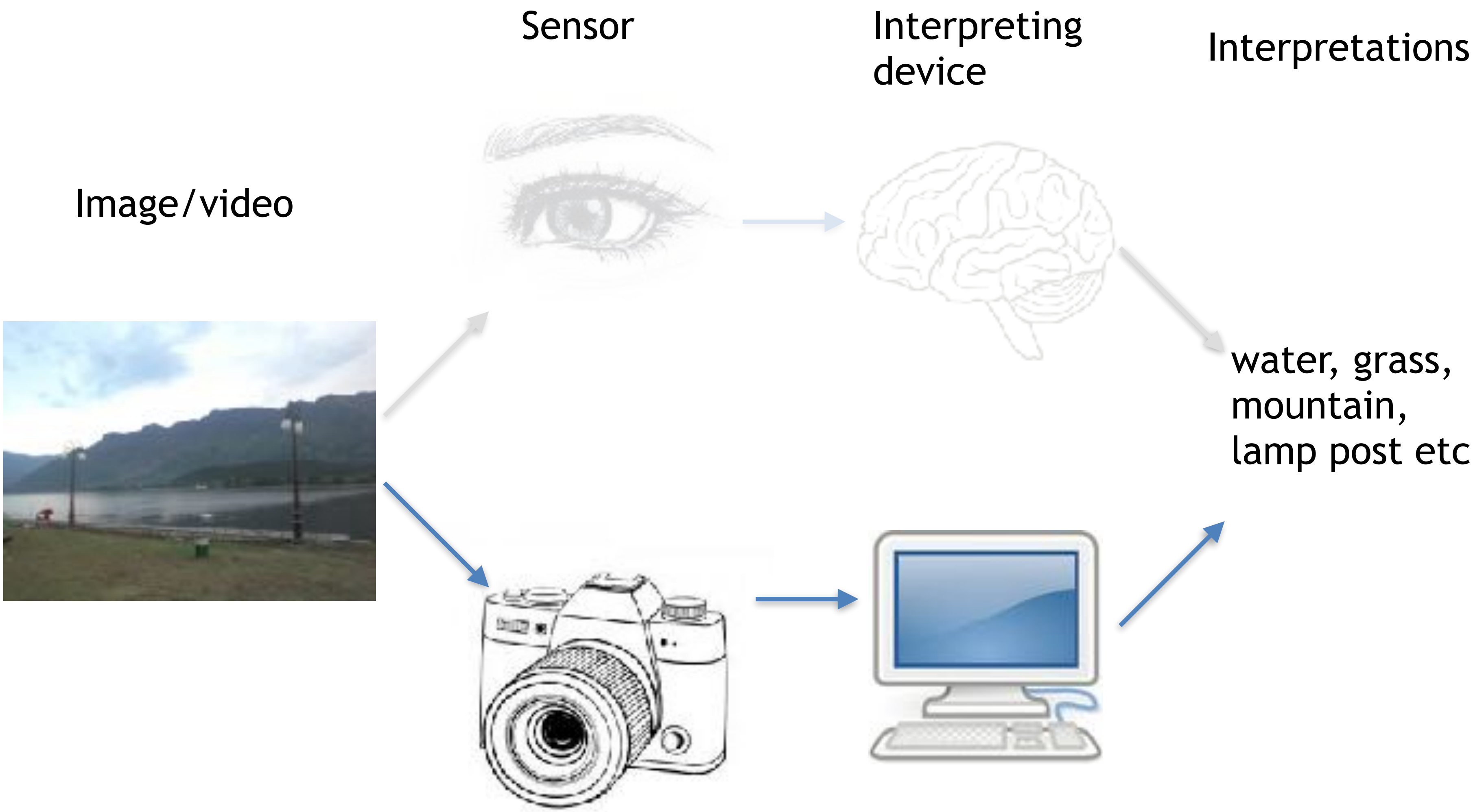
CMSC 426 - Review

Fall 2019

What is Computer Vision



Goal of Computer Vision

what we see

what computers see

230	25	34	123
45	0	10	52
65	11	210	42
78	87	56	90
23	18	29	61

Matrices

Transpose:

$$C_{m\times n} = A^T{}_{n\times m}$$

$$c_{ij} = a_{ji}$$

$$(A+B)^T = A^T + B^T$$

$$(AB)^T = B^T A^T$$

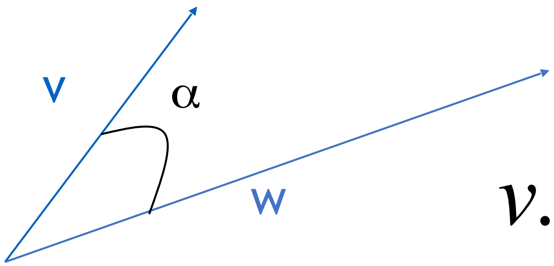
If $A^T = A$ A is symmetric

Examples:

$$\begin{bmatrix} 6 & 2 \\ 1 & 5 \end{bmatrix}^T = \begin{bmatrix} 6 & 1 \\ 2 & 5 \end{bmatrix}$$

$$\begin{bmatrix} 6 & 2 \\ 1 & 5 \\ 3 & 8 \end{bmatrix}^T = \begin{bmatrix} 6 & 1 & 3 \\ 2 & 5 & 8 \end{bmatrix}$$

Inner (dot) Product



$$v.w = (x_1, x_2).(y_1, y_2) = x_1 y_1 + x_2 . y_2$$

The inner product is a SCALAR!

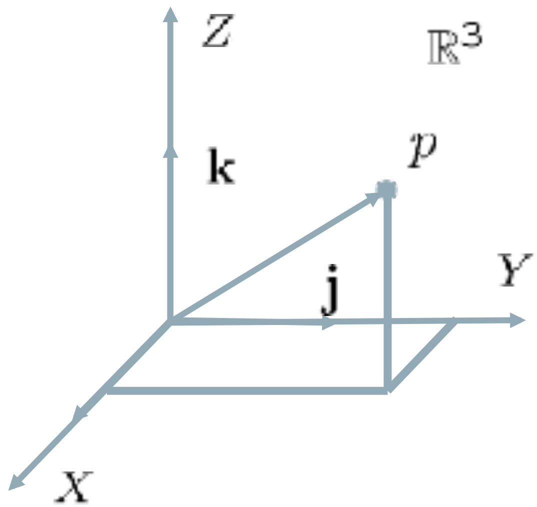
$$v.w = (x_1, x_2).(y_1, y_2) = \| v \| \cdot \| w \| \cos\alpha$$

$$v.w = 0 \Leftrightarrow v \perp w$$

Orthonormal Basis in 3D

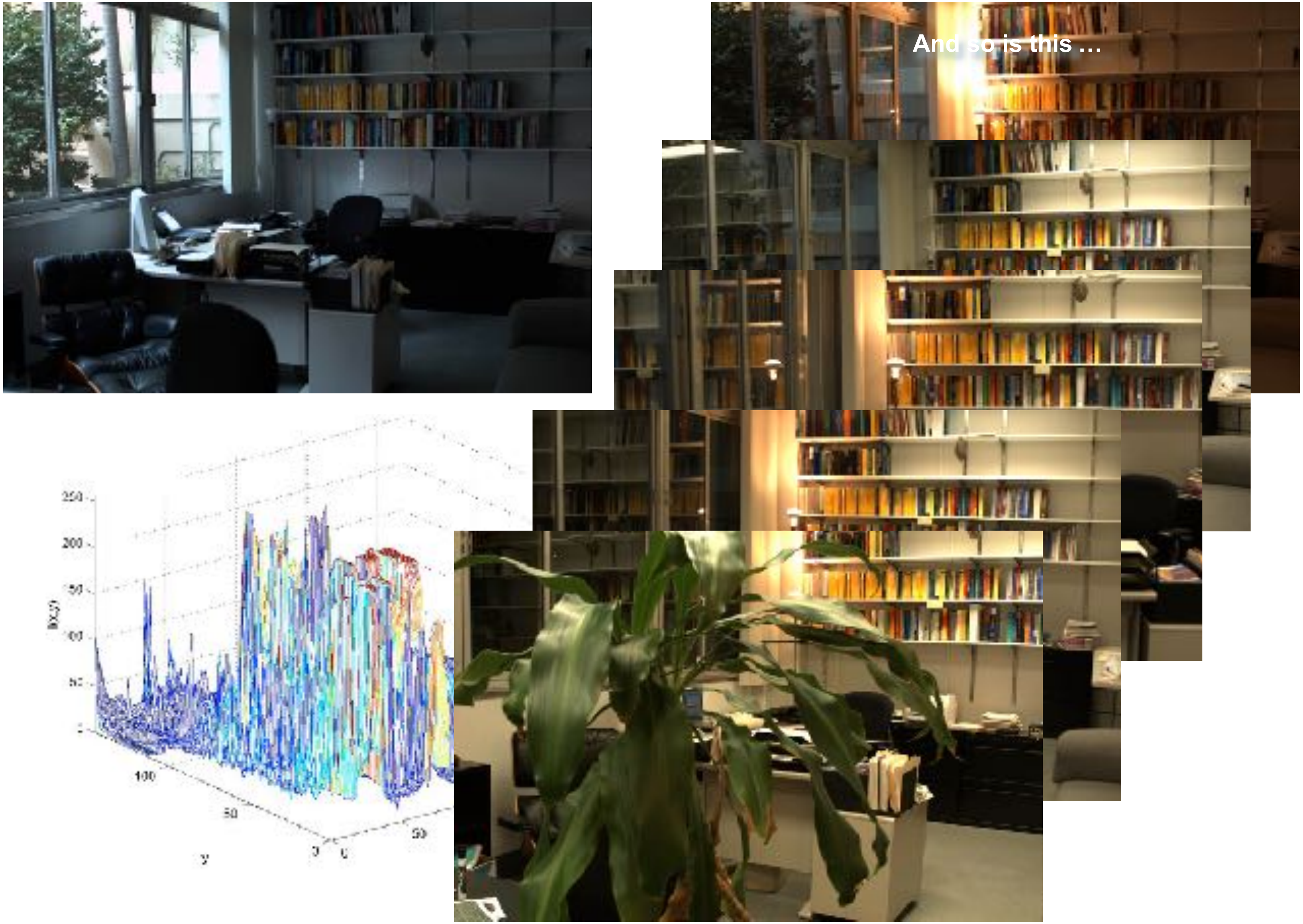
Standard base vectors:

$$\mathbf{i} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \qquad \mathbf{j} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

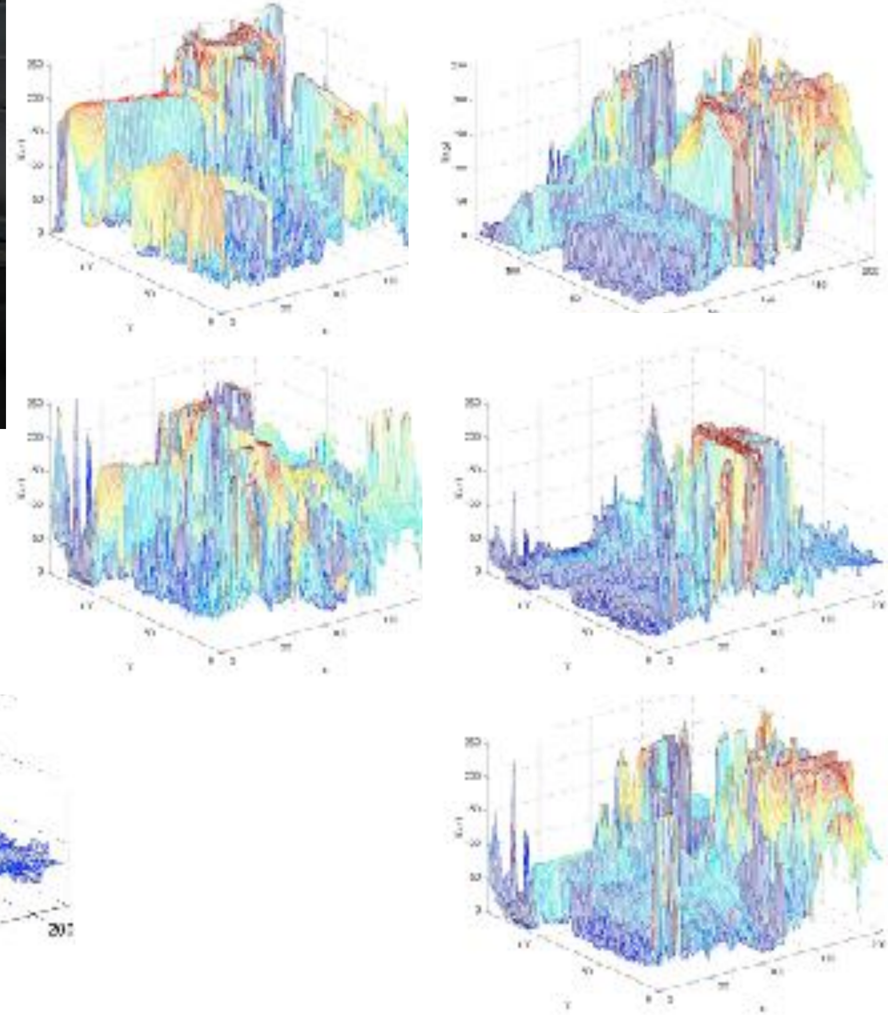


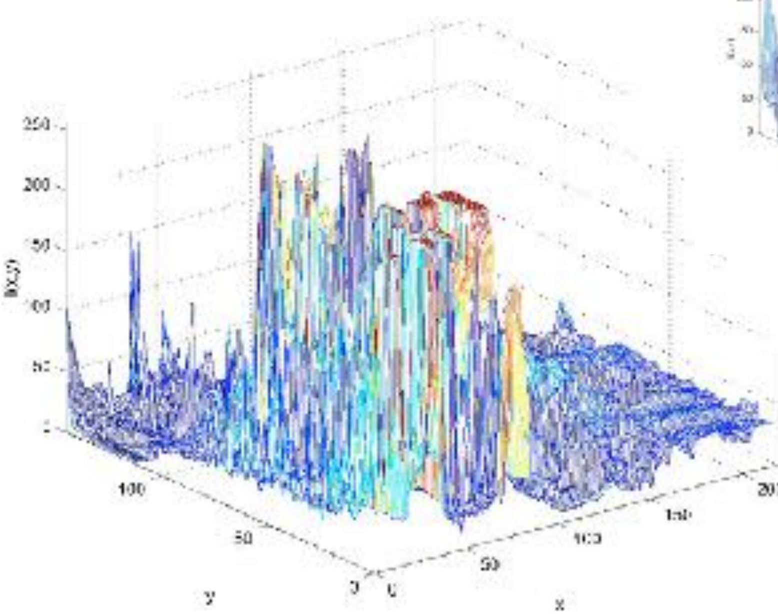
Coordinates of a point p in space:

$$\boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \in \mathbb{R}^3 \qquad \boldsymbol{X} = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = X.\mathbf{i} + Y.\mathbf{j} + Z.\mathbf{k}$$


And so is this …

And so are these!





