Fourier Analysis.

$$f(t) = \sum_{n=1}^{N} A_n \sin(2\pi n t + \phi_n)$$

$$= \sum_{n=1}^{N} a_n \cos(2\pi n t) + b_n \sin(2\pi n t)$$

$\sin(\alpha + \beta) = \sin\alpha \cos\beta + \cos\alpha \sin\beta$

$$= \frac{a_0}{2} + \sum_{n=1}^{N} a_n \cos(2\pi n t) + b_n \sin(2\pi n t)$$

$$f(t) = \sum_{n=-N}^{N} c_n e^{2\pi i n t}$$

?

$$c_k = \int_0^1 e^{-2\pi i k t} f(t)\, d(t)$$

For period T

$$f(t) = \sum_{-n=N}^{N} c_n e^{2\pi i n t/T}$$

$$c_k = \frac{1}{T}\int_0^T e^{-2\pi i n t/T} f(t)\, dt$$

DFT / FFT

2 3 4 1 3 1 7 8

x x x x x x x x

Image Gradients

Computational Photography (CSCI 3240U) & Computer Vision (CSCI 4220U)

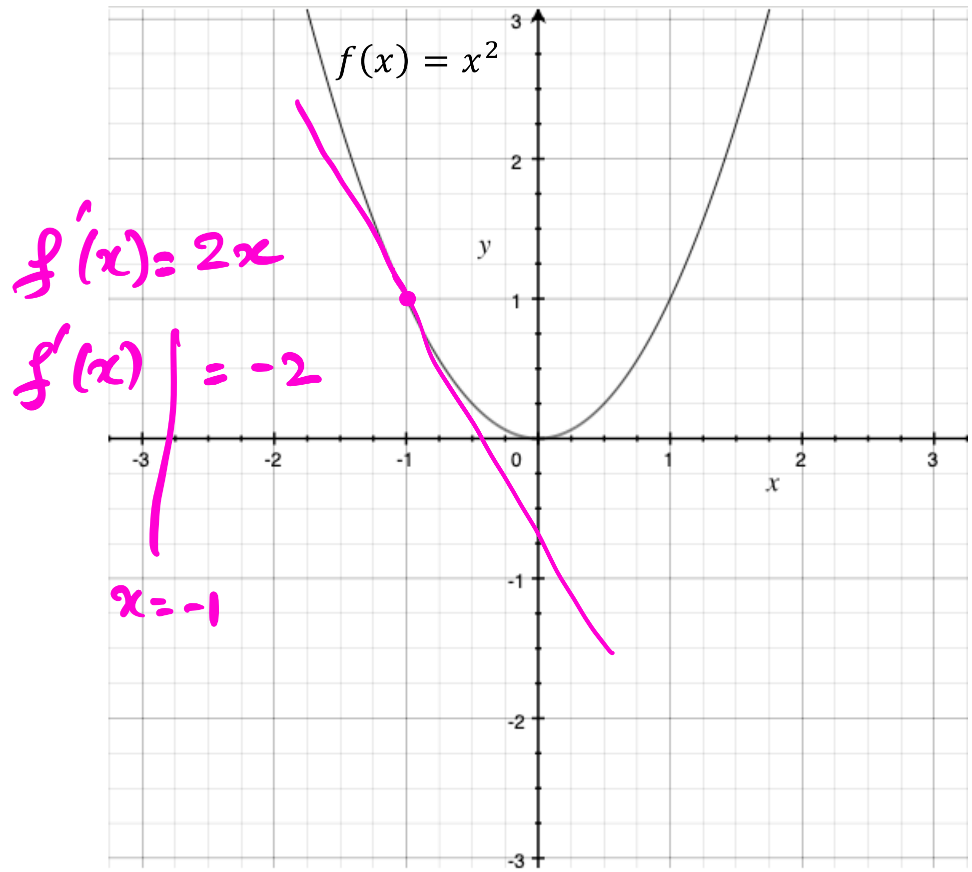
Faisal Z. Qureshi
http://vclab.science.ontariotechu.ca
OntarioTech UNIVERSITY

Today's lecture

- Why do we care about image gradients?
- Computing image gradients
- Sobel filters
- Gradient magnitude and directions
- Visualizing image gradients

