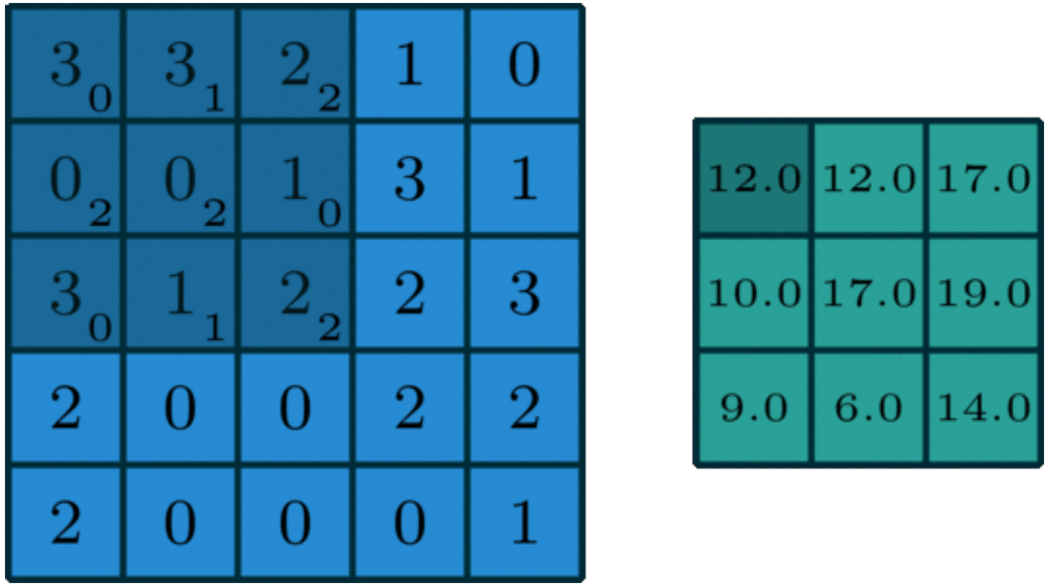
COMS20011 – Data-Driven Computer Science



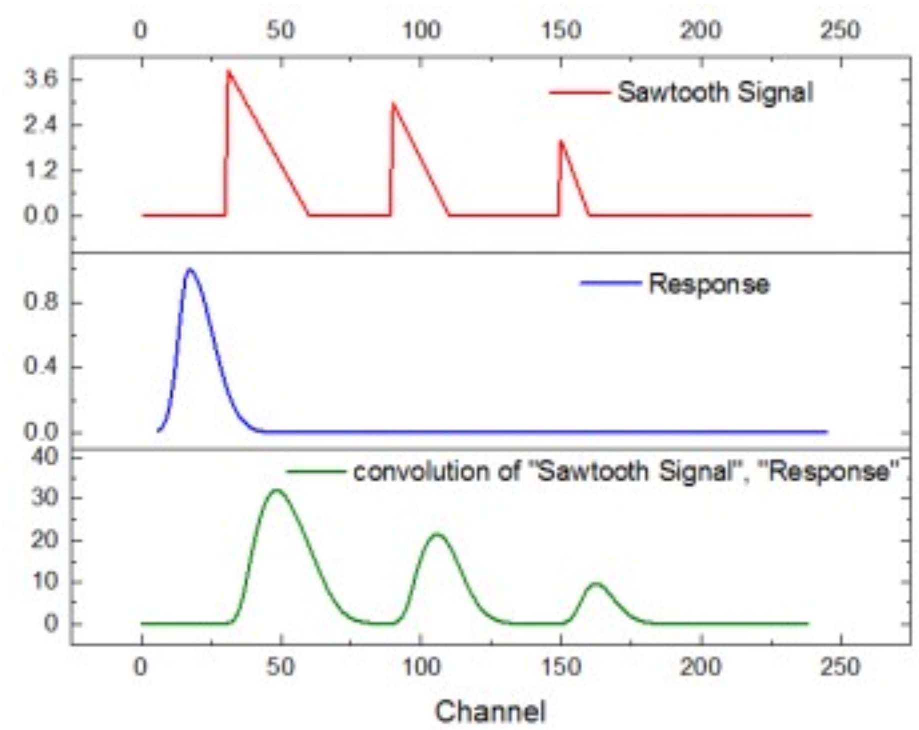
Convolutions

(for feature extraction)

April 2023

Majid Mirmehdi

Next in DDCS



Feature Selection and Extraction

- Signal basics and Fourier Series
- 1D and 2D Fourier Transform
- Another look at features
- **Convolutions**

https://www.originlab.com/doc/Origin-Help/Convolution

Spatial Filtering

We can filter signals and symbols in the spatial/time domain:

- introduce some form of enhancement
  - remove noise/outliers
  - smoothing/averaging out detail
  - sharpening/highlighting detail
- prepare for next stage of processing
  - feature extraction

Filters are also referred to as *kernels* or *masks*.

