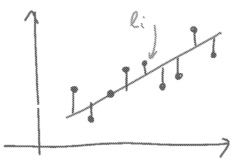
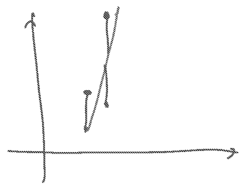
Poor model

- The solution obtained is optimal in the sense of minimizing the objective:

$$E(a,b) = \sum_i \big(\underbrace{y_i - (ax_i + b)}_{e_i}\big)^2$$

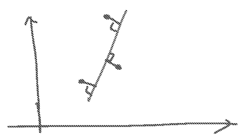
- Is this a good objective?





Minimizing geometric distance

- line equation in homogeneous coordinates:
- point p is on the line ℓ if:

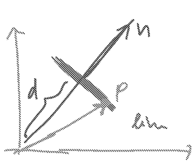


$$\underbrace{\ell^T}_{(a,b,c)^T} \underbrace{p}_{(x,y,1)} = 0$$

line in homogeneous coordinates

Line coefficients in homogeneous coordinates

$$\underbrace{\ell^T}_{(a,b,c)} \underbrace{p}_{(x,y,1)} = 0 \;\Rightarrow\; \underbrace{a x + b y}_{\text{normal coordinates}} + \underbrace{c}_{\text{inverse distance from origin}} = 0$$



for p to be on the line:

$$p \cdot n = d$$

$$n_x x + n_y y - d = 0$$

$$\Leftrightarrow \underbrace{a x + b y}_{\text{normal}} + \underbrace{c}_{\text{negative distance}} = 0$$

* If $n = (a,b)$ is not normalized, c is only proportional to distance

Geometric line fitting

$$\begin{cases} E(\ell) = \sum_i (\ell^T p_i)^2 \\ \ell^* = \arg\min_{\ell} E(\ell) \end{cases} \qquad p_i = (x_i, y_i, 1)$$

$$E(\ell) = \sum \ell^T p_i p_i^T \ell = \ell^T \underbrace{\sum p_i p_i^T}_{S} \ell \equiv \ell^T S \ell$$

$$\nabla E(\ell) = 0 \Rightarrow S\ell = 0$$

$\Rightarrow \ell =$ eigenvector of S belonging to zero eigenvalue

Correlation matrix

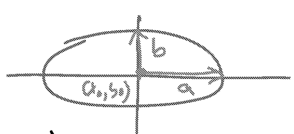
$$S \equiv \underbrace{\sum \underset{3\times 1}{p_i} \underset{1\times 3}{p_i^T}}_{3\times 3} = \text{correlation matrix of points} \qquad p_i = (x_i, y_i, 1)$$

$$= \begin{bmatrix} \sum x_i^2 & \sum x_i y_i & \sum x_i \\ \sum x_i y_i & \sum y_i^2 & \sum y_i \\ \sum x_i & \sum y_i & n \end{bmatrix} = D^T D \qquad D = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ \vdots & & \\ x_m & y_m & 1 \end{bmatrix}$$

Ellipse fitting

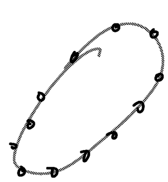
* Explicit axis-aligned equation:

$$\left(\frac{x - x_0}{a}\right)^2 + \left(\frac{y - y_0}{b}\right)^2 = 1$$



* Implicit equation (conic curve):

$$\begin{cases} ax^2 + bxy + cy^2 + dx + ey + f = 0 \\ b^2 - 4ac < 0 \end{cases}$$



Ellipse fitting

* Implicit ellipse equation:

$$\ell^T p = 0$$

$\ell = (a, b, c, d, e, f)$ model parameters

$p = (x^2, xy, y^2, x, y, 1)$

* convert input points

$$\{(x_i, y_i)\} \rightarrow \{p_i\} \qquad p_i = (x_i^2, x_i y_i, y_i^2, x_i, y_i, 1)$$

