Technical Report TR-2004-08
University of British Columbia

Owed to a Martingale: A Fast Bayesian On-Line EM Algorithm for Multinomial Models

Eric Brochu • Nando de Freitas • Kejie Bao °

September 11, 2004

• Department of Computer Science, University of British Columbia
2366 Main Mall, Vancouver, BC Canada V6T 1Z4
{*ebrochu, nando*}*@cs.ubc.ca*

° Department of Computer Science, University of Toronto
10 King's College Road, Toronto, ON Canada M5S 3G4
{*kbao*}*@cs.toronto.edu*

Abstract

This paper introduces a fast Bayesian online expectation maximization (BOEM) algorithm for multinomial mixtures. Using some properties of the Dirichlet distribution, we derive expressions for adaptive learning rates that depend solely on the data and the prior's hyperparameters. As a result, we avoid the problem of having to tune the learning rates using heuristics. In the application to multinomial clustering, choosing the prior's hyperparameters is an easy task. Our experiments on large real data sets demonstrate that our Bayesian online learning algorithms are fast and provide accurate regularized solutions. We prove asymptotic convergence of our algorithms using stochastic approximation theory.

1 Introduction

Clustering and classifying massive discrete datasets is a ubiquitous problem in web applications. Typical datasets include documents with images, text and music (Brochu and de Freitas 2002, Brochu, de Freitas and Bao 2003), web-links visited by users when shopping (Sen and Hansen 2003) and logs of states and actions in on-line computer games (Lipson, Kueck, Brochu and de Freitas 2003). What makes this problem hard is that the number of parameters and data can be very large. The first issue can be dealt by adopting principled Bayesian priors and regularizers while the second requires the development fast of on-line learning algorithms.

A popular approach is to model the data with a finite mixture of multinomial distributions, whose parameters are estimated with the EM algorithm (McKendrick 1926, Hartley 1958, Baum, Petrie, Soules and Weiss 1970, Dempster, Laird and Rubin 1977). However, the EM algorithm requires that the entire data set be read at each iteration, and hence fails to scale well as the data set becomes large. To bypass this limitation, on-line EM algorithms which update the parameter estimates using only a single datum at a time have been proposed in statistics (Titterington 1984, Titterington, Smith and Makov 1985, Celeux and Diebolt 1992), machine learning (Sato and Ishii 1998, Sato 1999, Sato 2001) and computer vision (Petrovic, Jojic, Frey and Huang 2003, Jepson, Fleet and El-Maraghi 2003).

These on-line EM algorithms can be shown to be stochastic approximation algorithms of the Robins-Monro type (Bertsekas and Tsitsiklis 1996, Kushner and Yin 1997). This connection allows us to prove asymptotic convergence of the algorithms under some assumptions on the learning rates. It is well known, that learning rates of the form $a/(t+b)$, where t denotes the iteration index and a and b are user chosen parameters, ensure asymptotic convergence. *The important open problem is choosing these parameters in non-asymptotic real life scenarios. That is, what values are guaranteed to give us acceptable results in a finite number of iterations?*

Here we propose a Bayesian solution to this problem. Our work builds on a much earlier treatment that appeared under the name of *quasi-Bayes* in (Smith and Makov 1978). In this publication, the authors dealt with the problem of mixtures with known mixture parameters. Here, we extend the method to parameterized mixture components. In doing so, we show that it is possible to obtain analytical expressions for the learning rates that depend only on the parameters of the Dirichlet priors. As a result, the choice of prior in our models automatically leads to a choice of the parameters of the on-line learning algorithms. Our experiments on large real data sets demonstrate

that our Bayesian on-line learning algorithms are fast and provide accurate regularized solutions. Finally, we prove convergence of our algorithms using stochastic approximation tools.

2 Probabilistic Model and Batch EM

Our data is a set of discrete observations $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in_a})$, for $i = 1, 2, \ldots, T$. That is, x_{ia} is the fequency of occurrence of the discrete state a at time i. In text mining, we can think of x_{ia} as the number of times word a appears in document i. The model also applies when x_i is a matrix and we are interested in clustering Markov sequences, such as music or weblogs. We adopt the following multinomial mixture model to cluster the data:

$$\mathbf{x}_i|(\boldsymbol{\lambda}, \boldsymbol{\varphi}) \overset{iid}{\sim} \sum_{c=1}^{n_c} \lambda_c \prod_{a=1}^{n_a} {\varphi_{a,c}}^{x_{i,a}}$$

Here n_c is the number of clusters, n_a is the number of discrete attributes, $\lambda_c = p(c)$ is the prior probability of each cluster and $\varphi_{a,c} = p(a|c)$ is the probability of item a in cluster c. The unknown parameters are $\boldsymbol{\lambda}$ and $\boldsymbol{\varphi}$.