Spatial Filtering

Many spatial filters are implemented with ***convolution*** masks.

To do convolution, we need to know about ***neighbourhoods***.

Symbolic Data

ATAGACATGGC ......

1D signal data

3	2	4	4	2	6	......

2D signal data

3	2	4	4	2	6
3	4	5	4	3	6
4	2	4	4	3	3
3	2	4	4	2	6
3	2	4	5	2	6

Spatial Filtering

Many spatial filters are implemented with ***convolution*** masks.

To do convolution, we need to know about ***neighbourhoods***.

Symbolic Data

ATAGACATGGC ......

neighbours of G?

1D signal data

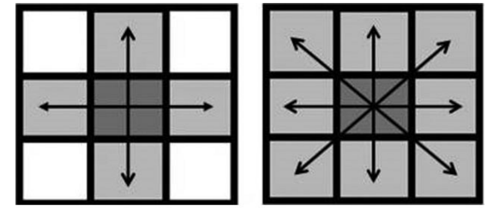
3	2	4	4	2	6	......

2D signal data

3	2	4	4	2	6
3	4	5	4	3	6
4	2	5	4	3	3
3	0	4	1	2	6
3	2	4	5	2	6

Convolution mask is applied to each signal sample and its neighbourhood.

2D Spatial Filtering - Connectivity


(a) (b)

Determine the connectivity for ***neighbourhoods:***

2D signal data

4–connectivity

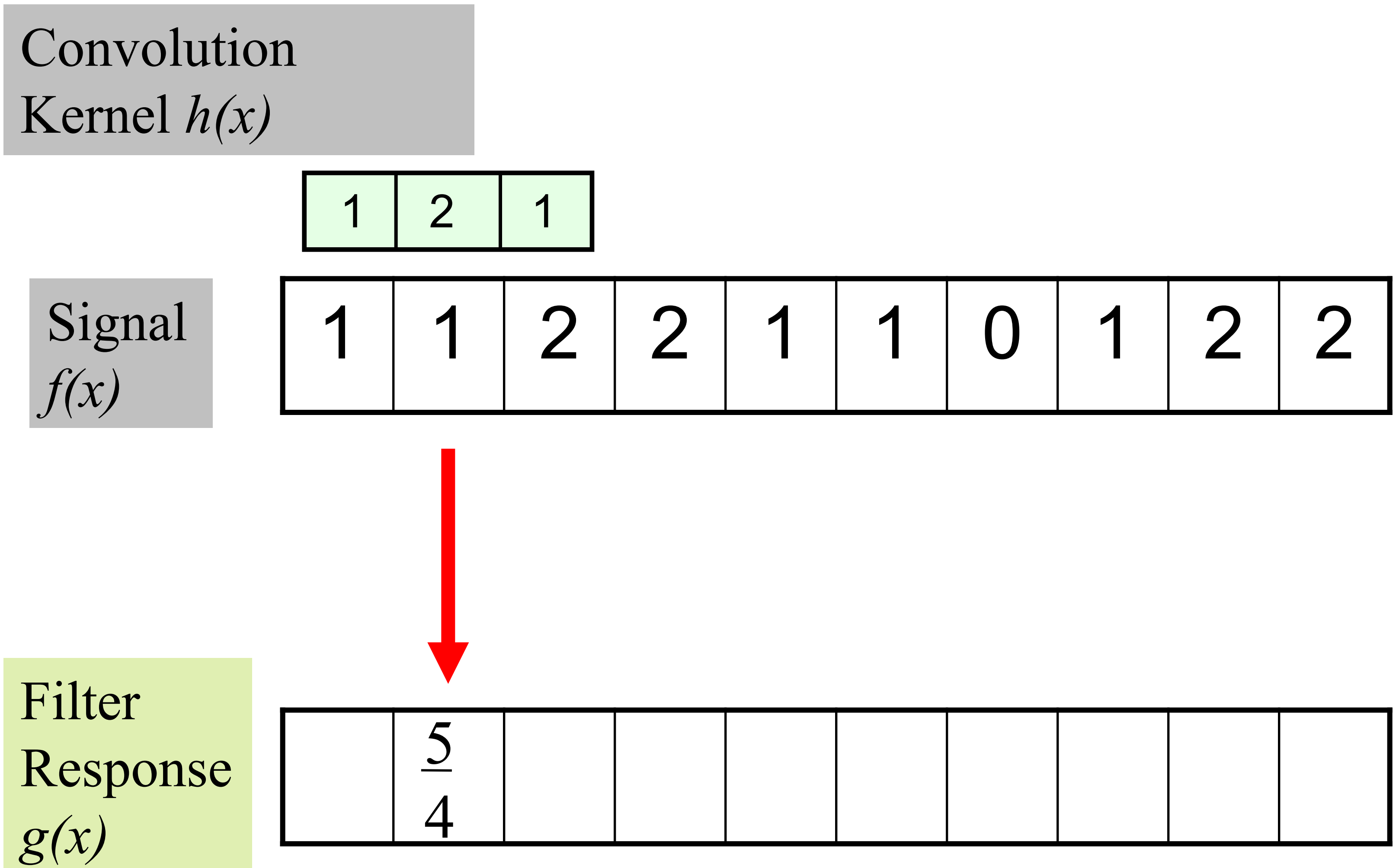
3	2	4	4	2	6
3	4	5	4	3	6
4	2	5	4	3	3
3	0	4	1	2	6
3	2	4	5	2	6

8–connectivity.

