Convexity, Maximum Likelihood and All That

Adam Berger
School of Computer Science
Carnegie Mellon University
aberger@cs.cmu.edu

Abstract

This note is meant as a gentle but comprehensive introduction to the *expectation-maximization (EM)* and *improved iterative scaling (IIS)* algorithms, two popular techniques in maximum likelihood estimation. The focus in this tutorial is on the foundation common to the two algorithms: convex functions and their convenient properties. Where examples are called for, we draw from applications in human language technology.

1 Introduction

The task is to characterize the behavior of a real or imaginary stochastic process. By "stochastic process," we mean something which generates a sequence of observable output values. These values can be viewed as a discrete time series. We denote a single observation by y, a random variable which takes on values in some alphabet $\mathcal{Y}$. The modelling problem is to come up with an accurate (in a sense made precise later) model $p(y)$ of the process. If the identity of y is influenced by some conditioning information $x \in \mathcal{X}$, then we might seek instead a conditional model $p(y \mid x)$.

The *language modelling problem*, for instance, is to construct a conditional probability distribution function (p.d.f.) $p(y \mid x)$, where y is identity of the next word in some text, and x is the conditioning information, such as the identity of the preceding words. Machine translation [Br91], word-sense disambiguation [Be96], part-of-speech tagging [Me90] and parsing of natural language [Bl92] are just a few other human language-related domains involving stochastic modelling.[1]

We will concern ourselves in this document with deriving the expectation- maximization (EM) and improved iterative scaling (IIS) algorithms. Both discover an optimal (locally for EM, globally for IIS) member of a parametric family of models. Before we can shed light

[1] Purists may object to the abuse of notation here, committed in the sake of simplicity: x can refer to a random variable or a particular value of the random variable; $p(x)$ can refer both to a conditional p.d.f. and to the probability assigned by the p.d.f. to the event x. The proper meaning should always be clear from context.

on either algorithm, we need to introduce some of the major players: maximum likelihood, convexity, and auxiliary functions.

1.1 Maximum likelihood

Let us imagine that in some observed sample of output of the process, the symbol y was generated $\tilde{c}(y)$ times. We write $\tilde{p}(y) \equiv \tilde{c}(y)/\sum_{y\in\mathcal{Y}} \tilde{c}(y)$ for the probability of symbol y, and $\tilde{p}$ to represent the entire sample.

Then the probability (or *likelihood*) of the sample $\tilde{p}$ with respect to an unconditional model $p(y)$ is

$$Pr(\tilde{p} \mid p) = \prod_{y\in\mathcal{Y}} p(y)^{\tilde{c}(y)} \tag{1}$$

Both the EM and the IIS algorithms are *maximum likelihood*, meaning that within some prescribed family $\mathcal{F}$ of models, these algorithms seek

$$p^{\star} \equiv \operatorname*{argmax}_{p\in\mathcal{F}} Pr(\tilde{p} \mid p) \tag{2}$$

The likelihood is monotonically related to the average per-symbol log-likelihood,

$$L(\tilde{p} \mid p) \equiv \sum_{y\in\mathcal{Y}} \tilde{p}(y) \log p(y), \tag{3}$$

So $p^{\star} = \operatorname{argmax}_{p\in\mathcal{F}} L(\tilde{p} \mid p)$. Since it's so much more convenient, we will take this, rather than (2), as the definition of the maximum likelihood model.

A Bayesian aside

Maximum likelihood estimation is just a degenerate form of Bayesian modelling where the prior over models p is uniform. That is,

$$\begin{aligned}
\operatorname*{argmax}_{p} Pr(p \mid \tilde{p}) &= \operatorname*{argmax}_{p} \frac{Pr(\tilde{p} \mid p) Pr(p)}{Pr(\tilde{p})} && \text{by Bayes' law} \\
&= \operatorname{argmax}_p Pr(\tilde{p} \mid p) Pr(p) && \text{since } Pr(\tilde{p}) \text{ is independent of } p \\
&= \operatorname{argmax}_p Pr(\tilde{p} \mid p) && \text{if } Pr(p) \text{ is uniform} \\
&= \operatorname{argmax}_p L(\tilde{p} \mid p)
\end{aligned}$$

But despite the Bayesian connection, there's really no theoretically compelling reason to rely on maximum likelihood for parameter estimation. After all, the sample of observed output which constitutes $\tilde{p}$ is only a representative sample of the process being modelled. The justification for maximum likelihood lies in its empirical success in selected applications (such as speech recognition [Ba83]) and in the convenient algorithms it gives rise to—such as the EM and IIS algorithms.

