CPSC 540: Machine Learning
Expectation Maximization

Mark Schmidt
University of British Columbia

Winter 2020
Last Time: Learning with MAR Values

We discussed learning with “missing at random” values in data:

\[
X = \begin{bmatrix}
1.33 & 0.45 & -0.05 & -1.08 & ?
\1.49 & 2.36 & -1.29 & -0.80 & ?
-0.35 & -1.38 & -2.89 & -0.10 & ?
0.10 & -1.29 & 0.64 & -0.46 & ?
0.79 & 0.25 & -0.47 & -0.18 & ?
2.93 & -1.56 & -1.11 & -0.81 & ?
-1.15 & 0.22 & -0.11 & -0.25 & ?
\end{bmatrix}
\]

Imputation approach:
- Guess the most likely value of each ?, fit model with these values (and repeat).

K-means clustering algorithm is a special case:
- Gaussian mixture \((\pi_c = 1/k, \Sigma_c = I)\) and ? being the cluster \((? \in \{1, 2, \cdots, k\})\).
Parameters, Hyper-Parameters, and Nuisance Parameters

- Are the \( ? \) values “parameters” or “hyper-parameters”?

- **Parameters:**
  - Variables in our model that we optimize based on the training set.

- **Hyper-Parameters**
  - Variables that control model complexity, typically set using validation set.
  - Often become degenerate if we set these based on training data.
  - We sometimes add optimization parameters in here like step-size.

- **Nuisance Parameters**
  - Not part of the model and not really controlling complexity.
  - An alternative to optimizing (“imputation”) is to consider all values.
    - Based on marginalization rule for probabilities.
    - Consider all possible imputations, and weight them by their probability.
Expectation Maximization Notation

- **Expectation maximization (EM)** is an optimization algorithm for MAR values:
  - Applies to problems that are easy to solve with “complete” data (i.e., you knew \(?\)).
  - Allows probabilistic or “soft” assignments to MAR (or other nuisance) variables.
    - Imputation approach is sometimes called “hard” EM.
  - EM is among the most cited paper in statistics.

- **EM notation**: we use \( O \) as observed data and \( H \) as hidden (?) data.
  - Semi-supervised learning: observe \( O = \{X, y, \bar{X}\} \) but don’t observe \( H = \{\bar{y}\} \).
  - Mixture models: observe data \( O = \{X\} \) but don’t observe clusters \( H = \{z^i\}_{i=1}^n \).

- We use \( \Theta \) as parameters we want to optimize.
  - In Gaussian mixtures this will be the \( \pi_c, \mu_c, \) and \( \Sigma_c \) variables.
The Two Likelihoods: “Complete” and “Marginal”

- **“Complete” likelihood**: likelihood with known hidden values, \( p(O, H \mid \Theta) \).
  - We assume that this is “nice”. Maybe it has a closed-form MLE or is convex.

- **“Marginal” likelihood**: likelihood with unknown hidden values, \( p(O \mid \Theta) \).
  - This is our usual likelihood, the thing we actually want to optimize.

The “complete” and “marginal” likelihoods are related by the marginalization rule:

\[
\underbrace{p(O \mid \Theta)}_{\text{“marginal”}} = \sum_{H_1} \sum_{H_2} \cdots \sum_{H_m} p(O, H \mid \Theta) = \sum_{H} p(O, H \mid \Theta) \quad .
\]

where we sum over all possible \( H \equiv \{H_1, H_2, \ldots, H_m\} \).

- For mixture models, this sums over all possible clusterings \((k^n \text{ values})\).
- Replace the sums by integrals for continuous hidden values.
Expectation Maximization Bound

- The negative log-likelihood (that we want to optimize) thus has the form

\[ -\log p(O \mid \Theta) = -\log \left( \sum_H p(O, H \mid \Theta) \right), \]

which has a sum inside the log.
- This does not preserve convexity: minimizing it is usually NP-hard.

Both EM and imputation are based on the approximation:

\[ -\log \left( \sum_H p(O, H \mid \Theta) \right) \approx -\sum_H \alpha_H \log p(O, H \mid \Theta) \]

where \( \alpha_H \) is some probability for the assignment \( H \) to the hidden variables.
- An expectation over “complete” log-likelihood.
- This is useful when the approximation is easier to minimize.
Expectation Maximization Bound

- Each iteration of EM and imputation optimize the approximation:
  \[ \Theta^{t+1} \in \arg\min_{\Theta} - \sum_{H} \alpha_{t}^{H} \log p(O, H | \Theta). \]

  where the probabilities \( \alpha_{t}^{H} \) are updated after each iteration \( t \).

