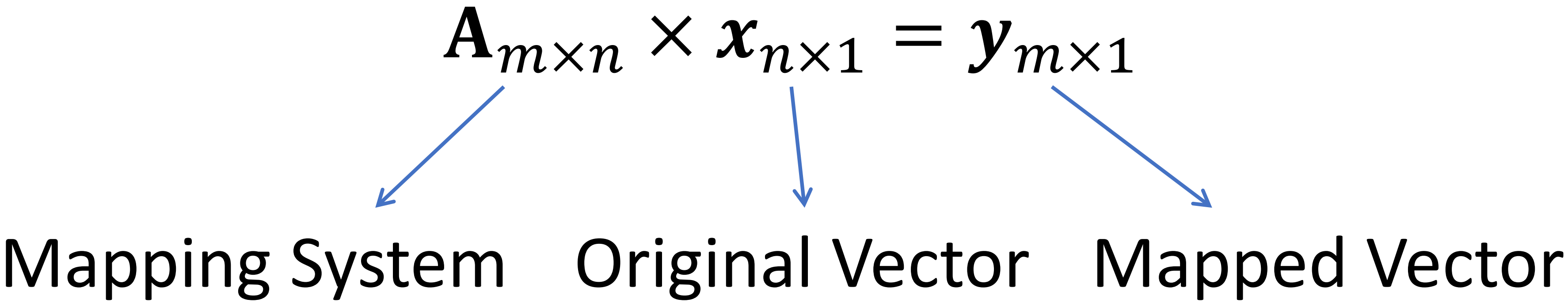
Background on Estimation for Computer Vision

Overview of the Lecture

- Eigenvalues and eigenvectors
- Eigen-Decomposition (applied to Covariance matrix)
- SVD
- Solving a (non-homogeneous) system of linear equations (LS)
- Solving a homogeneous system of linear equations (Orthog. L S)
- Line fitting with Orth. LS
- Line fitting with LS
- Line fitting using LS with Regularization
- Line fitting with RANSAC

Mapping of Vectors

A matrix is a mapping system of vectors!



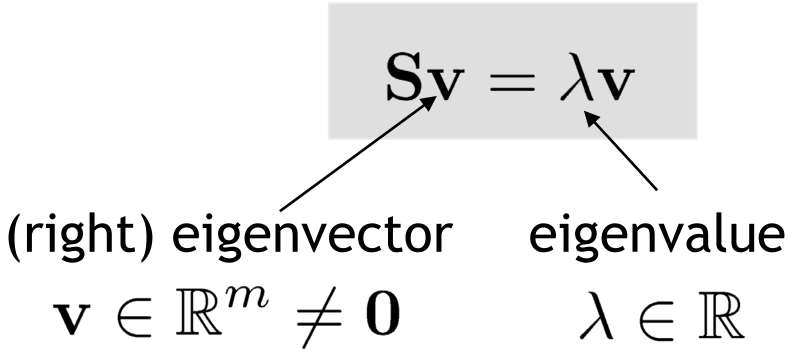
$$\mathbf{A}_{m\times n} \times \boldsymbol{x}_{n\times 1} = \boldsymbol{y}_{m\times 1}$$

Mapping System Original Vector Mapped Vector

How does it work?

Eigenvalues & Eigenvectors

- **Eigenvectors** (for a square $m \times m$ matrix $\mathbf{S}$)



$$\mathbf{S}\mathbf{v} = \lambda \mathbf{v}$$

(right) eigenvector

$\mathbf{v} \in \mathbb{R}^m \neq \mathbf{0}$

eigenvalue

$\lambda \in \mathbb{R}$

Example

$$\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

- How many eigenvalues are there at most?

$$\mathbf{S}\mathbf{v} = \lambda \mathbf{v} \iff (\mathbf{S} - \lambda \mathbf{I})\,\mathbf{v} = \mathbf{0}$$

only has a non-zero solution if $\quad |\mathbf{S} - \lambda \mathbf{I}| = 0$

This is a m-th order equation in λ which can have at most m distinct solutions (roots of the characteristic polynomial) – can be complex even though S is real.

Matrix-vector multiplication

$$S = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

has eigenvalues 3, 2, 0 with corresponding eigenvectors

$$v_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \qquad v_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \qquad v_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

On each eigenvector, S acts as a multiple of the identity matrix: but as a different multiple on each.

Any vector (say $x=\begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$) can be viewed as a combination of the eigenvectors: $x = 2v_1 + 4v_2 + 6v_3$

Eigenvalues & Eigenvectors

For symmetric matrices, eigenvectors for distinct eigenvalues are **orthogonal**

$$Sv_{\{1,2\}} = \lambda_{\{1,2\}} v_{\{1,2\}}, \text{ and } \lambda_1 \neq \lambda_2 \Rightarrow v_1 \bullet v_2 = 0$$

All eigenvalues of a real symmetric matrix are **real**.

$$\text{for complex } \lambda, \text{ if } |S - \lambda I| = 0 \text{ and } \mathrm{S} = \mathrm{S}^{\mathrm{T}} \Rightarrow \lambda \in \Re$$

All eigenvalues of a positive semidefinite matrix are **non-negative**

$$\overbrace{\forall w \in \Re^n, w^T S w \geq 0,} \text{ then if } \mathrm{Sv} = \lambda \mathrm{v} \Rightarrow \lambda \geq 0$$

Example

- Let

$$S = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

Real, symmetric.

- Then

$$S - \lambda I = \begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix} \Rightarrow (2-\lambda)^2 - 1 = 0.$$

- The eigenvalues are 1 and 3 (nonnegative, real).
- The eigenvectors are orthogonal (and real):

$$\begin{pmatrix} 1 \\ -1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Plug in these values and solve for eigenvectors.

Eigen/diagonal Decomposition

- Let $\mathbf{S} \in \mathbb{R}^{m \times m}$ be a **square** matrix with ***m*** **linearly independent eigenvectors** (a "non-defective" matrix)
- **Theorem**: Exists an **eigen decomposition**

$$\mathbf{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$$

diagonal

Unique for distinct eigen-values

  - (cf. matrix diagonalization theorem)
- Columns of ***U*** are **eigenvectors** of ***S***
- Diagonal elements of $\mathbf{\Lambda}$ are **eigenvalues** of $\mathbf{S}$

$$\mathbf{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_m), \quad \lambda_i \geq \lambda_{i+1}$$

Symmetric Eigen Decomposition

- If $\mathrm{S} \in \mathbb{R}^{m\times m}$ is a **symmetric** matrix:
- **Theorem**: Exists a (unique) **eigen decomposition**

$$S = Q\Lambda Q^T$$

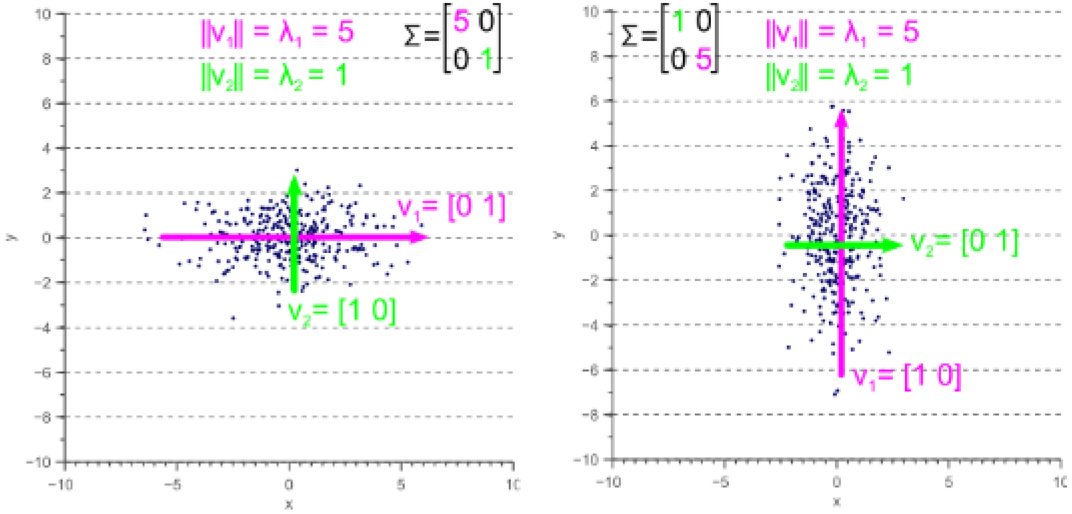
- where ***Q*** is **orthogonal:**
  - $\boldsymbol{Q^{-1} = Q^T}$
  - Columns of ***Q*** are normalized eigenvectors
  - Columns are orthogonal.
  - (everything is real)

If S is co-variance matrix

$$S = \frac{1}{n}\sum_i (x_i - \bar{x})(x_i - \bar{x})^T$$ where $\bar{x}$ is the mean of the set of points $\{x_i\}$