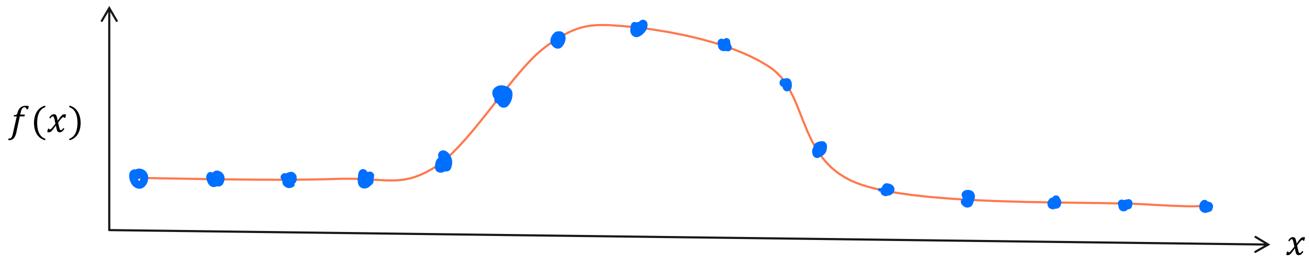
Derivative

$$\frac{df}{dx} = \lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon}$$



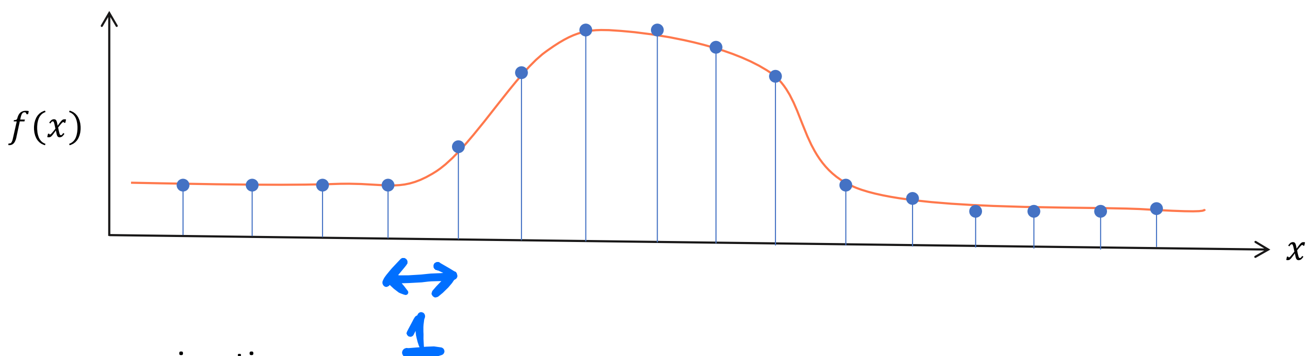
Derivative

$$\frac{df}{dx} = \lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon}$$



Derivative

$$\frac{df}{dx} = \lim_{\epsilon \to 0} \frac{f(x+\epsilon) - f(x)}{\epsilon}$$



Finite-difference approximation

$$\frac{df}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f(x+1) - f(x)}{(x+1) - x} = f(x+1) - f(x)$$

$\Delta x = 1$

Use finite difference approximation to compute image derivatives

$$\frac{df}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f(x+1) - f(x)}{(x+1) - x} = f(x+1) - f(x)$$

	0	1	2	3	4	5	6
$I =$	**1**	**1**	**9**	**8**	**6**	**0**	**0**
$\frac{dI}{dx}$: $I' =$	0	8	-1	-2	-6	0	?
$I'' =$	+8	-9	-1	-4	6	?	?

Use finite difference approximation to compute image derivatives

$$\frac{df}{dx} \approx \frac{\Delta f}{\Delta x} = \frac{f(x+1) - f(x)}{(x+1) - x} = f(x+1) - f(x)$$

$I =$

1	1	9	8	6	0	0
0	1	2	3	4	5	6

-1 1
-1 1

$I' =$

0	8	-1	-2	-6	0	?

$I * [1, -1] =$

0	8	-1	-2	-6	0	//////

FLIP

Finite Diff. Approx.