We introduce the latent random variables $z_i \in \{1, \ldots, n_c\}$ to indicate that a particular document $\mathbf{x}_i$ belongs to a specific group c. These indicator variables are drawn from a multinomial distribution, $z_i \sim \mathcal{M}_{n_c}(1; \boldsymbol{\lambda})$, which admits the density

$$p(z_i|\boldsymbol{\lambda}) = \prod_{c=1}^{n_c} \lambda_c^{\mathbb{I}_c(z_i)}.$$

That is, $p(z_i = c) = \lambda_c$. The latent variables enable us to write the mixture model as a nested multinomial model:

$$p(\mathbf{x}_i, z_i|\boldsymbol{\lambda}, \boldsymbol{\varphi}) = \prod_{c=1}^{n_c} \left[\lambda_c \prod_{a=1}^{n_a} \varphi_{a,c}^{x_{i,a}} \right]^{\mathbb{I}_c(z_i)},$$

where $\mathbb{I}_c(z_i) = 1$ if $\mathbf{x}_i$ belongs to group c and $\mathbb{I}_c(z_i) = 0$ otherwise. The learning problem under consideration can be supervised (when all the indicator variables are known), semi-supervised or unsupervised (when none of the indicator variables are known).

The posterior distribution of the indicator variables follows easily:

$$\xi_c^{(i)} \triangleq p(z_i = c|\boldsymbol{\lambda}, \boldsymbol{\varphi}, \mathbf{x}_i) = \frac{p(\mathbf{x}_i, z_i = c|\boldsymbol{\lambda}, \boldsymbol{\varphi})}{p(\mathbf{x}_i|\boldsymbol{\lambda}, \boldsymbol{\varphi})} = \frac{\lambda_c \prod_{a=1}^{n_a} \boldsymbol{\varphi}_{a,c}^{x_{i,a}}}{\sum_{c'=1}^{n_c} \lambda_{c'} \prod_{a=1}^{n_a} \boldsymbol{\varphi}_{a,c'}^{x_{i,a}}}. \quad (1)$$

We place standard conjugate Dirichlet priors on the mixing coefficients $\boldsymbol{\lambda} \sim \mathcal{D}_{n_c}(\alpha)$, admitting the following density

$$p(\boldsymbol{\lambda}|\alpha) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)\cdots\Gamma(\alpha_{n_c})} \prod_{c=1}^{n_c} \lambda_c^{\alpha_c - 1} \mathbb{I}_{\{\sum_c \lambda_c = 1\}},$$

where $\Gamma(\cdot)$ denotes the Gamma function and $\alpha_0 = \sum_c \alpha_c$ is the sum of the hyper-parameters α. Similarly, we place independent Dirichlet priors on the component parameters $\boldsymbol{\varphi}_c \sim \mathcal{D}_{n_a}(\beta_c)$.

Using Bayes rule, the posterior distributions of the parameters are:

$$\boldsymbol{\lambda}|(\mathbf{z}, \mathbf{x}, \alpha) \sim \mathcal{D}_{n_c}\left(k_1 + \alpha_1, \ldots, k_{n_c} + \alpha_{n_c}\right) \quad (2)$$

$$\boldsymbol{\varphi}_c|(\mathbf{z}, \mathbf{x}, \beta) \sim \mathcal{D}_{n_a}\left(\sum_{i=1}^{T} x_{i,1}\mathbb{I}_c(z_i) + \beta_{1,c}, \ldots, \sum_{i=1}^{T} x_{i,n_a}\mathbb{I}_c(z_i) + \beta_{n_a,c}\right) \quad (3)$$

where where $k_c \triangleq \sum_{i=1}^{T} \mathbb{I}_c(z_i)$ denotes the total number of data samples assigned to class c. The posterior means of these distributions are