Eigenvalues represent covariances

Eigenvectors represent linearly independent directions of variation in data



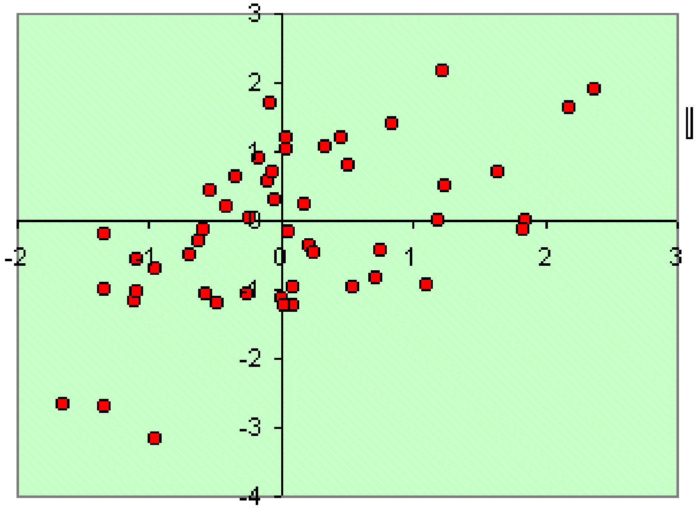
Eigenvectors of a covariance matrix of data shown in blue. Data is drawn from a gaussian distribution.

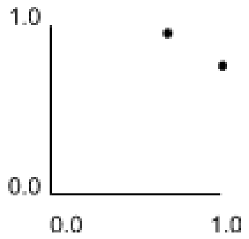
Physical interpretation

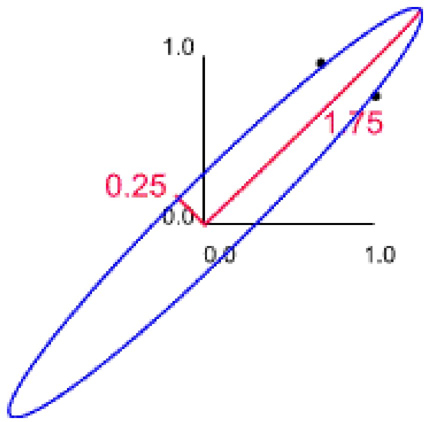
- Consider a covariance matrix, **S**, i.e., S = 1/n A A^T for some A

$$\mathbf{S} = \begin{bmatrix} 1 & .75 \\ .75 & 1 \end{bmatrix} \Rightarrow \lambda_1 = 1.75, \lambda_2 = 0.25$$

1.00	0.75
0.75	1.00

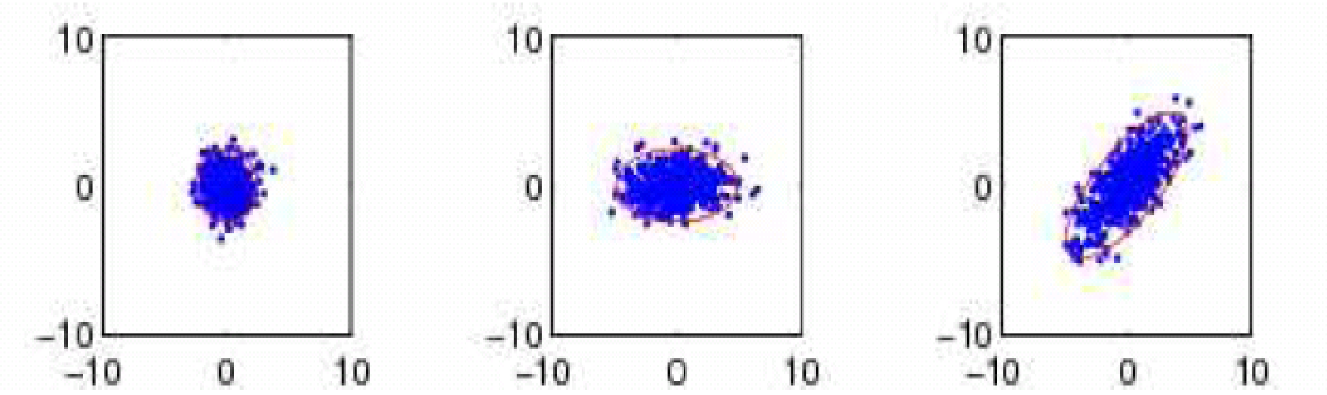




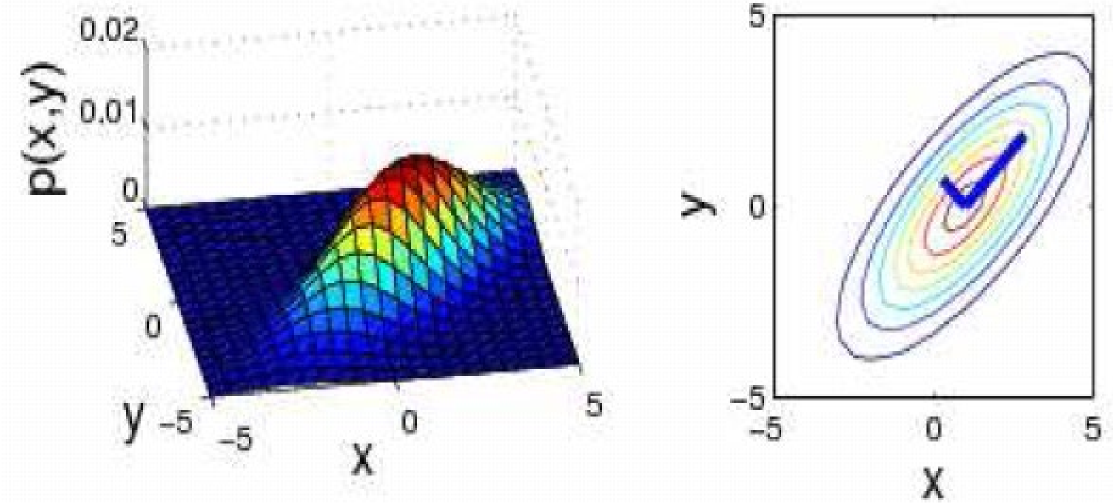


- Error ellipse with the major axis as the larger eigenvalue and the minor axis as the smaller eigenvalue

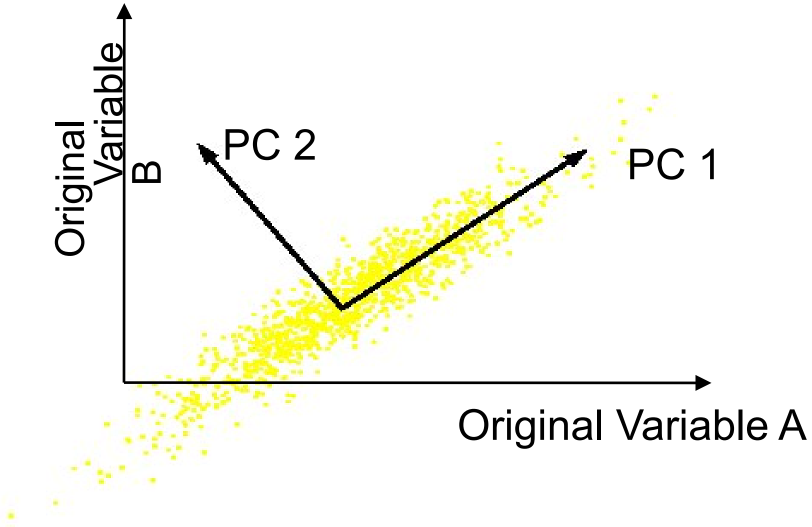
Spherical, diagonal, full covariance



$$\Sigma = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{pmatrix} \qquad \Sigma = \begin{pmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{pmatrix}$$



The concept is used in Principal Component Analysis (PCA)



- Orthogonal directions of greatest variance in data
- Projections along PC1 (Principal Component) discriminate the data most along any one axis

Singular Value Decomposition

For an $m \times n$ matrix $\mathbf{A}$ of rank r there exists a factorization (Singular Value Decomposition = **SVD**) as follows:

$$A = \underset{\boxed{m\times m}}{\underset{\nearrow}{U}}\,\underset{\boxed{m\times n}}{\underset{\uparrow}{\Sigma}}\,\underset{\boxed{V \text{ is } n\times n}}{\underset{\nwarrow}{V^T}}$$

The columns of ***U*** are orthogonal eigenvectors of $\boldsymbol{AA^T}$.

The columns of ***V*** are orthogonal eigenvectors of $\boldsymbol{A^TA}$.

