Model fitting

Schedule

#	Date	Topic
1	20.09.	Introduction & Pinhole model
2	27.09.	Feature extraction
3	04.10.	Camera models & calibration
4	11.10.	Optical flow & Particle filtering
5	18.10.	Model fitting
6	25.10.	- no lecture -
7	01.11.	Stereo matching & Multi-view
8	08.11.	Shape from X
9	15.11.	Structure-from-Motion
10	22.11.	Specific object recognition
11	29.11.	Object category recognition
12	06.12.	Image Segmentation
13	13.12.	Research Overview & Lab Tours
14	20.12.	Tracking

Overview

- Model Fitting
- Multi-Model Fitting
- Robustness
- Mixture Models
- Model Selection
- Hough Transform

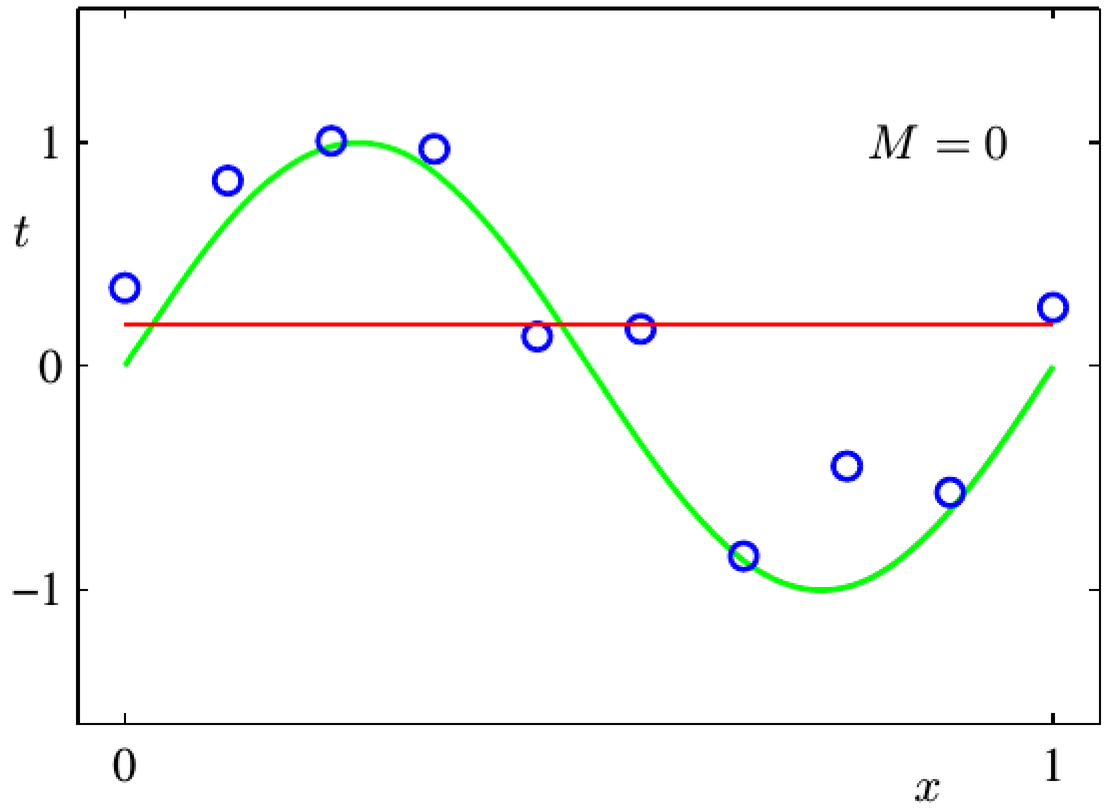
Model Fitting

Model Fitting

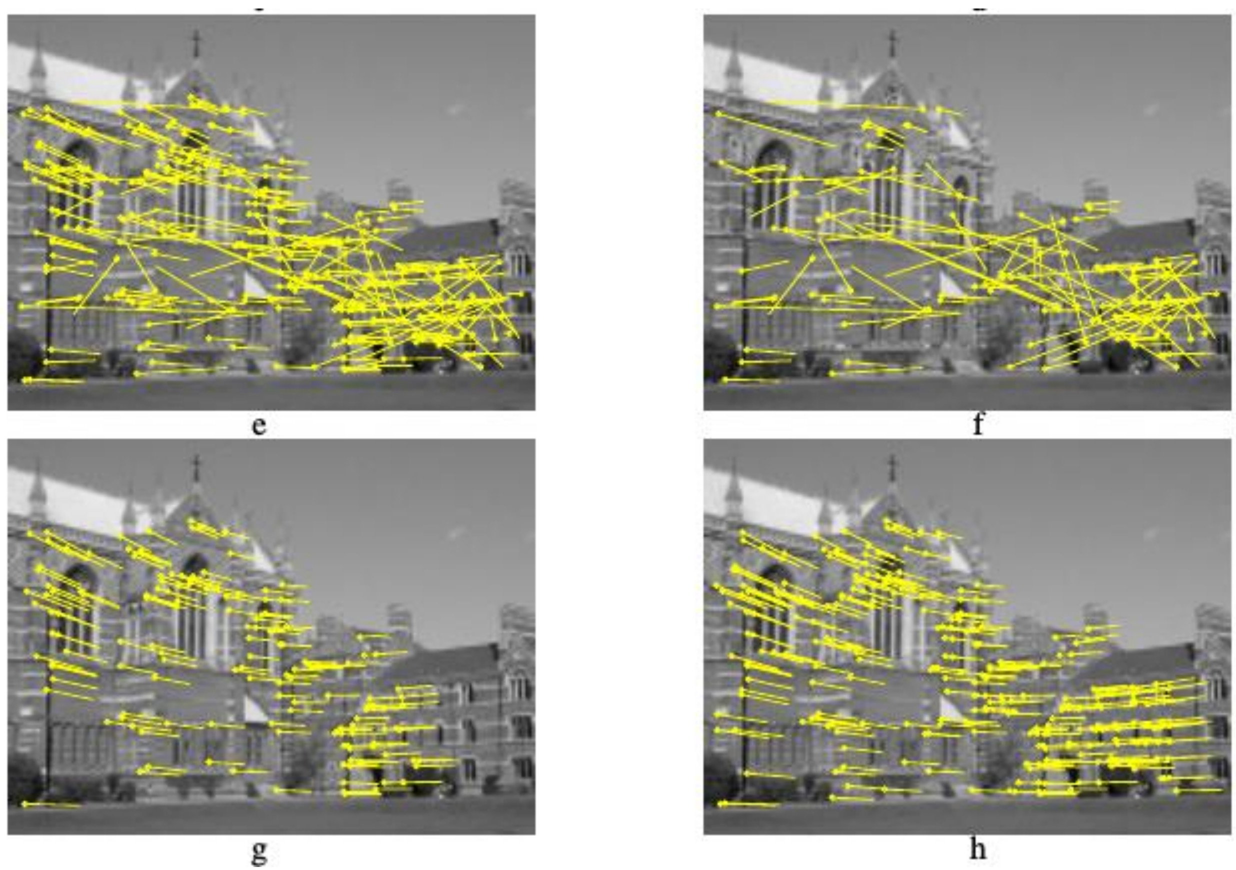
- Choose a parametric object/some objects to represent a set of tokens
- Most interesting case is when criterion is not local
  - can't tell whether a set of points lies on a line by looking only at each point and the next.
- Three main questions:
  - what object represents this set of tokens best?
  - which of several objects gets which token?
  - how many objects are there?

(you could read line for object here, or circle, or ellipse or...)

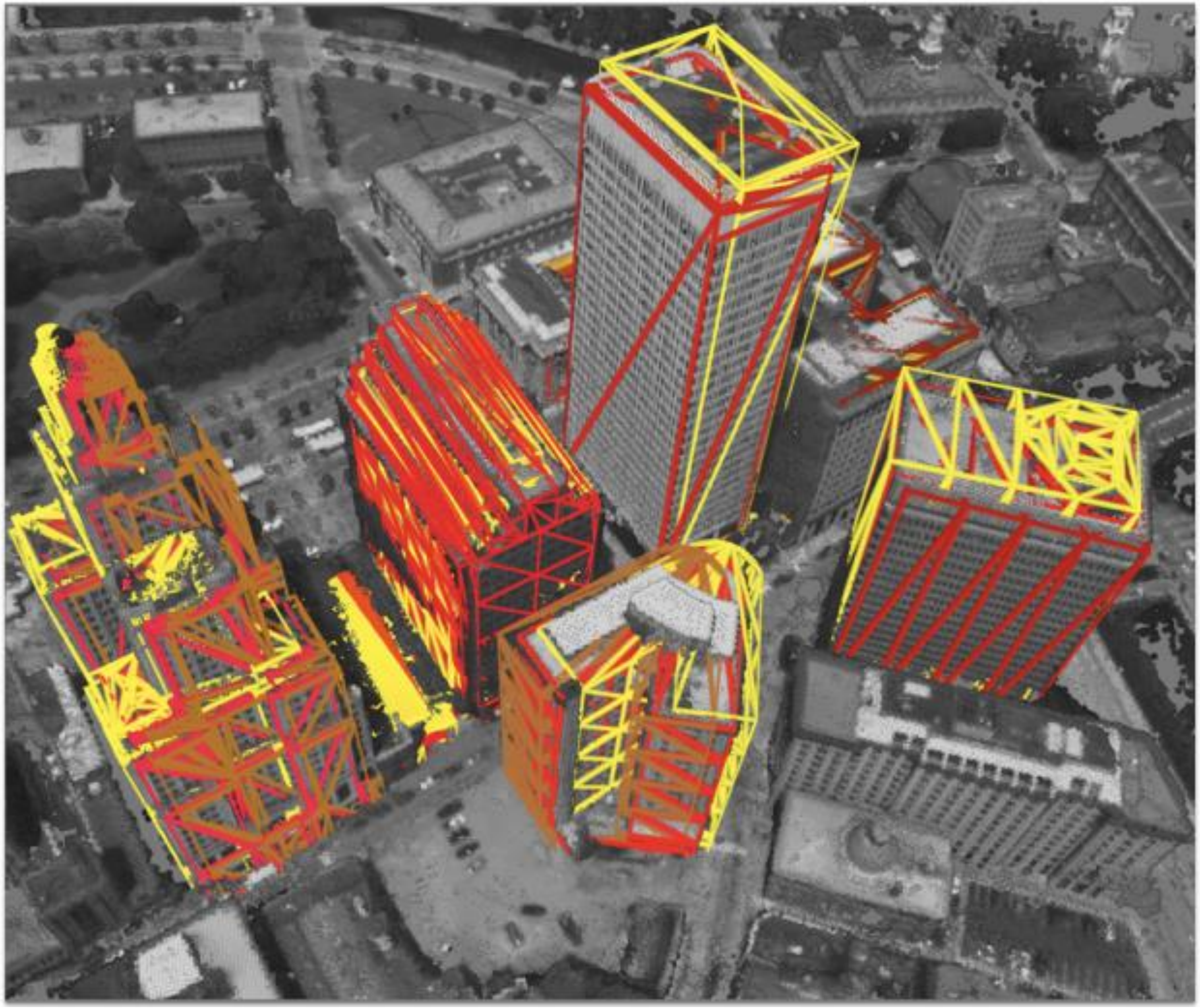
Model Fitting



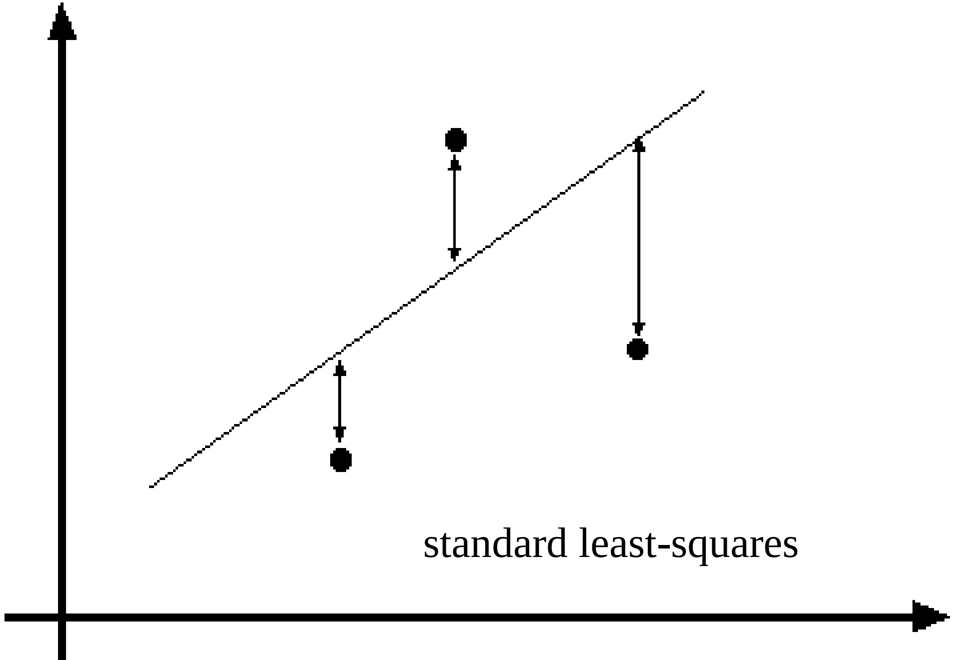
Model Fitting



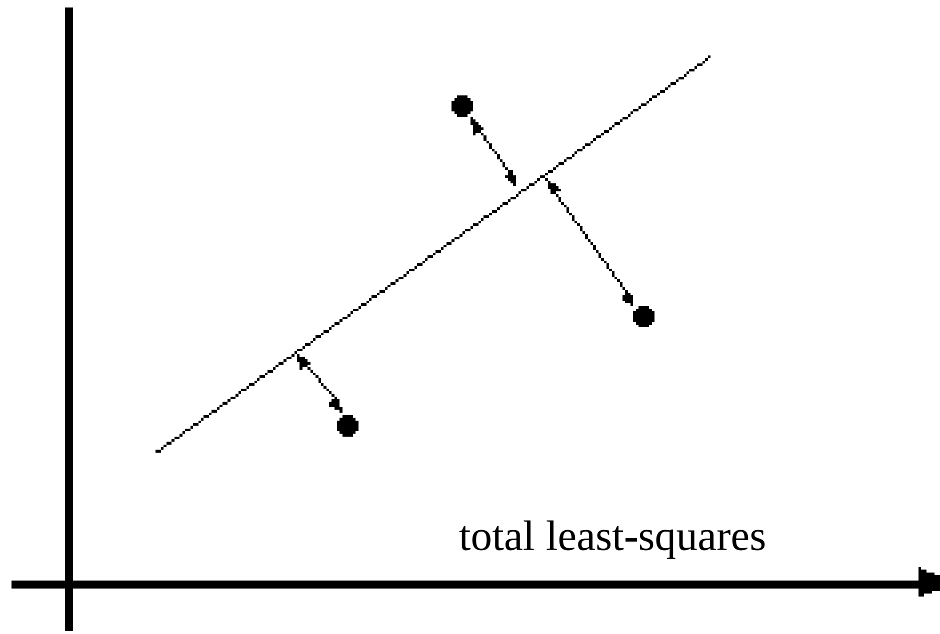
Model Fitting



Line Fitting
Blackboard



Choice of
model is important



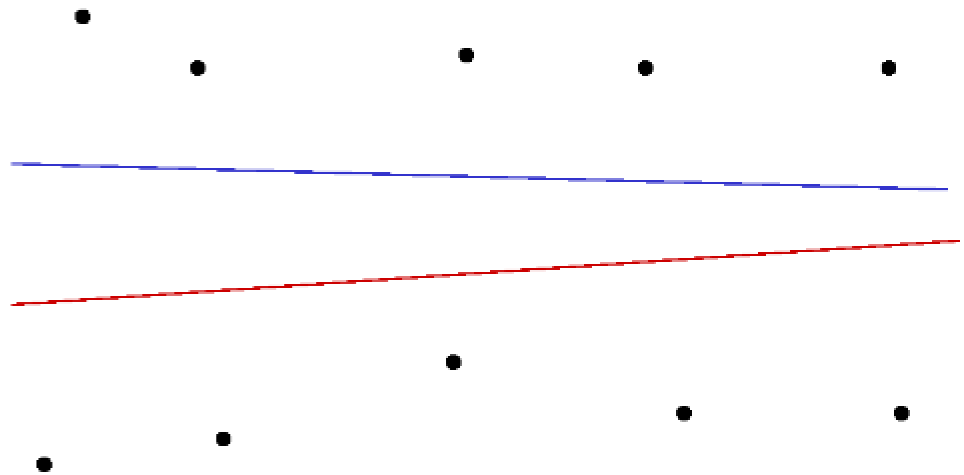
Multi-Model Fitting

Algorithm 15.2: K-means line fitting by allocating points to the closest line and then refitting.

