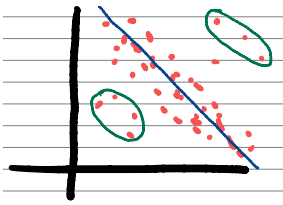
Robust estimation

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

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



○ → outliers, they are removed

This model can be inaccurate because we remove outliers. This outliers can be valid data points and they elimination could lead to a wrong direction

* Approaches:
  - M-estimators → Modify the loss function to reduce the impact of outliers. Outliers are not discarded, their weights are down
  - RANSAC → Use a subset of data points and checks which points fit well. Repeat to find the best model with the highest number of inliers

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

distance between the point x and the model prediction defined by θ

- Robust estimation:

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

ρ_σ → loss function that reduce the impact of outliers

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

Geman-McClure estimator: Type of loss function to reduce the influence of outliers

Loss function of Geman-McClure estimator

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

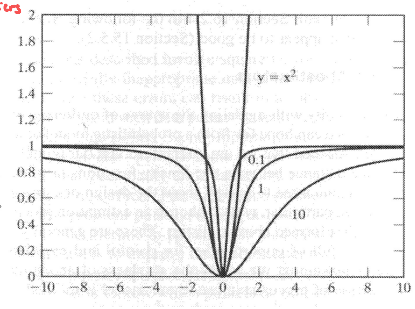
Parameter that control how aggressively the estimator treat outliers

Residual or error

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

This value limits the contribution of large residuals to the total error

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



larger σ
⇓
wider valley
↓
larger σ reduce the penalty for larger residuals

Geman-McClure estimator

$$E(\theta) = \sum \rho_\sigma \left(d(x_i; \theta) \right)$$ ← we try to minimize for robust estimation using its gradient

$$\text{gradient} \rightarrow \nabla E(\theta) = \sum \left(\frac{\partial}{\partial d} \rho_\sigma(d) \right) \left(\frac{\partial}{\partial \theta} d(\theta) \right)$$

* $\frac{\partial}{\partial d} \rho_\sigma(d)$ → Represents how the loss function ρ_σ changes with respect to the residual d

* $\frac{\partial}{\partial \theta} d(\theta)$ → Represents the residual changes with respect to the model parameters

not bound

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

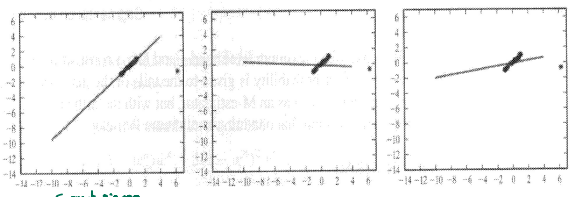
← Gradients grows linearly with d, meaning large residuals (outliers) have a big influence on the overall model

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

← Gradient becomes smaller, outliers are down-weighted and have less impact compare to least square

Summary: Square error (MSE) is more sensitive that Geman-McClure estimator because higher residuals have less impact

Robust estimation example



Correct sigma

Right sigma
Capture the trend of the inliers while ignoring the outlier

Small sigma
The model is sensitive to small residuals (inclusive inliers are treated as potential outliers)

Large sigma
All point are treated as inliers

Selecting bandwidth parameter

- large σ ⇒ include more points
- small σ ⇒ 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

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

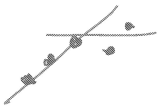
* To overcome incorrect initial guess of σ repeat several times and select best solution (smaller objective)

① Select a subset
② Select a σ (large $\sigma \rightarrow$ more points)
③ Fit the model to get θ (based on minimizing M-estimator loss function)
④ Compute the σ now using the median distance of points
⑤ Continue until the objective is decreasing

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



Random → RANSAC → Consensus
Sample

① Select subset
② Train this subset
③ Choose the best → Explain the majority of outliers / Does not have outliers (hope)

• Uses small sets in hope at least one will not have outliers

Parameters:

n = # points drawn at each evaluation → Define the size of subset
d = min # points needed to estimate model → Estimate a (Line → At least 2 point; Plane → At least 3 points)
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:
  - Largest consensus set
  (- or smallest error)

① Select subset
② Fit a model to the points
③ Find inliers (distance $< t$)
④ Recompute the model (if d inliers are found) → Improve the accuracy
⑤ Update K and t (iterations, threshold)

Best model
Less error (or)
Best consensus set

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

How to select
- t → Median distance from the model
- k based on
  - p → Probability that at least one trial does not include outliers (normally 0.99) — User-selected
  - w → Probability that any given point is an inlier (normally 0.5) — Estimated

Estimating RANSAC parameters

probability that all k experiments failed:

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

$(1-p)$: Probability that at least one trial succeeds (no outliers)

$(1-w^n)$: Probability that a random subset of n points contains only inliers

$\log(1-p) = k \log(1-w^n)$ ← Taking log of both sides...

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

large p → large k — A higher confidence level requires more iterations

small w → large k — If w is small, means there are many outliers

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

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

Segmentation

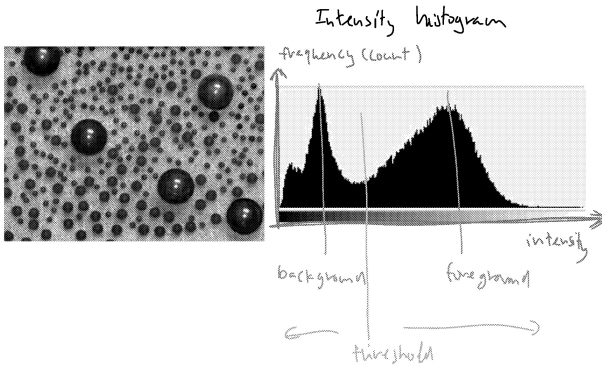
* Split image into sub-parts:
  1) Based on **color**
  2) Based on **spatial location**
  3) Based on **features**
  4) Based on **semantics**

Problem statement

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

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

Color segmentation



Spatial context

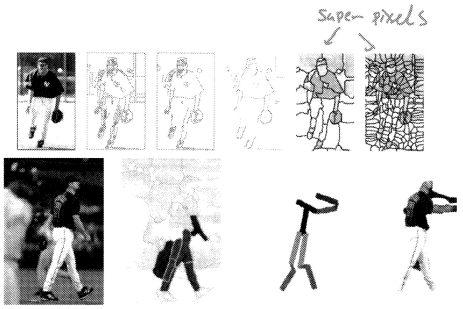


FIGURE 9.14: Superpixels can expose structure in images that other representations conceal (e.g. human body segments tend to appear as long, thin segments). **(a)** In the top row, an image together with three different edge maps and superpixels computed at two "scales". Note that the coarser superpixels 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:

$$F(x) = \begin{bmatrix} x \\ I(x) \\ L(x) \end{bmatrix} \begin{array}{l} \leftarrow \text{location} \\ \leftarrow \text{intensity} \\ \leftarrow \text{local characteristics} \\ \quad \text{(e.g. texture)} \end{array}$$

* Apply clustering

Clustering

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