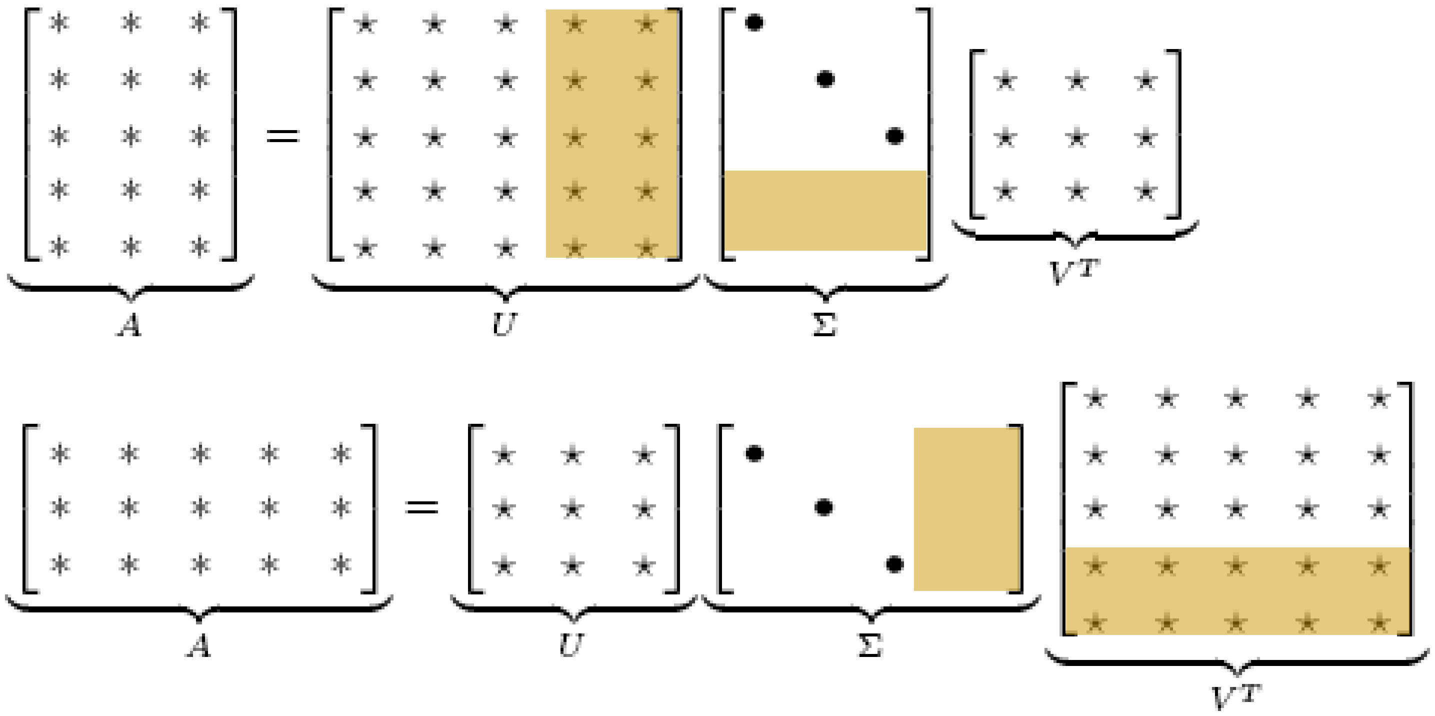
Eigenvalues $\lambda_1 \ldots \lambda_r$ of $\boldsymbol{AA^T}$ are the eigenvalues of $\boldsymbol{A^TA}$.

$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = diag\left(\sigma_1 \ldots \sigma_r\right) \quad \longleftarrow \boxed{\textit{Singular values.}}$$

Singular Value Decomposition

- Illustration of SVD dimensions and sparseness



SVD example

Let $$A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Thus *m*=3, *n*=2. Its SVD is

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Typically, the singular values arranged in decreasing order.

Visualization of multiplying with A

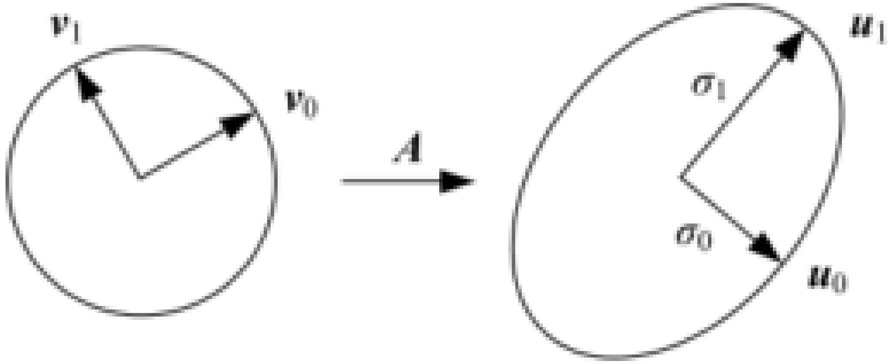


Figure A.1 The action of a matrix $\boldsymbol{A}$ can be visualized by thinking of the domain as being spanned by a set of orthonormal vectors $\boldsymbol{v}_j$, each of which is transformed to a new orthogonal vector $\boldsymbol{u}_j$ with a length σ_j. When $\boldsymbol{A}$ is interpreted as a covariance matrix and its eigenvalue decomposition is performed, each of the $\boldsymbol{u}_j$ axes denote a principal direction (component) and each σ_j denotes one standard deviation along that direction.

Applications of SVD in Linear Algebra

- Inverse of a n x n square matrix, **A**
  - If **A** is non-singular, then $\mathbf{A}^{-1} = (\mathbf{U\Lambda V}^{T})^{-1} = \mathbf{V\Lambda}^{-1}\mathbf{U}^{T}$ where $\Lambda^{-1}=\mathrm{diag}(1/\lambda_1, 1/\lambda_1,\cdots, 1/\lambda_n)$
  - If **A** is singular, then $\mathbf{A}^{-1} = (\mathbf{U\Lambda V}^{T})^{-1}$¼ $\mathbf{V\Lambda}_0^{-1}\mathbf{U}^{T}$ where $\Lambda_0^{-1}=\mathrm{diag}(1/\lambda_1, 1/\lambda_2,\cdots, 1/\lambda_i,0,0,\cdots,0)$
- Least squares solutions of a mxn system
  - $\mathbf{Ax}=\mathbf{b}$ (**A** is mxn, m¸n) $=(\mathbf{A}^{T}\mathbf{A})\mathbf{x}=\mathbf{A}^{T}\mathbf{b}$ **)** $\mathbf{x}=(\mathbf{A}^{T}\mathbf{A})^{-1}\ \mathbf{A}^{T}\mathbf{b}=\mathbf{A}^{+}\mathbf{b}$
  - If $\mathbf{A}^{T}\mathbf{A}$ is singular, $\mathbf{x}=\mathbf{A}^{+}\mathbf{b}$ **=** $(\mathbf{V\Lambda_0}^{-1}\mathbf{U}^{T})\mathbf{b}$ where $\Lambda_0^{-1} = \mathrm{diag}(1/\lambda_1, 1/\lambda_2,\cdots, 1/\lambda_i,0,0,\cdots,0)$
- Condition of a matrix
  - Condition number measures the degree of singularity of **A**
    - Larger the value of λ_1/λ_n, closer **A** is to being singular

http://www.cse.unr.edu/~bebis/MathMethods/SVD/lecture.pdf

Line Fitting:

- If we know which points belong to the line, how do we find the “optimal” line parameters?
    - Least squares

- What if there are outliers?
    - Robust fitting, RANSAC

- What if there are many lines?
    - Voting methods: RANSAC, Hough transform

- What if we’re not even sure it’s a line?
    - Model selection

Linear equations
solving with Standard Least Squares (LS)

Ax = b

$$\begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 0 & 0 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix}$$

$$e^2 = \|A\mathbf{x} - b\|^2$$

$$A\mathbf{x} = b$$

$$A^T A\mathbf{x} = A^T b$$ Normal equations

$$\bar{\mathbf{x}} = (A^T A)^{-1} A^T b$$

•If A has linearly independent columns, A^TA is square, symmetric and invertible

$$A^\dagger = (A^T A)^{-1} A^T$$

is so called pseudoinverse of matix A

Homogeneous Systems of equations

$$A\mathbf{x} = 0$$

There is a unique trivial solution x = 0

We need to impose some constraint to avoid trivial Solution, for example

$$||\mathbf{x}|| = 1$$

Find such x that $||A\mathbf{x}||^2$ is minimized

$$||A\mathbf{x}||^2 = \mathbf{x}A^T A\mathbf{x}$$

Solution: eigenvector associated with the smallest eigenvalue of A^TA

or singular vector associated with smallest singular value

$A = U\Sigma V^T$: Since singular values are sorted from large to small, this is the last column of V (last row of V^T)