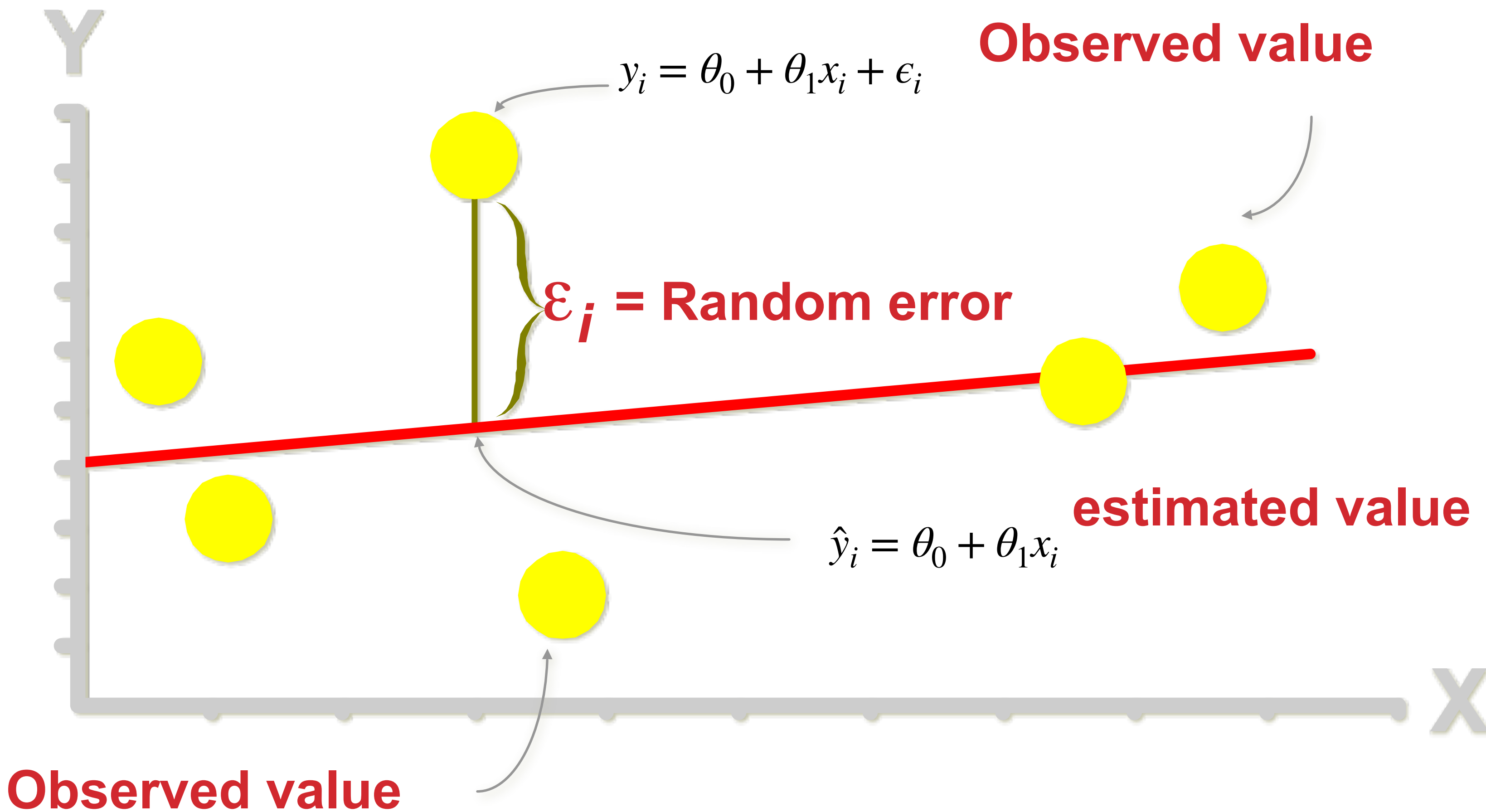
Example

- Single Variable Linear Regression

estimate $\hat{y}_i = \theta_0 + \theta_1 x_i$

x Area(sq. ft.)	y Price (in 1000$)
1600	220
1400	180
2100	350
…	…
….	….
2400	500

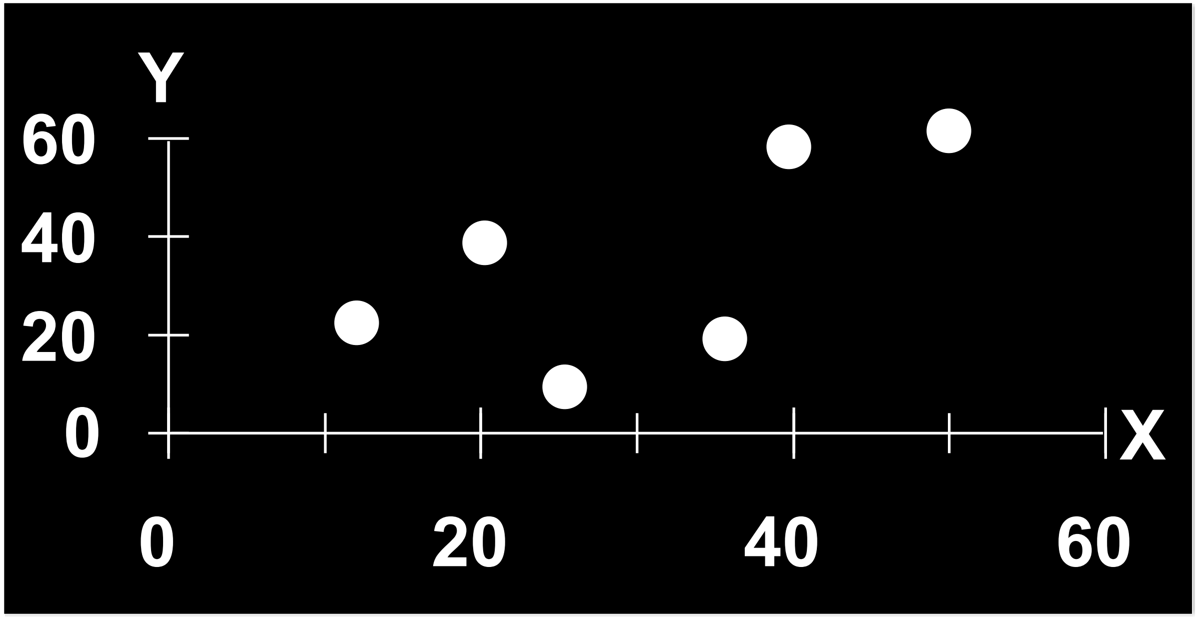
LINEAR REGRESSION



[WF]

SCATTER PLOT

Plot all (X_i, Y_i) pairs, and plot your learned model



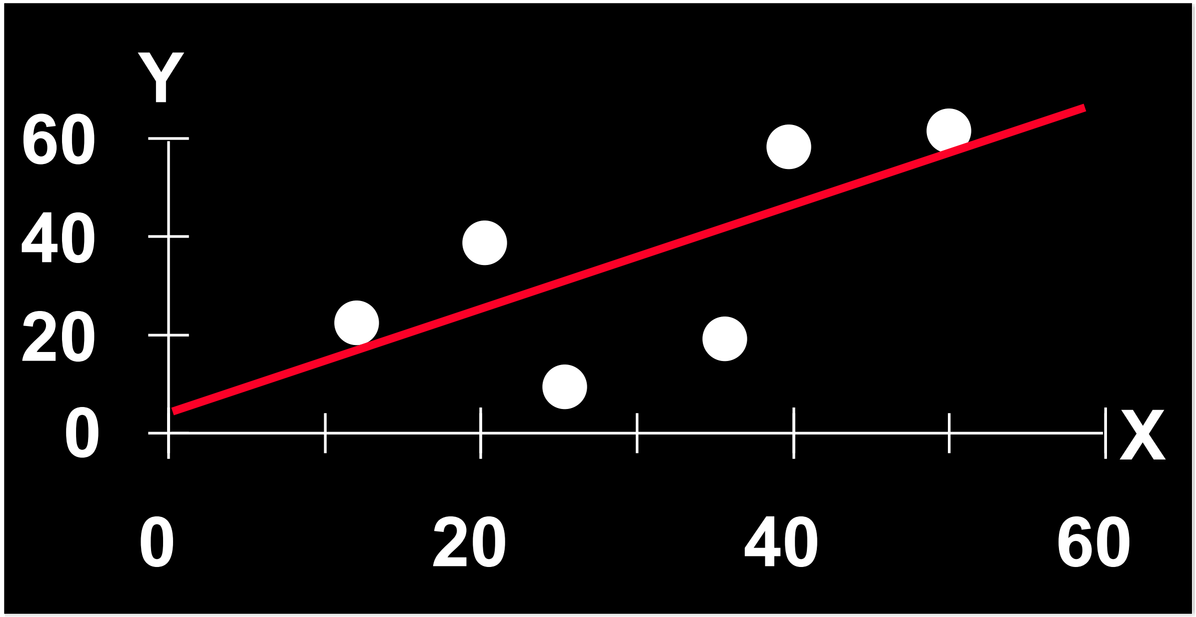
[WF]

QUESTION

How would you draw a line through the points?

How do you determine which line "fits the best" …?

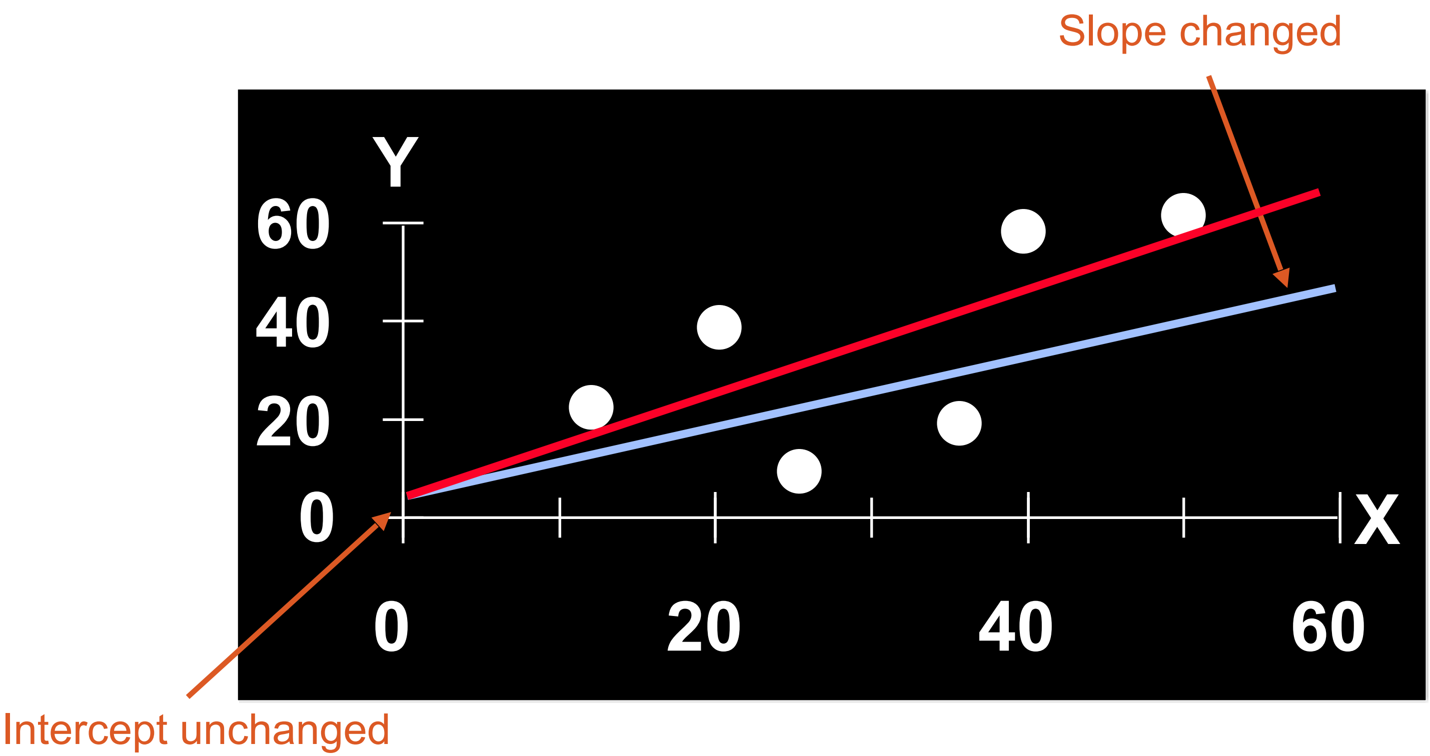
??????????



