Computer Vision & Image Processing CSE 473 / 573

Instructor - Kevin R. Keane, PhD

TAs - Radhakrishna Dasari, Yuhao Du, Niyazi Sorkunlu

Lecture 25

October 27, 2017

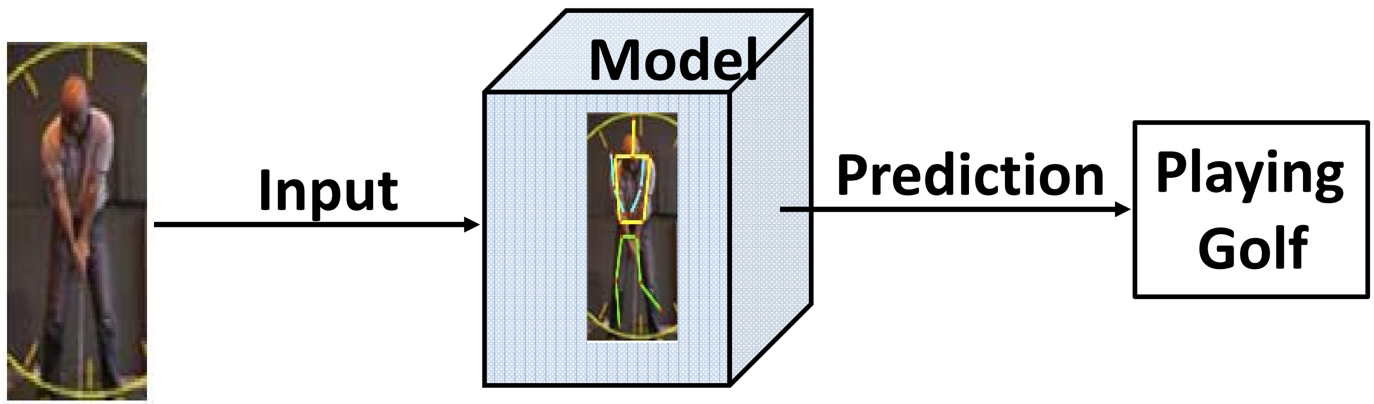
Line fitting and RANSAC

Schedule

- Last class
  - End of segmentation discussion
- Today
  - Robust model fitting
- Readings for today:
  - Forsyth and Ponce chapter 9

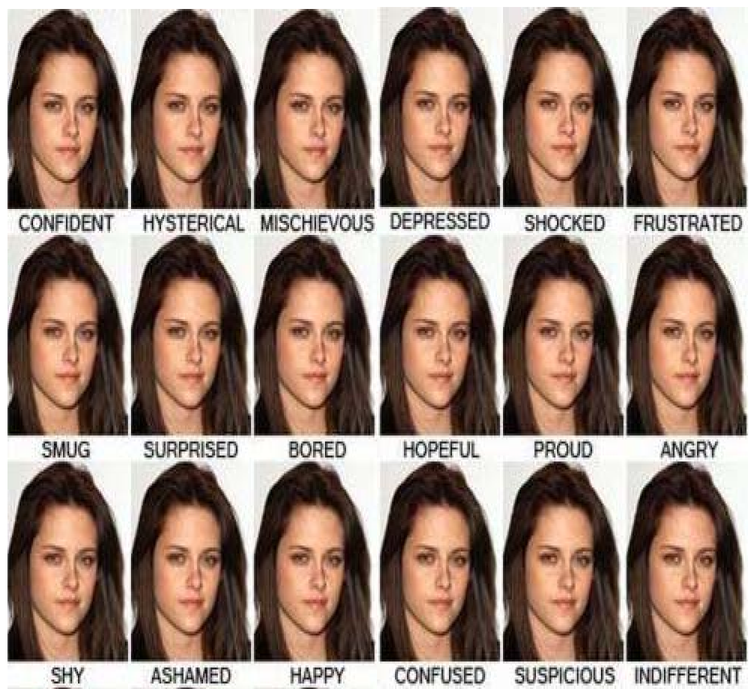
Mathematical Models

- Compact Understanding of the World



Mathematical Models - Example

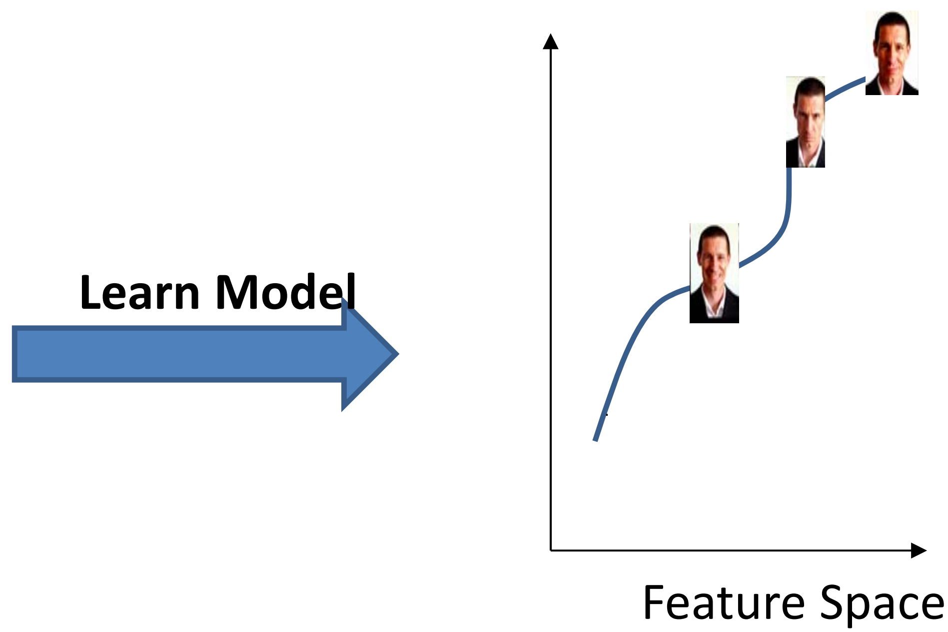
- Face Recognition with varying expressions



- Too Easy…

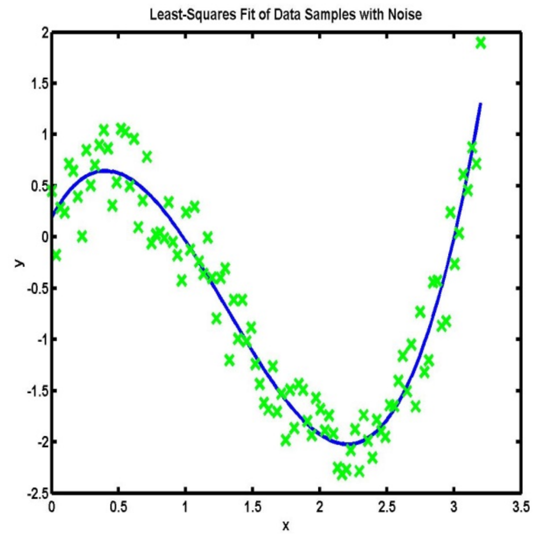
Mathematical Models

- Face Recognition with varying expressions

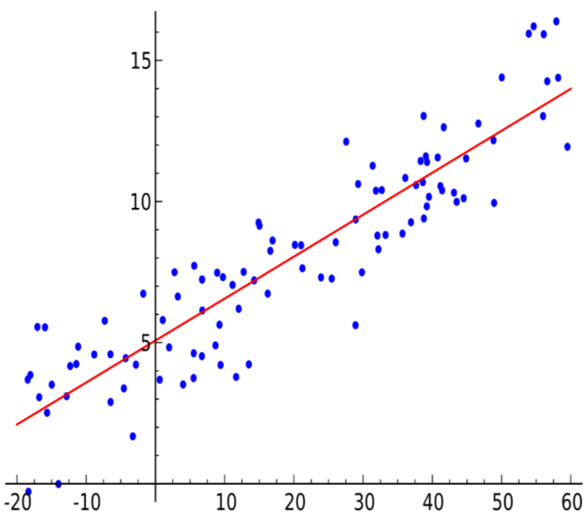


I. Least Squares

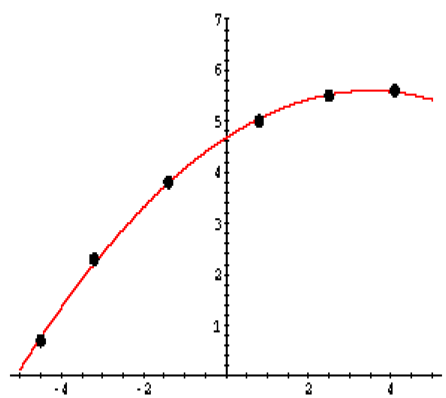
- Fitting Curves/Learning Data Manifolds



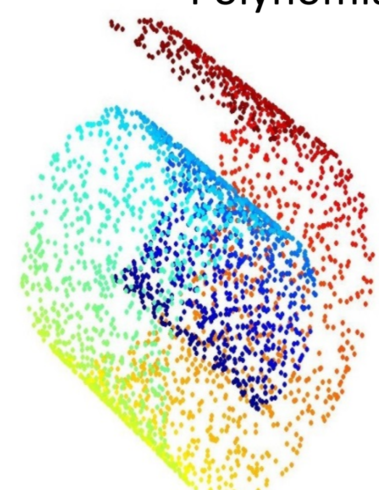
Fitting Higher Degree Polynomials



Fitting Line



Fitting Quadratic Curve



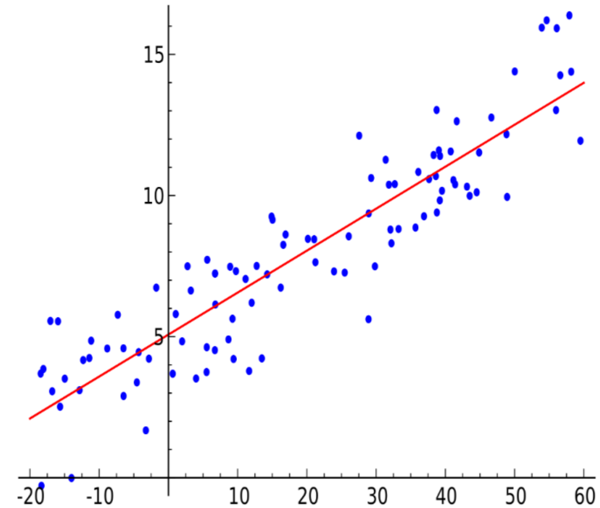
Learning Manifolds

Line Fitting

- Goal: Find a line that best explains the observed data

- Target: y_i
- Data: x_i
- Line parameter: w,b
- Line Model:

$$y_i = w\,x_i + b$$



Fitting Line

Line Fitting

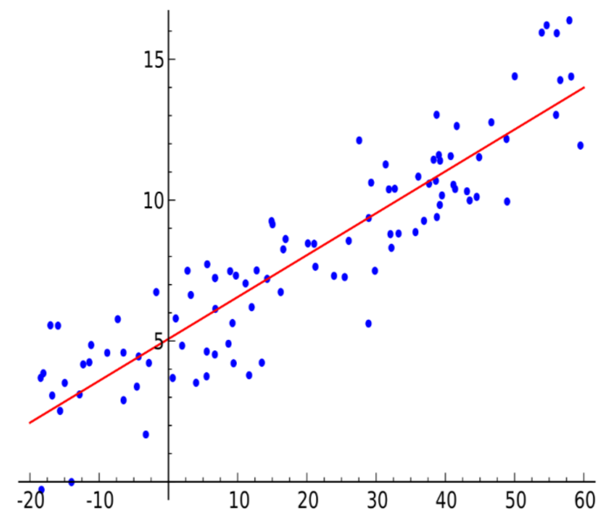
- Line Model:

$y_i = w\ x_i + b$

- Too many samples!