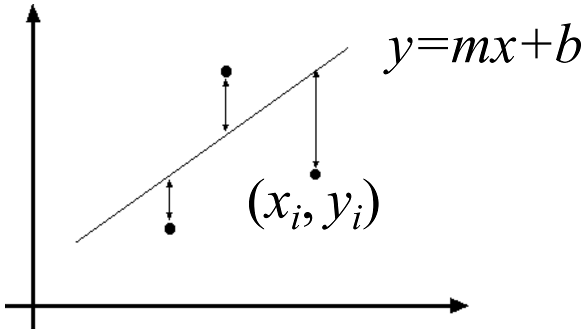
Least squares line fitting

Data: $(x_1, y_1), \ldots, (x_n, y_n)$

Line equation: $y_i = m\,x_i + b$

Find (m, b) to minimize

$$\boxed{E = \sum_{i=1}^{n} (y_i - mx_i - b)^2}$$



Type equation here.

$$Y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \qquad X = \begin{bmatrix} x_1 & 1 \\ \vdots & \vdots \\ x_n & 1 \end{bmatrix} \qquad B = \begin{bmatrix} m \\ b \end{bmatrix}$$

$$E = \|Y - XB\|^2 = (Y - XB)^T (Y - XB) = Y^T Y - 2(XB)^T Y + (XB)^T (XB)$$

$$\frac{dE}{dB} = 2X^T XB - 2X^T Y = 0$$

$$\boxed{X^T XB = X^T Y}$$

Normal equations: Least Squares solution to $XB=Y$

$$\boxed{B = (X^T X)^{-1} (X^T Y)}$$

Problem with “vertical” least squares

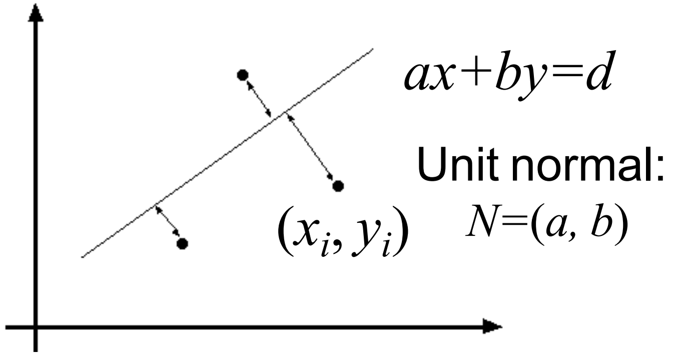
- Not rotation-invariant
- Fails completely for vertical lines

Total least squares (or Orthogonal Least Squares)

Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$

Find (a, b, d) to minimize the sum of squared perpendicular distances

$$\boxed{E = \sum_{i=1}^{n} (ax_i + by_i - d)^2}$$



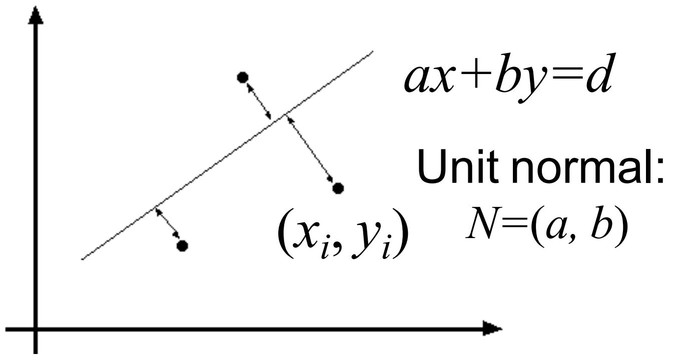
Not stable to find three values (which could have eigenvalues of very different value; small eigenvalues are critical)

Instead, since the best fit line must pass through the center of mass of the set of points, center the data (move mass center to origin).
This reduces the problem to finding 2 parameters (the unit vector normal to the line).
Then solve for d.

Total least squares

Distance between point (x_i, y_i) and line $ax+by=d$ $(a^2+b^2=1)$: $|ax_i + by_i - d|$

Find (a, b, d) to minimize the sum of squared perpendicular distances



$$\boxed{E=\sum_{i=1}^{n}(ax_i+by_i-d)^2}$$

$$\frac{\partial E}{\partial d}=\sum_{i=1}^{n}-2(ax_i+by_i-d)=0 \qquad d=\frac{a}{n}\sum_{i=1}^{n}x_i+\frac{b}{n}\sum_{i=1}^{n}y_i=a\bar{x}+b\bar{y}$$

Solve linear equations in (a,b) only

$$E=\sum_{i=1}^{n}(a(x_i-\bar{x})+b(y_i-y))^2 = \left\|\begin{bmatrix} x_1-\bar{x} & y_1-\bar{y} \\ \vdots & \vdots \\ x_n-\bar{x} & y_n-\bar{y}\end{bmatrix}\begin{bmatrix} a \\ b\end{bmatrix}\right\|^2 = (UN)^T(UN)$$

$$\frac{dE}{dN}=2(U^TU)N=0$$

Solution to $(U^TU)N = 0$, subject to $||N||^2 = 1$: eigenvector of U^TU associated with the smallest eigenvalue (least squares solution to *homogeneous linear system* $UN = 0$)

Total least squares

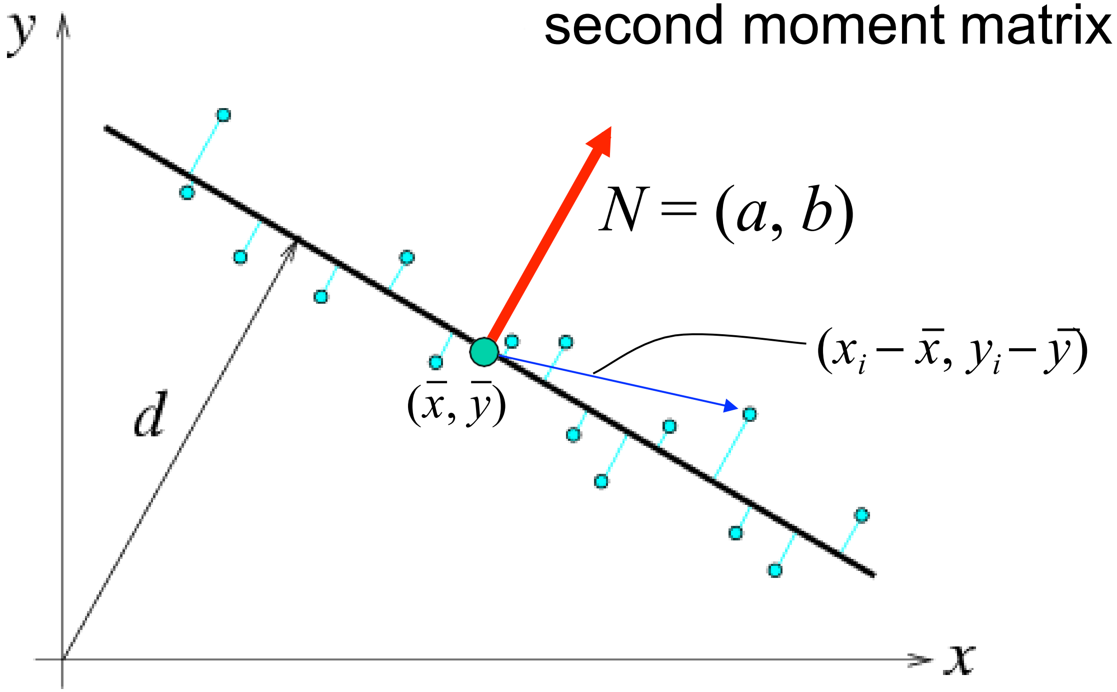
$$U = \begin{bmatrix} x_1 - \overline{x} & y_1 - \overline{y} \\ \vdots & \vdots \\ x_n - \overline{x} & y_n - \overline{y} \end{bmatrix} \quad U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \overline{x})^2 & \sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y}) \\ \sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y}) & \sum_{i=1}^{n} (y_i - \overline{y})^2 \end{bmatrix}$$