QUESTION

How would you draw a line through the points?

How do you determine which line “fits the best” ?????????

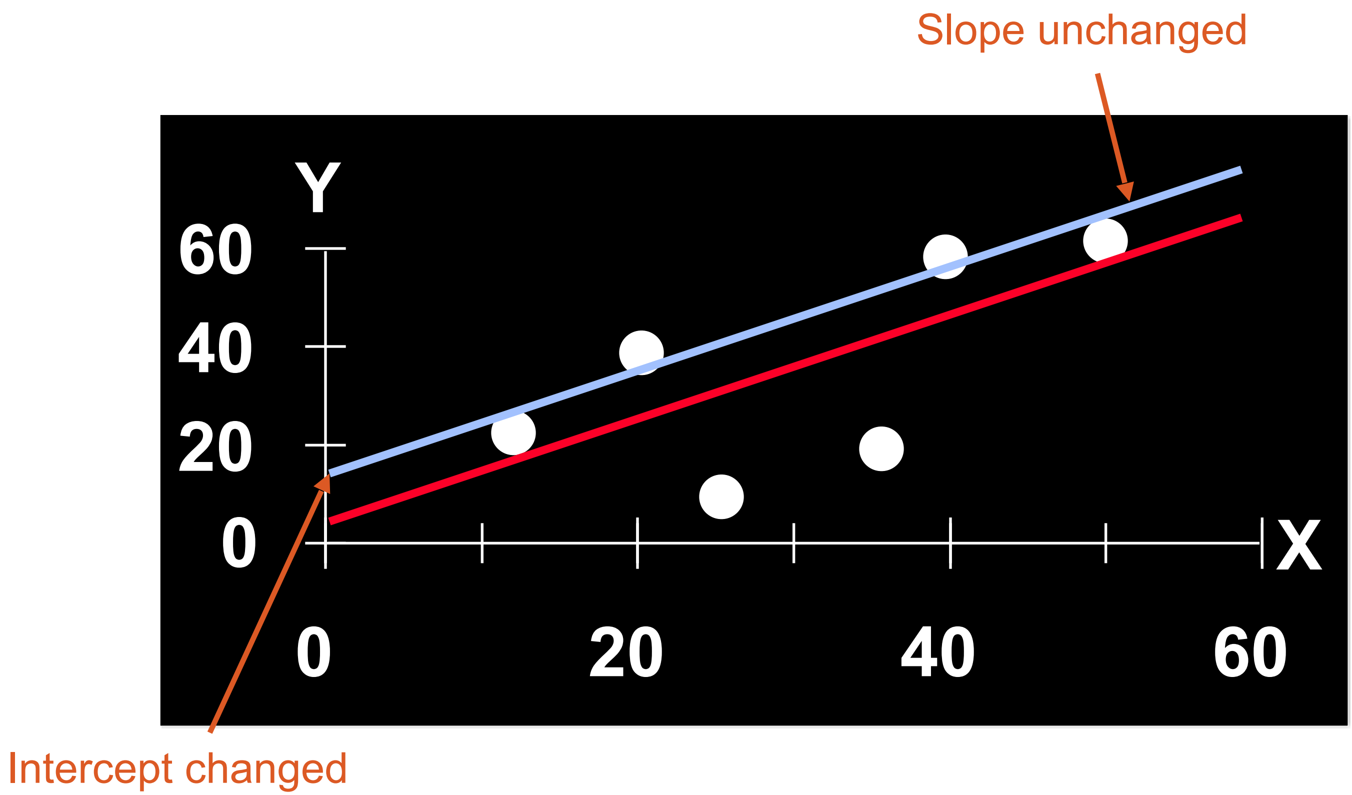


[WF]

QUESTION

How would you draw a line through the points?

How do you determine which line “fits the best” ?????????



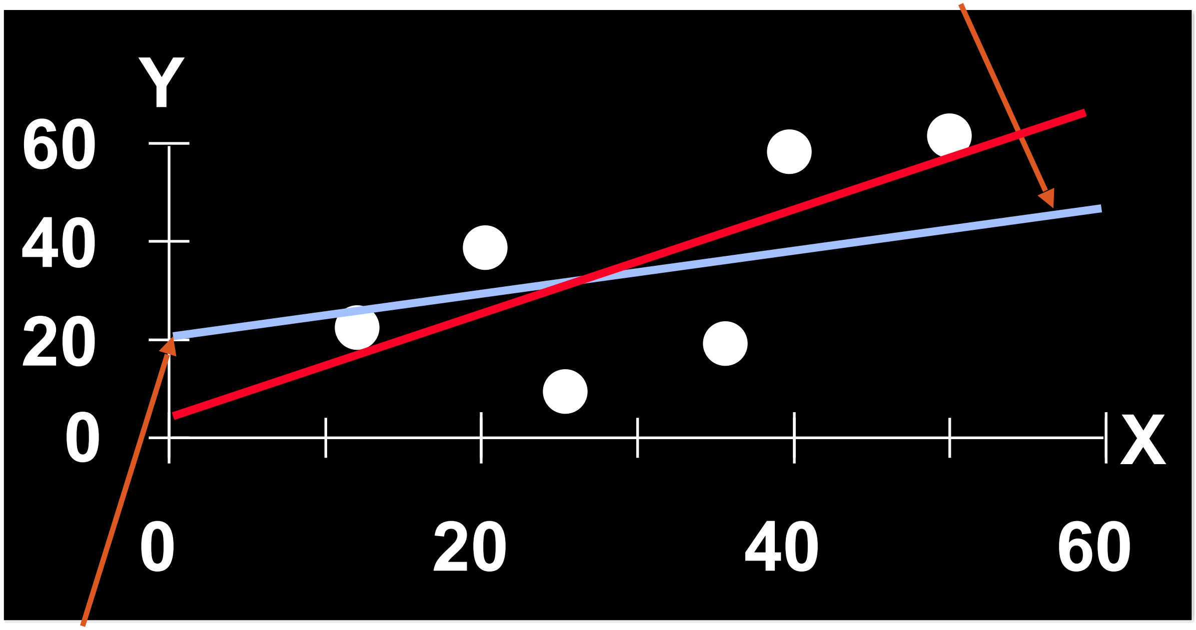
[WF]

QUESTION

How would you draw a line through the points?

How do you determine which line "fits the best" ?????????

Slope changed



Intercept changed

[WF]

LEAST SQUARES

Best fit: difference between the true (observed) Y-values and the estimated Y-values is minimized:

- Positive errors offset negative errors …
- … square the error!

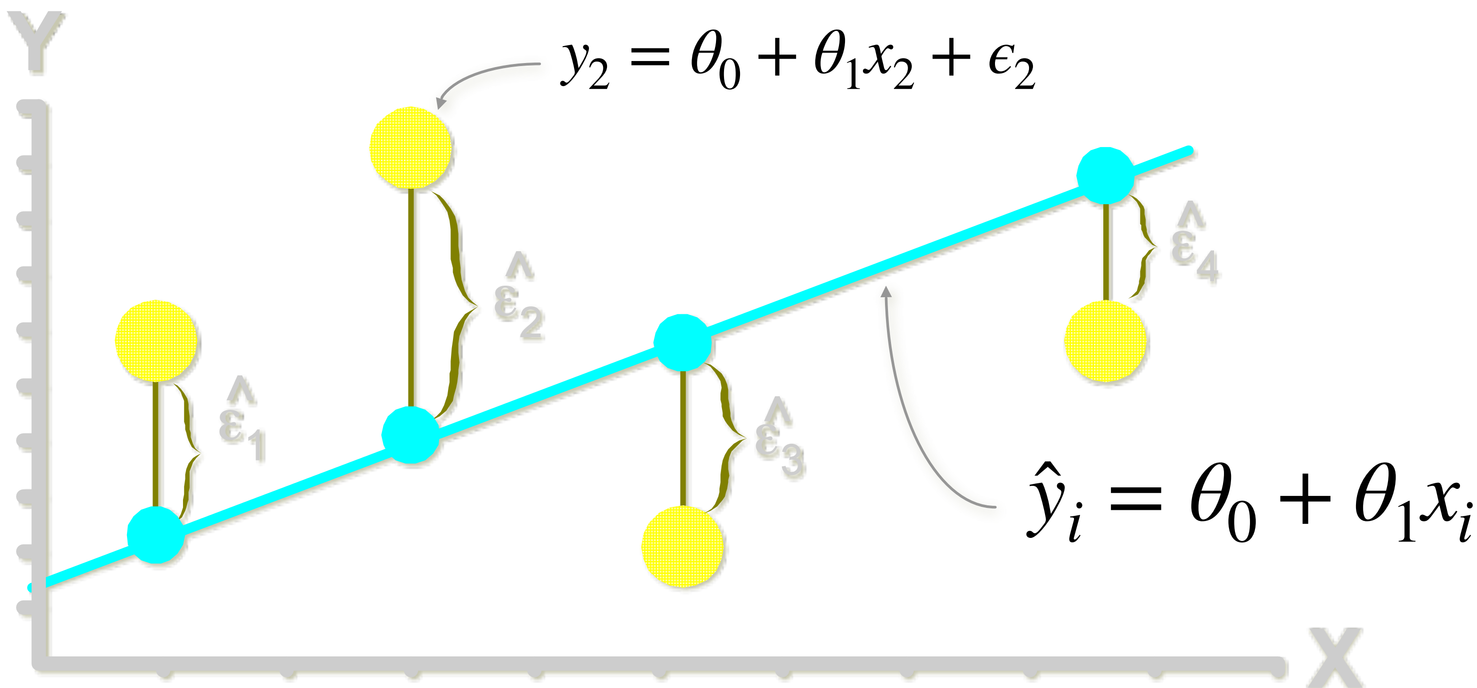
$$\sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \sum_{i=1}^{n} \epsilon_i^2$$

Least squares minimizes the sum of the squared errors

[WF]

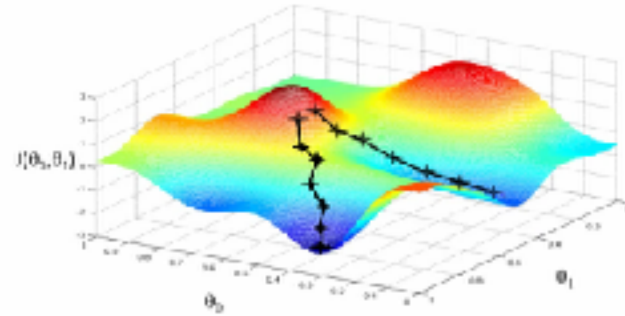
LEAST SQUARES, GRAPHICALLY

LS Minimizes $\sum_{i=1}^{n} \epsilon_i^2 = \epsilon_1^2 + \epsilon_2^2 + \ldots + \epsilon_n^2$



[WF]

Multivariate Regression



- Multi Linear Regression

$$y_i = \theta_0 x_{i0} + \theta_1 x_{i1} + \theta_2 x_{i2} + \ldots + \theta_n x_{in}$$

	y	x_0	x_1	x_2	x_3
	Price (in 1000$)		Area(sq. ft.)	# Bathrooms	# Bedrooms
	220	1	1600	2.5	3
y_i	180	1	1400	1.5	3
	350	1	2100	3.5	4
	…	…	…	…	…
	….	….	….	…	…
	500	1	2400	4	5

$$x_i = \begin{array}{cl} \begin{bmatrix} 1 \\ 1400 \\ 1.5 \\ 3 \end{bmatrix} & \begin{array}{l} x_{i0} \\ x_{i1} \\ x_{i2} \\ x_{i3} \end{array} \end{array}$$

