Learning with weak supervision

CS 6355: Structured Prediction

THE UNIVERSITY OF UTAH

What we have seen so far…

- What is structured output?
- Joint scoring functions over multiple interacting decisions
- Various families of inference algorithms
  - Search over a combinatorial space
- Learning in the fully supervised setting

The difficulty with supervised learning

Annotated data is expensive and costs increase when…

- *…a task requires specialized expertise*

  E.g. "Only a trained linguist or a board certified radiologist can label my data"

- *…labeling examples involves making multiple decisions*

  E.g. "Annotate this sentence with a parse tree"

  (instead of a single binary decision)

The difficulty with supervised learning

Annotated data is expensive and costs increase when…

- *…a task requires specialized expertise*

  E.g. "Only a trained linguist or a board certified radiologist can label my data"

- *…labeling examples involves making multiple decisions*

  E.g. "Annotate this sentence with a parse tree"
  (instead of a single binary decision)

Creating labeled examples for structured output problems is expensive and time consuming

What if: The labels are missing

Training data: $D = \overset{\{\mathbf{x}_i\}}{\cancel{\{(\mathbf{x}_i, \mathbf{y}_i)\}}}$

Or perhaps we have

A small number of labeled examples

+

Extra information in the form of unlabeled examples, or constraints

This lecture

- What is weak supervision?

- The expectation maximization algorithm
  - A general template for learning with weak supervision

- Learning with latent variables

- Learning with constraints

This lecture

- What is weak supervision?
- The expectation maximization algorithm
  – A general template for learning with weak supervision
- Learning with latent variables
- Learning with constraints

Weak supervision: The motivation

What can we do with unlabeled or partially labeled examples?

Assume that we know what the task is, that is, the definition of the structure in question

Example:

- Suppose we know that my task involves predicting a sequence of labels, but…
- …we don’t have any labeled data

The many forms of supervision

- Labeled examples
  - We have examples with the output structure labeled
- Unlabeled/partially labeled examples
  - We have examples, and perhaps also parts of the output structures for some of them
- Hard constraints
  - Restrict the space of output structures that can exist
- Soft constraints
  - Similar to hard constraints, but allows violations
- Distant/indirect supervision
  - We know of another task that is correlated in a well defined way with the task we care about
- Heuristics
  - We can write simple programs that are reasonably good on specific examples

The many forms of supervision

Other kinds of supervision exist.

- Labeled examples
  - We have examples with the output structure labeled
- Unlabeled/partially labeled examples
  - We have examples, and perhaps also parts of the output structures for some of them
- Hard constraints
  - Restrict the space of output structures that can exist
- Soft constraints
  - Similar to hard constraints, but allows violations
- Distant/indirect supervision
  - We know of another task that is correlated in a well defined way with the task we care about
- Heuristics
  - We can write simple programs that are reasonably good on specific examples

The many forms of supervision

Expensive

Typically cheaper

- Labeled examples
  - We have examples with the output structure labeled
- Unlabeled/partially labeled examples
  - We have examples, and perhaps also parts of the output structures for some of them
- Hard constraints
  - Restrict the space of output structures that can exist
- Soft constraints
  - Similar to hard constraints, but allows violations
- Distant/indirect supervision
  - We know of another task that is correlated in a well defined way with the task we care about
- Heuristics
  - We can write simple programs that are reasonably good on specific examples

Other kinds of supervision exist.

The many forms of supervision

Expensive

Typically cheaper

- Labeled examples
  - We have examples with the output structure labeled
- Unlabeled/partially labeled examples
  - We have examples, and perhaps also parts of the output structures for some of them
- Hard constraints
  - Restrict the space of output structures that can exist
- Soft constraints
  - Similar to hard constraints, but allows violations
- Distant/indirect supervision
  - We know of another task that is correlated in a well defined way with the task we care about
- Heuristics
  - We can write simple programs that are reasonably good on specific examples

Other kinds of supervision exist.

Usually we have a mix of different kinds of supervision.

The many forms of supervision

Expensive

Typically cheaper

- Labeled examples
  - We have examples with the output structure labeled
- Unlabeled/partially labeled examples
  - We have examples, and perhaps also parts of the output structures for some of them
- Hard constraints
  - Restrict the space of output structures that can exist
- Soft constraints
  - Similar to hard constraints, but allows violations
- Distant/indirect supervision
  - We know of another task that is correlated in a well defined way with the task we care about
- Heuristics
  - We can write simple programs that are reasonably good on specific examples

Other kinds of supervision exist.

Usually we have a mix of different kinds of supervision.

How do we systematically take advantage of such signals?

This lecture

- What is weak supervision?

- The expectation maximization algorithm
  - A general template for learning with weak supervision

- Learning with latent variables

- Learning with constraints

Expectation Maximization

- A meta-algorithm to estimate a probability distribution when some part of the output is missing
  - The entire output could be missing (i.e. unlabeled examples)
  - A part of the output could be missing (i.e., partially labeled examples)

Expectation Maximization

- A meta-algorithm to estimate a probability distribution when some part of the output is missing
  - The entire output could be missing (i.e. unlabeled examples)
  - A part of the output could be missing (i.e., partially labeled examples)

- Needs assumptions about the underlying probability distribution
  - Performance sensitive to the validity of this assumption (and also the initial guess of the parameters)

- Converges to a local maximum of the likelihood function

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \mathrm{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

But our model doesn’t directly tell us about this probability.
It only knows $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$.

Let's revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

How do we state the probability term in terms of $P_\theta(\boldsymbol{x}_i, \boldsymbol{y}_i)$?

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

How do we state the probability term in terms of $P_\theta(\boldsymbol{x}_i, \boldsymbol{y}_i)$?

Answer: Marginalize the missing terms

Let's revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

How do we state the probability term in terms of $P_\theta(\boldsymbol{x}_i, \boldsymbol{y}_i)$?

Answer: Marginalize the missing terms

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

How do we state the probability term in terms of $P_\theta(\boldsymbol{x}_i, \boldsymbol{y}_i)$?

Answer: Marginalize the missing terms

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log P(\mathbf{x}_i \mid \theta)$$

How do we state the probability term in terms of $P_\theta(\boldsymbol{x}_i, \boldsymbol{y}_i)$?

Answer: Marginalize the missing terms

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

Sum over all structures for the example $\mathbf{x}_i$

Let’s revisit maximum likelihood estimation

Given unlabeled examples $D = \{\mathbf{x}_i\}$, we want to learn the parameters θ that defines a probability distribution $P_\theta(\mathbf{x}_i, \mathbf{y}_i)$

Find parameters that maximize the likelihood (or equivalently log-likelihood) of the data

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

Goal: Maximize this expression in terms of the parameters

This maximization is not easy. Sum inside log

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\text{D} \mid \theta)$

$$\text{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

Why do we want to maximize this? Because this gives us the maximum likelihood estimate

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \mathrm{LL}(\mathrm{D} \mid \theta)$

$$\mathrm{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

This is true for *any* probability distribution $Q_i(\mathbf{y})$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\text{D} \mid \theta)$

$$\text{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

This is true for *any* probability distribution $Q_i(\mathbf{y})$

The summation over $\mathbf{y}$ is the definition of expectation with respect to $Q_i(\mathbf{y})$

$$E_{z \sim Q}[f(z)] = \sum_z Q(z) f(z)$$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\text{D} \mid \theta)$

$$\text{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

$$E_{z \sim Q}[f(z)] = \sum_z Q(z) f(z)$$

Let us build an approximation

What we want: Maximize log likelihood$(D \mid \theta) = \mathrm{LL}(\mathrm{D} \mid \theta)$

$$\mathrm{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

$$E_{z \sim Q}[f(z)] = \sum_z Q(z) f(z)$$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \mathrm{LL}(\mathrm{D} \mid \theta)$

$$\mathrm{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

$$E_{z \sim Q}[f(z)] = \sum_z Q(z) f(z)$$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\text{D} \mid \theta)$

$$\text{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

$$= \sum_i \log E_{\mathbf{y} \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right]$$

How do we proceed now?

We haven’t made the problem any simpler by just rewriting it.