second moment matrix

Total least squares

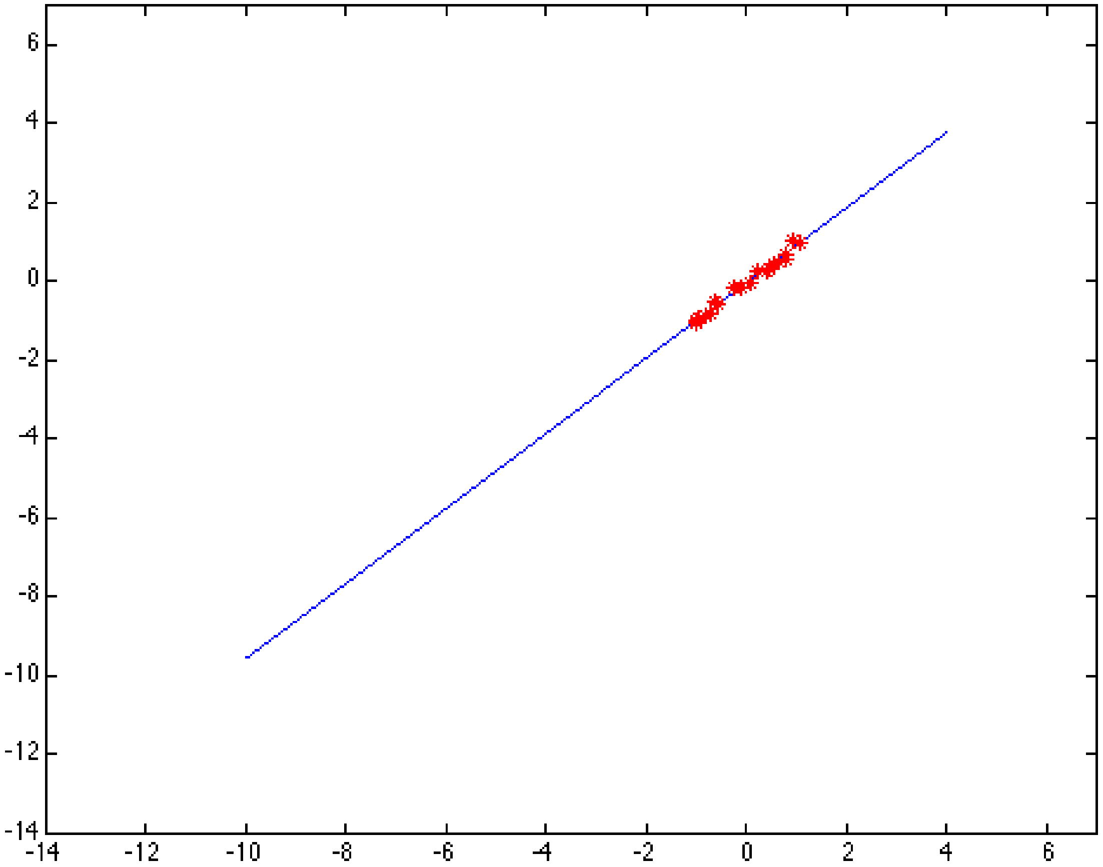
$$U = \begin{bmatrix} x_1 - \bar{x} & y_1 - \bar{y} \\ \vdots & \vdots \\ x_n - \bar{x} & y_n - \bar{y} \end{bmatrix} \qquad U^T U = \begin{bmatrix} \sum_{i=1}^{n} (x_i - \bar{x})^2 & \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \\ \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) & \sum_{i=1}^{n} (y_i - \bar{y})^2 \end{bmatrix}$$

second moment matrix



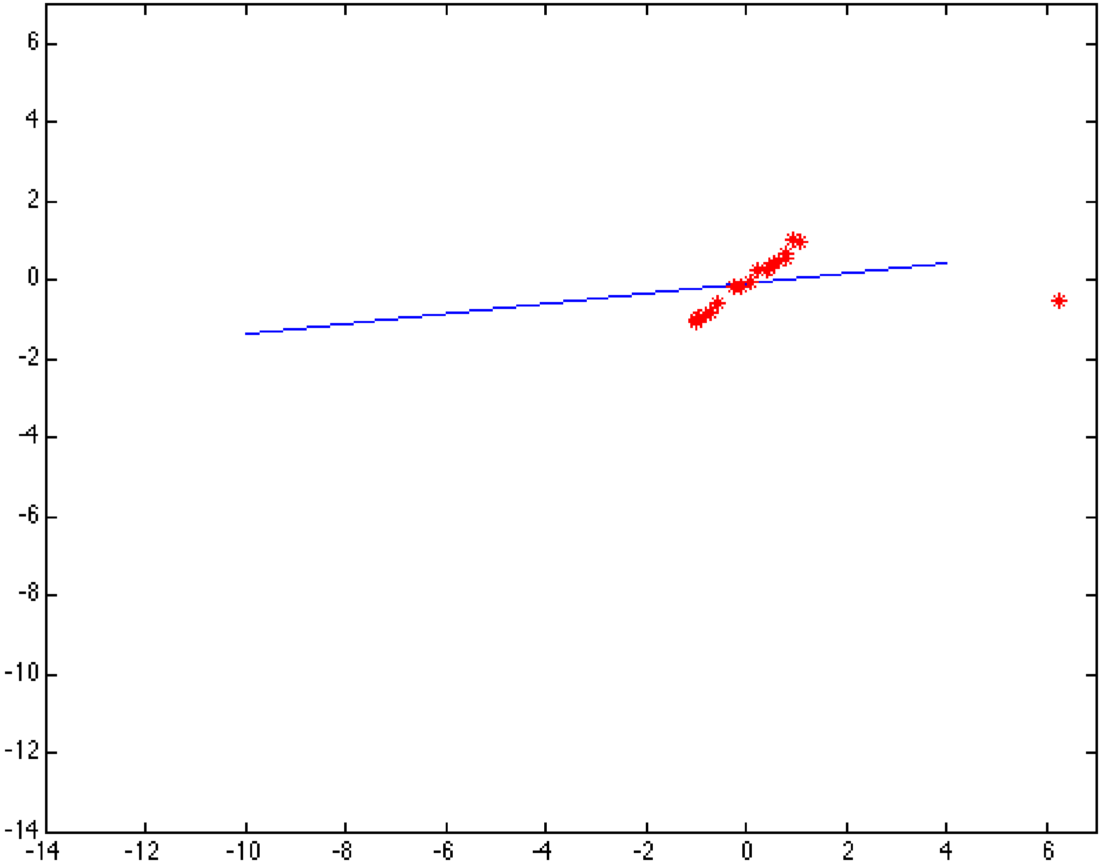
Least squares: Robustness to noise

Least squares fit to the red points:



Least squares: Robustness to noise

Least squares fit with an outlier:



Problem: squared error heavily penalizes outliers

Least Squares Fitting with Regularization

Noise is amplified by inverse of Σ (matrix of eigenvalues)

Sensitivity of a matrix is characterized by condition number

(large condition number causes large noise amplification)

$$\kappa = \frac{\sigma_{max}}{\sigma_{min}} = \|\frac{\lambda_{max}}{\lambda_{min}}\|$$

