Robust Methods in Computer Vision

CS 763

Ajit Rajwade

Outline

- Least squares estimates
- Limitations of least squares and need for robustness
- Least median of squares method
- RANSAC method
- Application scenarios

Least squares Estimates

- Consider quantity *y* related to quantity *x* in the form $y = f(x;\mathbf{a})$.
- Here **a** is a vector of parameters for the function *f*. For example, $y = mx + c$, where $\mathbf{a} = (m,c)$.
- Now consider, we have *N* data points (x_i,y_i) where the y_i could be possibly corrupted by noise, and want to estimate **a**.
- This is done by minimizing the following w. r. t. **a**:

$$\sum_{i=1}^{N}(y_i - f(x_i;\mathbf{a}))^2$$

Least squares Estimates

- Why did we minimize the squared error loss? What would happen if we changed the power to 4? Or 1 or 3 (with absolute value)?

- Let us assume that the noise affecting y_i is (1) additive, and (2) Gaussian with mean zero and some known standard deviation σ. Then:

$$y_i = f(x_i; \mathbf{a}) + \eta_i, \eta_i \sim N(0, \sigma)$$

Least squares Estimates

- Let us also assume that the noise values affecting the different samples are independent of each other.
- Now given some value of **a** and some x_i, the probability density of y_i is:

Likelihood of y_i

Likelihood of $\{y_i\}$, i=1 to N

$$p(y_i \mid x_i, \mathbf{a}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y_i - f(x_i;\mathbf{a}))^2}{2\sigma^2}}$$

$$p(\{y_i\}_{i=1}^N \mid \{x_i\}_{i=1}^N, \mathbf{a}) = \prod_{i=1}^N p(y_i \mid x_i, \mathbf{a}) \text{ by independence assumption on noise}$$

Least squares Estimates

- The probability density of a random variable Y at a value y is defined as follows:

$$p_Y(y) = \lim_{\delta \to 0} \frac{P(y \le Y \le y + \delta)}{\delta}$$

Least Squares Estimates

- We want to find a value of **a** which **maximizes** this probability density. This is called the **maximum likelihood estimate** of **a**.
- ..equivalent to **maximizing** the log of this probability density (why?)
- ..equivalent to **minimizing** the **negative** log of this probability density, i.e.

$$J(\mathbf{a}) = \sum_{i=1}^{N} \frac{(y_i - f(x_i;\mathbf{a}))^2}{2\sigma^2} + \log\left(\sigma\sqrt{2\pi}\right)$$

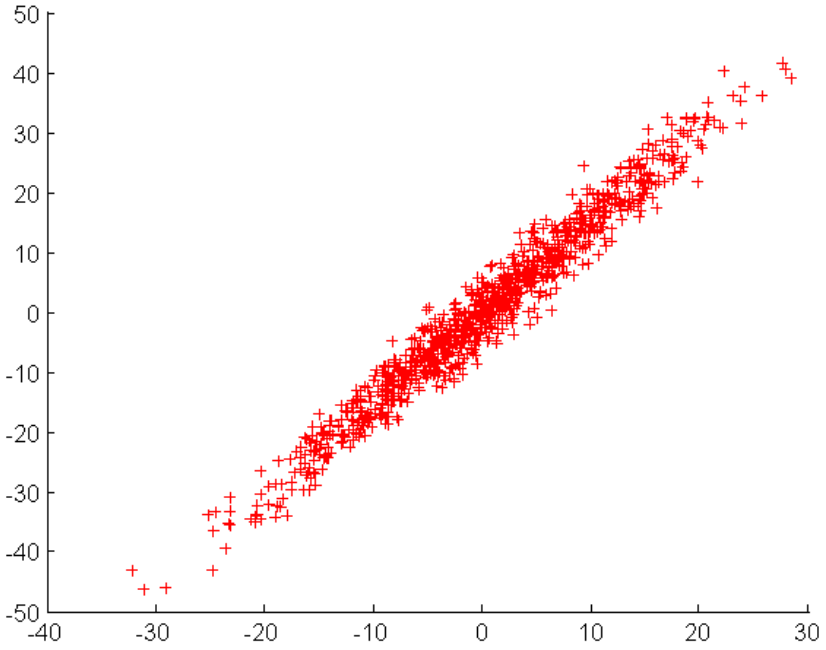
Least Squares Estimates

- This shows us that the least squares estimate is the same as the **maximum likelihood** estimate under the assumption that the noise affecting different samples was ***independent*** and ***Gaussian*** distributed with ***fixed*** variance and mean 0.
- Why maximum likelihood estimate of **a**? Intuitively, it is the value of **a** that best agrees with or supports the observations $\{y_i\}$, $1 \leq i \leq N$.