Partial derivatives

$$f(x,y,z) = 3x^3y + zy - 3z^3$$

$$\begin{bmatrix} \dfrac{\partial f}{\partial x} = 9x^2y \\[2ex] \dfrac{\partial f}{\partial y} = 3x^3 + z \\[2ex] \dfrac{\partial f}{\partial z} = y - 9z^2 \end{bmatrix}$$

→ gradient vector

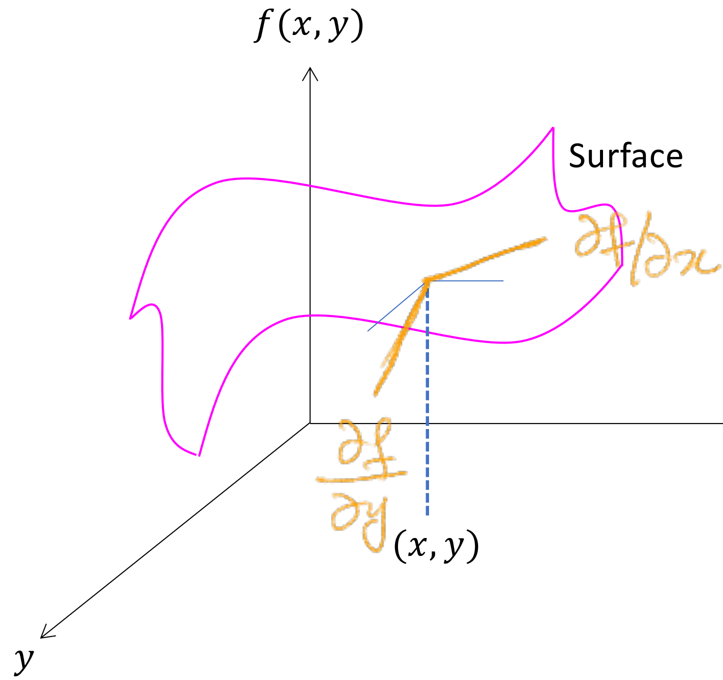


Image derivatives in x and y directions

$I =$

1	1	9	8	1
8	8	8	8	8
1	3	5	8	1
5	3	2	8	6

Aside:

$I_x = I * [1, -1]$

$= CC(I, [-1, 1])$

$I_x = I * [1, -1] =$

0	8	-1		

$I_y = I * [1, -1]^T =$

7	7			

Image gradient ∇I

$$\nabla I = \left[\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y}\right]$$

$I_x =$

0	8	-1	-7	
0	0	0	0	
2	2	3	-7	
-2	-1	6	-2	

$I_y =$

7	7	-1	0	7
-7	-5	-3	0	7
4	0	-3	0	5

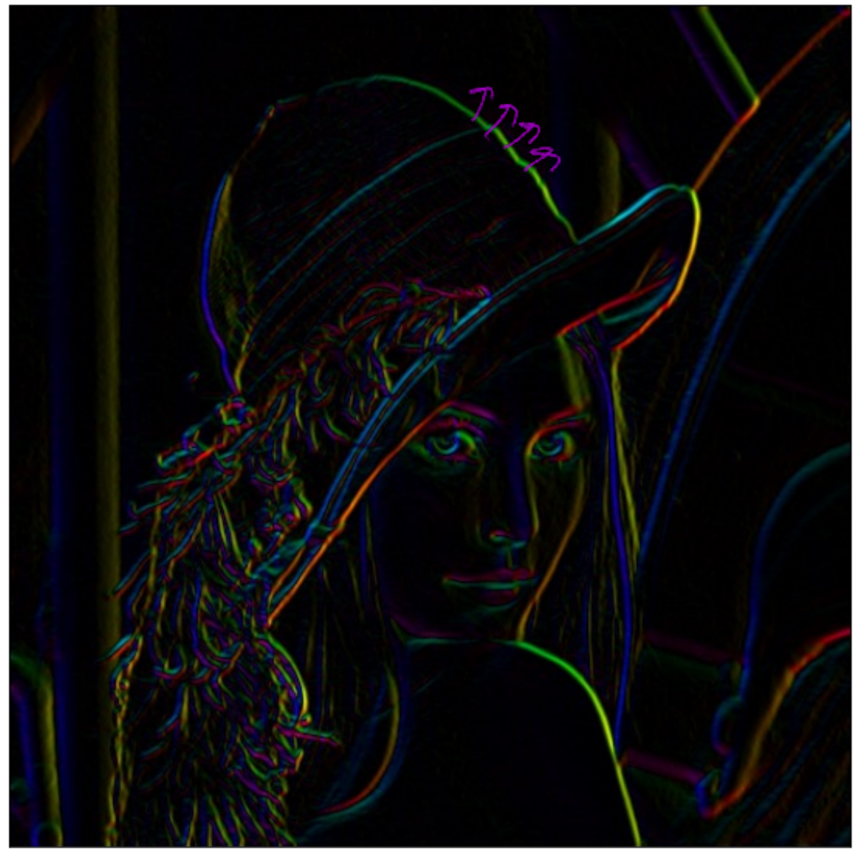
$I =$

1	1	9	8	1
8	8	8	8	8
1	3	5	8	1
5	3	2	8	6

Gradient direction and magnitude

$$\|\nabla I\| = \sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$$ magnitude

