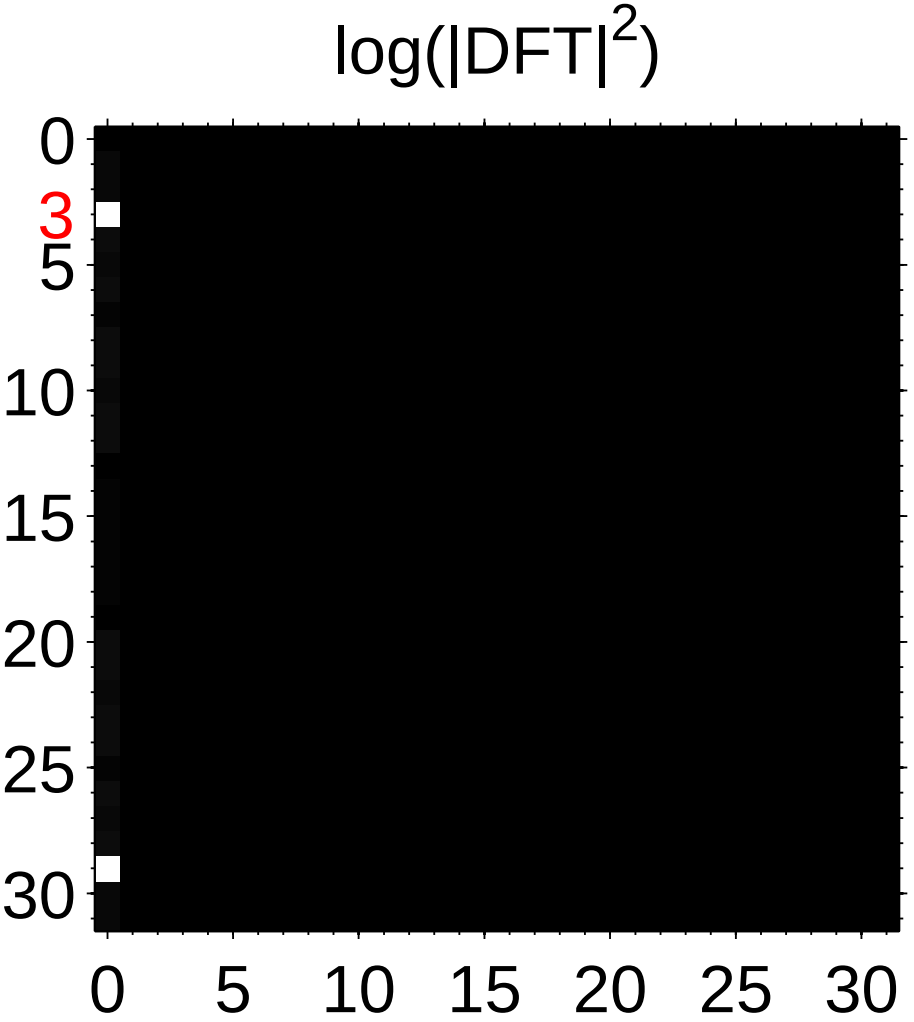
log(|DFT|^2)



Examples (ii)

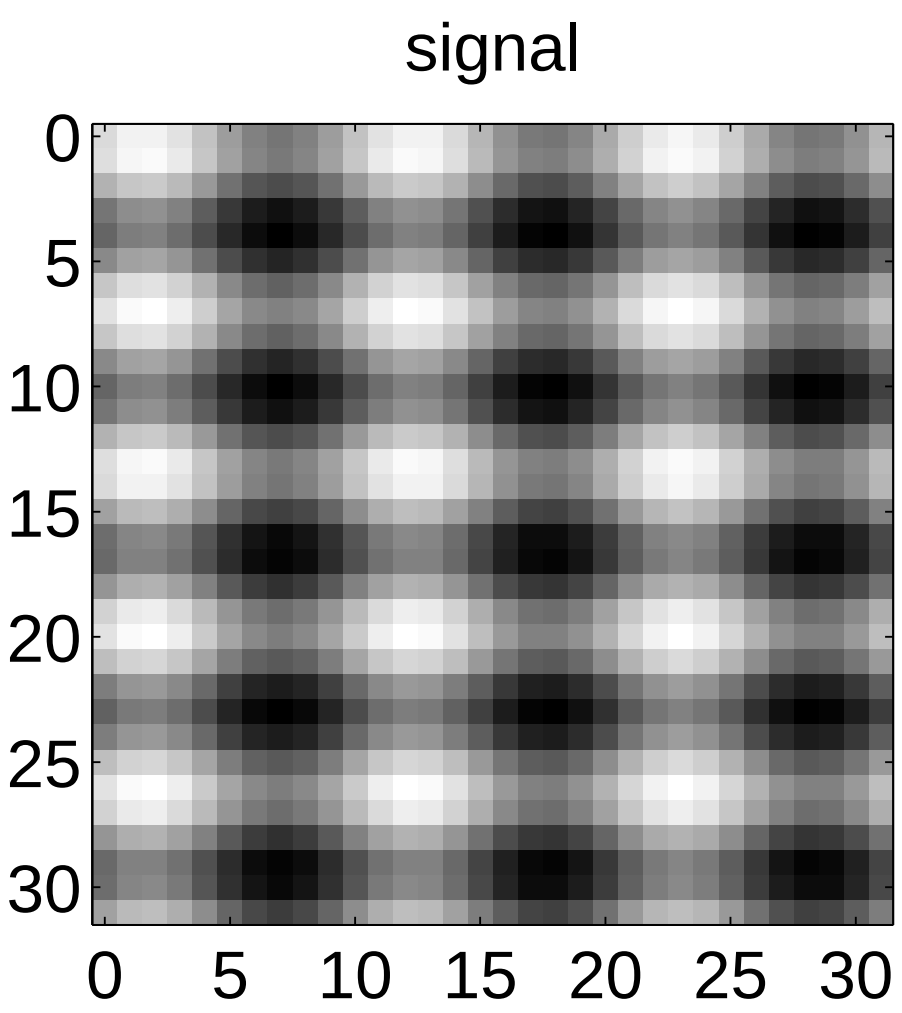
$$x(n,k) = \sin(2\pi \mathbf{3} n/N)$$

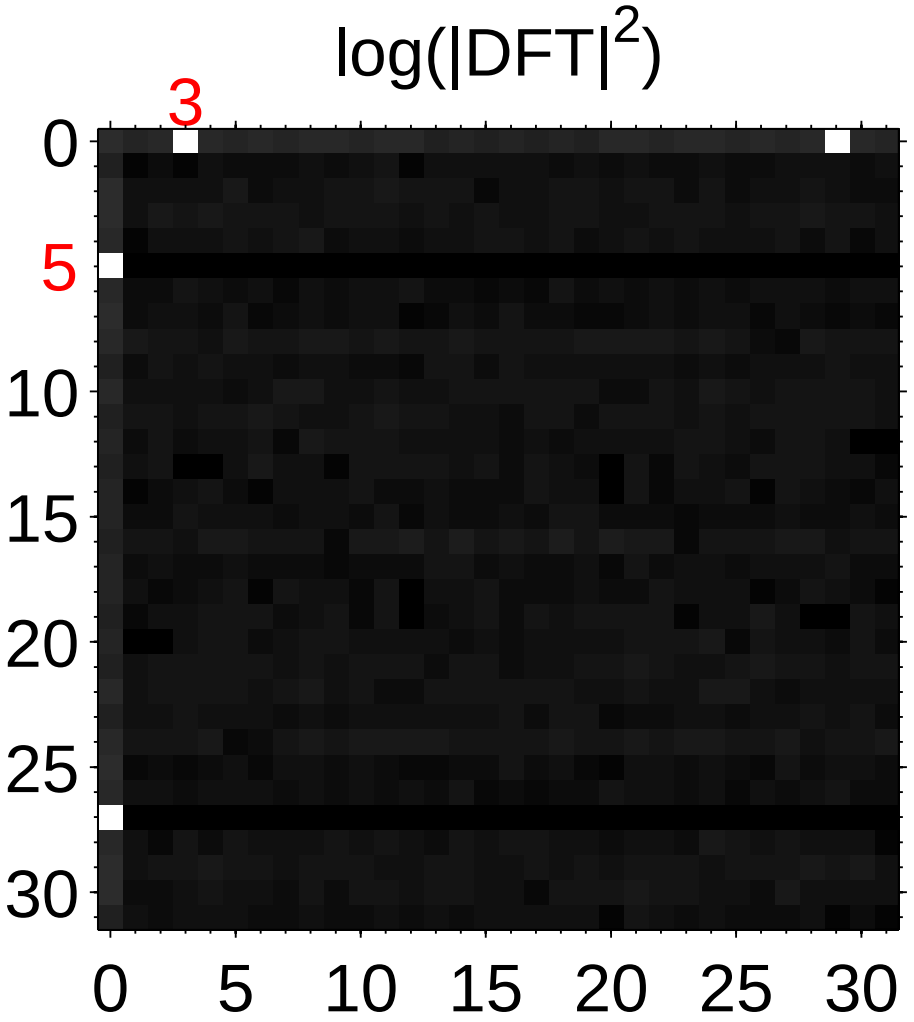




Examples (iii): superposition

$$x(n,k) = \sin(2\pi \mathbf{5} n/N) + \sin(2\pi \mathbf{3} k/N)$$

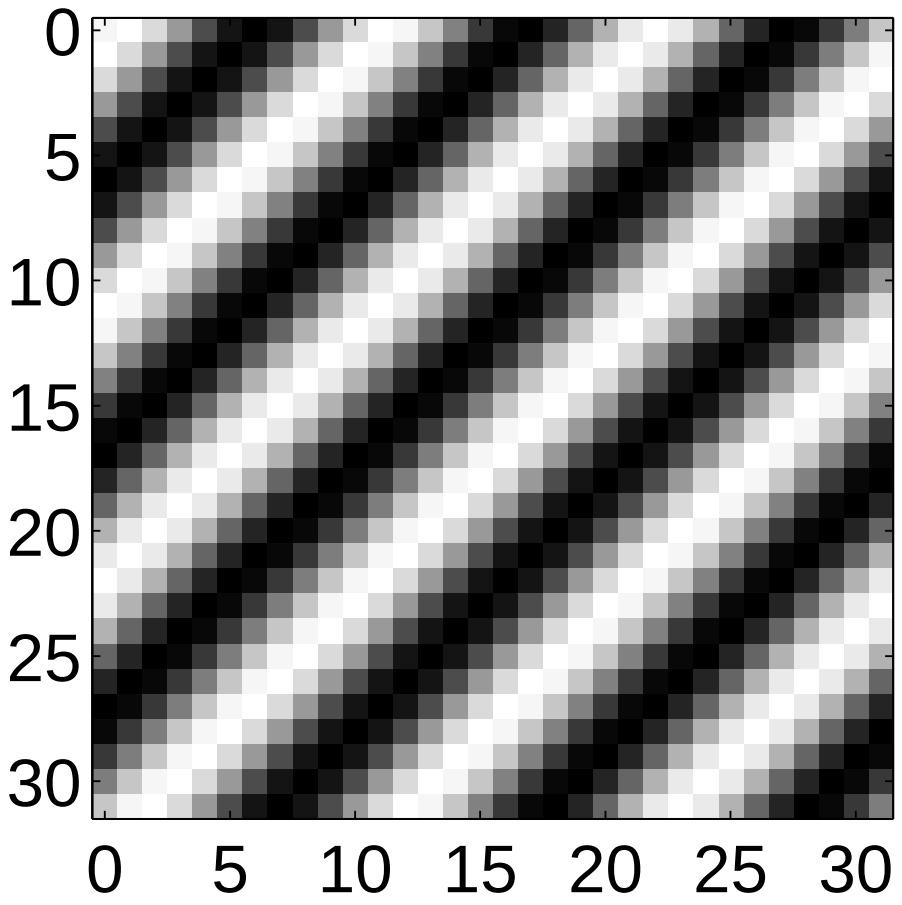




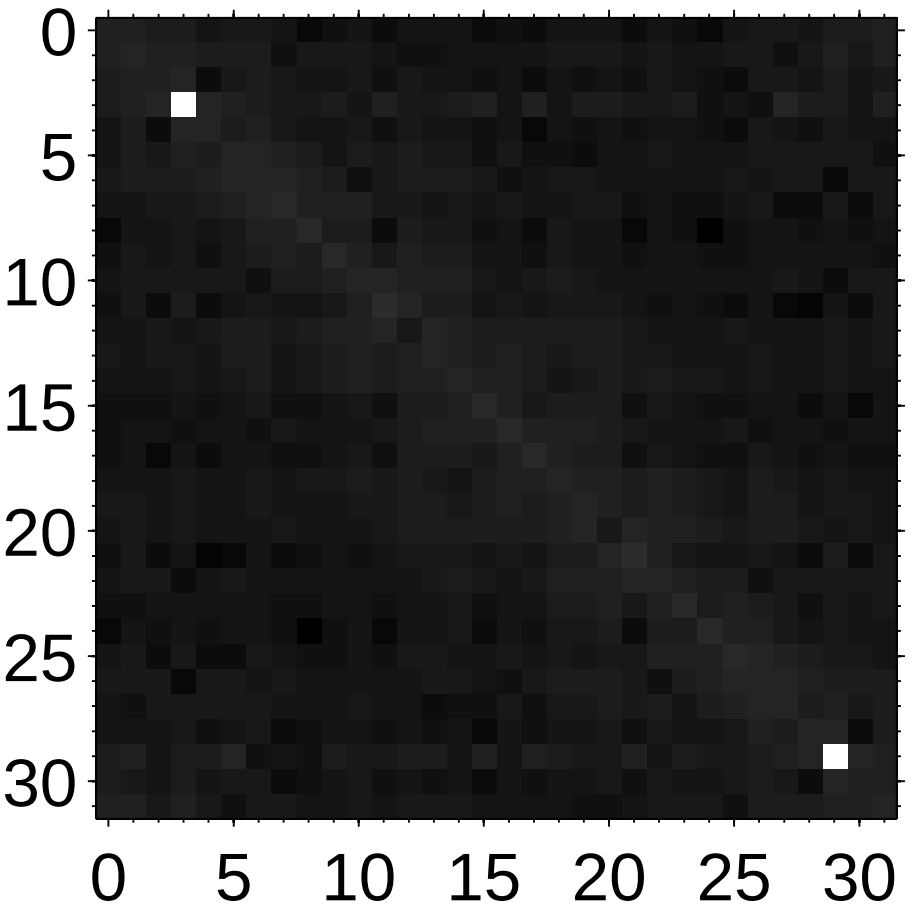
Examples (iv)

$$x(n,k) = \sin\left(2\pi \mathbf{3}(n+k)/N\right)$$

signal



$\log(|\mathrm{DFT}|^2)$



DFT and symmetry

The symmetry of the 2D FT depends on the symmetry of the function.

$$\begin{aligned} F(-s,-v) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{i2\pi(sx+ty)}\,dx\,dy \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(-x,-y)e^{-i2\pi(sx+ty)}\,dx\,dy \\ &= \mathcal{F}\{f(-x,-y)\} \end{aligned}$$

As before (in 1D), but now we reflect through the origin.

- similar result to before relating complex conjugates etc.

