Missing variable problems

- In many vision problems, if some variables were known the maximum likelihood inference problem would be easy
  - fitting; if we knew which line each token came from, it would be easy to determine line parameters
  - segmentation; if we knew the segment each pixel came from, it would be easy to determine the segment parameters
  - many, many, others!

Missing variable problems

- Strategy
  - estimate appropriate values for the missing variables
  - plug these in and now estimate parameters
  - re-estimate appropriate values for missing variables, continue
- Example with lines
  - guess which line gets which point
  - now fit the lines
  - now reallocate points to lines, using our knowledge of the lines
  - now refit, etc.
- We’ve seen this line of thought before (k means)

Missing variables - strategy

- In the Expectation-Maximization algorithm, we use the expected values of the missing values as the estimate.
- Thus iterate until convergence
  - replace missing variable with **expected** values, given **fixed** values of parameters
  - fix missing variables, choose parameters to maximize likelihood given fixed values of missing variables
- Line example
  - iterate till convergence
  - allocate each point to a line with a **weight**, which is the probability of the point given the line
  - refit lines to the weighted set of points
- Converges to local extremum
- Unlike K-means, we do not make a hard assignment; rather we the expected value (weight) which is a **soft** assignment.

Iterative Approach

- EM is basically gradient descent on the log likelihood.
- Gradient descent is working towards a local optimum by moving in the direction of maximum change (should know this).
- In the case of our least squares fit to a line, we end up with weights for each points which are indicative of how likely that point is on the line.
- We can easily fit this if the weights are constant.
- However, the weights are function of the line parameters---hence the iterative approach.

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)
  - Note that P(comes from noise)=(1-P(comes from line))

Lines and robustness

P(point | line and noise parameters)
= P(point | line)P(comes from line)
+ P(point | noise)P(comes from noise)

= P(point|line)□ + P(point|noise)(1-□)

Estimating the mixture model

- Introduce a set of hidden variables, δ, one for each point. They are one when the point is on the line, and zero when off.
- If these are known, the negative log-likelihood becomes:

- Here K is a normalizing constant, kn is the noise intensity (comments on choosing this later), and the parameters of the line are θ and c.

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

Details optional

Substituting for delta

- Now substitute □ by its expected value (for given □=(□, c, □) for each point)

$$
\begin{aligned}
E(\square_i) \;&=\; 1 * P(\square_i = 1 \mid \square, \mathbf{x}_i) + 0 * P(\square_i = 0 \mid \square, \mathbf{x}_i) \\
&= P(\square_i = 1 \mid \square, \mathbf{x}_i) \\
&\quad P(\mathbf{x}_i \mid \square_i = 1, \square,)
\end{aligned}
$$

Details optional

Substituting for delta

$$P\left(\square_i = 1 \mid \square, \mathbf{x}_i\right) = \frac{P\left(\mathbf{x}_i \mid \square_i = 1, \square\right)P\left(\square_i = 1\right)}{P\left(\mathbf{x}_i \mid \square_i = 1, \square\right)P\left(\square_i = 1\right) + P\left(\mathbf{x}_i \mid \square_i = 0, \square\right)P\left(\square_i = 0\right)}$$

$$= \frac{\exp\left(\square 1/_{2\square^2}\left[x_i \cos\square + y_i \sin\square + c\right]^2\right)\square}{\exp\left(\square 1/_{2\square^2}\left[x_i \cos\square + y_i \sin\square + c\right]^2\right)\square + \exp\left(\square k_n\right)\left(1 \square \square\right)}$$