Review image 1st - Robust estimation

When the data points have some outliers that are not of the same property/not in line with the majority of data points, then the distance between that point and the line becomes larger-->thereby making the squared error more bigger--> so it becomes more influential and alters the model path and brings the fitting line more unaligned.
The more wrong data point , the more influential it is on the model

Robust estimation

* Naive approach: to solve the outlier issue
  - Fit model to all points
  - compute distance of each point from model
  - Discard points with largest distance
  - Fit model to remaining points

If * Initial model is inaccurate and so we drift in the wrong direction.

* Approaches:
  - M-estimators
  - RANSAC

M-estimators

- Mean square Error (MSE) fitting:

$$E(\theta) = \sum d^2(x_i; \theta)$$

e.g. $d^2 = (l^T x_i)^2$ for line fitting

- Robust estimation:

$$E(\theta) = \sum \rho_\sigma(d(x_i; \theta))$$

MSE is a special case where $\rho_\sigma(x) = x^2$

trying to reduce the distance between the points-->squared

instead of square, we have a different function

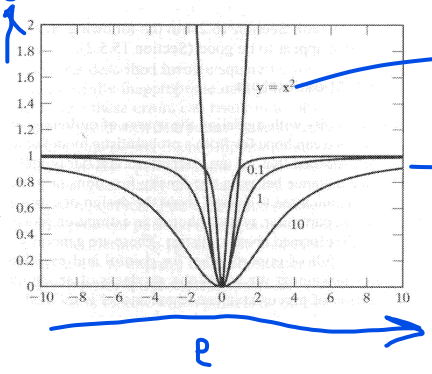
Geman-McClure estimator

$$\rho_\sigma(x) = \frac{x^2}{x^2 + \sigma^2}$$

$x \gg \sigma \Rightarrow \rho_\sigma(x) = 1$

$x \ll \sigma \Rightarrow \rho_\sigma(x) = \frac{x^2}{\sigma^2}$

larger σ ⇓ wider valley



if the function is square, we see that for small error value, the squared d --> loss function varies steeply giving large values of loss

for sigma = 0.1,1,10, we see that the parabola changes offering a variance of values gradually and even if an outlier is very very wrong it can only make an impact of maximium value 1

Refer image

Geman-McClure estimator

a different function

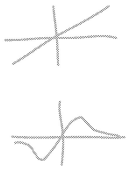
$$E(\theta) = \sum \rho_\sigma \left(d(x_i; \theta) \right)$$

$$\nabla E(\theta) = \sum \frac{\partial}{\partial d} \rho_\sigma(d) \frac{\partial}{\partial \theta} d(\theta)$$

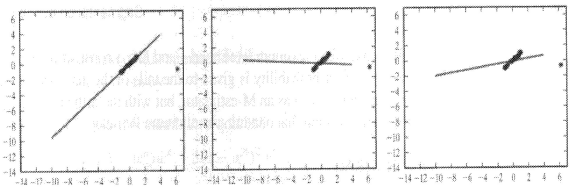
not bound

For $\rho_\sigma(d) = d^2 \Rightarrow \frac{\partial \rho}{\partial d} = 2d$

For $\rho_\sigma = \frac{d^2}{d+\sigma^2} \Rightarrow \frac{\partial \rho}{\partial d} = \frac{2d\sigma^2}{(d^2+\sigma^2)^2}$



Robust estimation example



Right sigma | Small sigma | Large sigma

Selecting bandwidth parameter

- large $\sigma \Rightarrow$ include more points
- small $\sigma \Rightarrow$ include fewer points
- variable estimation:
  start with large σ and decrease as converging

$$\sigma^{(n)} = 1.5 \text{ median } \{ d(x_i; \theta^{(n-1)}) \}$$

$\sigma^{(n)}$: estimate at step n; $\theta^{(n-1)}$: parameters at step $n-1$

helps to average the errors

M-estimator summary

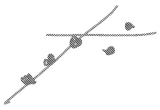
1) Draw a large set of points uniformly at random
2) Select initial value of σ
3) Fit model $\rightarrow \theta^{(i)}$
4) Compute $\sigma^{(i)}$ using median distance of points
5) continue while objective is decreasing

* To overcome incorrect initial guess of σ repeat several times and select best solution (smallest objective)