- Minimize error:

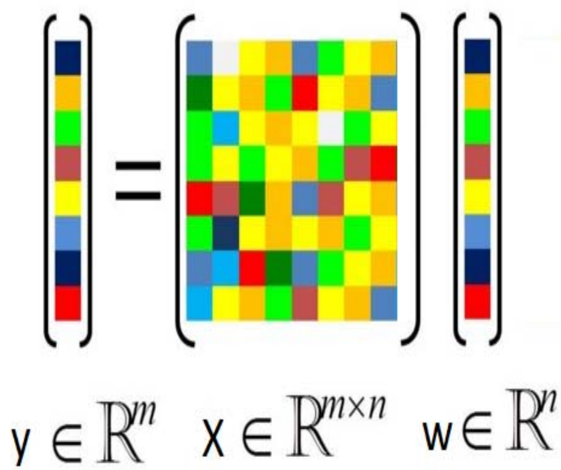
$$\min_{w,b}.\ \sum_{i=1}^{N}(y_i - w\ x_i + b)^2$$

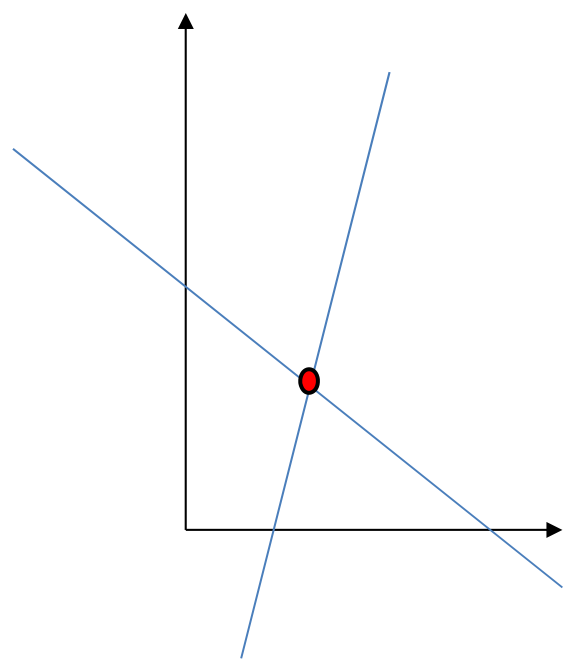


Fitting Line

#Samples(m) vs #Model-Parameters(n)

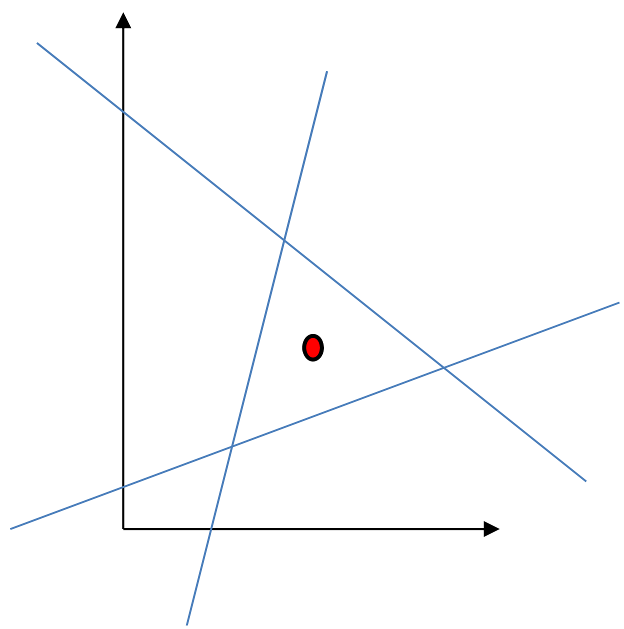
- Case 1 (m=n): Unique Solution
- **w=X\y**
- No least square required

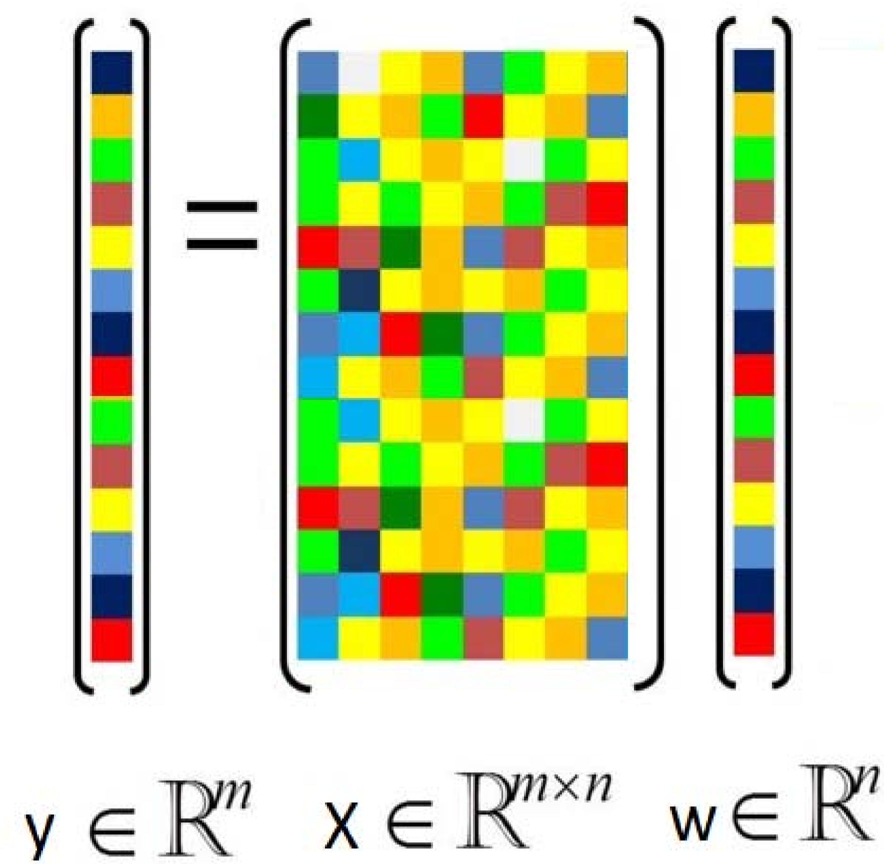




#Samples(m) vs #Model-Parameters(n)

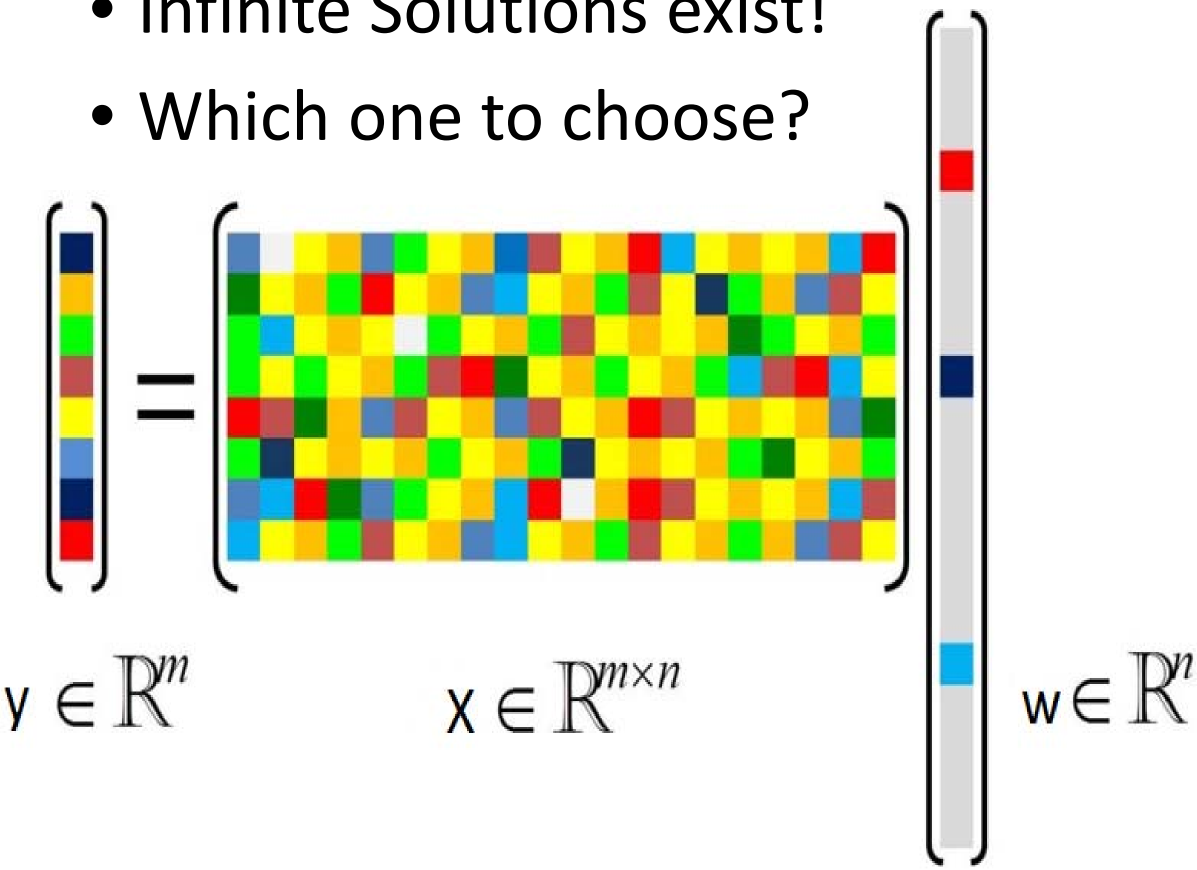
- Case 2 (m>n): Over-determined system of equations
- No Solution exists!
- Hence, we minimize error (fitting)