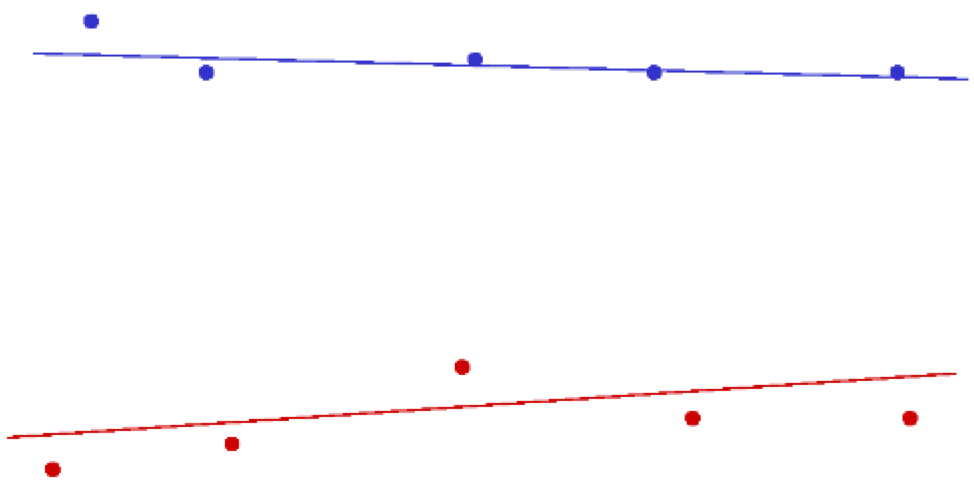
Hypothesize k lines (perhaps uniformly at random)
or
Hypothesize an assignment of lines to points
  and then fit lines using this assignment