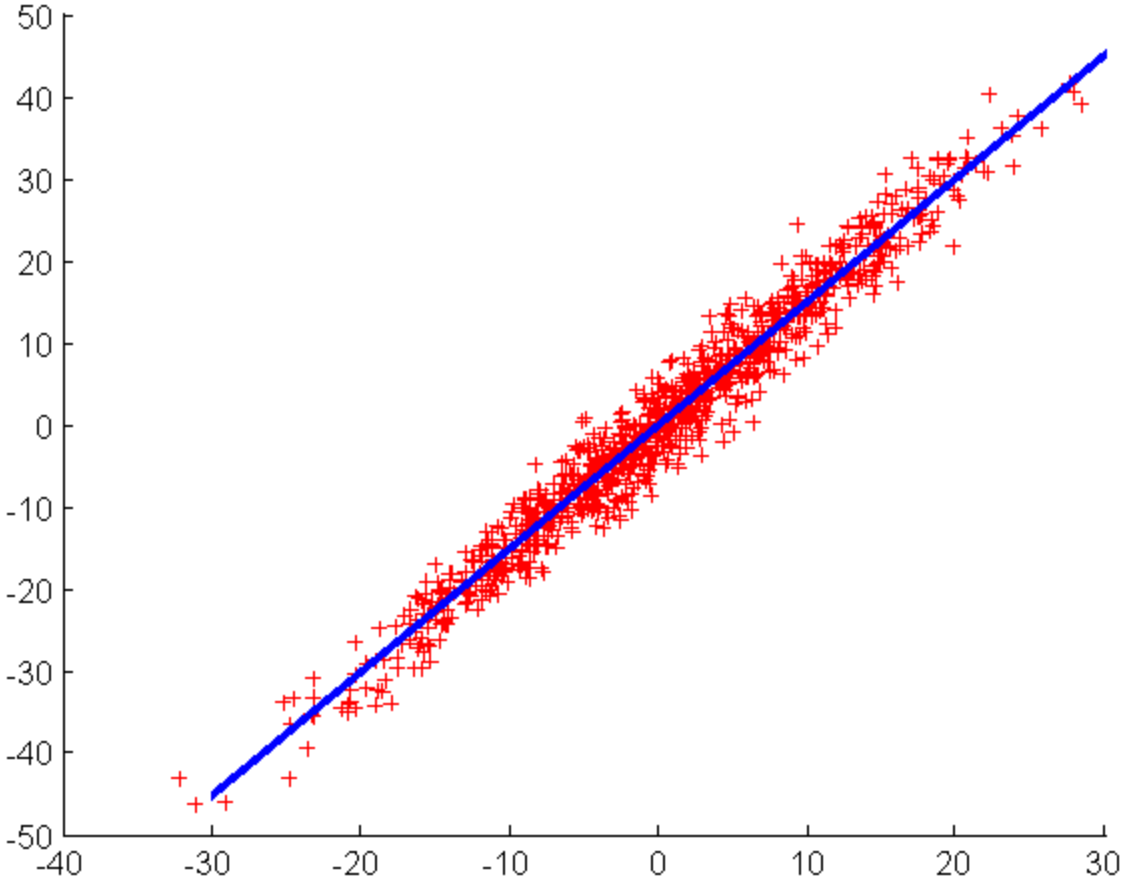
Least squares fit of a line

$$(m^*, c^*) = \arg\min J(m,c) = \sum_{i=1}^{N} (y_i - mx_i - c)^2$$

$$\therefore \begin{pmatrix} \sum_i x_i^2 & \sum_i x_i \\ \sum_i x_i & N \end{pmatrix} \begin{pmatrix} m \\ c \end{pmatrix} = \begin{pmatrix} \sum_i x_i y_i \\ \sum_i y_i \end{pmatrix}$$




```
N = 1000;x = 10*randn(N,1);
y = 1.5*x + randn(N,1)*3;
 scatter(x,y,'r+')
```



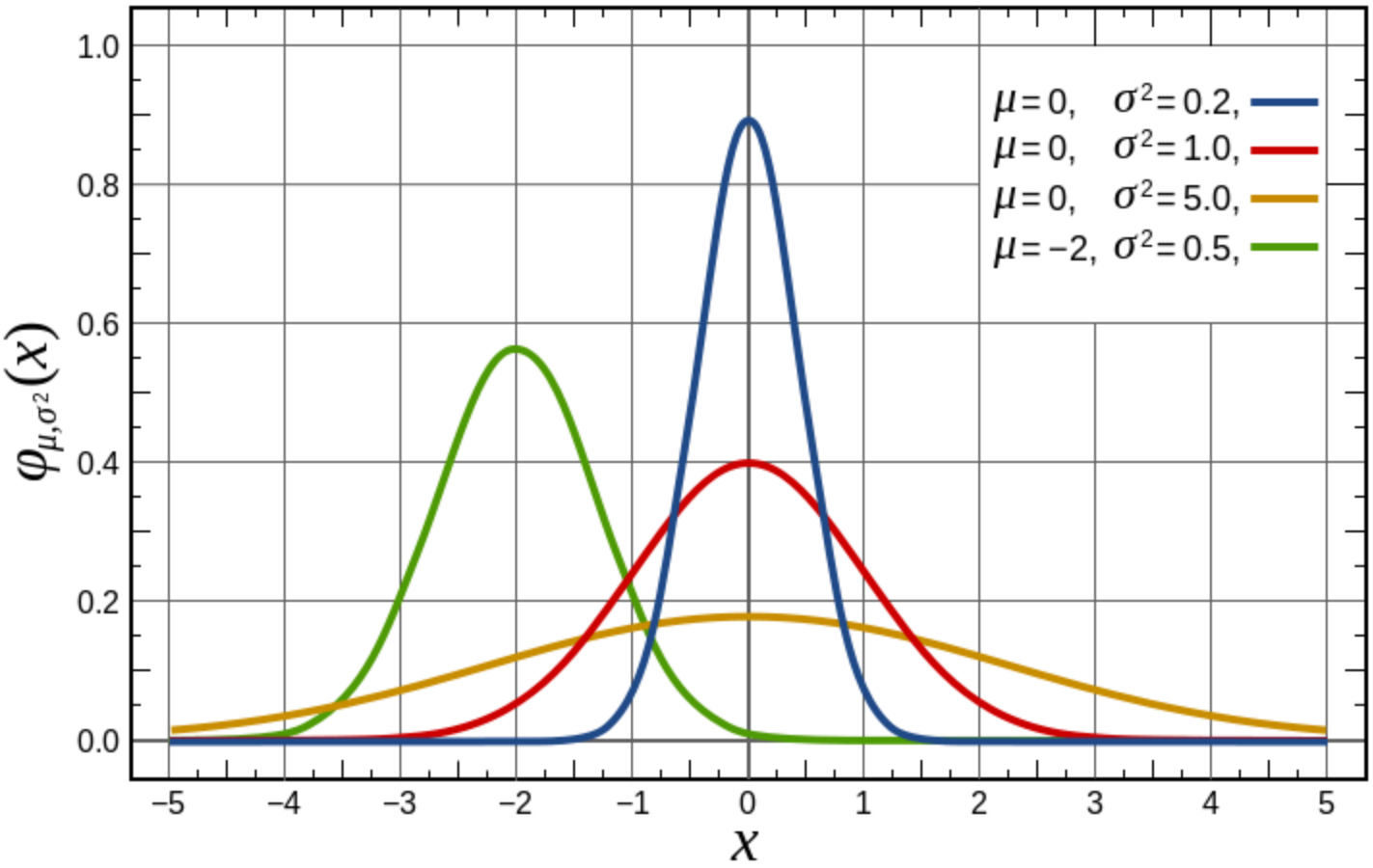
Result of a least-squares estimate under Gaussian noise:

Estimated slope: 1.5015 (versus 1.5)
Estimated intercept: 0.088 (versus 0)

Other Least-squares solutions in computer vision

- Camera calibration – SVD
- Parametric motion estimation – SVD or pseudo-inverse (affine, rotation, homography, etc)
- Fundamental/essential matrix estimation – SVD *(we will study this later in stereo-vision)*
- Optical Flow (Horn-Shunck as well as Lucas-Kanade) *(we will study this later)*