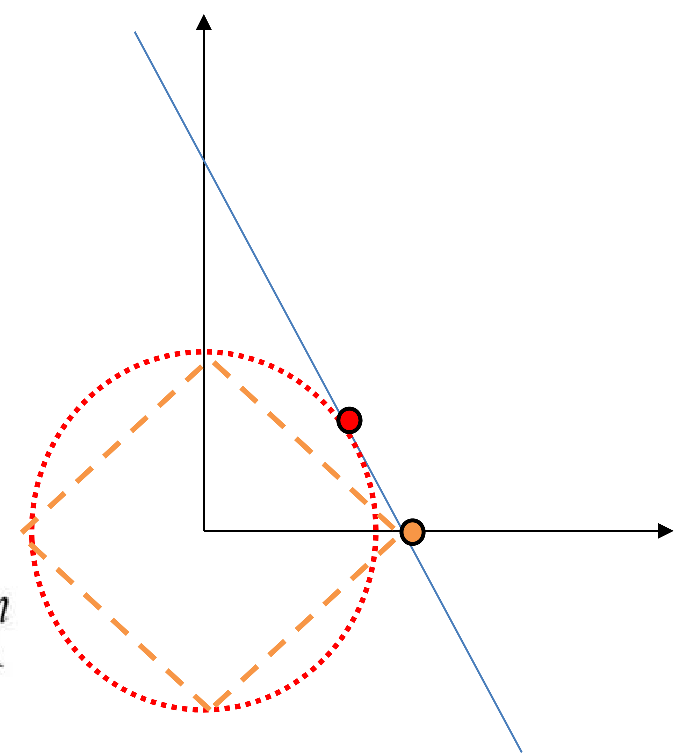




#Samples(m) vs #Model-Parameters(n)

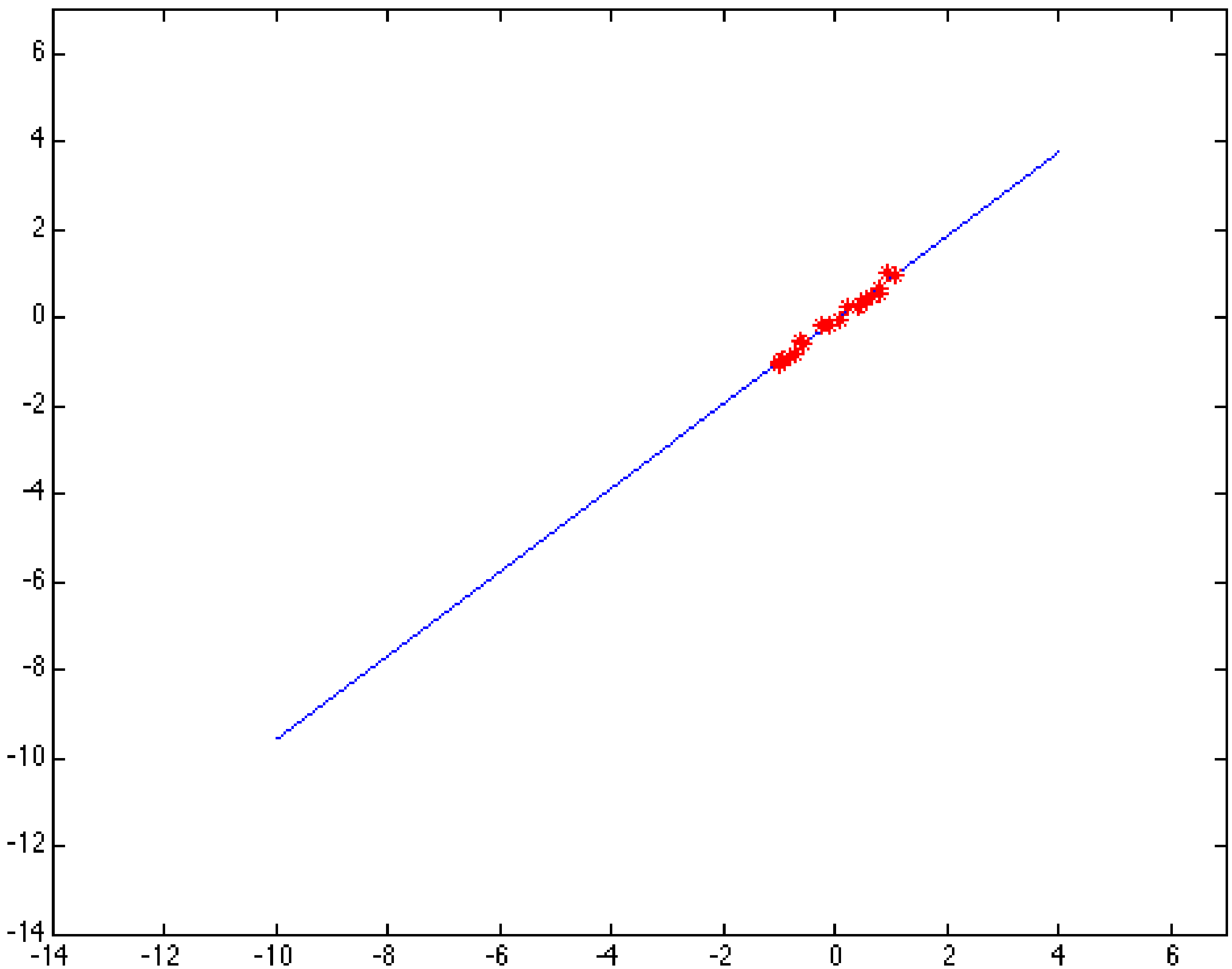
- Case 3 (m<n): Under-determined system of equations
- Infinite Solutions exist!
- Which one to choose?