DFT and symmetry

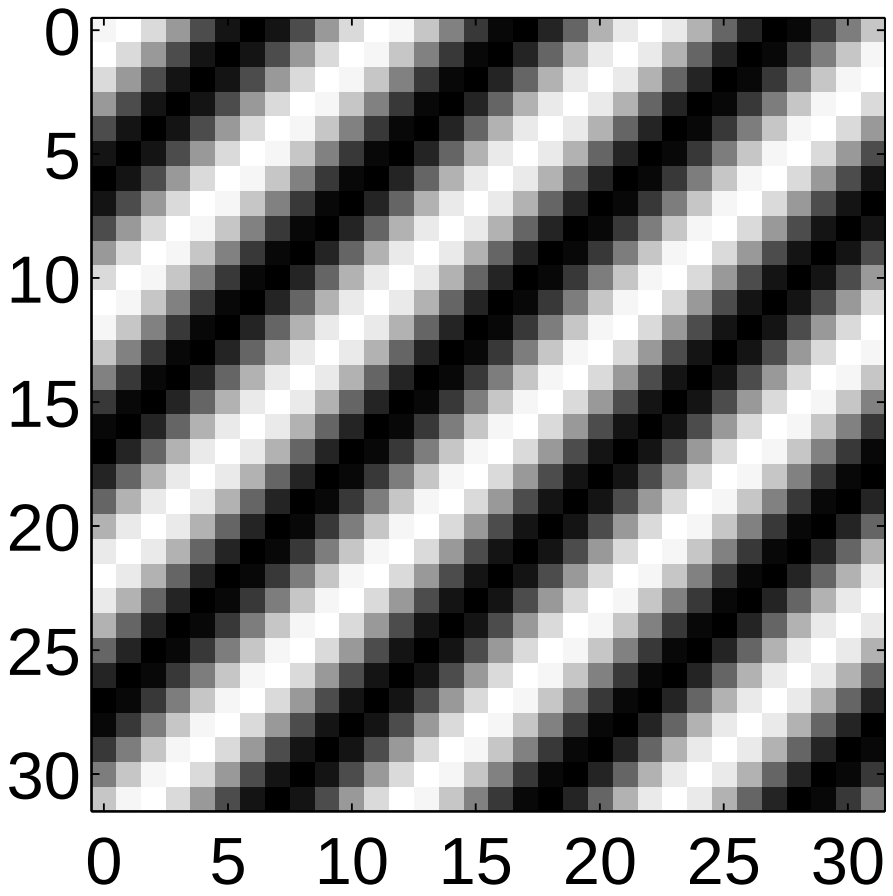
Power-spectrum of 2D DFT will be symmetric about the center (zero frequency).

- Equivalent to real time series produces even power-spectrum.
- In matlab, use `fftshift` to see the plots this way.

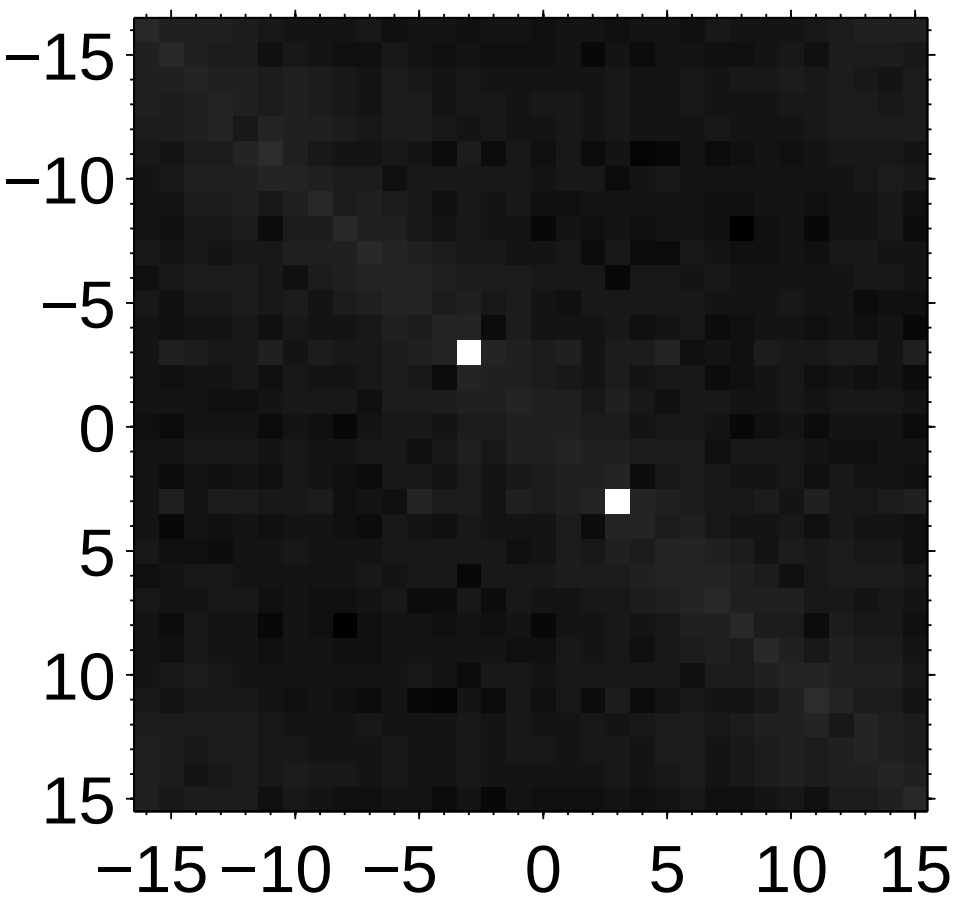
Examples (iv-b)

as before using `fftshift`

signal



$\log(|\mathrm{DFT}|^2)$



2D DFT of a sinusoid

$$f(x,y) = \cos\left[2\pi s_0(\cos(\theta)x + \sin(\theta)y)\right]$$

Multiply by $g_\tau(u)g_\tau(v) =$ Gaussians with standard dev. τ, where u an v are given in the coord. transform below.

$$F(s,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g_\tau(u)g_\tau(v)\, f(x,y)e^{-i2\pi(sx+ty)}\,dx\,dy$$

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g_\tau(u)g_\tau(v)\cos\left[2\pi s_0(\cos(\theta)x + \sin(\theta)y)\right]e^{-i2\pi(sx+ty)}\,dx\,d$$

Change co-ordinates, so that $u = \cos(\theta)x + \sin(\theta)y$ and $v = -\sin(\theta)x + \cos(\theta)y$, and the transformation is just a rotation so the Jacobian is $J = \begin{vmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{vmatrix} = 1.$

2D DFT of a sinusoid

Note $g(r)$ remains unchanged by a rotation in the co-ordinates.

$$F(s,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g_\tau(u)g_\tau(v)\cos\left[2\pi s_0 u\right] e^{-i2\pi u(\cos(\theta)s+\sin(\theta)t)} e^{-i2\pi v(-\sin(\theta)s+\cos(\theta)t)}\, du\, dv$$

$$= \int_{-\infty}^{\infty} g_\tau(v) e^{-i2\pi v(-\sin(\theta)s+\cos(\theta)t)}\, dv \int_{-\infty}^{\infty} g_\tau(u)\cos\left[2\pi s_0 u\right] e^{-i2\pi u(\cos(\theta)s+\sin(\theta)t)}\, du$$

The second integral is just the Fourier transform of a cosine (with a Gaussian), so as the Gaussian width increases $\tau \longrightarrow \infty$, it becomes the FT of a cosine.

$$\frac{1}{2}\left[\delta(\cos(\theta)s+\sin(\theta)t-s_0)+\delta(\cos(\theta)s+\sin(\theta)t+s_0)\right]$$

A pair of parallel "ridges" in the (s,t) plane.

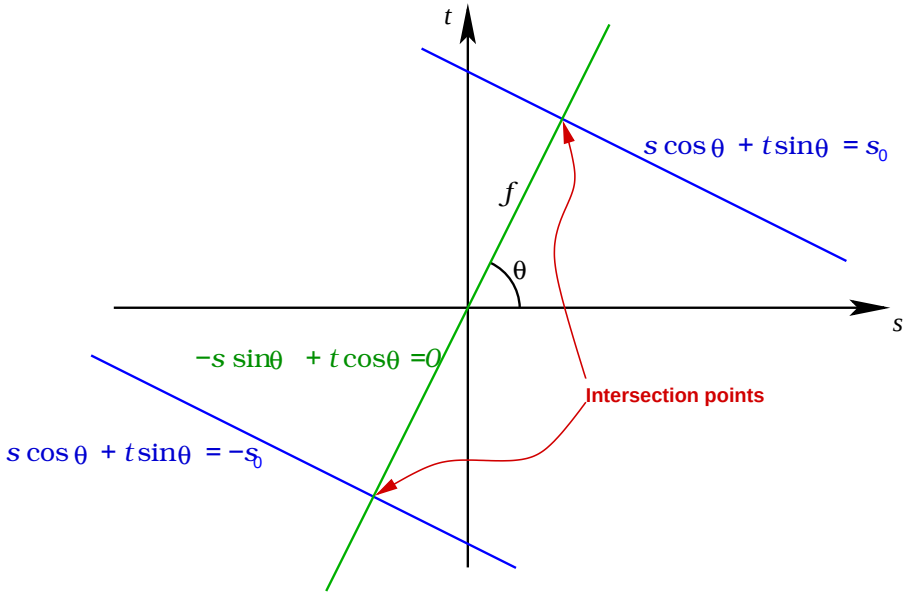
2D DFT of a sinusoid

Result

$$F(s,t) = \int_{-\infty}^{\infty} e^{-i2\pi v(-\sin(\theta)s+\cos(\theta)t)}\,dv \int_{-\infty}^{\infty} \cos\left[2\pi s_0 u\right] e^{-i2\pi u(\cos(\theta)s+\sin(\theta)t)}\,du$$

$$= \frac{1}{2}\delta(-\sin(\theta)s+\cos(\theta)t)\left[\delta(\cos(\theta)s+\sin(\theta)t-s_0)+\delta(\cos(\theta)s+\sin(\theta)t+s_0)\right]$$

A pair of parallel ridges, at right angles to another ridge, intersecting at two points.



2D DFT of a sinusoid

$f(x,y) = \cos[2\pi s_0(\cos(\theta)x + \sin(\theta)y)]$ represents a sinusoid rotated by θ around the center of the plane.

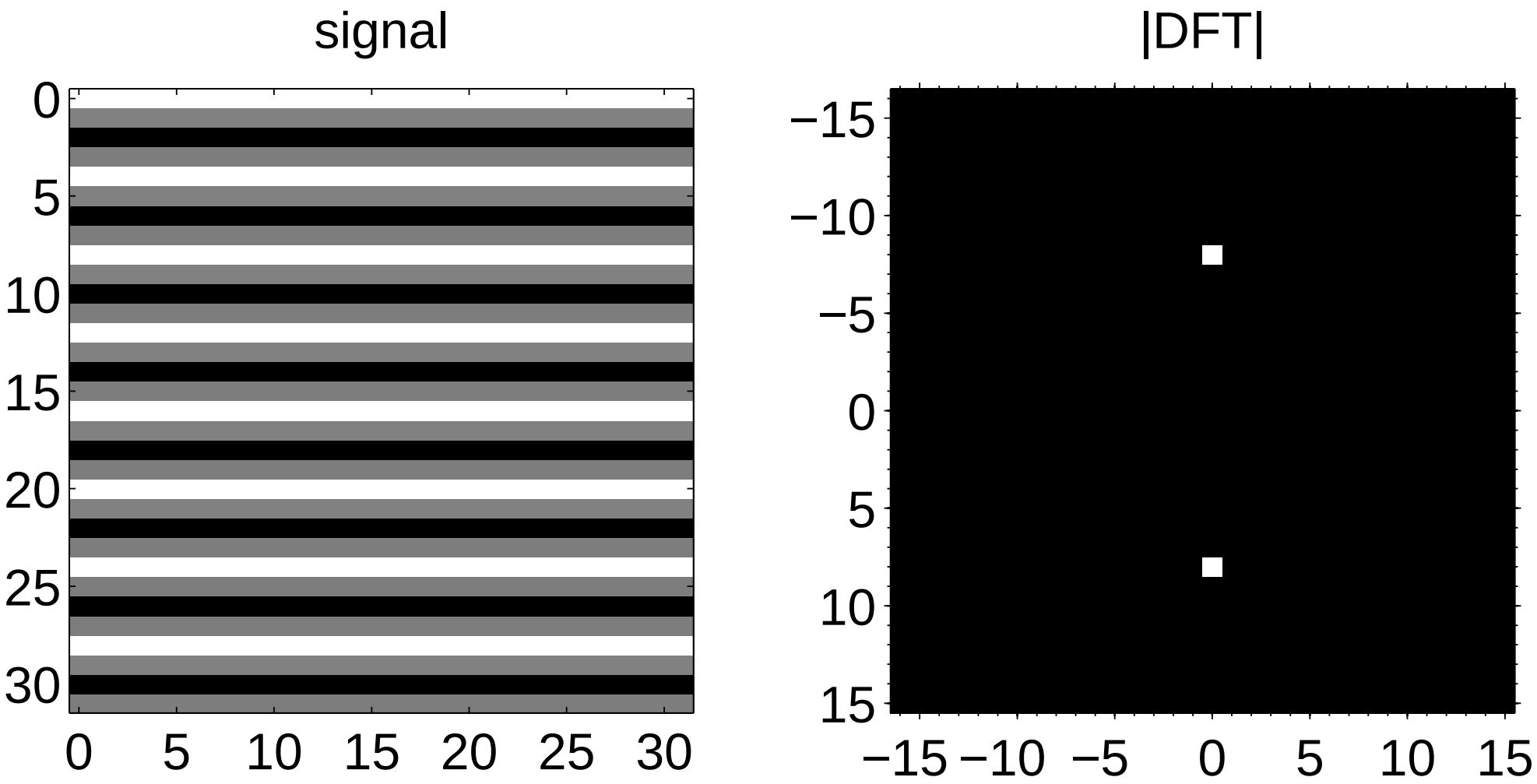
We have seen that the result is that the FT is also rotated by θ, about the center.

This holds more generally – a spatial rotation causes a rotation in the frequency domain.

However, its not quite that simple for discrete transforms, where there is no such thing as an exact rotation (except by $k\pi/2$).