Jensen's inequality

If f is a convex function and X is a random variable, then

$$f(E[X]) \leq E[f(X)]$$

Or: If f is a concave function and X is a random variable, then

$$f(E[X]) \geq E[f(X)]$$

Let’s apply Jensen’s inequality

If f is a concave function and X is a random variable, then

$$f(E[X]) \geq E[f(X)]$$

Let's apply Jensen's inequality

If f is a concave function and X is a random variable, then

$$f(E[X]) \geq E[f(X)]$$

Let us apply this to the following function:

$$\log E_{y \sim Q_i}\left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right]$$

Let’s apply Jensen’s inequality

If f is a concave function and X is a random variable, then

$$f(E[X]) \geq E[f(X)]$$

Let us apply this to the following function:

$$\log E_{y \sim Q_i}\left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right]$$

$\log(x)$ is a concave function in x and *the expression inside the expectation* is a random variable

$$\log E_{y \sim Q_i}\left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right] \geq E_{y \sim Q_i}\left[\log \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right]$$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\text{D} \mid \theta)$

$$\begin{aligned}
\text{LL}(D \mid \theta) &= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y}) \\
&= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right) \\
&= \sum_i \log E_{\mathbf{y} \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right]
\end{aligned}$$

By Jensen's inequality: $\log E_{y \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)} \right] \geq E_{y \sim Q_i} \left[\log \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)} \right]$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\text{D} \mid \theta)$

$$\text{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

$$= \sum_i \log E_{\mathbf{y} \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right]$$

$$\geq \sum_i E_{\mathbf{y} \sim Q_i} \left[\log \frac{P_\theta(x_i, y)}{Q_i(\mathbf{y})} \right]$$

By Jensen's inequality: $\log E_{y \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)} \right] \geq E_{y \sim Q_i} \left[\log \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)} \right]$

Let us build an approximation

What we want: Maximize $\log \text{likelihood}(D \mid \theta) = \text{LL}(\mathrm{D} \mid \theta)$

$$
\begin{aligned}
\text{LL}(D \mid \theta) &= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y}) \\
&= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right) \\
&= \sum_i \log E_{\mathbf{y} \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right] \\
&\geq \sum_i E_{\mathbf{y} \sim Q_i} \left[\log \frac{P_\theta(x_i, y)}{Q_i(\mathbf{y})} \right] \\
&= \sum_i E_{\mathbf{y} \sim Q_i} [\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{\mathbf{y} \sim Q_i} [\log Q_i(\mathbf{y})]
\end{aligned}
$$

Rewrite log + linearity of expectation

Let us build an approximation

What we want: Maximize log likelihood$(D \mid \theta) = \mathrm{LL}(\mathrm{D} \mid \theta)$

$$\begin{aligned}
\mathrm{LL}(D \mid \theta) &= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y}) \\
&= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right) \\
&= \sum_i \log E_{\mathbf{y} \sim Q_i} \left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right] \\
&\geq \sum_i E_{\mathbf{y} \sim Q_i} \left[\log \frac{P_\theta(x_i, y)}{Q_i(\mathbf{y})} \right] \\
&= \sum_i E_{\mathbf{y} \sim Q_i} [\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{\mathbf{y} \sim Q_i} [\log Q_i(\mathbf{y})]
\end{aligned}$$

Greater than

Let us build an approximation

What we want: Maximize log likelihood$(D \mid \theta) = \mathrm{LL}(\mathrm{D} \mid \theta)$

$$\mathrm{LL}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$= \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} \left(Q_i(\mathbf{y}) \cdot \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(\mathbf{y})} \right)$$

Great
than

The strategy: Let us maximize this lower bound on the likelihood instead

$$\geq \sum_i E_{\mathbf{y} \sim Q_i} \left[\log \frac{P_\theta(x_i, y)}{Q_i(\mathbf{y})} \right]$$

$$= \sum_i E_{\mathbf{y} \sim Q_i} [\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{\mathbf{y} \sim Q_i} [\log Q_i(\mathbf{y})]$$

Expectation Maximization: The strategy

What we want (but can't have)

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

The strategy: Think of log probabilities as random variables

Learn by repeatedly maximizing a lower bound of the log likelihood

$$F(\theta, Q_i) = \sum_i E_{y \sim Q_i}[\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{y \sim Q_i}[\log Q_i(y)]$$

Expectation Maximization: The strategy

Learning by maximizing expected log likelihood of the data

$$F(\theta, Q_i) = \sum_i E_{y \sim Q_i}[\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{y \sim Q_i}[\log Q_i(y)]$$

Expectation Maximization: The strategy

Learning by maximizing expected log likelihood of the data

$$F(\theta, Q_i) = \sum_i E_{y\sim Q_i}[\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{y\sim Q_i}[\log Q_i(y)]$$

Still need to decide what is a good Q_i

What we would like is the one that makes this lower bound tight

$$\log E_{y\sim Q_i}\left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right] \geq E_{y\sim Q_i}\left[\log \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right]$$

Jensen's inequality

Expectation Maximization: The strategy

Learning by maximizing expected log likelihood of the data

$$F(\theta, Q_i) = \sum_i E_{y\sim Q_i}[\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{y\sim Q_i}[\log Q_i(y)]$$

Still need to decide what is a good Q_i

What we would like is the one that makes this lower bound tight

$$\log E_{y\sim Q_i}\left[\frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right] \geq E_{y\sim Q_i}\left[\log \frac{P_\theta(\mathbf{x}_i, \mathbf{y})}{Q_i(y)}\right]$$

Jensen's inequality

We can show that if we had an estimate of the θ, say θ^t, then a tight lower bound is given by setting

$$Q_i(y) = P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

Expectation Maximization

- Initialize the parameters θ^0

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
    - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

  - M-Step: Find θ^{t+1} by maximizing with respect to θ

$$F(\theta, Q^t) = \sum_i E_{y\sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{y\sim Q_i^t}[\log Q_i^t(y)]$$

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

  - M-Step: Find θ^{t+1} by maximizing with respect to θ

$$F(\theta, Q^t) = \sum_i E_{y\sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})] - \sum_i E_{y\sim Q_i^t}[\log Q_i^t(y)]$$

Independent of θ

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

  - M-Step: Find θ^{t+1} by solving the maximization problem

$$\theta^{t+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example $\mathbf{x}_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

Intuitively: What is distribution over the latent variables for this set of parameters

  - M-Step: Find θ^{t+1} by solving the maximization problem

$$\theta^{t+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

Intuitively: What is distribution over the latent variables for this set of parameters

  - M-Step: Find θ^{t+1} by solving the maximization problem

$$\theta^{t+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

Intuitively: Using the current estimate for the hidden variables, what is the best set of parameters for the entire data

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

Given the parameters θ^t, we can compute this function. Why?

  - M-Step: Find θ^{t+1} by solving the maximization problem

$$\theta^{t+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

This step needs can be solved either analytically or algorithmically.

- Return final θ

Expectation Maximization

- Initialize the parameters θ^0
- Repeat until convergence (t = 1, 2, …)
  - E-Step: For every example **x**$_i$, estimate for every y

$$Q_i^t(y) \leftarrow P_{\theta^t}(\mathbf{y} \mid \boldsymbol{x_i})$$

Given the parameters θ^t, we can

EM is a meta-algorithm. To be fully instantiated, we need:

- A definition of the probability distributions,
- An algorithm for the E step
- An algorithm (or an analytical solution) for the M step

ther

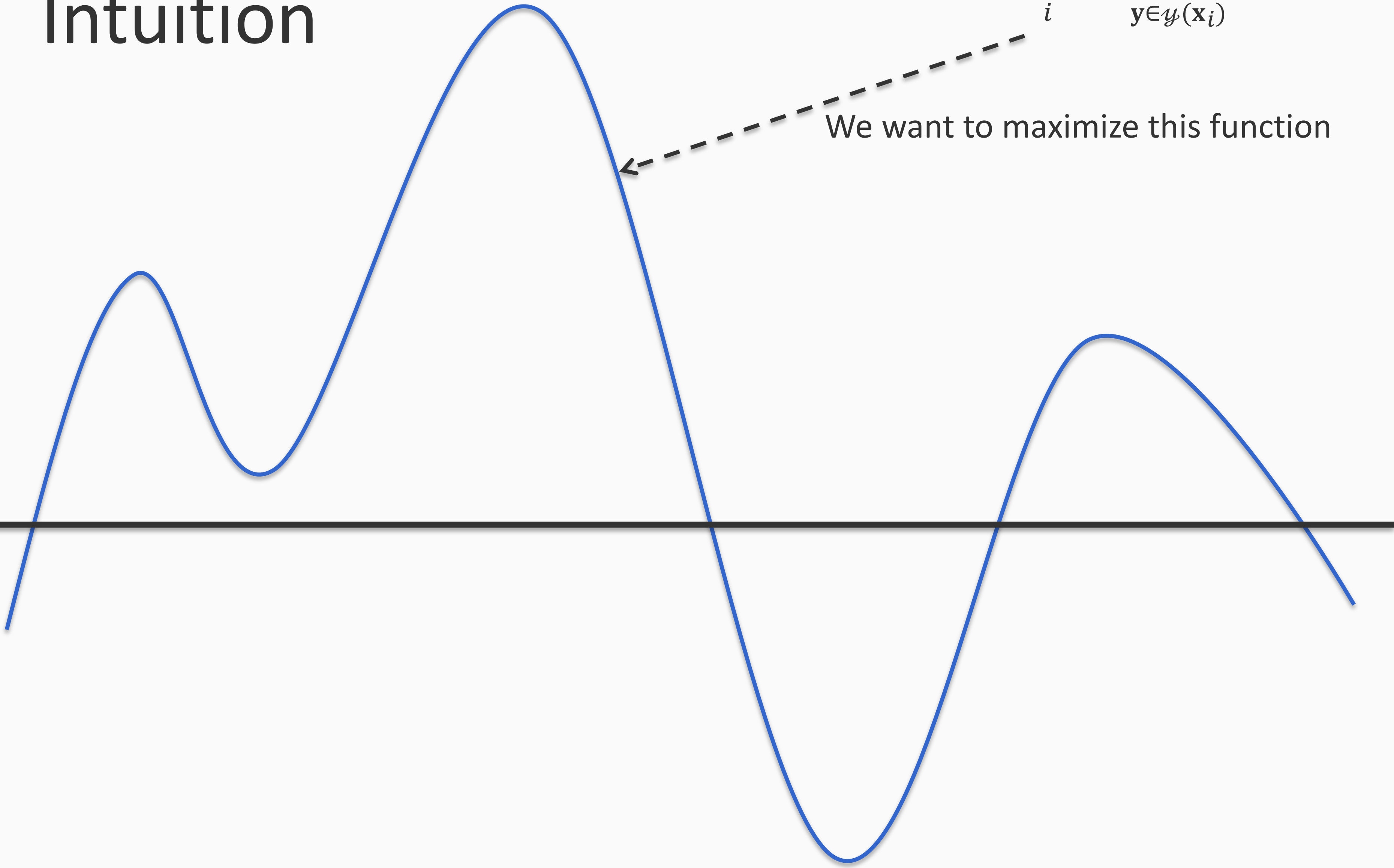
analytically for simple problems or

more often algorithmically.

- Return final θ

Intuition

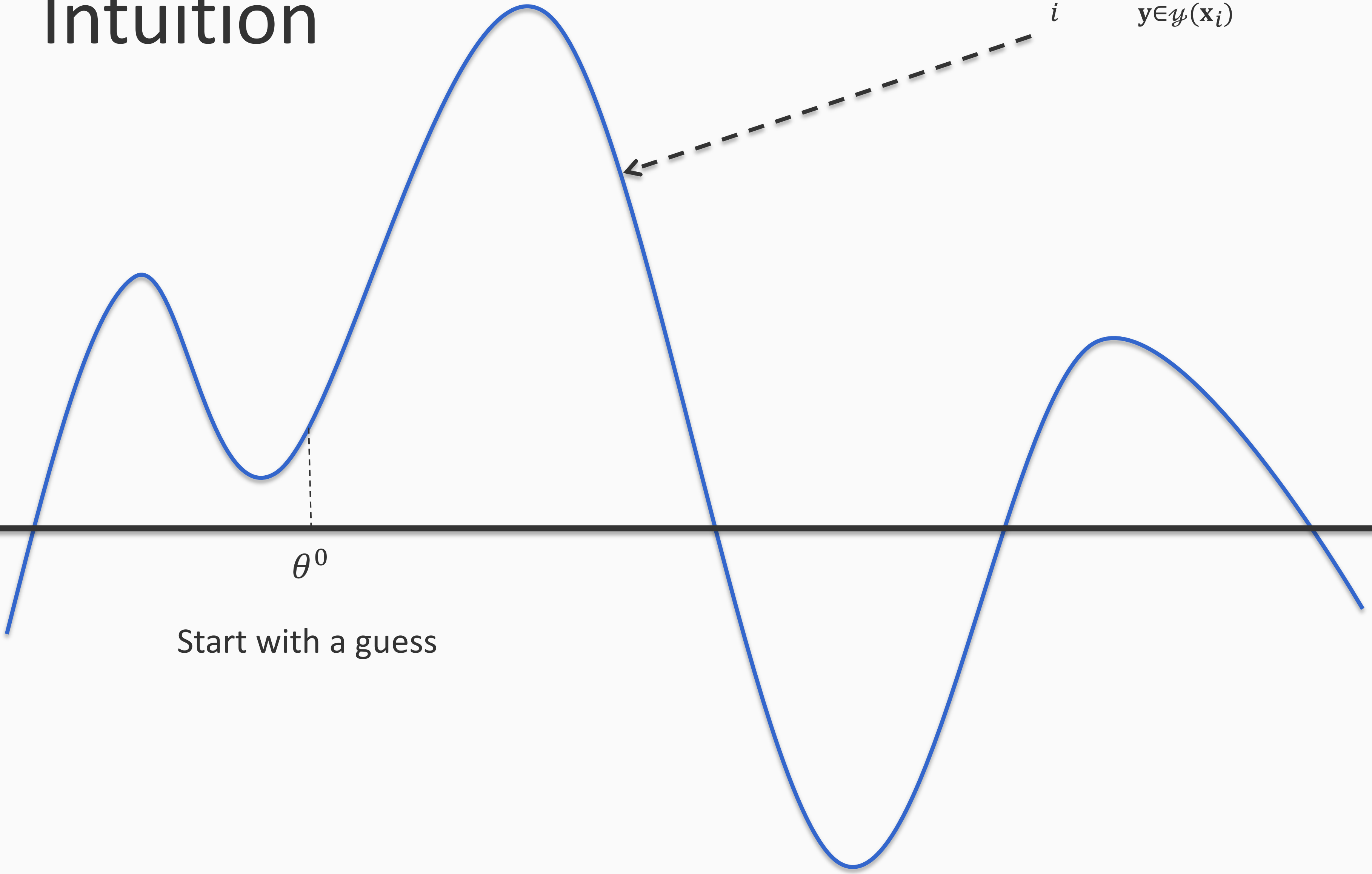
$$\text{log likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$