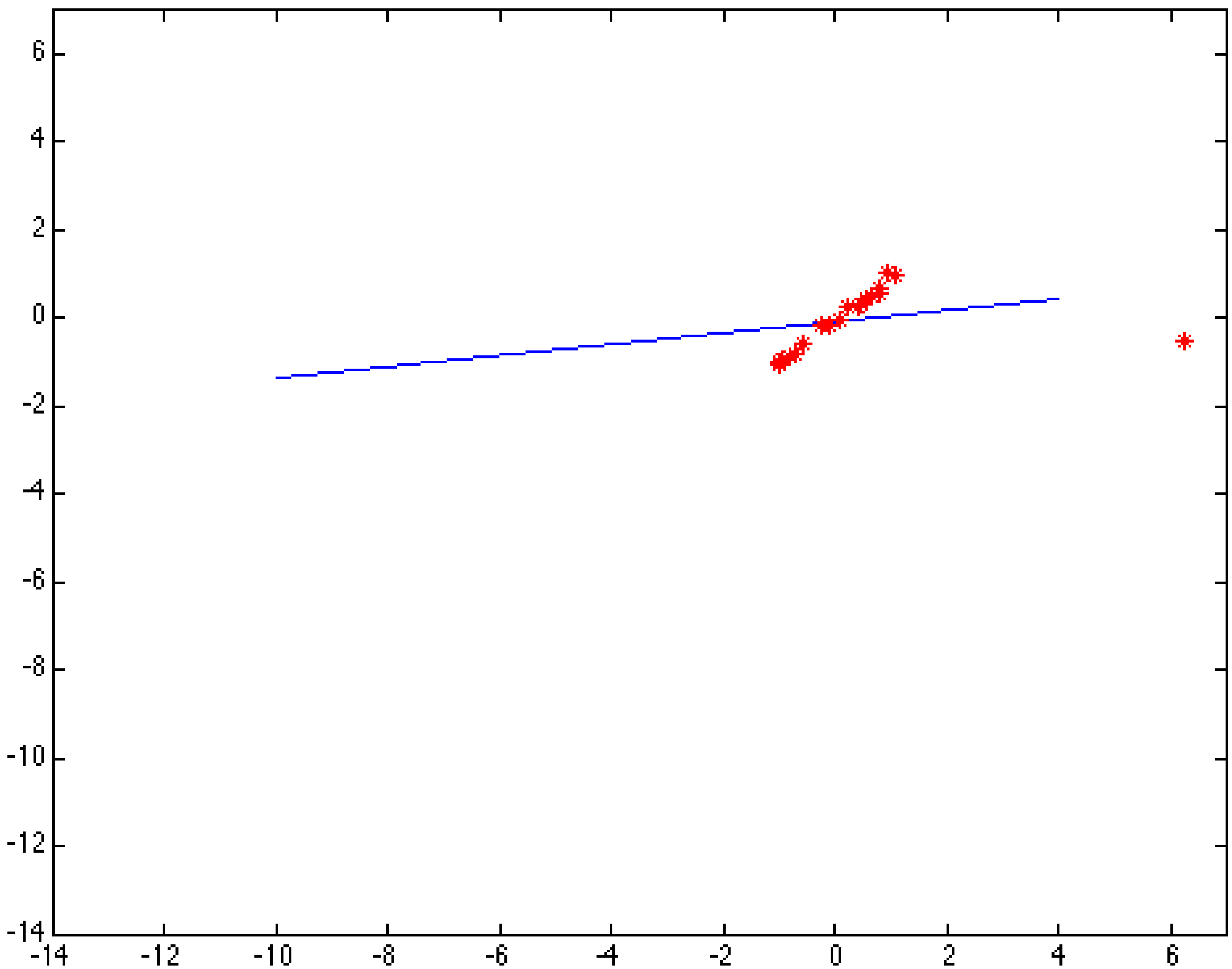


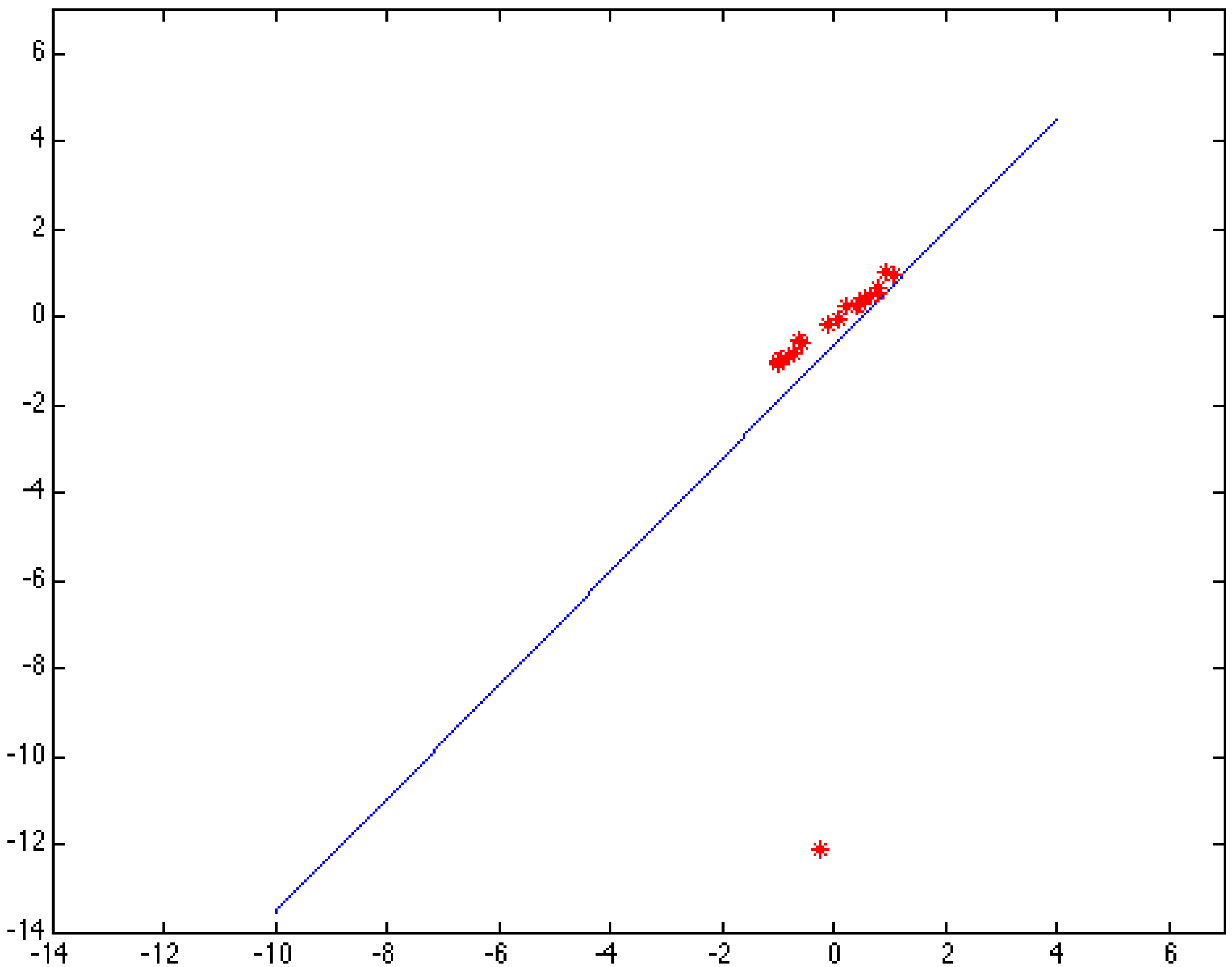


Robustness

- As we have seen, squared error can be a source of bias in the presence of noise points
  - One fix is EM - details in F&P textbook
  - Another is an M-estimator (we will look at this shortly)
    - Square nearby distances, threshold far away
  - A third is RANSAC
    - Search for good points