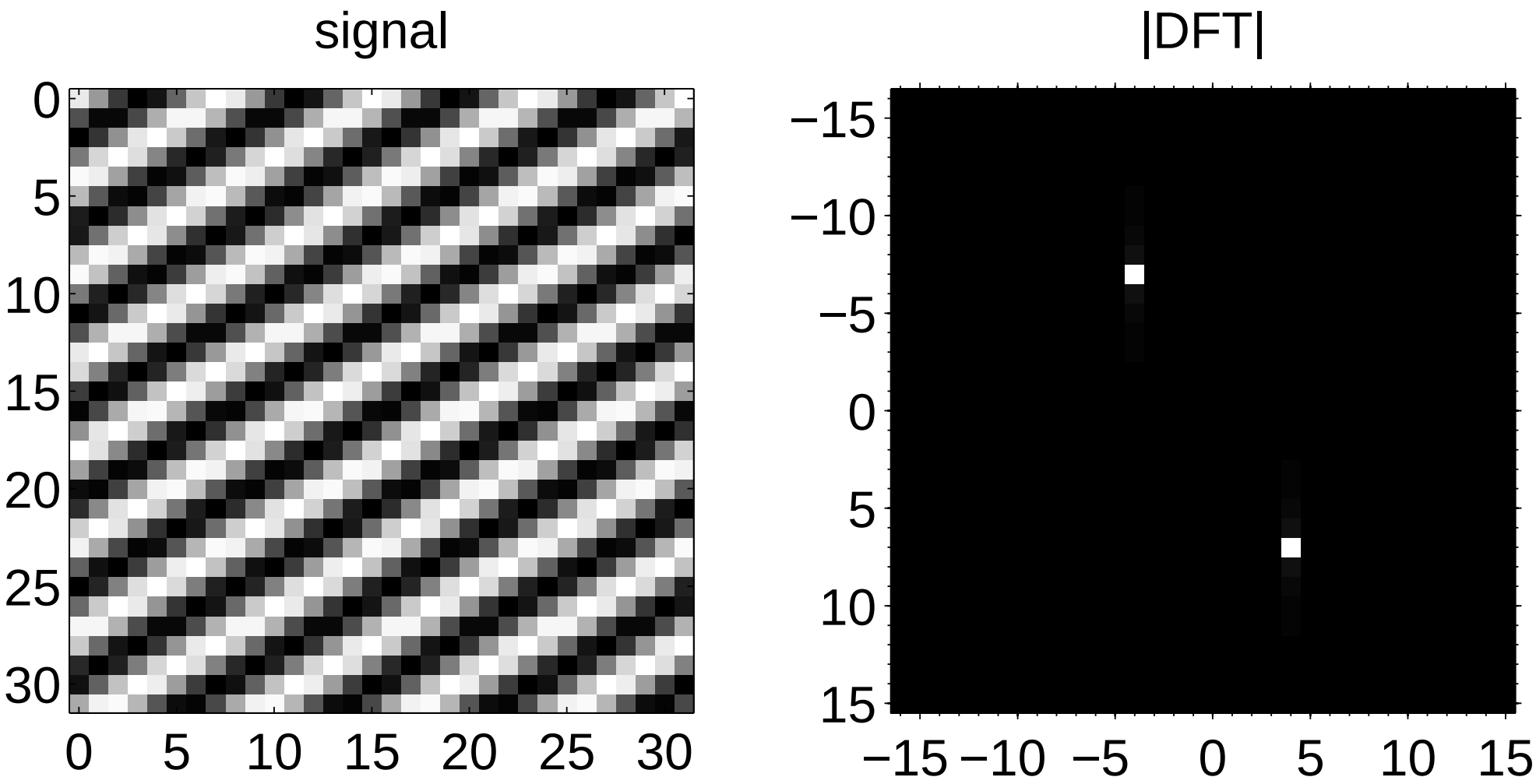
Examples (v)

$x(n,k) = \sin(2\pi 8n/N)$ with `fftshift`



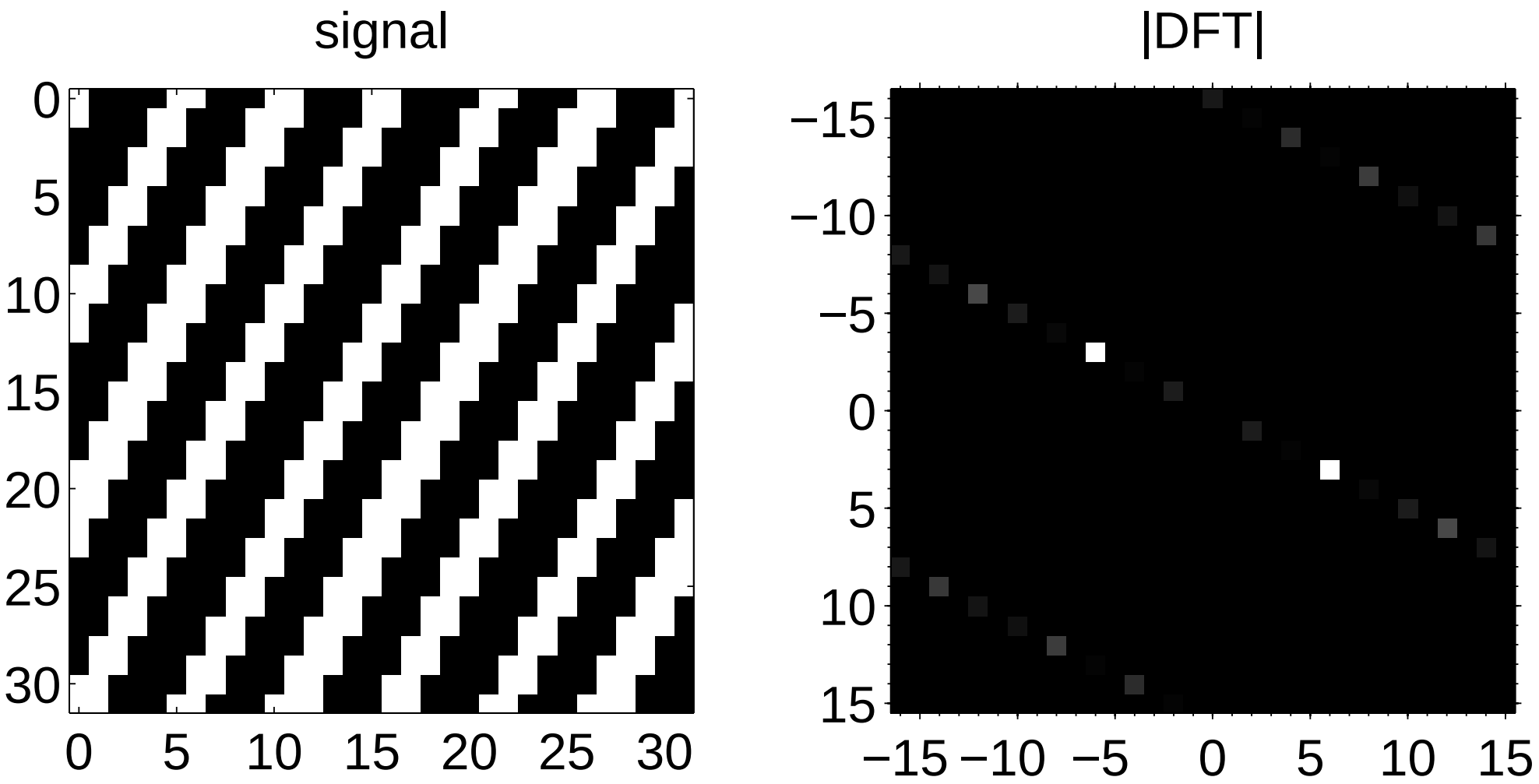
Examples (v-b)

$x(n,k) = \sin\left(2\pi 8(\cos(\theta)n + \sin(\theta)k)/N\right)$ with `fftshift`



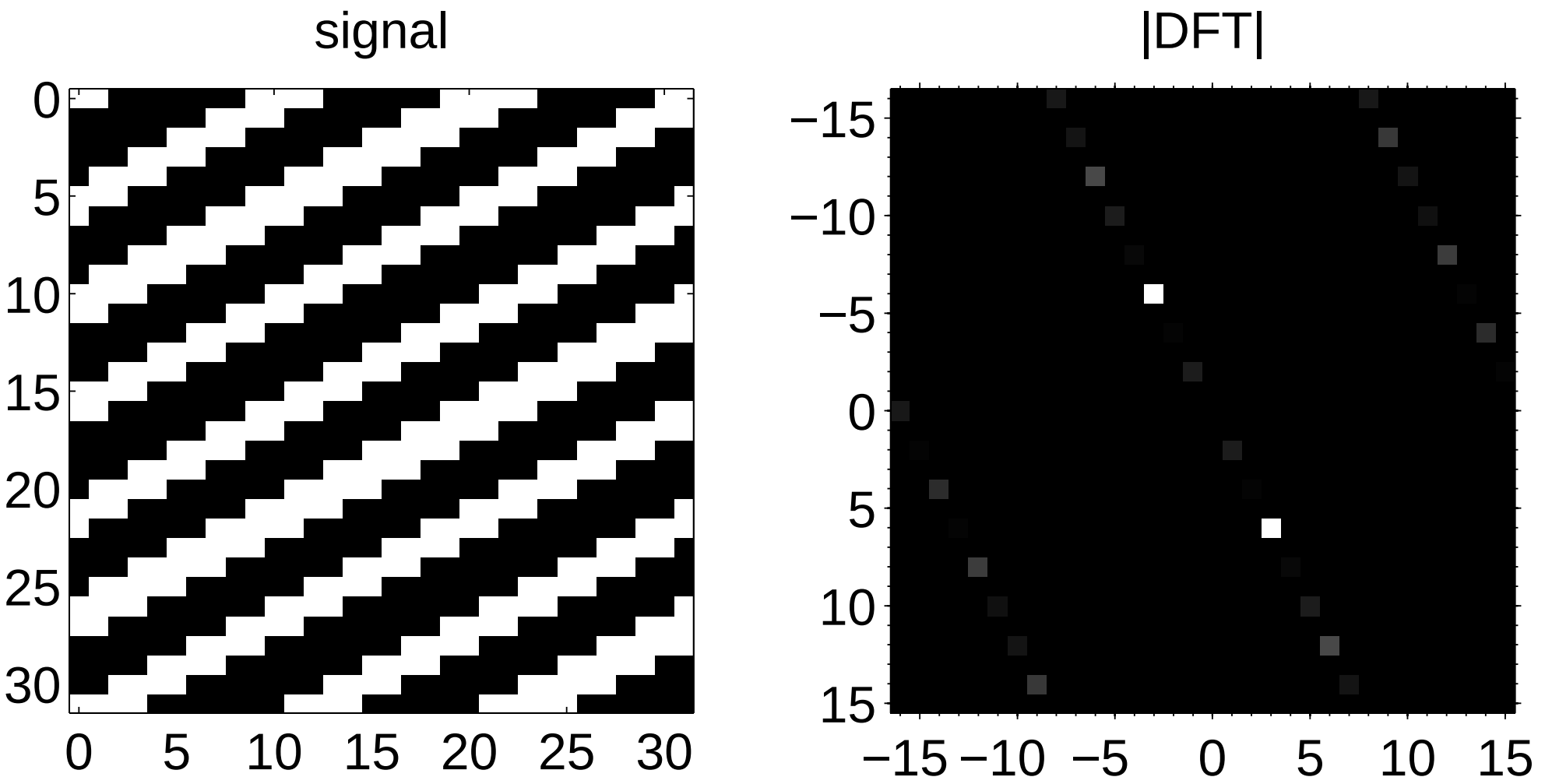
Examples (vi)

$x(n,k) = I\{\sin(2\pi(n+2k)/N) > 0.2\}$ with `fftshift`



Examples (vii)

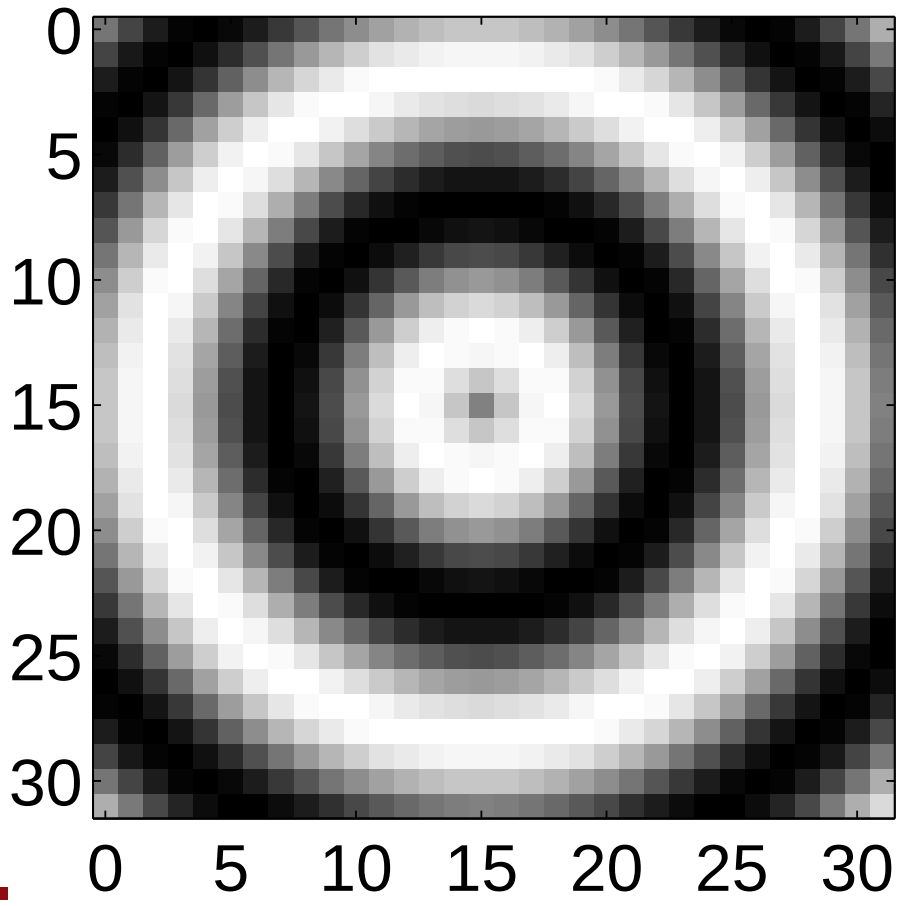
$x(n,k) = I\{\sin(2\pi(2n+k)/N) > 0.2\}$ with `fftshift`



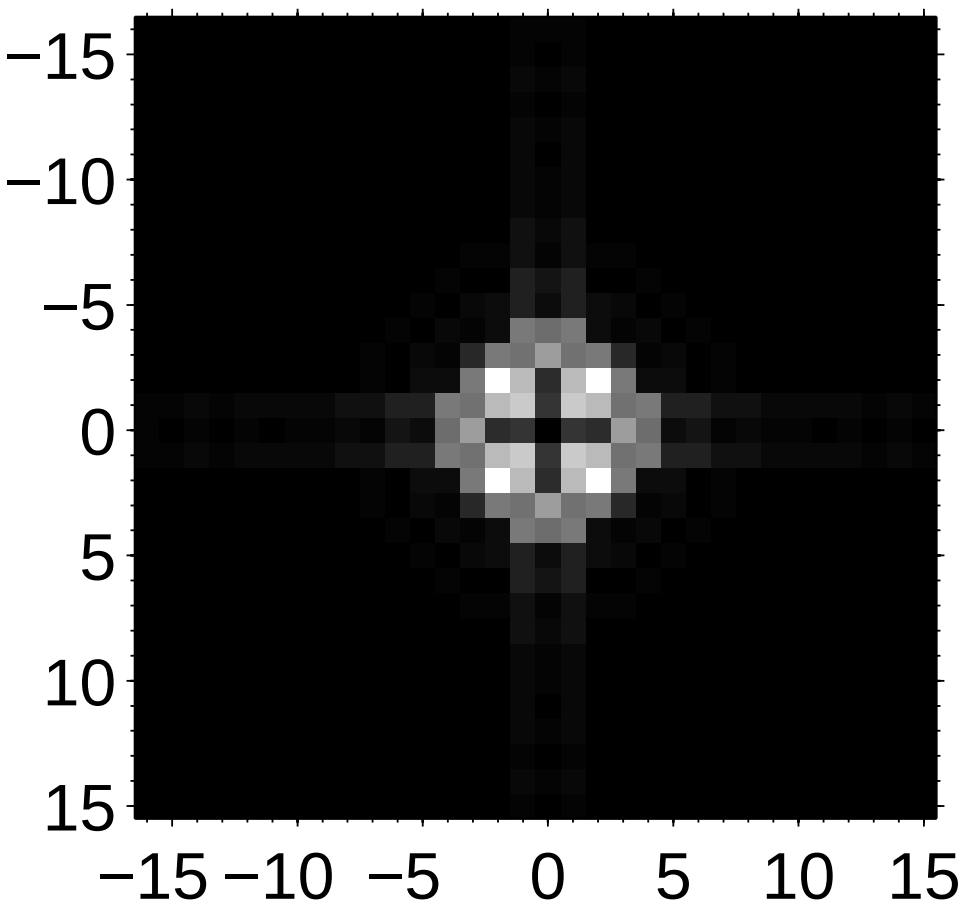
Examples (viii)