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



Filters for computing image derivatives

Sobel

$H_x =$

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

$H_y =$

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

Prewire

$H_x =$

1	0	-1
1	0	-1
1	0	-1

$H_y =$

1	1	1
0	0	0
-1	-1	-1

Roberts

$H_x =$

0	-1
1	0

$H_y =$

1	0
0	-1

Image gradients

- Image derivatives and gradients highlight edge pixels

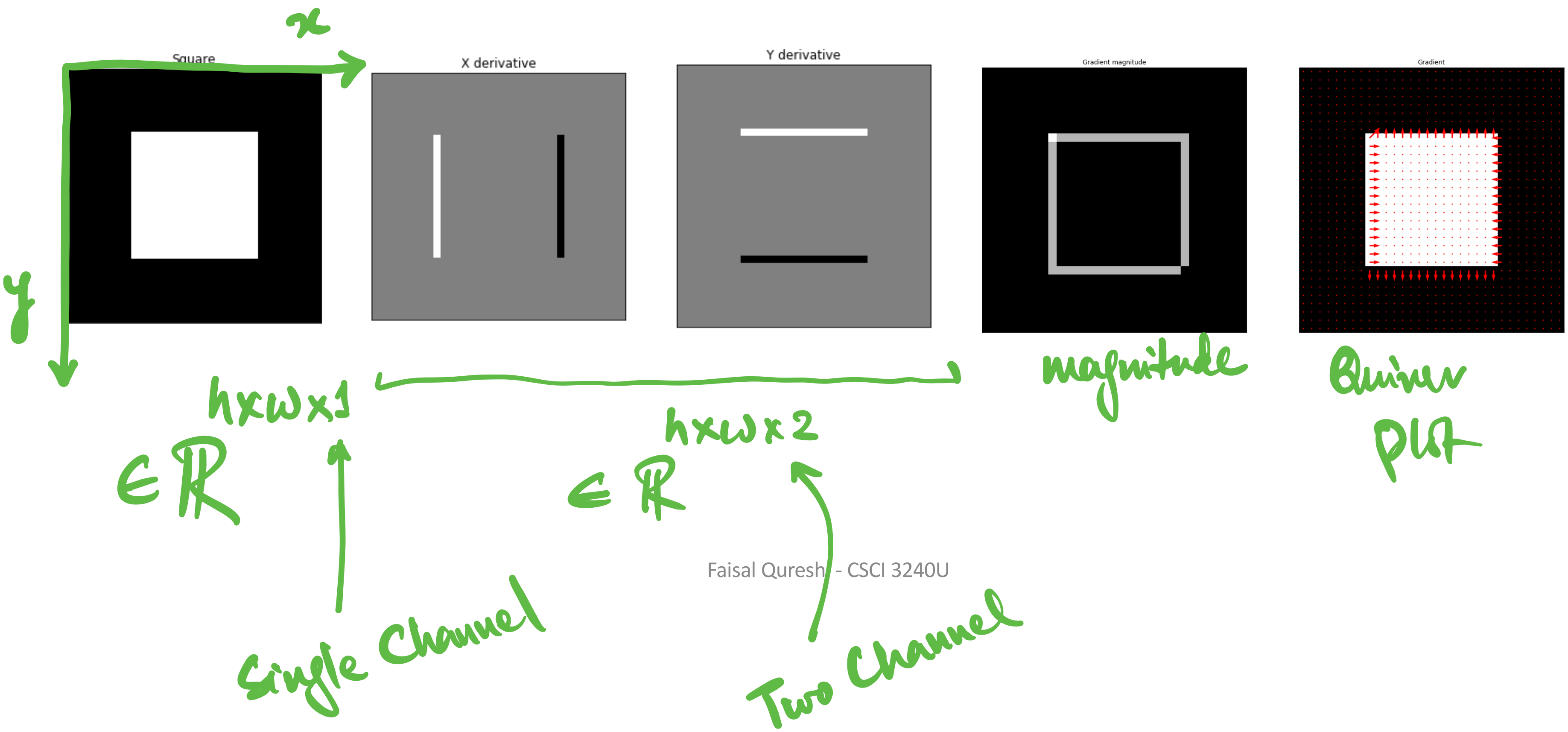
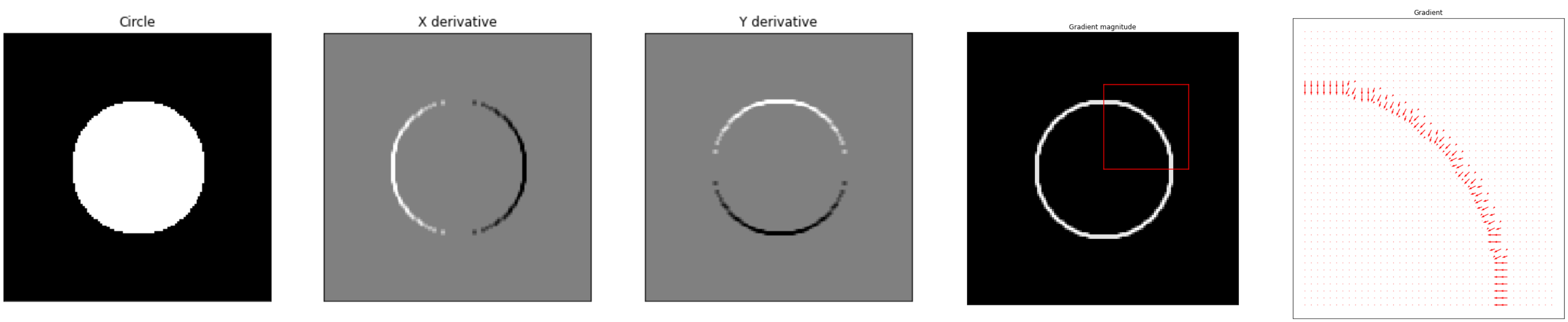