Instead of solving

$$\arg\min_x \|b - Ax\|$$

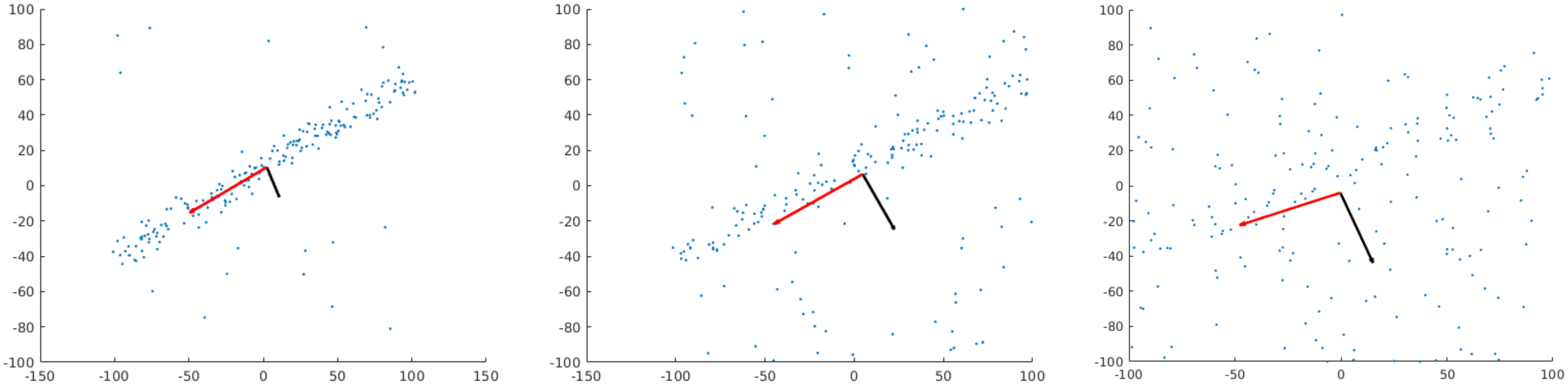
We solve

$$\arg\min_x \|b - Ax\| + \lambda\|x\|^2$$

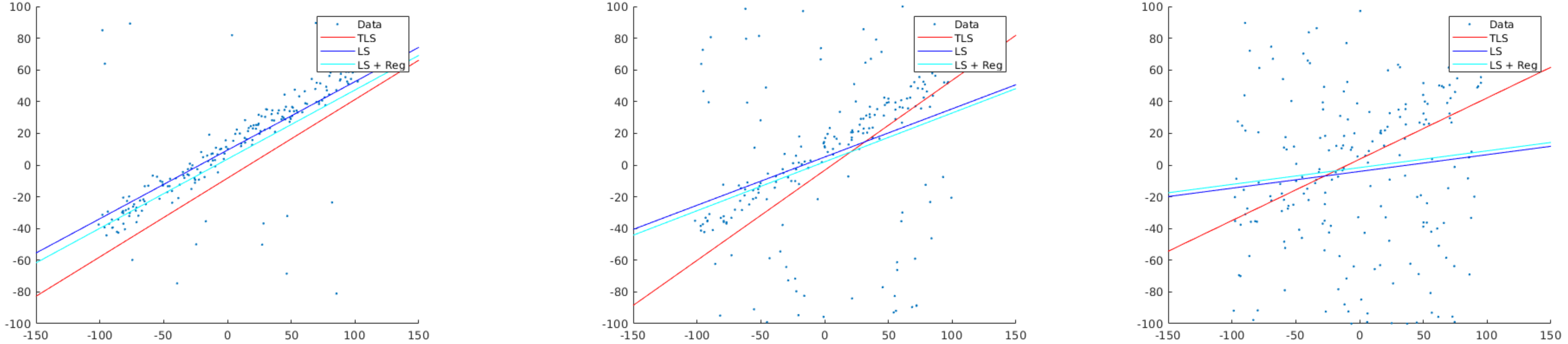
Solution

$$x = (A^T A + \lambda I)^{-1} A^T b$$

Covariance od Data in HW1



Fitting with LS, TLS and LS + Regularization

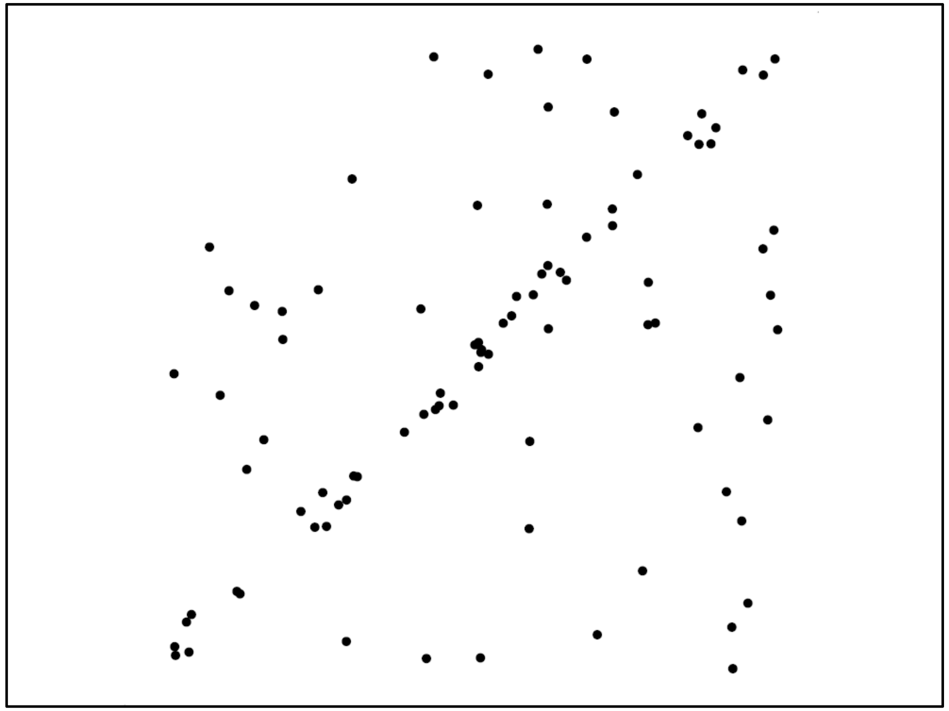


RANSAC

- Robust fitting can deal with a few outliers – what if we have very many?
- Random sample consensus (RANSAC): Very general framework for model fitting in the presence of outliers
- Outline
  - Choose a small subset of points uniformly at random
  - Fit a model to that subset
  - Find all remaining points that are "close" to the model and reject the rest as outliers
  - Do this many times and choose the best model

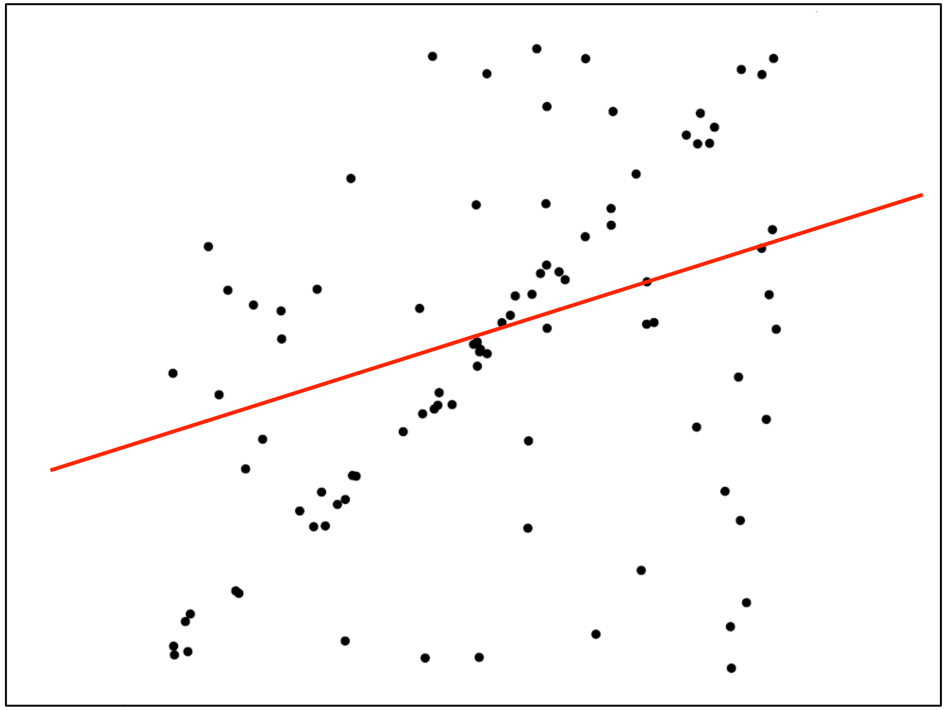
M. A. Fischler, R. C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM, Vol 24, pp 381-395, 1981.