Ellipse fitting

* Find parameters ℓ by minimizing algebraic distance:

$$\begin{cases} E(\ell) = \sum_{i=1}^{m} (\ell^T p_i)^2 = \ell^T S \ell \\ S = \sum p_i p_i^T \\ \ell^* = \underset{\ell}{\operatorname{argmin}}\, E(\ell) \quad \text{s.t.} \quad b^2 - 4ac < 0 \end{cases}$$

Ellipse fitting

* Rewriting the constraint:

$$b^2 - 4ac < 0 \Leftrightarrow \ell^T C \ell = -1$$

$$C = \begin{bmatrix} 0 & 0 & -2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Ellipse fitting

* Modified objective:

$$\begin{cases} E(\ell) = \ell^T S \ell + \lambda(\ell^T C \ell + 1) \\ \ell^* = \underset{\ell}{\arg\min}\ E(\ell) \end{cases}$$

$\nabla E(\ell) = 0$

$\Rightarrow \cancel{2} S \ell = \cancel{2} \lambda C \ell$

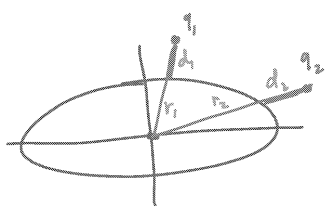
$\Rightarrow \ell = \lambda_1 S^{-1} C \ell \implies$ ℓ^* is the eigenvector of $S^{-1}C$ belonging to its negative eigenvalue

Algebraic distance

* When x_i is on the ellipse : $\ell^T x_i = 0$

- when x_i is off the ellipse : $q_i = \ell^T x_i$ provides a measure for distance from the ellipse.

algebraic distance $\equiv q_i \equiv \ell^T x_i \approx \dfrac{d_i}{d_i + r_i}$

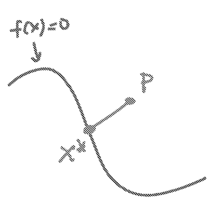


$d_1 \approx d_2$ but $q_1 > q_2$ (because $r_2 > r_1$)

Geometric distance

* Distance from a point to arbitrary implicit curve: $f(x) = 0$

$$\begin{cases} d(p, f) = |p - x^*| \\ x^* \text{ is closest point} \end{cases}$$



* solution:

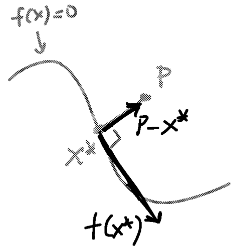
$d(p, f) = |p - x^*| = \dfrac{|f(p)|}{|\nabla f(x^*)|} \leftarrow$ algebraic distance

Geometric distance

* To find x^* solve:

$$\begin{cases} f(x^*) = 0 \\ (P - x^*) \cdot t(x^*) = 0 \end{cases}$$

tangent at x^*



* To find tangent at x^*
  1) compute gradient: $\nabla f(x^*) = \left[\frac{\partial f}{\partial x} \;\; \frac{\partial f}{\partial y}\right]^T$
  2) Rotate by -90°: $t = R(90)\nabla f(x^*) = \left[\frac{\partial f}{\partial y} \;\; -\frac{\partial f}{\partial x}\right]^T$

Geometric distance

* Approximated solution:

$$d(P, f) = |P - x^*| = \frac{|f(P)|}{|\nabla f(x^*)|} \approx \frac{|f(P)|}{|\nabla f(P)|}$$

geometric distance

algebraic distance

$\Rightarrow$ reduce the algebraic distance for points with large gradient $|\nabla f(P)|$

Geometric distance

* Geometric distance fitting:

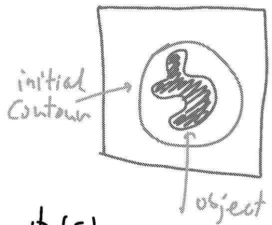
$$\begin{cases} E(l) = \sum_i \frac{|f(P_i; l)|}{|\nabla f(P_i; l)|} \\ \text{where} \quad f(P_i; l) = l^T P_i \\ l^* = \underset{l}{\operatorname{argmin}}\, E(l) \quad \text{s.t.} \quad b^2 - 4ac < 0 \end{cases}$$