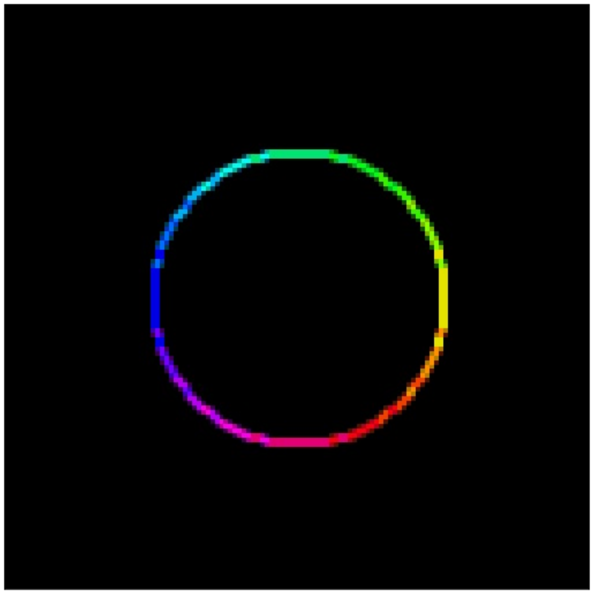
Image gradients

- Image derivatives and gradients highlight edge pixels



Visualizing image gradients

- Use color to visualize gradients (or any 2D field)
  http://csundergrad.science.uoit.ca/courses/cv-notes/notebooks/07-image-derivatives.html