RANSAC for line fitting example



Source: R. Raguram

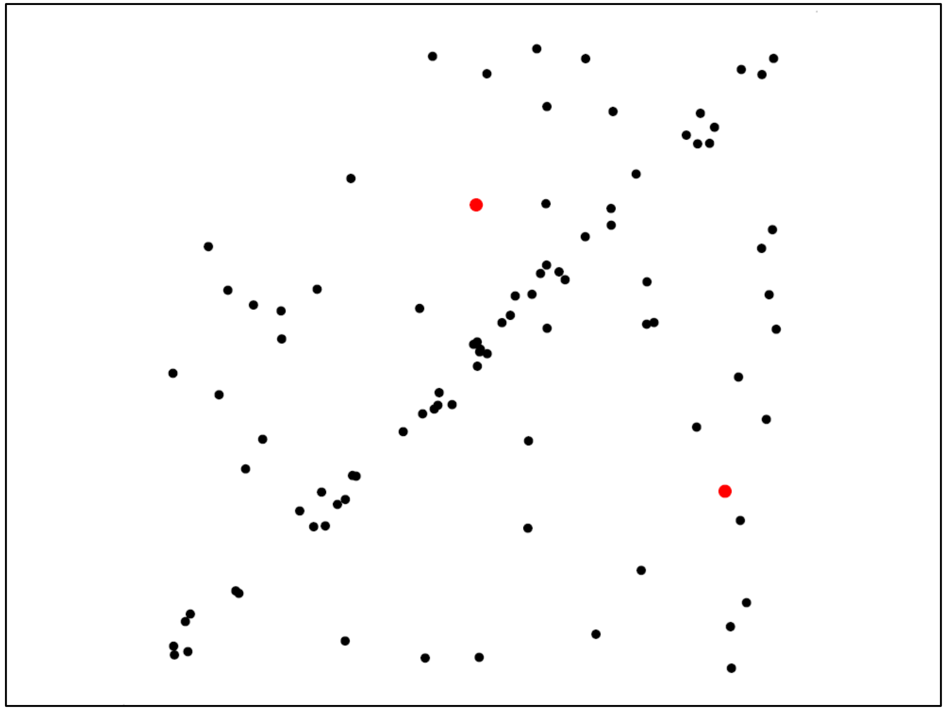
RANSAC for line fitting example



Least--squares fit

Source: R. Raguram

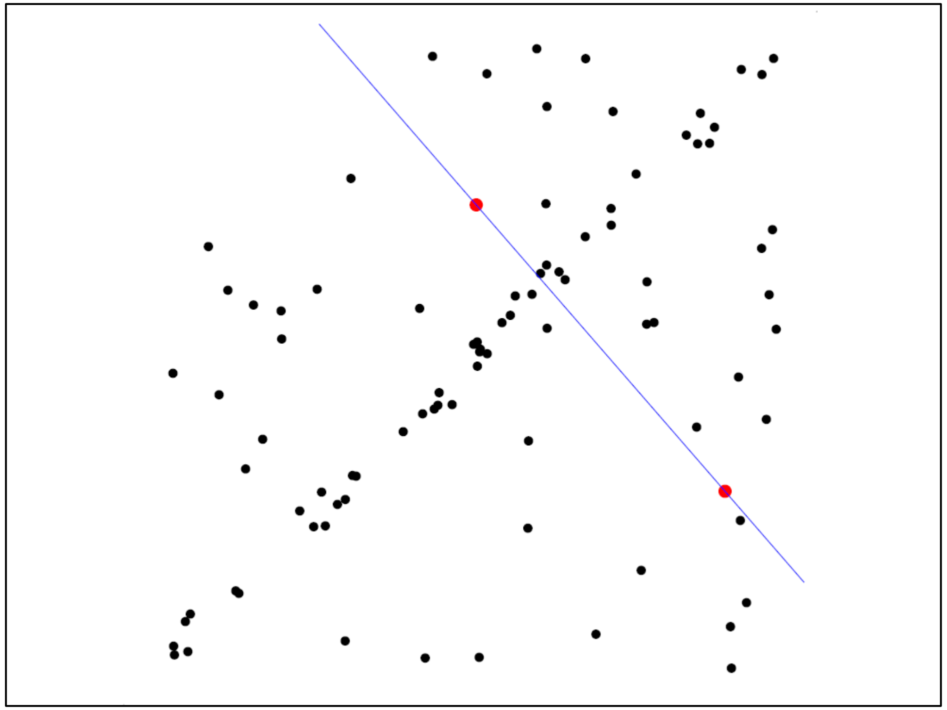
RANSAC for line fitting example



1. Randomly select minimal subset of points

Source: R. Raguram

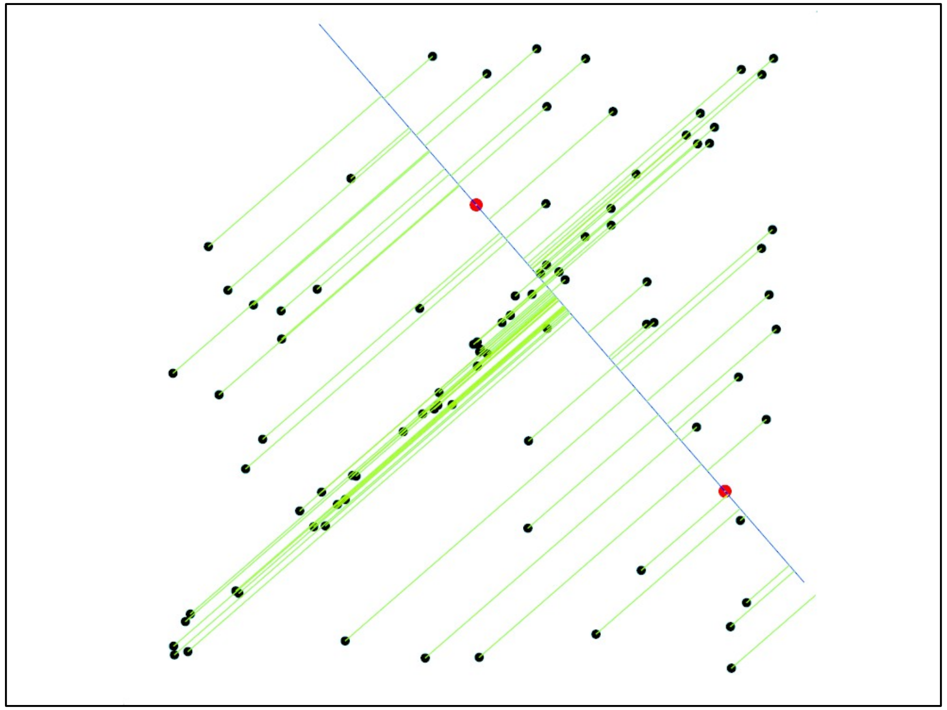
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model

Source: R. Raguram

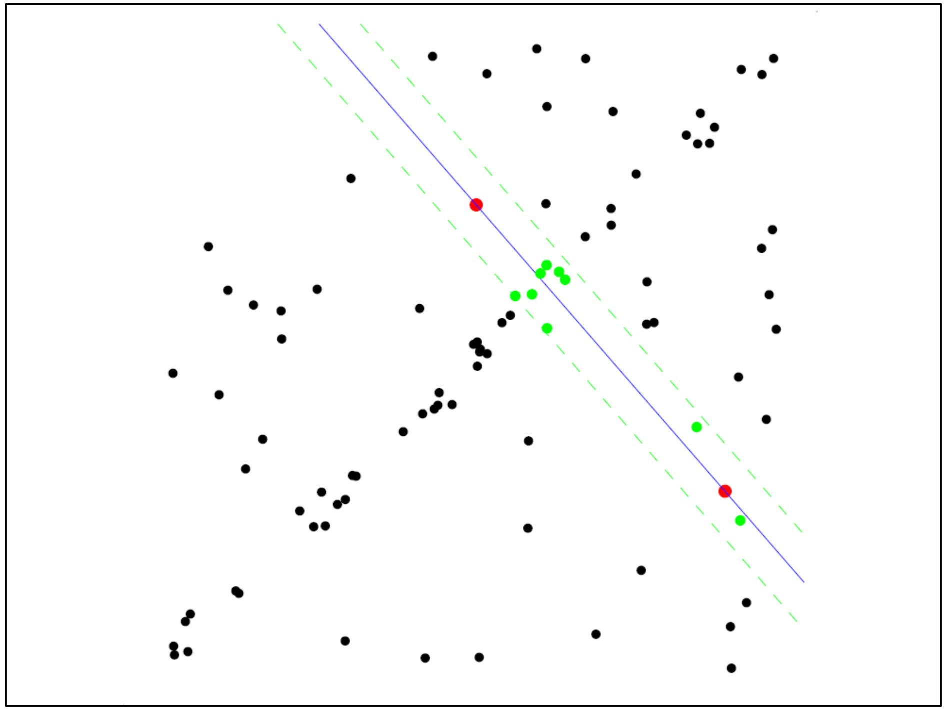
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function

Source: R. Raguram

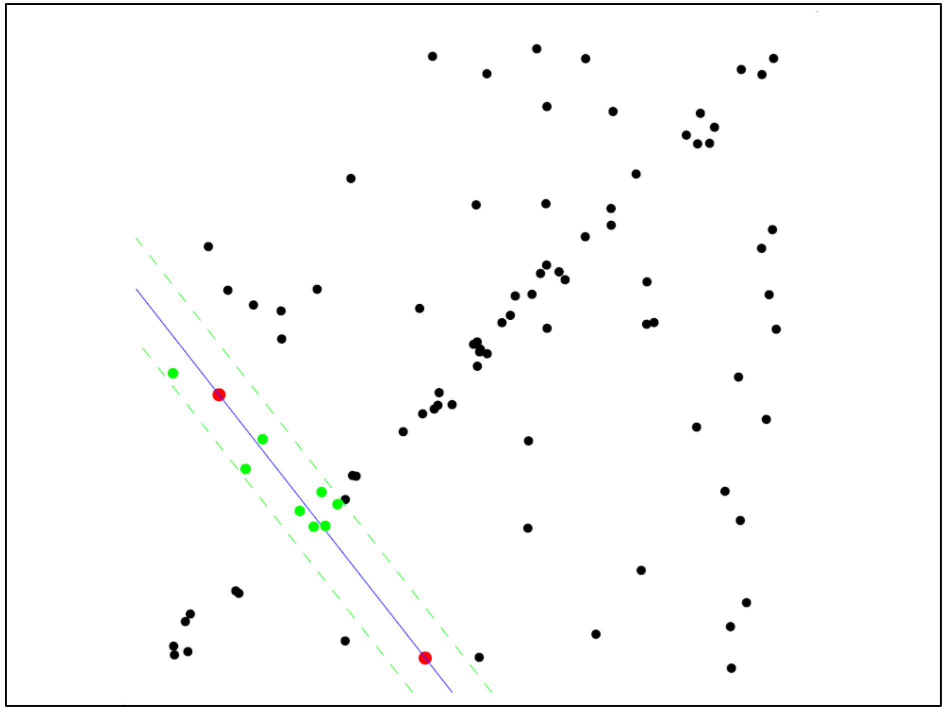
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error func<on
4. Select points consistent with model