$$x(n,k) = \sin\left(2\pi\sqrt{(n/N-1/2)^2+(k/N-1/2)^2}\right)$$ with `fftshift`

signal

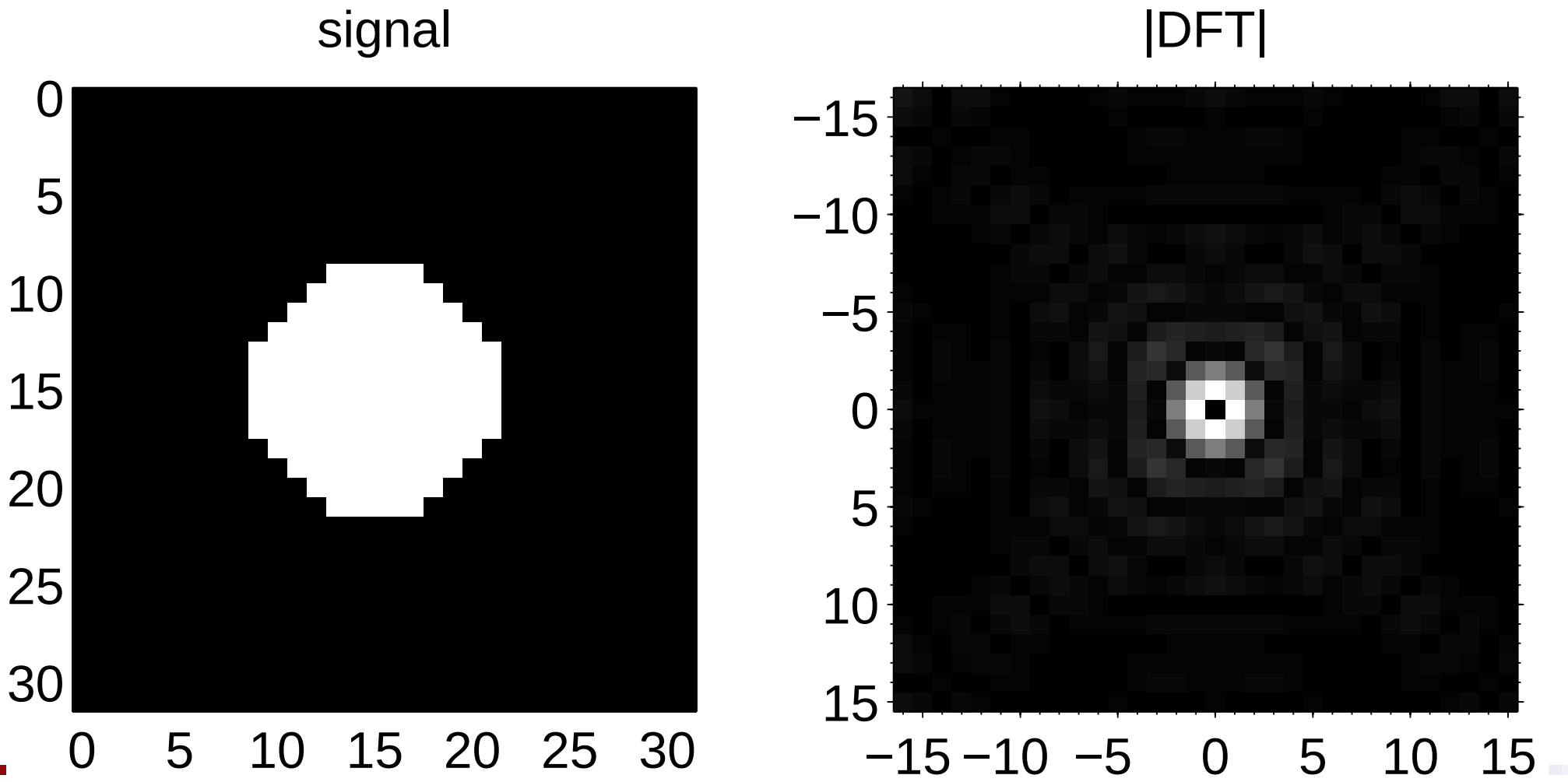


|DFT|



Examples (viiib)

$x(n,k) = I\left\{\sqrt{(n/N-1/2)^2+(k/N-1/2)^2} < 0.2\right\}$ with `fftshift`



Radial symmetry

Radially symmetric signal produces radially symmetric DFT

- we know that a rotation in space domain, causes equivalent rotation in frequency domain.
- rotation doesn't change $f(x,y)$, so $F(s,t)$ must also be invariant.
- Remember discretization effects limit radial symmetry.

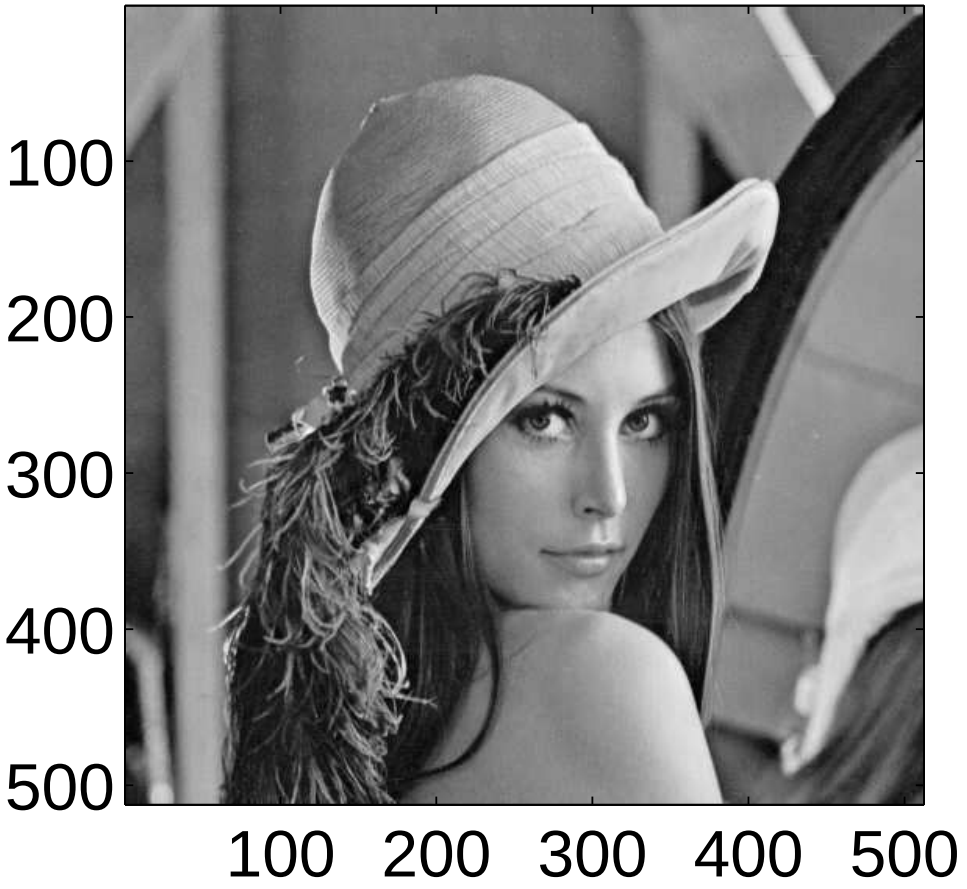
Given radial symmetry can get Hankel transform:

- useful where the system has radial symmetry
- e.g. optical systems, such as lenses.

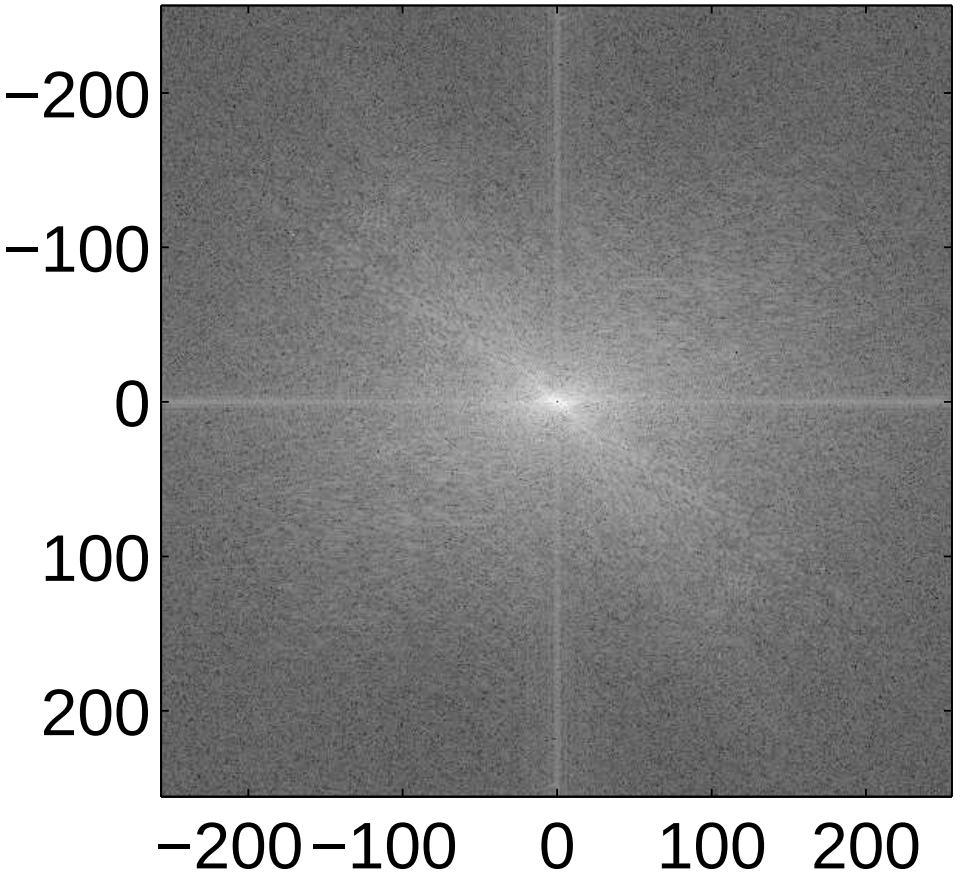
Examples (Lena)

Lena image and power-spectra plotted using `fftshift`

signal



$\log(|DFT|^2)$



Aliasing in Images

Just as in 1D signals, when we quantize we introduce noise, and when we sample, we can introduce aliasing. However, aliasing in images (and other higher dimensional signals) can take many forms, and you have probably seen the effect before.

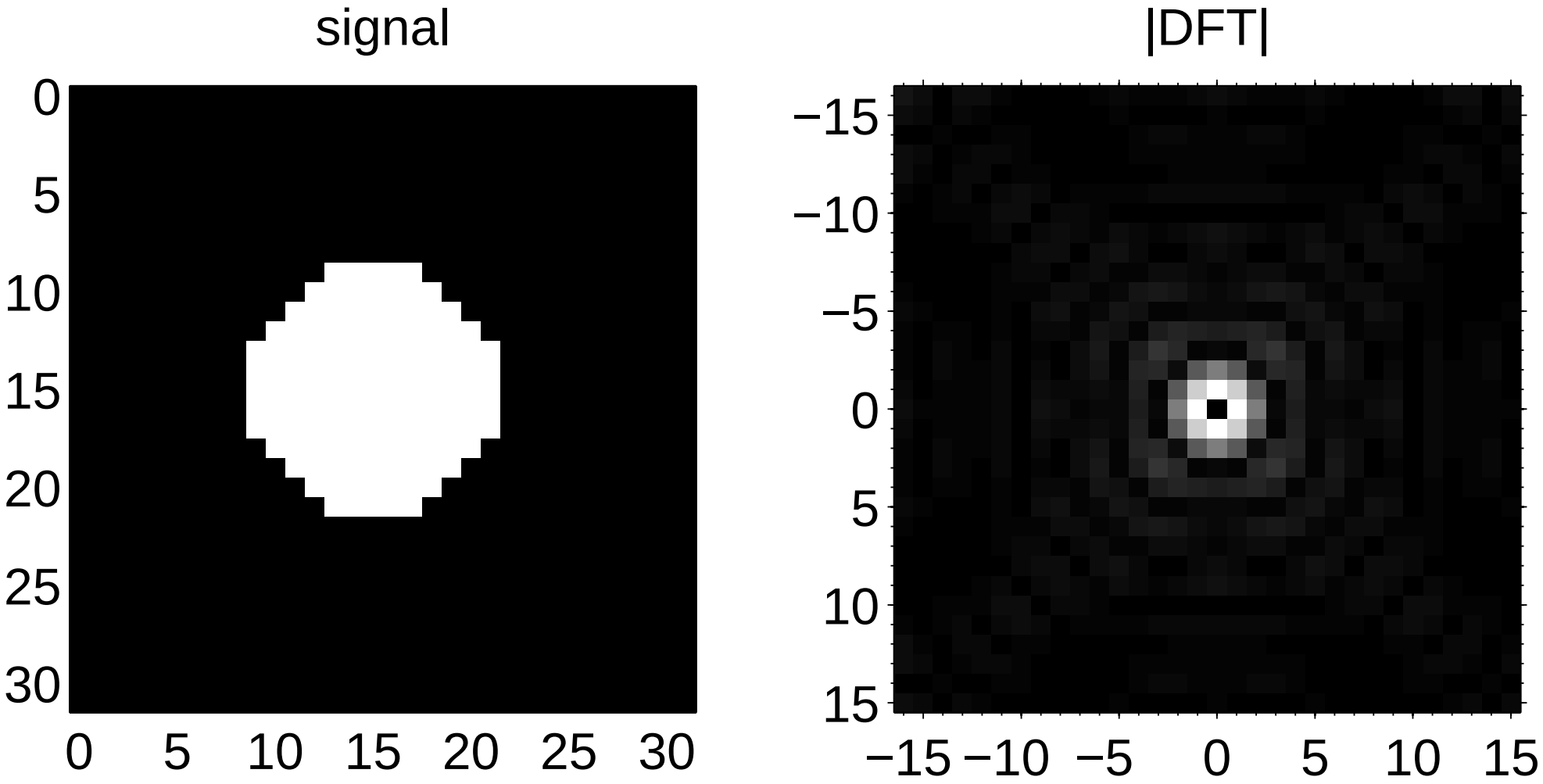
Aliasing in images

Aliasing in images is similar to that in time signals.