All Observation Model

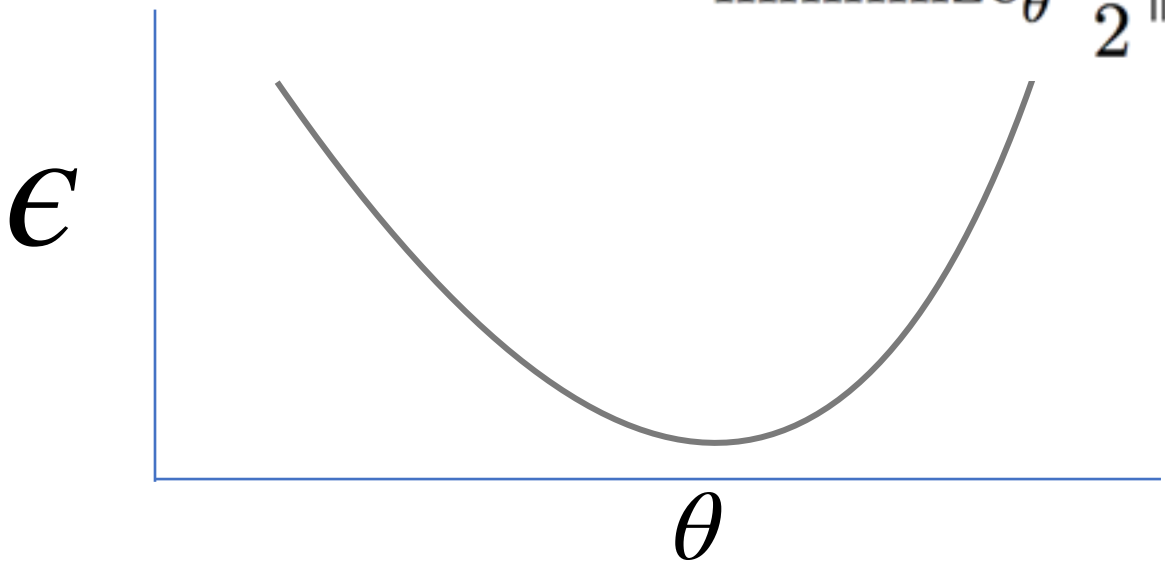
- Matrix Notation
  For all observations

$$
\begin{bmatrix}
x_{10} & x_{11} & x_{12} & .. & .. & x_{1m} \\
x_{20} & x_{21} & x_{22} & .. & .. & x_{2m} \\
x_{30} & x_{31} & x_{32} & .. & .. & x_{3m} \\
. & . & . & . & . & . \\
. & . & . & . & . & . \\
. & . & . & . & . & . \\
x_{n0} & x_{n1} & x_{n2} & .. & .. & x_{nm}
\end{bmatrix}
\begin{bmatrix}
\theta_0 \\ \theta_1 \\ \theta_2 \\ . \\ \theta_m
\end{bmatrix}
=
\begin{bmatrix}
y_1 \\ y_1 \\ y_2 \\ . \\ . \\ . \\ y_n
\end{bmatrix}
$$

$$\hat{Y} = X\theta$$

ERROR FUNCTION

$$\text{minimize}_\theta \ \frac{1}{2}\|X\theta - y\|_2^2$$



$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \sum_{i=1}^{n}\epsilon_i^2$$

LEAST SQUARES

Recall: points where the gradient equals zero are minima.

$$\nabla_\theta \frac{1}{2}\|X\theta - y\|_2^2 = X^T(X\theta - y)$$

So where do we go from here?????????

$$X^T(X\theta - y) = 0$$

Solve for model parameters θ

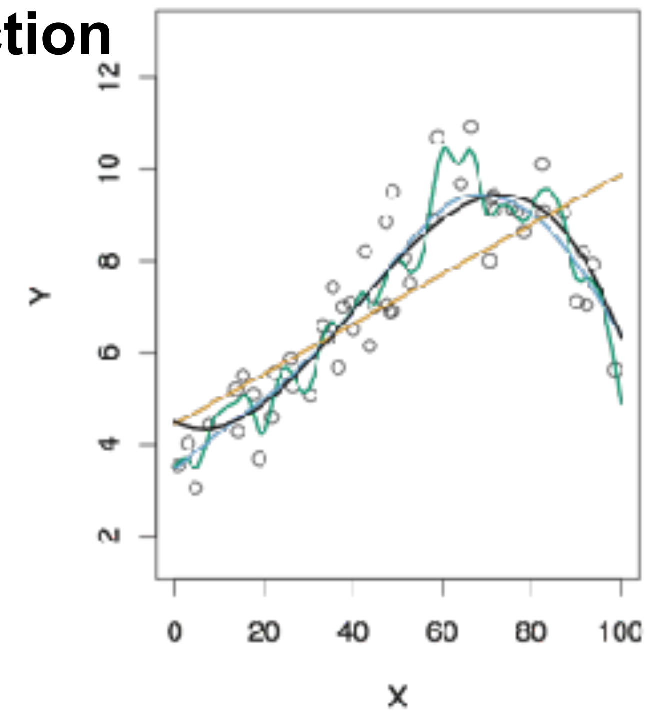
$$X^T X\theta - X^T y = 0 \Rightarrow X^T X\theta = X^T y$$

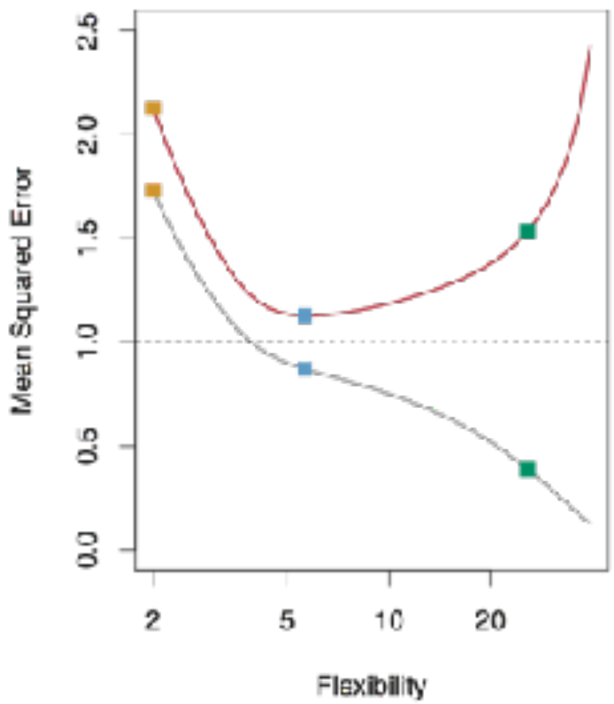
$$(X^T X)^{-1} X^T X\theta = (X^T X)^{-1} X^T y$$

$$\boxed{\theta = (X^T X)^{-1} X^T y}$$

OVERFITTING - SOLUTION

- **Model Selection**





- **Regularization (Ridge Regression)**

REGULARIZATION

- **For linear Regression, loss function**

$$L(\theta) = \frac{1}{2n}\sum_{i=1}^{n}\left(h_\theta(x_i) - y_i\right)^2$$

- **L2 norm or the sum of squares:**

$$\theta_1^2 + \theta_2^2 + \ldots + \theta_m^2 = \sum_{j=1}^{m}\theta_j^2 \approx |\theta|_2^2 \text{ – L2 norm}$$

- **Updated loss function:**

$$\min_{\theta_1,\theta_2,\ldots,\theta_m} \qquad L(\theta) = \frac{1}{2n}\left[\sum_{i=1}^{n}\left(h_\theta(x_i) - y_i\right)^2 + \lambda\sum_{j=1}^{m}\theta_j^2\right]$$

λ – tuning/regularization parameter

NORMAL EQUATION - CLOSED FORM SOLUTION

$$\min_{\theta_1,\theta_2,\ldots,\theta_m} \frac{1}{2n}\left[\left(X\theta - y\right)^T\left(X\theta - y\right) + \lambda\theta^T\theta\right]$$

$$\nabla_\theta L(\theta) = \frac{\partial}{\partial\theta}\left(\frac{1}{2n}\left[\left(X\theta - y\right)^T\left(X\theta - y\right) + \lambda\theta^T\theta\right]\right) = 0$$

Recall: For linear regression without regularization

$$\theta = (X^T X)^{-1} X^T y$$

For linear regression with regularization (ridge regression)

$$\theta = (X^T X + \lambda I)^{-1} X^T y$$

SPECTRAL THEOREM

Theorem: If $X \in \mathbb{R}^{m \times n}$ is symmetric matrix (meaning $X^T = X$),

then, there exist real numbers $\lambda_1, \ldots, \lambda_n$ (the eigenvalues)

and orthogonal, non-zero real vectors $\phi_1, \phi_2, \ldots, \phi_n$

(the eigenvectors) such that for each $i = 1, 2, \ldots, n$:

$$X\phi_i = \lambda_i \phi_i$$

SPECTRAL THEOREM

If $A \in \mathbb{R}^{m \times n}$ is symmetric matrix, then the , $m \times m$ matrix AA^T

and the $n \times n$ matrix $A^T A$ are both symmetric

We can apply Spectral theorem to the matrices AA^T and $A^T A$

Question: How are the eigenvalues and the eigenvectors of these matrices related?

SPECTRAL THEOREM

Using Spectral theorem

$$(A^T A)\phi = \lambda\phi$$

$$AA^T(A\phi) = \lambda(A\phi)$$

Conclusion:

The matrices AA^T and $A^T A$ share the same nonzero eigenvalues

To get an eigenvector of AA^T from $A^T A$ multiply ϕ on the left by A

Very powerful, particularly if number of observations, n, and the number of features, m, are drastically different in size.

For PCA: $Cov(A, A) = AA^T$

SINGULAR VALUE DECOMPOSITION

Theorem : $$A_{mn} = U_{mm} \Sigma_{mn} V_{nn}^T$$

A - Rectangular matrix, $m \times n$

Columns of U are orthonormal eigenvectors of AA^T

Columns of V are orthonormal eigenvectors of A^TA

Σ is a diagonal matrix containing the square roots of eigenvalues from U or V in descending order

SVD - EXAMPLE

$$A = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

$$A_{3\times 2} = U_{3\times 3}\Sigma_{3\times 2}V^T_{2\times 2}$$

$$A = \begin{bmatrix} \frac{\sqrt{6}}{3} & 0 & -\frac{1}{\sqrt{3}} \\ \frac{\sqrt{6}}{6} & -\frac{\sqrt{2}}{2} & \frac{1}{\sqrt{3}} \\ \frac{\sqrt{6}}{6} & \frac{\sqrt{2}}{2} & \frac{1}{\sqrt{3}} \end{bmatrix} \begin{bmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

K-MEANS - ITERATIVE ALGORITHM

1. Initialize by choosing K observations as centroids.

$$m_1, m_2, \ldots, m_k$$

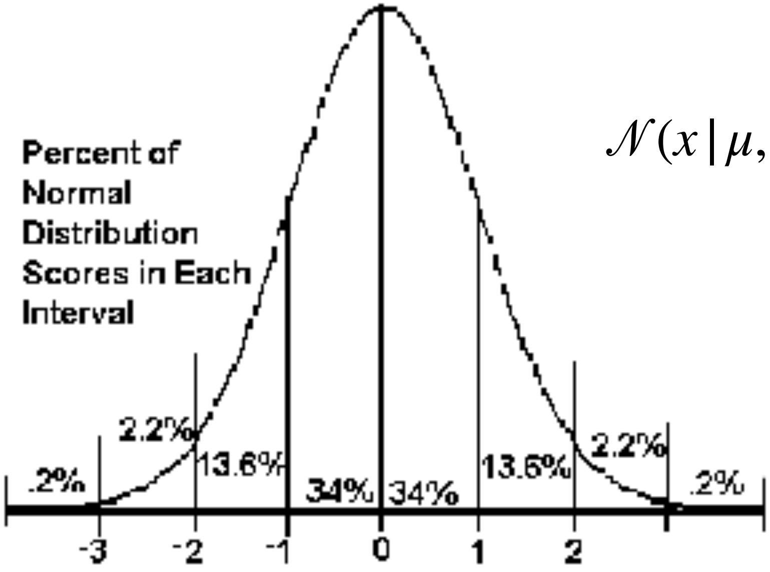
2. Assign each observation i to the cluster with the nearest centroid, i.e,

$$\min_{1 \leq k \leq K} ||x_i - m_k||^2$$

3. Update centroids $m_k = \bar{x}_k$
4. Iterate steps 2 and 3 until convergence.

Notation: Normal distribution 1D case

N(μ , σ) is a 1D normal (Gaussian) distribution with mean μ and standard deviation σ (so the variance is σ^2.



$$\mathcal{N}(x\,|\,\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}} exp\left\{ -\frac{1}{2\sigma^2}(x-\mu)^2 \right\}$$