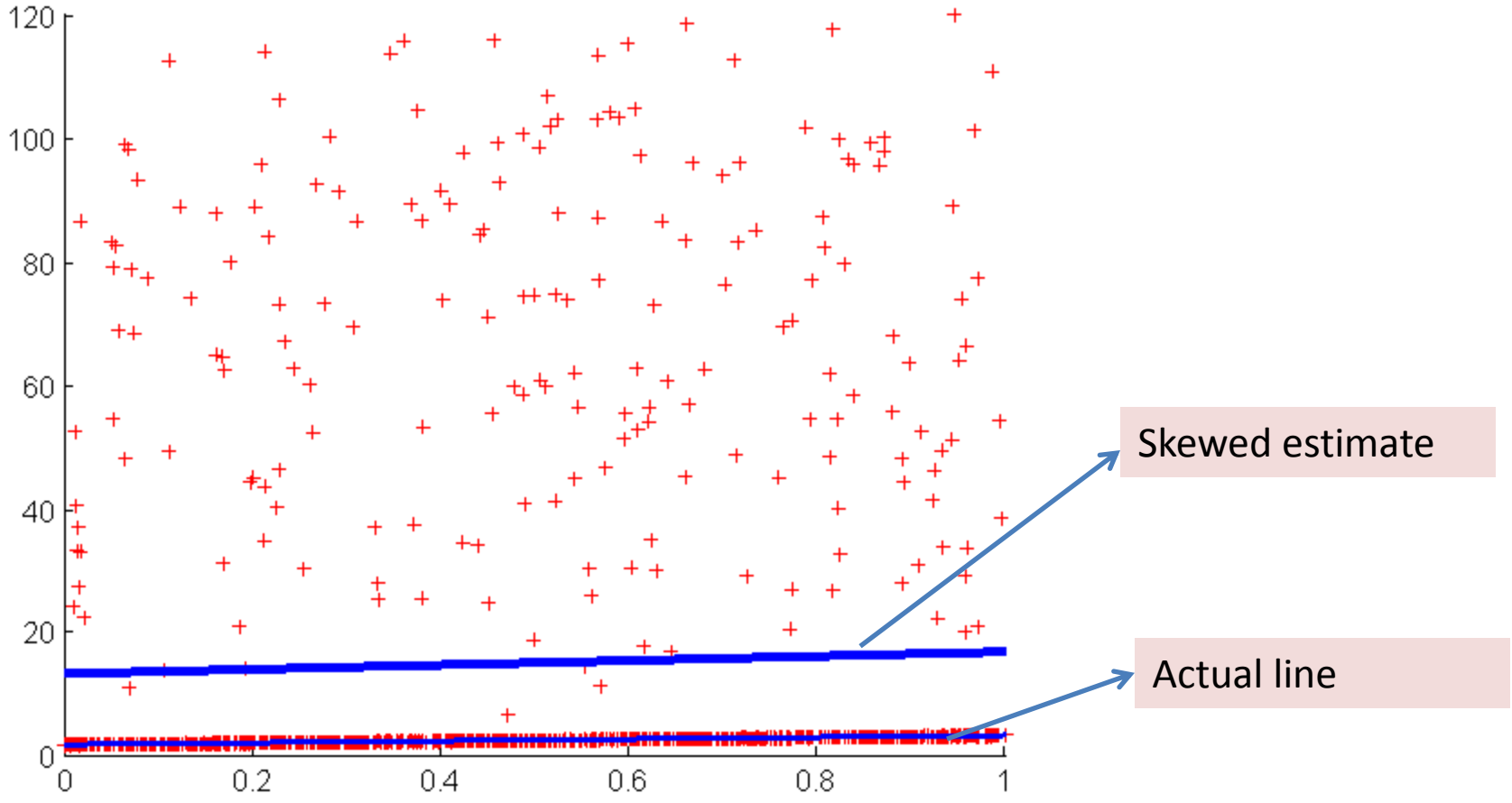
Outliers and Least-squares estimates



Observation: Let *x* be a random variable with a Gaussian distribution. Then the probability that *x* takes on any value in a small range far away from the mean (typically at a distance of more than +/- 3σ) is **very low**. See diagram above.

Outliers and Least-squares

- The upshot of the previous observation is this: the least squares estimate assumes that most points will lie ***close*** to the true (unknown) model – else their probability would be very low.
- Now, suppose the given dataset contains wild outliers, i.e. stray points that simply do not obey the model.
- These outliers will skew the least squares estimate – as it tries to force a solution which maximizes the likelihood of the ***outliers*** as well.
- Since outliers were extremely unlikely under the Gaussian probability density, the model (during maximum likelihood estimation) has to change itself to make the outliers more likely.



20% of the points are outliers. They have skewed the estimate of the slope from 1.5 to 3.6 and the intercept from 2 to 13.5.

Examples of outliers: (1)

- Salt and pepper noise in images

Salt and pepper noise (a special case of impulse noise)

Gaussian noise

Don’t worry about this example right now – we will encounter it when we study shape from shading and photometric stereo

Examples of outliers: (2)

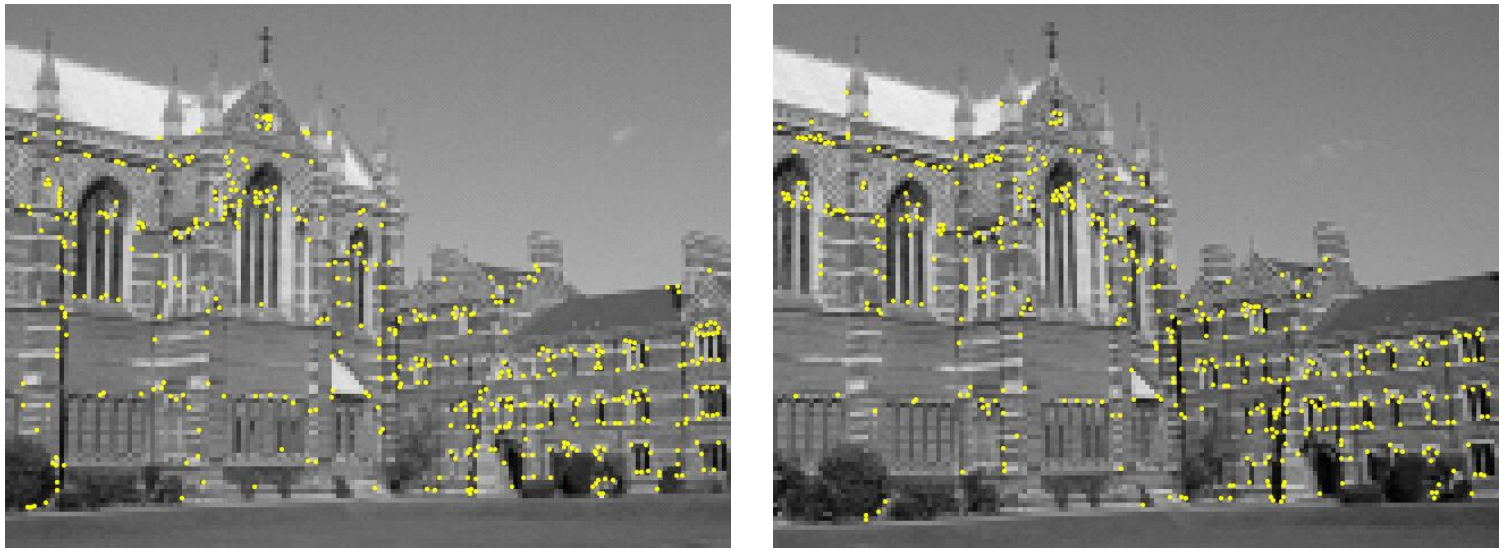
- Shadows and specularities act as outliers in photometric stereo!

Fig. 7. **Input images (4 out of 40).** With all kinds of corruptions: Specularity, cast shadow, attached shadow.

Images taken from paper:
perception.csl.illinois.edu/matrix-rank/Files/**robust** v19.pdf

Examples of outliers: (3)

- Estimation of the spatial transformation between images (could be translation, rotation, affine or homography) requires *N*+ pairs of corresponding points. Some of these correspondences can be faulty.



Examples of outliers: (3)

- Some of these correspondences can be faulty for various reasons
- ❑ occlusions/ difference in field of view / shadows
- ❑ algorithm errors
- ❑ identical objects in the scene
- ❑ change in the position of some objects in the scene (even though the global motion is homography, these objects will not conform to that same motion model).