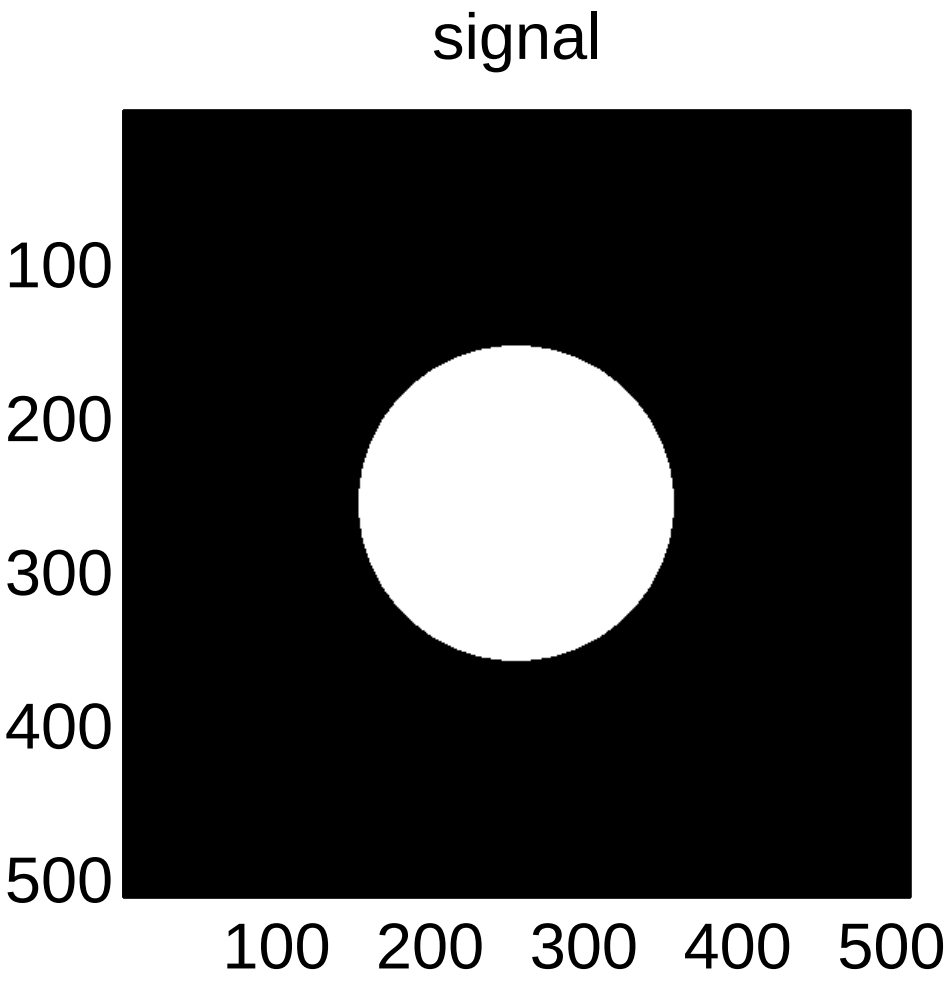
- an image is a sampled spatial field
- cameras average over a small angle for each pixel, so effectively low-pass the image field before sampling.
- CGI generation by sampling of underlying mathematical model.
- produces jagged edges in images "jaggies"
- Moire patterns

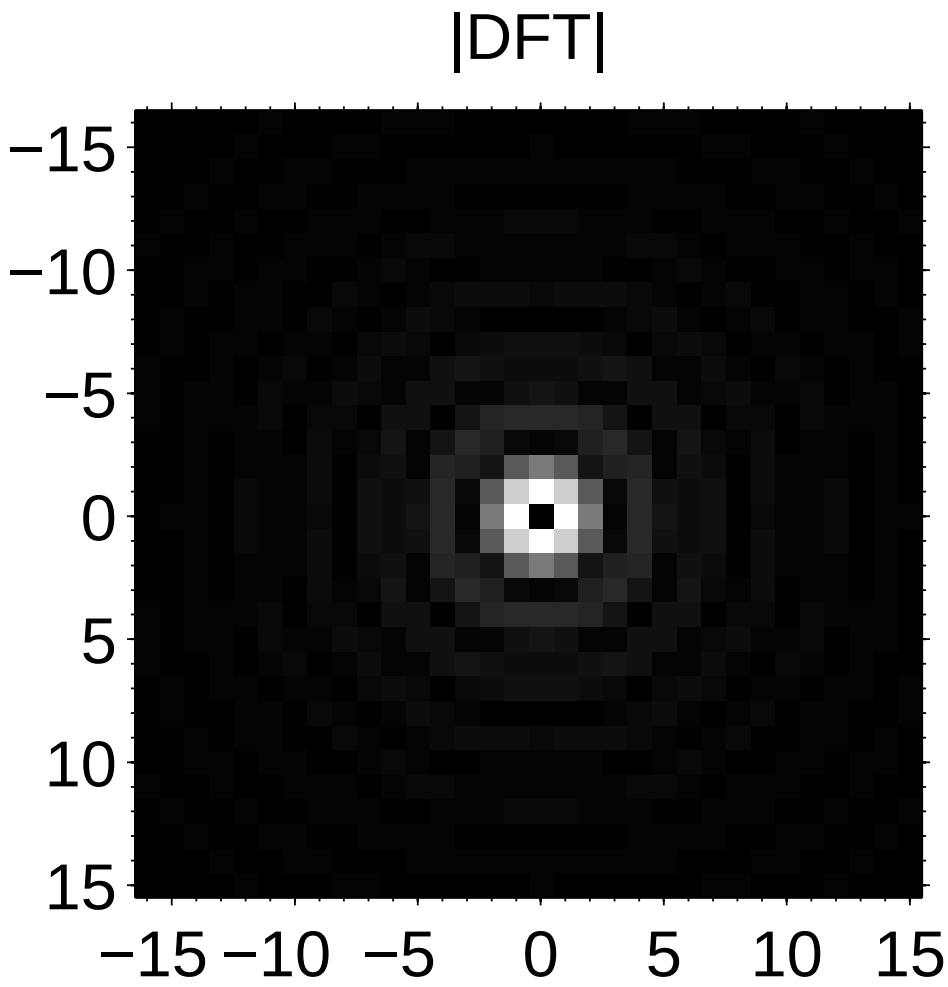
Solution is low-pass prefiltering of data (as before).

Jaggies

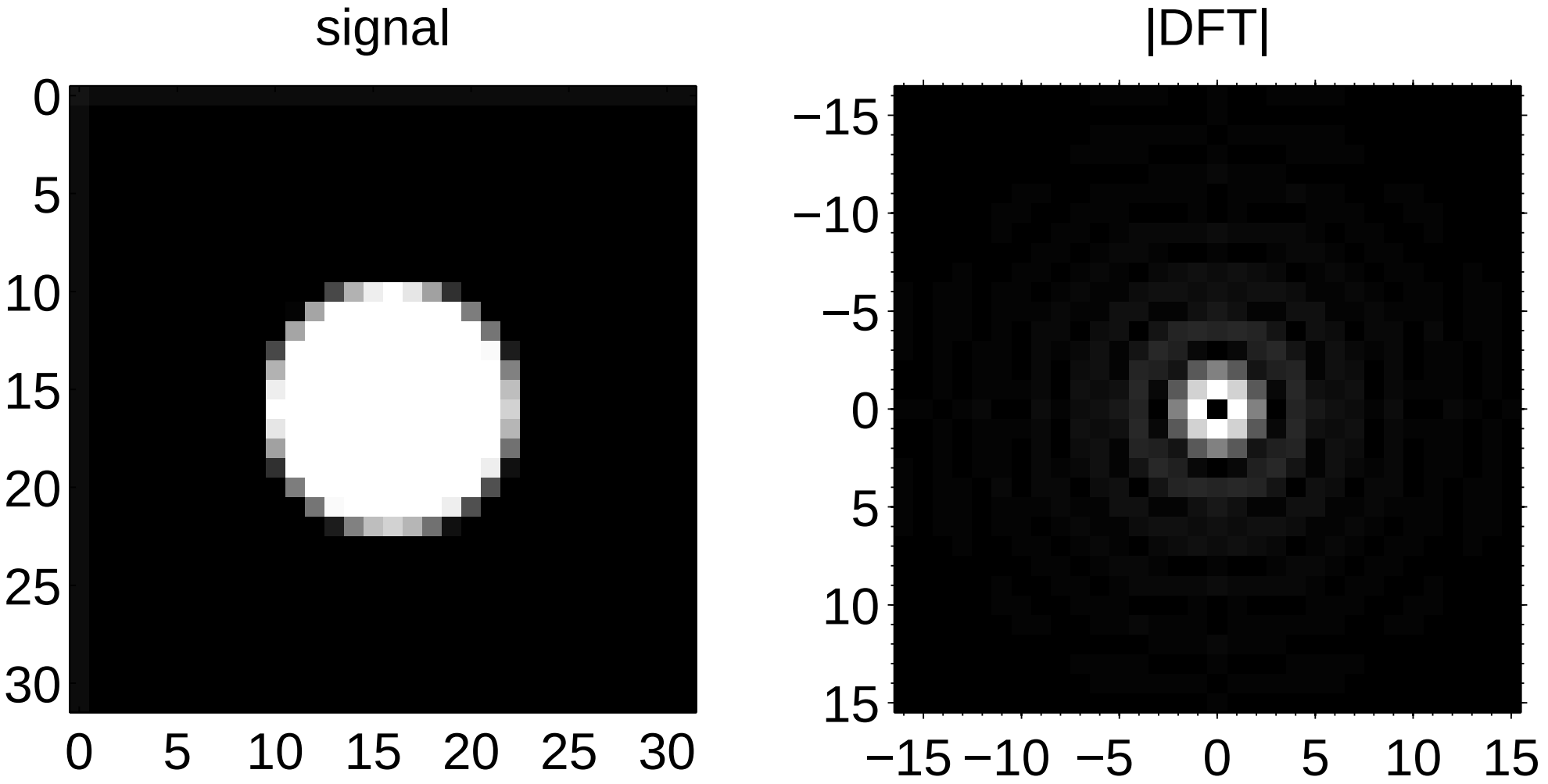


Jaggies (reduced by enhanced resolut

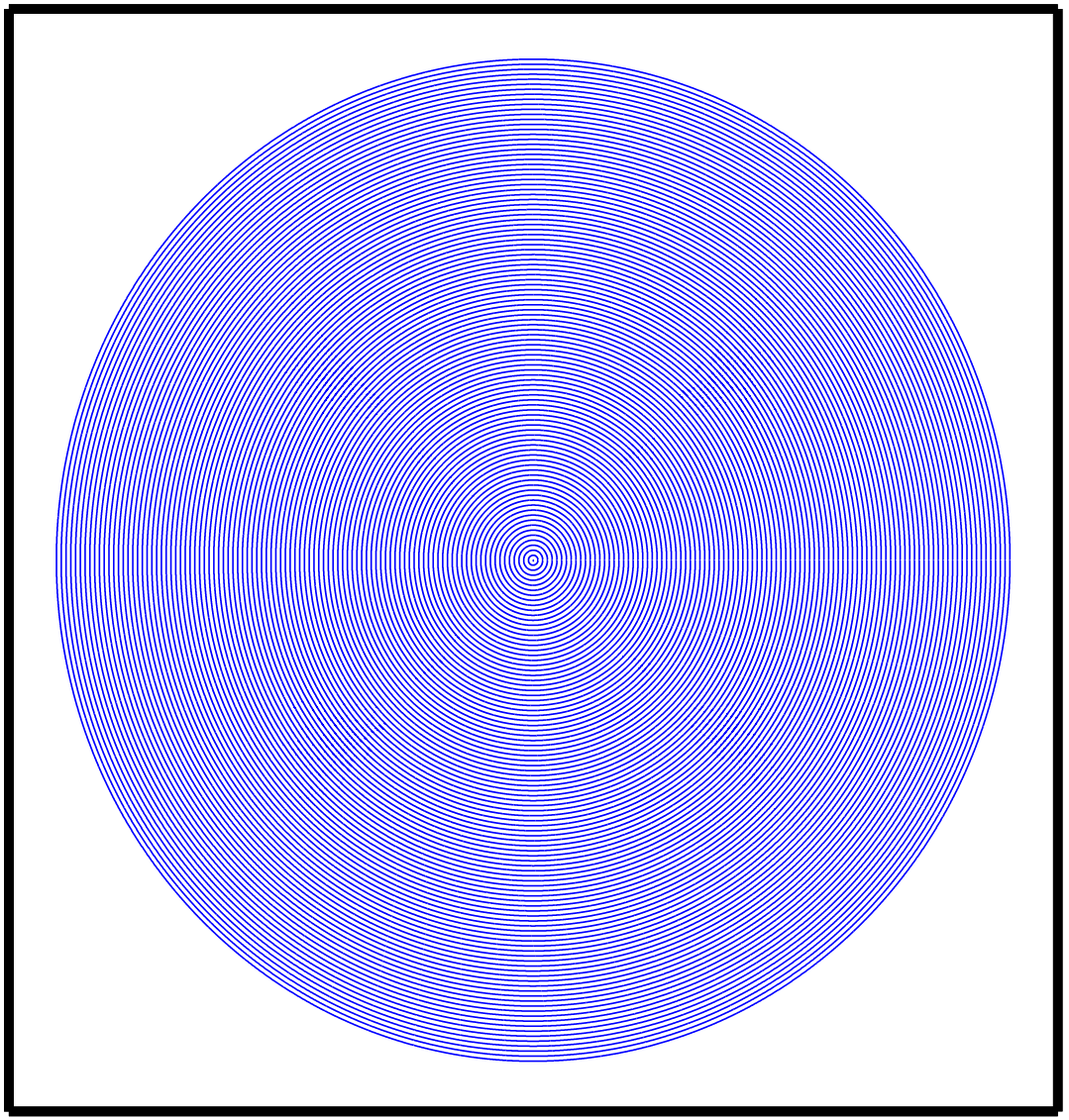




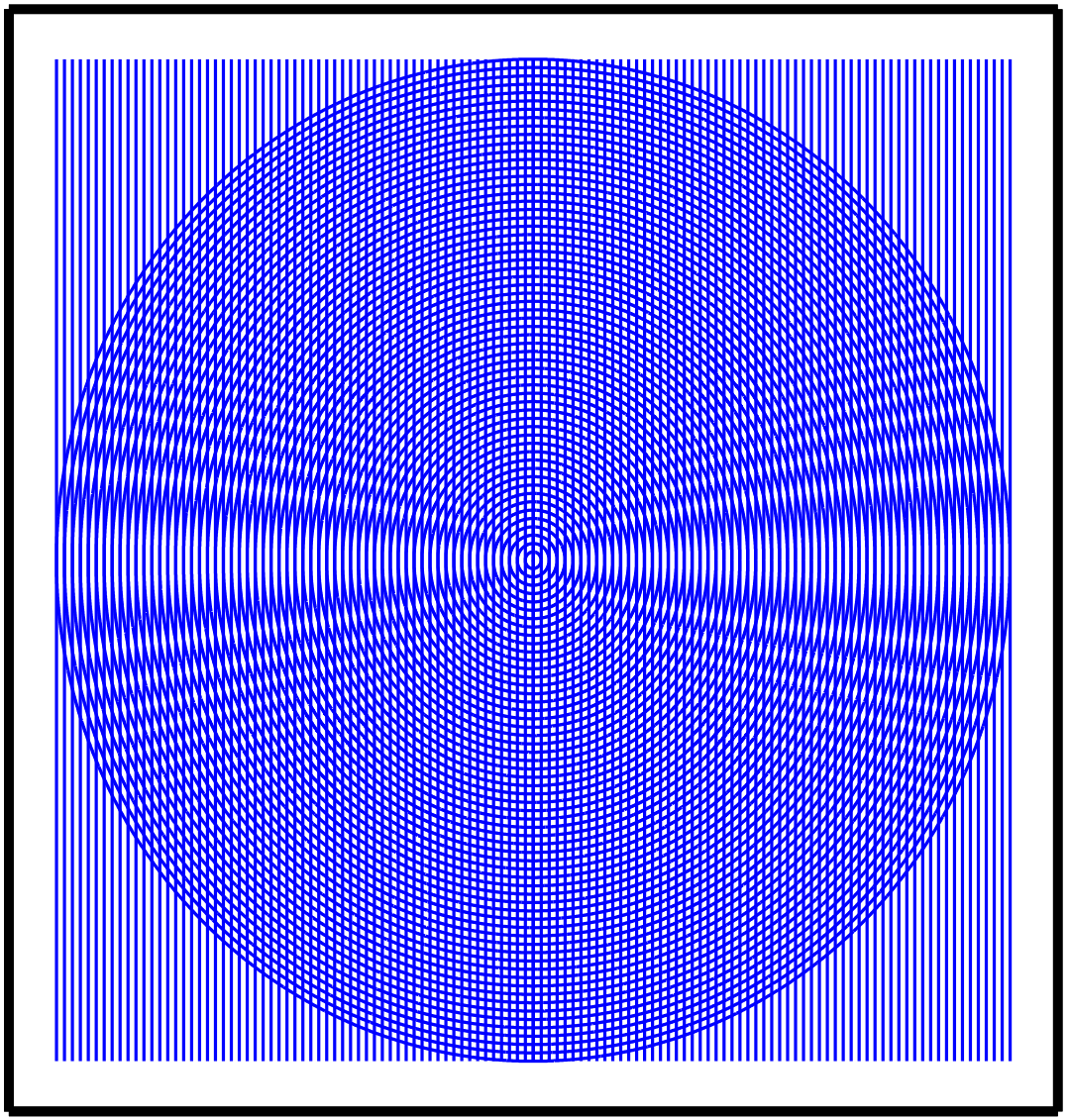
Jaggies (reduced by pre-filtering)