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

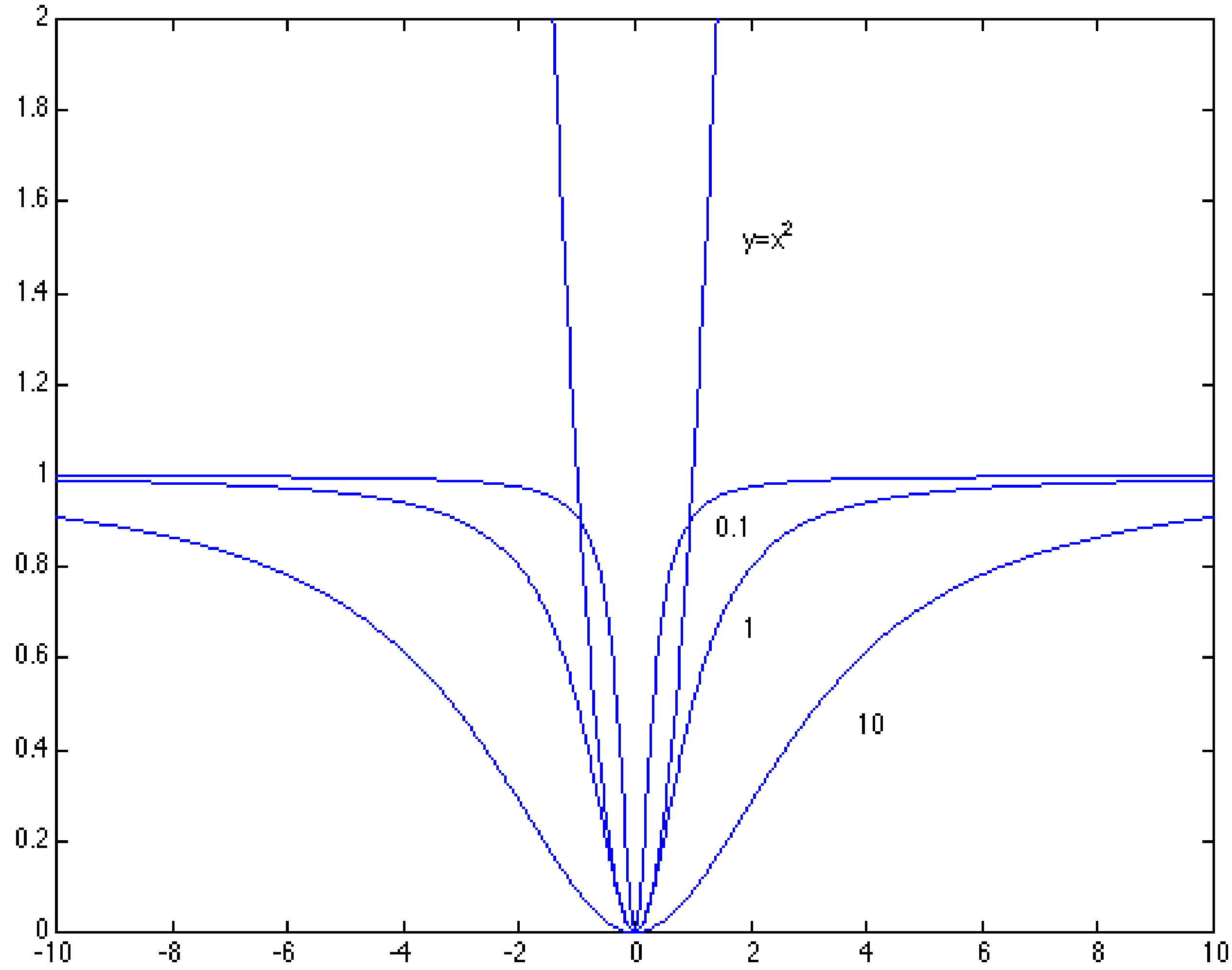

6
4
2
0
-2
-4
-6
-8
-10
-12
-14
-14 -12 -10 -8 -6 -4 -2 0 2 4 6

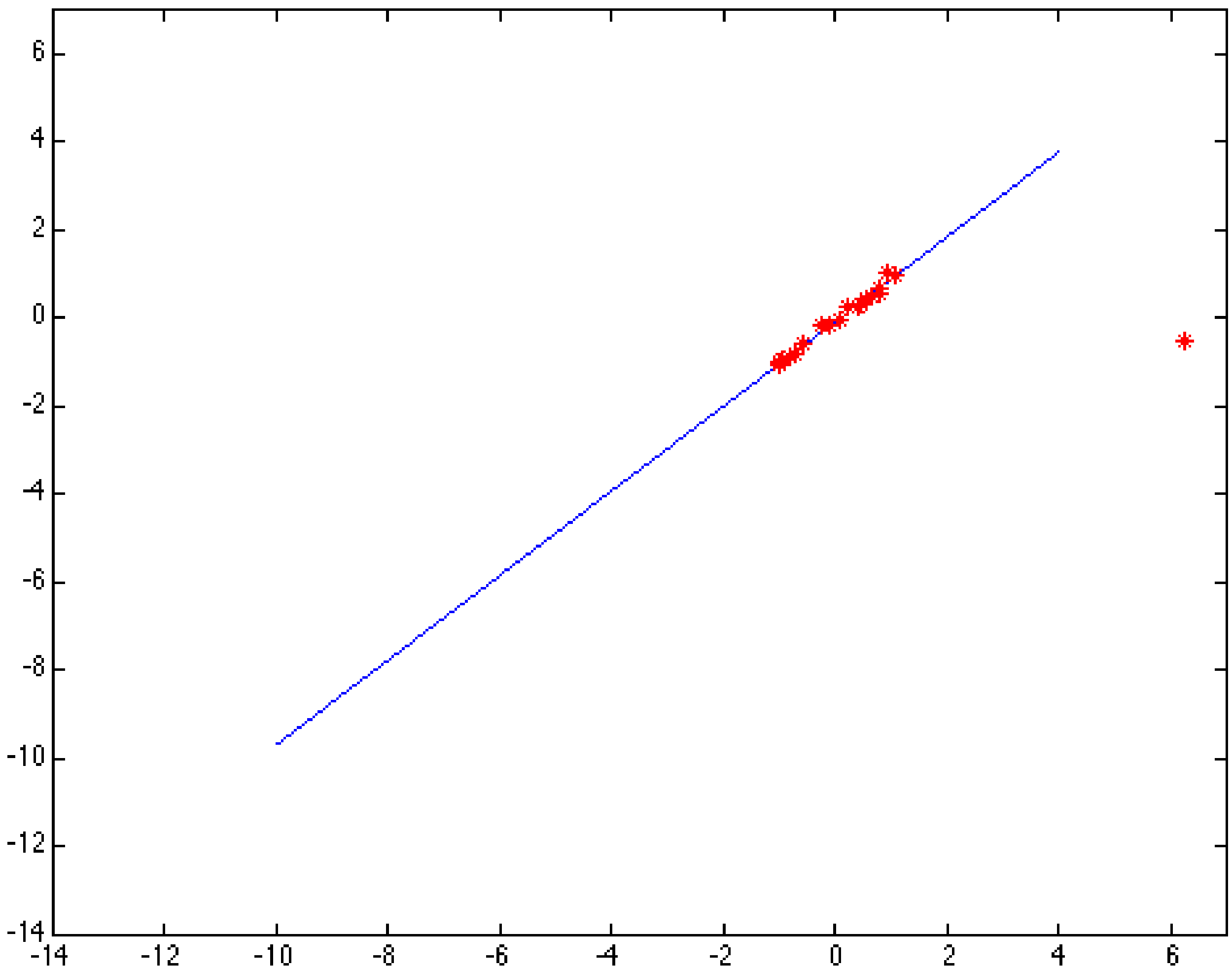


6
4
2
0
-2
-4
-6
-8
-10
-12
-14
-14
-12
-10
-8
-6
-4
-2
0
2
4
6

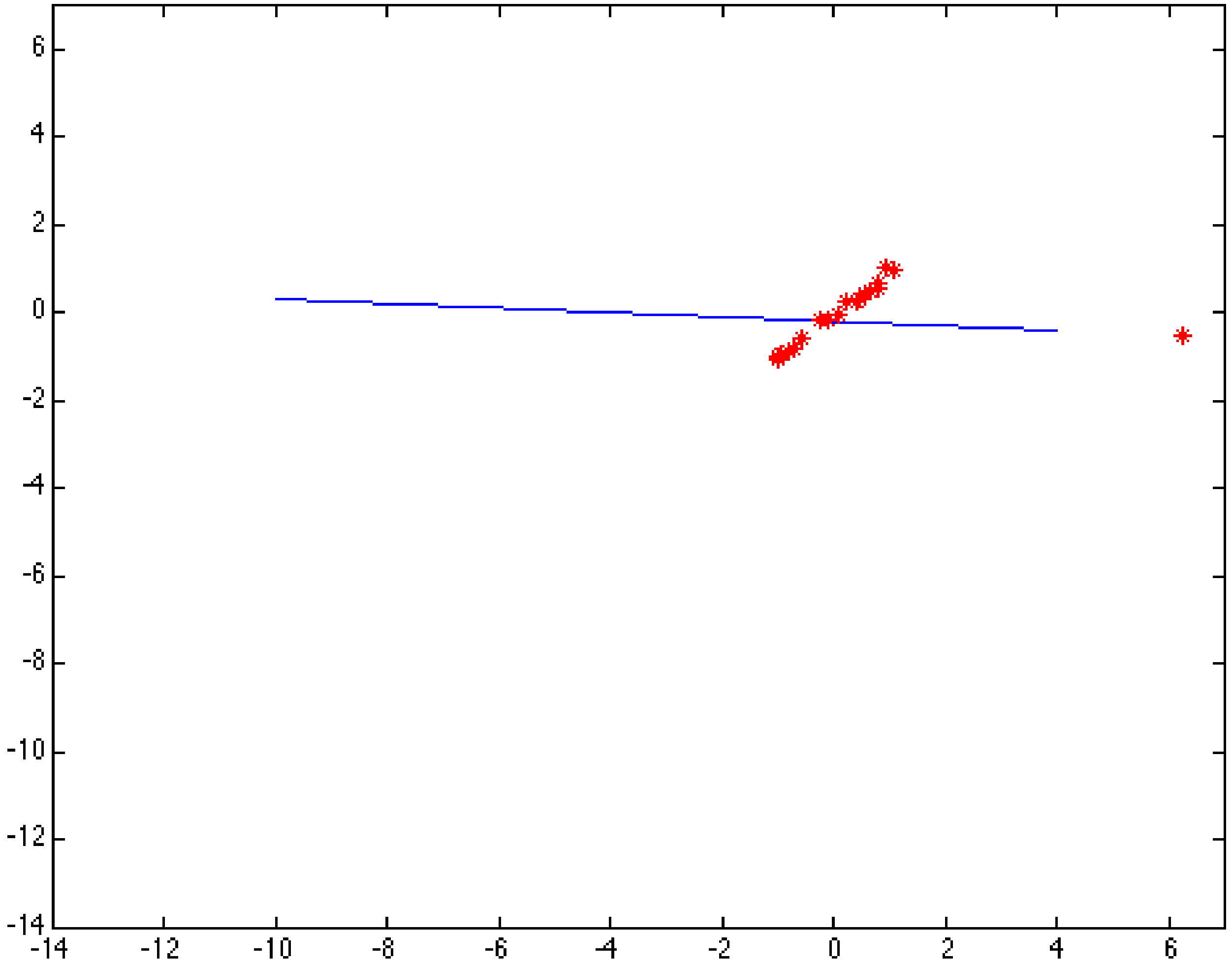
Plot showing varying *s*

$$\frac{d^2}{d^2 + s^2}$$

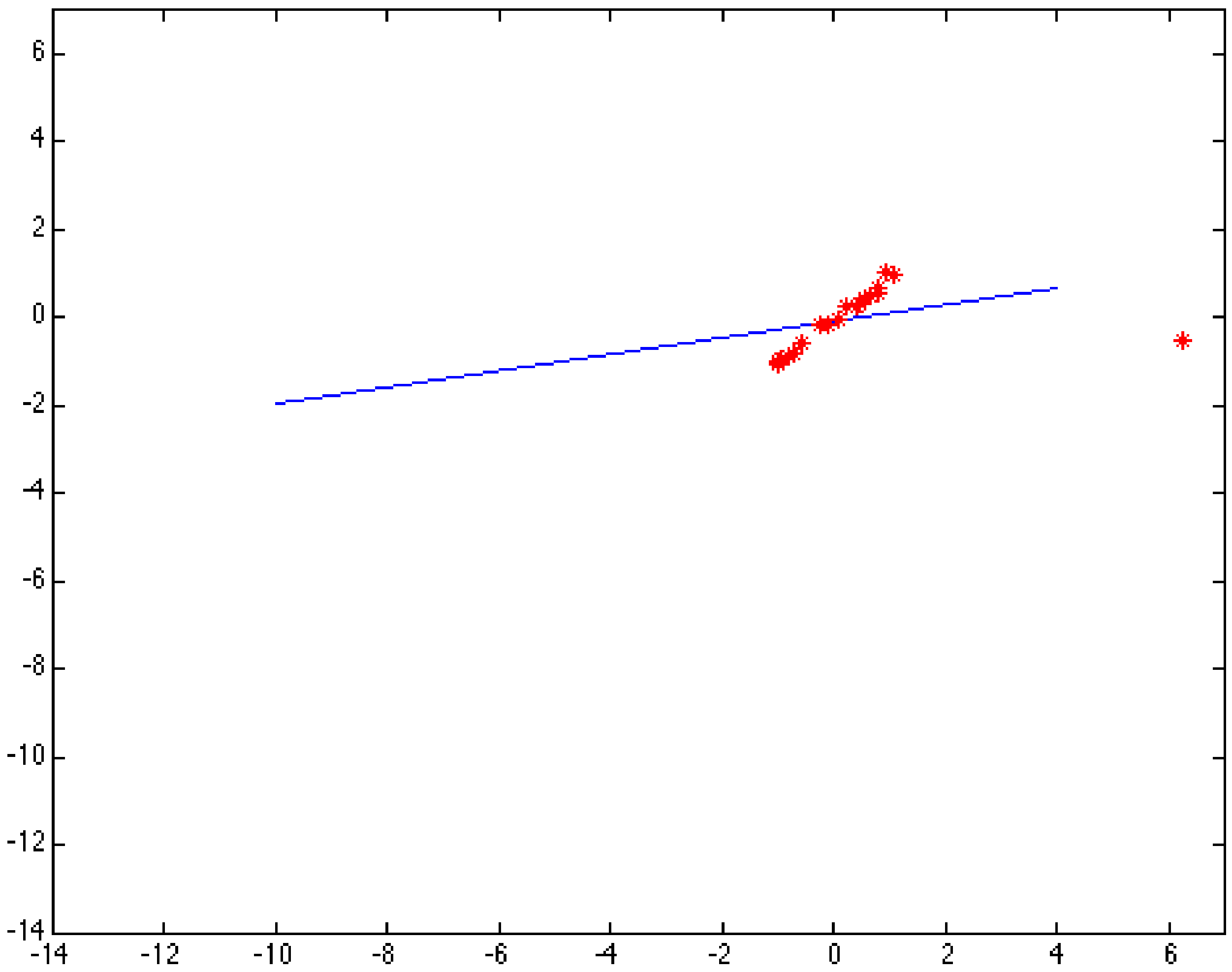



6
4
2
0
-2
-4
-6
-8
-10
-12
-14
-14
-12
-10
-8
-6
-4
-2
0
2
4
6

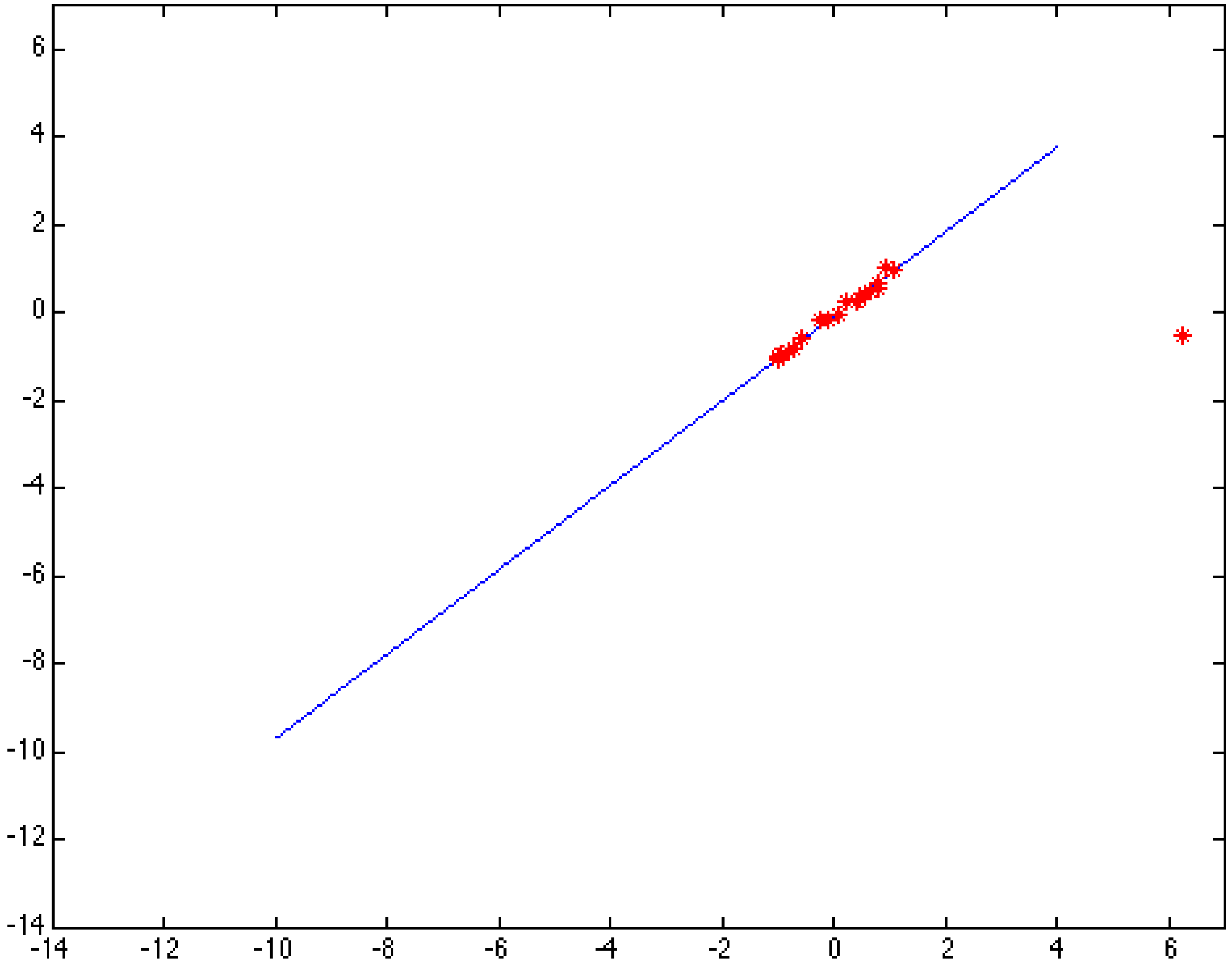
Too small – all directions equally penalized, so line could go in any direction



Too large – same as least squares, impacted by outlier

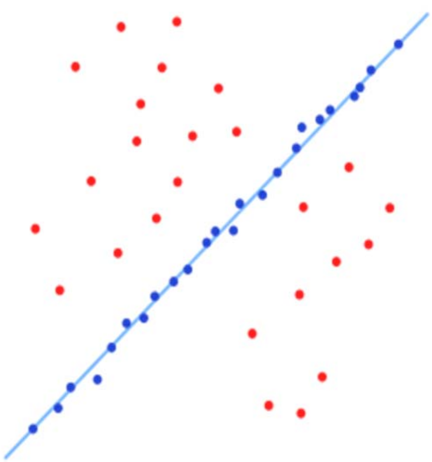


Just right – outliers don't distort fit



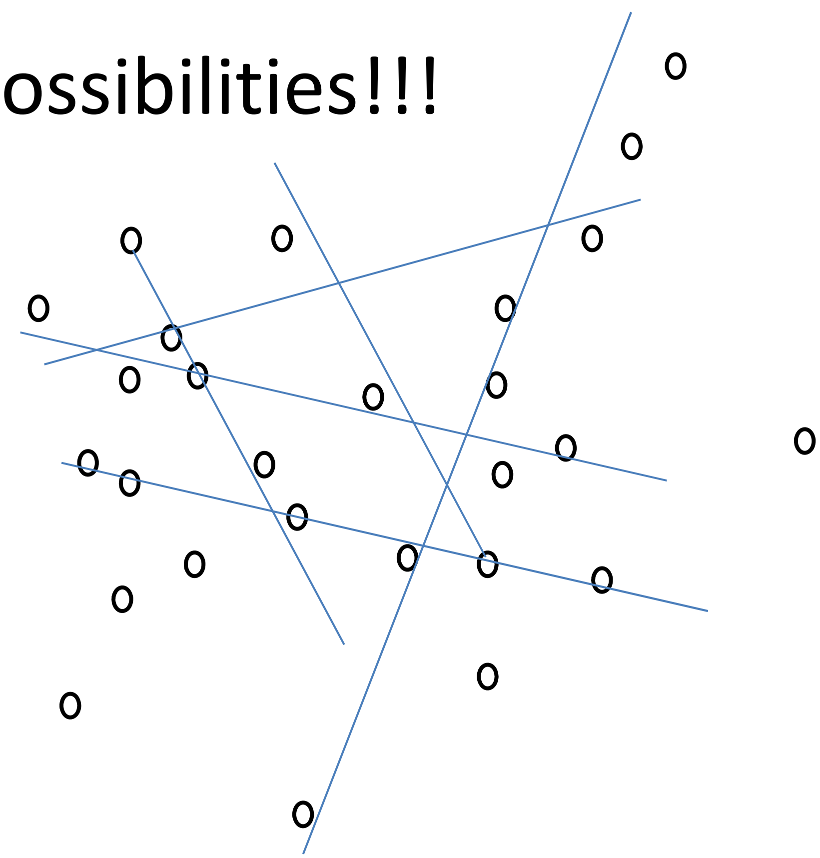
III. RANSAC

- **Ran**dom **Sa**mple **C**onsensus