3	2	4	4	2	6
3	4	5	4	3	6
4	2	5	4	3	3
3	0	4	1	2	6
3	2	4	5	2	6

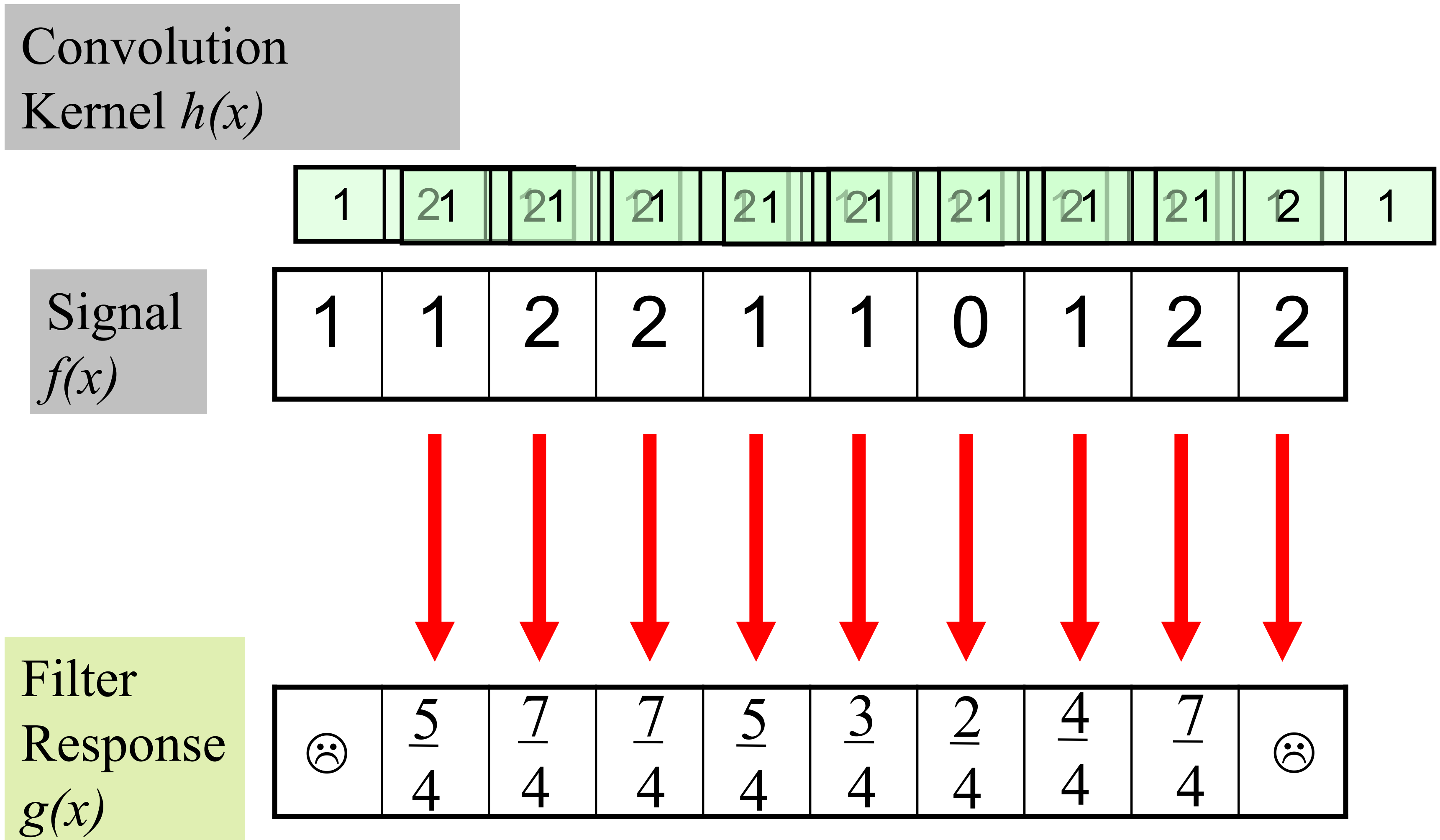
Convolution/Correlation

$$g(x) = h(x) * f(x)$$



Convolution

$$g(x) = h(x)*f(x)$$



Convolution

$$g(x) = h(x)*f(x)$$

- f is the signal, h is the convolution filter
  - h has an origin

$$\frac{1}{5}\begin{bmatrix} -1 & 3 & -1 \end{bmatrix}$$

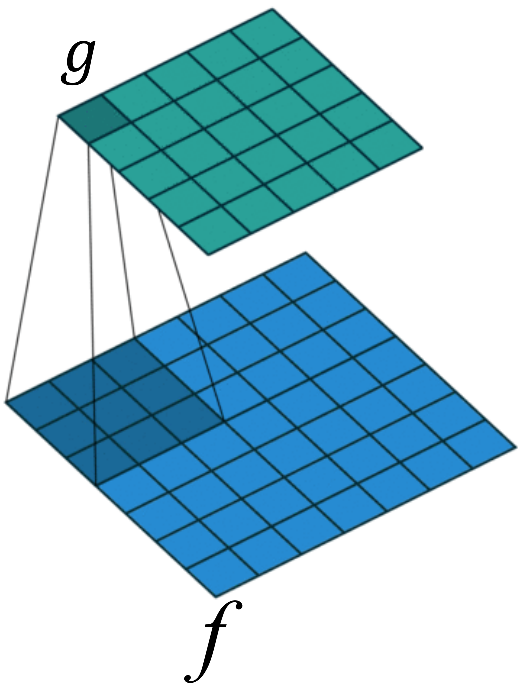
Example 1D kernel

  - Normalization factor (sum of the absolute values of the filter) is also part of the filter!

- The discrete version of convolution is defined as:

$$g(x) = \sum_{m=-s}^{s} f(x-m)h(m) \qquad \text{for } s \geq 1$$

Spatial Filtering using 2D Convolution



Simple example

f

3_0	3_1	2_2	1	0
0_2	0_2	1_0	3	1
3_0	1_1	2_2	2	3
2	0	0	2	2
2	0	0	0	1

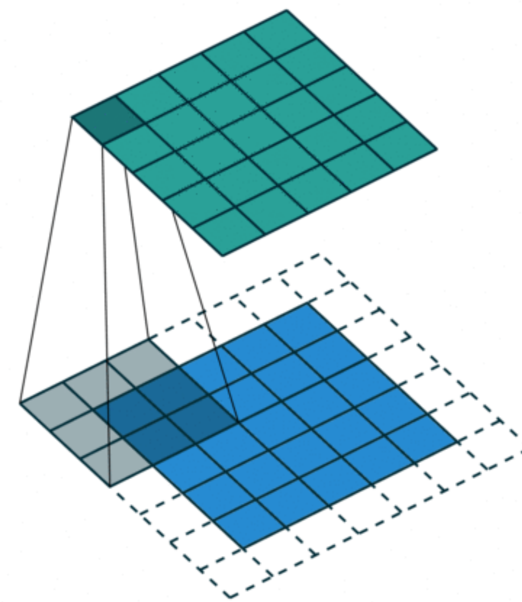
h

0	**1**	**2**
2	**2**	**0**
0	**1**	**2**

g

12.0	12.0	17.0
10.0	17.0	19.0
9.0	6.0	14.0

Use padding for
same size result



https://towardsdatascience.com/intuitively-understanding-convolutions-for-deep-learning-1f6f42faee1