Moire patterns

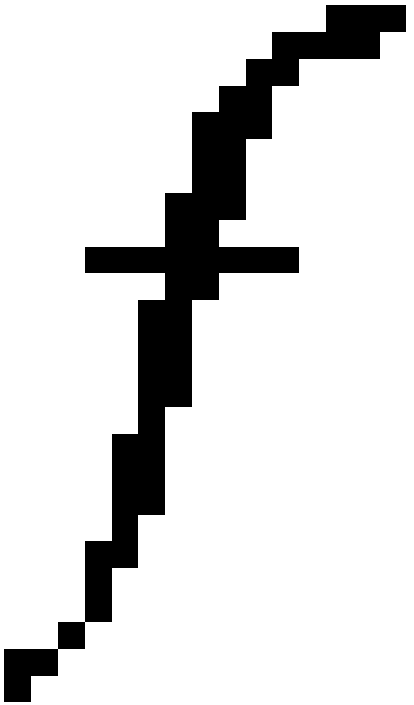


Moire patterns

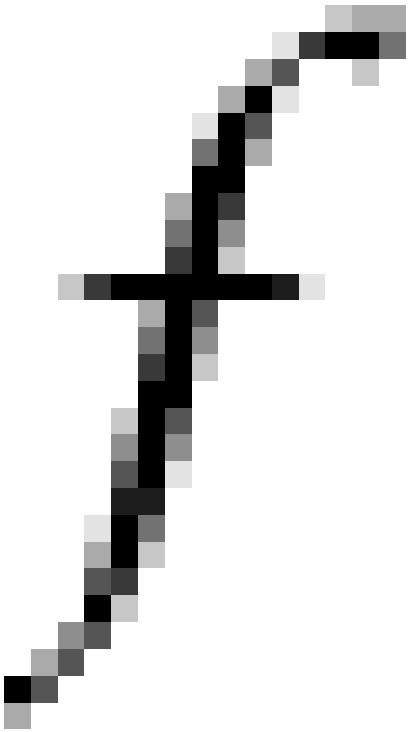


Anti-aliased fonts

Aliased



Anti-aliased

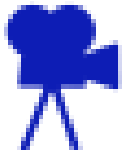


Anti-aliased CGI

Computer-Generated Images (CGI)

- one way to generate is **raytracing**
- generate a ray for each image pixel, and trace its path, including reflections, and refraction.
- computationally expensive (there are faster but less exact methods), so don't want to generate more than one ray per pixel
- but jagged edges move (somewhat randomly), generating marching ants.
- to get **good** results need to **oversample**, and average (e.g. a low-pass operation)

Aliasing: crawling ants



Resampling applications: video

There are many variations of display for images

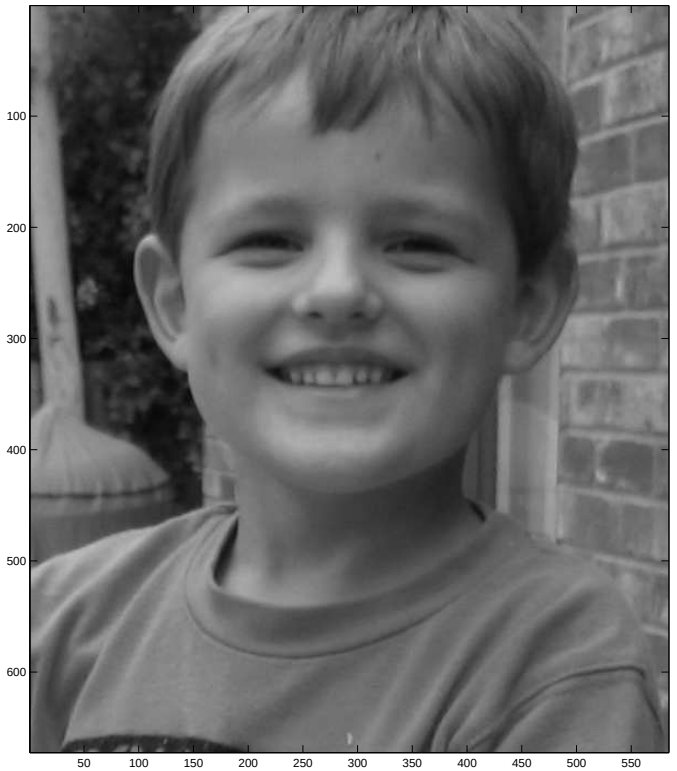
Display	**width $\times$ height**	**Depth**	**rate**
VGA monitor	640×480	4 bit	60 Hz
PAL TV	720×576 (625)	-	50 Hz
HDTV 1080i	1920×1080	8 bit	50 Hz
Plasma TV (typical)	1024×768	14 bit	variable
Film	$\sim 3000 \times \sim 2000$	~ 12 bit	24 Hz
Workstation	1920×1200	24 bit	60 Hz
B&W laser printer	6600×5500	1 bit	-

We need to be able to convert between these (e.g. if your standard DVD player is hooked up to a high-def plasma screen).

Resampling applications: printing

Problem of resampling has been around for a long while

- Printers put ink on a page
  - think of this as 1 bit quantization (ink, or no ink)
  - so how can you do a picture, with only 1 bit?



becomes

Newsprint

Newspapers came up with solutions many years ago

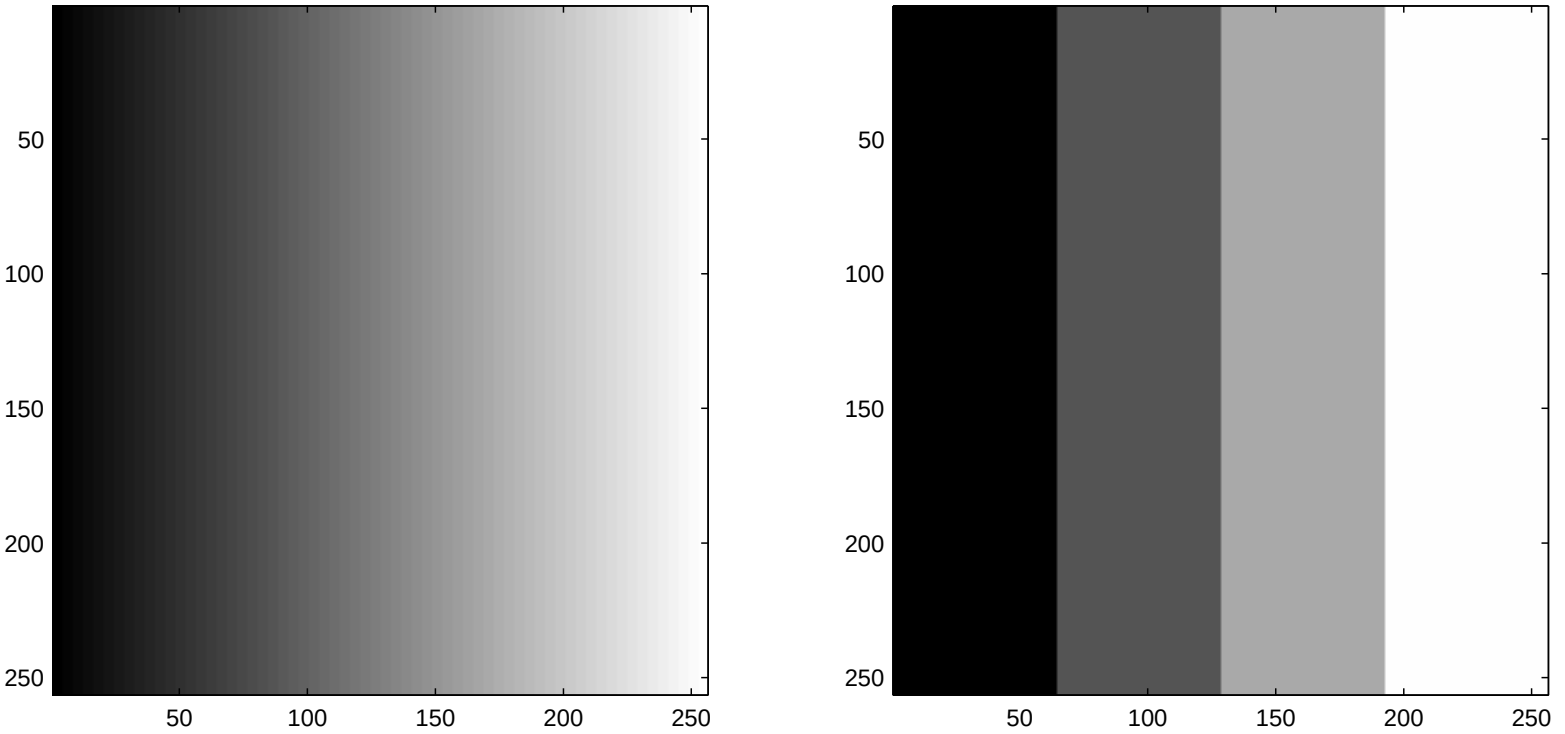
from the Australian, Dec 11th, 2005

Resampling applications: printing

- many printers have only 1 bit quantization
  - e.g. B&W laser printers
- ink jets (typically) have 3 or 4 color inks
  - but you can't mix them, so need to put together somehow?
- printers typically have better spatial resolution than a monitor
  - e.g. 1200 dpi (dots per inch)
  - compare to a monitor with perhaps 72 dpi
- so we tradeoff sampling vs quantization
  - it works because our eyes (or other sensors) effectively do a low pass before sampling

Half-toning

- problem is that we have limited quantization

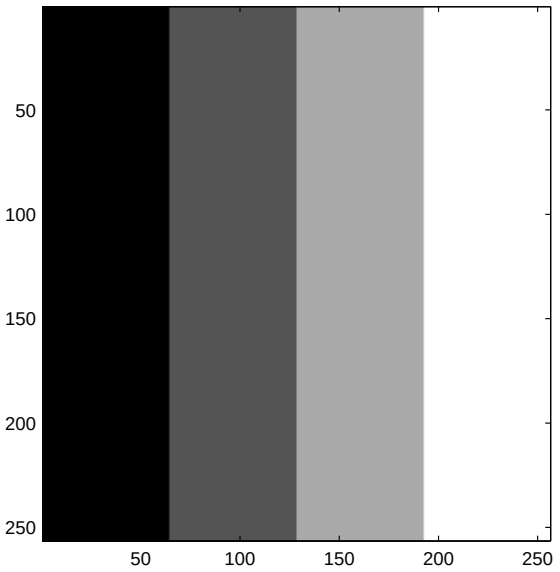


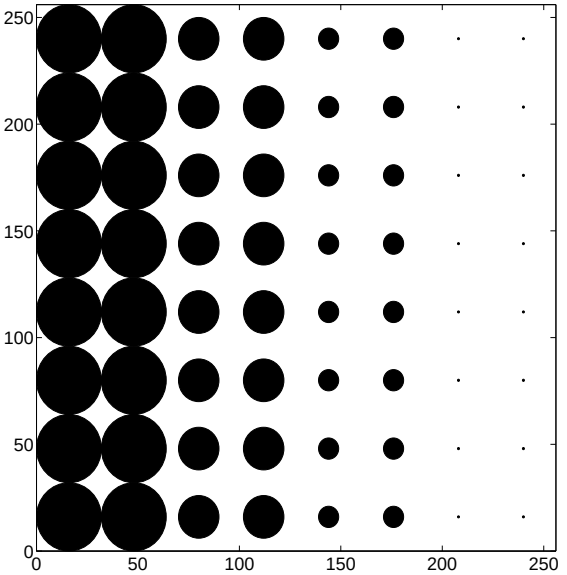
- exchange spatial resolution for quantization
- processing is called half-toning (for printing)

Halftoning

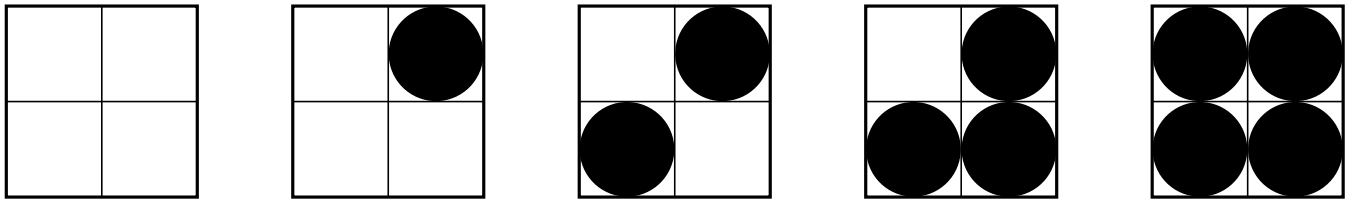
Processing is called half-toning (for printing)

- use dots of varying size





- use patterns of dots



level 1 2 3 4 5

Dithering

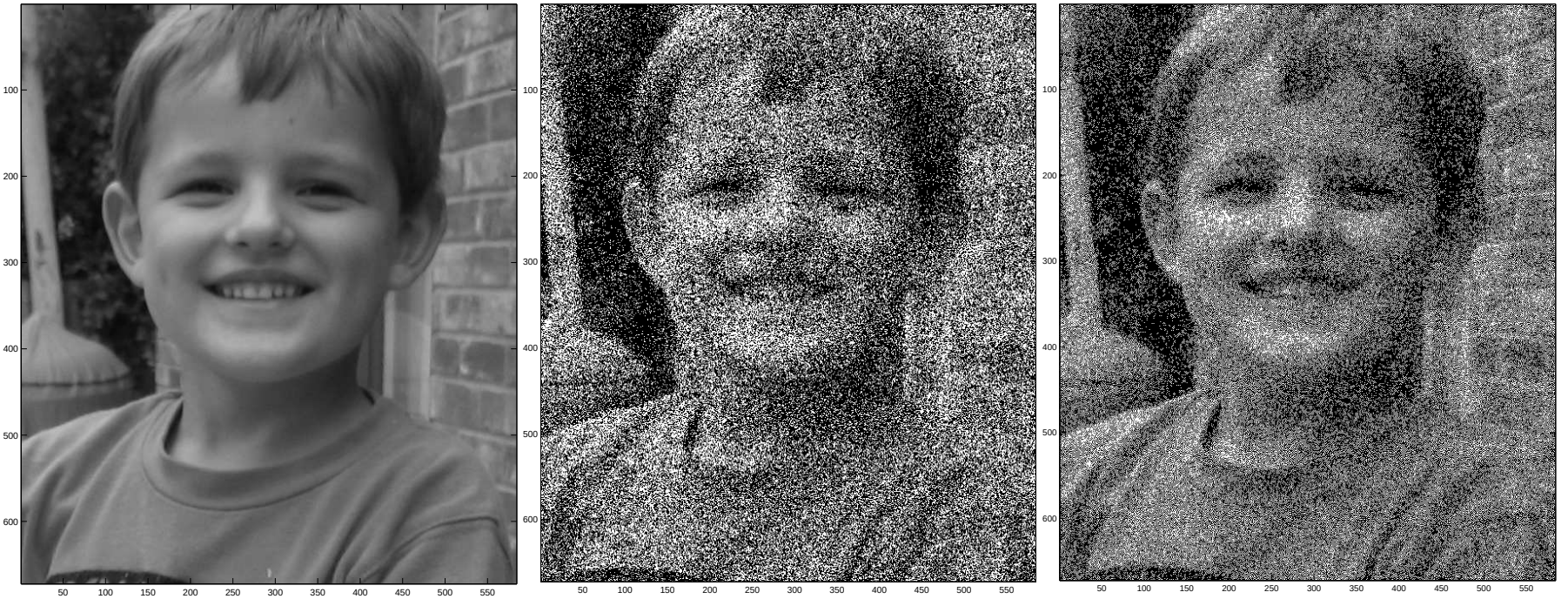
Problem with simple approaches above:

- introduce some patterns into the image
- these might be perceptable
- want to add some randomization
- example:

$$P(x,y) = \text{round}\left[I(x,y) + \text{rand}(x,y)\right]$$

- Example applet
  http://www.markschulze.net/halftone/

Examples of Dithering



1. original image
2. dithered by adding a random value
3. Floyd-Steinberg error diffusion dithering

Extreme example: ascii art

- Sample the image down to grayscale with less than 8-bit precision, and then assign a character for each value.
- Ideally character is related to grayscale
- also shape can be used to help define lines.