- Used for Parametric Matching/Model Fitting
- Applications:



Line Fitting

- Fit the best possible Line to these points
- Brute Force Search – 2^N possibilities!!!
- Not Feasible
- Better Strategy?

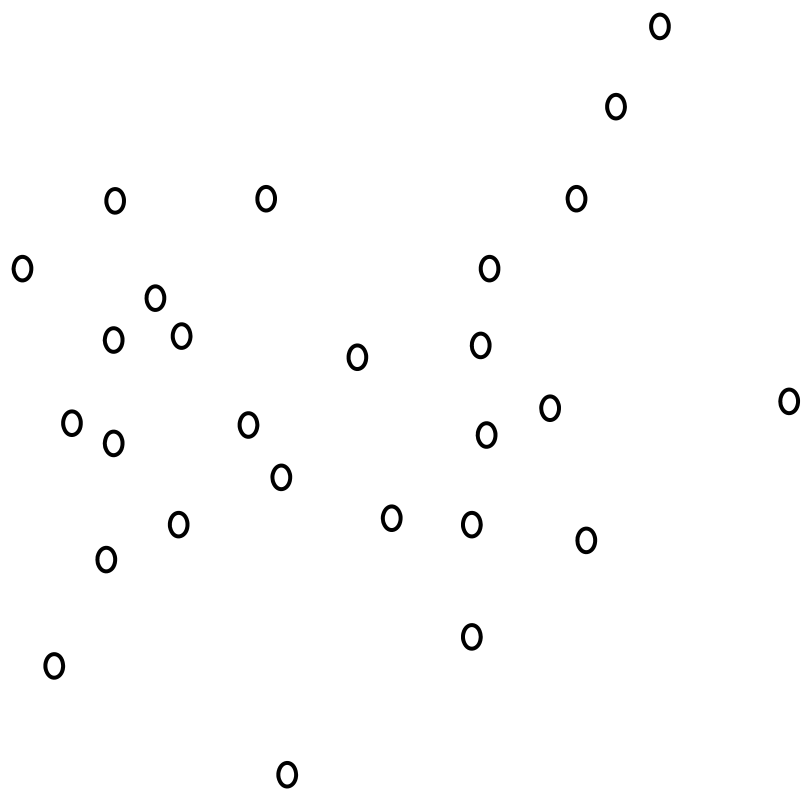


RANSAC

- Assumes the data contains both inliers and outliers
  - Select a random subset of the original data - hypothetical inliers.
  - A model is fitted to the set of hypothetical inliers.
  - All other data are then tested against the fitted model using some model-specific loss function
  - The estimated model is reasonably good if sufficiently many points have been classified as part of the consensus set.
  - Afterwards, the model may be improved by reestimating it using all members of the consensus set.

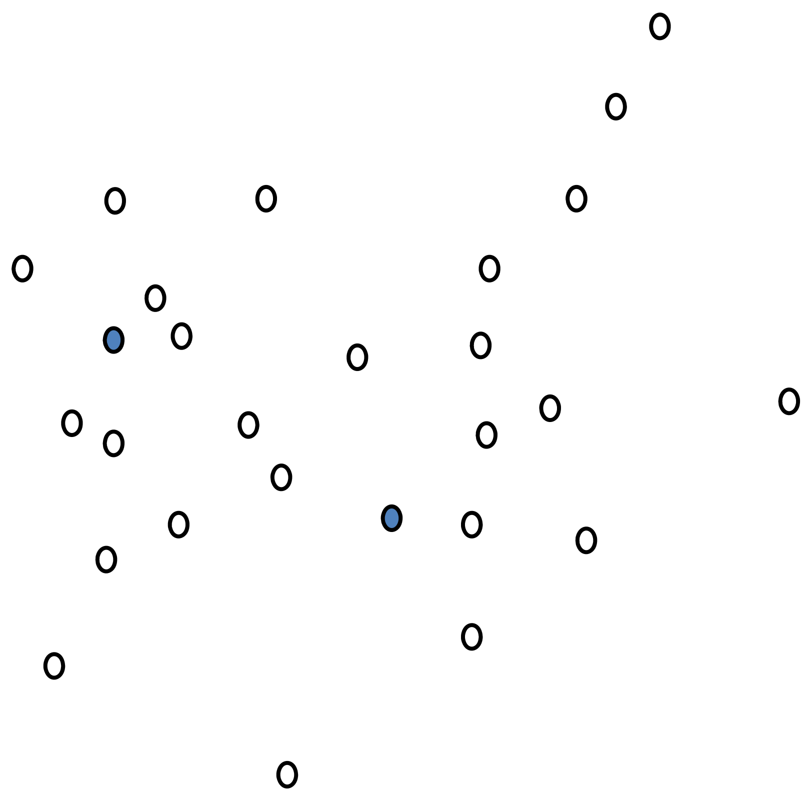
How RANSAC Works

- Random Search – Much Faster!!!