We want to maximize this function

Intuition

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

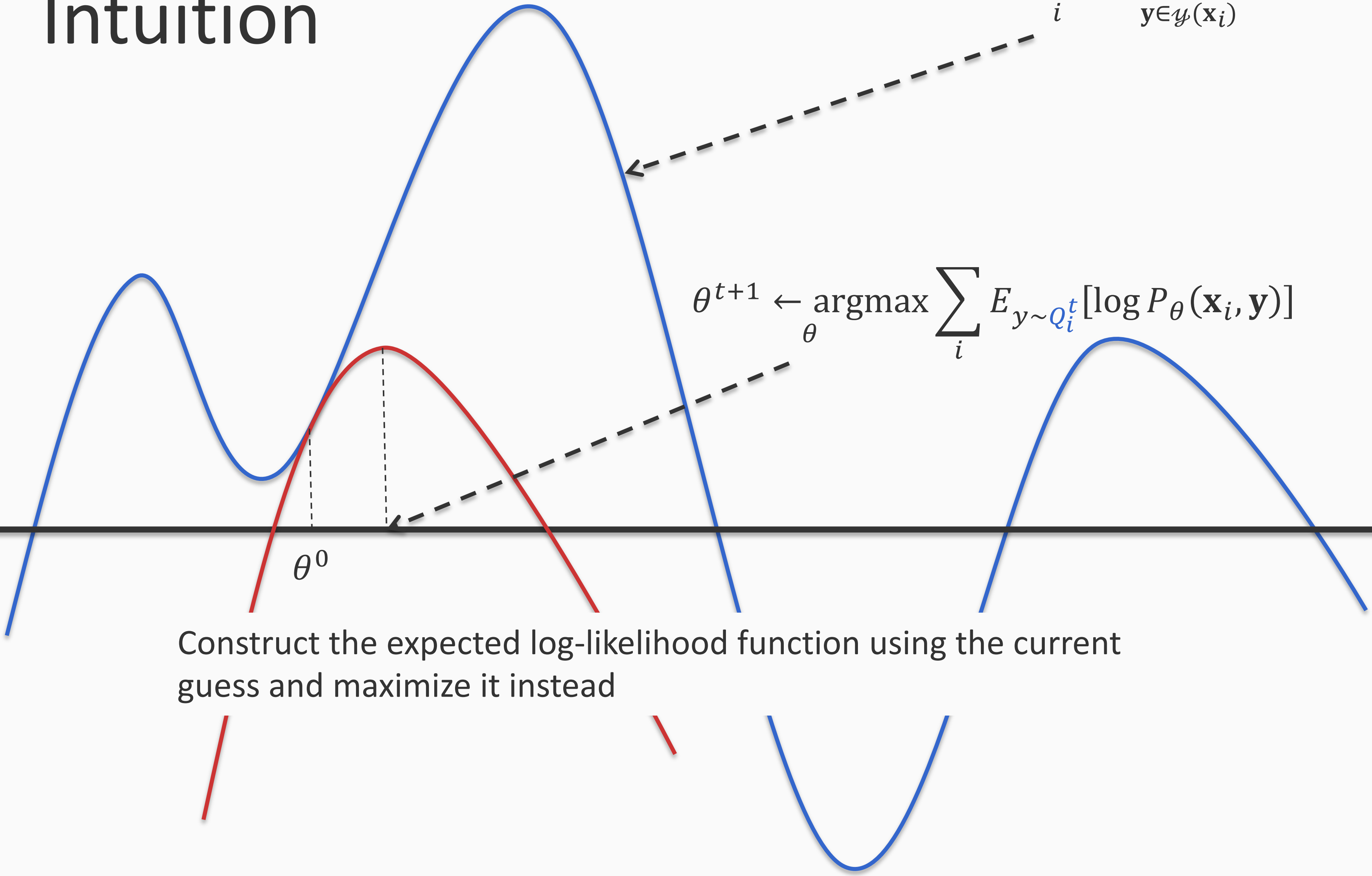


Start with a guess

Intuition

$$\text{log likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

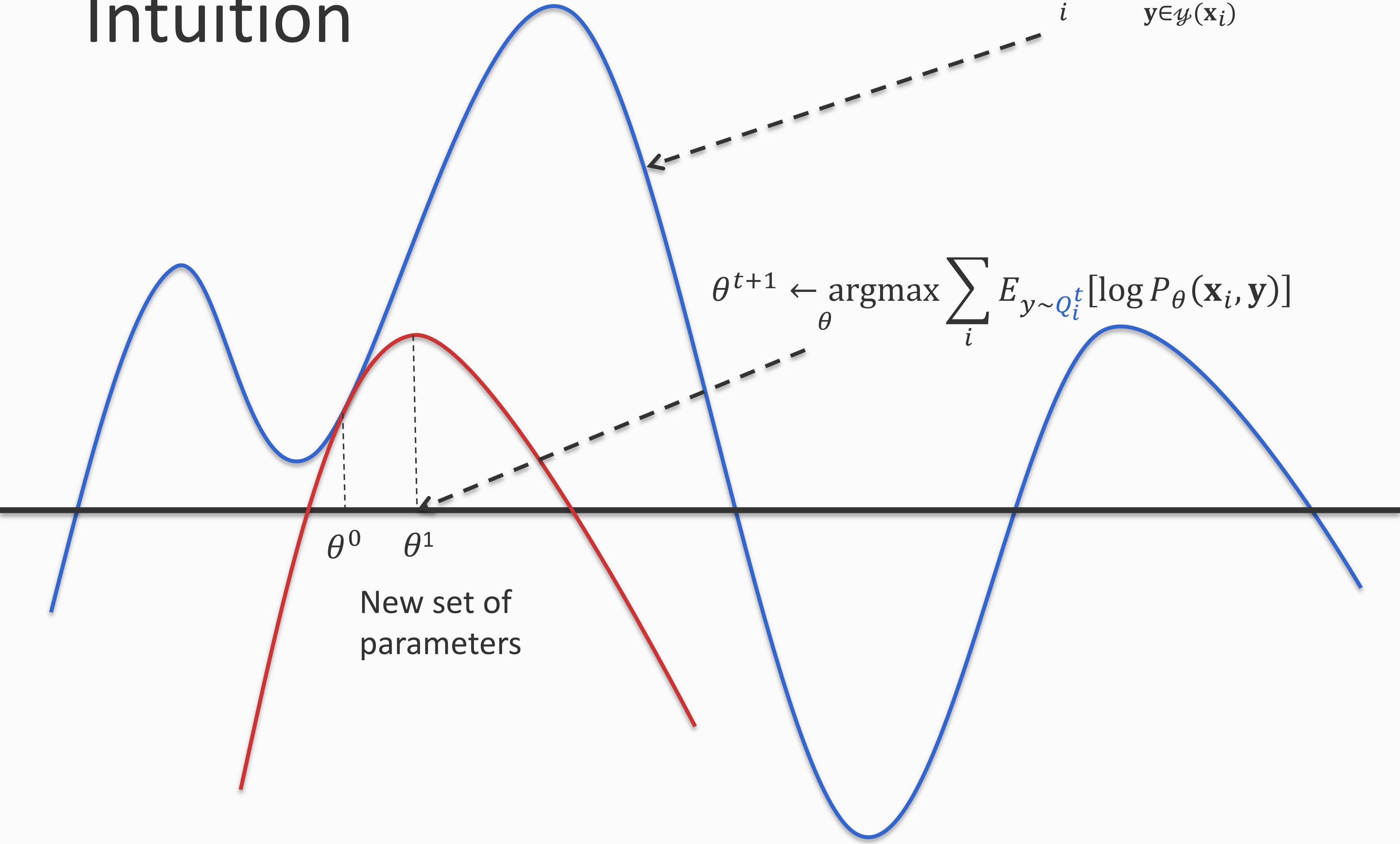
$$\theta^{t+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$



Construct the expected log-likelihood function using the current guess and maximize it instead