Extreme example: ascii art

⇒

```
=IZm#mmWWQWQQQW&>+-:::==vvSnXYY1uXXXSX1isilvvSX1><ii|:=|vv
|vZ#mBWWWWWWWWY~:===||indvvnovooonu1o1iv32owoXuI><|i=+=%v}
oomWWWWWWBWQB(<suqX1vvn*+<omqm#XmXXqZXoXqqZ#Bmmoc||||=ii+=
({$WWWWWWWW#nommZ1|li|:::<mWmmm#X21XZo#mBWWWWWWQmoo2>=il|s
;+3WWWWWWWW&m#WBl|||+++||vXTYYY1lIvvXXX#mmWWWWWBBm2I|vvvvv
:=]QWWQWWmWm#BWel||+|||||||+|||iiIvn2XXZ#BWWWWWWWW1nSSovnn
;;<QWQWQWWW#Wm#ei=:;==|||||||ivvIvIII1X#mWWWQWWWEin1****|
>:<$WWQWWmmmWBZ+=;|ivXXZZXv|ivXmZ##ZXwwwX##WWQWWWk><==>=|l
c:=dWWQWQW1v$#>;<vwm#mWBmZ+:-<X#mWWWWmmm####WWQWWZvviiiiiI
C:;3QWWQWEX#SS;:--~+"!!"+:: .=iI*YYYYYYYSX##WWWWWmovvv<nns
h=>)WmWWQQ2evd>::....:_===_=iiuna>;:;==ivo##mWm###1nvsi2no
Q=1=WWWWWWm%il====i%uoX1IX#mZmm#XXXuauoqX###m##XX+==<vvvvs
W>+=3QWWWWWQa;;|vIvSXmasaawXmmmqwwXZXXZ#U###2XXe=:==|i||ll
Xc;=]WWWWWWWWQp|||iI1YVVX11v13XX#UVS1nXZZ##YXXv>=><%ii>+++
?^=+IWBBWBWWQWWoilil|=:===|illvvviilvXZ##m#=n1i=<=|||i=+++
:====*?TV$QWQWWWsivvi;::--=-=-++=inomX#mmB#=vi|:)ss>|isiIi
;=====|=ivd$WWQ#liiInna>||||iiaawm##mWWWmm#=v%|:)S21vXXXZo
sssvaouwwqmm#WWW>{ollvnXXqqqqmm#mWmWWWWBmWE|2o|:=|IIi*l*Yv
nnauXqd###ZVTT*l|}n%ivnnXZ##mmmBWBWWWWBBW#XoX1|;=innovvvv|
XX1Y!""~..iu2}===aB||v2XXX##X##mmBmm###Z#ZZSoonsiiII1vvvi|
+=:.;;==uZY|=:;:<#(-=<1IllIISZ####X#Z#Z#X2onv1li|||||i=<iv
i|==a|nX2<||====<nas>>|i|svuqqqXXZZXXSooo1vno1llvaonli||I1
+:vZ1oY+iv>==++||ilv2XUU###XXXSXXXX21no1vuZ2lnomZSvii|vlll
aomXva=vl+=|+|=|||IIvnnnoooXXXX2onlvnoIumS1lomZYlisvIvsvnS
Sm#mZiv|=+|=++=|+|ilIvvvoXXSnnvn1ivoIvwZ1ivd#XonnoXS2113#v
XZmZ|||=+||+||=+||ivnoXXXnnvvvvl|v1|vZSulvdZZYsuX2nvvilnZh
dZ#oawwwwpwwwwwwwumqXX1IlIvvI|||%i<w#nZoXqmSvoXX11ili|vXX#
```

Applications of Transforms in 2D

There are many examples of applications for 2D transforms. We examine briefly JPEG image compression and digital watermarks.

Application: JPEG compression

Application: JPEG compression

Example stats

compression	**size (kb)**
no compression	$2304 = 1024 \times 768 \times 24/(8 \times 1024)$
JPEG quality=0.75	62
JPEG quality=0.50	37
JPEG quality=0.25	24
JPEG quality=0.10	13
JPEG quality=0.05	9

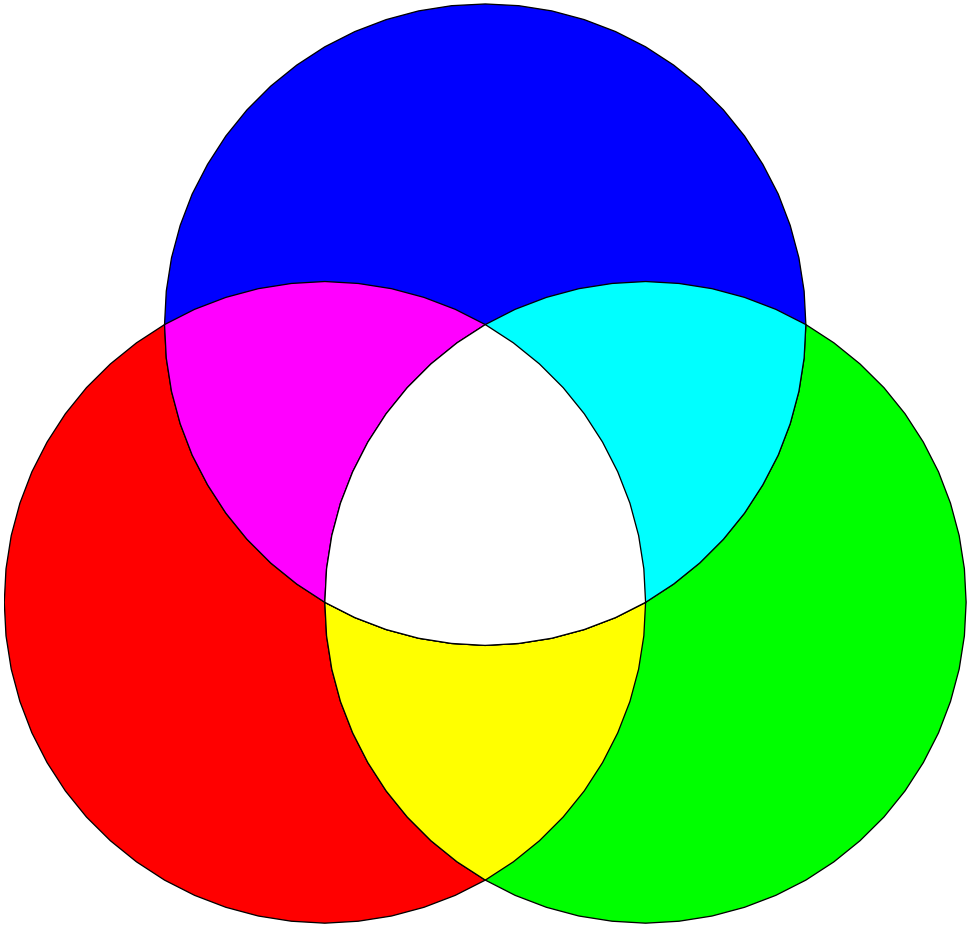
JPEG algorithm

Steps:

- color transform RGB to YIQ, and downsample I, Q
- divide image into blocks of 8x8
- For each 8x8 block
  - DCT
  - quantize
  - encode quantized bits

Color in images

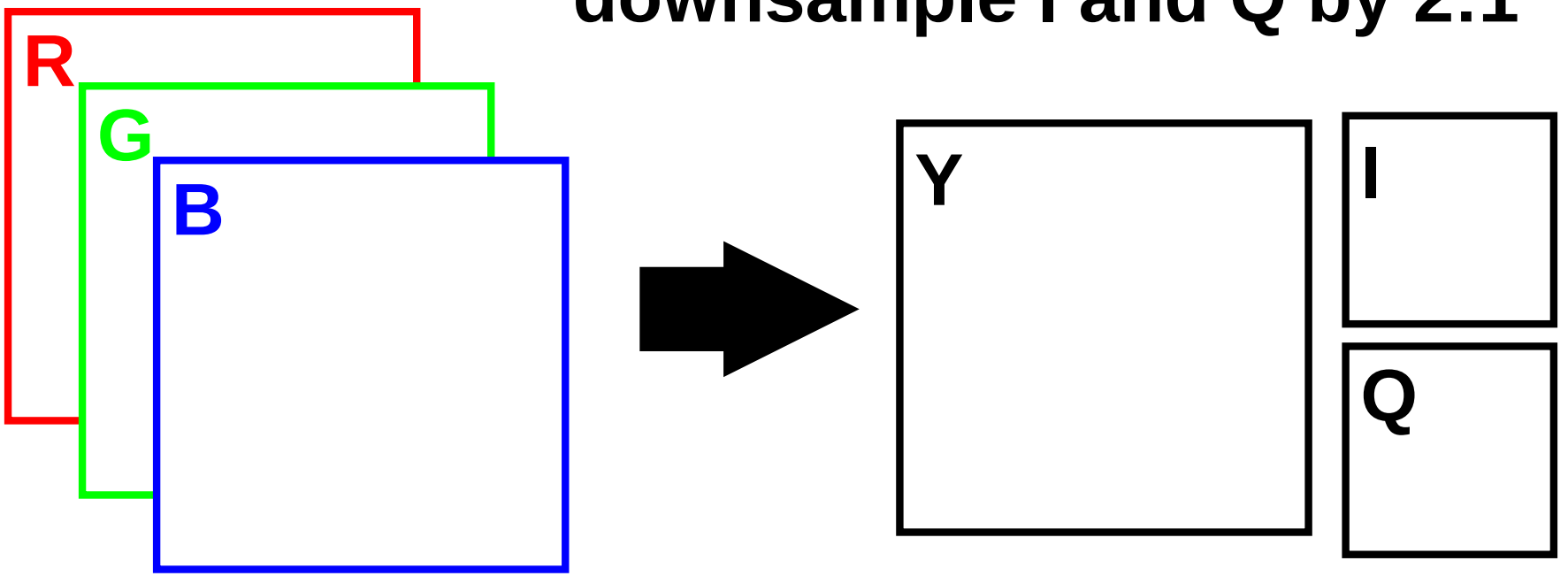
- Primary colors = Red, Green, Blue (RGB)
- combinations of these give colors



Color	Hexadecimal
aqua	#00ffff
gray	#808080
green	#008000
lime	#00ff00
maroon	#800000
navy	#000080
olive	#808000
purple	#800080
red	#ff0000
silver	#c0c0c0
teal	#008080
white	#ffffff
yellow	#ffff00
black	#000000
blue	#0000ff
fuchsia	#ff00ff

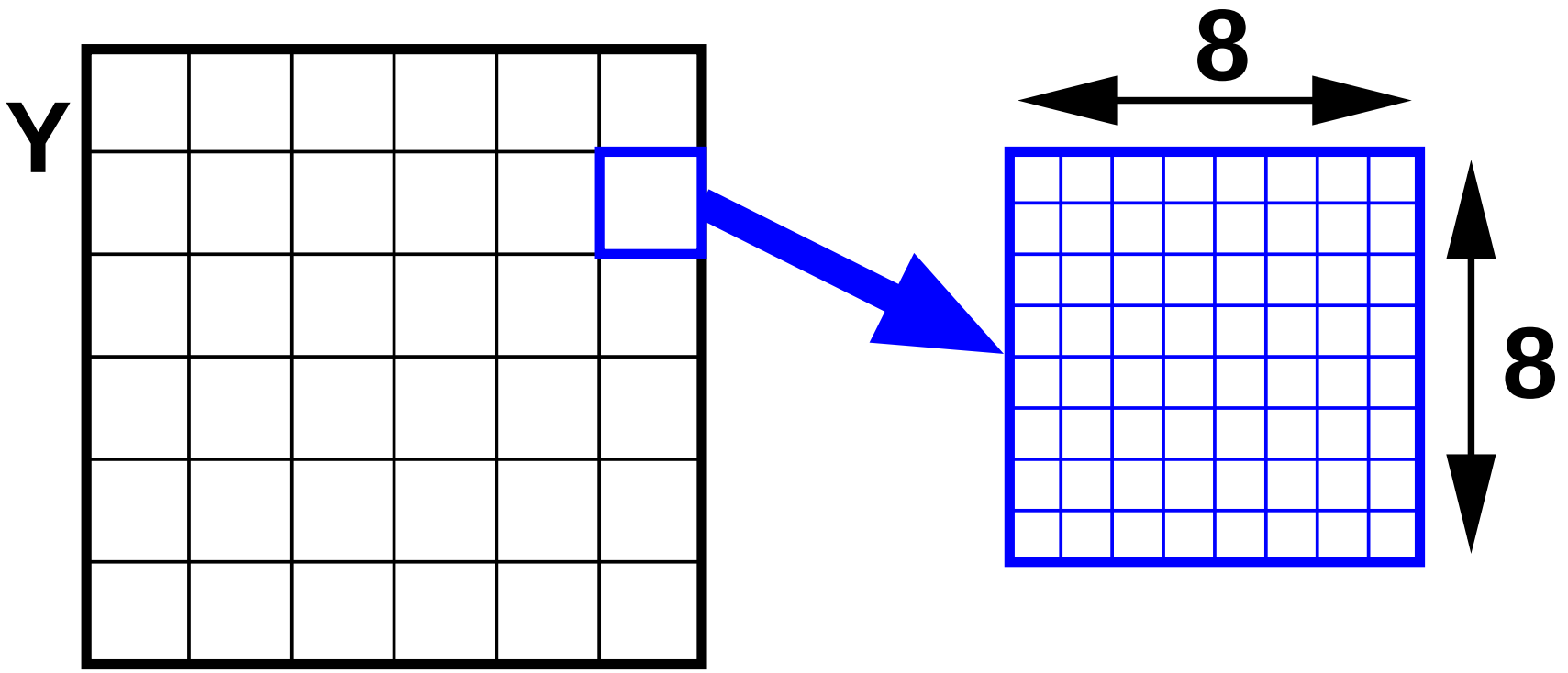
JPEG compression (color transform)

convert RGB to YIQ and downsample I and Q by 2:1



We can downsample I and Q because our eyes are less sensitive to these components.