Until convergence
  Allocate each point to the closest line
  Refit lines
end

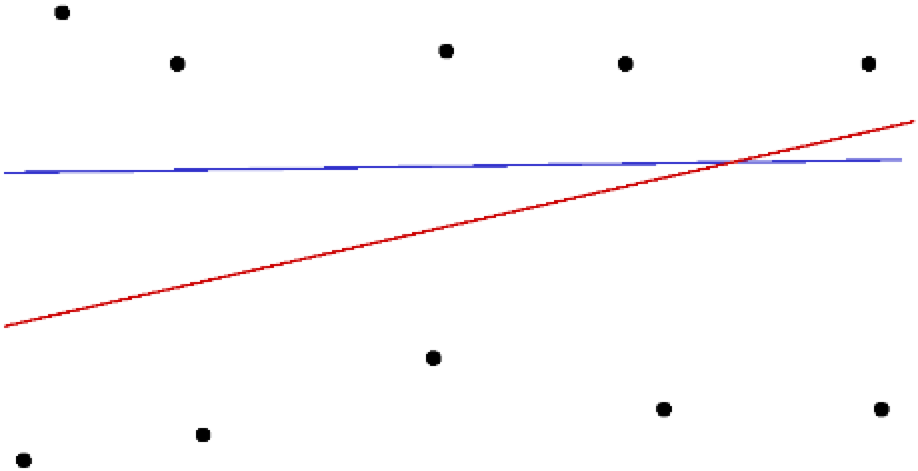
K-means line fitting



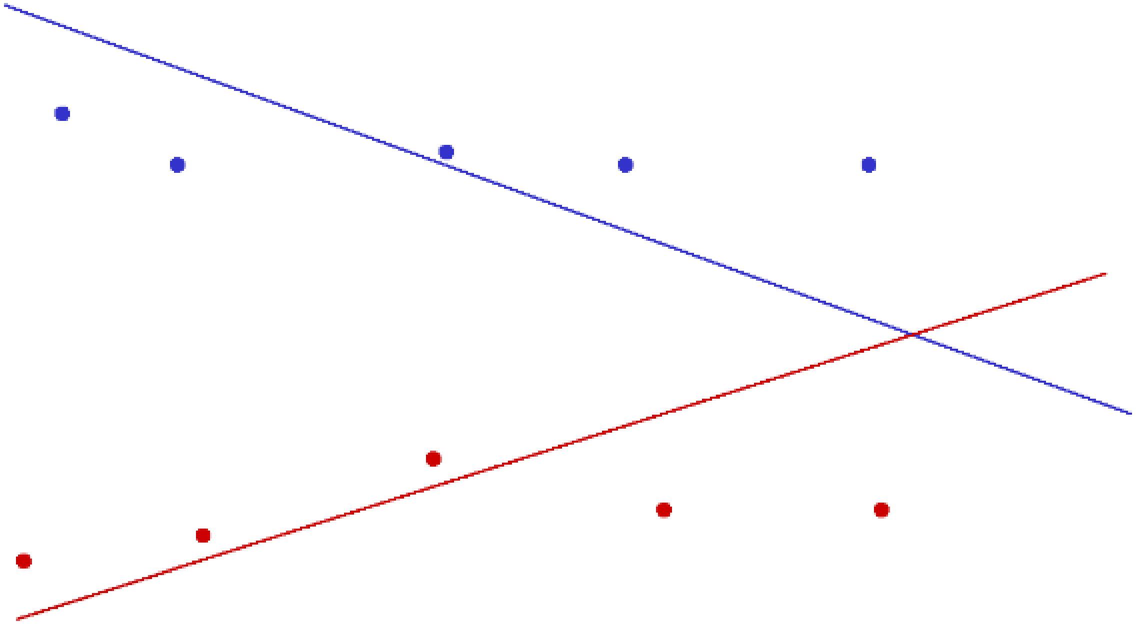
K-means line fitting



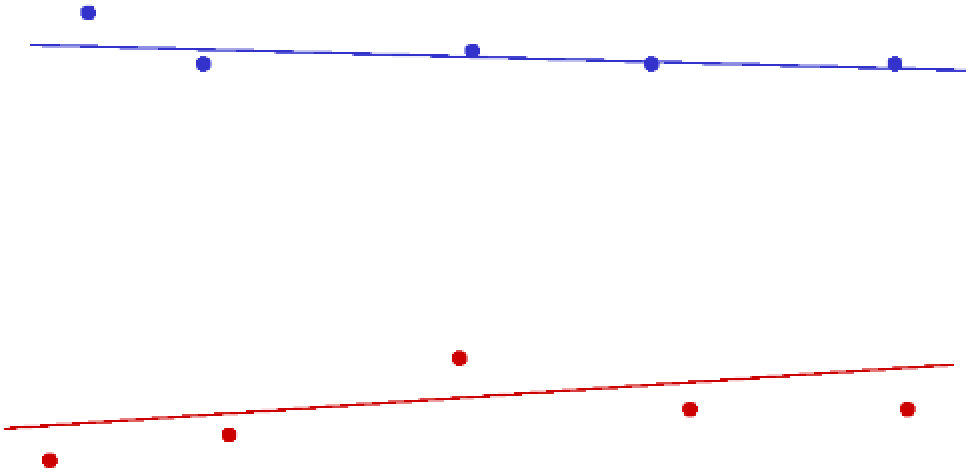
K-means line fitting



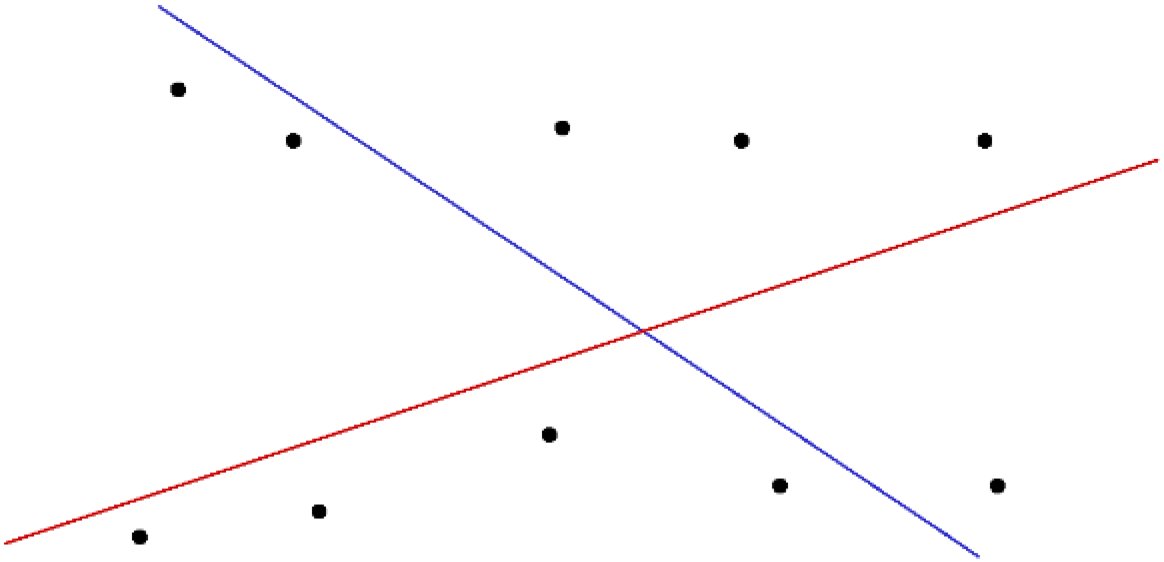
K-means line fitting



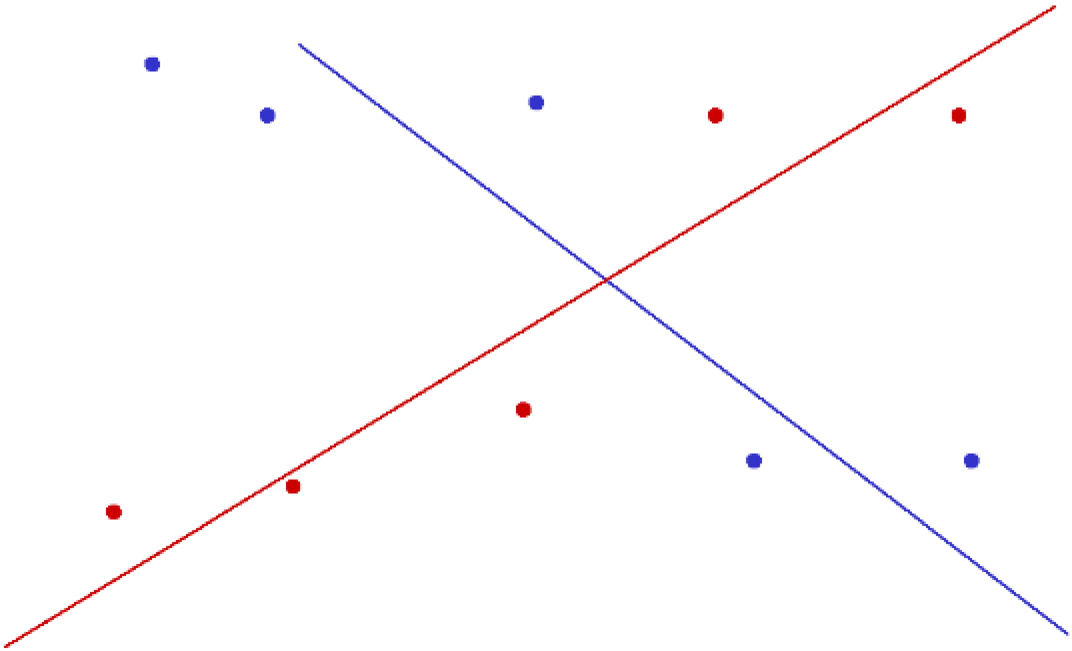
K-means line fitting



K-means line fitting

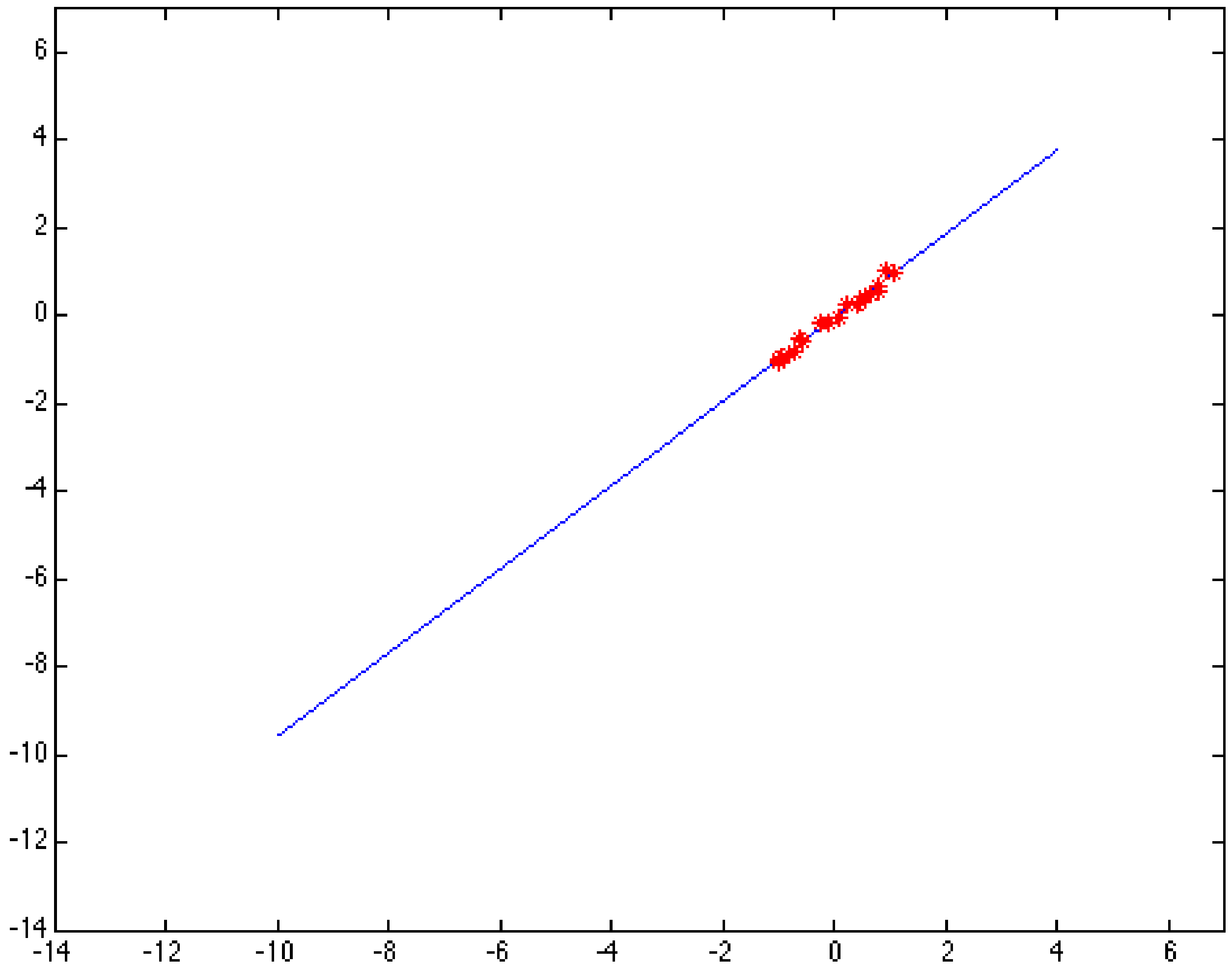


K-means line fitting

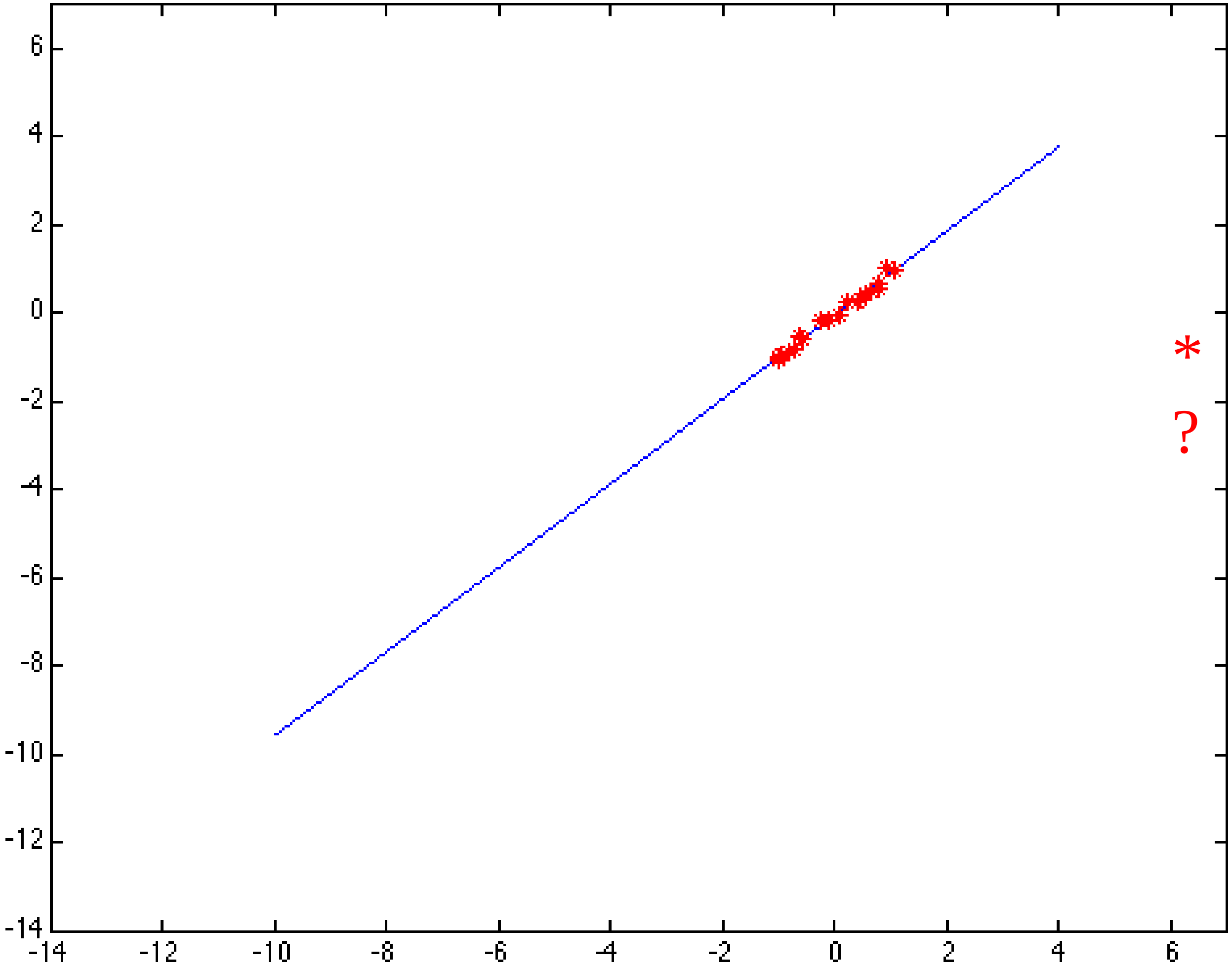


Robustness

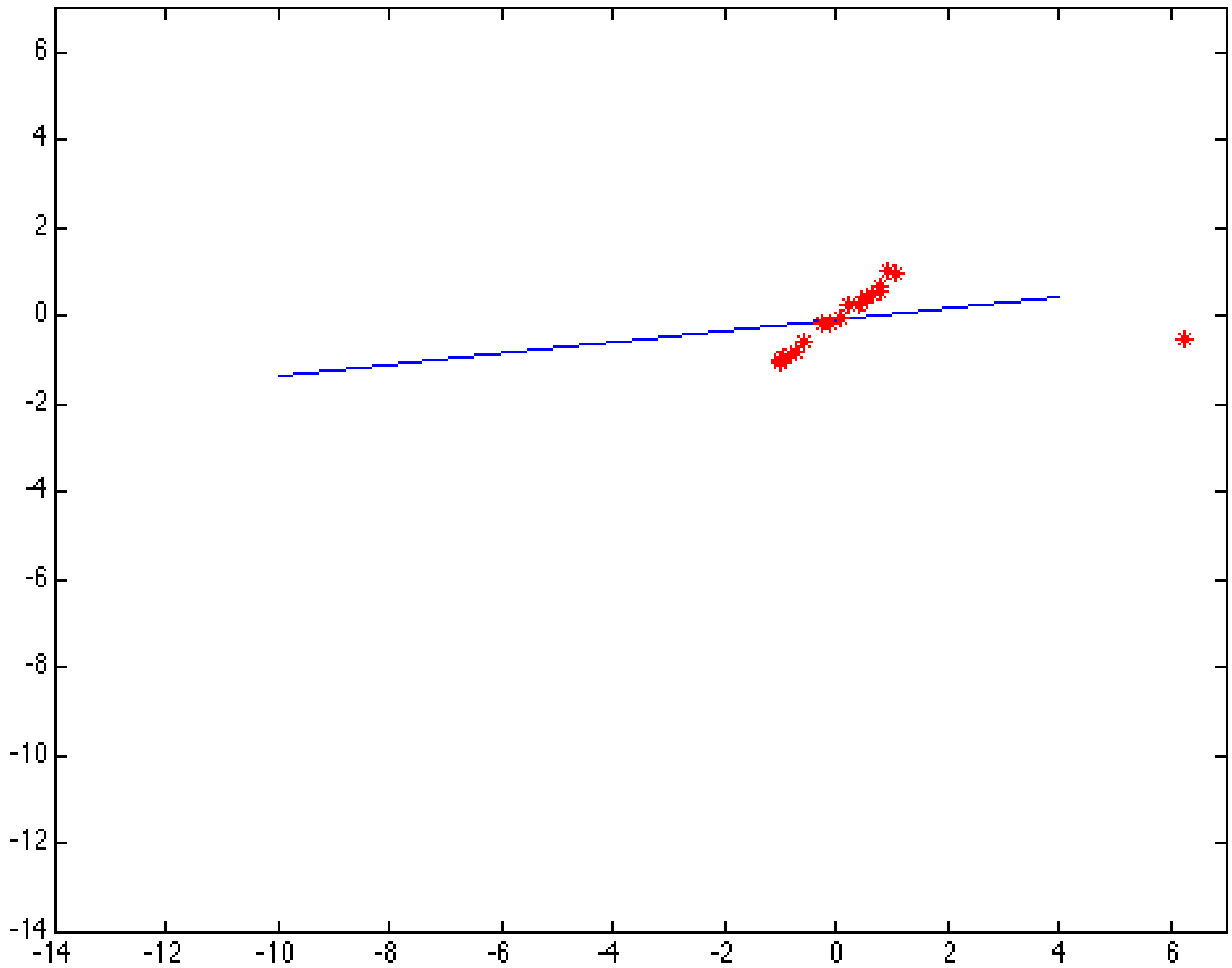
Least square fitting



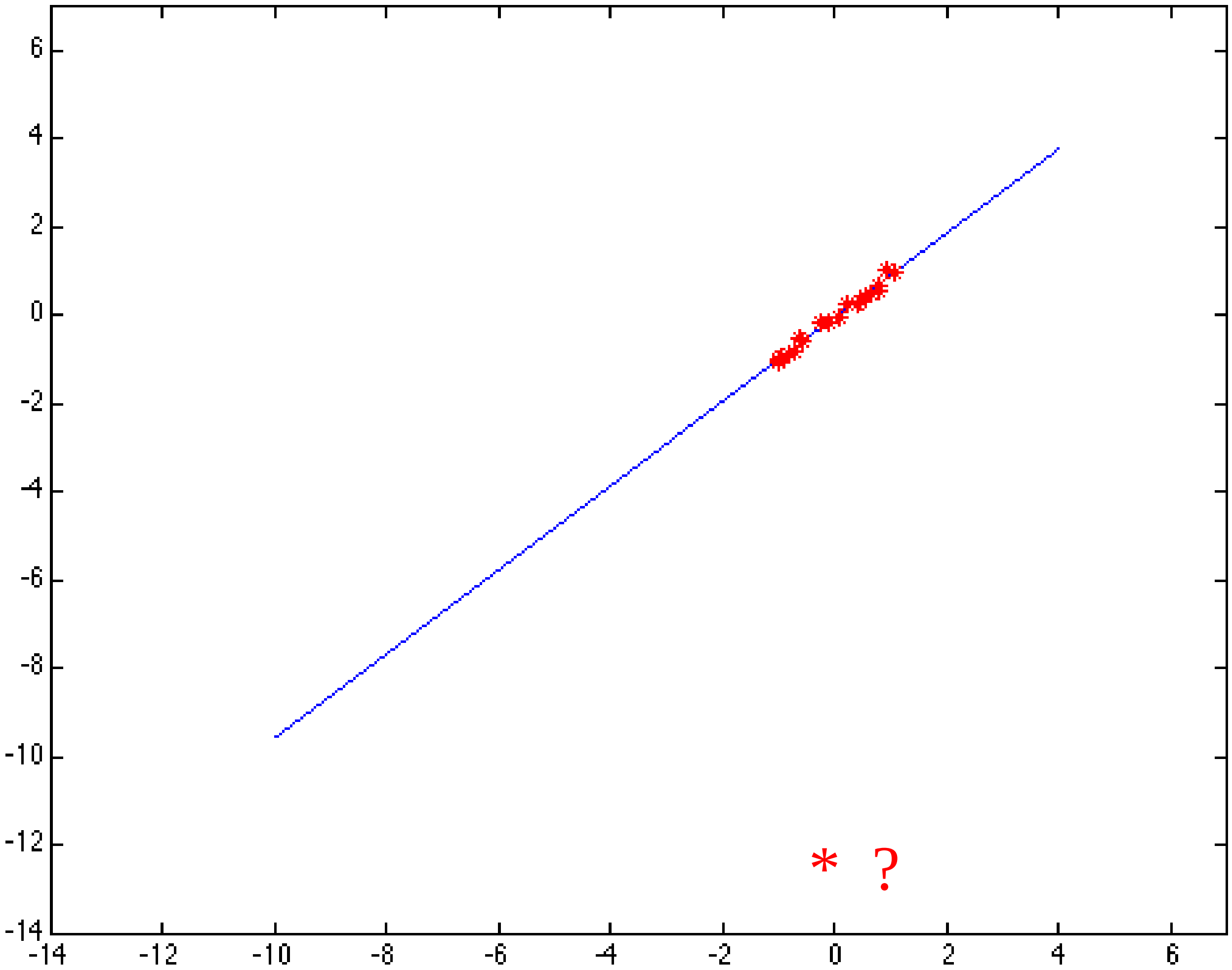
Least square fitting



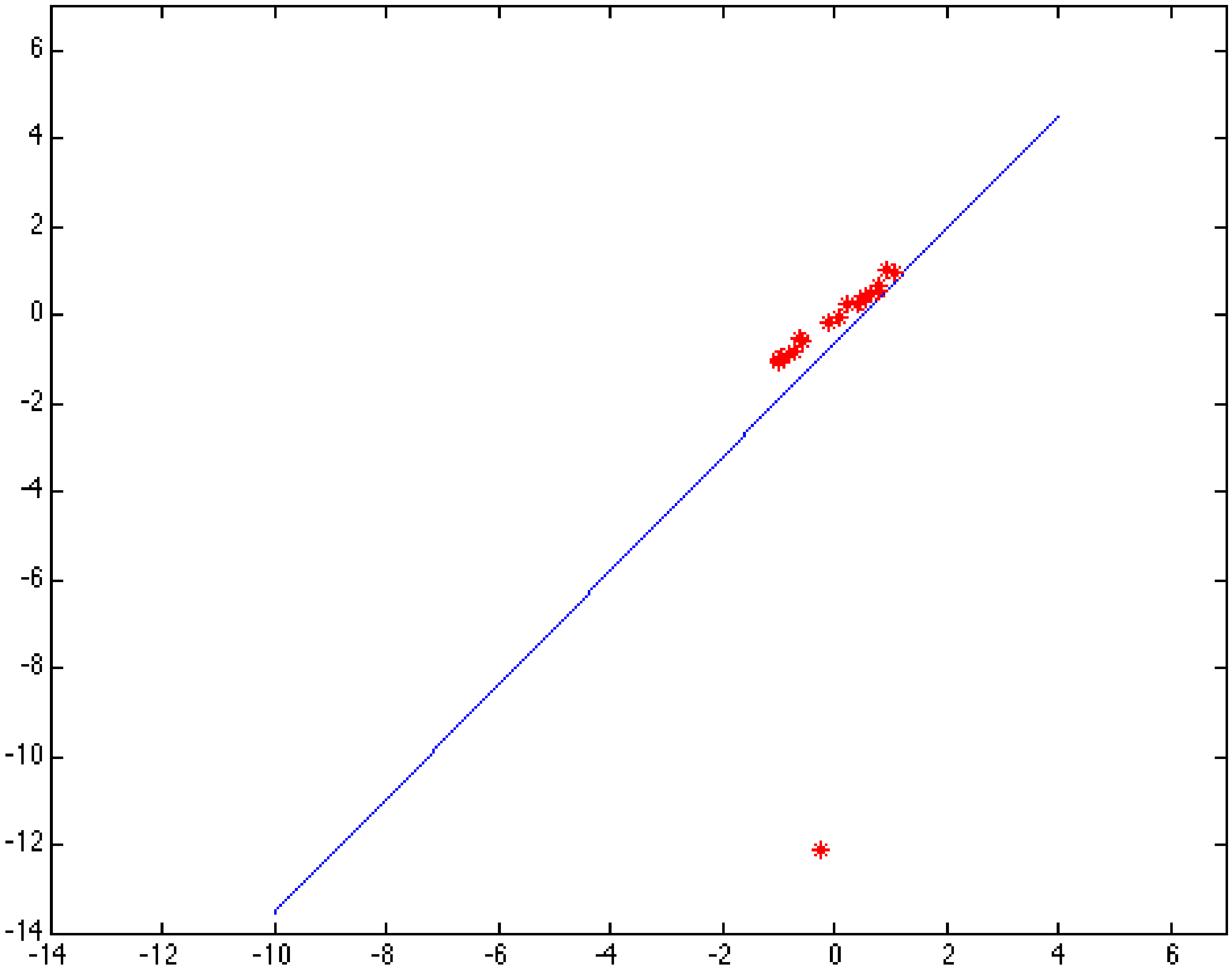
Least square fitting is sensitive to outliers

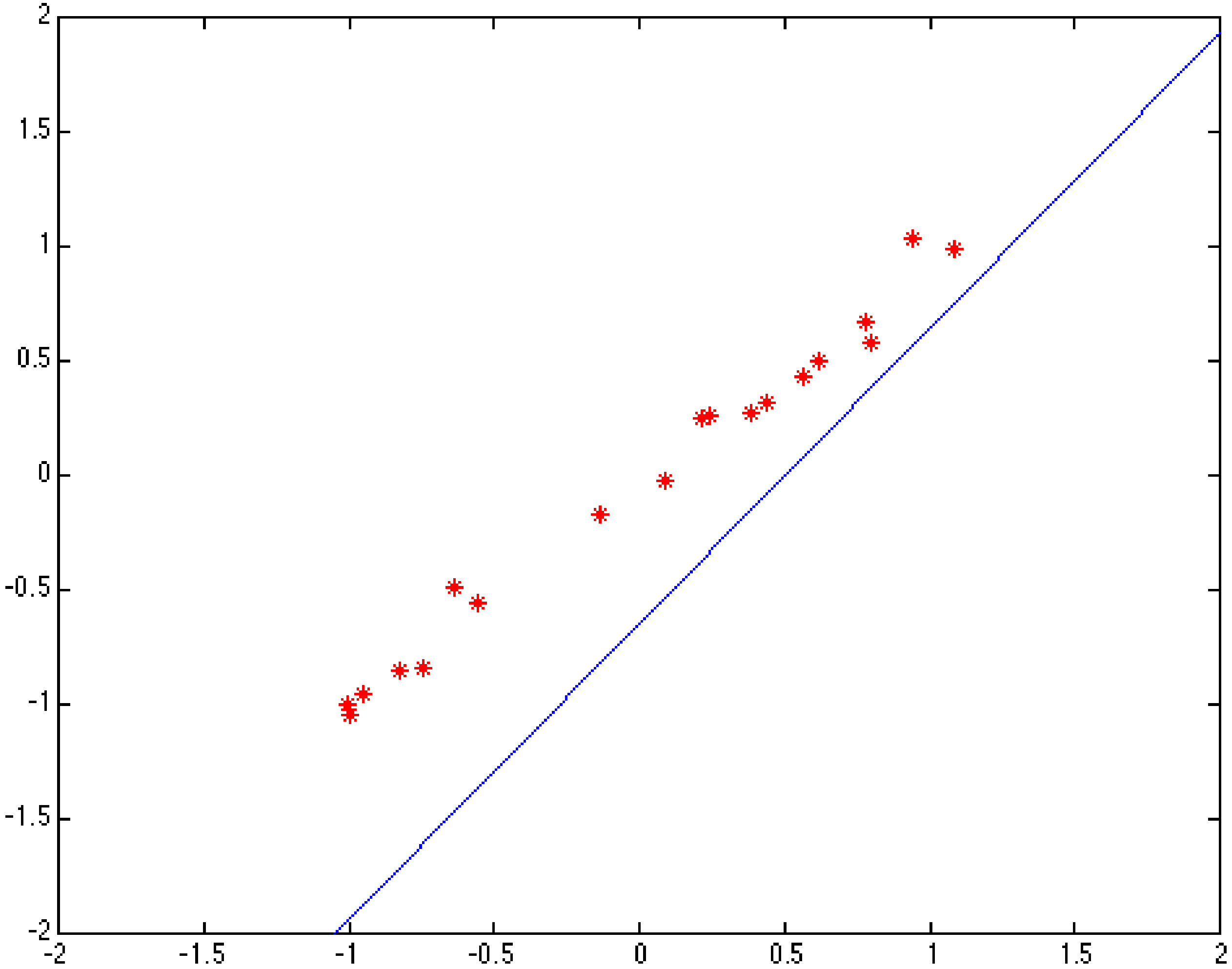


Least square fitting



Least square fitting is sensitive to outliers




2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
-1
-0.5
0
0.5
1
1.5
2

Robustness

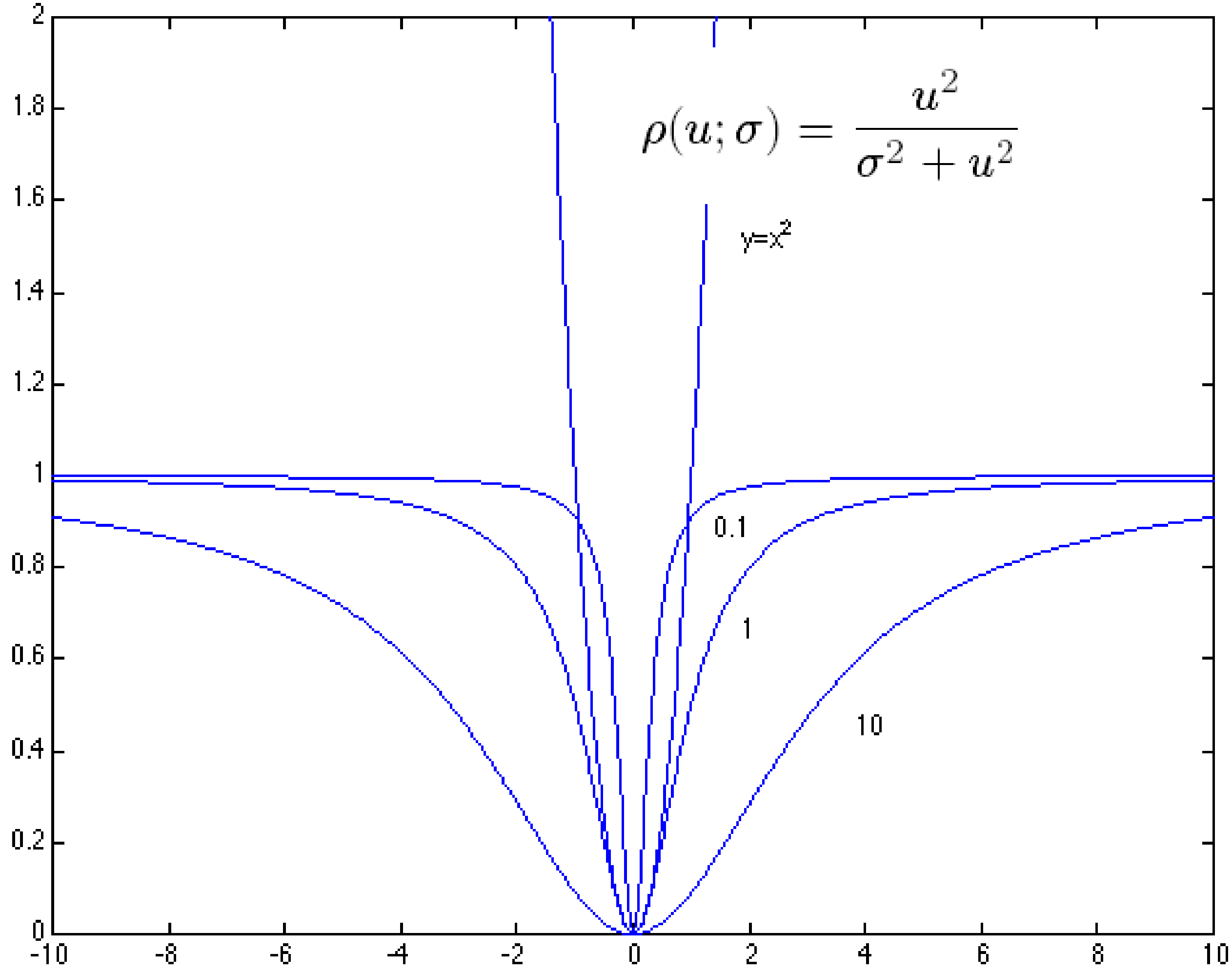
- Squared error leads to bias in the presence of outliers
  - One fix is EM
  - Another is an M-estimator
    - Square nearby, near-constant far away
  - A third is RANSAC
    - Search for good points

M-estimators

- Generally, minimize

$$\sum_i \rho\left(r_i\left(x_i,\theta\right);\sigma\right)$$

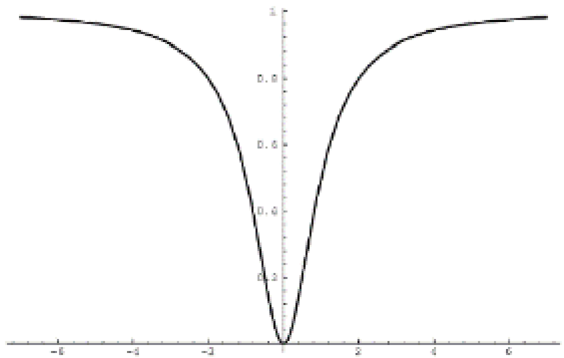
where $r_i\left(x_i,\theta\right)$ is the residual



$$\rho(u;\sigma) = \frac{u^2}{\sigma^2 + u^2}$$
y=x²
0.1
1
10
2
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-10
-8
-6
-4
-2
0
2
4
6
8
10

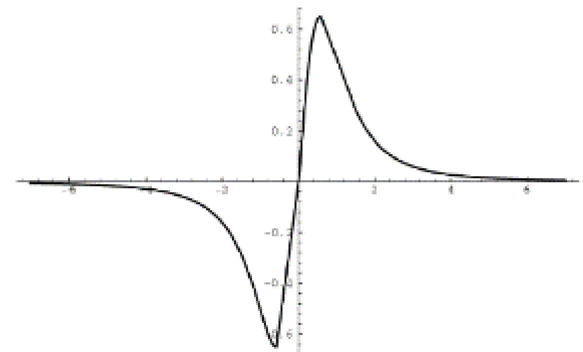
Robust Estimation

A quadratic ρ function gives too much weight to outliers
Instead, use robust norm:

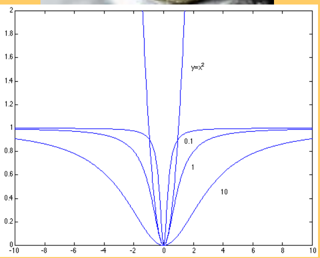


$$\rho(r,\sigma)=\frac{r^2}{\sigma^2+r^2}$$

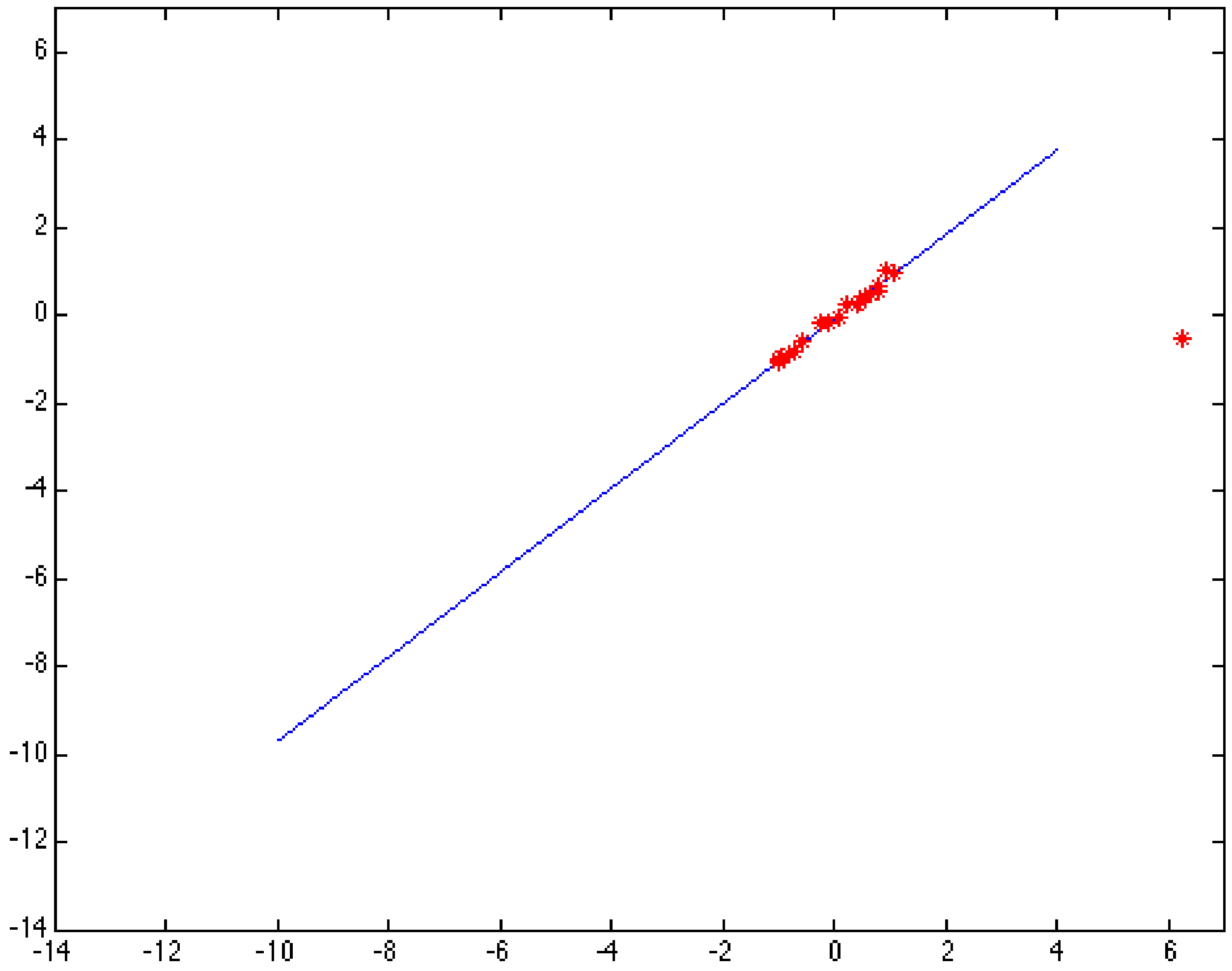
Influence function
(d/dr of norm):