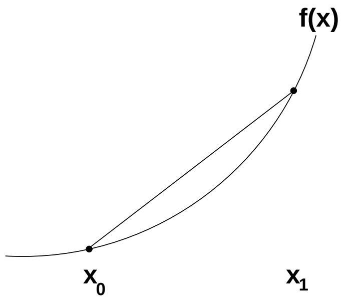
On the flip side, maximum likelihood's prime liability is the *overfitting* problem. Overfitting is what happens when a model does its job too well, becoming so faithful to the data it was trained on that it doesn't generalize well to new data. Backing up to some middle ground requires ad hoc techniques such as smoothing the model parameters [Je80] [Ka87] using some data held out from the training. There have been some efforts to introduce alternative parameter-estimation approaches which avoid this difficulty [Br87].

1.2 Convexity

Recall that a function $f(x)$ is *convex* ("concave up") if

$$f(\alpha x_0 + (1-\alpha)x_1) \leq \alpha f(x_0) + (1-\alpha)f(x_1) \quad \text{for all } 0 \leq \alpha \leq 1 \tag{4}$$

That is, if one selects any two points x_0 and x_1 in the domain of a convex function, the function always lies on or under the chord connecting x_0 and x_1:



A sufficient condition for convexity—the one taught in high school calculus—is that f is convex if and only if $f''(x) \geq 0$. But this is not a necessary condition, since f may not be everywhere differentiable; (4) is preferable because it applies even to non-differentiable functions, such as $f(x) = \mid x \mid$ at $x = 0$.

A multivariate function may, of course, be convex in any number of its arguments.

Jensen's inequality

Among the most useful properties of convex functions is that if f is convex in x, then

$$\begin{aligned} f(E[x]) &\leq E[f(x)] \quad \text{or} \\ f(\sum_x p(x)x) &\leq \sum_x p(x)f(x) \end{aligned} \tag{5}$$

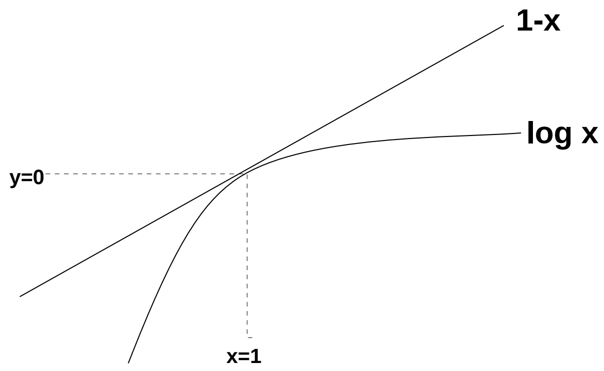
where $p(x)$ is a p.d.f. This follows from (4) by a simple inductive proof.

What this means, for example, is that (for any p.d.f. p), the following two conditions hold:

$$\sum_x p(x)\log q(x) \geq \log \sum_x p(x)q(x) \qquad \text{since } -\log \text{ is convex} \tag{6}$$

$$\exp \sum_x p(x)q(x) \leq \sum_x p(x)\exp q(x) \qquad \text{since exp is convex} \tag{7}$$

We'll also find use for the fact that a concave function always lies below its tangent; in particular, $\log x$ lies below its tangent at $x = 1$:



1.3 Auxiliary functions

At their most general, auxiliary functions are simply pointwise lower (or upper) bounds on a function. We've already seen an example: $x-1$ is an auxiliary function for $\log x$ in the sense that $x - 1 \geq \log x$ for all x. This observation might prove useful if we're trying to establish that some function $f(x)$ lies on or above $\log x$: if we can show $f(x)$ lies on or above $x - 1$, then we're done, since $x - 1$ itself lies above $\log x$. (Incidentally, it's also true that $\log x$ is an auxiliary function for $x - 1$, albeit in the other direction).