* No explicit solution to $\nabla E(l) = 0$

  $\Rightarrow$ iterative numerical solution (e.g. Gradient descent)

Active contours (snakes)

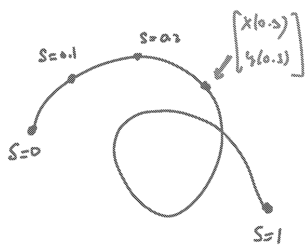
- Gradually deform initial contour to fit object boundaries



- Use parametric curve $\phi(s)$ to represent contour

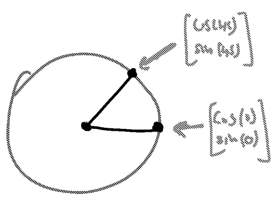
Parametric curves

$$\phi(s) = \begin{bmatrix} x(s) \\ y(s) \end{bmatrix}$$



* example:

$$\phi(s) = \begin{bmatrix} \cos(s) \\ \sin(s) \end{bmatrix}$$



Error functional

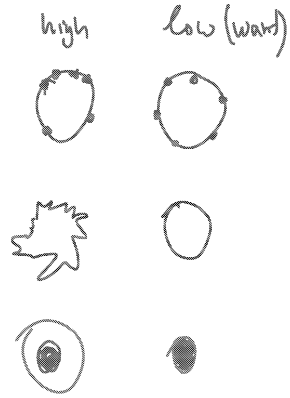
The unknown we are seeking is a function $\phi(s)$

$\Rightarrow$ error functional

$$E[\phi(s)] = \int_{\phi(s)} \Big(\underbrace{\alpha(s) E_{cont} + \beta(s) E_{curv}}_{\text{internal energy}} + \underbrace{\gamma(s) E_{img}}_{\text{external energy}} \Big) ds$$

$\alpha(s), \beta(s), \gamma(s)$ are coefficients of the different energy terms

Error functional



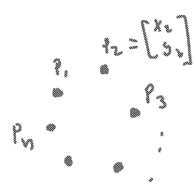
Continuity energy: $E_{cont} = \left|\frac{d\phi}{ds}\right|^2$

curvature energy: $E_{curv} = \left|\frac{d^2\phi}{ds^2}\right|^2$

image energy: $E_{img} = -|\nabla I|^2$

Discrete functional

Discrete case: $\phi(s) \rightarrow \{p_i\}_{i=1}^{n}$



$$E_{cont} = \left|\frac{d\phi}{ds}\right|^2 = |p_{i+1} - p_i|^2$$

$$E_{curv} = \left|\frac{d^2\phi}{ds^2}\right|^2 = |(p_{i+1} - p_i) - (p_i - p_{i-1})|^2$$
$$= |p_{i+1} - 2p_i + p_{i-1}|^2$$

$$E_{img} = -|\nabla I|^2$$

Discrete functional

$$E(\{p_i\}) = \sum_{i=1}^{n} \alpha_i (|p_{i+1} - p_i| - d)^2$$
$$+ \sum_{i=1}^{n} \beta_i |p_{i+1} - 2p_i + p_{i-1}|^2$$
$$- \sum_{i=1}^{n} \gamma_i |\nabla I(p_i)|^2$$

average distance between points to prevent shrinking

$\alpha_i, \beta_i, \gamma_i$ are user selected parameters

$\{p_i\}$ are the unknown model parameters

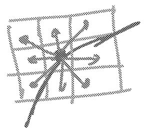
d is a computed parameter

(also zero β_i at corners to allow discontinuity)

Discrete functional

To minimize the error functional E:

- start with initial guess $\{p_i\}_{i=1}^{n}$
- compute d (average distance between points)
- For each point p_i try to minimize E by testing what happens when moving to neighboring locations
- Repeat while E is decreasing



Selecting coefficients

- Select $\alpha_i, \beta_i, \gamma_i$ to normalize the different terms to a similar scale and set the energy terms importance
- To allow for piecewise curves, set $\beta_i = 0$ to points with high curvature: i.e. when: $|p_{i+1} - 2p_i + p_{i-1}| > \tau$