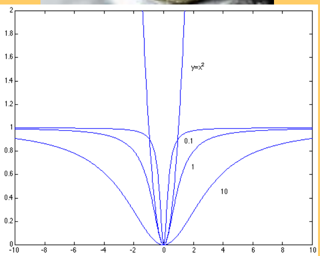


$$\psi(r,\sigma)=\frac{2r\sigma^2}{(\sigma^2+r^2)^2}$$

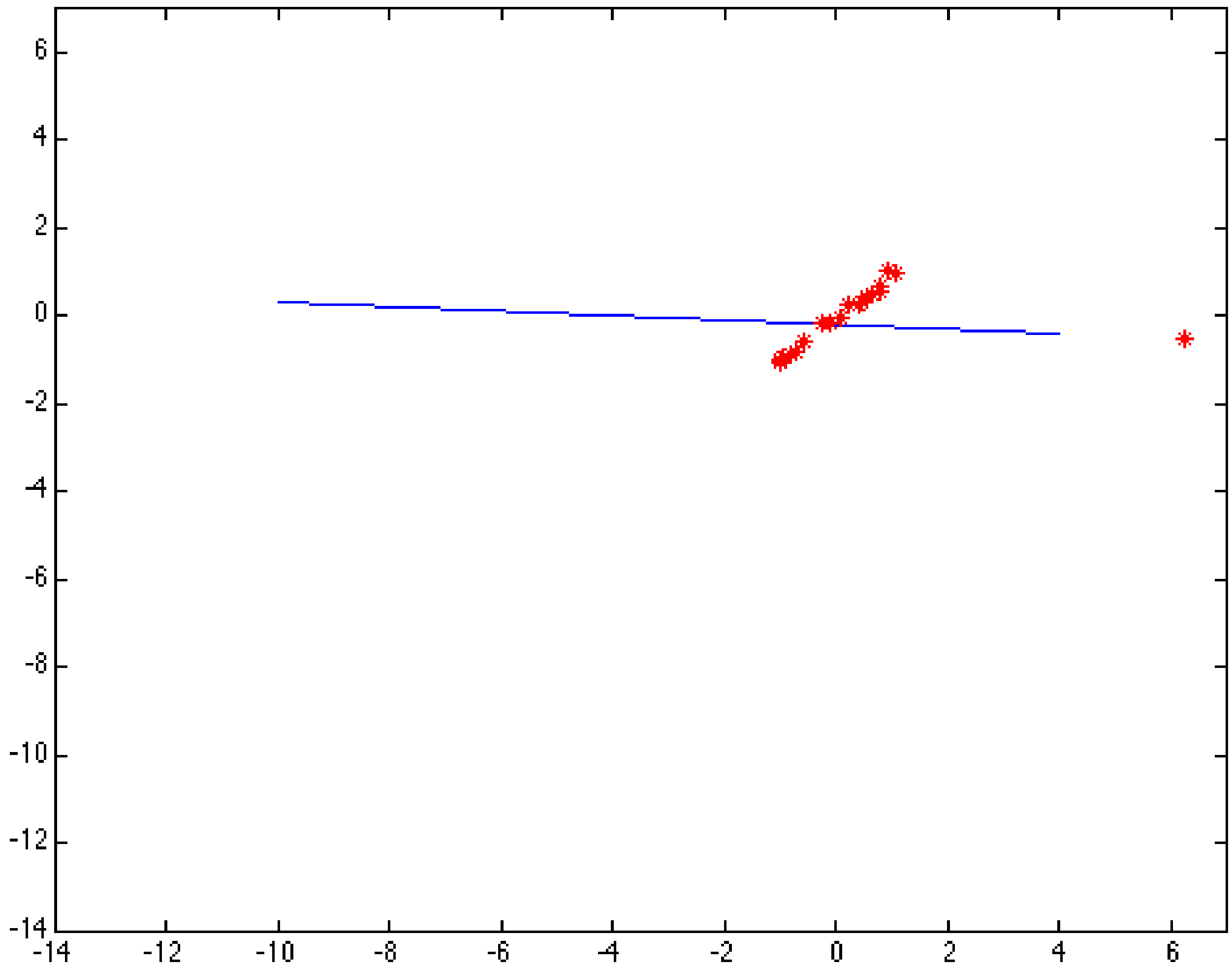


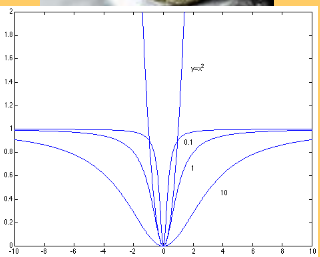
Good choice of scale σ



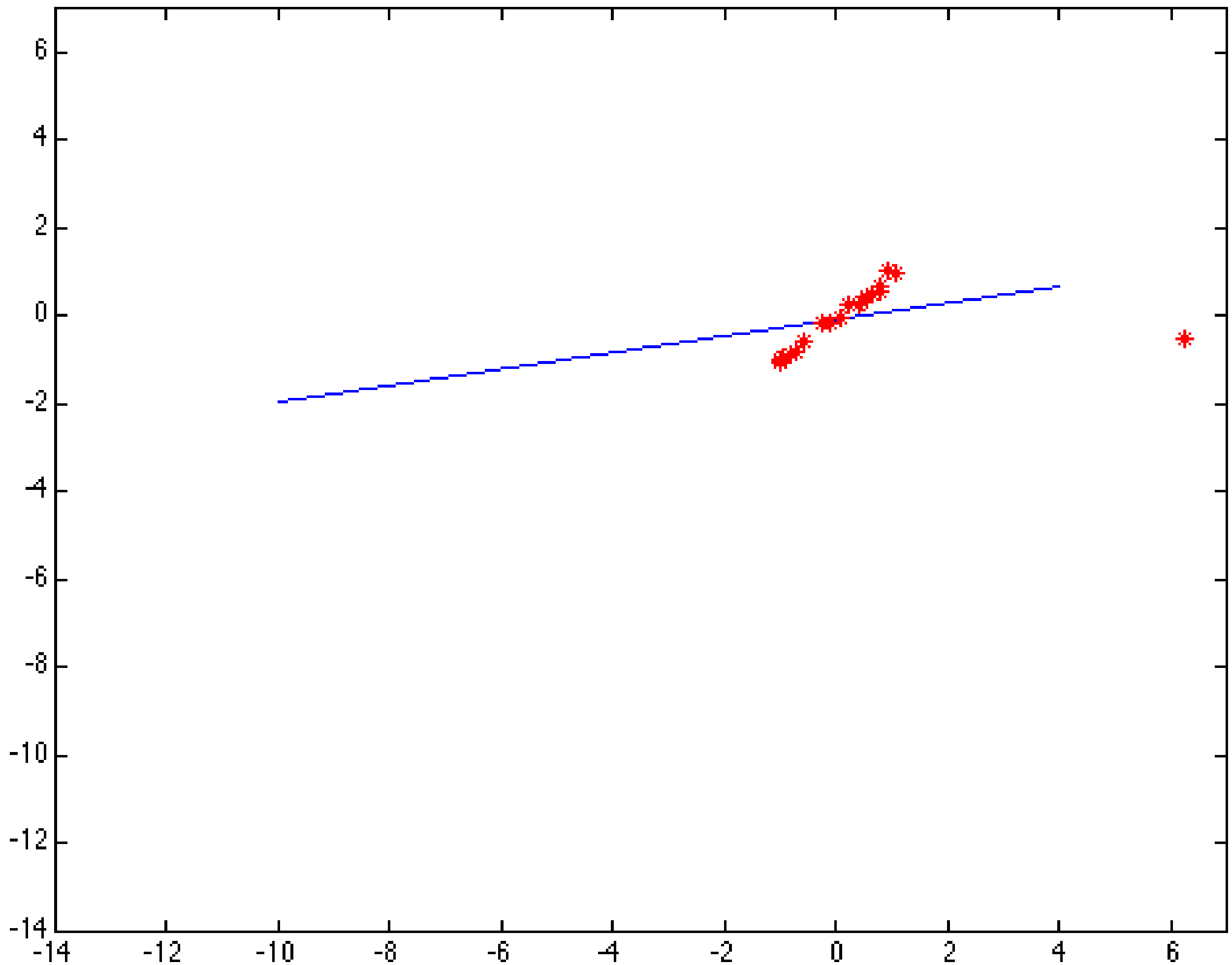


Too small





Too large



RANSAC

- Choose a small subset uniformly at random
- Fit to that
- Anything that is close to result is signal; all others are noise
- Refit
- Do this many times and choose the best

- Issues
  - How many times?
    - Often enough that we are likely to have a good line
  - How big a subset?
    - Smallest possible
  - What does close mean?
    - Depends on the problem
  - What is a good line?
    - One where the number of nearby points is so big it is unlikely to be all outliers

Objective

Robust fit of a model to a data set S which contains outliers.

Algorithm

(i) Randomly select a sample of s data points from S and instantiate the model from this subset.
(ii) Determine the set of data points S_i which are within a distance threshold t of the model. The set S_i is the consensus set of the sample and defines the inliers of S.
(iii) If the size of S_i (the number of inliers) is greater than some threshold T, re-estimate the model using all the points in S_i and terminate.
(iv) If the size of S_i is less than T, select a new subset and repeat the above.
(v) After N trials the largest consensus set S_i is selected, and the model is re-estimated using all the points in the subset S_i.

Algorithm 4.4. *The RANSAC robust estimation algorithm, adapted from [Fischler-81]. A minimum of s data points are required to instantiate the free parameters of the model. The three algorithm thresholds t, T, and N are discussed in the text.*

Distance threshold

Choose t so probability for inlier is e.g. 0.95

- Often empirically
- Zero-mean Gaussian noise σ then $d_\perp^2$ follows χ_m^2 distribution with m=codimension of model

(dimension+codimension=dimension space)

Codimension	Model	t^2
1	line,F	$3.84\sigma^2$
2	H,P	$5.99\sigma^2$
3	T	$7.81\sigma^2$

How many samples?
Blackboard

How many samples?

Choose *N* so that, with probability *p*, at least one random sample is free from outliers. e.g. *p*=0.99

$$\left(1-\left(1-e\right)^{s}\right)^{N}=1-p$$

$$N=\log\left(1-p\right)/\log\left(1-\left(1-e\right)^{s}\right)$$

	proportion of outliers e						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

Adaptively determining the number of samples

e is often unknown a priori, so pick worst case, e.g. 50%, and adapt if more inliers are found, e.g. 80% would yield e=0.2

- $N = \infty$, sample_count$= 0$.
- While $N >$ sample_count Repeat
  - Choose a sample and count the number of inliers.
  - Set $\epsilon = 1 -$ (number of inliers)/(total number of points)
  - Set N from ϵ and (4.18) with $p = 0.99$.
  - Increment the sample_count by 1.
- Terminate.

Algorithm 4.5. *Adaptive algorithm for determining the number of RANSAC samples.*

RANSAC for Homography Estimation

Objective

Compute the 2D homography between two images.

Algorithm

(i) **Interest points:** Compute interest points in each image.

(ii) **Putative correspondences:** Compute a set of interest point matches based on proximity and similarity of their intensity neighbourhood.

(iii) **RANSAC robust estimation:** Repeat for N samples, where N is determined adaptively as in algorithm 4.5:

- (a) Select a random sample of 4 correspondences and compute the homography $\mathtt{H}$.
- (b) Calculate the distance $d_\perp$ for each putative correspondence.
- (c) Compute the number of inliers consistent with $\mathtt{H}$ by the number of correspondences for which $d_\perp < t = \sqrt{5.99}\,\sigma$ pixels.

Choose the $\mathtt{H}$ with the largest number of inliers. In the case of ties choose the solution that has the lowest standard deviation of inliers.

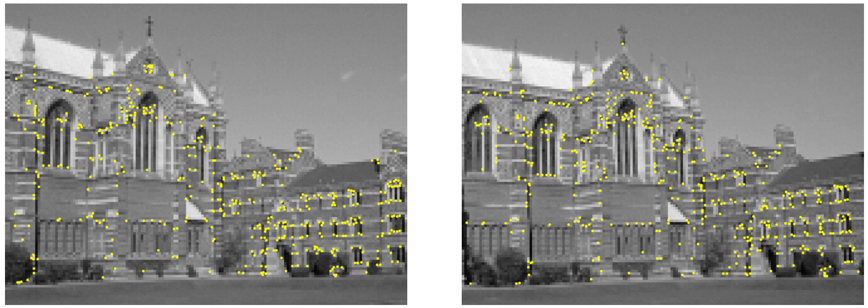
(iv) **Optimal estimation:** re-estimate $\mathtt{H}$ from all correspondences classified as inliers, by minimizing the ML cost function (4.8–*p*95) using the Levenberg–Marquardt algorithm of section A6.2(*p*600).

(v) **Guided matching:** Further interest point correspondences are now determined using the estimated $\mathtt{H}$ to define a search region about the transferred point position.

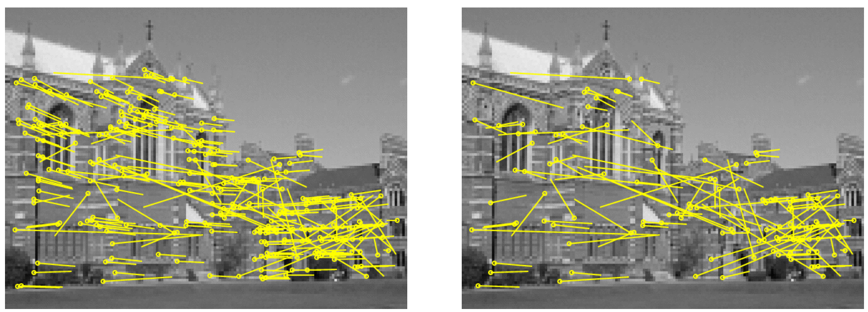
The last two steps can be iterated until the number of correspondences is stable.

Algorithm 4.6. *Automatic estimation of a homography between two images using RANSAC.*

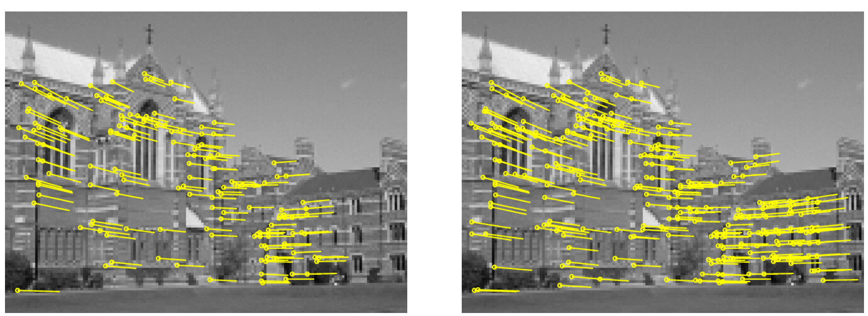
Example: robust computation



Interest points
(500/image)
(640x480)



Correspondences: 268
Outliers: 117
Inliers: 151



Final Result
after **guided matching**
and MLE

More on robust estimation

- Least Median of Squares (LMedS), an alternative to RANSAC

  (minimize Median residual instead of maximizing inlier count)

- Enhancements to RANSAC
  - Randomized RANSAC (pre-test on subset)
  - Sample 'good' matches more frequently
  - …

  e.g. http://www.cs.unc.edu/~rraguram/research.html

- RANSAC is also somewhat robust to bugs, sometimes it just takes a bit longer…

Mixture Models

Missing variable problems

- If some variables were known the inference problem would be easy for many vision tasks:
  - **robust fitting:** if we knew which token is an outlier, robust fitting would be easy
  - **multi-object fitting:** if we knew which line each token came from, it would be easy to determine the line parameters
  - **segmentation:** if we knew the segment each pixel came from, it would be easy to determine the segment parameters
  - **fundamental matrix estimation:** if we knew which feature corresponded to which, it would be easy to determine the fundamental matrix

Missing variable problems

- **Algorithm:**
  - estimate values for the missing variables
  - estimate parameters given missing variables
  - re-estimate missing variables, continue
- **Example:**
  - associate points to lines
  - now fit the lines to these points
  - iterate ..
  - We have seen this algorithm before!

Soft Assignments: Expectation Maximization (EM)

- Replace missing variables with **expected values**, given fixed values of parameters
- Fix missing variables, choose parameters to maximize likelihood

Blackboard

Lines and robustness

- We have one line and n points
- Some come from the line, some from “noise”
- This is a mixture model:

- We wish to determine
  - line parameters
  - P(comes from line)

$$
\begin{aligned}
&P(\text{point} \mid \text{line and noise params}) \\
&\quad = P(\text{point} \mid \text{line})P(\text{comes from line}) + \\
&\quad\ \ P(\text{point} \mid \text{noise})P(\text{comes from noise}) \\
&\quad = P(\text{point} \mid \text{line})\lambda + P(\text{point} \mid \text{noise})(1-\lambda)
\end{aligned}
$$

Estimating the mixture model

- Introduce a set of hidden variables, δ, one for each point. They are 1 when the point is on the line, and 0 when off.
- If these are known, the negative log-likelihood becomes (the line's parameters are ϕ, c):
- Here K is a normalising constant, k_n is the noise intensity (we'll choose this later).