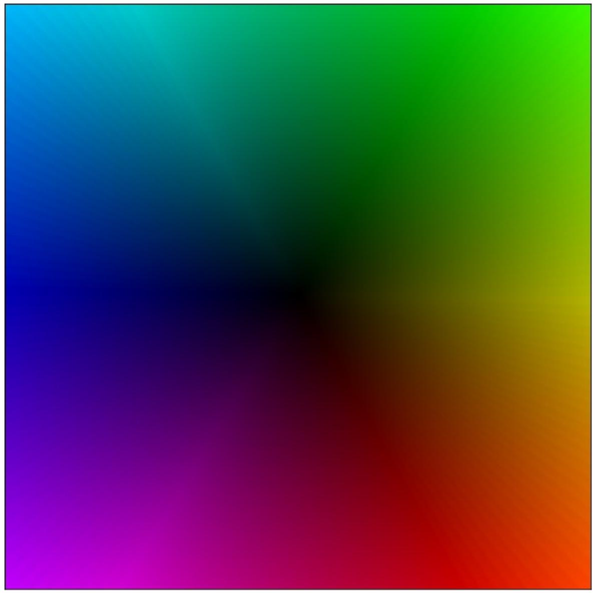
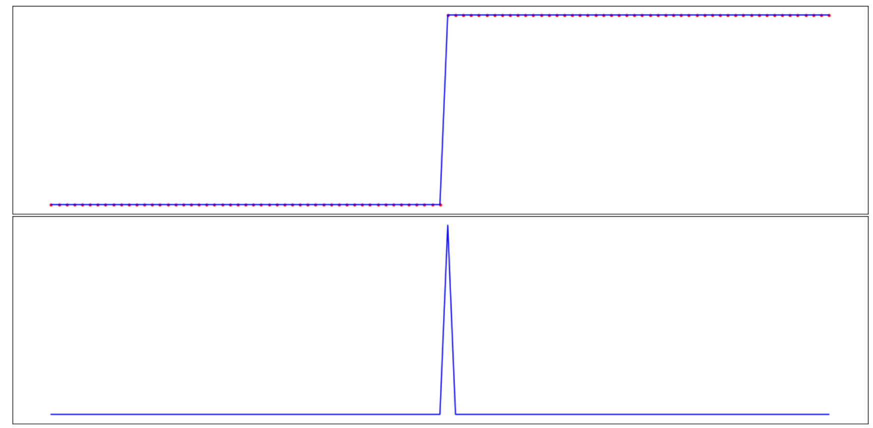
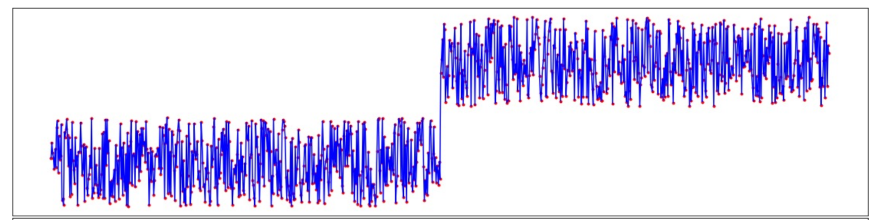
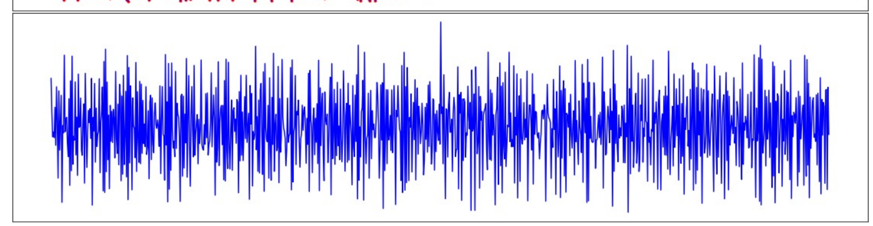