Intuition

$$\text{log likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

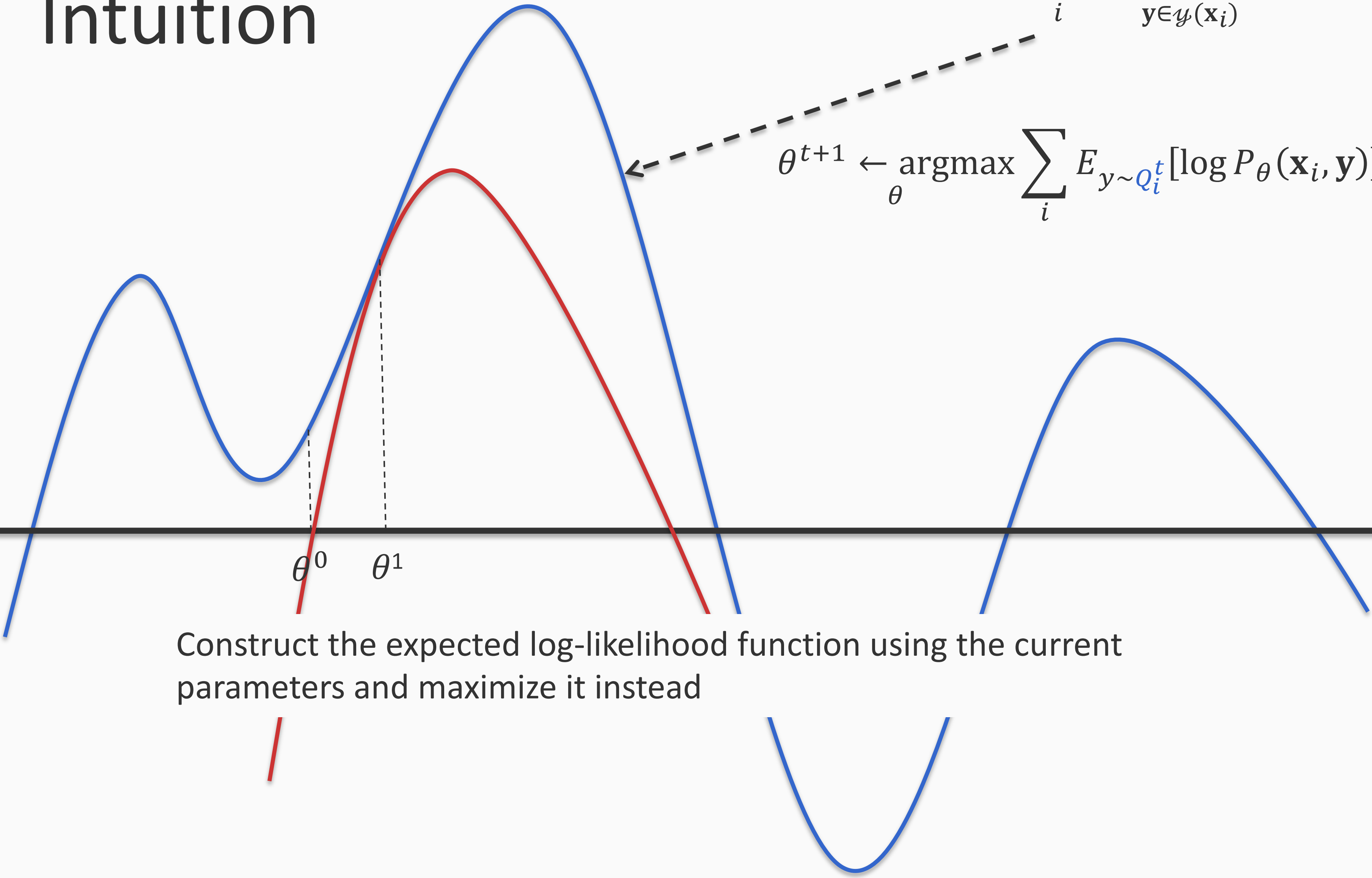


$$\theta^{t+1} \leftarrow \underset{\theta}{\text{argmax}} \sum_i E_{y\sim Q_i^t}[\log P_\theta(\mathbf{x}_i,\mathbf{y})]$$

New set of parameters

Intuition

$$\text{log likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

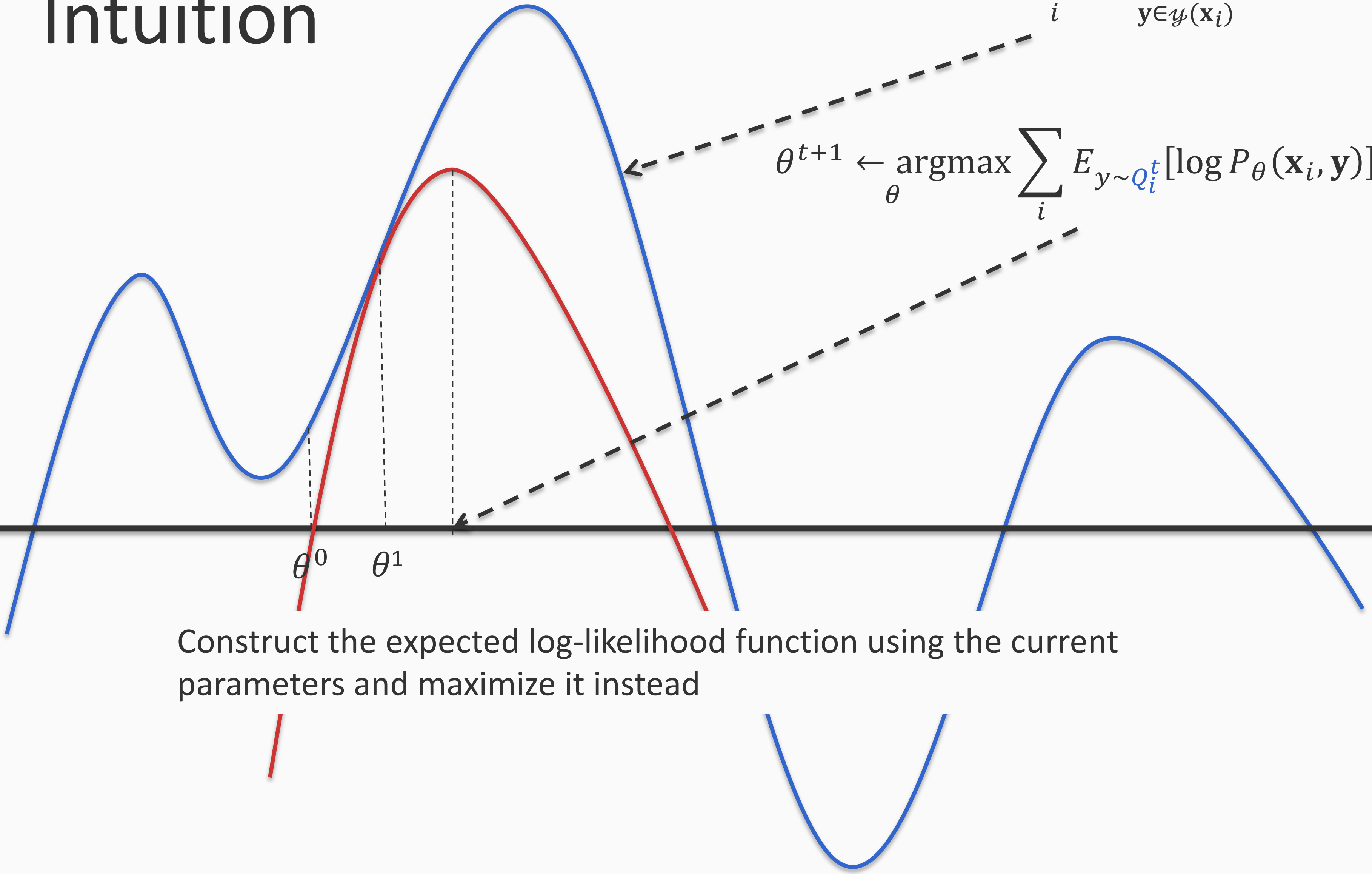


$$\theta^{t+1} \leftarrow \underset{\theta}{\text{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

Construct the expected log-likelihood function using the current parameters and maximize it instead