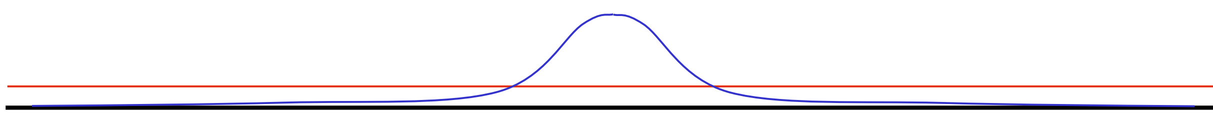
$$Q_c(x;\theta) = \sum_i \begin{pmatrix} \delta_i \left(\dfrac{(x_i \cos\phi + y_i \sin\phi + c)^2}{2\sigma^2} \right) + \\ (1-\delta_i) k_n \end{pmatrix} + K$$

Substituting for delta

- We shall substitute the expected value of δ, for a given θ
- recall $\theta=(\phi, c, \lambda)$
- E(δ_i)=1. P(δ_i=1|θ)+0....

$$P(\delta_i = 1 \mid \theta, \mathbf{x}_i) = \frac{P(\mathbf{x}_i \mid \delta_i = 1, \theta) P(\delta_i = 1)}{P(\mathbf{x}_i \mid \delta_i = 1, \theta) P(\delta_i = 1) + P(\mathbf{x}_i \mid \delta_i = 0, \theta) P(\delta_i = 0)}$$

$$= \frac{\exp\left(-\frac{1}{2\sigma^2}\left[x_i \cos\phi + y_i \sin\varphi + c\right]^2\right)\lambda}{\boxed{\exp\left(-\frac{1}{2\sigma^2}\left[x_i \cos\phi + y_i \sin\varphi + c\right]^2\right)\lambda} + \boxed{\exp(-k_n)(1-\lambda)}}$$



Algorithm for line fitting

- Obtain some start point

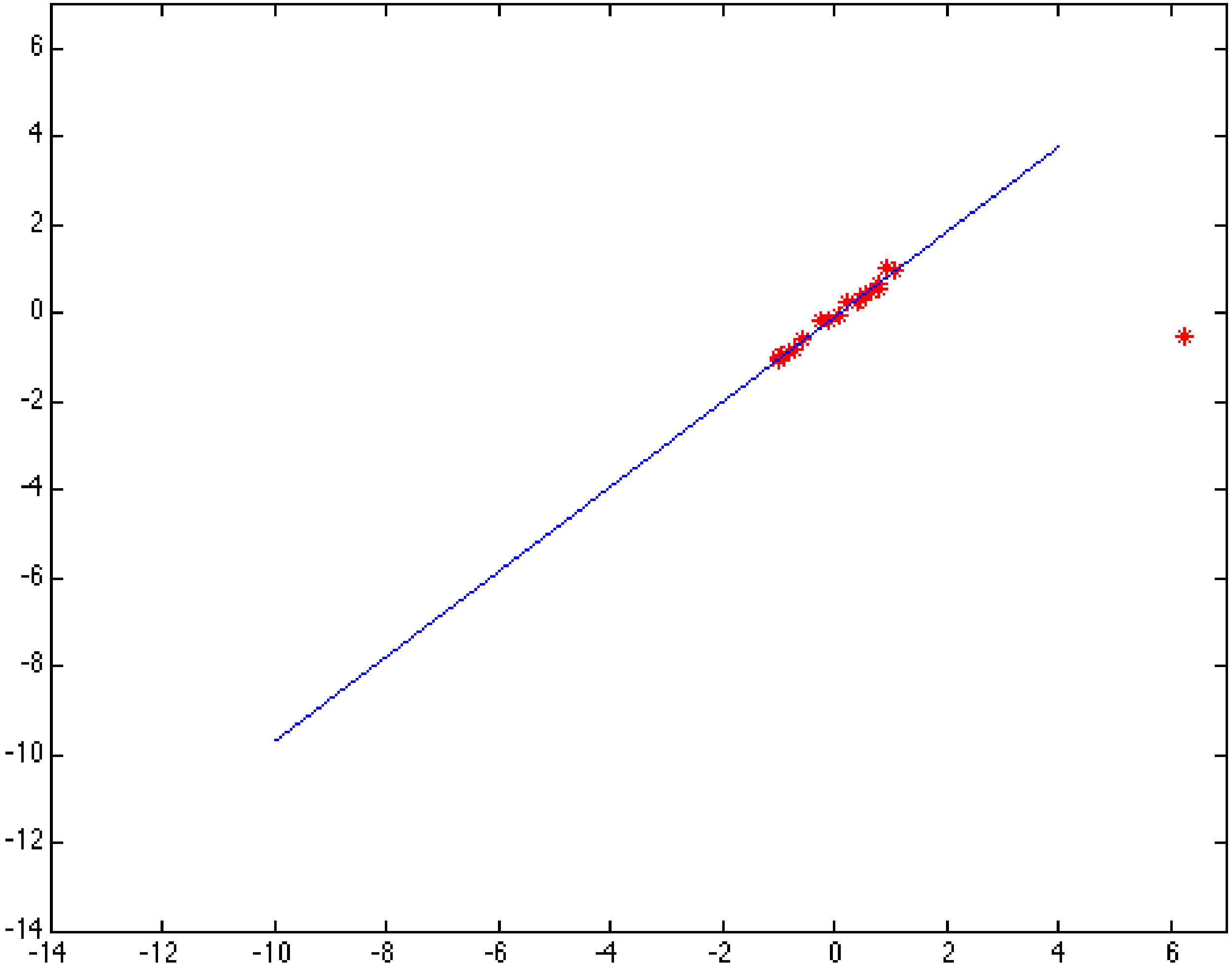
$$\theta^{(0)} = \left(\phi^{(0)}, c^{(0)}, \lambda^{(0)}\right)$$

- Compute δ's using formula above

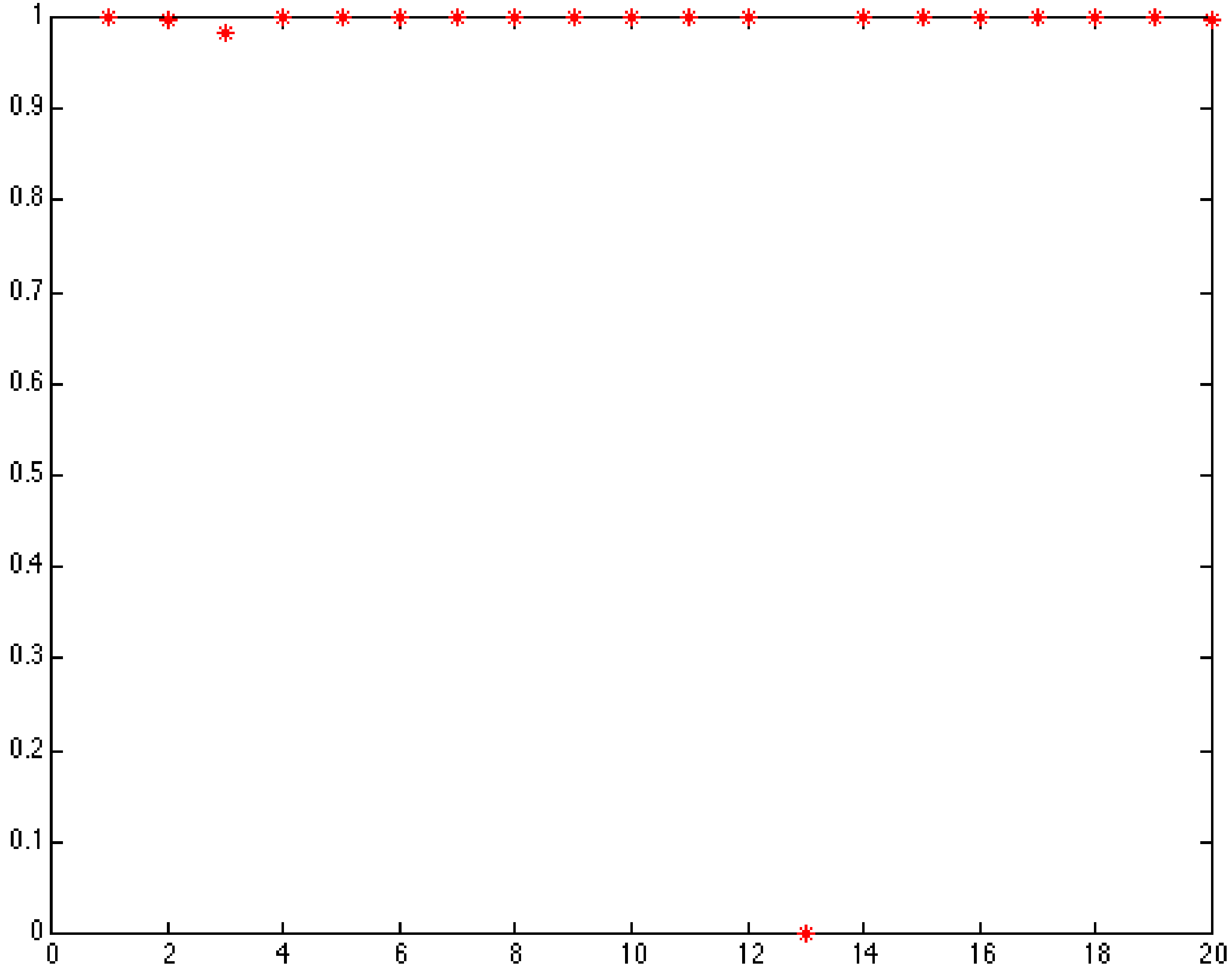
- Now compute maximum likelihood estimate of $\theta^{(1)}$

  - ϕ, c come from fitting to weighted points
  - λ comes by counting

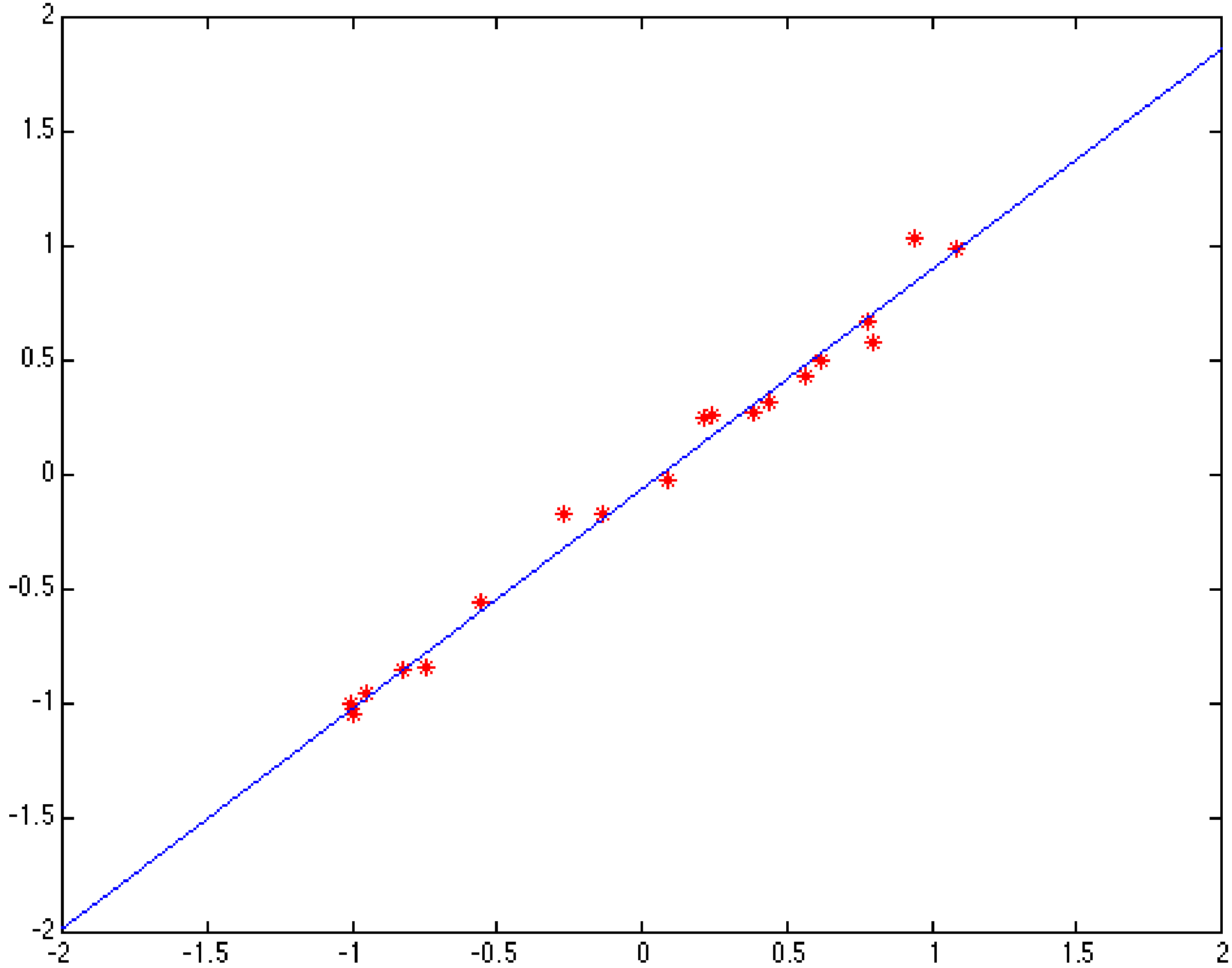
- Iterate to convergence


6
4
2
0
-2
-4
-6
-8
-10
-12
-14
-14
-12
-10
-8
-6
-4
-2
0
2
4
6

The expected values of the deltas at the maximum:



Closeup of the fit



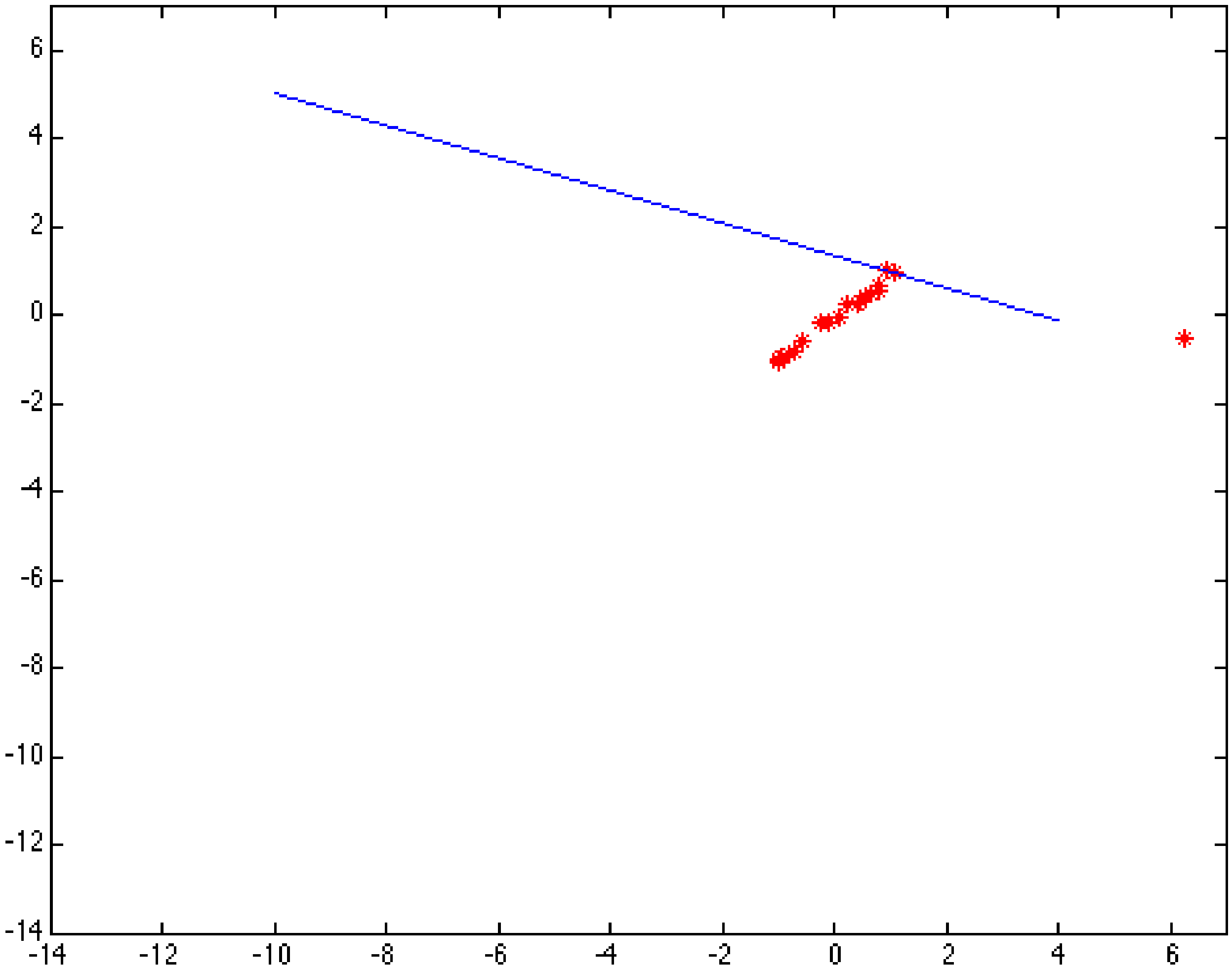
Choosing parameters

- What about the noise parameter, and the sigma for the line?
  - several methods
    - first principles (seldom really possible)
    - guessing (precise choice often doesn't matter much)
  - if k_n is large, all points considered inliers
    - biases the fit if outliers present
    - rule of thumb: its better to fit to the better fitting points - if this is hard to do, then the model could be a problem

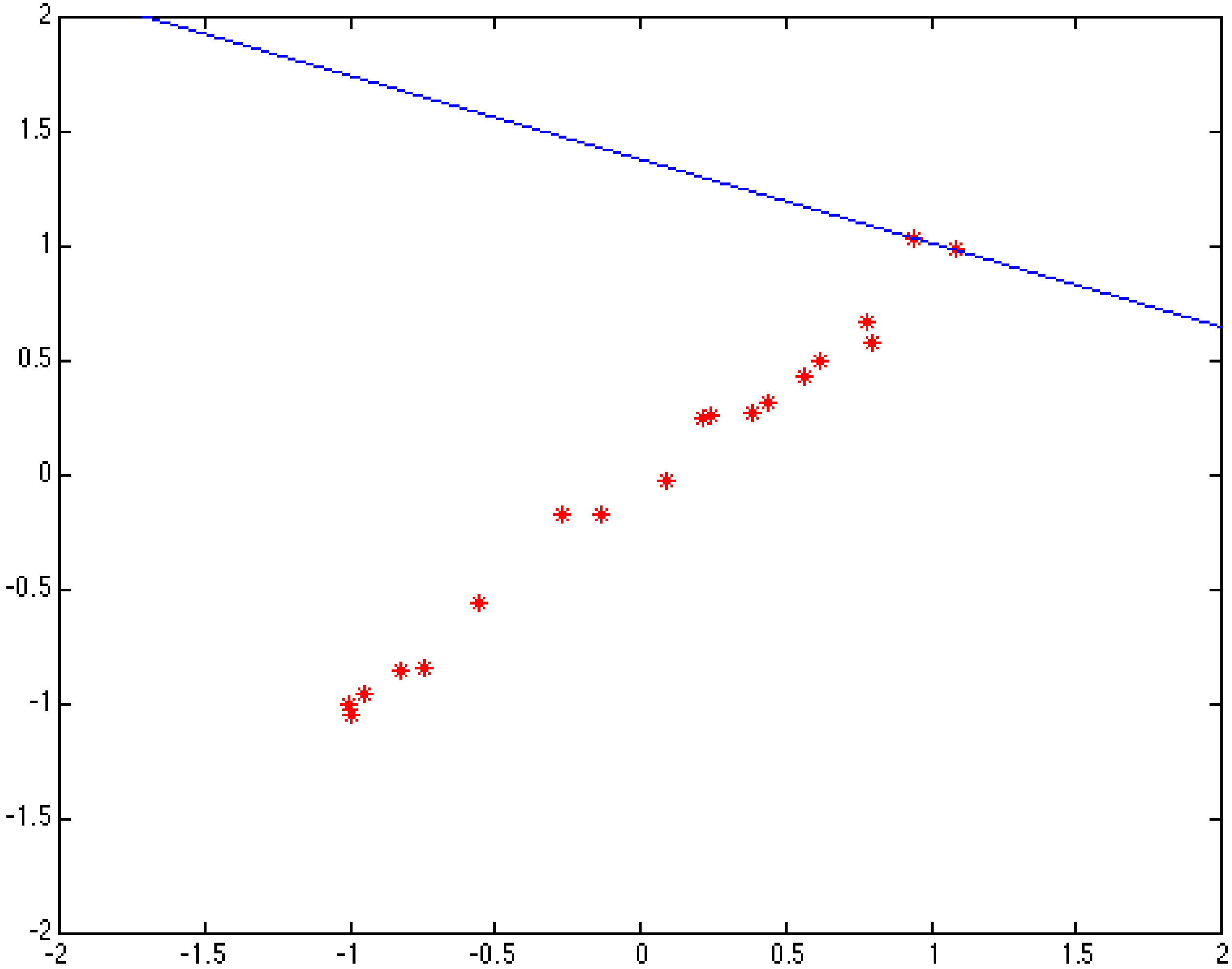
Issues with EM

- Local maxima
  - can be a serious nuisance in some problems
  - no guarantee that we have reached the “right” maximum
- Initialization
  - k means to cluster the points is often a good idea