- Imputation sets \( \alpha_{t}^{H} = 1 \) for the most likely \( H \) given \( \Theta^{t} \) (all other \( \alpha_{t}^{H} = 0 \)).
  - It assumes that the imputations are correct, then optimizes with the guess.

- In EM we set \( \alpha_{t}^{H} = p(H | O, \Theta^{t}) \), weighting \( H \) by probability given \( \Theta^{t} \).
  - It weighs different imputations by their probability, then optimizes.
Expectation Maximization as Bound Optimization

- We’ll show that the EM approximation minimizes an upper bound,

\[- \log p(O \mid \Theta) \leq - \sum_{H} p(H \mid O, \Theta^t) \log p(O, H \mid \Theta) + \text{const.,} \]

what we want

what we optimize

- Geometry of expectation maximization as “bound optimization”:
  - At each iteration \( t \) we optimize a bound on the function.
Expectation Maximization (EM)

- So **EM** starts with $\Theta^0$ and sets $\Theta^{t+1}$ to maximize $Q(\Theta \mid \Theta^t)$.

- This is typically written as two steps:
  1. **E-step**: Define expectation of complete log-likelihood given last parameters $\Theta^t$,
     
     $$Q(\Theta \mid \Theta^t) = \sum_H p(H \mid O, \Theta^t) \log p(O, H \mid \Theta)$$
     
     which is a weighted version of the “nice” $\log p(O, H)$ values.

  2. **M-step**: Maximize this expectation to generate new parameters $\Theta^{t+1}$,
     
     $$\Theta^{t+1} = \arg\max_{\Theta} Q(\Theta \mid \Theta^t).$$
Expectation Maximization for Mixture Models

- In the case of a mixture model with extra “cluster” variables \( z^i \) EM uses

\[
Q(\Theta \mid \Theta^t) = \mathbb{E}_{z \mid X, \Theta}[\log p(X, z \mid \Theta)]
\]

\[
= \sum_{z^1=1}^k \sum_{z^2=1}^k \cdots \sum_{z^n=1}^k p(z \mid X, \Theta^t) \log p(X, z \mid \Theta)
\]

\[
= \sum_{z^1=1}^k \sum_{z^2=1}^k \cdots \sum_{z^n=1}^k \left( \prod_{i=1}^n p(z^i \mid x^i, \Theta^t) \right) \left( \sum_{i=1}^n \log p(x^i, z^i \mid \Theta) \right)
\]

\[
= (\text{see EM notes, tedious use of distributive law and independences})
\]

\[
= \sum_{i=1}^n \sum_{z^i=1}^k p(z^i \mid x^i, \Theta^t) \log p(x^i, z^i \mid \Theta).
\]

- **Sum over** \( k^n \) clusterings turns into **sum over** \( nk \) 1-example assignments.

- Same simplification happens for semi-supervised learning, we’ll discuss why later.
Expectation Maximization for Mixture Models

- In the case of a mixture model with extra “cluster” variables $z^i$ EM uses

$$Q(\Theta \mid \Theta^t) = \sum_{i=1}^n \sum_{z^i=1}^k r^i_c p(z^i \mid x^i, \Theta^t) \log p(x^i, z^i \mid \Theta).$$

- This is just a weighted version of the usual likelihood.
  - We just need to do MLE in weighted Gaussian, weighted Bernoulli, etc.

- We typically write update in terms of responsibilitites (easy to calculate),

$$r^i_c \triangleq p(z^i = c \mid x^i, \Theta^t) = \frac{p(x^i \mid z^i = c, \Theta^t)p(z^i = c \mid \Theta^t)}{\sum_{c' = 1}^k p(x^i \mid z^i = c', \Theta^t)p(z^i = c' \mid \Theta^t)} \quad \text{(Bayes rule)},$$

the probability that cluster $c$ generated $x^i$.
  - In $k$-means $r^i_c = 1$ for most likely cluster and 0 otherwise.
  - You may get underflow when computing $r^i_c$ (see bonus for log-domain tricks).
Expectation Maximization for Mixture of Gaussians

- For mixture of Gaussians, E-step computes all $r^i_c$ and M-step minimizes the weighted NLL:

  $$
  \pi^{t+1}_c = \frac{1}{n} \sum_{i=1}^{n} r^i_c \quad \text{(proportion of examples soft-assigned to cluster } c) \n  $$

  $$
  \mu^{t+1}_c = \frac{1}{\sum_{i=1}^{n} r^i_c} \sum_{i=1}^{n} r^i_c x^i \quad \text{(mean of examples soft-assigned to cluster } c) \n  $$

  $$
  \Sigma^{t+1}_c = \frac{1}{\sum_{i=1}^{n} r^i_c} \sum_{i=1}^{n} r^i_c (x^i - \mu^{t+1}_c)(x^i - \mu^{t+1}_c)^\top \quad \text{(covariance of examples soft-assigned to } c). \n  $$

- Now you would compute new responsibilities and repeat.
  - Notice that there is no step-size.