Intuition

$$\text{log likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$



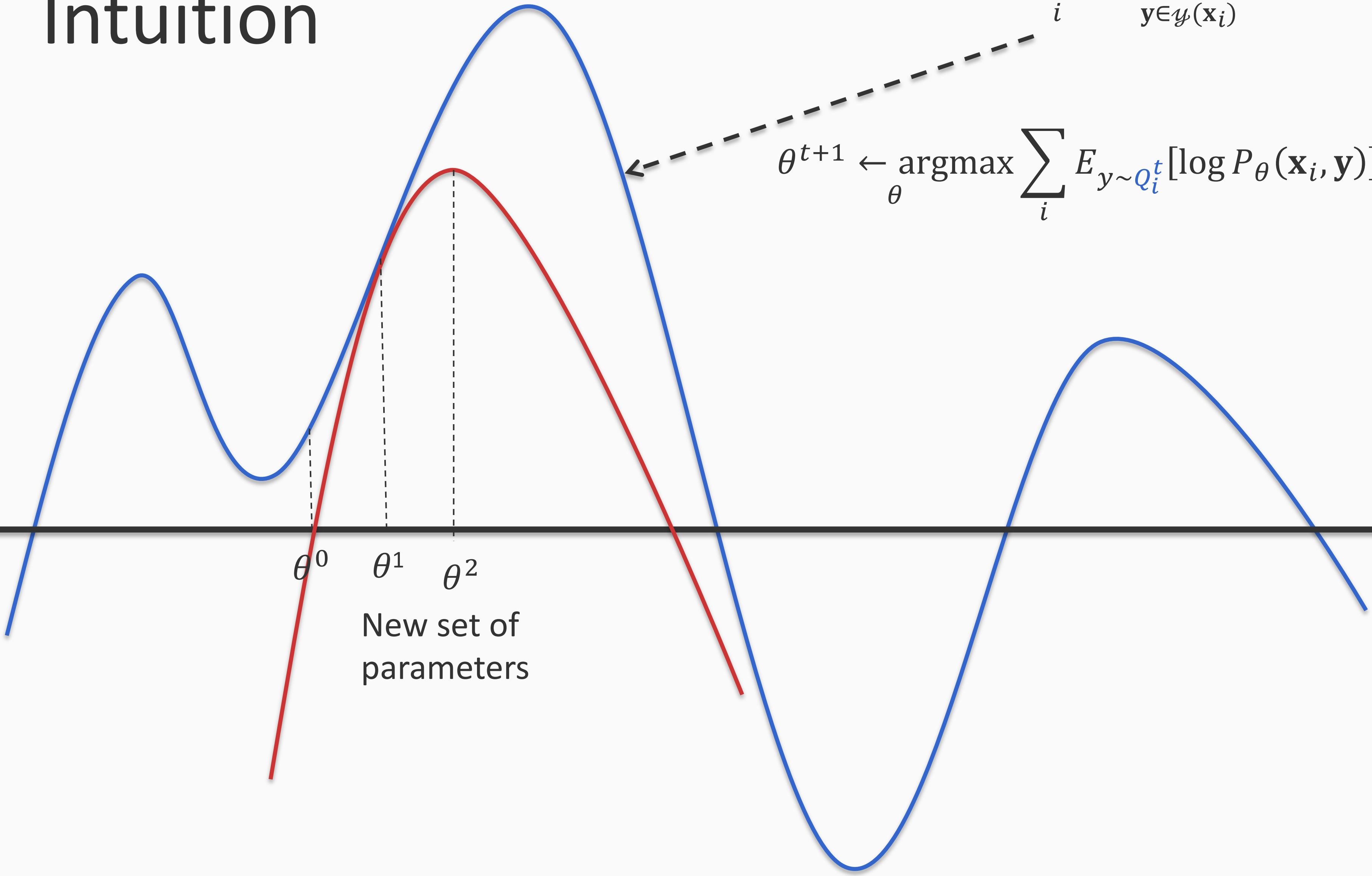
$$\theta^{t+1} \leftarrow \underset{\theta}{\text{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

Construct the expected log-likelihood function using the current parameters and maximize it instead

Intuition

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

$$\theta^{t+1} \leftarrow \operatorname*{argmax}_\theta \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

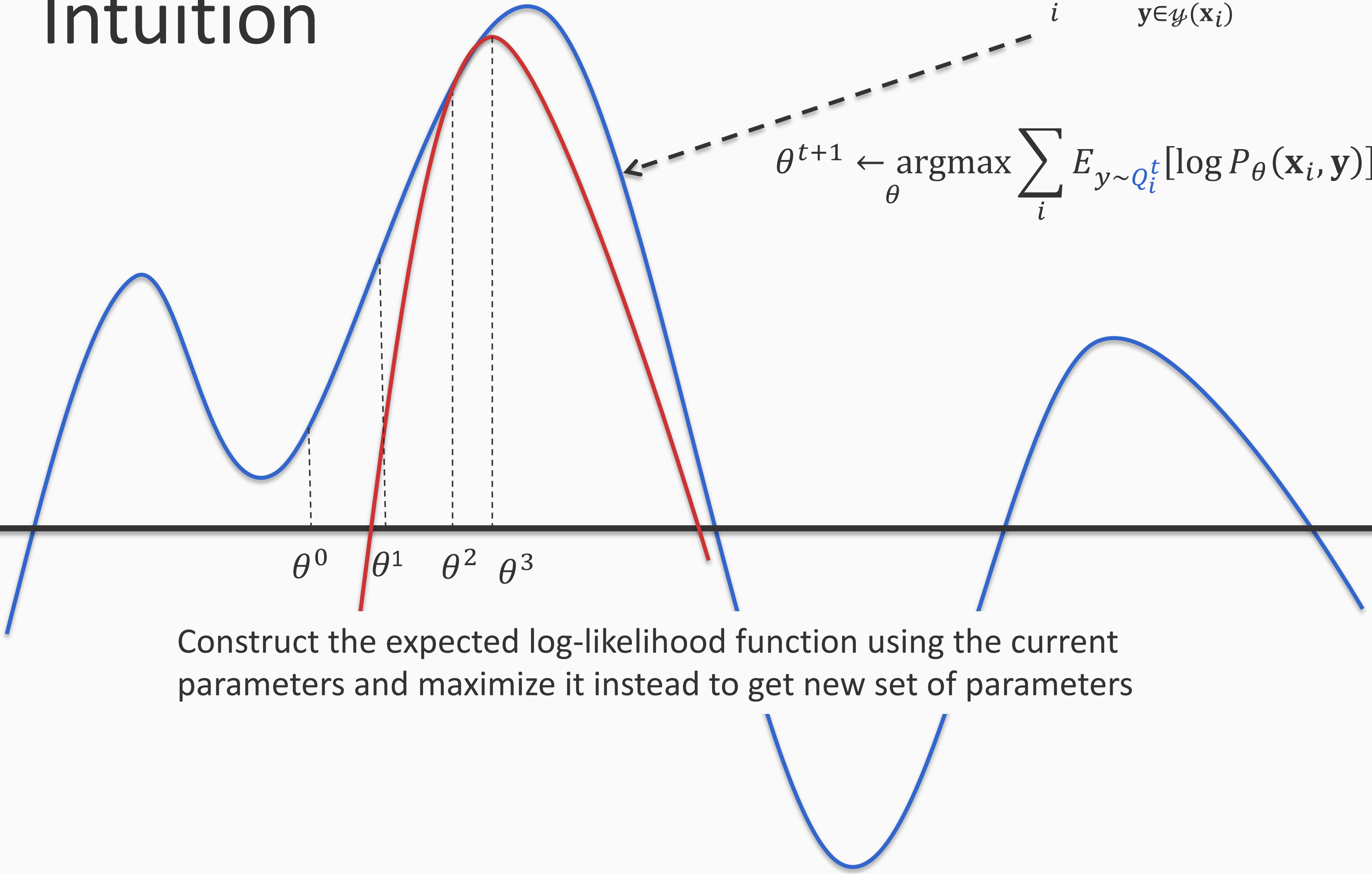


New set of parameters

Intuition

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y}\in\mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$