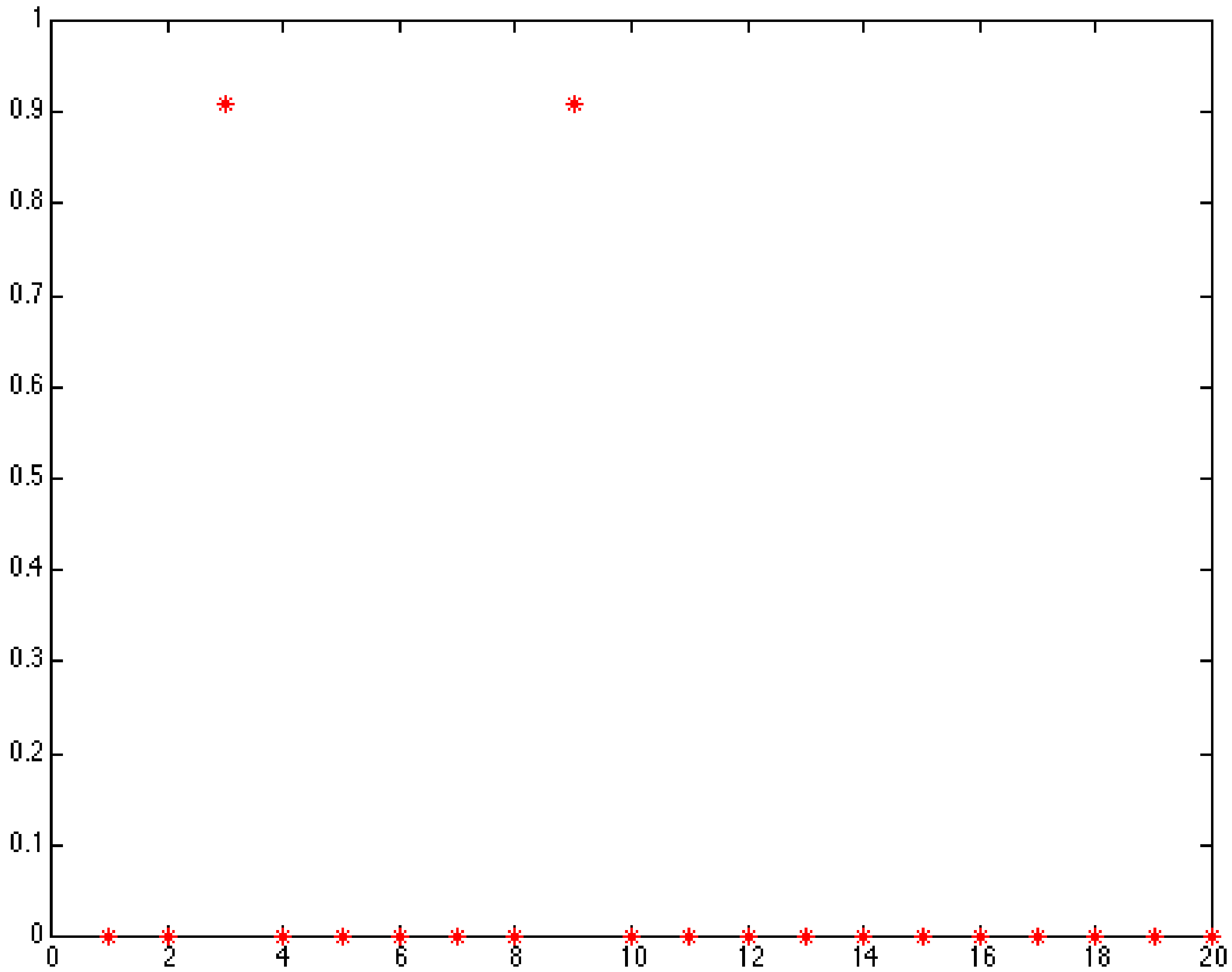
Local maximum



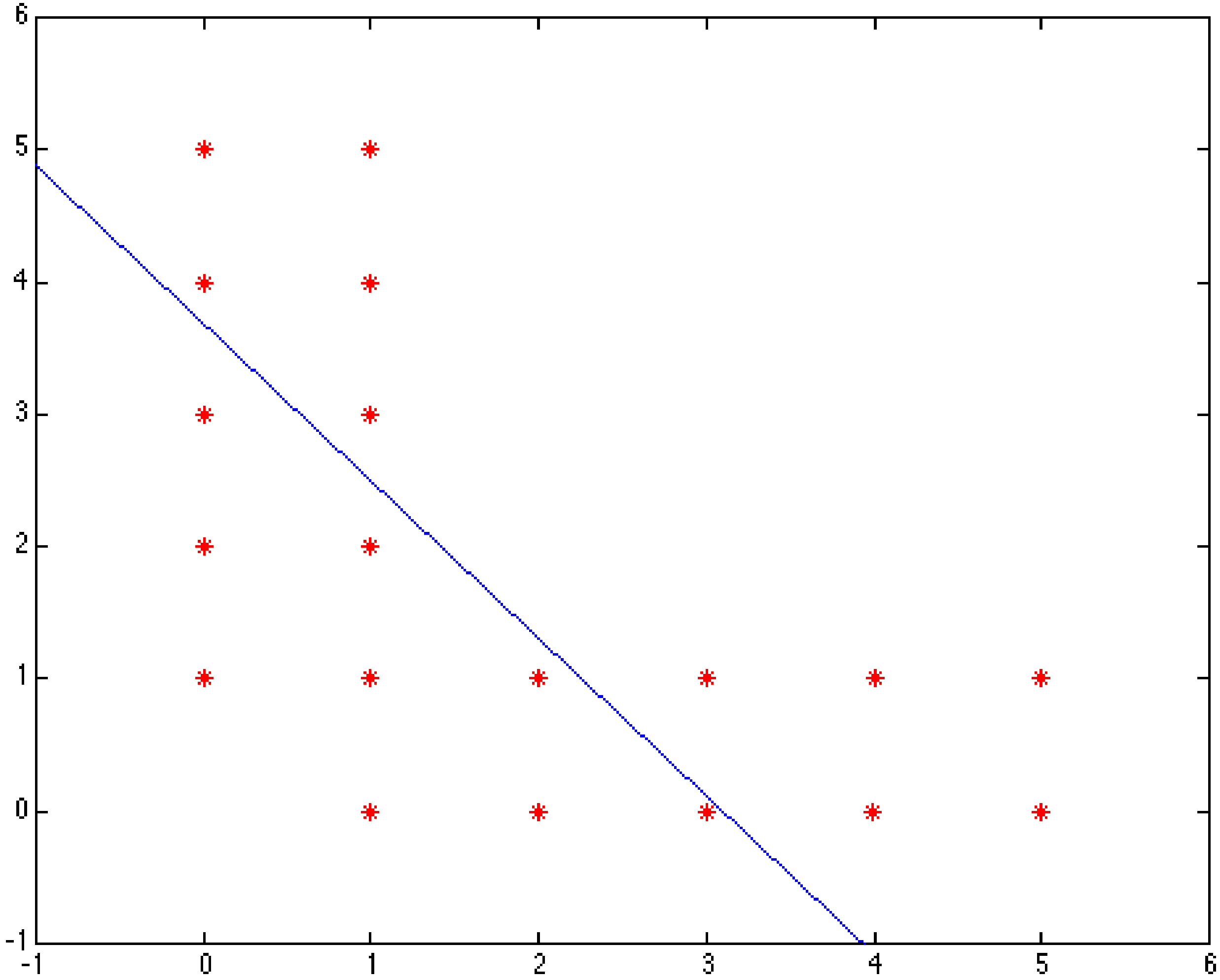
which is an excellent fit to **some** points