Multivariate Normal distribution

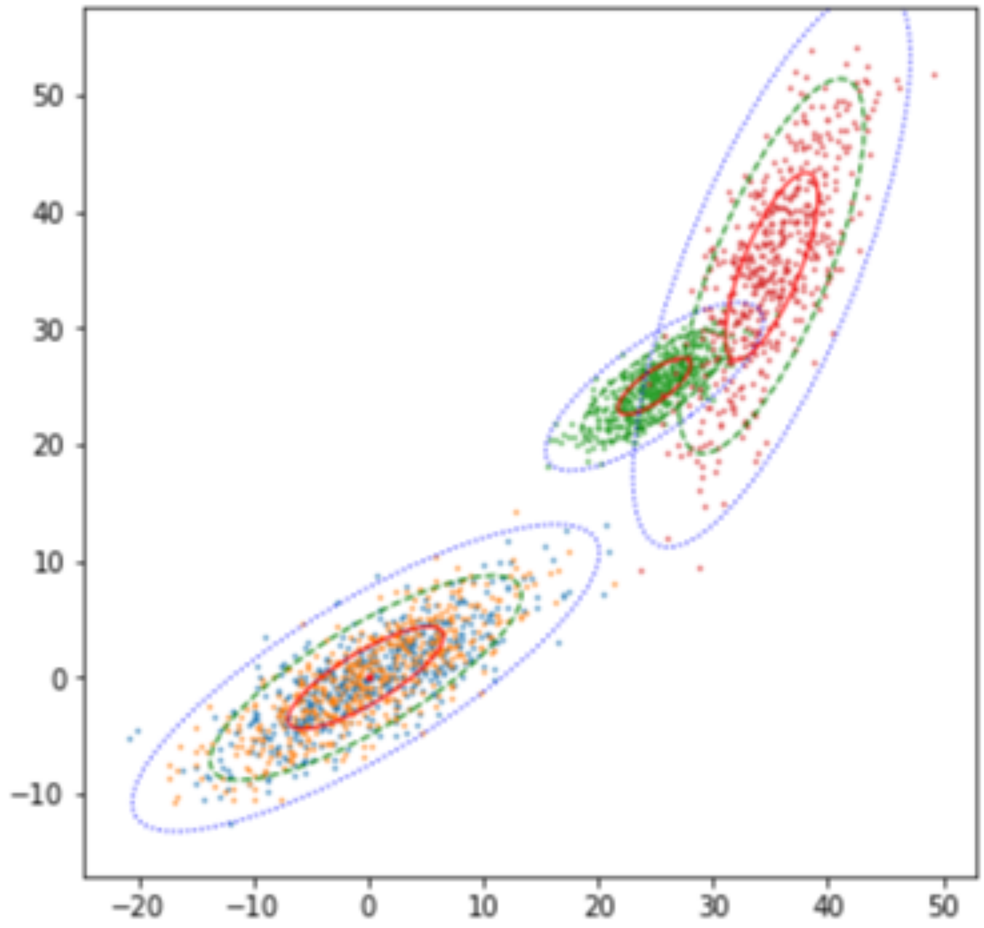
$$\mathcal{N}(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} exp\left\{ -\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) \right\}$$

x is a D dimensional vector

μ is a D-dimensinal mean vector

Σ is a D x D covariance matrix

Multi-modal dataset



Gaussian Mixtures Model

Parameters - μ, Σ, π

$$\sum_{k=1}^{K} \pi_k = 1 \qquad ; \qquad 0 \leq \pi_k \leq 1$$

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x \,|\, \mu_k, \Sigma_k)$$

$$\mathcal{N}(x \,|\, \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}} \frac{1}{|\Sigma|^{\frac{1}{2}}} exp\left\{ -\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu) \right\}$$

x is a D dimensional vector

μ is a D-dimensinal mean vector

Σ is a D x D covariance matrix

Maximum Likelihood Estimate

$$\mathcal{N}(x\,|\,\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{D}{2}}} \frac{1}{|\,\Sigma\,|^{\frac{1}{2}}} exp\left\{ -\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) \right\}$$

$$ln\mathcal{N}(x\,|\,\mu, \Sigma) = -\frac{D}{2} ln\, 2\pi - \frac{1}{2} ln\, \Sigma - \frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)$$

Once Optimal values of the parameters are found,

the solution will correspond to the Maximum Likelihood Estimate (MLE)

Expectation Maximization

For lack of a closed form solution

$$ln\, p(X\,|\,\pi, \mu, \Sigma) = \sum_{i=1}^{n} ln \sum_{k=1}^{K} \pi_k \mathcal{N}(x_i\,|\,\mu_k, \Sigma_k)$$

We will use an iterative technique

$Step1$ – Choose some initial values for the means, covariances and mixing coefficients, evaluate log likelihood

$Step2$

E-step: Use current values for the parameters to evaluate the posterior probabilities

$$\gamma(z_{ik}) = p(z_k = 1\,|\,x_i) = \frac{p(z_k = 1)p(x_i\,|\,z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(x_i\,|\,z_j = 1)}$$

$$= \frac{\pi_k \mathcal{N}(x_i\,|\,\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(x_i\,|\,\mu_j, \Sigma_j)}$$

Expectation Maximization

$Step3$

M-step: re-estimate means, covariances and mixing coefficients

$$\mu_k^{new} = \frac{1}{N_k} \sum_{i=1}^{N} \gamma(z_{ik}) x_i$$

$$\Sigma_k^{new} = \frac{1}{N_k} \sum_{i=1}^{N} \gamma(z_{ik})(x_i - \mu_k^{new})(x_i - \mu_k^{new})^T$$

$$\pi_k^{new} = \frac{N_k}{N}$$

where $N_k = \sum_{i=1}^{N} \gamma(z_{ik})$

$Step4$

– Evaluate the log likelihood

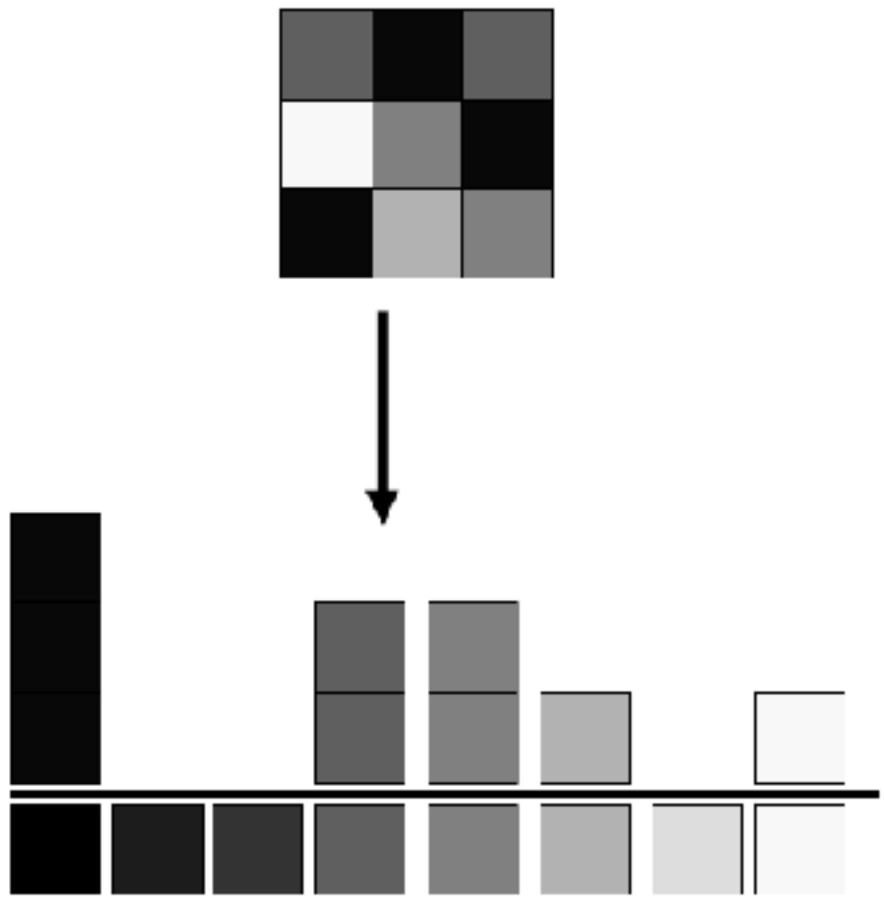
$$ln\, p(X \mid \pi, \mu, \Sigma) = \sum_{i=1}^{n} ln \sum_{k=1}^{K} \pi_k \mathcal{N}(x_i \mid \mu_k, \Sigma_k)$$

What is a digital image?

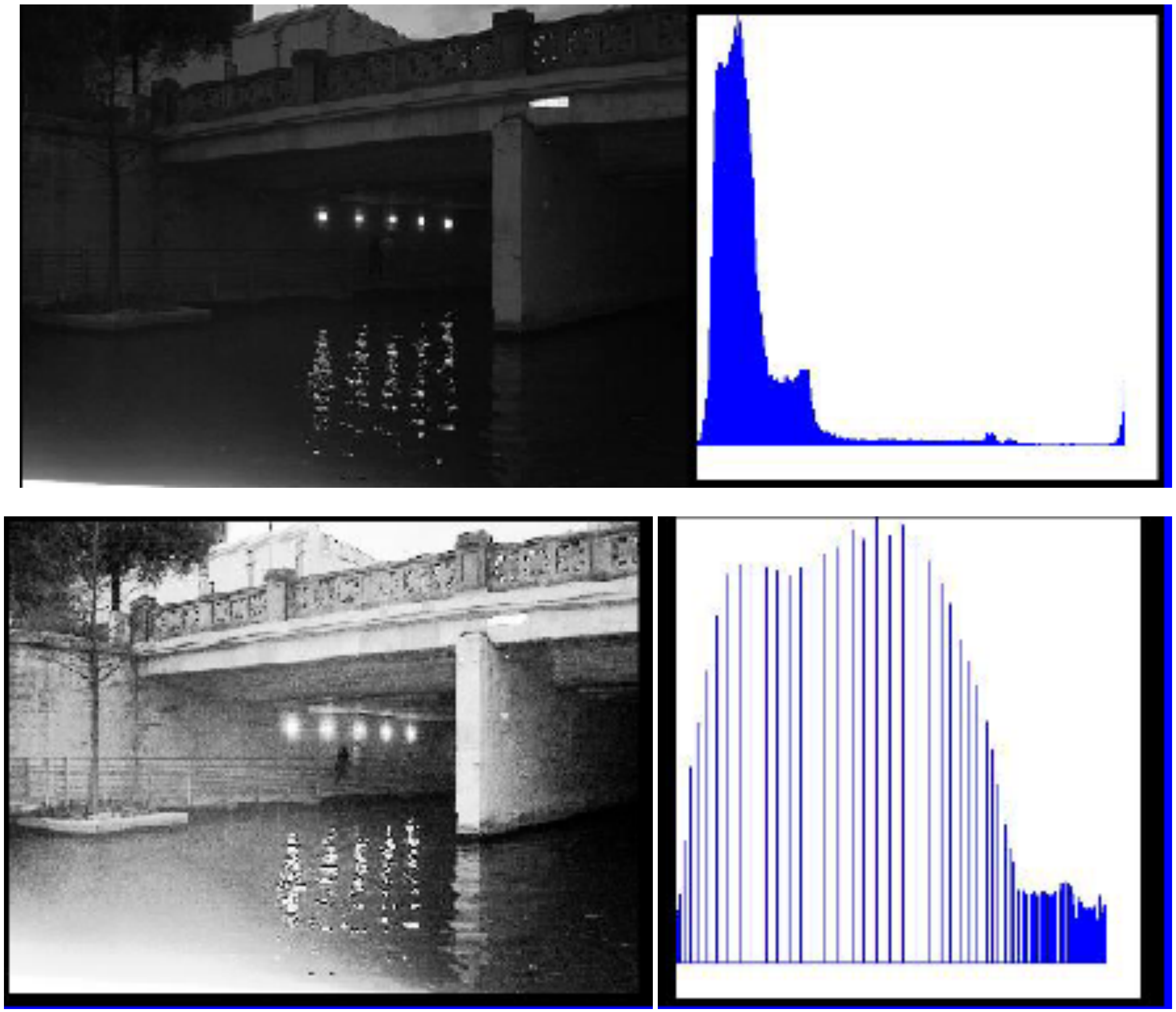
- In computer vision we usually operate on **digital** (**discrete**) images:
  - **Sample** the 2D space on a regular grid
  - **Quantize** each sample (round to nearest integer)
- If our samples are Δ apart, we can write this as:
- $f[i\ ,j] = \text{Quantize}\{\ f(i\ \Delta,\ j\ \Delta)\ \}$
- The image can now be represented as a matrix of integer values

	$j \rightarrow$							
	62	79	23	119	120	105	4	0
$i \downarrow$	10	10	9	62	12	78	34	0
	10	58	197	46	46	0	0	48
	176	135	5	188	191	68	0	49
	2	1	1	29	26	37	0	77
	0	89	144	147	187	102	62	208
	255	252	0	166	123	62	0	31
	166	63	127	17	1	0	99	30

A simple image and its histogram



Examples of histogram equalization



8 x 8 Image

Value	Count	Value	Count	Value	Count
52	1	64	2	72	1
55	3	65	3	73	2
58	2	66	2	75	1
59	3	67	1	76	1
60	1	68	5	77	1
61	4	69	3	78	1
62	1	70	4	79	2
63	2	71	2	83	1

v, Pixel Intensity	cdf(v)	h(v), Equalized v
52	1	0
55	4	12
58	6	20
59	9	32
60	10	36
61	14	53
62	15	57
63	17	65
64	19	73
65	22	85
66	24	93

$$h(v) = round\left(\frac{cdf(v) - cdf_{min}}{(M \times N) - cdf_{min}} \times (L-1)\right)$$

M – width $\quad$ M – height $\quad$ L – Number of gray levels

Mean filtering (average over a neighborhood)

$F[x, y]$

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

$G[x, y]$

	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	

Correlation & Convolution

- Basic operation to extract information from an image.

- These operations have two key features:

    - shift invariant
    - linear

- Applicable to 1-D and multi dimensional images.

Correlation Example - 1D

I	.	.	.	.	.	.	.	.	.	2	3	6	5	5	1	8	9	7	.	.	.	.	.	.	.	.	.

$*$ $*$ $*$

$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{3}$

||

$\frac{9}{3}$	$\frac{7}{3}$	$\frac{0}{3}$

Σ

G	.	.	.	.	.	.	.	.	.	$\frac{5}{3}$	$\frac{11}{3}$	$\frac{14}{3}$	$\frac{16}{3}$	$\frac{11}{3}$	$\frac{14}{3}$	6	8	$\frac{16}{3}$	.	.	.	.	.	.	.	.	.