Source: R. Raguram

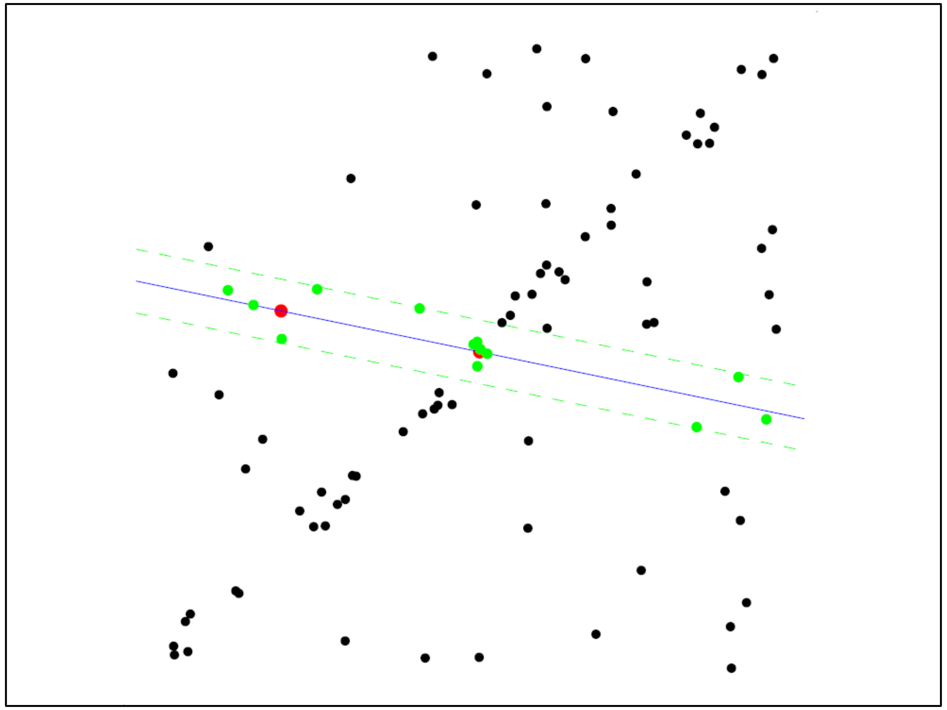
RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error func<on
4. Select points consistent with model
5. Repeat *hypothesize–and–verify* loop

Source: R. Raguram

RANSAC for line fitting example

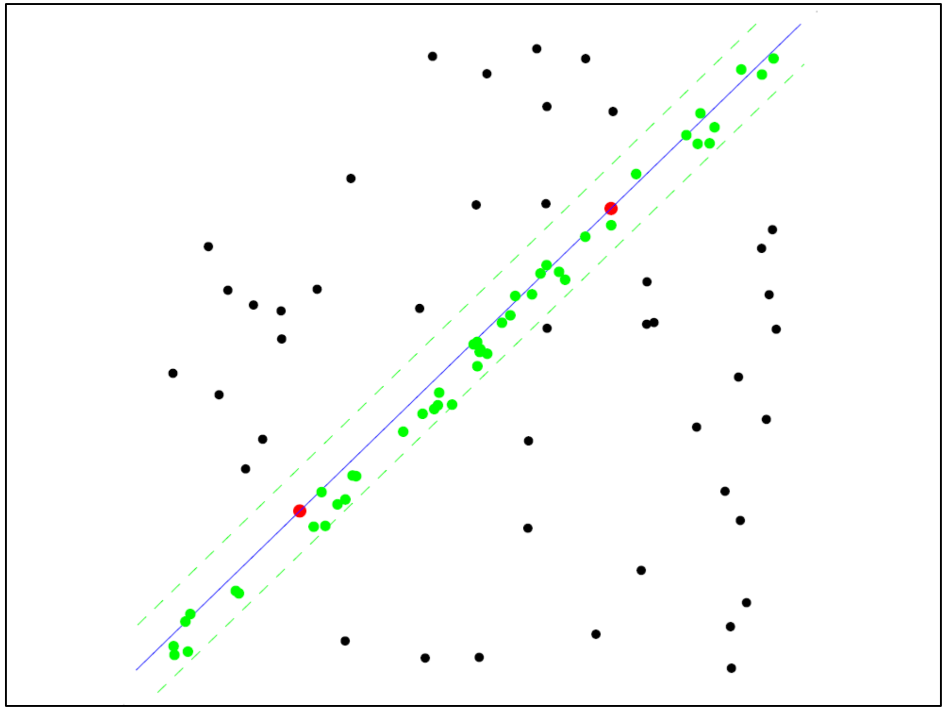


1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error func<on
4. Select points consistent with model
5. Repeat *hypothesize–and–verify* loop

Source: R. Raguram

RANSAC for line fitting example

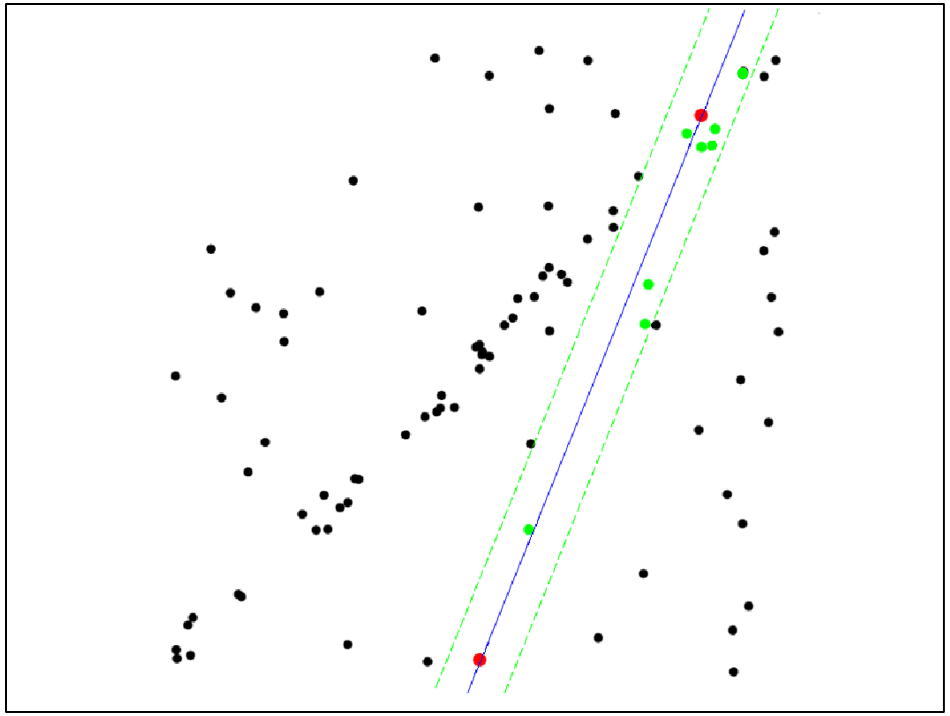
Uncontaminated sample



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error func<on
4. Select points consistent with model
5. Repeat *hypothesize–and–verify* loop

Source: R. Raguram

RANSAC for line fitting example



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error func<on
4. Select points consistent with model
5. Repeat *hypothesize–and–verify* loop

Source: R. Raguram

RANSAC for line fitting

Repeat ***N*** times:

- Draw ***s*** points uniformly at random
- Fit line to these ***s*** points
- Find inliers to this line among the remaining points (i.e., points whose distance from the line is less than ***t***)
- If there are ***d*** or more inliers, accept the line and refit using all inliers

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

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

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

	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

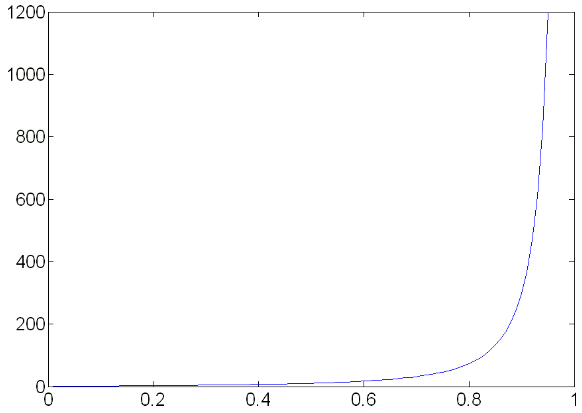
Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)

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

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



Source: M. Pollefeys

Choosing the parameters

- Initial number of points *s*
  - Typically minimum number needed to fit the model
- Distance threshold *t*
  - Choose *t* so probability for inlier is *p* (e.g. 0.95)
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$
- Number of samples *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g. *p*=0.99) (outlier ratio: *e*)
- Consensus set size *d*
  - Should match expected inlier ratio

Source: M. Pollefeys

Adaptively determining the number ofsamples

- Inlier ratio *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
- Adaptive procedure:
  - *N*=∞, *sample_count* =0
  - While *N* >*sample_count*
    - – Choose a sample and count the number of inliers
    - – Set e = 1 – (number of inliers)/(total number of points)
    - – Recompute *N* from *e:*

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

    - – Increment the *sample_count* by 1

Source: M. Pollefeys

RANSAC pros and cons

- Pros
  - Simple and general
  - Applicable to many different problems
  - Often works well in practice
- Cons
  - Lots of parameters to tune
  - Doesn’t work well for low inlier ratios (too many iterations, or can fail completely)
  - Can’t always get a good initialization of the model based on the minimum number of samples