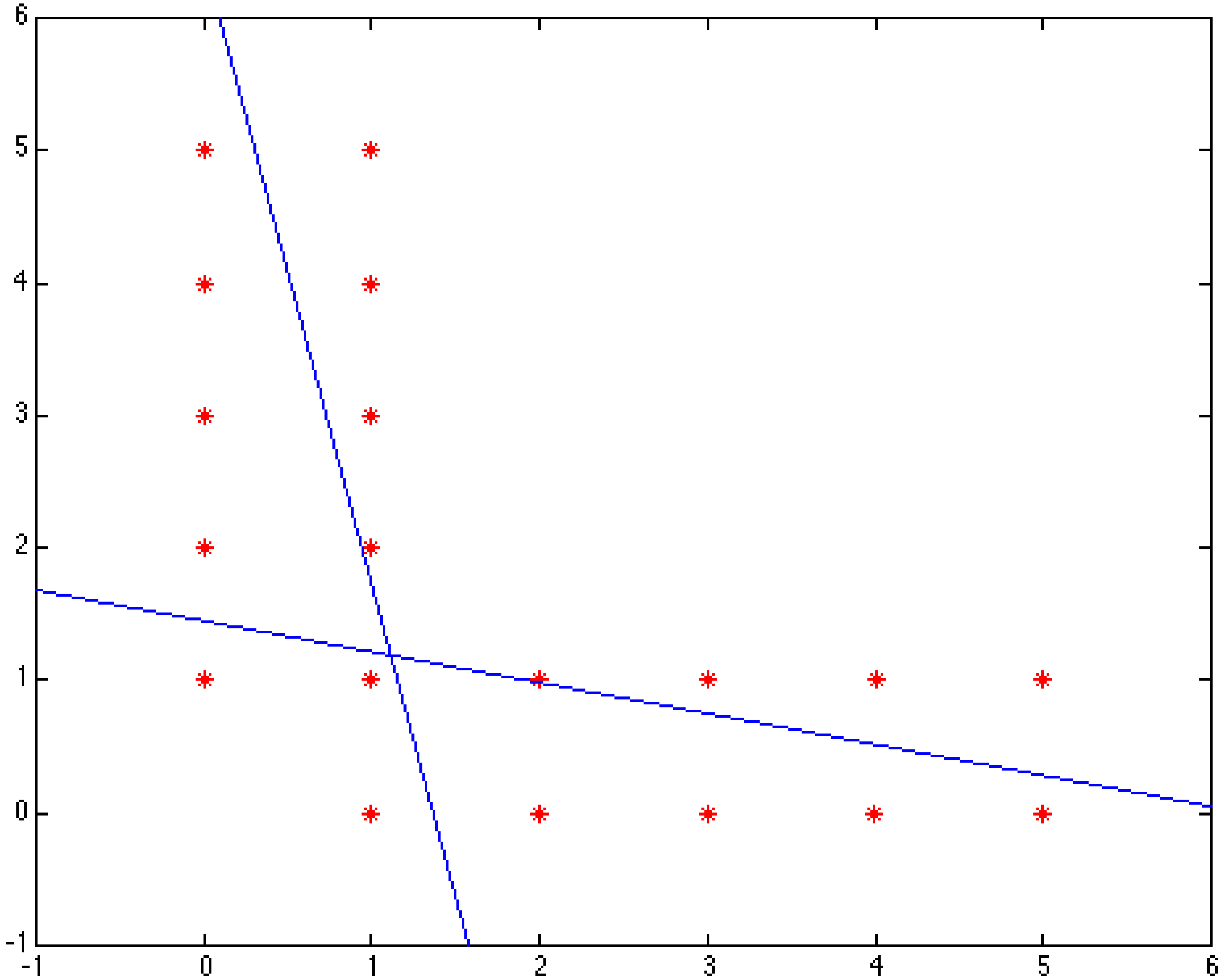
deltas for this maximum



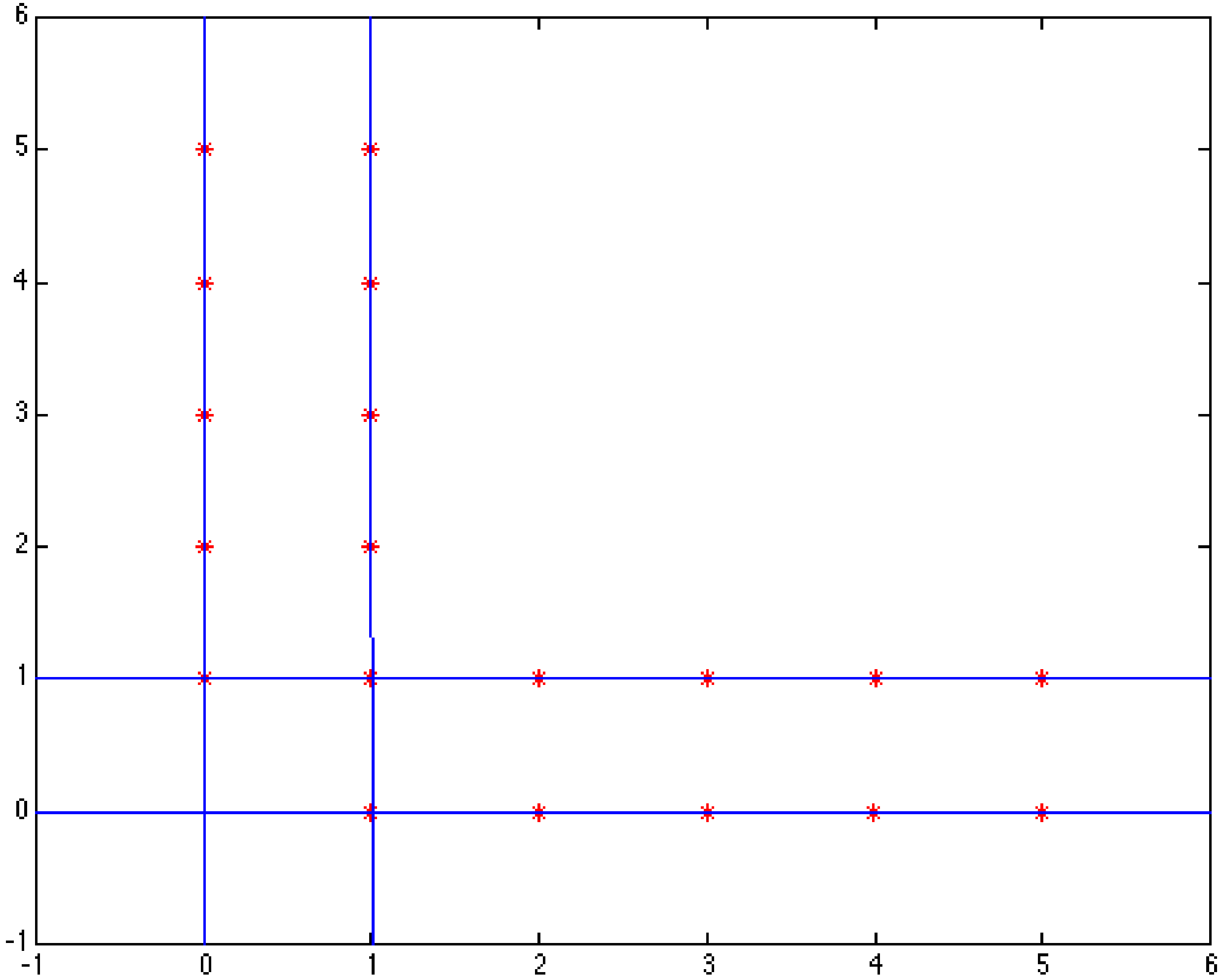
Result of EM fitting, with one line



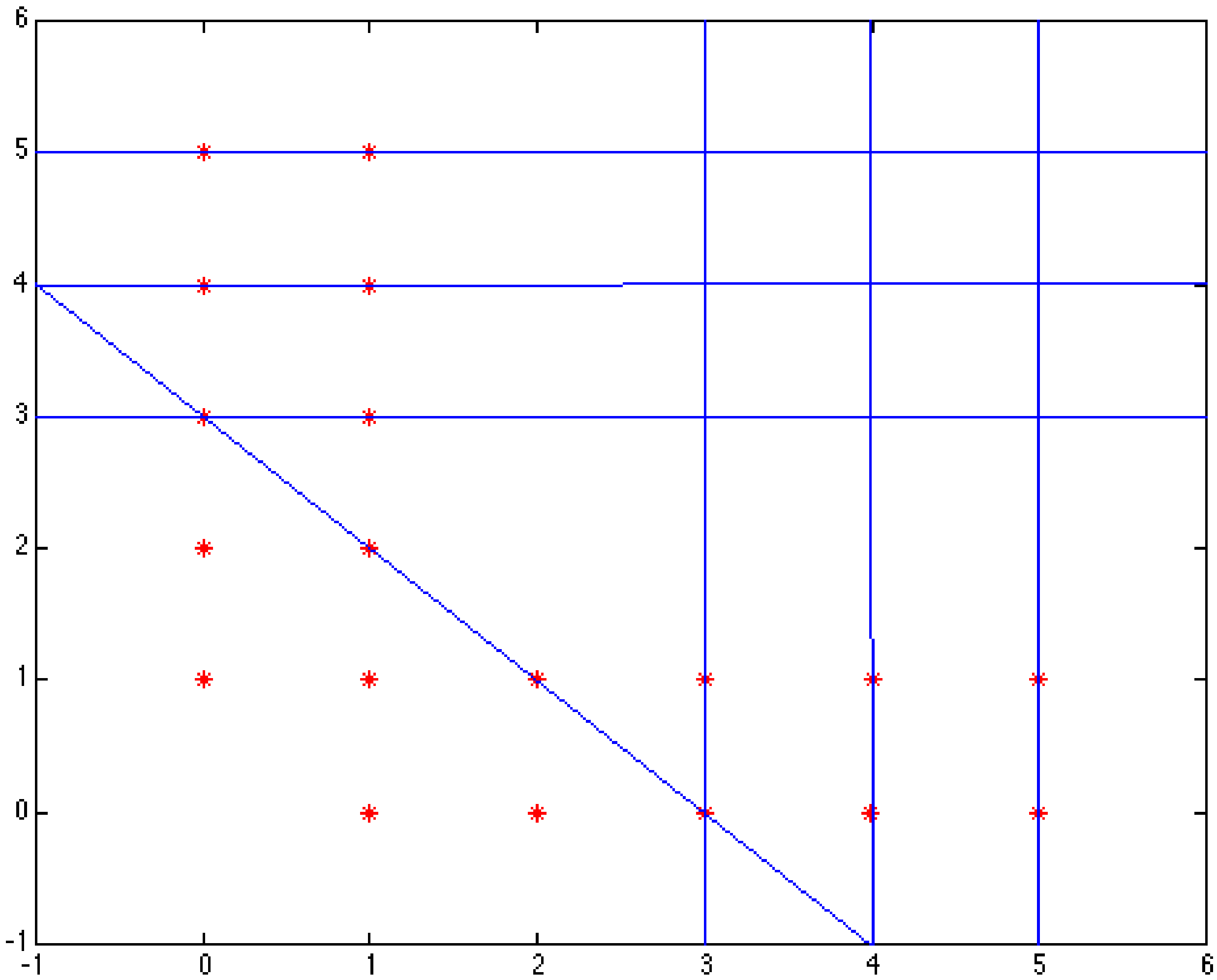
Result of EM fitting, with two lines



A dataset that is well fitted by four lines



Estimation Result with 7 Lines



Segmentation with EM

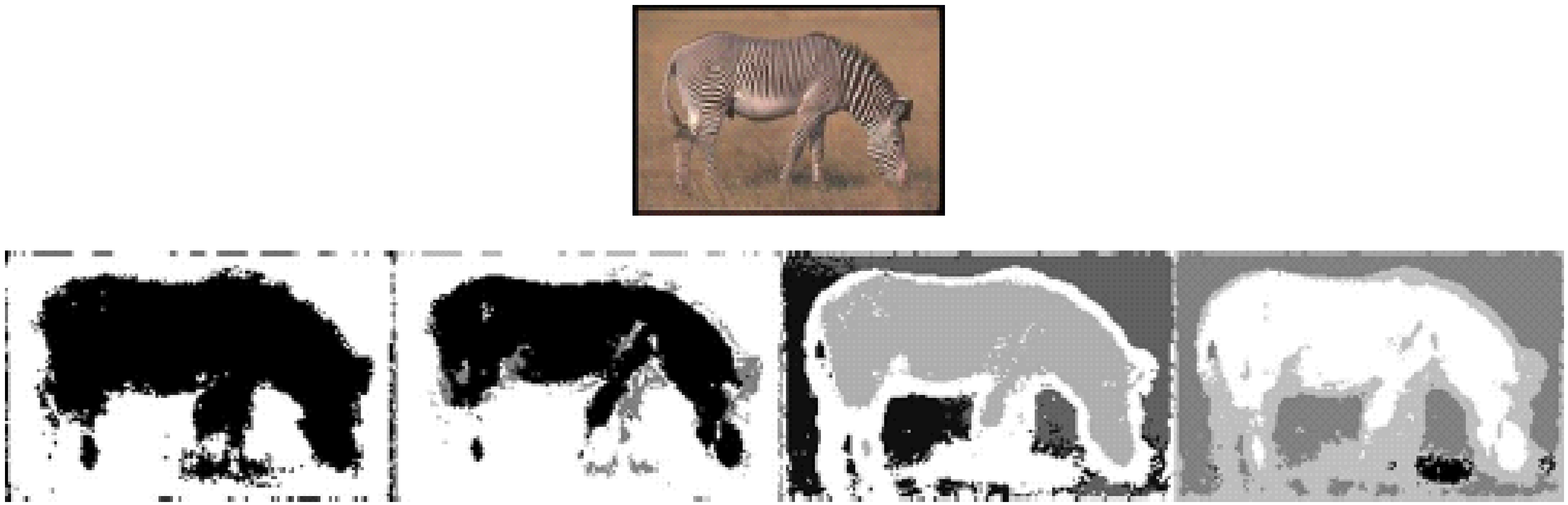


Figure from “Color and Texture Based Image Segmentation Using EM and Its Application to Content Based Image Retrieval”,S.J. Belongie et al., Proc. Int. Conf. Computer Vision, 1998, c1998, IEEE

Motion segmentation with EM

- Model image pair (or video sequence) as consisting of regions of parametric motion
  - affine motion is popular

$$\begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

- Now we need to
  - determine which pixels belong to which region
  - estimate parameters

- Likelihood
  - assume

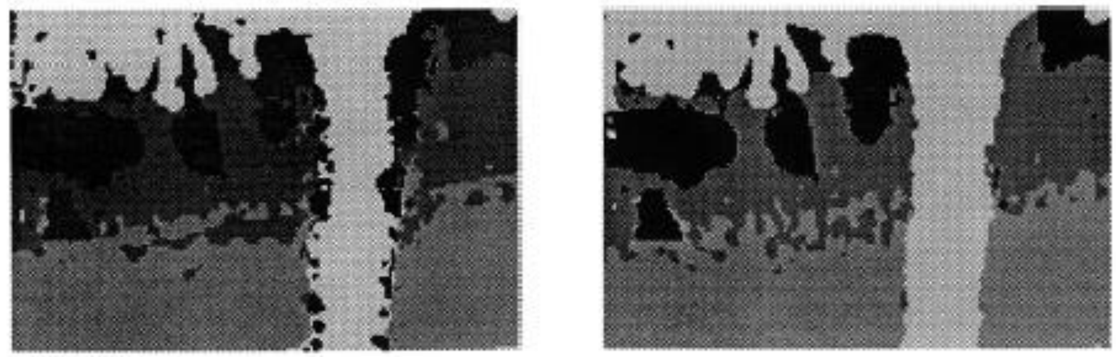
$$I(x,y,t) = I(x+v_x, y+v_y, t+1) + noise$$

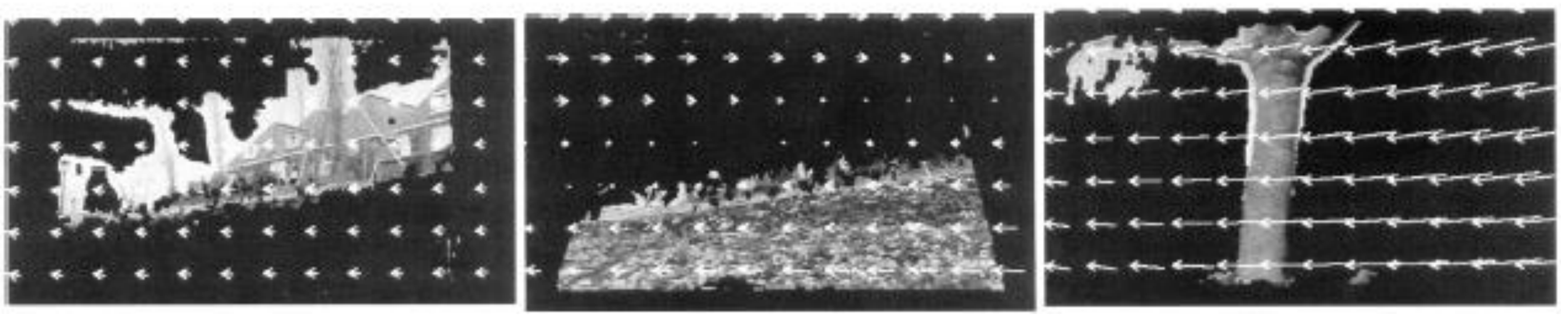
- Straightforward missing variable problem, rest is calculation

Three frames from the MPEG “flower garden” sequence

Figure from “Representing Images with layers,”, by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

Grey level shows region no. with highest probability





Segments and motion fields associated with them

Figure from "Representing Images with layers,", by J. Wang and E.H. Adelson, IEEE Transactions on Image Processing, 1994, c 1994, IEEE

Some Generalities

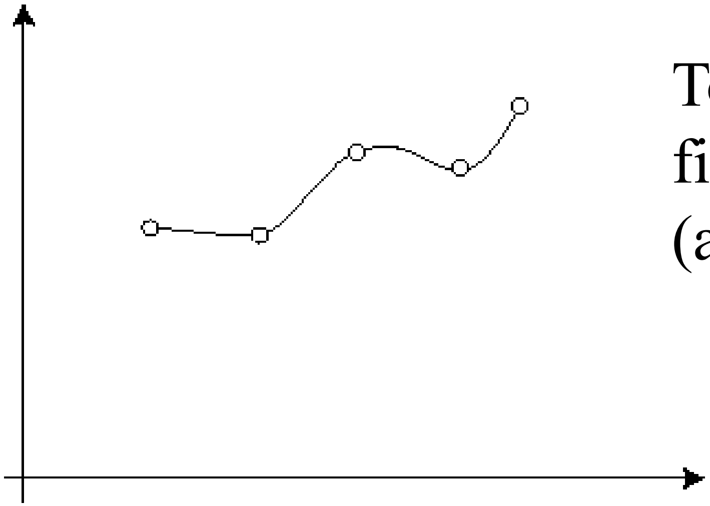
- Many, but not all problems that can be attacked with EM can also be attacked with RANSAC
  - need to be able to get a parameter estimate with a manageably small number of random choices.
  - RANSAC is usually better
- Didn't present in the most general form
  - in the general form, the likelihood may not be a linear function of the missing variables
  - in this case, one takes an expectation of the likelihood, rather than substituting expected values of missing variables

Model Selection

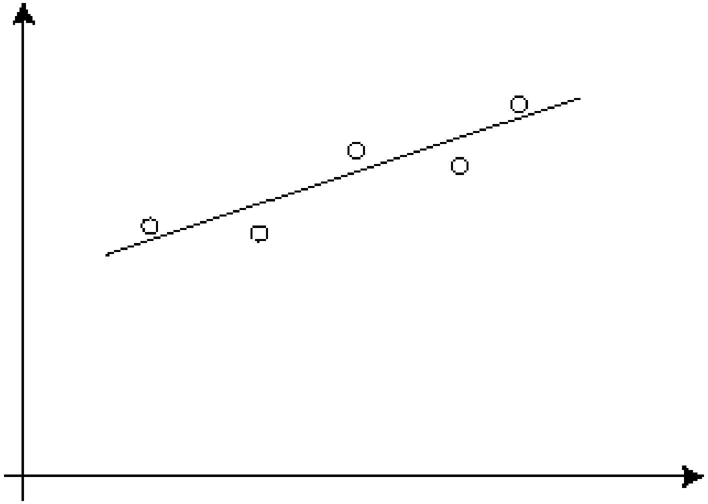
Model Selection

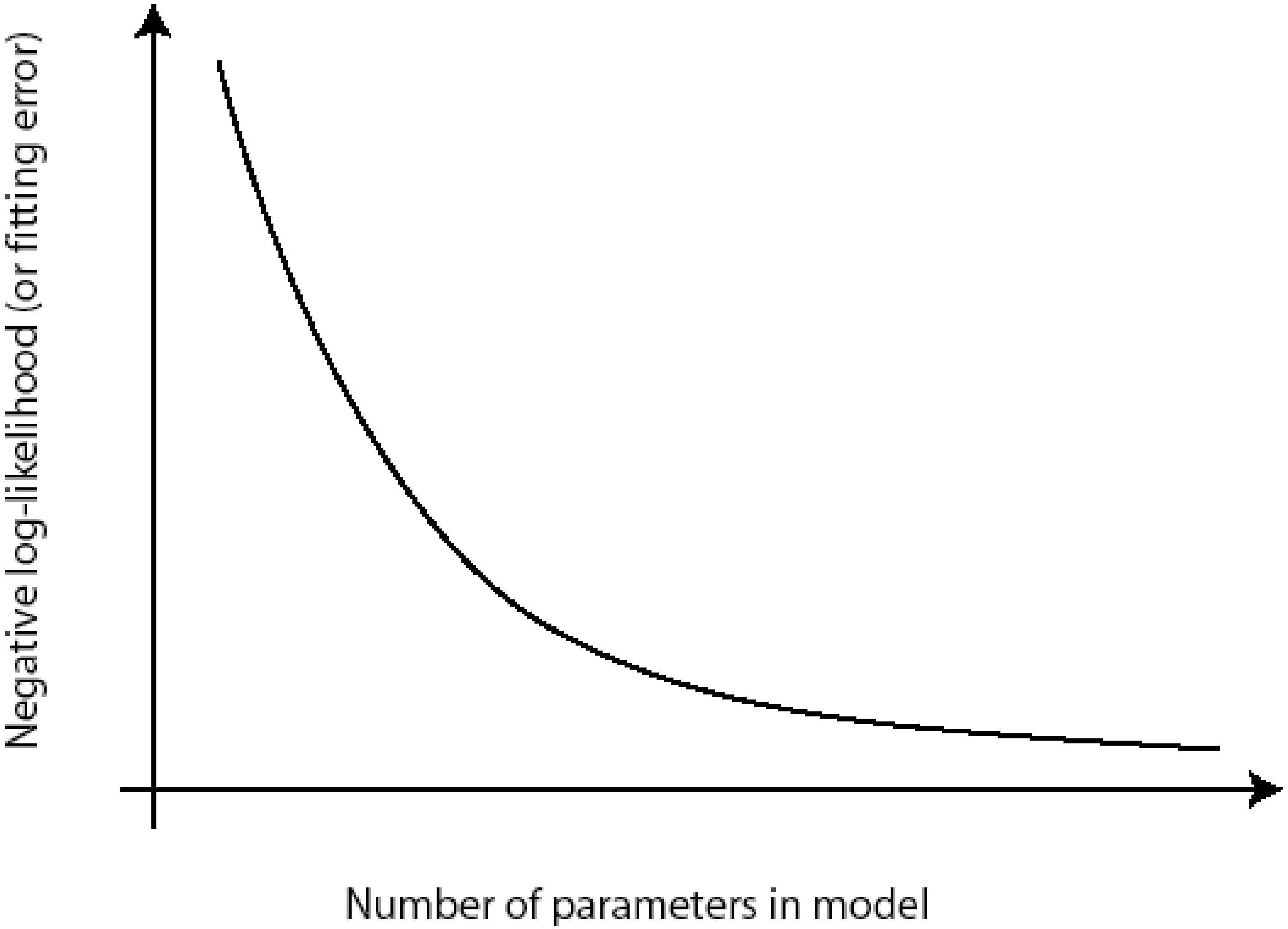
- We wish to choose a model to fit to data
  - e.g. is it a line or a circle?
  - e.g is this a perspective or orthographic camera?
  - e.g. is there an aeroplane there or is it noise?

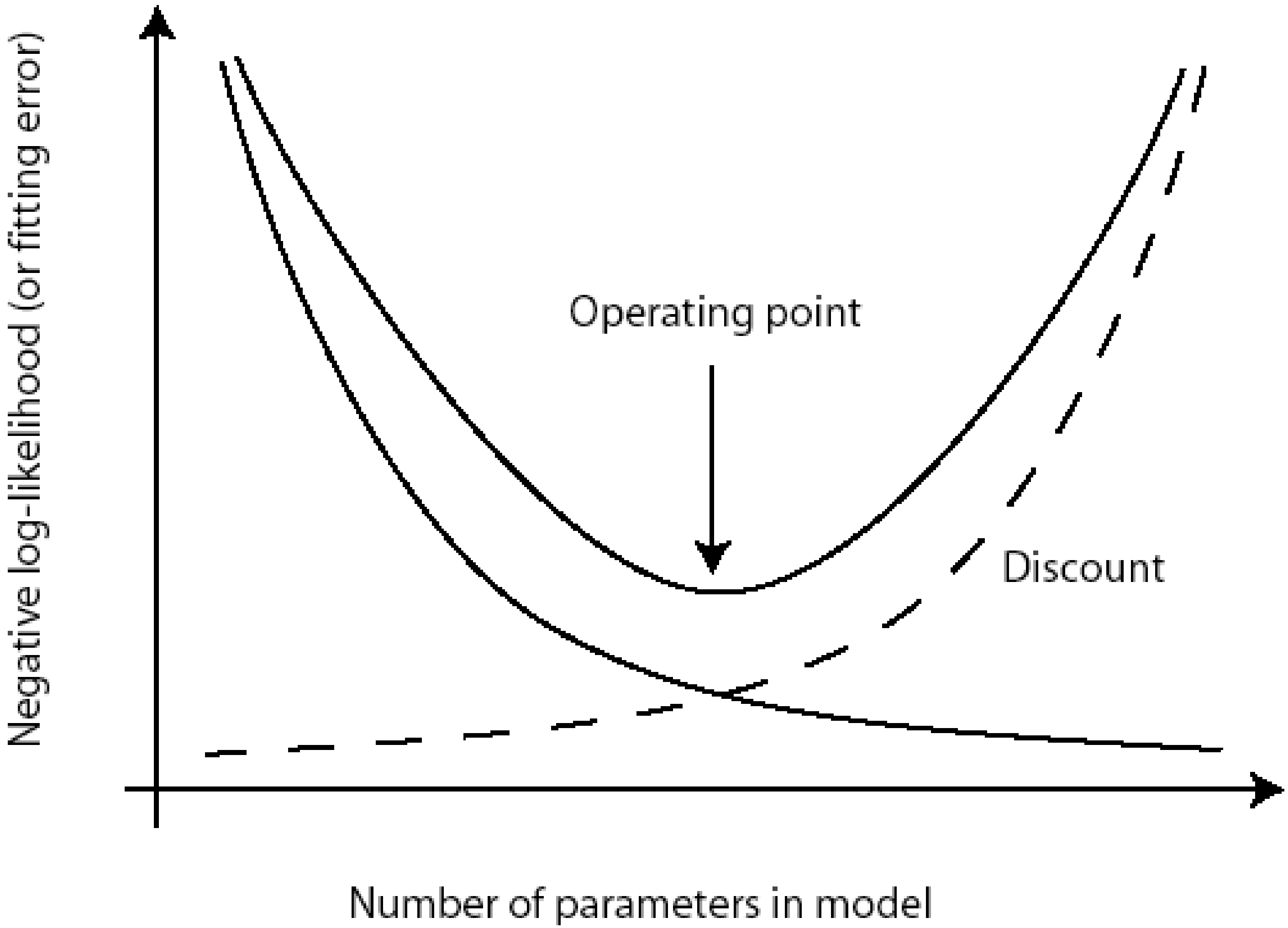
- Issue
  - In general, models with more parameters will fit a dataset better, but are poorer at prediction
  - This means we can't simply look at the negative log-likelihood (or fitting error)



Top is not necessarily a better fit than bottom
(actually, almost always worse)




Negative log-likelihood (or fitting error)
Number of parameters in model



We can discount the fitting error with some term in the number of parameters in the model.