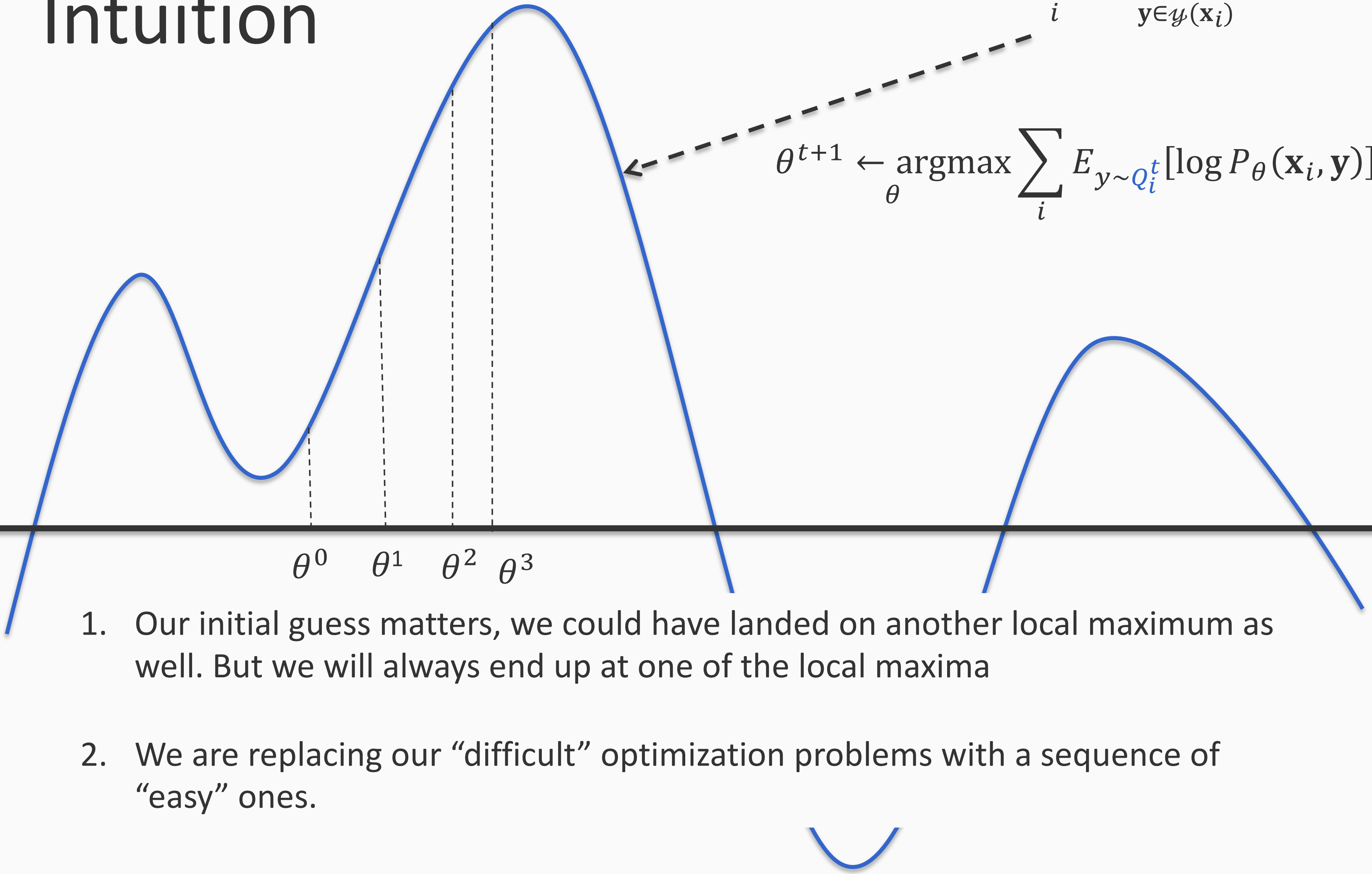
$$\theta^{t+1} \leftarrow \underset{\theta}{\operatorname{argmax}} \sum_i E_{y\sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$



Construct the expected log-likelihood function using the current parameters and maximize it instead to get new set of parameters

Intuition

$$\log \text{likelihood}(D \mid \theta) = \sum_i \log \sum_{\mathbf{y} \in \mathcal{Y}(\mathbf{x}_i)} P_\theta(\mathbf{x}_i, \mathbf{y})$$



$$\theta^{t+1} \leftarrow \underset{\theta}{\text{argmax}} \sum_i E_{y \sim Q_i^t}[\log P_\theta(\mathbf{x}_i, \mathbf{y})]$$

1. Our initial guess matters, we could have landed on another local maximum as well. But we will always end up at one of the local maxima

2. We are replacing our “difficult” optimization problems with a sequence of “easy” ones.

Comments about EM

- Will converge to a local maximum of the log-likelihood
  - Different initializations can give us different final estimates of probabilities

- How many iterations
  - Till convergence. Keep track of expected log likelihood across iterations and if the change is smaller than some ^{2} then stop

- What we need to specify the learning algorithm
  - A task-specific definition of the probabilities
  - A way to solve the maximization (the M-step)

- Gives us a general template for building models when we don’t have fully labeled data

This lecture

- What is weak supervision?

- The expectation maximization algorithm
  - A general template for learning with weak supervision

- Learning with latent variables

- Learning with constraints

Learning with latent variables

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- The partial structure could be empty, giving us the fully unsupervised setting
- Or it could have labeled some parts of the structure for each example

Learning with latent variables

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- Initialize the parameters θ^0 randomly

- Iterate for t = 1, 2,… :
  - Use the current model to "complete" each example
  - $\theta^{t+1} \leftarrow$ Train model using the newly completed examples

- Return the final set of parameters

Learning with latent variables

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- Initialize the parameters θ^0 randomly

This step requires some type of inference

- Iterate for t = 1, 2,… :
  - Use the current model to “complete” each example
  - $\theta^{t+1} \leftarrow$ Train model using the newly completed examples

- Return the final set of parameters

Learning with latent variables

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- Initialize the parameters θ^0 randomly

This step requires some type of inference

- Iterate for t = 1, 2,… :
  - Use the current model to "complete" each example
  - $\theta^{t+1} \leftarrow$ Train model using the newly completed examples

- Return the final set of parameters

The completed examples could involve a hard decision (i.e. a structure), or a soft decision(i.e. a probability for each label)

Specific instances of this idea

- The EM algorithm for hidden Markov Models: The Baum Welch algorithm
- When the learner in the model update step is the structural SVM, we get the latent structured SVM
  - (with some caveats about the cost of misclassification, see Yu & Joachims 2009 for details)
- We can instantiate an EM algorithm for CRF style models as well

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

A fully labeled example may look like

$\boldsymbol{x}$	B	B	C	A

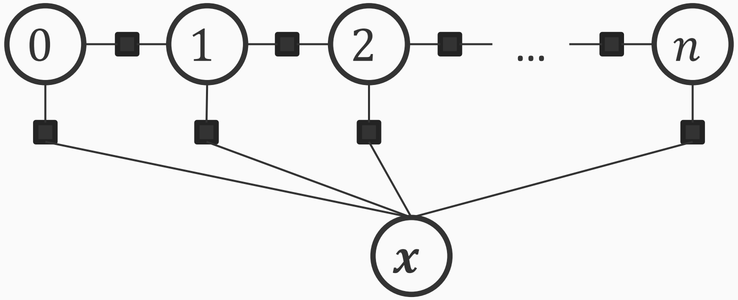
A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

A fully labeled example may look like

$\boldsymbol{x}$	B	B	C	A

Suppose we are assuming this model:



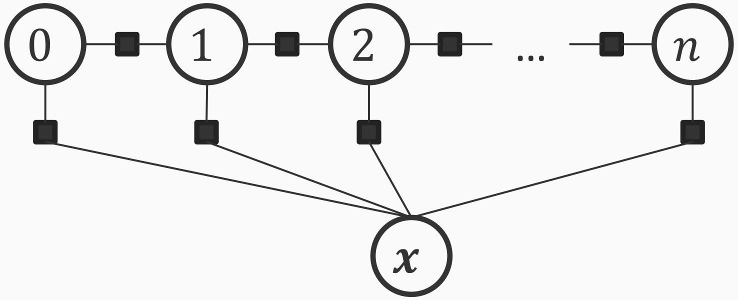
A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

A fully labeled example may look like

$\boldsymbol{x}$	B	B	C	A

Suppose we are assuming this model:



We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

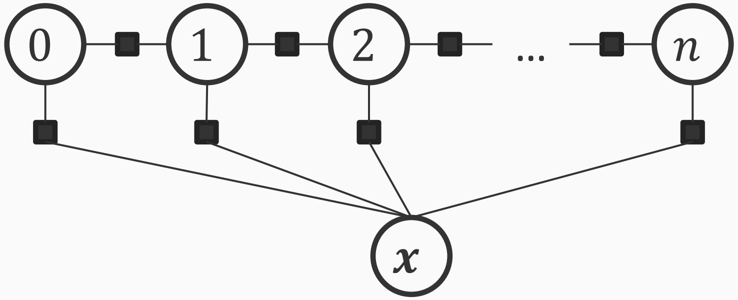
A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

A fully labeled example may look like

$\boldsymbol{x}$	B	B	C	A

Suppose we are assuming this model:



We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

What can we do with this data?