JPEG compression (blocks)



break image into 8x8 blocks

The Discrete Cosine Transform (DCT

For an $N_1 \times N_2$ image A, the DCT is defined as

$$B(k_1,k_2) = 4 \sum_{i=0}^{N_1-1} \sum_{j=0}^{N_2-1} A(i,j) \cos\left[\pi \frac{k_1}{N_1}\left(i+\frac{1}{2}\right)\right] \cos\left[\pi \frac{k_2}{N_2}\left(j+\frac{1}{2}\right)\right]$$

Real-even DFT of half-shifted input

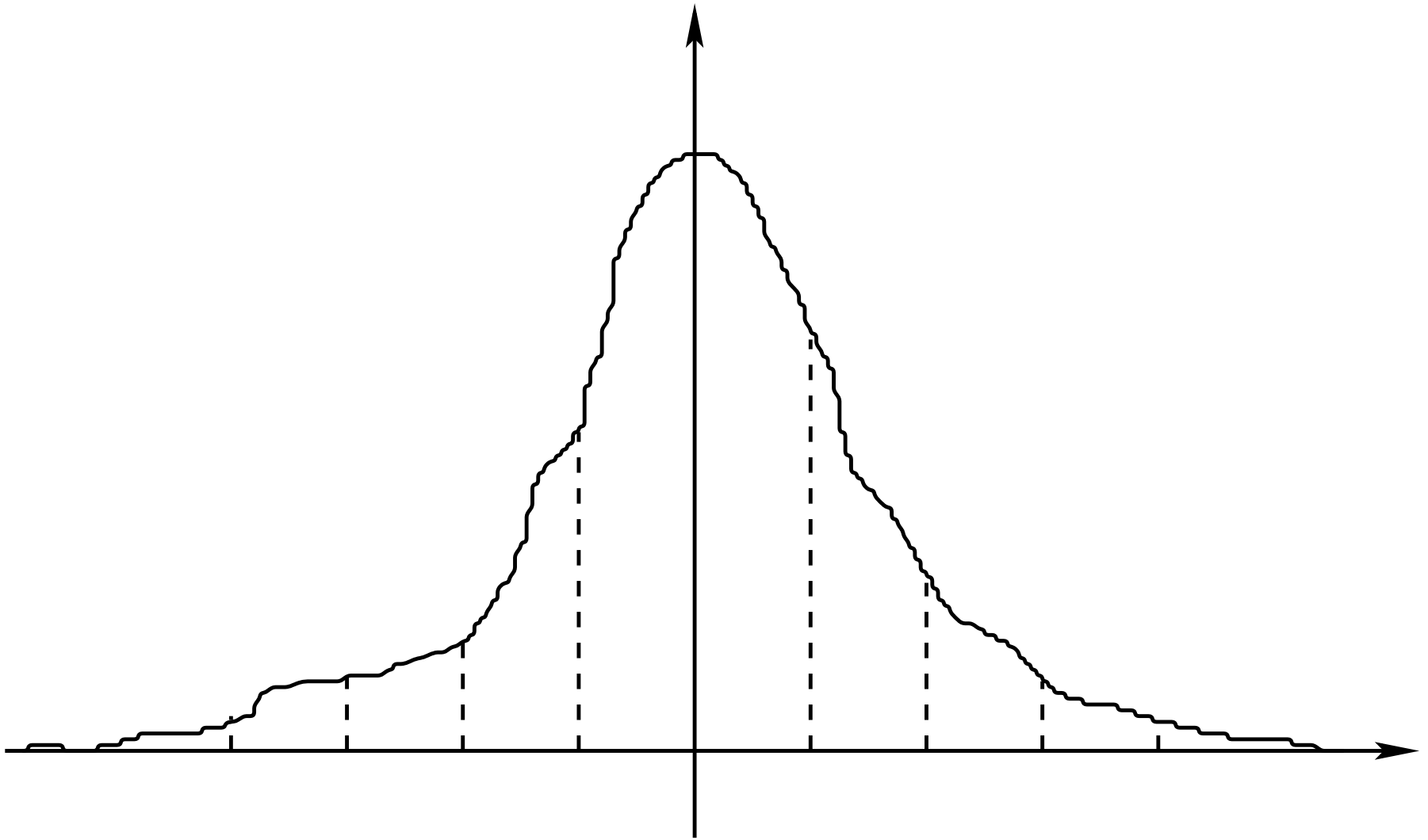
As before there are other possible definitions (e.g. see http://en.wikipedia.org/wiki/Discrete_cosine_transform).

JPEG compression - quantization

- quantization refers to setting the values to discrete values.
- if a small number of values are used they can be encoded in a small number of bits (e.g. 4 values, in 2 bits).
- tradeoff between quality and compression.
- uniform compression applies the same quantization across all DCT coefficients.
- high-frequencies in images aren't as visible to naked eye
- hence quantize higher frequencies more, with minimal loss in quality

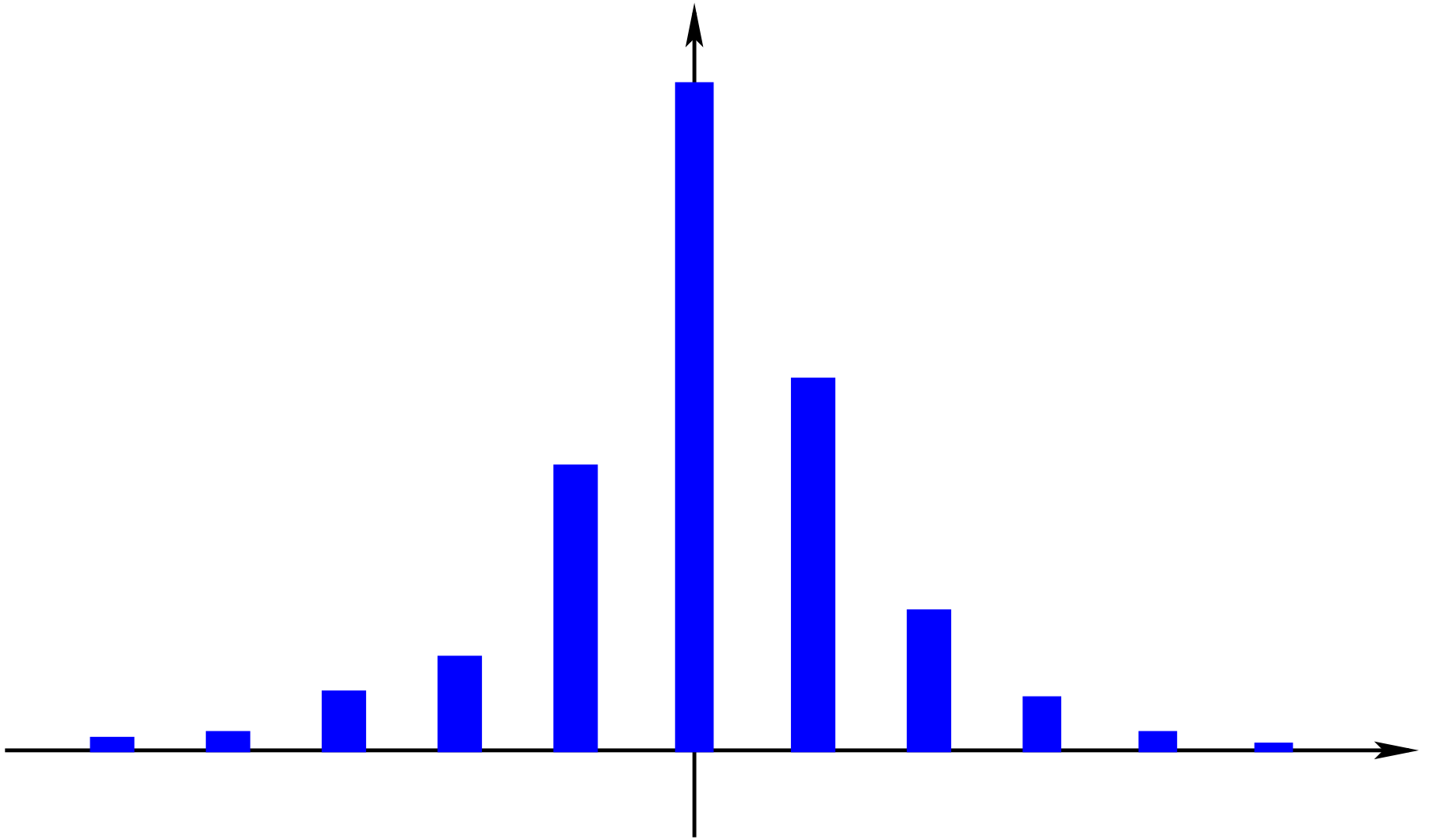
Quantization

We start with some continuous distribution



Quantization

Then quantize the distribution into a number of levels

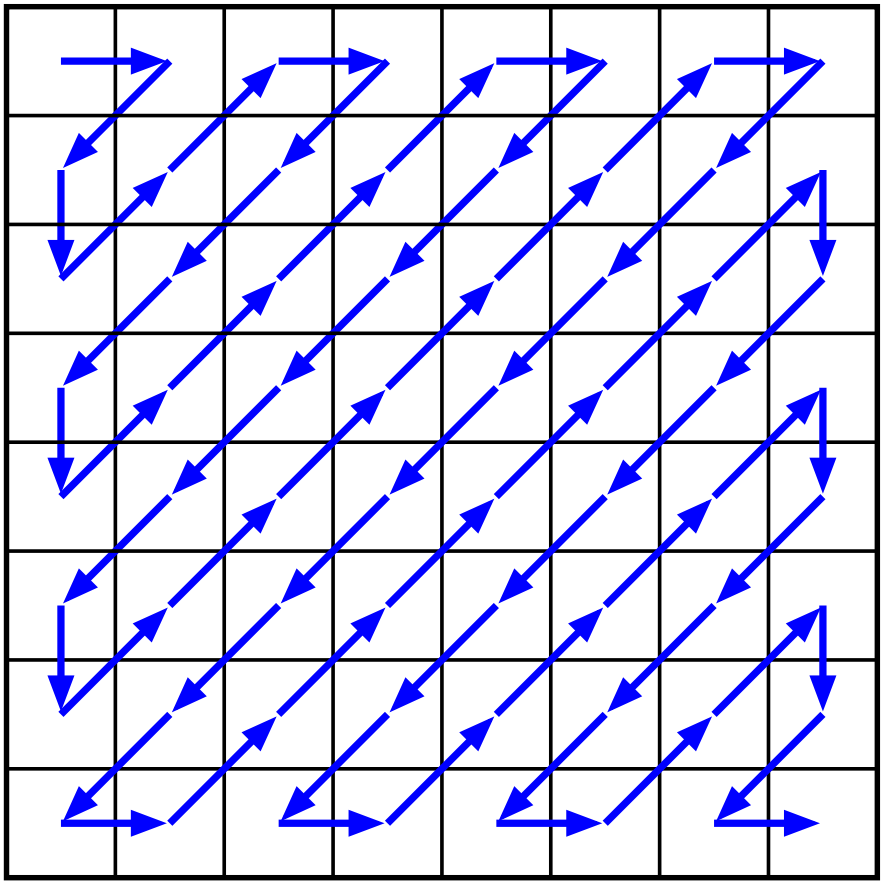


Quantization matrix

- the **quantization matrix** is an 8×8 matrix of step sizes for quantization of the corresponding element of the DCT.
- quantize each element by $\lfloor F[k_1, k_2]/q(k_1, k_2) \rfloor$ where $q(k_1, k_2)$ is the quantization matrix.
- top left are lower frequencies, so smaller q values, increasing towards bottom right.
- typically, many values turn out to be zero: this is good for the next stage.

JPEG compression - Encoding

The levels in the quantized block are encoded sequentially using standard coding techniques, such as Huffman, or Arithmetic encoding, and Run Length Encoding (RLE). The order is important, because the quantization will leave many zeros, mostly in the lower right quadrant. RLE can effectively encode these groups of zeros.



JPEG compression properties

- JPEG compression is lossy (looses information)
- one can't get back to the original just from the compressed information
- loss vs quality is tunable by changing the quantization tables.
- somewhat adaptive (through small blocks), but not very.
- low quality introduces observable artifacts

Application: JPEG compression

Quality = 0.75

Application: JPEG compression

Quality = 0.25

Application: JPEG compression

Quality = 0.10

Application: JPEG compression

Quality = 0.05

Steganography

Steganography: (covered writing) The art and science of hiding information by embedding messages within other, seemingly harmless messages.

- has often been used to code information in text (see following example)
- more recently, used to encode info. in other forms of data
  - images
  - audio
- coding can be done using least significant bits, so it appears as noise
- we can be even cleverer!

Steganography

The German Embassy in Washington, DC, sent these messages in telegrams to their headquarters in Berlin during World War I (Kahn, 1996).

> PRESIDENT'S EMBARGO RULING SHOULD HAVE IMMEDIATE NOTICE. GRAVE SITUATION AFFECTING INTERNATIONAL LAW. STATEMENT FORESHADOWS RUIN OF MANY NEUTRALS. YELLOW JOURNALS UNIFYING NATIONAL EXCITEMENT IMMENSELY.
>
> APPARENTLY NEUTRAL'S PROTEST IS THOROUGHLY DISCOUNTED AND IGNORED. ISMAN HARD HIT. BLOCKADE ISSUE AFFECTS PRETEXT FOR EMBARGO ON BYPRODUCTS, EJECTING SUETS AND VEGETABLE OILS.

Steganography

Reading the first character of every word in the first message or the second character of every word in the second message.

PRESIDENT'S EMBARGO RULING SHOULD HAVE IMMEDIATE NOTICE. GRAVE SITUATION AFFECTING INTERNATIONAL LAW. STATEMENT FORESHADOWS RUIN OF MANY NEUTRALS. YELLOW JOURNALS UNIFYING NATIONAL EXCITEMENT IMMENSELY.

APPARENTLY NEUTRAL'S PROTEST IS THOROUGHLY DISCOUNTED AND IGNORED. ISMAN HARD HIT. BLOCKADE ISSUE AFFECTS PRETEXT FOR EMBARGO ON BYPRODUCTS, EJECTING SUETS AND VEGETABLE OILS.

Steganography

Reading the first character of every word in the first message or the second character of every word in the second message will yield the following hidden text:

PERSHING SAILS FROM N.Y. JUNE 1

Applications: digital watermarks

Watermarks are a particular case: we want to add to an image user data that is

- recoverable (given the key)
- hard to detect (without the key)
- hard to get rid of

allows one to find copyright violations. e.g.

- finding (and proving) that an image on a web page was illegally copied from its owner.

Watermarks

VERY non-trivial to get right (security is always hard).
It must resist natural transformations of the image, e.g.

- image cropping
- recompression
- changing file types
- rotation

Plus be hard for an opponent to

- remove deliberately.
- detect and identify
- forge a watermark

Least significant bits would be lost in something as simple as compression!

Watermarks in the frequency domain

- In JPEG compression, we found that some values could be quantized with minimal visual impact.
- similarly, we could deliberately change the values a little with minimal impact on the image quality
- change low frequency components, as they are less quantized during compression, and less likely to be lost.
- apply to whole image so that cropping doesn't cause a problem
- use crypto techniques on the information to make it hard to access/detect

Applications: digital watermarks

A somewhat easier problem is tamper-proof watermark

- put some mark onto the image such that transformation of the image in any way will damage the mark.
- then if the image has been altered, we can find out
- e.g. for use in chain of evidence, to prove that a digital image was not altered.
- again we can hide this information in the DCT coefficients.