Discounts

- AIC (an information criterion)
  - choose model with smallest value of

$$-2L\left(D;\theta^*\right)+2p$$

  - p is the number of parameters

- BIC (Bayes information criterion)
  - choose model with smallest value of

$$-2L\left(D;\theta^*\right)+p\log N$$

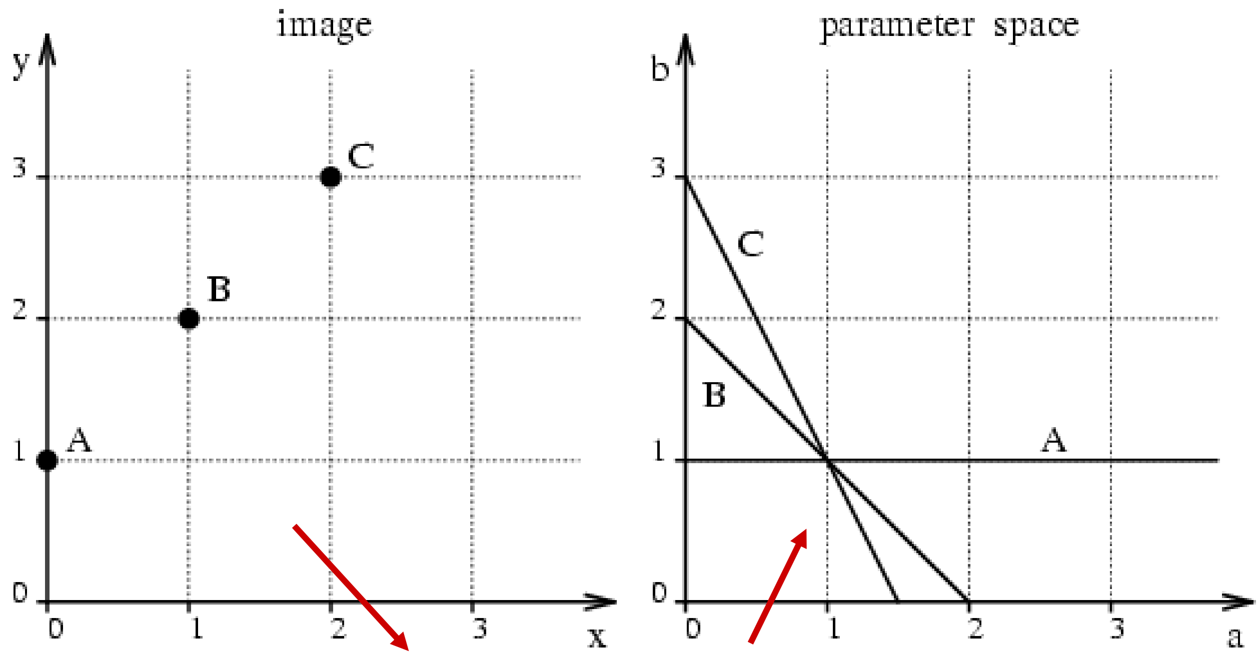
  - N is the number of data points

Cross-validation

- Split data set into two pieces, fit to one, and compute negative log-likelihood on the other
- Average over multiple different splits
- Choose the model with the smallest value of this average
- The difference in averages for two different models is an estimate of the difference in KL divergence of the models from the source of the data

Hough Transform

Hough transform : straight lines

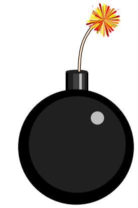


$$b = (-x)\,a + y$$

implementation :

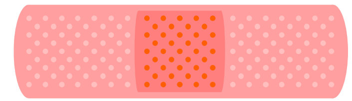
1. the parameter space is discretised
2. a counter is incremented at each cell where the lines pass
3. peaks are detected

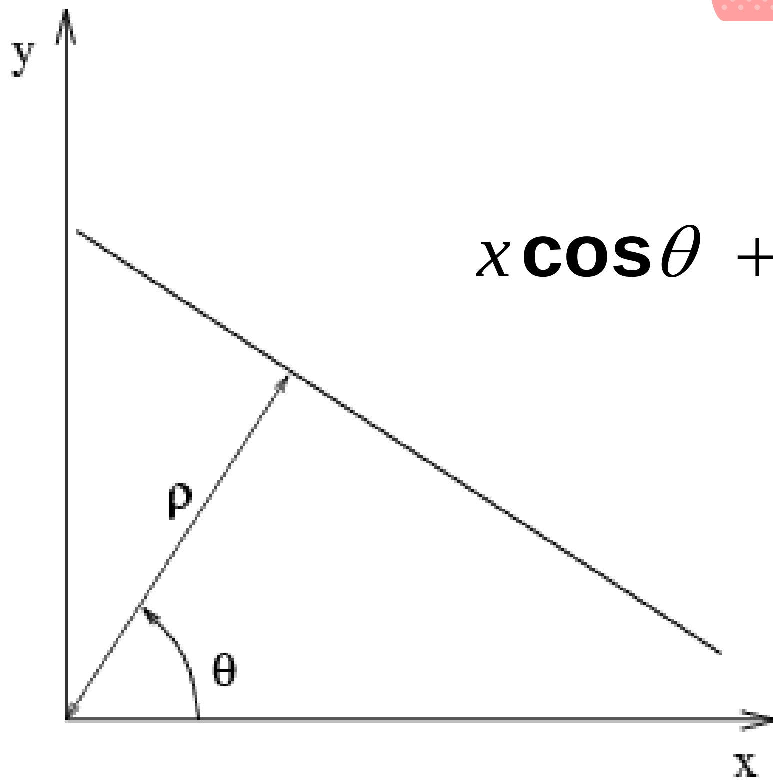
Hough transform : straight lines



problem : unbounded parameter domain
vertical lines require infinite *a*

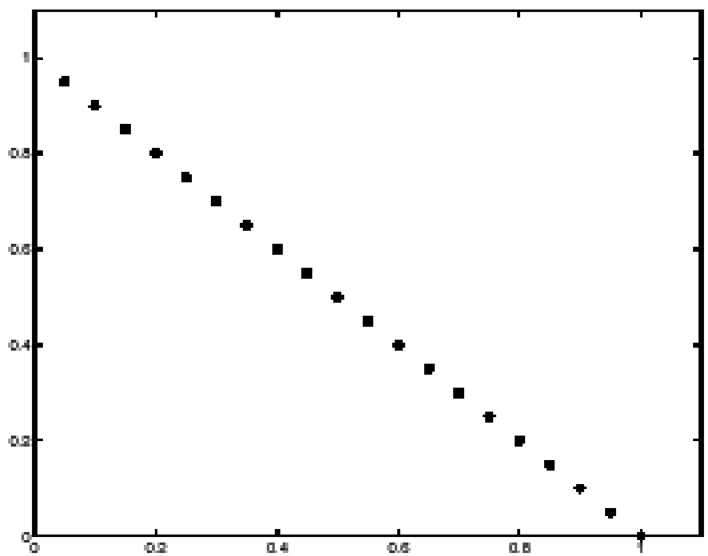
alternative representation:



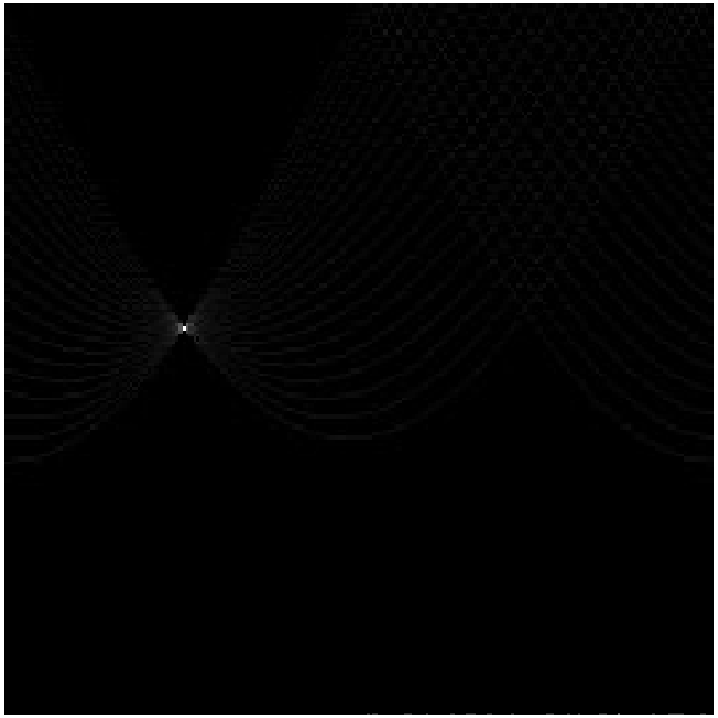


$$x\,\mathbf{cos}\theta + y\,\mathbf{sin}\theta = \rho$$

Each point will add a cosine function in the (θ,ρ) parameter space tokens



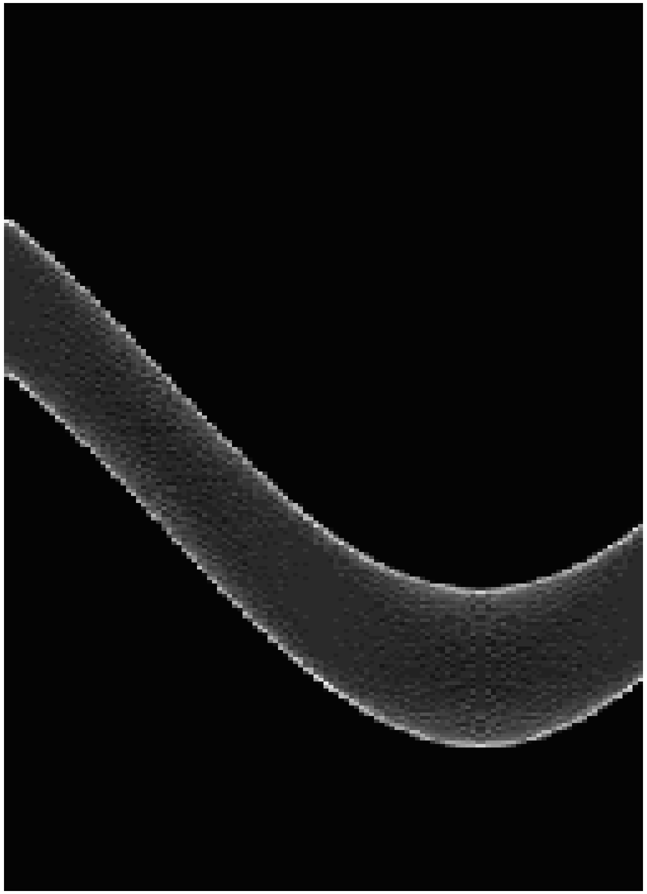
votes



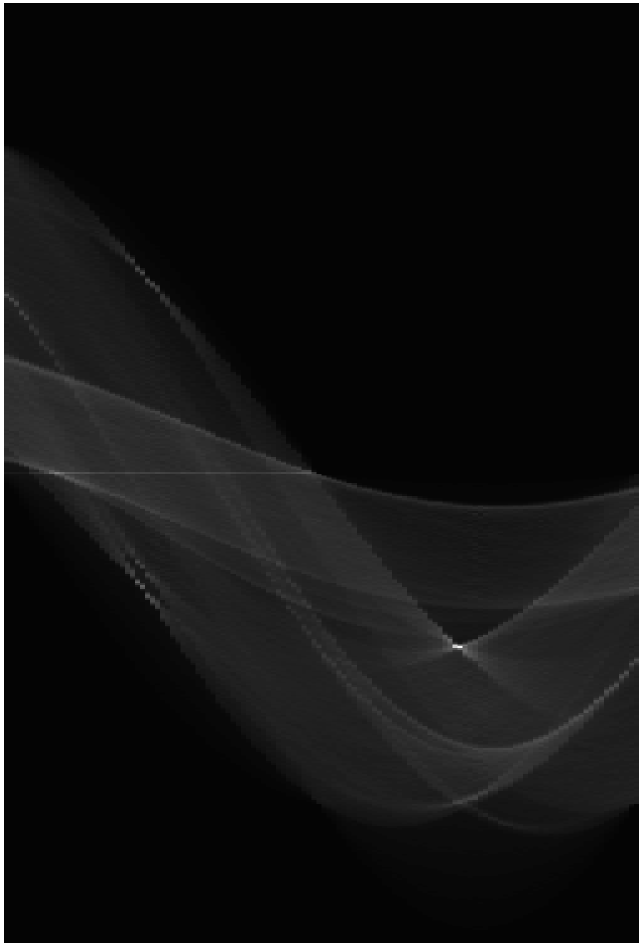
Hough transform : straight lines

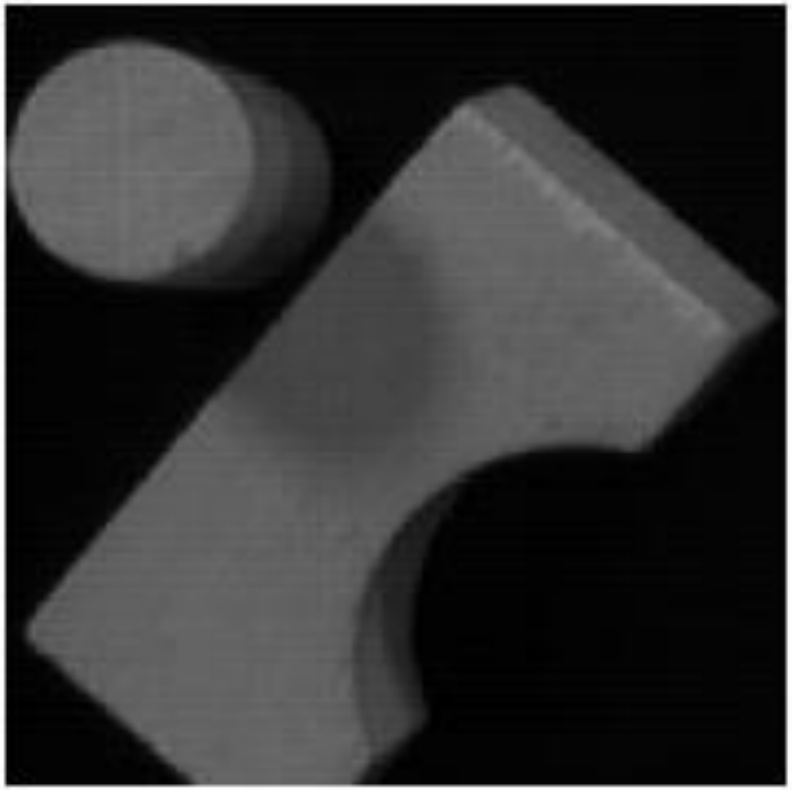
Square :

Circle :



Hough transform : straight lines



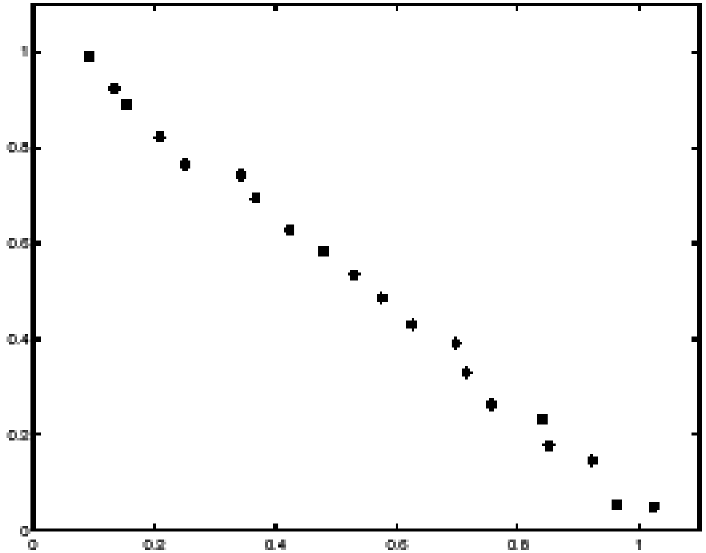


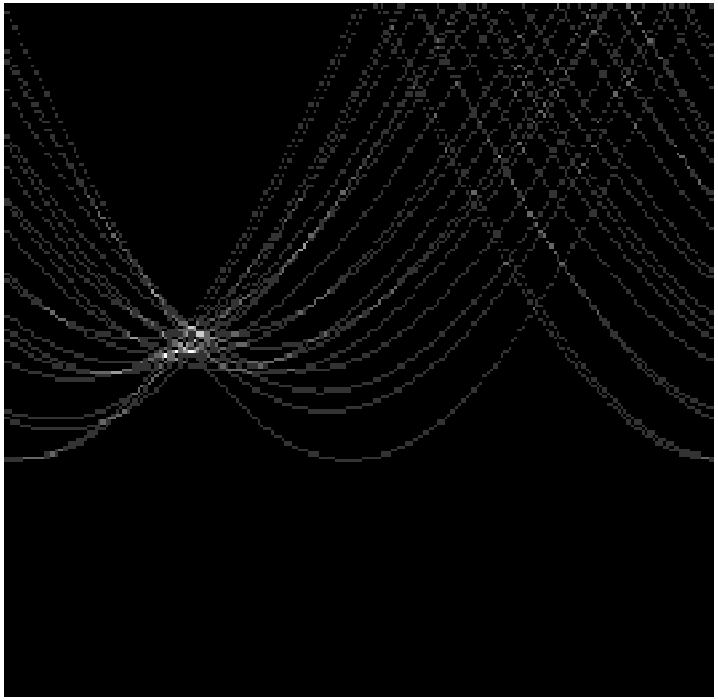
Problems of Hough Transform

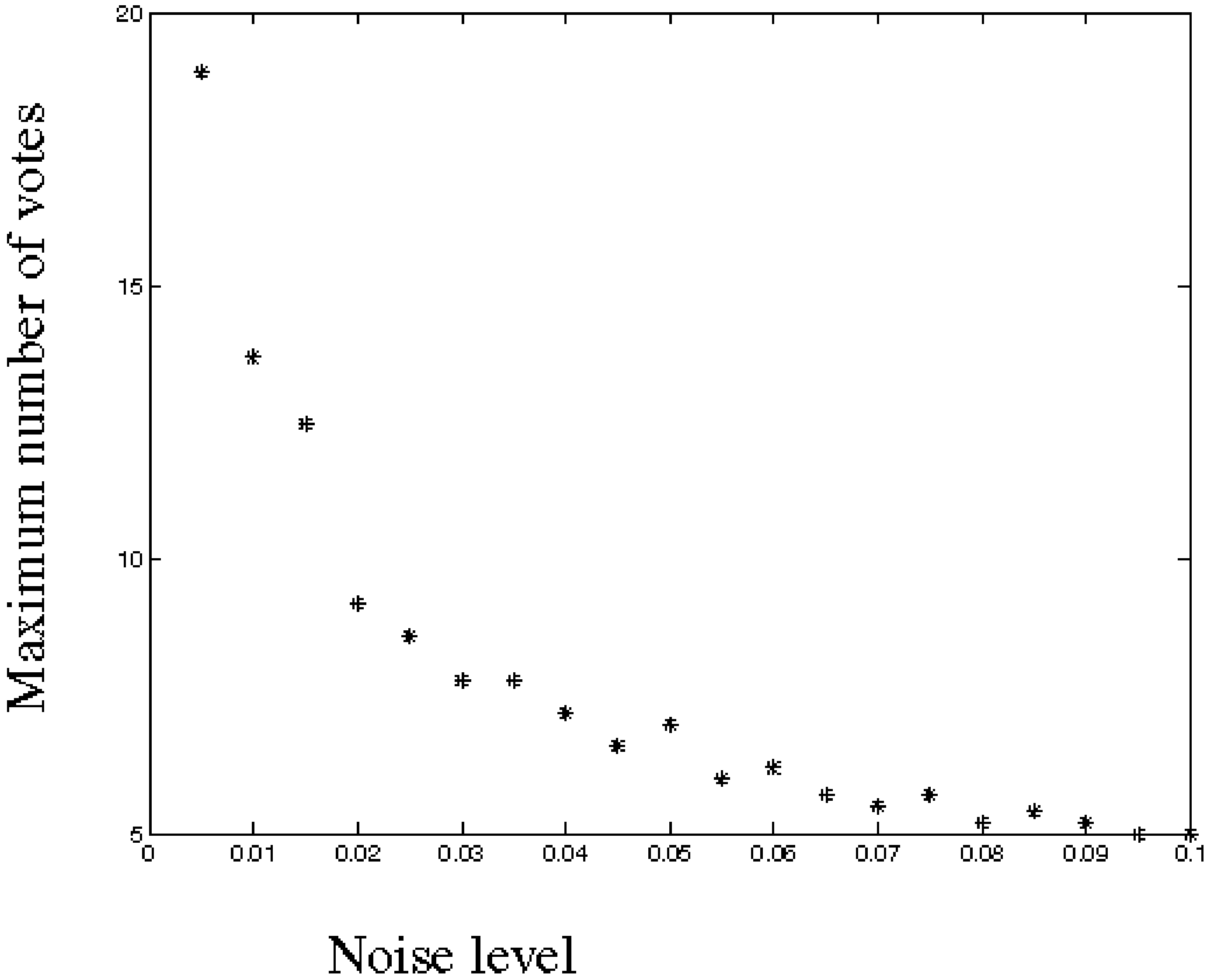
- How big should the cells be?
  - too big, and we cannot distinguish between quite different lines
  - too small, and noise causes lines to be missed)
- How many lines?
  - count the peaks in the Hough array
- Who belongs to which line?
  - tag the votes
- Hardly ever satisfactory in practice, because problems with noise and cell size defeat it

tokens

votes






Maximum number of votes
Noise level
20
15
10
5
0
0.01
0.02
0.03
0.04
0.05
0.06
0.07
0.08
0.09
0.1

Thank You