We'll be making use of a particular type of auxiliary function: one that bounds the change in log-likelihood between two models. If p_λ is one model and $p_{\lambda'}$ another, then we'll be interested in the quantity $L(\tilde{p} \mid p_{\lambda'}) - L(\tilde{p} \mid p_\lambda)$, the *gain* in log-likelihood from using $p_{\lambda'}$ instead of p_λ. For the remainder of this discussion, we'll define $A(\lambda', \lambda)$ to be an auxiliary function only if

- $L(\tilde{p} \mid p_{\lambda'}) - L(\tilde{p} \mid p_\lambda) \geq A(\lambda', \lambda)$
- $A(\lambda, \lambda) = 0$

Together, these conditions imply that if we can find a λ' such that $A(\lambda', \lambda) > 0$, then $p_{\lambda'}$ is a better model than p_λ—in a maximum likelihood sense, of course.

2 EM algorithm

The standard setting for the EM algorithm is as follows. We can observe data y from the stochastic process we're trying to model, but this data is an *incomplete* representation of the process. We denote the complete data by h (h for "partly hidden"). One might think of the y_i as the observed output at time i, and h_i as the state of the process at time i.

The EM algorithm is an especially convenient tool for handling Hidden Markov Models (HMMs). HMMs are a generalization of traditional Markov models: whereas each state-to-state transition on a Markov model causes a symbol to be emitted deterministically, each state-state transition on an HMM contains a *probability distribution* over possible emitted symbols. Using the EM algorithm, we can think of the state as the hidden information and the emitted symbol as the observed output. For example, in an HMM part-of-speech model, the observable data are the words and the hidden states are the parts of speech.

The EM algorithm arises in other human-language settings as well. In a parsing model, the words are again the observed output, but now the hidden state is the parse of the sentence.

In [Br91], a English-French translation model is described in which the alignment between the words in the French sentence and its translation represents the hidden information.

We postulate a parametric model $p_\lambda(y,h)$ of the process, with marginal distribution $p_\lambda(y) = \sum_h p_\lambda(y,h)$. Given some empirical sample $\tilde{p}(y)$, the principle of maximum likelihood dictates that we find the λ which maximizes its likelihood. The difference in log-likelihood between models $p_{\lambda'}$ and p_λ is

$$
\begin{aligned}
L(\tilde{p} \mid p_{\lambda'}) - L(\tilde{p} \mid p_\lambda) &= \sum_y \tilde{p}(y) \log \frac{p_{\lambda'}(y)}{p_\lambda(y)} \\
&= \sum_y \tilde{p}(y) \log \sum_h \frac{p_{\lambda'}(y,h)}{p_\lambda(y)} \\
&= \sum_y \tilde{p}(y) \log \sum_h \frac{p_{\lambda'}(y,h) p_\lambda(h \mid y)}{p_\lambda(y,h)} \qquad \text{applying Bayes' law to } p_\lambda(y) \\
&\geq \underbrace{\sum_y \tilde{p}(y) \sum_h p_\lambda(h \mid y) \log \frac{p_{\lambda'}(y,h)}{p_\lambda(y,h)}}_{\text{Call this } Q(\lambda' \mid \lambda)} \qquad \text{applying (6)} \qquad (8)
\end{aligned}
$$

We've established that $L(\tilde{p} \mid p_{\lambda'}) - L(\tilde{p} \mid p_\lambda) \geq Q(\lambda' \mid \lambda)$. It's also true (by inspection) that $Q(\lambda \mid \lambda) = 0$. Together, these earn Q the honored title of auxiliary function to L. If we can find a λ' for which $Q(\lambda' \mid \lambda) > 0$, then $p_{\lambda'}$ has a higher (log)-likelihood than p_λ.

This observation is the basis of the EM (expectation-maximization) algorithm:

EM Algorithm

Initialization: `pick a starting value` λ_0 `for the parameters`

`Repeat until log-likelihood convergences`

E-step: `compute` $Q(\lambda' \mid \lambda_i)$

M-step: $\lambda \leftarrow \operatorname{argmax}_{\lambda'} Q(\lambda' \mid \lambda)$

A few points are in order about the algorithm.

- The algorithm is greedy, insofar as it attempts to take the best step from the current λ at each iteration, paying no heed to the global behavior of $L(\tilde{p} \mid p_\lambda)$. The line of reasoning culminating in (8) established that each step of the EM algorithm can never produce an inferior model. But this doesn't rule out the possibility of
  - Getting "stuck" at a local maximum
  - Toggling between two local maxima with very different λ but identical likelihoods.