2D Convolution

- The discrete version of 2D convolution is defined as

$$g(x,y)=\sum_{m=-1}^{1}\sum_{n=-1}^{1} f(x-m,y-n)h(m,n)$$

Shorthand form:

$$g=f*h$$

Convolution symbol

h

	-1	0	1
-1	**-1**	**0**	**1**
0	**-2**	**0**	**2**
1	**-1**	**0**	**1**

f

		$y-1$	y	$y+1$	
$x-1$		**43**	**12**	**61**	
x		**44**	**45**	**60**	
$x+1$		**43**	**50**	**61**	

-68

$$\begin{aligned}
& f(x+1,y+1)h(-1,-1) \\
&+ f(x+1,y)h(-1,0) \\
&+ f(x+1,y-1)h(-1,1) \\
&+ f(x,y+1)h(0,-1) \\
&+ f(x,y)h(0,0) \\
&+ f(x,y-1)h(0,1) \\
&+ f(x-1,y+1)h(1,-1) \\
&+ f(x-1,y)h(1,0) \\
&+ f(x-1,y-1)h(1,1)
\end{aligned}$$

2D Correlation

- The discrete version of 2D correlation is defined as

$$g(x,y)=\sum_{m=-1}^{1}\sum_{n=-1}^{1} f(x+m,y+n)h(m,n)$$

h

	-1	0	1
-1	**-1**	**0**	**1**
0	**-2**	**0**	**2**
1	**-1**	**0**	**1**

f

		$y-1$	y	$y+1$	
$x-1$		**43**	**12**	**61**	
x		**44**	**45**	**60**	
$x+1$		**43**	**50**	**61**	

68

Correlation=Convolution when kernel is symmetric under 180° rotation, e.g.