- EM for fitting mixture of Gaussians in action:
  https://www.youtube.com/watch?v=B36fzChfyGU
Expectation Maximization

Discussing of EM for Mixtures of Gaussians

- EM and mixture models are used in a ton of applications.
  - One of the default unsupervised learning methods.
- EM usually doesn’t reach global optimum.
  - Classic solution: restart the algorithm from different initializations.
  - Lots of work in CS theory on getting better initializations.
- MLE for some clusters may not exist (e.g., only responsible for one point).
  - Use MAP estimates or remove these clusters.
- EM does not fix “propagation of errors” from imputation approach.
  - But it reduces problem by incorporating a “confidence” over different imputations.
- Can you make it robust?
  - Use mixture of Laplace of student t distributions.
  - Don’t have closed-form EM steps: compute responsibilities then need to optimize.
Outline

1. Expectation Maximization
2. Monotonicity of EM
Monotonicity of EM

- Classic result is that EM iterations are monotonic:
  \[ \log p(O \mid \Theta^{t+1}) \geq \log p(O \mid \Theta^t) , \]
  - We don’t need a step-size and this is useful for debugging.

- We can show this by proving that the below picture is “correct”:
  
  The \( Q \) function leads to a global bound on the original function.
  - At \( \Theta^t \) the bound matches original function.
    - So if you improve on the \( Q \) function, you improve on the original function.
Let’s show that the $Q$ function gives a global upper bound on NLL:

$$-\log p(O \mid \Theta) = -\log \left( \sum_H p(O, H \mid \Theta) \right)$$ (marginalization rule)

$$= -\log \left( \sum_H \alpha_H \frac{p(O, H \mid \Theta)}{\alpha_H} \right)$$ (for $\alpha_H \neq 0$)

$$\leq - \sum_H \alpha_H \log \left( \frac{p(O, H \mid \Theta)}{\alpha_H} \right),$$

because $-\log(z)$ is convex and the $\alpha_H$ are a convex combination.
Monotonicity of EM

Using that log turns multiplication into addition we get

\[-\log p(O \mid \Theta) \leq - \sum_H \alpha_H \log \left( \frac{p(O, H \mid \Theta)}{\alpha_H} \right)\]

\[= - \sum_H \alpha_H \log p(O, H \mid \Theta) + \sum_H \alpha_H \log \alpha_H\]

\[= -Q(\Theta \mid \Theta^t) - \text{entropy}(\alpha),\]

so we have the first part of the picture, \(-\log p(O \mid \Theta^{t+1}) \leq -Q(\Theta \mid \Theta^t) + \text{const.}\)

- Entropy is a measure of how “random” the \(\alpha_H\) values are.
- \(Q\) behaves more like true objective for \(H\) that are more “predictable”.

Now we need to show that this holds with equality at \(\Theta^t\).
Bound on Progress of Expectation Maximization

- To show equality at $\Theta^t$ we use definition of conditional probability,

$$p(H \mid O, \Theta^t) = \frac{p(O, H \mid \Theta^t)}{p(O \mid \Theta^t)} \quad \text{or} \quad \log p(O \mid \Theta^t) = \log p(O, H \mid \Theta^t) - \log p(H \mid O, \Theta^t)$$

- Multiply by $\alpha_H$ and summing over $H$ values,

$$\sum_H \alpha_H \log p(O \mid \Theta^t) = \sum_H \alpha_H \log p(O, H \mid \Theta^t) - \sum_H \alpha_H \log \underbrace{p(H \mid O, \Theta^t)}_{Q(\Theta^t \mid \Theta^t)} = Q(\Theta^t \mid \Theta^t) + \text{entropy}(\alpha),$$

- Which gives the result we want:

$$\log p(O \mid \Theta^t) \sum_H \alpha_H = Q(\Theta^t \mid \Theta^t) + \text{entropy}(\alpha),$$
Bound on Progress of Expectation Maximization

Thus we have the two bounds

\[
\log p(O \mid \Theta) \geq Q(\Theta \mid \Theta^t) + \text{entropy}(\alpha)
\]
\[
\log p(O \mid \Theta^t) = Q(\Theta^t \mid \Theta^t) + \text{entropy}(\alpha).
\]

Subtracting these and using \(\Theta = \Theta^{t+1}\) gives a stronger result,

\[
\log p(O \mid \Theta^{t+1}) - \log p(O \mid \Theta^t) \geq Q(\Theta^{t+1} \mid \Theta^t) - Q(\Theta^t \mid \Theta^t),
\]

that we improve objective by at least the decrease in \(Q\).