  Under certain assumptions, it can be shown [La93a] that $\lim_n \Phi^n = \Phi^\star$; that is, eventually the EM algorithm converges to the optimal parameter values. Unfortunately, these assumptions are rather restrictive and aren't typically met in practice.

It may very well happen that the space is very "bumpy," with lots of local maxima. In this case, the result of the EM algorithm depends on the starting value λ_0; The algorithm might very well end up at a saddle point. One can enlist any number of heuristics for high-dimensional search in an effort to find the global maximum, such as selecting a number of different starting points, searching by simulating annealing, and so on.

- Along the same line, if each iteration is computationally expensive, it can sometimes pay to try to speed convergence by using second-derivative information. This technique is known variously as Aitken's acceleration algorithm or "stretching." Often it's unviable because Q'' is hard to compute.

- In certain settings it can be hard to maximize $Q(\lambda' \mid \lambda)$, but rather easy to find *some* λ' for which $Q(\lambda' \mid \lambda) > 0$. But that's just fine: picking this λ' still improves the likelihood, though the algorithm is no longer greedy and may well run slower. This version of the algorithm—replacing the M-step of the algorithm with an step which simply takes a step in the right direction, rather than the maximal step in the right direction—is known as the GEM algorithm (G for "generalized").

- A practical issue concerning the EM algorithm is that the sum over the hidden states h in computing (8) can, in practice, be an exponential sum. For instance, the hidden state might represent part-of-speech labelings for a sentence. If there exist T different part of speech labels, then a sentence of length n has T^n possible labelings, and thus the sum is over T^n hidden states. Often some cleverness suffices to sidestep this computational hurdle—usually by relying on some underlying Markov property of the model. Such cleverness is what distinguishes the forward-backward algorithm [Po88].

Example: linear interpolation

A quite typical problem in statistical modelling is to construct a "hyper-model" which is the linear interpolation of a collection of models. We start with an observed sample of output $\{y_1, y_2, \ldots y_T\}$ and a collection of models $p_1(y), p_2(y) \ldots p_N(y)$. We seek the maximum likelihood member of the family of distributions

$$\mathcal{F} \equiv \left\{ p_\lambda(y) = \sum_i \lambda_i p_i(y) \mid \lambda_i \in [0,1] \text{ and } \sum_i \lambda_i = 1 \right\}$$

Members of $\mathcal{F}$ are just linear interpolations of the individual models p_i, with different members distributing their weights differently across the models. The problem is to find the best distribution of weights, namely $p^\star_\lambda \equiv \text{argmax}_{p\in\mathcal{F}} L(\tilde{p} \mid p)$. On the face of it, this appears to be an $(N-1)$-dimensional search problem. But the problem yields quite easily to an EM approach. Imagine the interpolated model p_λ is at any time in one of N states, $s \in \{1, 2, \ldots N\}$, with

- λ_i: the *a priori* probability of being in state i;

- $p_\lambda(s = i, y) = \lambda_i p_i(y)$: the probability of being in state i and producing output y;

- $p_\lambda(s = i \mid y) = \dfrac{\lambda_i p_i(y)}{\sum_i \lambda_i p_i(y)}$: the probability of being in state i, *given* that y is the current output

A convenient way to think of this is that in state i, the interpolated model relies on the i'th model. The appropriate version of (8) is, in this case,

$$Q(\lambda' \mid \lambda) = \sum_y \tilde{p}(y) \sum_{s=1}^{N} p_\lambda(s \mid y) \log \frac{p_{\lambda'}(y,s)}{p_\lambda(y,s)}$$

The EM algorithm says to find the λ' maximizing $Q(\lambda' \mid \lambda)$—subject, in this case, to $\sum_i \lambda'_i = 1$. Applying the method of Lagrange multipliers,

$$\begin{aligned}
\frac{\partial}{\partial \lambda'_i}\left[Q(\lambda' \mid \lambda) - \alpha(\sum_i \lambda'_i - 1)\right] &= 0 \\
\sum_y \tilde{p}(y) p_\lambda(s = i \mid y) \frac{1}{p_{\lambda'}(y, s = i)} p_i(y) - \alpha &= 0 \\
\sum_y \tilde{p}(y) p_\lambda(s = i \mid y) \frac{1}{\lambda'_i} - \alpha &= 0
\end{aligned}$$