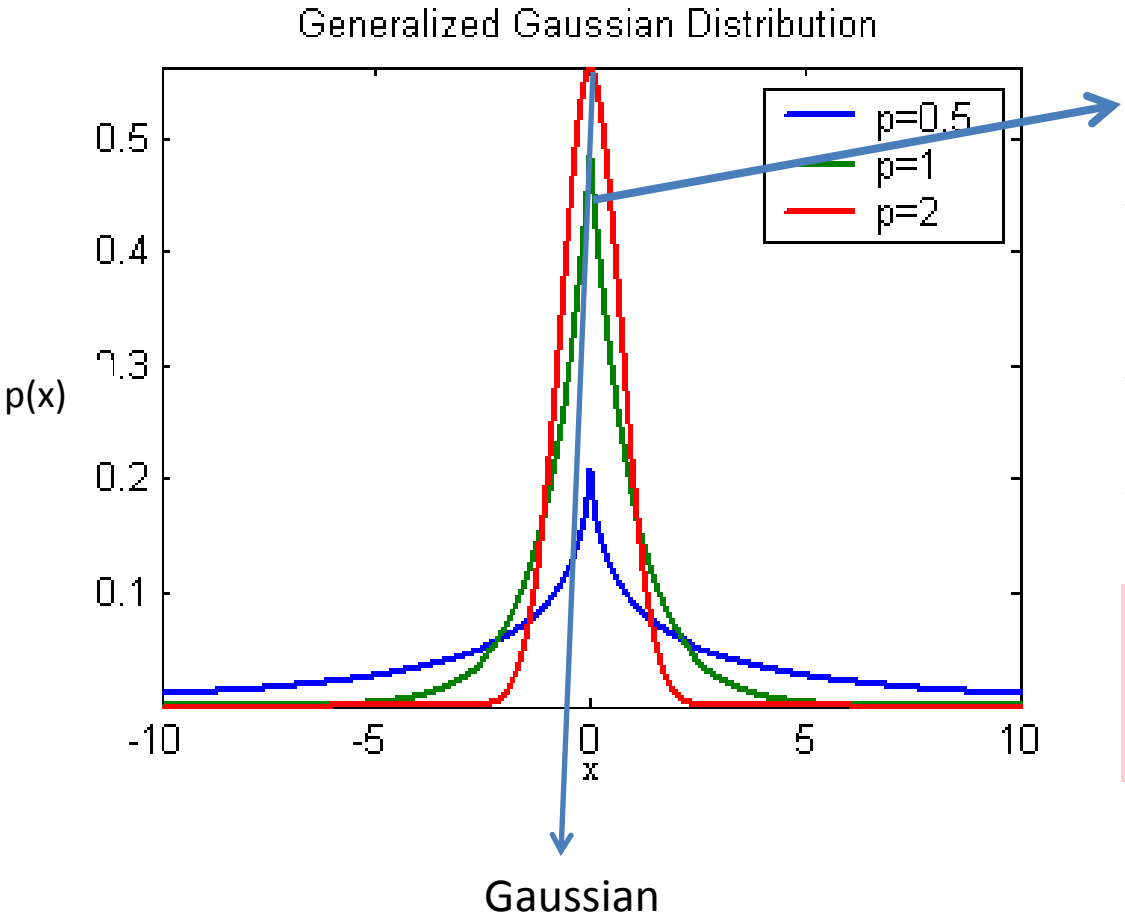
http://petapixel.com/2012/10/22/an-eerie-time-lapse-of-seattle-minus-all-the-humans/

Examples of outliers: (4)

- The motion between consecutive frames of a video in the following link may be modeled as affine. But some corresponding pairs of points (example: on independently moving objects) don't conform to that model – those are outliers.

  https://www.youtube.com/watch?v=17VAuBL1Lxc

Dealing with Outliers: (1)

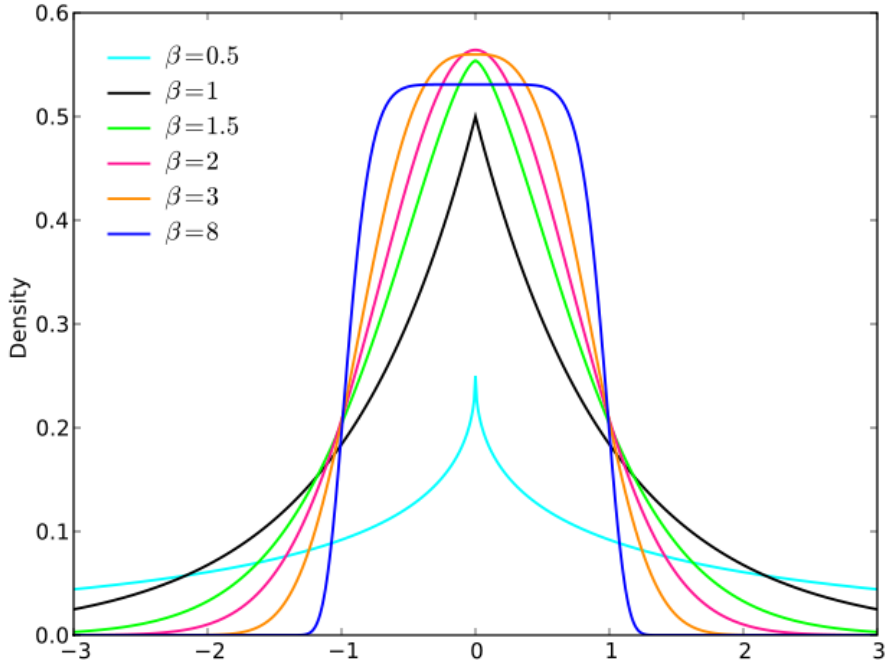


Laplacian probability density function: a distribution which has **heavier** tails than the Gaussian. This means that if a random variable is Laplacian distributed, the probability that it can take values far away from the mean, is higher than what it would be if the variable were Gaussian distributed.

$$p(x) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}}$$

The Laplacian (or Laplace) pdf is **not** to be confused with the Laplacian of a function $f(x,y)$, given as $f_{xx}(x,y) + f_{yy}(x,y)$ that we had studied in image processing.

Dealing with Outliers: (1)

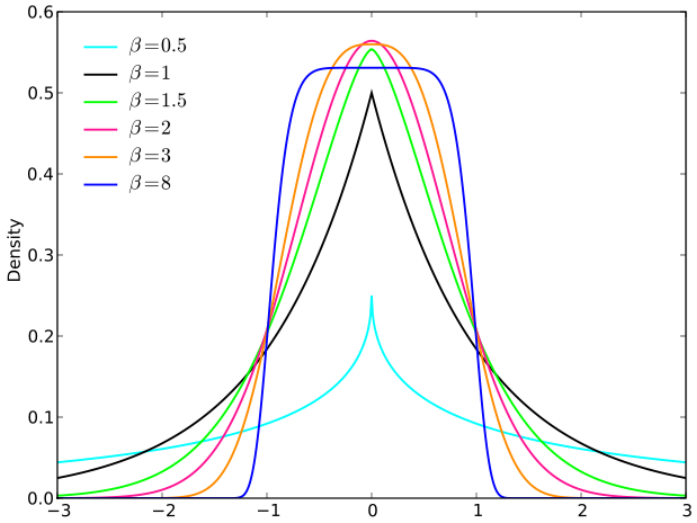


$$p(x) = \frac{\beta}{2\alpha\Gamma(1/\beta)} e^{-\left(\frac{|x-\mu|}{\alpha}\right)^{\beta}}$$

The Laplacian probability density function is a special case of the family of **Generalized Gaussian probability density functions** with **shape parameter** β and **scale parameter** α. As β reduces below 1, the density function becomes heavier tailed.