Cross-Correlation and Convolution

$$2 \times 15 + 1 \times 1 = 31$$

$$\begin{bmatrix} 5 & 15 & 4 & 0 & -1 \\ 10 & 1 & 5 & 1 & 0 \\ 6 & 9 & 11 & 1 & -1 \\ 0 & -1 & 5 & 15 & 4 \\ 1 & 0 & 10 & 1 & 5 \end{bmatrix} \circ \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 31 & & & & \\ & & & & \\ & & & & \\ & & & & \\ & & & & \end{bmatrix}$$

Cross-Correlation and Convolution

5	15	4	0	-1
10	1	5	1	0
6	9	11	1	-1
0	-1	5	15	4
1	0	10	1	5

$\circ$

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

=

31	-7	-30	-15	-1
26	-6	-23	-27	-3
18	10	0	-30	-18
7	24	25	-19	-32
-1	23	16	-11	-17

Image, I   Filter/template   Output image

Cross-Correlation - Mathematically

$1D$

$$G = F \circ I[i] = \sum_{u=-k}^{k} F[u]I[i+u]$$ F has $2k+1$ elements

Box filter $F[u] = \frac{1}{3}$ for $u = -1,0,1$ and 0 otherwise

Cross-correlation filtering - 2D

Let's write this down as an equation. Assume the averaging window is (2k+1)x(2k+1):

$$G[i,j] = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[i+u, j+v]$$

We can generalize this idea by allowing different weights for different neighboring pixels:

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i+u, j+v]$$

This is called a **cross-correlation** operation and written:

$$G = F \circ I$$

F is called the "filter," "kernel," or "mask."

Convolution

Filter is flipped before correlating

$1D$ $\qquad$ F has $2k+1$ elements

$$G = F * I[i] = \sum_{u=-k}^{k} F[u]I[i-u]$$

Box filter $F[u] = \frac{1}{3}$ for $u = -1,0,1$ and 0 otherwise

for example, convolution of 1D image with the filter [3,5,2]

is exactly the same as correlation with the filter [2,5,3]

Convolution filtering - 2D

For 2D the filter is flipped and rotated

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F[u,v] I[i-u, j-v]$$

Correlation and convolution are identical for symmetrical filters

Convolution with the filter

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

is the same as Correlation with the filter

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

Gaussian Filtering

A Gaussian kernel gives less weight to pixels further from the center of the window

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

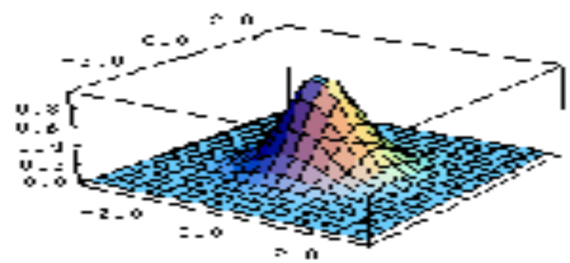
$F[x, y]$

$\frac{1}{16}$

1	2	1
2	4	2
1	2	1

$H[u, v]$

This kernel is an approximation of



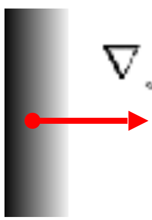
$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$

Image gradient

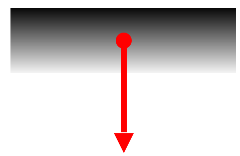
The gradient of an image:

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

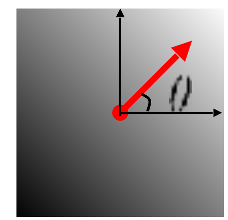
The gradient points in the direction of most rapid change in intensity



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$



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

The gradient direction is given by:

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

- How does this relate to the direction of the edge?

The *edge strength* is given by the gradient magnitude

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

The discrete gradient

How can we differentiate a *digital* image f[x,y]?

- Option 1: reconstruct a continuous image, then take gradient
- Option 2: take discrete derivative (finite difference)

$$\frac{\partial f}{\partial x}(x,y) = \frac{f(x+1,y) - f(x-1,y)}{2}$$

How would you implement this as a cross-correlation?

H

The Sobel operator

Better approximations of the derivatives exist

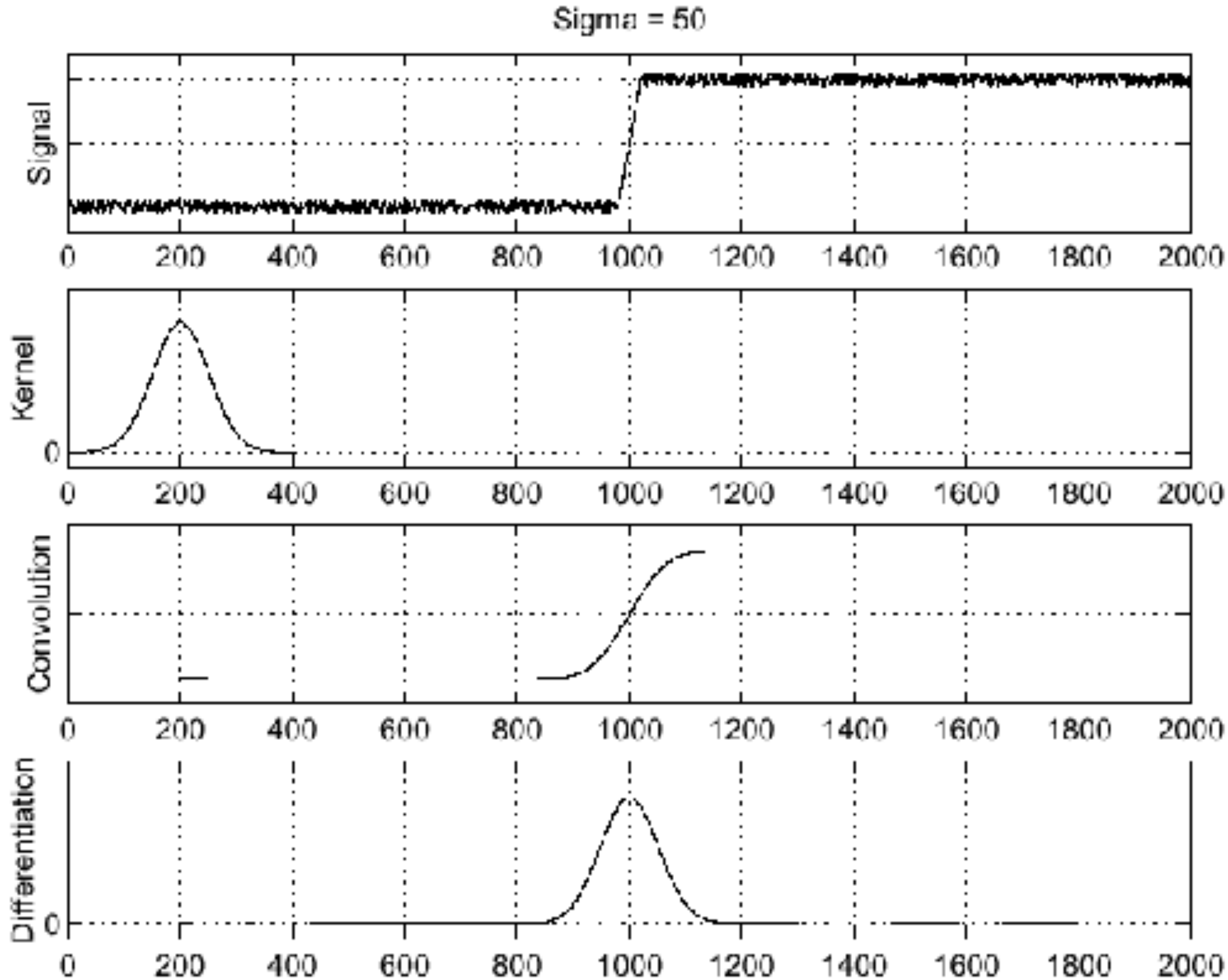
- The *Sobel* operators below are very commonly used

$$\frac{1}{8}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \frac{1}{8}\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -2 & -1 \\ \hline \end{array}$$

$$s_x \qquad\qquad\qquad s_y$$

- The standard defn. of the Sobel operator omits the 1/8 term
  - doesn't make a difference for edge detection
  - the 1/8 term **is** needed to get the right gradient value, however

Solution: smooth first



f

h

$h * f$

$\frac{\partial}{\partial x}(h * f)$

Where is the edge? Look for peaks $\frac{\partial}{\partial x}(h * f)$

Derivative theorem of convolution

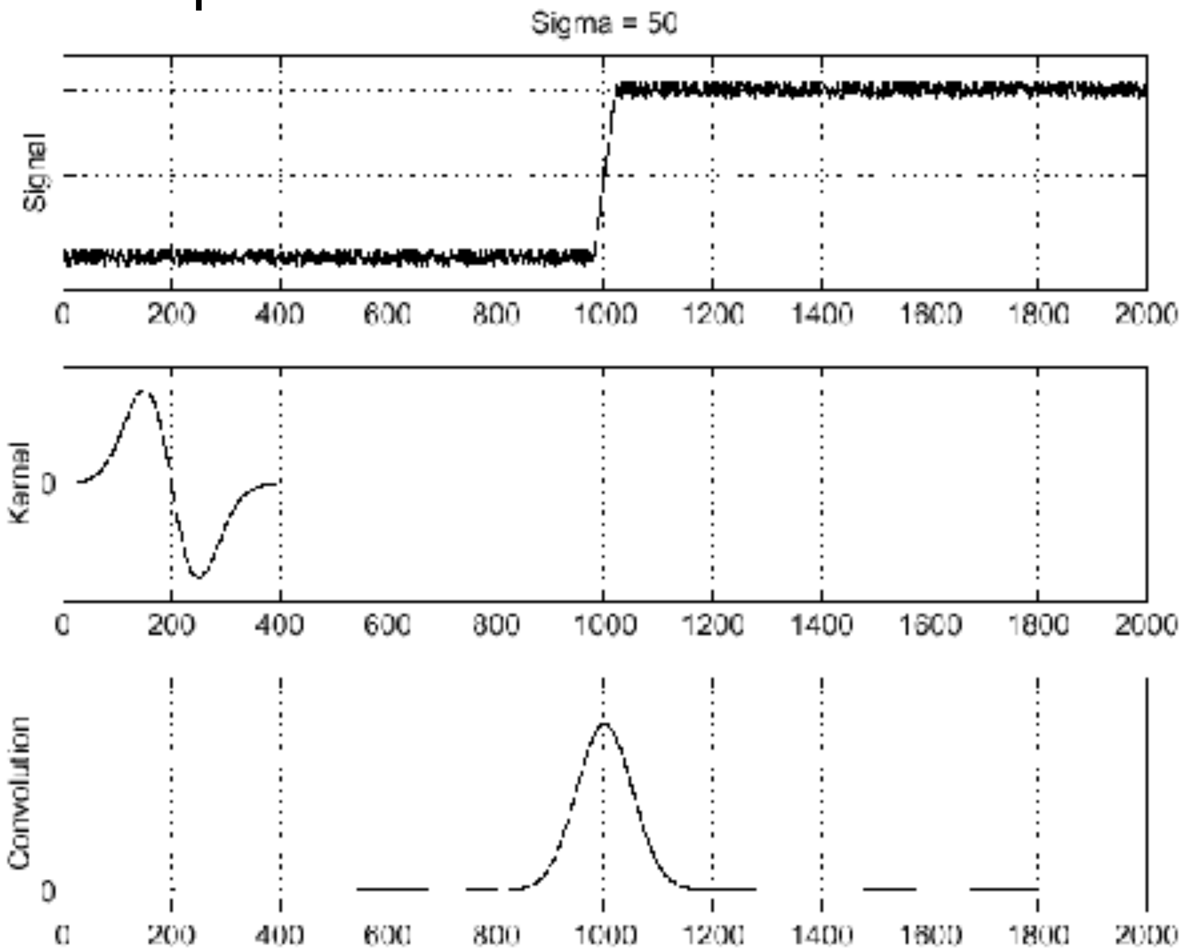
$$\frac{\partial}{\partial x}(h * f) = (\frac{\partial}{\partial x}h) * f$$