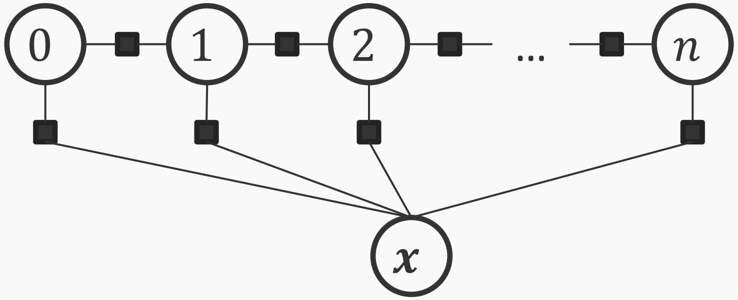
A simple example

Setting: Inputs are denoted by $\boldsymbol{x}_i$, and each input is associated with a collection of outputs each of which can be A, B or C.

A fully labeled example may look like

$\boldsymbol{x}$	B	B	C	A

Suppose we are assuming this model:



We have three training examples:

$\boldsymbol{x}_1$	_	A	_	
$\boldsymbol{x}_2$	B	_	_	C
$\boldsymbol{x}_3$	A	_		

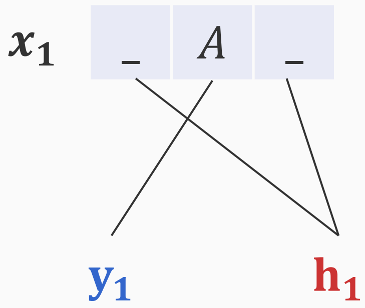
What can we do with this data?

If we had the parameters of the model, we can predict the missing labels

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:



$\boldsymbol{x_1}$ _ A _

$\mathbf{y_1}$ $\mathbf{h_1}$

The annotated part of the output for this example

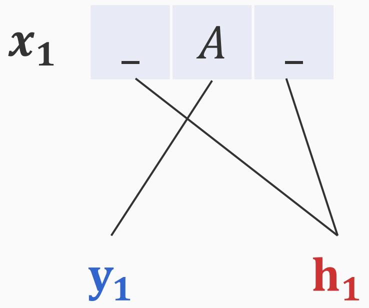
The unannotated (i.e. hidden) part of the output for this example

If we had the parameters of the model, we can predict the missing labels

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:



The annotated part of the output for this example

The unannotated (i.e. hidden) part of the output for this example

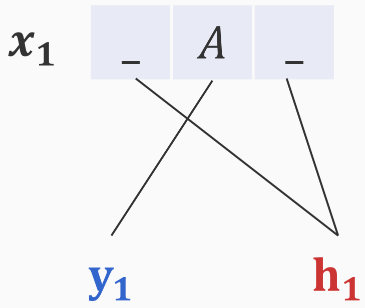
If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:



The annotated part of the output for this example

The unannotated (i.e. hidden) part of the output for this example

If we had the parameters of the model, we can predict the missing labels

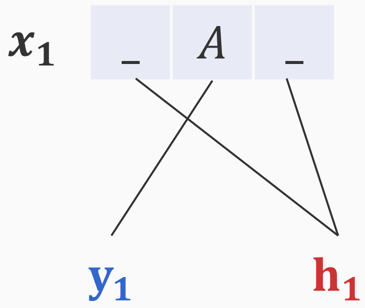
Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

We can ask: What is the best value for $\mathbf{h_1}$ for this example as per the current model?

A simple example

Setting: Inputs are denoted by $\boldsymbol{x}_i$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:



The annotated part of the output for this example

The unannotated (i.e. hidden) part of the output for this example

If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

We can ask: What is the best value for $\mathbf{h}_1$ for this example as per the current model?

$$\mathbf{h}_1^* = \operatorname*{argmax}_{\mathbf{h}} score(\mathbf{x}_1, \mathbf{y}_1, \mathbf{h}, \mathbf{w})$$

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

We can ask: What is the best value for $\boldsymbol{h}$ for each example as per the current model?

$$\mathbf{h}_i^* = \underset{\mathbf{h}}{\operatorname{argmax}}\, score(\mathbf{x_i}, \boldsymbol{y}_i, \mathbf{h}, \mathbf{w})$$

Perform inference to find best estimates for hidden variables

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

↓

Complete the missing labels using $\mathbf{w}$

$\boldsymbol{x_1}$	B	A	B	
$\boldsymbol{x_2}$	B	A	C	C
$\boldsymbol{x_3}$	A	A		

If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

Complete the missing labels using $\mathbf{w}$

$\boldsymbol{x_1}$	B	A	B	
$\boldsymbol{x_2}$	B	A	C	C
$\boldsymbol{x_3}$	A	A		

If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

Use this newly completed data to train new update for $\mathbf{w}$

Use any loss function

A simple example

Setting: Inputs are denoted by $\boldsymbol{x_i}$, and each input is associated with a collection of outputs each of which can be A, B or C.

We have three training examples:

$\boldsymbol{x_1}$	_	A	_	
$\boldsymbol{x_2}$	B	_	_	C
$\boldsymbol{x_3}$	A	_		

Complete the missing labels using $\mathbf{w}$

$\boldsymbol{x_1}$	B	A	B	
$\boldsymbol{x_2}$	B	A	C	C
$\boldsymbol{x_3}$	A	A		

Use this newly completed data to train new update for $\mathbf{w}$

Use any loss function

If we had the parameters of the model, we can predict the missing labels

Suppose the current model was denoted by $\mathbf{w}$ and the corresponding scoring function as per the factor graph is $score(\mathbf{x}, \mathbf{y}, \mathbf{h}, \mathbf{w})$

This lecture

- What is weak supervision?
- The expectation maximization algorithm
  - A general template for learning with weak supervision
- Learning with latent variables
- Learning with constraints

Constraint Driven Learning

Suppose we have some constraints about the output structures, but don’t have any (or much) labeled data

Examples:

- Every part of speech sequence should have a verb
- Every image patch that is recognized as a bicycle should have at least one patch that is recognized as a wheel

Constraint Driven Learning

Suppose we have some constraints about the output structures, but don’t have any (or much) labeled data

Examples:

- Every part of speech sequence should have a verb
- Every image patch that is recognized as a bicycle should have at least one patch that is recognized as a wheel

How do we incorporate constraints into the learning process?

Constraint Driven Learning

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- Initialize the parameters θ^0 randomly

This step requires some type of inference

- Iterate for t = 1, 2,… :
    - – Use the current model to “complete” each example
    - – $\theta^{t+1} \leftarrow$ Train model using the newly completed examples

- Return the final set of parameters

Constraint Driven Learning

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- Initialize the parameters θ^0 randomly
- Iterate for t = 1, 2,… :
  - Use the current model to “complete” each example with constrained inference
  - $\theta^{t+1} \leftarrow$ Train model using the newly completed examples
- Return the final set of parameters

This step requires some type of inference

Constraint Driven Learning

Given a dataset with examples $\mathbf{x}_i$ that are labeled with a partial structure $\mathbf{y}_i$

- Initialize the parameters θ^0 randomly

- Iterate for t = 1, 2,… :
  - Use the current model to "complete" each example with constrained inference
  - $\theta^{t+1} \leftarrow$ Train model using the newly completed examples

- Return the final set of parameters

Variants of this idea exist. See [Chang et al, 2007, 2012]

Learning with constraints

General idea: In the step where we expand (or complete) the partially labeled structures, use knowledge to guide the learner

- Any of the inference algorithms we have seen can be used

- *Posterior regularization* Ganchev et al [2010] show that this idea applies for the case where we are not making a hard decision, but using the EM algorithm we have seen before

- Why do constraints help?
  - They restrict the search space for inference algorithms using knowledge
  - Constraints guide learning by making the learner explore the "better" parts of the parameter space

Final words

- Annotated data is expensive, more so for structures

- We can and should design learning algorithms that take advantage of any available supervision
  - Could be labeled, unlabeled or partially labeled examples
  - Could be knowledge

- The EM algorithm and its variants form a general schematic for designing such algorithms
  - Generally has one phase where we predict labels or distributions over labels for examples, and another where we use these predictions to improve the model

- Enforcing constraints during the prediction phase the can help reduce demand on labeled data