Question: Why do we care for heavier tails? Because they ensure that the wild outliers are more likely to occur (than the Gaussian pdf). Consequently, a maximum likelihood estimation assuming heavy-tailed noise models will be less affected by outliers.

Dealing with Outliers: (1)

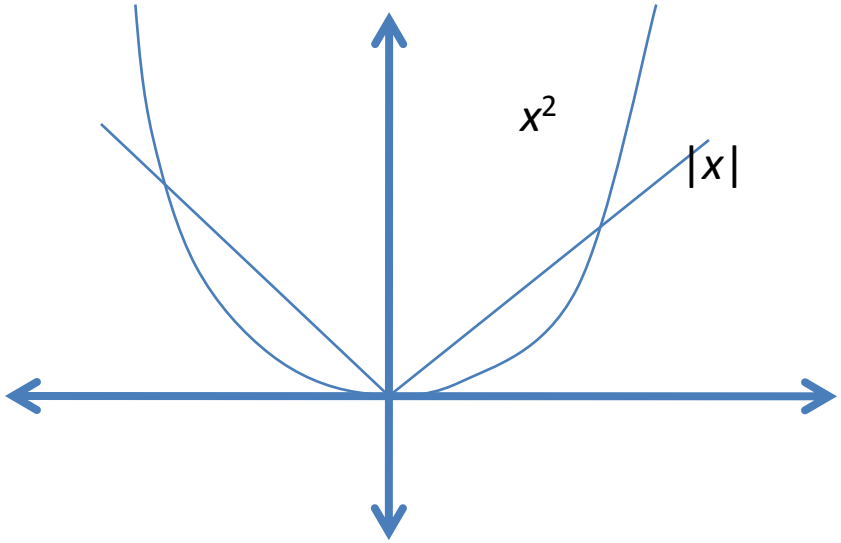


$$p(x) = \frac{\beta}{2\alpha\Gamma(1/\beta)} e^{-\left(\frac{|x-\mu|}{\alpha}\right)^{\beta}}$$

Note that x^p grows faster than x^q beyond |x| = 1, for p > q

Why do these Generalized Gaussians with shape parameter β < 2 have heavier tails than the usual Gaussian (i.e. β = 2)?

Consider a zero-mean Gaussian and a zero-mean Laplacian (without loss of generality), i.e. β = 1.
The term inside the exponential in a Gaussian is x^2, where it is $|x|$ for a Laplacian.



Dealing with Outliers: (1)

- Assume the noise has a Laplacian distribution.
- The maximum likelihood estimate of **a** is then given by minimizing the following:

$$J(\mathbf{a}) = \sum_{i=1}^{N} \frac{|y_i - f(x_i;\mathbf{a})|}{b} - \log(2b) \approx \sum_{i=1}^{N} \frac{\sqrt{(y_i - f(x_i;\mathbf{a})^2 + \varepsilon}}{b} - \log(2b)$$

- Unfortunately, there is no closed-form solution (based on inverse/pseudo-inverse) in this case – unlike the case for squared error!
- One will need iterative methods like adaptive gradient descent.

Dealing with Outliers: (1)

- One will need iterative methods like adaptive gradient descent.

Repeat till there is no change in $\mathbf{a}$

{

$\alpha = \alpha_{\max}$

while $(\alpha > \alpha_{\min})$

{

$\mathbf{a}' = \mathbf{a} - \alpha \dfrac{dJ(\mathbf{a})}{d\mathbf{a}}, 0 < \alpha << 1$

if $J(\mathbf{a}') > J(\mathbf{a}), \alpha = \alpha_{\max} / 2$

else $\{\mathbf{a} = \mathbf{a}'; \text{break}\}$

}

}

α = **step size** of gradient descent

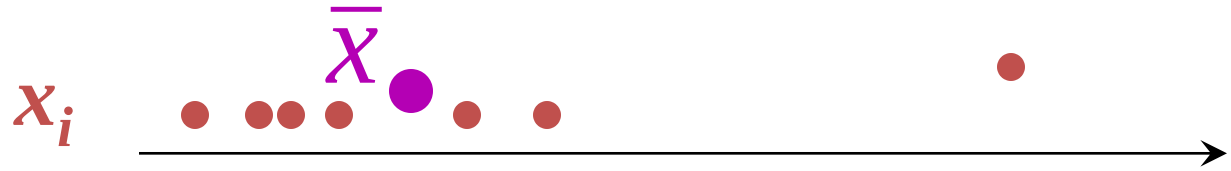
Gradient descent converges to a local minimum of the energy function (or objective function), i.e. J(**a**) in this slide, if the step size α is "small enough" to never.

Unfortunately, too small a step size is too expensive. A large step size may lead to increase in the energy function across iterations.

So we pick the largest possible step-size (within a given range) that reduces the energy – this is called **gradient descent with adaptive step-size** or **adaptive gradient descent**.

Robust statistics – simple example

Find "best" representative for the set of numbers



L2: $J(\bar{x}) = \sum_i |\bar{x} - x_i|^2 \rightarrow \min$

L1: $J(\bar{x}) = \sum_i |\bar{x} - x_i| \rightarrow \min$

Influence of x_i on E: $x_i \rightarrow x_i + \Delta$

$J_{new} \cong J_{old} + 2(x_i - \bar{x}) \cdot \Delta + \Delta^2$

proportional to $|\bar{x} - x_i|$

Outliers influence the most

$\bar{x} = \text{mean}(x_i)$

$J_{new} \cong J_{old} \pm \Delta$

equal for all x_i

Majority decides

$\bar{x} = \text{median}(x_i)$

Slide: Courtesy - Darya Frolova, Recent progress in optical flow

Elections and Robust statistics

many ordinary people

a very rich man

wealth

Oligarchy

Votes proportional to the wealth

like in L2 norm minimization

Democracy

One vote per person

like in L1 norm minimization

Slide: Courtesy - Darya Frolova, Recent progress in optical flow

New ways of defining the mean

- We know the mean as the one that minimizes the following quantity:

$$E(\mu) = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n} \rightarrow \mu = \frac{\sum_{i=1}^{n} x_i}{n}$$

- Changing the error to sum of absolute values, we get:

$$E(\mu) = \frac{\sum_{i=1}^{n} | x_i - \mu |}{n} \rightarrow \mu = median(\{x_i\}_{i=1}^{n})$$

We will prove this in class!

New ways of defining the mean

- We can also use errors of the following type with $0 < q \leq 1$:

$$E(\mu) = \frac{\left(\sum_{i=1}^{n} | x_i - \mu |^q \right)^{1/q}}{n}$$

- Optimizing the above requires iterative methods (no closed form solutions).
- The mean computed using $0 < q \leq 1$ is quite robust to outliers – with q greater than or equal to 2, the mean is susceptible to outliers.