Now, to ease the notational burden, we introduce the shorthand

$$\begin{aligned}
C_i &\equiv \sum_y \tilde{p}(y) p_\lambda(s = i \mid y) \frac{1}{\lambda'_i} \\
&= \frac{1}{\lambda'_i} \sum_y \tilde{p}(y) \frac{\lambda_i p_i(y)}{\sum_i \lambda_i p_i(y)} \qquad (9)
\end{aligned}$$

Then, applying the normalization constraint, we have $\lambda'_i = \frac{C_i}{\sum_i C_i}$. Intuitively, C_i is the expected number of times the i'th model is used in generating the observed sample, given the current estimates for $\{\lambda_1, \lambda_2, \ldots \lambda_n\}$. In this case, the EM algorithm amounts to:

- Initialize λ ($\lambda_i = 1/N$ is a good choice, but $\lambda_i = 0$ isn't)
- Repeat until convergence:
  - *E-step*: compute $C_1, C_2, \ldots C_N$, given the current λ, using (9).
  - *M-step*: set $\lambda_i \leftarrow \frac{C_i}{\sum_i C_i}$

This is, once you think about it, quite an intuitive approach to the problem. Since we don't know the linear interpolation weights, we'll guess them, apply the interpolated model to the data, and see how much each individual model contributes to the overall prediction. Then we can update the weights to favor the models which had a better track record, and iterate. It's not difficult to imagine that someone might think up this algorithm without having the mathematical equipment (in the EM algorithm) to prove anything about it. In fact, at least two people did [Ha58] [Welch].

3 IIS algorithm

We now consider a different family of conditional distributions—feature-based exponential models. Unlike the EM algorithm, we now assume nothing is hidden; the task is the pure maximum likelihood problem of discovering the optimal values for the model parameters, given some empirical sample $\tilde{p}(x,y)$.

The form of the model is

$$p_\lambda(y \mid x) \equiv \frac{1}{Z_\lambda(x)} \exp\left(\sum_{i=1}^{n} \lambda_i f_i(x,y)\right) \tag{10}$$

where

- $f_i(x,y)$ is a binary-valued function—called a "feature"—of (x,y). Associated with the model p_λ is some finite collection of n such functions. How one might select these functions is not the topic of this note. There exist methods for automatically "learning" informative features from a large collection; see [De97].

- λ_i is a real-valued weight associated with f_i. Technically, λ_i is the Lagrange multiplier corresponding to the feature f_i in a certain constrained optimization problem. In this sense, the absolute value of λ_i is a measure of the "importance" of the feature f_i. We denote by λ the vector of weights: $\lambda \equiv \{\lambda_1, \lambda_2, \ldots \lambda_n\}$.

- $Z_\lambda(x)$ is a normalizing factor, required to make $p_\lambda(y \mid x)$ a probability distribution:

$$Z_\lambda(x) = \sum_y \exp\left(\sum_{i=1}^{n} \lambda_i f_i(x,y)\right) \tag{11}$$

The log likelihood of the exponential model (10) is

$$L(\tilde{p} \mid p_\lambda) = \sum_{x,y} \tilde{p}(x,y) \sum_i \lambda_i f_i(x,y) - \sum_x \tilde{p}(x) \log \sum_y \exp \sum_i \lambda_i f_i(x,y)$$

Differentiating with respect to an individual parameter λ_i, we get

$$\begin{aligned} \frac{\partial L(\tilde{p} \mid \lambda)}{\partial \lambda_i} &= \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_{x,y} \tilde{p}(x) p_\lambda(y \mid x) f_i(x,y) \\ &= <\tilde{f}_i> - <f_i> \end{aligned} \tag{12}$$

Here $<\tilde{f}_i>$ denotes the expectation of $f_i(x,y)$ with respect to the empirical distribution $\tilde{p}$, and $<f_i>$ the expectation of $f_i(x,y)$ with respect to the distribution $\tilde{p}(x)p_\lambda(y \mid x)$.

Setting (12) to zero yields the condition for an extremum of the log-likelihood with respect to the single parameter λ_i. And the resulting condition—that the empirical expected probability of the feature f_i be equal to the model expected probability—is a very natural condition.

When $f_i(x,y)$ is binary-valued, as is the case here, this condition has an especially intuitive interpretation: the expected fraction of events (x,y) for which f_i is "on" (non-zero) should be the same according to the empirical distribution $\tilde{p}(x,y)$ and the model distribution $\tilde{p}(x)p_\lambda(y \mid x)$.