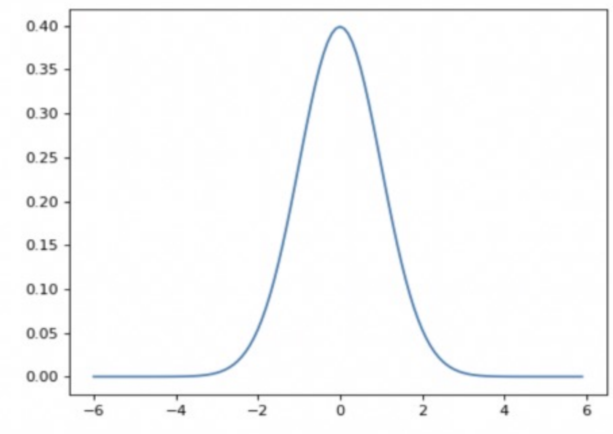
1	**2**	**1**

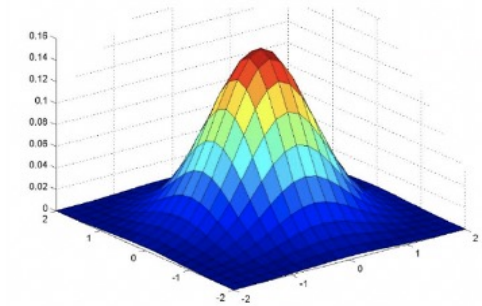
Example: Spatial Low/High Pass Filtering

- 1D: turning the treble/bass knob down on audio equipment!
- 2D: smooth/sharpen image

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \frac{1}{16}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

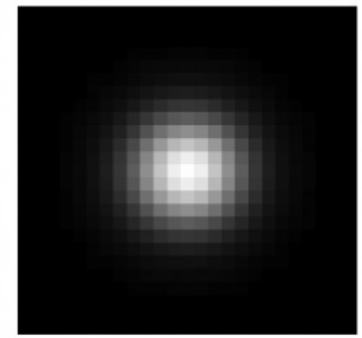
Gaussian Low Pass Filter





Gaussian kernel

$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$



Example:

$\frac{1}{16} \times$

1	2	1
2	4	2
1	2	1

$$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix} = \frac{1}{16}\begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix}\begin{pmatrix} 1 & 2 & 1 \end{pmatrix}$$

Example: Edge Features

$$f(x, y)$$

- Edges occur in images where there is discontinuity (or change) in the intensity function.
- Gradient points in the direction of most rapid change in intensity
- Biggest change → derivative has maximum magnitude.

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

Sobel Edge Detector

- 2D gradient measurement in two different directions.
- Uses these 3x3 convolution masks:

-1	0	+1
-2	0	+2
-1	0	+1

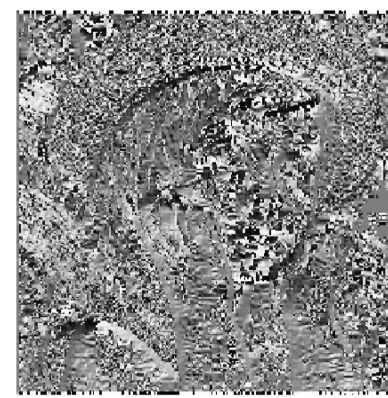
$*$

$\frac{\partial f}{\partial x} =$

Thresholded result

$\|\nabla f\|$

θ



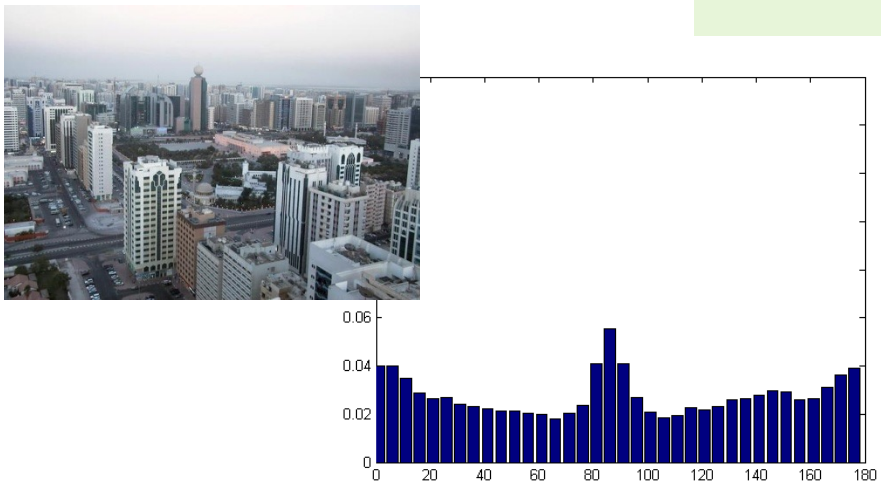
+1	+2	+1
0	0	0
-1	-2	-1

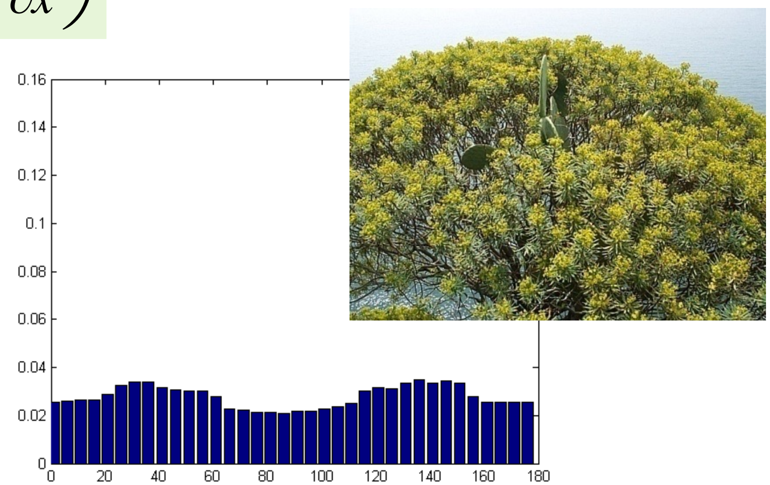
$*$

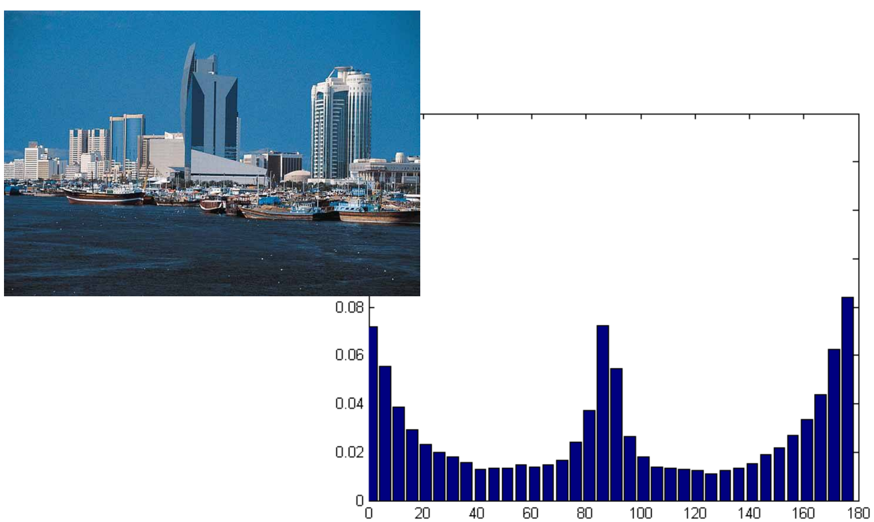
$\frac{\partial f}{\partial y} =$

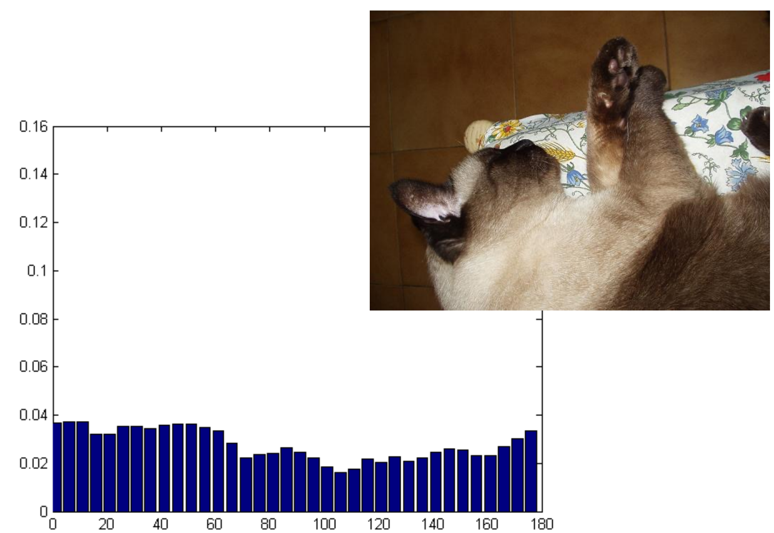
Histogram of Edge Gradient Directions

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$









Matlab: Sobel Edge Detection

```
% Sobel edge detection
A = imread('romina.gif');
fx = [-1 0 1; -2 0 2; -1 0 1]
fy = [1 2 1; 0 0 0; -1 -2 -1]
gx = conv2(double(A),double(fx))/8;
gy = conv2(double(A),double(fy))/8;
mag = sqrt((gx).^2+(gy).^2);
ang = atan(gy./gx);
figure; imagesc(mag); axis off; colormap gray
figure; imagesc(ang); axis off; colormap gray
```

See unit github page for code in Python.