New ways of defining the mean

- The earlier definitions of the mean were for scalars. They can be extended for vectors in some *d* > 1 dimensions as well.

$$E(\boldsymbol{\mu}) = \frac{\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{\mu})^2}{n} \rightarrow \boldsymbol{\mu} = \frac{\sum_{i=1}^{n}\mathbf{x}_i}{n}$$

- For other q-norms (0 <= q < 1), we have:

$$E(\boldsymbol{\mu}) = \frac{\left(\sum_{i=1}^{n}|\mathbf{x}_i - \boldsymbol{\mu}|^q\right)^{1/q}}{n}$$

Dealing with outliers: (2) LMedS

- LMedS = Least Median of Squares
- It works as follows:

$$J(\mathbf{a}) = \text{median}_{i=1:N}\ (y_i - f(x_i;\mathbf{a}))^2;$$
$$\text{select } \mathbf{a} \text{ for which } J(\mathbf{a}) \text{ is minimum.}$$

- This has no closed form solution either and you can't do gradient descent type of techniques as the median is not differentiable.
- But it has an "algorithmic" solution.

Dealing with outliers: (2) LMedS

- **Step 1:** Arbitrarily choose *k* out of *N* points where *k* is the smallest number of points required to determine **a**. Call this set of *k* points as $\mathbf{C}$.
  - Eg: If you had to do line fitting, *k* = ?
  - Eg: If you were doing circle fitting, *k* = ?
  - Eg: If you have to find the affine transformation between two point sets in 2D, you need *k* = ? correspondences.

Dealing with outliers: (2) LMedS

- **Step 2:** Determine **a** using an inverse (say) from $\mathbf{C}$.
- **Step 3:** Determine the squared residual errors for all the other *N-k* points, i.e. compute

$$\{e_i = (y_i - f(x_i; \mathbf{a}))^2\}_{i \notin C}$$

- **Step 4:** Compute med_C = median of $\{e_i\}$.
- Repeat all these four steps for *S* different subsets of *k* points each.
- Pick the estimate of **a** corresponding to the $\mathbf{C}$ that has the least value of med_C .
- What's the time complexity of this algorithm?

Dealing with outliers: (2) LMedS

- S = number of subsets. What should be the minimum value of S?
- Let's say that some fraction p $(0 < p < 1)$ of the N points are inliers ("good points").
- Then the probability that at least one of the S different subsets contains all inliers (i.e. yields good estimate of **a**) is: $P = 1-(1- p^k)^S$.
- Fix P to 0.99 (say) and compute S assuming you know p.
- Clearly S will increase hugely if either k is large (more parameters to determine) and/or if p is small (fewer inliers).

Dealing with outliers: (3) RanSaC

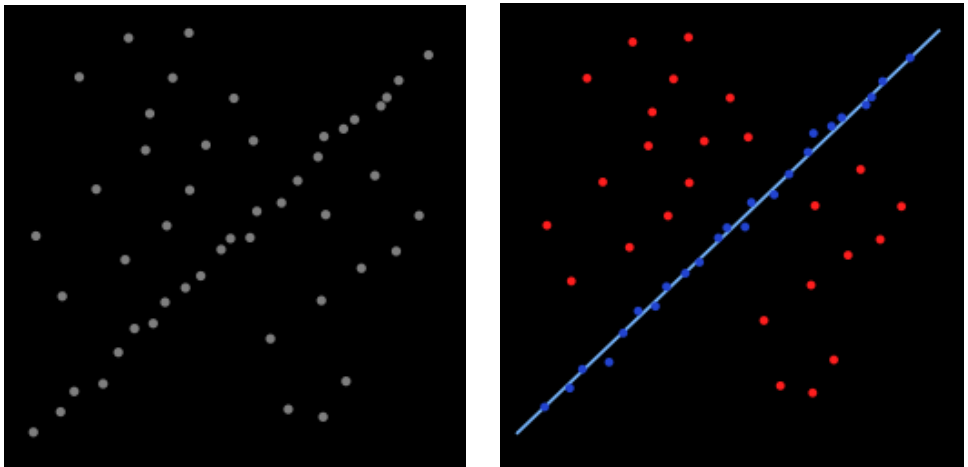
- RanSaC = **Ran**dom **Sa**mple **C**onsensus.
- Similar in spirit to LMedS.
- **Step 1:** Arbitrarily choose *k* out of *N* points where *k* is the smallest number of points required to determine **a**. Call this set of *k* points as $\mathbf{C}$.
- **Step 2:** Determine **a** using an inverse (say) from $\mathbf{C}$.
- **Step 3:** Determine the squared residual errors for all the other *N*-*k* points, i.e. compute

$$\{e_i = (y_i - f(x_i; \mathbf{a}))^2\}_{i \notin C}$$

Dealing with outliers: (3) RanSaC

- **Step 4:** Count the number of points for which e_i < threshold λ. These points form the "consensus set" for the chosen model.
- Repeat all 4 steps for multiple subsets and pick the subset which has maximum number of inliers and its corresponding estimate of **a**.
- Choice of *S* – same as LMedS.

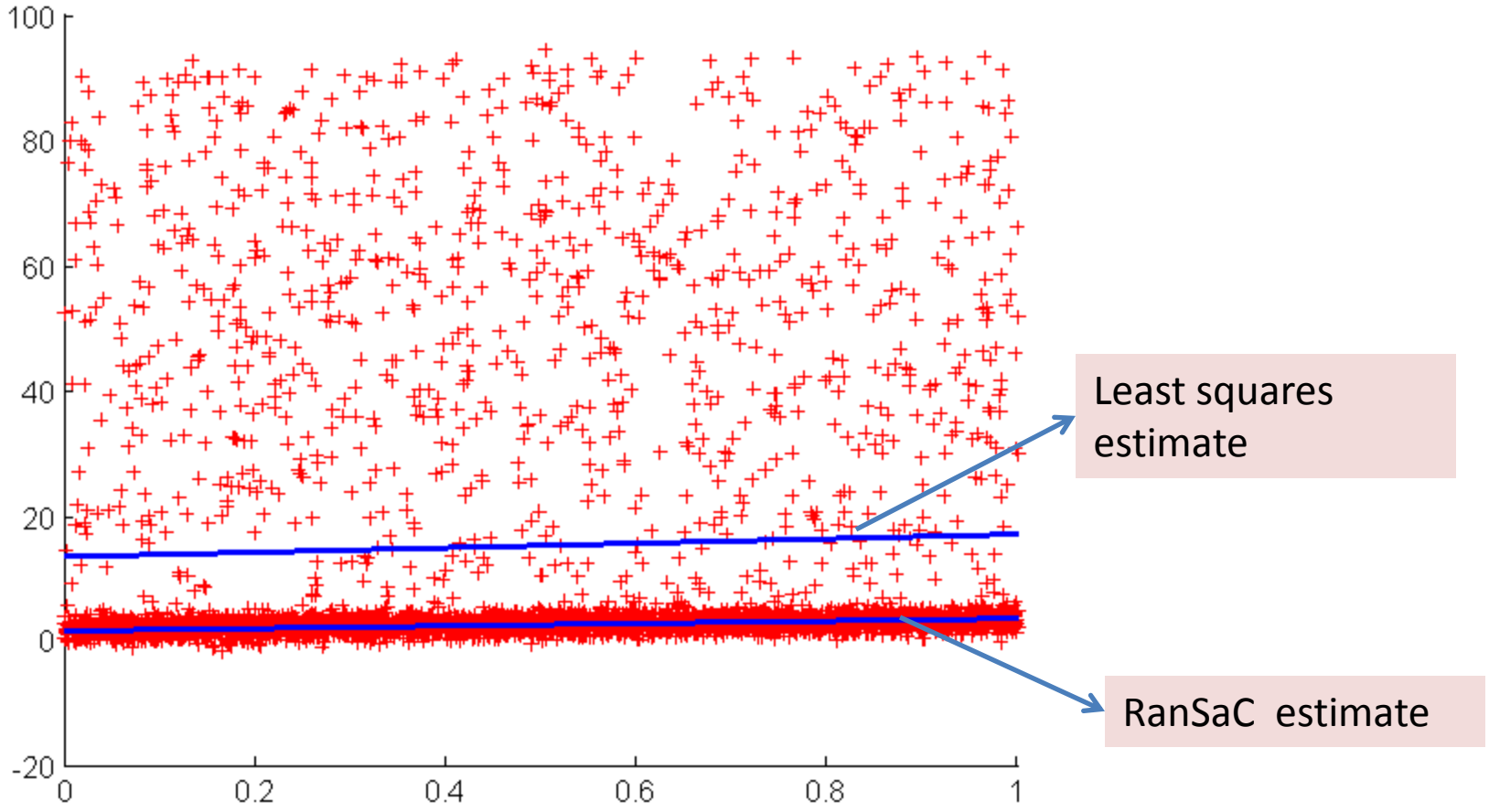
Sample result with RanSaC for line fitting.