Finding $\lambda^\star$

Say we have a model of the form (10) with some arbitrary set of parameters $\lambda \equiv \{\lambda_1, \lambda_2, \ldots \lambda_n\}$. We'd like to find a new set of parameters $\lambda + \delta \equiv \{\lambda_1 + \delta_1, \lambda_2 + \delta_2, \ldots \lambda_n + \delta_n\}$ which yield a model of higher log-likelihood. If we can find a procedure (a growth transformation) $\tau : \lambda \to \lambda + \delta$ which takes one set of parameters as input and produces a new set as output which is not inferior, we can apply the transformation τ until we reach its fixed point, a stationary point for λ.

With respect to a given empirical distribution $\tilde{p}(x, y)$, the change in log-likelihood from the model λ to the model $\lambda + \delta$ is

$$\begin{aligned} L(\tilde{p} \mid p_{\lambda+\delta}) - L(\tilde{p} \mid p_\lambda) &= \sum_{x,y} \tilde{p}(x,y) \log p_{\lambda'}(y \mid x) - \sum_{x,y} \tilde{p}(x,y) \log p_\lambda(y \mid x) \\ &= \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) - \sum_x \tilde{p}(x) \log \frac{Z_{\lambda'}(x)}{Z_\lambda(x)} \end{aligned}$$

We now make use of the inequality $-\log \alpha \geq 1 - \alpha$ (true for all $\alpha > 0$), to establish a lower bound on the change in likelihood:

$$\begin{aligned} & L(\tilde{p} \mid \lambda + \delta) - L(\tilde{p} \mid \lambda) \geq \\ & \qquad \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{p}(x) \frac{Z_{\lambda'}(x)}{Z_\lambda(x)} \\ &= \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{p}(x) \frac{\sum_y \exp \sum_i (\lambda_i + \delta_i) f_i(x,y)}{\sum_y \exp \sum_i \lambda_i f_i(x,y)} \\ &= \underbrace{\sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y \mid x) \exp \sum_i \delta_i f_i(x,y)}_{\text{Call this } \mathcal{A}(\delta \mid \lambda)} \end{aligned} \tag{13}$$

Since $L(\tilde{p} \mid \lambda + \delta) - L(\tilde{p} \mid \lambda) \geq \mathcal{A}(\delta \mid \lambda)$, we know that if we can find a δ for which $\mathcal{A}(\delta \mid \lambda) > 0$, then the model $\lambda + \delta$ is an improvement (in terms of log-likelihood) over the model λ. A greedy strategy for optimizing the parameters of a log-linear model of the form (10), then, is to find the δ which maximizes $\mathcal{A}(\delta \mid \lambda)$, set $\lambda \leftarrow \lambda + \delta$, and repeat. So long as $\mathcal{A}(\delta \mid \lambda) > 0$, we're guaranteed an improvement in likelihood by this technique.

The straightforward approach, then, would be to maximize $\mathcal{A}(\delta \mid \lambda)$ with respect to each δ_i. Unfortunately, this doesn't quite work: differentiating $\mathcal{A}(\delta \mid \lambda)$ with respect to δ_i yields an equation containing $\{\delta_1, \delta_2, \ldots \delta_n\}$; in other words, the contraint equations for the δ_i will be coupled. To get around this, we'll need the quantity

$$f^\#(x,y) \equiv \sum_i f_i(x,y)$$

If the f_i are binary-valued, $f^\#(x,y)$ has the simple interpretation of the number of features which "apply" (are non-zero) at x, y. We can rewrite $\mathcal{A}(\delta \mid \lambda)$ as

$$\mathcal{A}(\delta \mid \lambda) = \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y \mid x) \exp \left(f^\#(x,y) \sum_i \frac{\delta_i f_i(x,y)}{f^\#(x,y)} \right) \tag{14}$$

Notice that $\frac{f_i(x,y)}{f^\#(x,y)}$ is a p.d.f. over i, since it's always non-negative and sums to one over the natural numbers. This means we can apply Jensen's inequality—namely, for a p.d.f. $p(x)$,

$$\exp \sum_x p(x) q(x) \leq \sum_x p(x) \exp q(x)$$

to rewrite (14) as

$$\mathcal{A}(\delta \mid \lambda) \geq \underbrace{\sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\lambda(y \mid x) \sum_i \left(\frac{f_i(x,y)}{f^\#(x,y)} \right) e^{\delta_i f^\#(x,y)}}_{\text{Call this } \mathcal{B}(\delta \mid \lambda)} \tag{15}$$