RANSAC

Random Sample Consensus (RANSAC):

- perform multiple experiments
- choose best results
- use small sets in hope that at least one set will not have outliers



Parameters:

n = # points drawn at each evaluation for lines, n>=2 --->we generally take atleast the minimum

d = min # points needed to estimate model d= 2 for line

K = # trials

t = distance threshold to identify inliers

RANSAC Algorithm

* Repeat K times:
  - Draw n points uniformly at random (with replacement)
  - Fit a model to points
  - Find inliers in entire set (distance $< t$)
  - Recompute model (if at least d inliers)
  - update parameters (K, t)
* Choose best solution - generally used
  - Largest consensus set -more points agree on the same solution- in RANSAC
  (- or smallest error)

Estimating RANSAC parameters

* To estimate t use median distance from model.
* To estimate k use:

p: with probability of p at least one experiment does not have outliers (e.g. p = 0.99) user selected

w: probability that a point is an inlier (initially w = 0.5) estimated

caculate the distance for allpoints and take median and use that to identify if a point is a inlier or outlier---->we take median and not average because average is sensitive to outlier and affects the values

choosing p is based on how much we wish to wait

Estimating RANSAC parameters

probability that all k experiments failed:

$(1-p) = (1-w^n)^k$

$\log(1-p) = k \log(1-w^n)$

$k = \frac{\log(1-p)}{\log(1-w^n)}$

large p → large k

small w → large k

$w \leftarrow \frac{\#\text{ inliers}}{\#\text{ points}}$

update w, k every iteration but set upper bound for k

RANSAC can be used in neural network, but the cost to assess the model is high -->so, not used

Eg - dont iterate more than 10000 iterations

Segmentation

* split image into sub-parts:

Clustering
1) Based on color
2) Based on spatial location - similar pixels in the same spatial location/area

Classification
3) Based on features
4) Based on semantics pixels belonging to same person/object

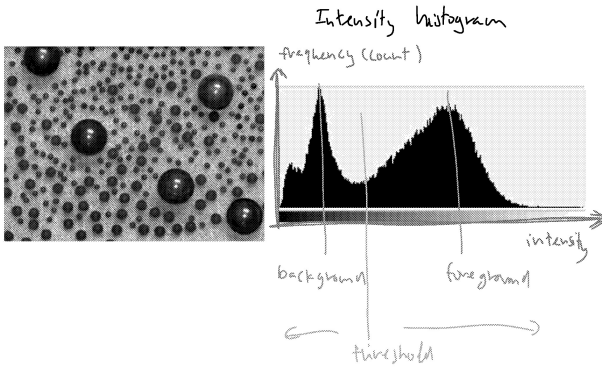
Problem statement

- Separate object(s) from background
⇔
- Find contours of objects

- Semantic image segmentation: label each pixel in the image with class label

Refer images

Color segmentation



Spatial context

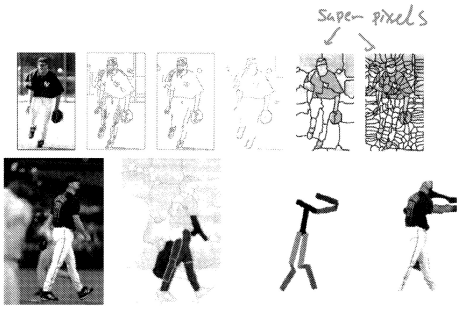


FIGURE 9.14: Superpixels can expose structure in images that other representations conceal (e.g. human body segments tend to appear as long, thin segments). **(a)** In the top row, an image together with three different edge maps and superpixels computed at two "scales". Note that the coarser superpixels tend to expose limb segments. **(b)** On the bottom row, another image, its superpixels, and two versions of the body layout inferred from the superpixel representation.

Feature based segmentation

* Define feature vector at each pixel x:

$$F(x) = \begin{bmatrix} x \\ I(x) \\ L(x) \end{bmatrix}$$

- x ← location of the pixel x
- $I(x)$ ← intensity — it is 1 number -->if b/w.....else 3 (rgb)
- $L(x)$ ← local characteristics (e.g. texture) — local window

* Apply clustering

Clustering

* Algorithms: ML
  - k-means
  - Mixture of Gaussians
  - Mean shift
  - Expectation maximization
  - Graph cuts
  - spectral clustering