Dealing with outliers: (3) RanSaC

- **Step 4:** Count the number of points for which e_i < threshold λ. These points form the "consensus set" for the chosen model.
- Alternatively:

1. Repeat all 4 steps for multiple subsets and pick the subset $\mathbf{C}$ which has maximum number of inliers.
2. Estimate **a** using *all the points which were inliers for* C.


100
80
60
40
20
0
-20
0
0.2
0.4
0.6
0.8
1
Least squares estimate
RanSaC estimate

RANSAC versus LMedS

- LMedS needs no threshold to determine what is an inlier unlike RanSaC.
- But RanSaC has one advantage. What?
  - LMedS will need at least 50% inliers (by definition of median).
  - RanSaC can tolerate a smaller percentage of inliers (i.e. larger percentage of outliers).

Expected number of RanSaC iterations

- The probability that at least one point in a chosen set of k points is an outlier = $1-p^k$.

- The probability that the i-th set is the *first* set that contains no outliers = $(1-p^k)^{i-1}p^k$ to be denoted as $Q(i)$.

Expected number of RanSaC iterations

- The expected number of sets to be drawn required to find the first no-outlier set =

$$= \sum_{i=1} iQ(i) = \sum_{i=1} i(1-p^k)^{i-1} p^k = p^k \sum_{i=1} i(1-p^k)^{i-1}$$

$$= p^k \sum_{i=1} \frac{d}{d(1-p^k)} (1-p^k)^i = p^k \frac{d}{d(1-p^k)} \left(\sum_{i=1} (1-p^k)^i \right)$$

$$= p^k \frac{d}{d(1-p^k)} \left(\frac{1}{1-(1-p^k)} \right) = \frac{p^k}{(p^k)^2} = p^{-k}$$

RanSaC Variant 1

- RanSaC picks the subset $\mathbf{C}$ with largest number of inliers (i.e. least number of outliers), which is equivalent to picking the subset that minimizes the following cost

$$J(C) = \sum_{i \notin C} \rho(e_i);$$

$$\rho(e_i) = 1, e_i^2 \geq T$$

$$= 0, e_i^2 < T$$

RanSaC Variant 1: MSAC

- One could instead minimize a cost function that gives weights to inliers to see how well they fit the model:

$$\hat{J}(C) = \sum_{i \notin C} \hat{\rho}(e_i);$$

$$\hat{\rho}(e_i) = T, e_i^2 \geq T$$

$$= e_i^2, e_i^2 < T$$

M-estimator: an estimator that weighs inliers by their "quality", and outliers by a fixed constant

- This variant is called MSaC (M-estimator sample consensus).

Reminder: Planar Homography

- Given two images of a coplanar scene taken from two different cameras, how will you determine the planar homography matrix **H**?
- How many point correspondences will you require?

$$\mathbf{p}_{2\,\mathrm{im}} = \begin{pmatrix} u_2 \\ v_2 \\ w_2 \end{pmatrix} = \hat{\mathbf{H}}\mathbf{p}_{1\,\mathrm{im}} = \begin{pmatrix} \hat{H}_{11} & \hat{H}_{12} & \hat{H}_{13} \\ \hat{H}_{21} & \hat{H}_{22} & \hat{H}_{23} \\ \hat{H}_{31} & \hat{H}_{32} & \hat{H}_{33} \end{pmatrix} \begin{pmatrix} u_1 \\ v_1 \\ w_1 \end{pmatrix}$$

$$x_{2,im} = \frac{\hat{H}_{11}u_1 + \hat{H}_{12}v_1 + \hat{H}_{13}w_1}{\hat{H}_{31}u_1 + \hat{H}_{32}v_1 + \hat{H}_{33}w_1} = \frac{\hat{H}_{11}x_1 + \hat{H}_{12}y_1 + \hat{H}_{13}}{\hat{H}_{31}x_1 + \hat{H}_{32}y_1 + \hat{H}_{33}}, x_{1,im} = \frac{u_1}{w_1}, y_{1,im} = \frac{v_1}{w_1}$$

$$y_{2,im} = \frac{\hat{H}_{21}u_1 + \hat{H}_{22}v_1 + \hat{H}_{23}w_1}{\hat{H}_{31}u_1 + \hat{H}_{32}v_1 + \hat{H}_{33}w_1} = \frac{\hat{H}_{21}x_1 + \hat{H}_{22}y_1 + \hat{H}_{23}}{\hat{H}_{31}x_1 + \hat{H}_{32}y_1 + \hat{H}_{33}}$$

$$x_{2i}x_{1i}\hat{H}_{31} + x_{2i}y_{1i}\hat{H}_{32} + x_{2i}\hat{H}_{33} - x_{1i}\hat{H}_{11} - y_{1i}\hat{H}_{12} - \hat{H}_{13} = 0$$

$$y_{2i}x_{1i}\hat{H}_{31} + y_{2i}y_{1i}\hat{H}_{32} + y_{2i}\hat{H}_{33} - x_{1i}\hat{H}_{21} - y_{1i}\hat{H}_{22} - \hat{H}_{23} = 0$$

$$\begin{pmatrix} -x_{1i} & -y_{1i} & -1 & 0 & 0 & 0 & x_{2i}x_{1i} & x_{2i}y_{1i} & x_{2i} \\ 0 & 0 & 0 & -x_{1i} & -y_{1i} & -1 & y_{2i}x_{1i} & y_{2i}y_{1i} & y_{2i} \end{pmatrix} \begin{pmatrix} \hat{H}_{11} \\ \hat{H}_{12} \\ \hat{H}_{13} \\ \hat{H}_{21} \\ \hat{H}_{22} \\ \hat{H}_{23} \\ \hat{H}_{31} \\ \hat{H}_{32} \\ \hat{H}_{33} \end{pmatrix} = \mathbf{0}$$

$\mathbf{Ah} = \mathbf{0}, \mathbf{A}$ has size $2N \times 9, \mathbf{h}$ has size 9×1

There will be *N* such pairs of equations (i.e. totally 2*N* equations), given *N* pairs of corresponding points in the two images

The equation **Ah** = **0** will be solved by computing the SVD of **A**, i.e. **A** = **USV**^{T}. The vector **h** will be given by the singular vector in corresponding to the null singular value (in the ideal case) or the null singular value.