Inequality holds for any choice of \(\Theta^{t+1}\).

- Approximate M-steps are ok: we just need to decrease \(Q\) to improve likelihood.

For imputation, we instead improve “complete” log-likelihood, \(\log p(O, H \mid \Theta^t)\).

- Which isn’t quite what we want, treats hidden data as a “parameter”.
Summary

- **Expectation maximization:**
  - Optimization with MAR variables, when knowing MAR variables make problem easy.
  - Instead of imputation, works with “soft” assignments to nuisance variables.
  - Maximizes log-likelihood, weighted by all imputations of hidden variables.

- **Monotonicity of EM:** EM is guaranteed not to decrease likelihood.

- Next time: generalizing histograms?
EM Alternatives

- Are there alternatives to EM?
  - Could use gradient descent, SGD, and so on.
  - Many variations on EM to speed up its convergence (for example, “adaptive” bound optimization).
  - **Spectral** and other recent methods have some global guarantees.
Avoiding Underflow when Computing Responsibilities

- Computing responsibility may underflow for high-dimensional $x^i$, due to
  \[ p(x^i \mid z^i = c, \Theta^t). \]

- Usual ML solution: do all but last step in log-domain.
  \[
  \log r_c^i = \log p(x^i \mid z^i = c, \Theta^t) + \log p(z^i = c \mid \Theta^t)
  - \log \left( \sum_{c' = 1}^{k} p(x^i \mid z^i = c', \Theta^t) p(z^i = c' \mid \Theta^t) \right).
  \]

- To compute last term, use "log-sum-exp" trick.
Log-Sum-Exp Trick

- To compute \( \log(\sum_i \exp(v_i)) \), set \( \beta = \max_i \{v_i\} \) and use:

\[
\log(\sum_c \exp(v_i)) = \log(\sum_i \exp(v_i - \beta + \beta)) \\
= \log(\sum_i \exp(v_i - \beta) \exp(\beta)) \\
= \log(\exp(\beta)) \sum_i \exp(v_i - \beta)) \\
= \log(\exp(\beta)) + \log(\sum_i \exp(v_i - \beta)) \\
= \beta + \log(\sum_i \underbrace{\exp(v_i - \beta)}_{\leq 1}).
\]

- Avoids overflows due to computing \( \exp \) operator.
Alternate View of EM as BCD

- We showed that given $\alpha$ the M-step minimizes in $\Theta$ the function

$$F(\Theta, \alpha) = -\mathbb{E}_\alpha[\log p(O, H | \Theta)] - \text{entropy}(\alpha).$$

- The E-step minimizes this function in terms of $\alpha$ given $\Theta$.
  - Setting $\alpha_H = p(H | O, \Theta)$ minimizes it.

- Note that $F$ is not the NLL, but $F$ and the NLL have same stationary points.

- From this perspective, we can view EM as a block coordinate descent method.

- This perspective is also useful if you want to do approximate E-steps.
Alternate View of EM as KL-Proximal

Using definitions of expectation and entropy and $\alpha$ in the last slide gives

$$F(\Theta, \alpha) = - \sum_H p(H | O, \theta^t) \log p(O, H | \Theta) + \sum_H p(H | O, \theta^t) \log p(H | O, \theta^t)$$

$$= - \sum_H p(H | O, \theta^t) \log \frac{p(O, H | \Theta)}{p(H | O, \theta^t)}$$

$$= - \sum_H p(H | O, \theta^t) \log \frac{p(H | O, \Theta)p(O | \Theta)}{p(H | O, \theta^t)}$$

$$= - \sum_H \log p(O | \Theta) - \sum_H p(H | O, \theta^t) \log \frac{p(H | O, \Theta)}{p(H | O, \theta^t)}$$

$$= NLL(\Theta) + KL(p(H | O, \theta^t) \| p(H | O, \Theta)).$$

From this perspective, we can view EM as a “proximal point” method.

Classical proximal point method uses $\frac{1}{2} \| \theta^t - \theta \|^2$, EM uses KL divergence.

From this view we can see that EM doesn’t depend on parameterization of $\Theta$.

If we linearize NLL and we multiply $KL$ term by $1/\alpha_k$ (step-size), we get the natural gradient method.