This saves us one operation:

$$f$$

$$\frac{\partial}{\partial x}h$$

$$(\frac{\partial}{\partial x}h) * f$$



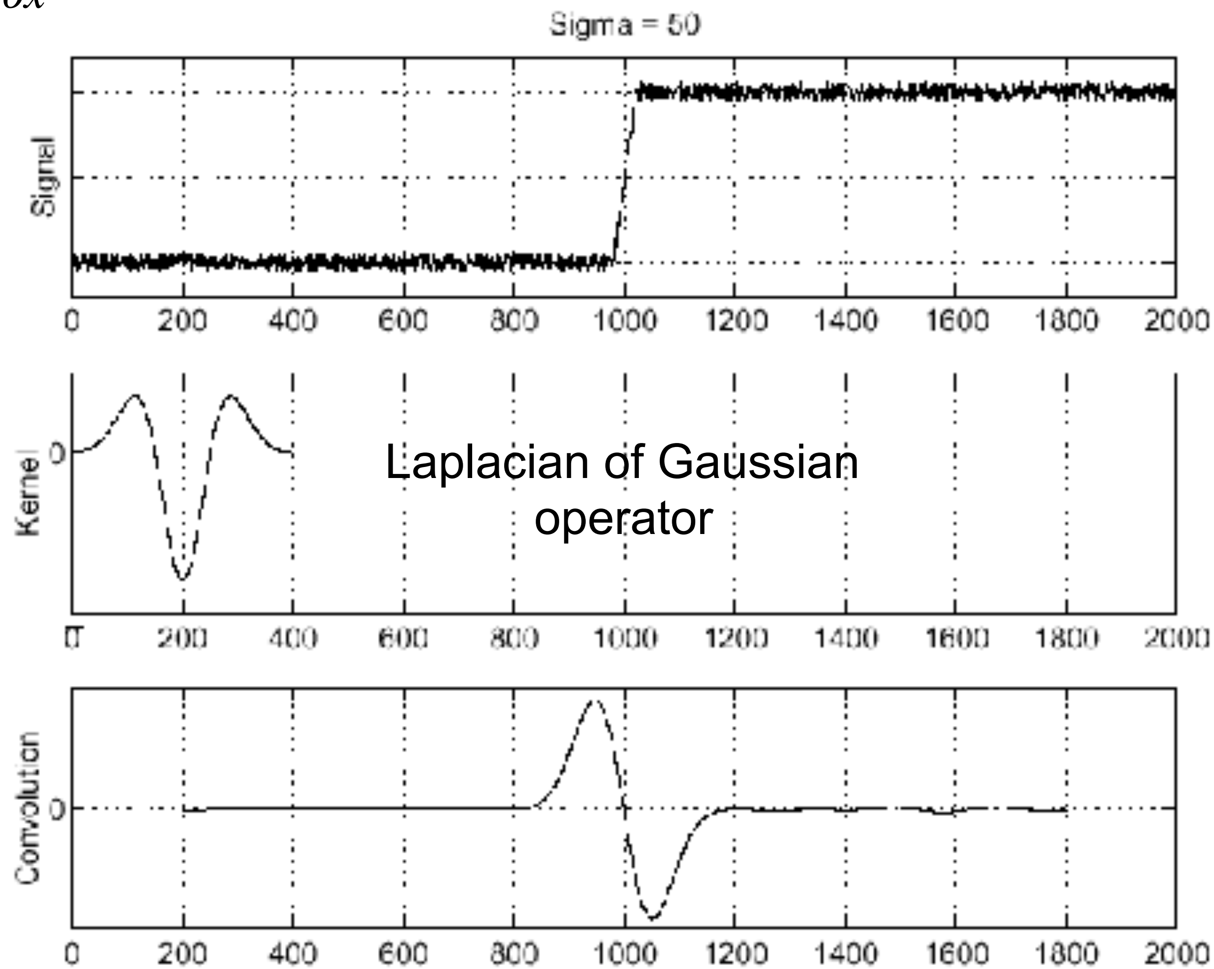
Laplacian of Gaussian

Consider $\frac{\partial^2}{\partial x^2}(h * f)$

f

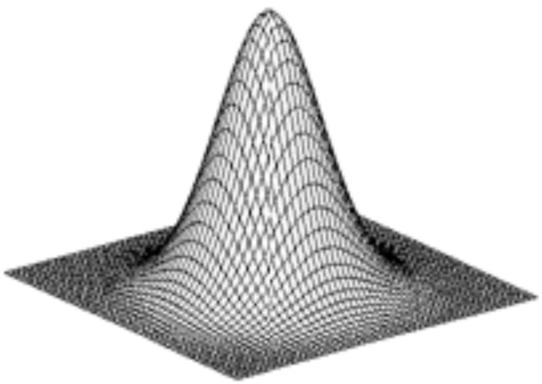
$\frac{\partial^2}{\partial x^2} h$

$(\frac{\partial^2}{\partial x^2} h) * f$



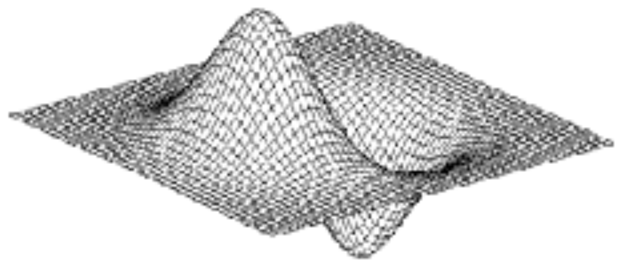
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



Gaussian

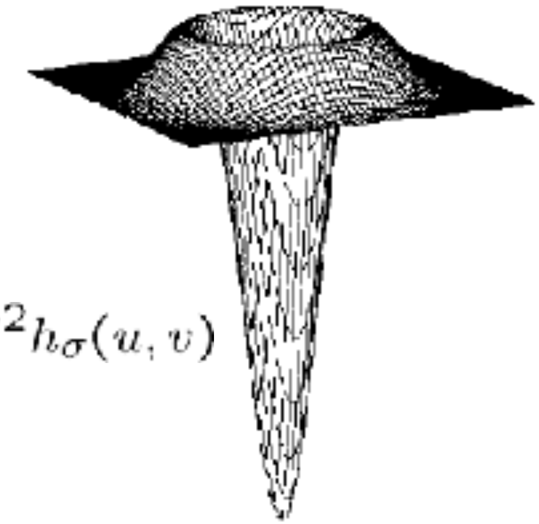
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



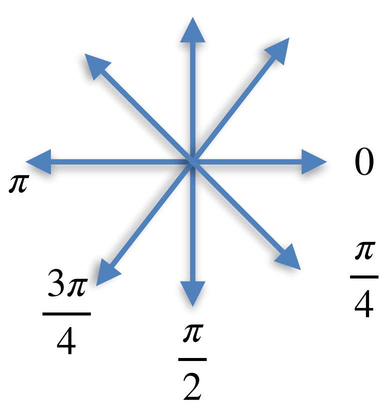
$$\nabla^2 h_\sigma(u,v)$$

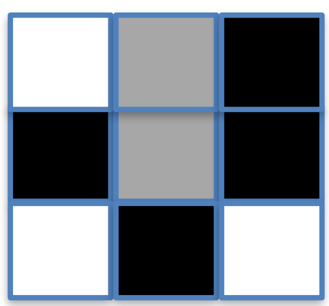
∇^2 is the **Laplacian** operator:

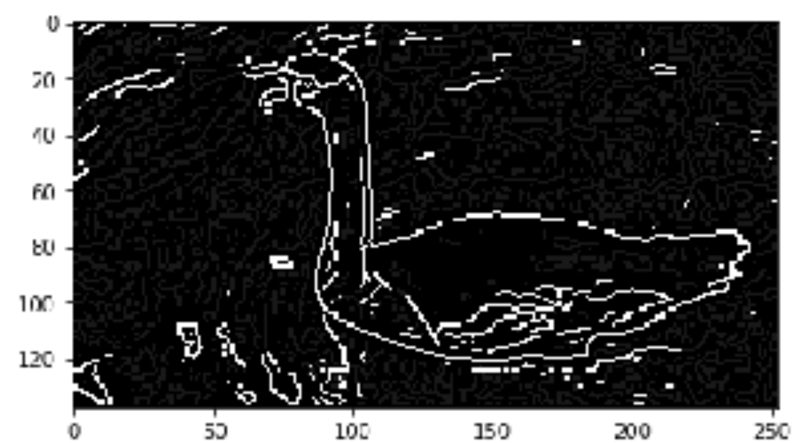
$$\nabla^2 f = \frac{\partial^2 f}{\partial u^2} + \frac{\partial^2 f}{\partial v^2}$$

Canny edge detector - Hysteresis

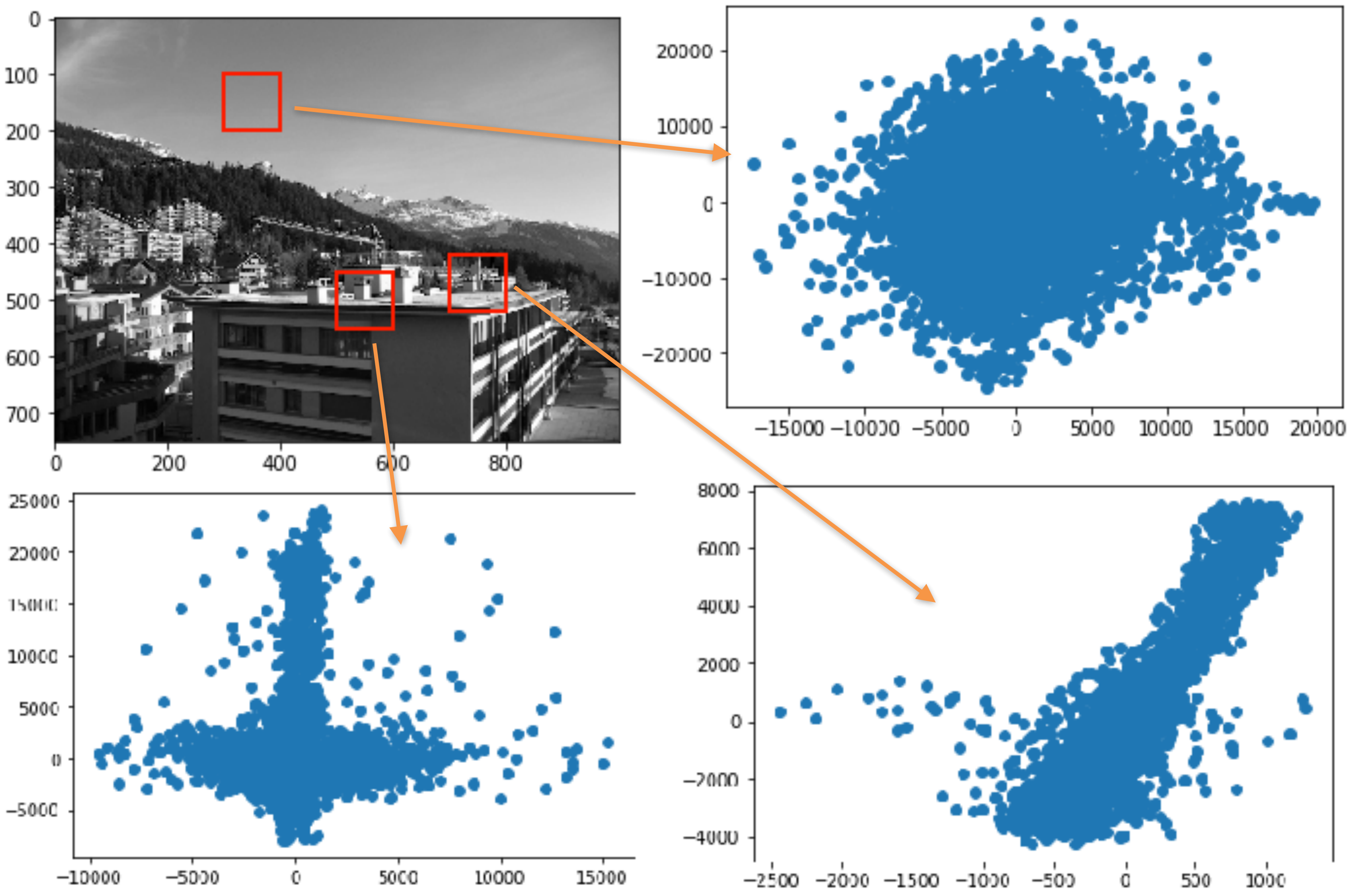
1. Smoothing (noise reduction)
2. Find derivatives (gradients)
3. Find magnitude and orientation of gradient
4. Non-maximum suppression:
   - Thin multi-pixel wide “ridges” down to single pixel width
5. Linking and thresholding (hysteresis):
   - Define two thresholds: low and high
   - replace with the strong edge if any of the neighboring pixels is strong, else make it irrelevant.