Spatial/Frequency Domain Filtering

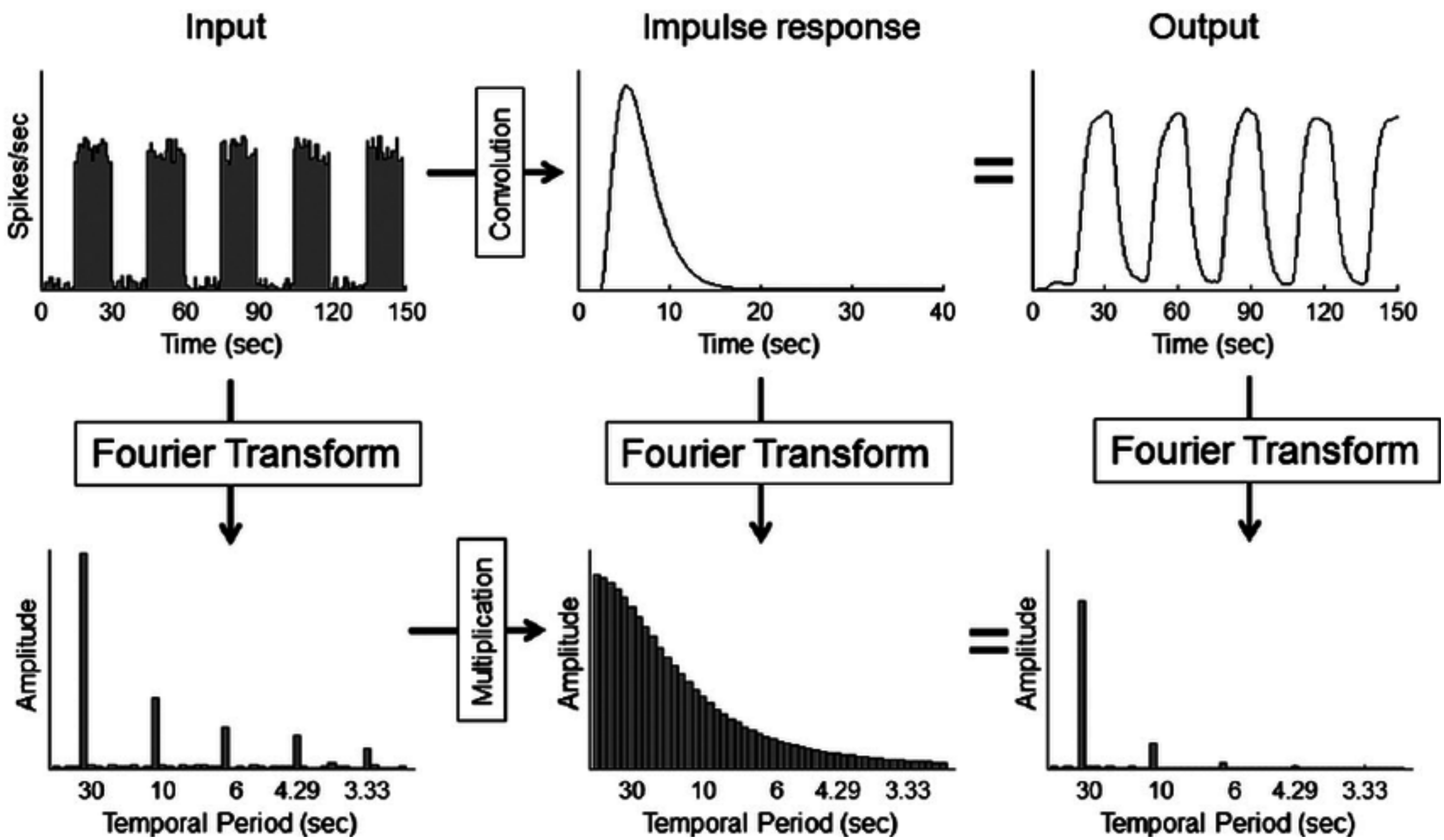
- Convolution Theorem:

Convolution in the spatial domain
is equivalent to
multiplication in the frequency domain
and vice versa