$\mathcal{B}(\delta \mid \lambda)$ is a new, not as tight, lower bound on the change in log-likelihood. That is,

$$L(\tilde{p} \mid \lambda + \delta) - L(\tilde{p} \mid \lambda) \geq \mathcal{B}(\delta \mid \lambda)$$

Differentiating $\mathcal{B}(\delta \mid \lambda)$ with respect to δ_i gives

$$\frac{\partial \mathcal{B}(\delta)}{\partial \delta_i} = \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_x \tilde{p}(x) \sum_y p_\lambda(y \mid x) f_i(x,y) e^{\delta_i f^\#(x,y)} \tag{16}$$

What's nice about (16) is that δ_i appears alone, without any other free parameters. Thus we can solve for each of the n free parameters $\{\delta_1, \delta_2, \ldots \delta_n\}$ individually by differentiating $\mathcal{B}(\delta \mid \lambda)$ with respect to each δ_i in turn. This suggests an iterative algorithm for finding the optimal values of $\lambda_1, \lambda_2 \ldots \lambda_n$:

IIS Algorithm

- `Start with some (arbitrary) value for each` λ_i
- `Repeat until convergence:`
  - `Solve` $\frac{\partial \mathcal{B}(\delta)}{\partial \delta_i} = 0$ `in (16) for` δ_i
  - `Set` $\lambda_i \leftarrow \lambda_i + \delta_i$

References

[EM algorithm]

[Ba72] L. Baum. An inequality and associated maximization technique in statistical estimation of probabilistic functions of a Markov process (1972). *Inequalities*, **3**, 1–8.

[De77] A. Dempster, N. Laird, and D. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, **39(B)**, 1–38.

[Ha58] H. Hartley (1958). Maximum likelihood estimation from incomplete data. *Biometrics* **14**, 174–194.

[La93a] J. Lafferty (1993). A derivation of the inside-outside algorithm from the EM algorithm. *IBM Technical Report.*

[Welch] L. Welch, unpublished.

[**Parameter estimation for exponential models**]

[Br59] D. Brown (1959). A note on approximations to discrete probability distributions. *Information and Control*, **2**, 386–392.

[Cs84] I. Csiszar and G. Tusnady (1984). Information geometry and alternating minimization procedures. *Statistics and Decisions, Supplemental Issue:1*, 205–237.

[De97] S. Della Pietra, V. Della Pietra and J. Lafferty (1997) Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19(4)**.

[**Also of interest**]

[Ba83] L. Bahl, F. Jelinek and R. Mercer (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-5**, 179–190.

[Be96] A. Berger, S. Della Pietra and V. Della Pietra (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, **22(1)**, 39–71.

[Bl92] E. Black, F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, and S. Roukos, (1992) Towards history-based grammars: using richer models for probabilistic parsing. *Proceedings of the DARPA Speech and Natural Language Workshop*, Arden House, New York.

[Br91] P. Brown, R. Mercer, S. Della Pietra, V. Della Pietra (1991). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* 19(2), 263—311.

[Br87] P. Brown (1987). The acoustic modelling problem in automatic speech recognition. Ph.D. thesis, Carnegie Mellon University.

[Je80] F. Jelinek and R. Mercer (1980). Interpolated estimation of Markov source parameters from sparse data. In *Proceedings, Workshop on Pattern Recognition in Practice*, Amsterdam, The Netherlands.

[Ka87] S. Katz (1987). Estimation of probabilities from sparse data for the langauge model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, **ASSP-35(3)**, 400–401, March.

[Me90] B. Merialdo (1990). Tagging text with a probabilistic model. *Proceedings of the IBM Natural Language ITL*, Paris, France, 161–172.

[Po88] A. Poritz. Hidden markov models: a guided tour. *IEEE Conference on Acoustics, Speech and Signal Processing.*

These notes were informed by unpublished notes on the EM algorithm, forward-backward algorithm, and linear interpolation, written by Peter Brown, Robert Mercer, John Lafferty, and Harry Printz.