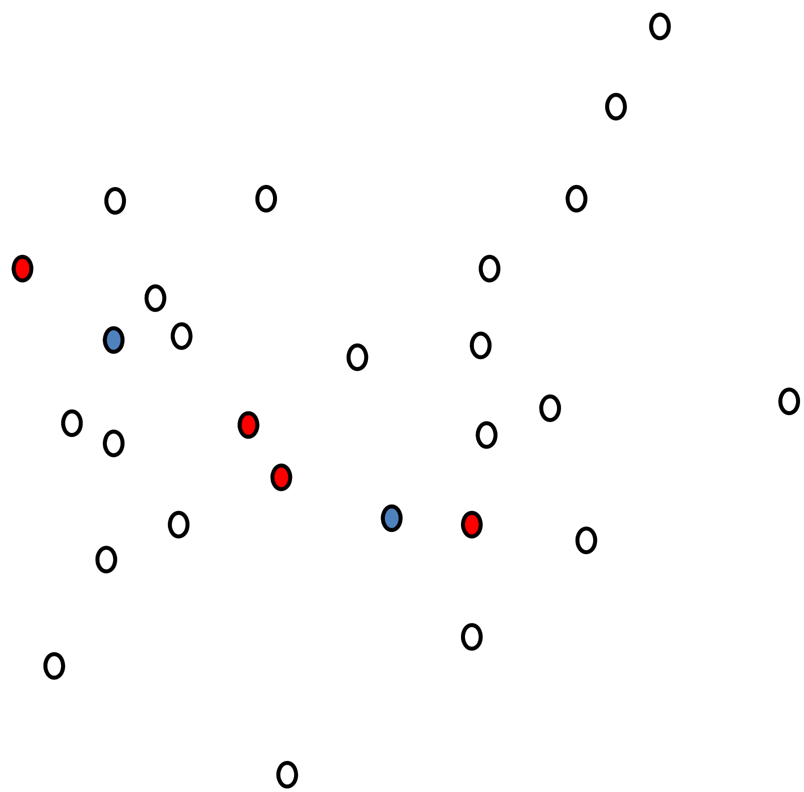
Line Fitting using RANSAC

- Iteration 1



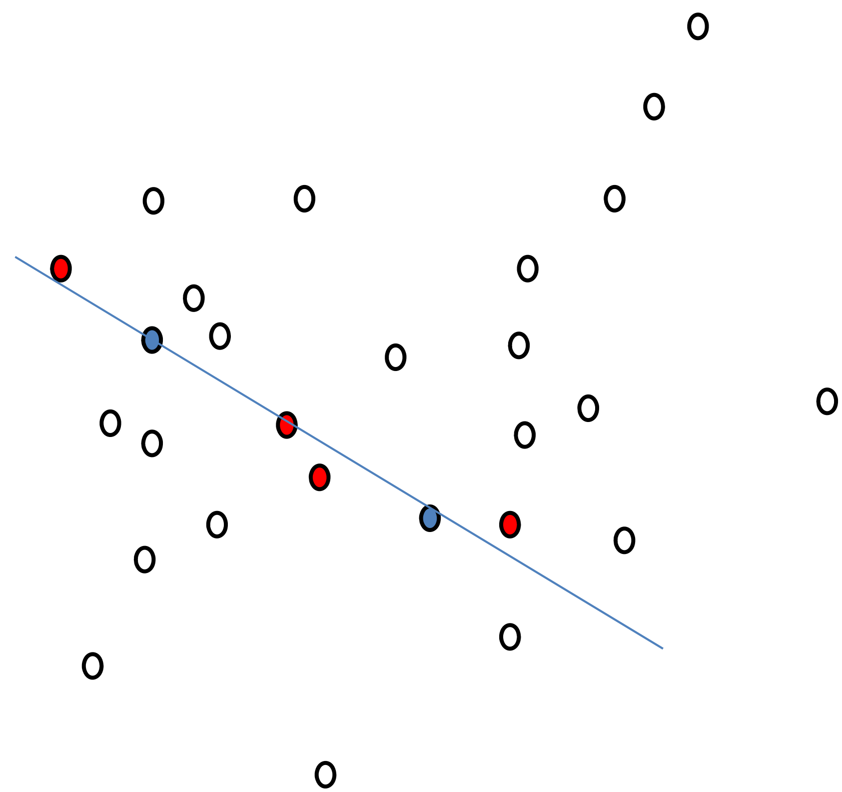
Line Fitting using RANSAC

- Iteration 1



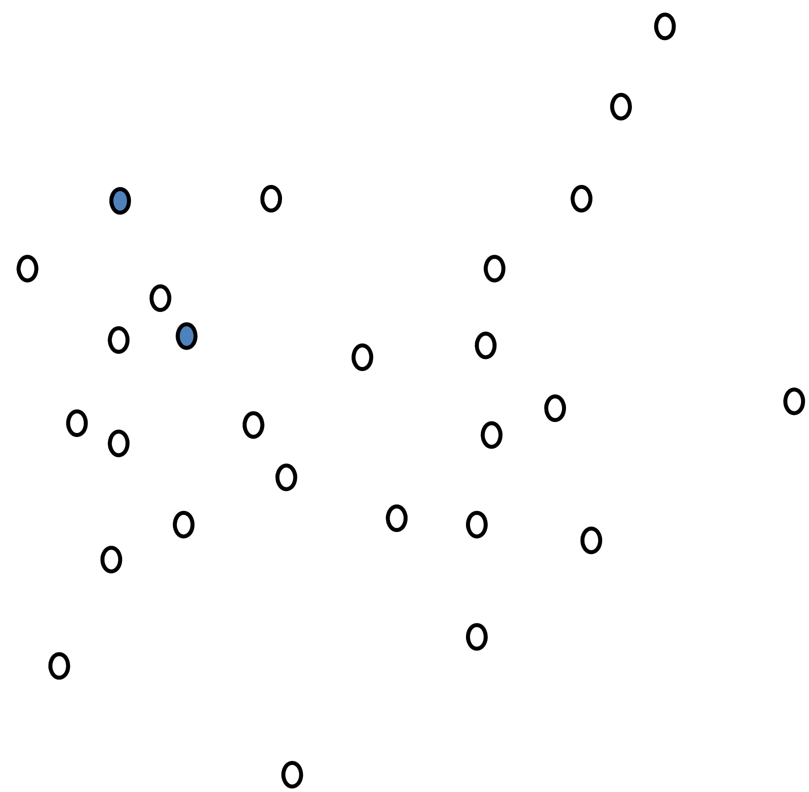
Line Fitting using RANSAC

- Iteration 1



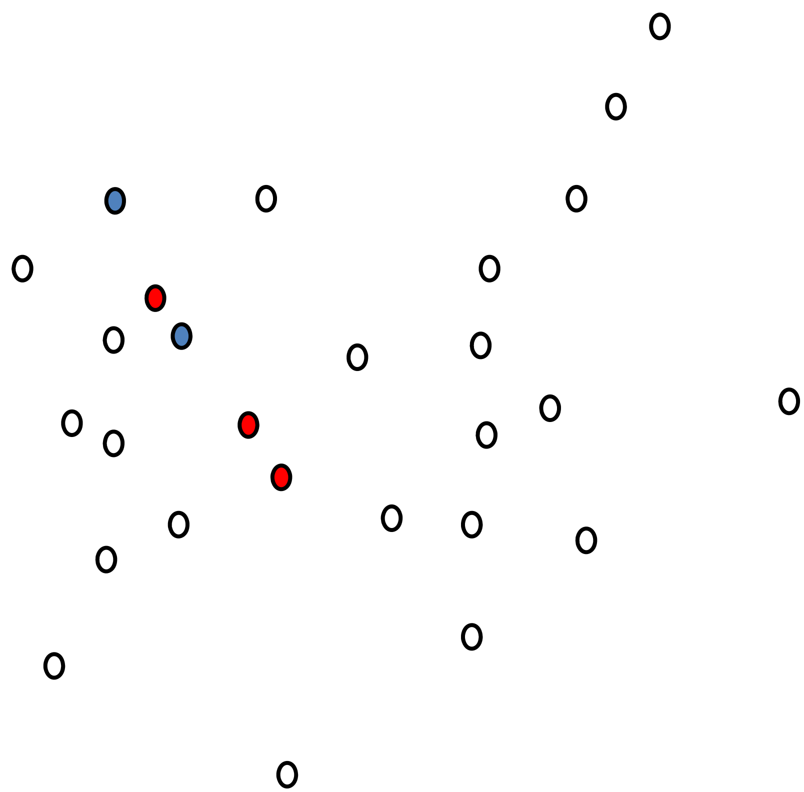
Line Fitting using RANSAC

- Iteration 2



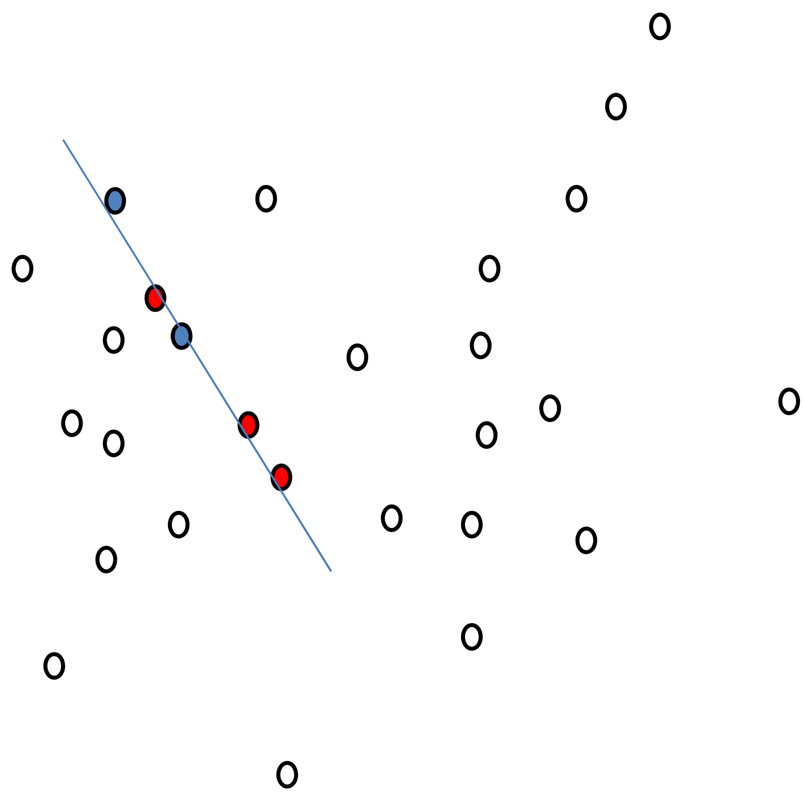
Line Fitting using RANSAC

- Iteration 2



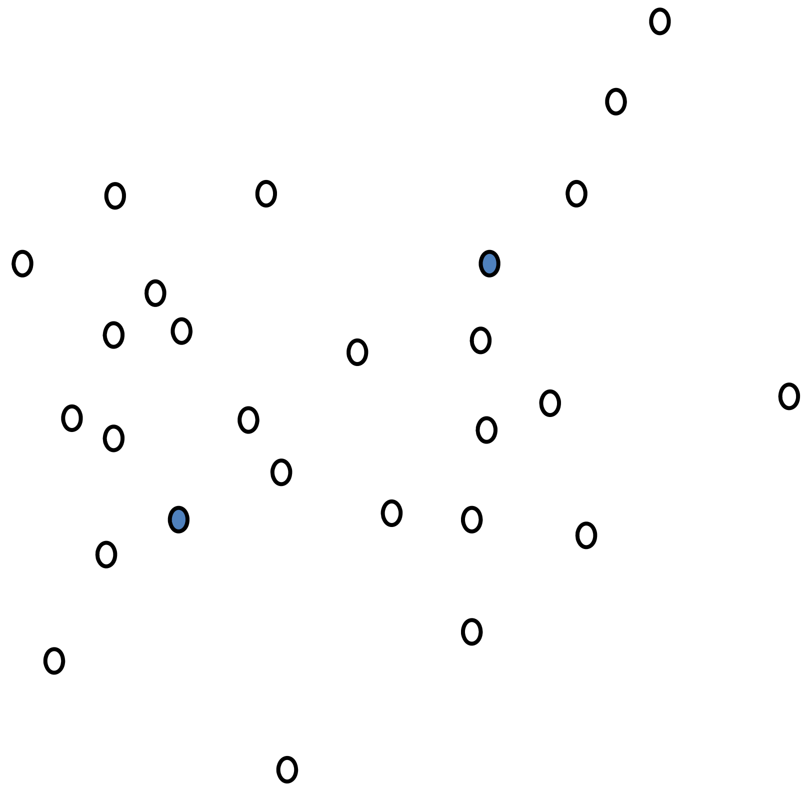
Line Fitting using RANSAC

- Iteration 2



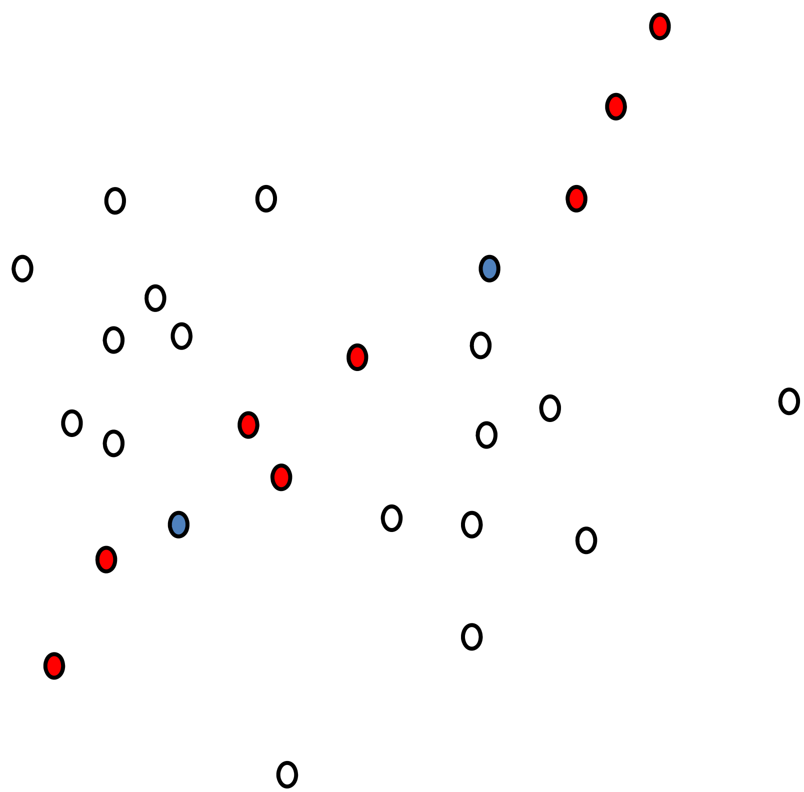
Line Fitting using RANSAC

- …
- Iteration 5



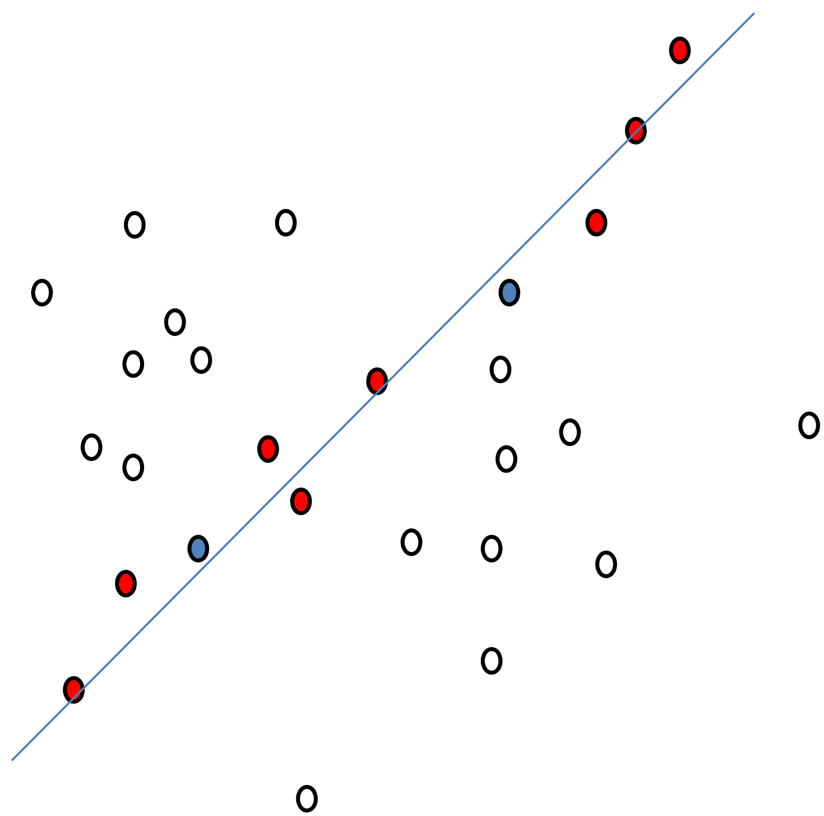
Line Fitting using RANSAC

- Iteration 5



Line Fitting using RANSAC

- Iteration 5



Why RANSAC Works?

- In general:

- $p = 1 – (1 - w^n)^k$

Where,
p = probability for selecting inliers
w = ratio of inliers to total #points
n = minimum #points required (for line = 2, circle =3)
k = #iterations

RANSAC Algorithms

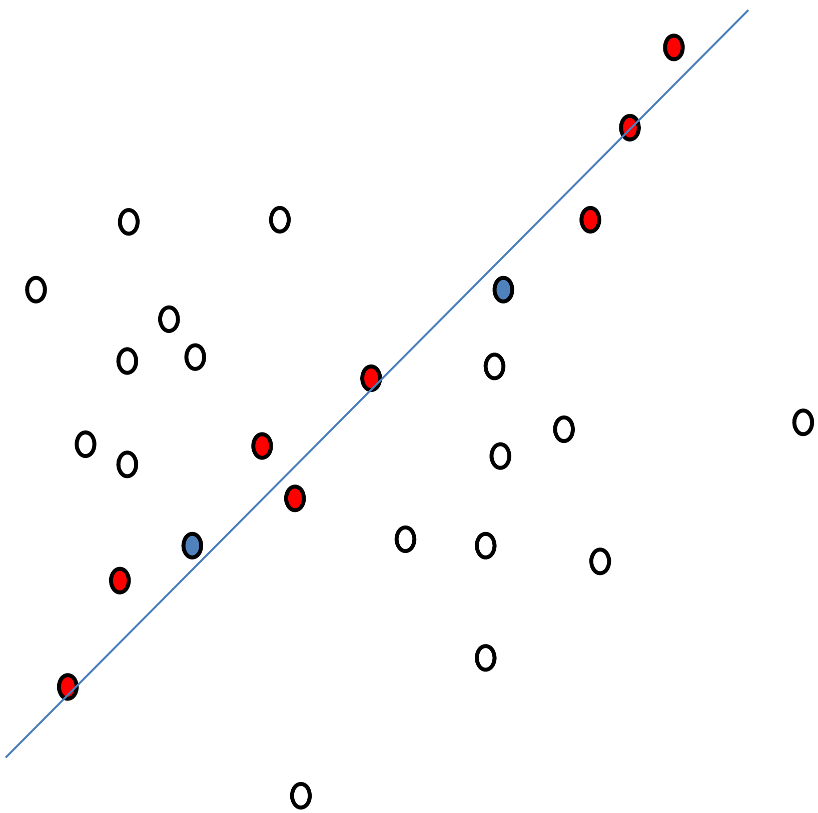
Determine:
- n—the smallest number of points required (e.g., for lines, $n = 2$, for circles, $n = 3$)
- k—the number of iterations required
- t—the threshold used to identify a point that fits well
- d—the number of nearby points required to assert a model fits well

Until k iterations have occurred
- Draw a sample of n points from the data uniformly and at random
- Fit to that set of n points
- For each data point outside the sample
  - Test the distance from the point to the structure against t; if the distance from the point to the structure is less than t, the point is close
- end
- If there are d or more points close to the structure then there is a good fit. Refit the structure using all these points. Add the result to a collection of good fits.

end

Use the best fit from this collection, using the fitting error as a criterion

Algorithm 10.4: RANSAC: Fitting Structures Using Random Sample Consensus.



Pros and Cons

\+ Robust estimation of the model parameters –

- high degree of accuracy even when a significant number of outliers are present in the data set.

\+ Useful in many advanced CV applications

\- RANSAC is not always able to find the optimal set - it usually performs badly when the number of inliers is less than 50% (better versions have been proposed)

\- No upper bound on the time it takes to compute these parameters .

\- It requires the setting of problem-specific thresholds.

\- RANSAC can only estimate one model for a particular data set.

Slide Credits

- Svetlana Lazebnik – UIUC
- Derek Hoiem – UIUC
- David Forsyth - UIUC

Questions

Next class

- **Image Alignment**
- Readings for next lecture:
  – Forsyth and Ponce chapter 12
- Readings for today:
  – Forsyth and Ponce chapter 10