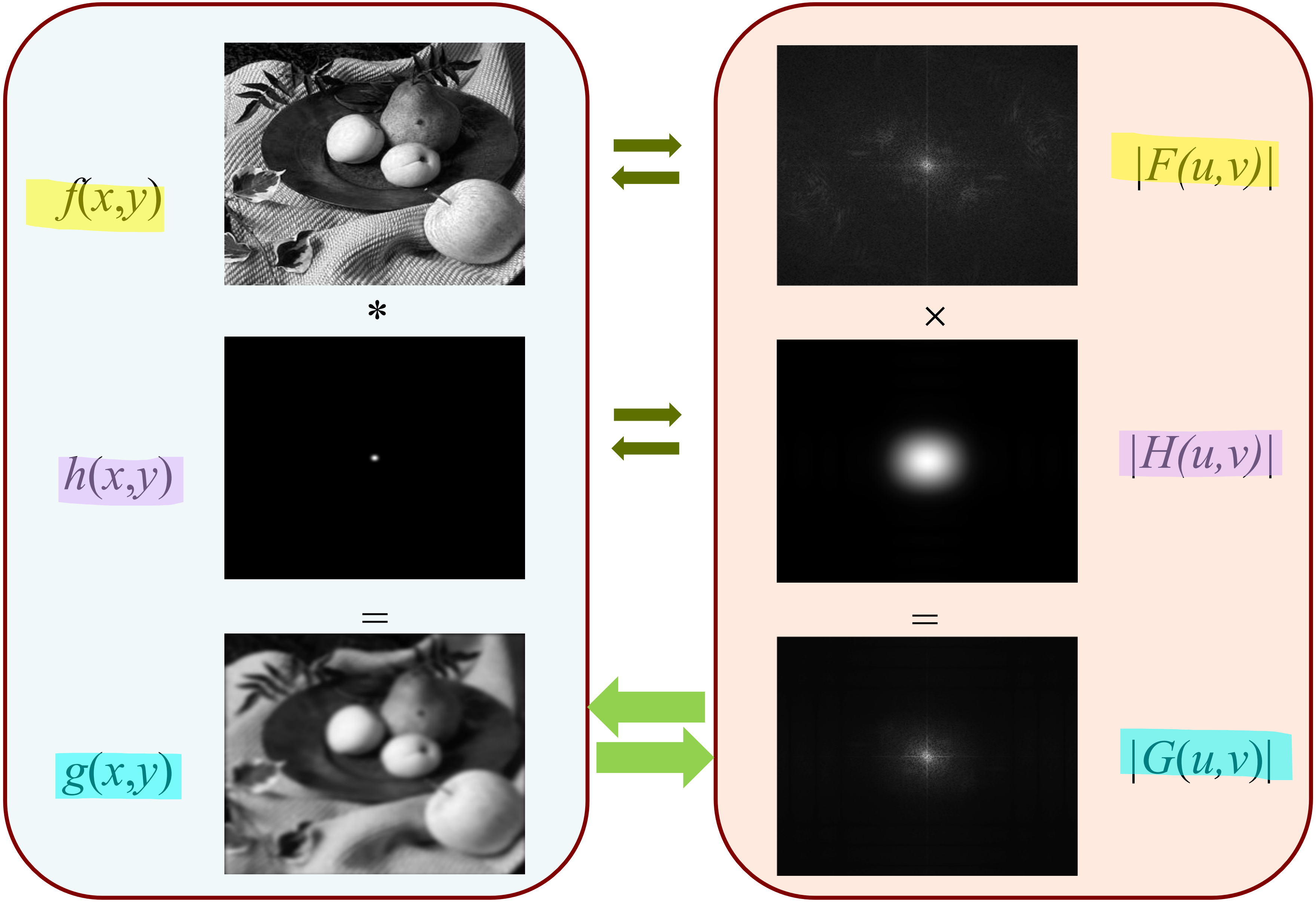
$$g(x,y) = f(x,y) * h(x,y) \quad \Longleftrightarrow \quad G(u,v) = F(u,v)\,H(u,v)$$

$$g(x,y) = f(x,y)\,h(x,y) \quad \Longleftrightarrow \quad G(u,v) = F(u,v) * H(u,v)$$

Example: Convolution in SD is Multiplication in FD



Example: Convolution in SD is Multiplication in FD

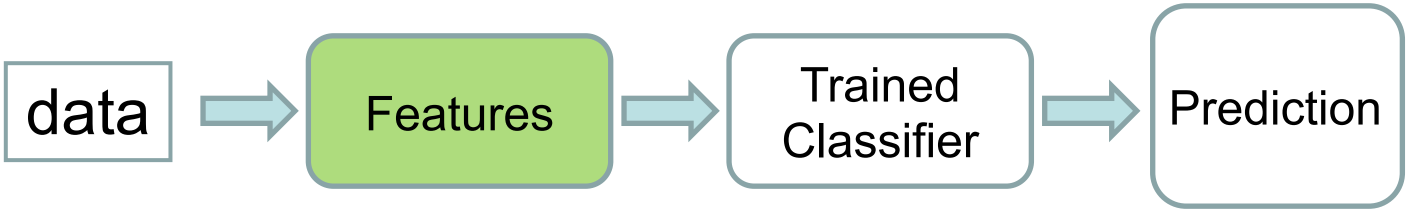


2D Fourier Transform (in Matlab)

```
f = imread('barbara.gif');    %read in image
z = fft2(double(f));          % do fourier transform
q = fftshift(z);              % puts u=0,v=0 in the centre
Magq = abs(q);                % magnitude spectrum
Phaseq=angle(q);              % phase spectrum
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Usually for viewing purposes:
imagesc(log(abs(q)+1));
colorbar;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
w = ifft2(ifftshift(q));      % do inverse fourier transform
imagesc(w);
```

See unit github page for code in Python.

What we covered



Feature Selection and Extraction

- Signal basics and Fourier Series
- 1D and 2D Fourier Transform
- Another look at features
- Convolutions