Application: RANSAC to determine Homography between two Images

- Determine sets Q_1 and Q_2 of salient feature points in both images, using the SIFT algorithm.
- Q_1 and Q_2 may have different sizes! Determine the matching points between Q_1 and Q_2 using methods such as matching of SIFT descriptors.
- Many of these matches will be near-accurate, but there will be outliers too!

Figure 5.4.: Image pairs and detected points used in homography experiments (D and E). Inliers are marked as dots in left images and outliers as crosses in right images.

Application: RANSAC to determine Homography between two Images

- Pick a set of any $k = 4$ pairs of points and determine homography matrix **H** using SVD based method.
- Determine the number of inliers – i.e. those point pairs for which:

$$\left\| \mathbf{q}_{1i} - \mathbf{H}\mathbf{q}_{2i} \right\|_2^2 \leq \varepsilon$$

- Select the estimate of **H** corresponding to the set that yielded maximum number of inliers!

1st image

2nd image

1st image: warped using estimated **H**

H =

0.57882301155793 0.06780863137907 -28.33314842189324
-0.06084045669542 0.56283594396435 30.61319941910327
0.00002958152711 -0.00003144483692 0.58195535780312

RANSAC result with 41% inliers (threshold on squared distance was 0.1) – point matching done using minor-Eigenvalue method with SSD based matching of 9 x 9 windows in a 50 x 50 neighborhood

1st image

2nd image

1st image: warped using estimated **H** and overlapped/merged with 2nd image – to show accuracy of alignment

Left: Result of warping 1st image using **H** estimated with simple least-squares on the matching points (No RANSAC).
Right: Result merged with 2nd image.
Notice that the estimation is quite poor.

Some cautions with RANSAC

- Consider a dataset with a cloud of points all close to each other (degenerate set). A model created from an outlier point and any point from a degenerate set will have a large consensus set!

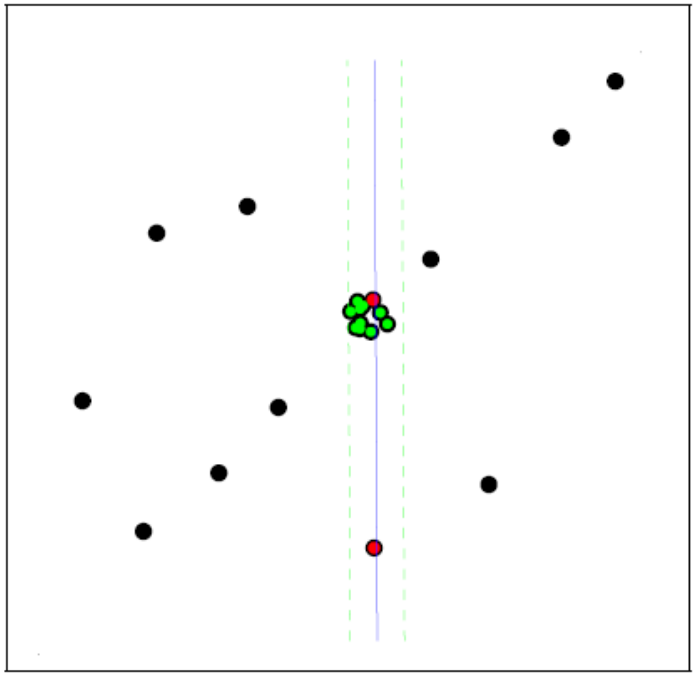


Image taken from Ph.D. thesis of Ondrej Chum, Czech Technical University, Prague

Some cautions with RANSAC

- A model created from a set of inliers need not always be optimal, i.e. it may have a very small consensus set.

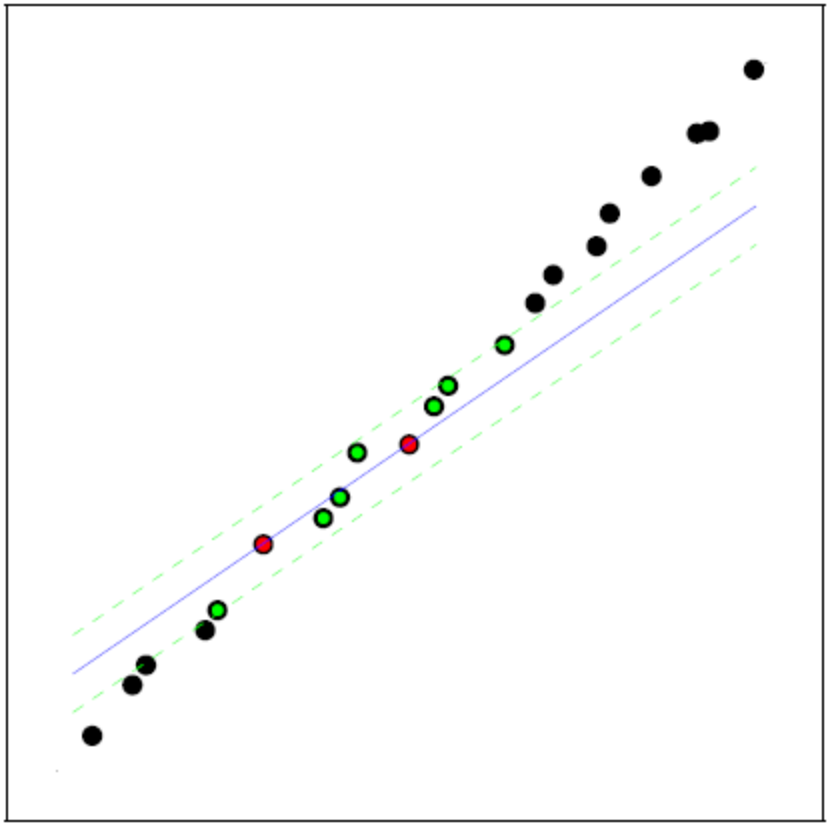


Image taken from Ph.D. thesis of Ondrej Chum, Czech Technical University, Prague

References

- Appendix A.7 of Trucco and Verri
- Article on Robust statistics by Chuck Stewart
- Original article on RanSaC by Fischler and Bolles
- Article on RanSaC variants by Torr and Zisserman