Image noise and gradients





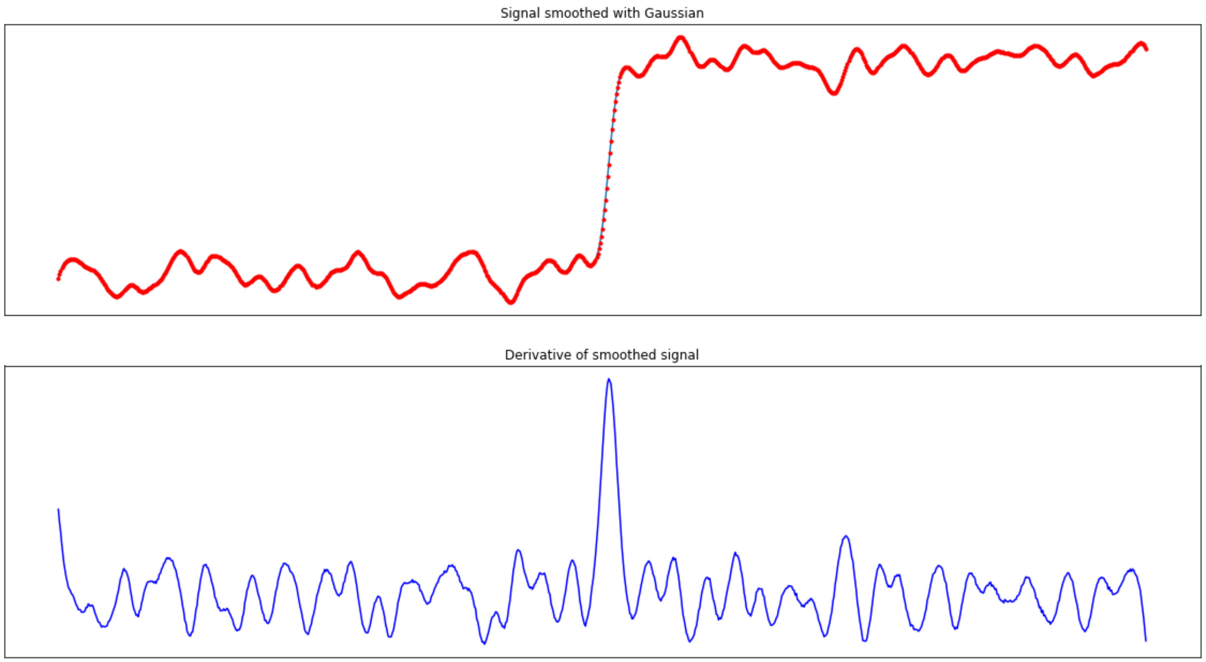
Noisy signal



Using gradient to find edges in the noisy signal

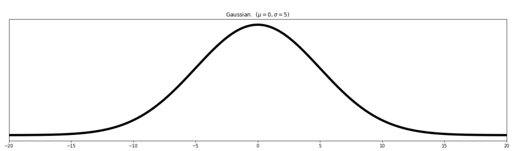
Computing gradients in practice

- Gaussian blur the signal before computing gradients

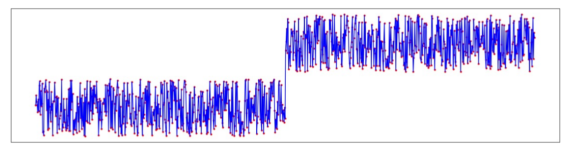


Computing gradients in practice

- Option 1
  - Filter the signal with a Gaussian kernel
  - Filter the signal with an appropriate kernel to compute gradient

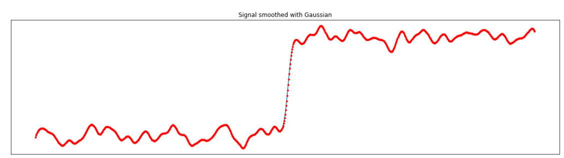


*



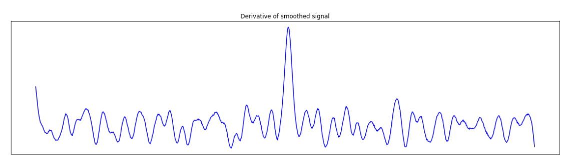
Noisy signal

=



After filtering with a Gaussian kernel

1st derivative of the smoothed signal

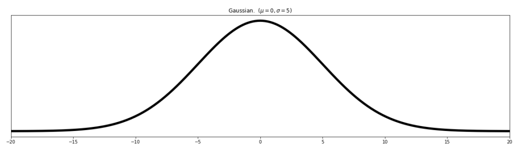


Computing gradients in practice

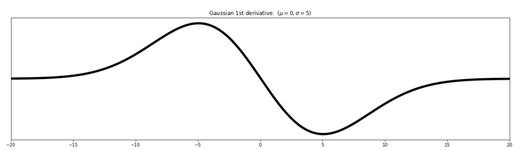
First derivative of the signal

- Option 2: use superposition principle
  - Compute derivative of the Gaussian filter and store the result
  - Filter the (noisy) signal with derivative of the Gaussian

Gaussian



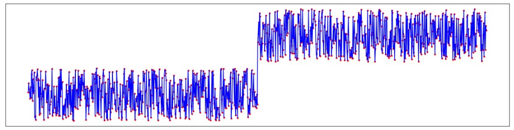
1st derivative of the Gaussian



Stored

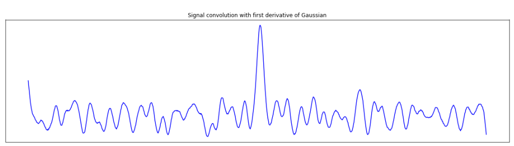
Saves one convolution at runtime

*



Noisy signal

=



1st derivative of the signal

Summary

- Image gradients
- Finite-difference approximation filters
- Gradient magnitude and direction
- Image noise and gradients