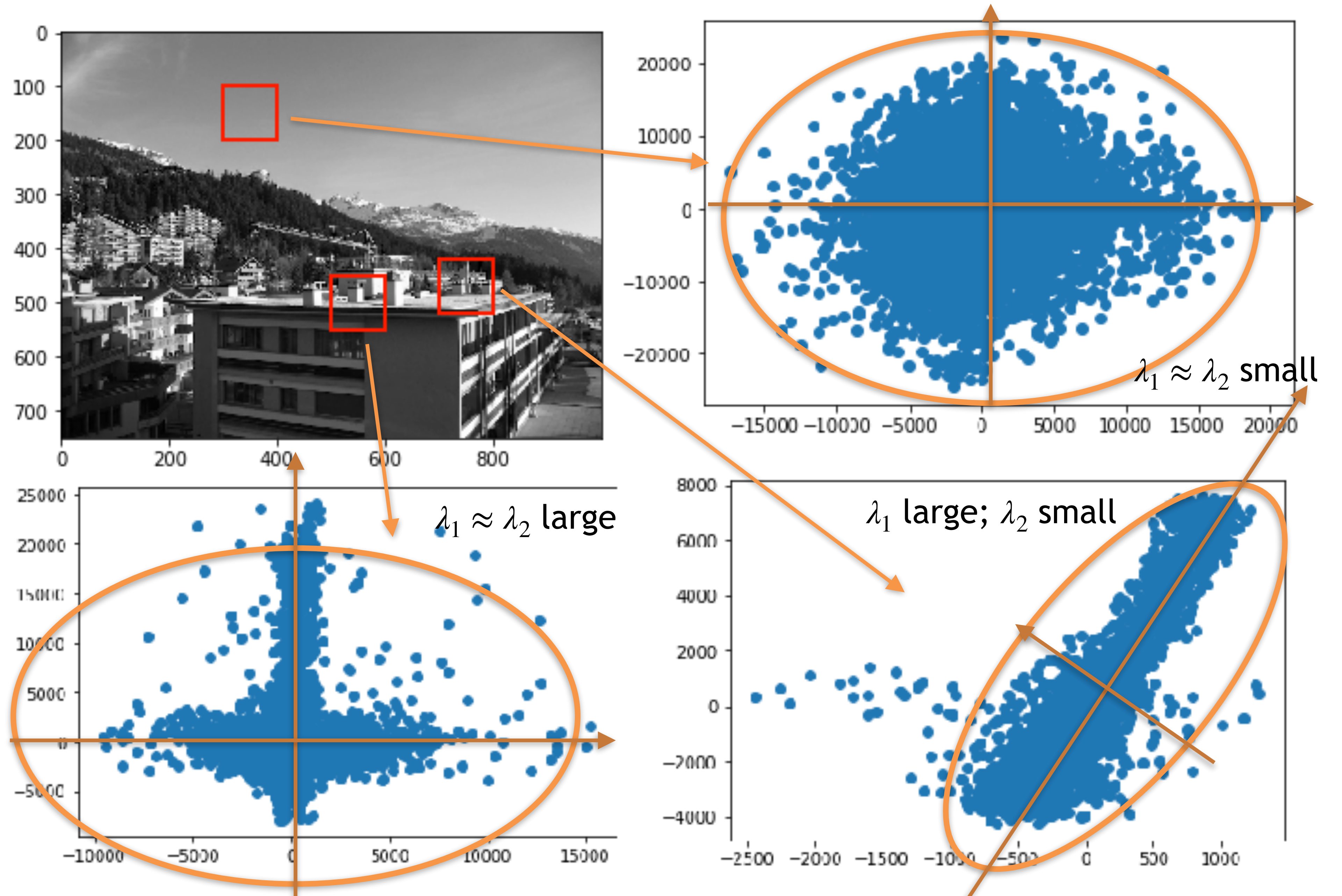




Background - Harris corner Detection



Gradient plots - I_x vs. I_y



Harris corner detector algorithm

-Compute magnitude of the gradient everywhere in x and y directions I_x, I_y

- Compute $I_x^2, I_y^2, I_x I_y$

- Convolve these three images with a Gaussian window, w. Find M for each pixel,

$$M = \sum w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

- Compute detector response, R at each pixel.

Score, $R = det(M) - k(trace(M))^2$

$$det(M) = \lambda_1 \lambda_2$$

$$trace(M) = \lambda_1 + \lambda_2$$

Harris corner detector algorithm

-Compute magnitude of the gradient everywhere in x and y directions I_x, I_y

- Compute $I_x^2, I_y^2, I_x I_y$

- Convolve these three images with a Gaussian window, w. Find M for each pixel,

$$M = \sum w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

- Compute detector response, R at each pixel.

$$R = det(M) - k(trace(M))^2$$

– find local maxima above some threshold on R. Compute nonmax suppression.

Corner detection

– If λ_1 and λ_2 are small, means we are in a flat region

– If $\lambda_1 >> \lambda_2$ significant change in one direction, it is an edge

– If $\lambda_1 \approx \lambda_2$, and both are large, it is a corner

Score, $R = det(M) - k(trace(M))^2$

$$det(M) = \lambda_1 \lambda_2$$

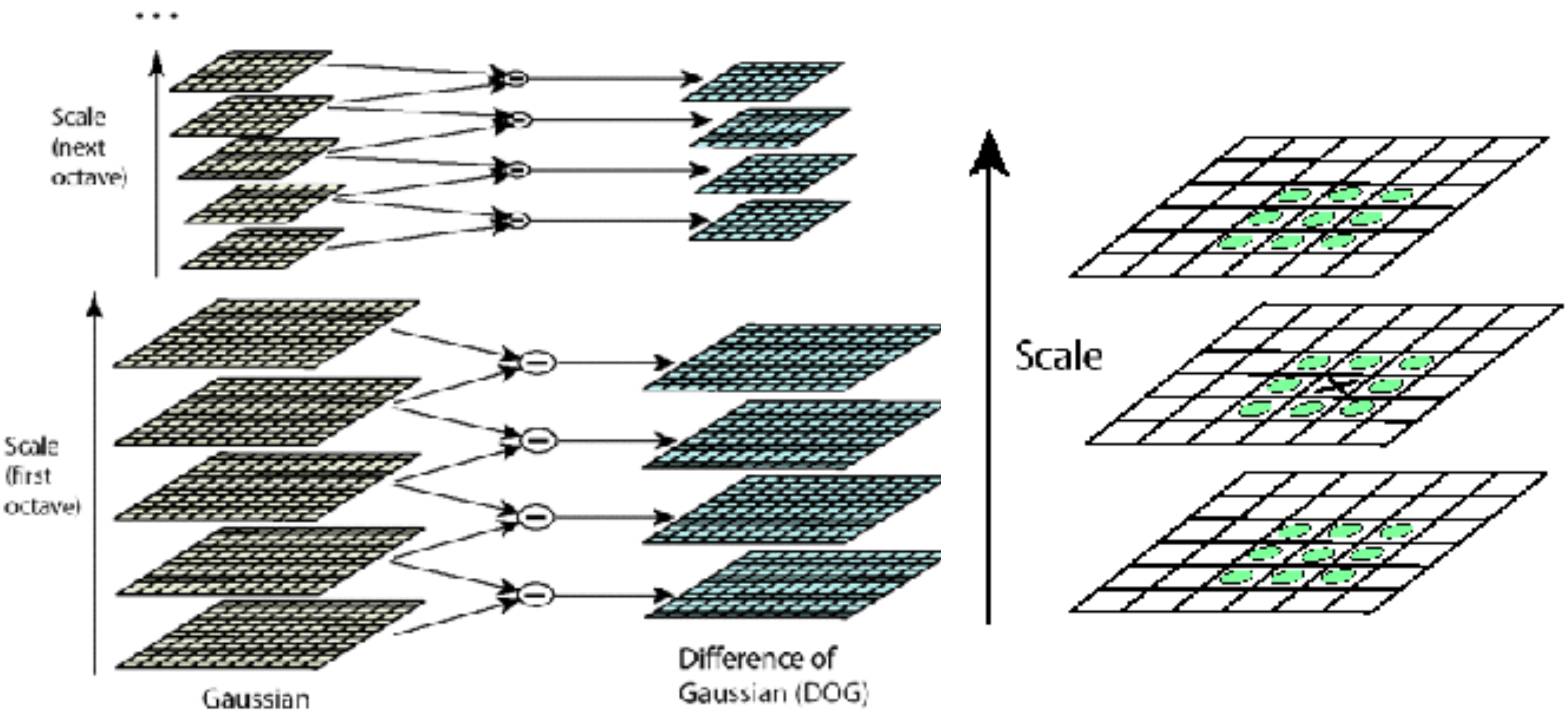
$$trace(M) = \lambda_1 + \lambda_2$$

k is an emppirically determined constant; $k = 0.04 - 0.06$

SIFT Algorithm

1. **Detection**
   – Detect points that can be repeatably selected under location/scale change
2. **Description**
   – Assign orientation to detected feature points
   – Construct a descriptor for image patch around each feature point
3. **Matching**

1. Feature detection - Key point detection



1. Key point localization

1) Sub-pixel/sub-scale interpolation using Taylor expansion

$$D(\overrightarrow{x}) = D + \frac{\partial D^T}{\partial \overrightarrow{x}}\overrightarrow{x} + \frac{1}{2}\overrightarrow{x}^T\frac{\partial^2 D}{\partial \overrightarrow{x}^2}\overrightarrow{x} \qquad ; \qquad \overrightarrow{x} = (x, y, \sigma)^T$$

Location of the extrema, $\hat{x} = -\frac{\partial^2 D}{\partial x^2}^{-1}\frac{\partial D}{\partial x}$

$$\frac{\partial D}{\partial x} = \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix} = \begin{bmatrix} \frac{D(x+1,y,\sigma) - D(x-1,y,\sigma)}{2} \\ \frac{D(x,y+1,\sigma) - D(x,y-1,\sigma)}{2} \\ \frac{D(x,y,\sigma+1) - D(x,y,\sigma-1)}{2} \end{bmatrix}$$

$$D(\hat{x}) = D + \frac{1}{2}\frac{\partial D}{\partial x}\hat{x}$$

Discard $|D(\hat{x})| < 0.03$

key points with low contrast

1. Key point localization - Eliminating edge response

1) Principal curvatures can be computed from a 2 x 2 Hessian matrix

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

$$\frac{\partial D}{\partial x} = \begin{bmatrix} \frac{\partial D}{\partial x} \\ \frac{\partial D}{\partial y} \\ \frac{\partial D}{\partial \sigma} \end{bmatrix} = \begin{bmatrix} \frac{D(x+1,y,\sigma) - D(x-1,y,\sigma)}{2} \\ \frac{D(x,y+1,\sigma) - D(x,y-1,\sigma)}{2} \\ \frac{D(x,y,\sigma+1) - D(x,y,\sigma-1)}{2} \end{bmatrix}$$

1. Feature detection - Keypoint localization

- Discard low-contrast/edge points
  1) Low contrast: discard keypoints with threshold < 0.03
  2) Edge points: high contrast in one direction, low in the other → compute principal curvatures from eigenvalues of 2x2 Hessian matrix, and limit ratio

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}$$

$$\mathrm{Tr}(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta,$$

$$\mathrm{Det}(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta.$$

$$r = \frac{\alpha}{\beta} \qquad \frac{\mathrm{Tr}(\mathbf{H})^2}{\mathrm{Det}(\mathbf{H})} < \frac{(r+1)^2}{r} \qquad r = 10$$

2. Orientation Assignment

- Assign orientation to keypoints
  - Create histogram of local gradient directions computed at selected scale

Orientation histogram has 36 bins each covering 10 degrees

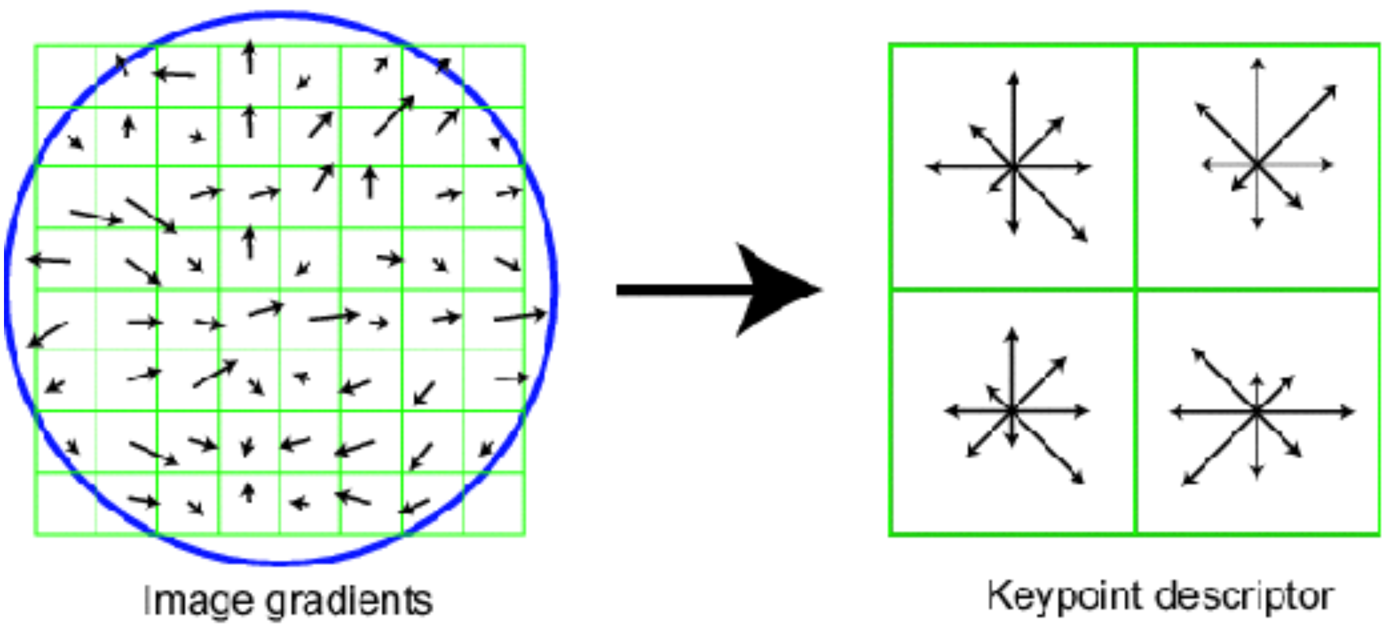
Peaks in the orientation histogram correspond to dominant directions of local gradients.

Any other local peak, within 80% of the highest peak is also used to create a key point with that orientation.

There may be multiple key points with same location and scale but different orientation.

2. Feature description

- Construct SIFT descriptor
  - Create array of orientation histograms
  - 8 orientations x 4x4 histogram array = 128 dimensions



3. Feature matching

- Example: 3D object recognition