$$\begin{aligned} \lambda_c^{(PME)} &= \frac{\alpha_c + \sum_{i=1}^{T} \mathbb{I}_c(z_i)}{\alpha_0 + T} \\ \varphi_{a,c}^{(PME)} &= \frac{\beta_{a,c} + \sum_{i=1}^{T} x_{i,a}\mathbb{I}_c(z_i)}{\beta_{0,c} + \sum_{i=1}^{T}\sum_{a'=1}^{n_a} x_{i,a'}\mathbb{I}_c(z_i)}, \end{aligned}$$

while the posterior modes are:

$$\begin{aligned} \lambda_c^{(MAP)} &= \frac{\alpha_c - 1 + \sum_{i=1}^{T} \mathbb{I}_c(z_i)}{\alpha_0 - n_c + T} \\ \varphi_{a,c}^{(MAP)} &= \frac{\beta_{a,c} - 1 + \sum_{i=1}^{T} x_{i,a}\mathbb{I}_c(z_i)}{\beta_{0,c} - n_a + \sum_{i=1}^{T}\sum_{a'=1}^{n_a} x_{i,a'}\mathbb{I}_c(z_i)} \end{aligned}$$

The problem with these equations is that indicators $\mathbb{I}_c(z_i)$ are unknown in unsupervised learning. In the EM approach, one replaces these indicators with their expectations $\xi_c^{(i)} = p(z_i = c|\mathbf{x}_i, \boldsymbol{\lambda}, \boldsymbol{\varphi}) = \mathbb{E}_{p(z_i|\mathbf{x}_i,\boldsymbol{\lambda},\boldsymbol{\varphi})}(\mathbb{I}_c(z_i))$ and alternates between an expectation (E) step and a maximisation (M) step. In the E step one computes $p(z_i = c|\mathbf{x}_i, \widehat{\boldsymbol{\lambda}}, \widehat{\boldsymbol{\varphi}})$ using equation (1). In the M step, we update the parameters using the following MAP estimates:

$$\begin{aligned} \widehat{\lambda}_c &= \frac{\alpha_c - 1 + \sum_{i=1}^{T} \xi_c^{(i)}}{\alpha_0 - n_c + T} \\ \widehat{\varphi}_{a,c} &= \frac{\beta_{a,c} - 1 + \sum_{i=1}^{T} x_{i,a}\xi_c^{(i)}}{\beta_{0,c} - n_a + \sum_{i=1}^{T}\sum_{a'=1}^{n_a} x_{i,a'}\xi_c^{(i)}} \end{aligned}$$

Setting all the hyperparameters (α, β) to 1, we obtain the maximum likelihood estimates. In practice, we do not typically favour any cluster so α is set to 1. β is a regularisation parameter that controls the number of clusters (de Freitas and Barnard 2001). The higher the value of β, the fewer clusters. This happens because by increasing β in all clusters, we are adding pseudo-counts to the clusters. Since some clusters cannot explain these extra counts, their probability λ_c goes to zero. In text mining, all the betas are often set to 2, giving rise to an estimator known as Laplace smoothing. The important thing is that setting β is easy and enables us to introduce a priori knowledge about the model complexity in a natural way.

The problem with EM is that it requires summing over the entire dataset. To circumvent this when the datasets are extremely large, we introduce on-line EM algorithms in the following sections.

3 On-line EM Algorithms via Weighted Statistics

The batch EM updates require that we compute the sufficient statistics:

$$\begin{aligned} \langle 1 \rangle_c^{(1:T)} &\triangleq \sum_{i=1}^{T} \xi_c^{(i)} \\ \langle x_{i,a} \rangle_c^{(1:T)} &\triangleq \sum_{i=1}^{T} x_{i,a} \xi_c^{(i)} \end{aligned}$$

The idea of on-line EM is to replace these statistics with weighted statistics, so that we pay more attention to the most recent data (Titterington 1984, Sato 1999). That is, we compute the weighted statistics of $f(\mathbf{x})$ at time t, $\langle f(\mathbf{x}) \rangle_c^{(t)}$, as follows

$$\langle f(\mathbf{x}) \rangle_c^{(t)} \triangleq \eta^{(t)} \sum_{\tau=1}^{t} \left(\prod_{s=\tau+1}^{t} \zeta^{(s)} \right) f(\mathbf{x}^{(\tau)}) \xi_c^{(\tau)} \tag{4}$$

where $\mathbf{x}^{(\tau)}$ is the data at time τ, $\xi_c^{(\tau)} = p(z^{(\tau)} = c | x^{(\tau)}, \theta^{(\tau-1)})$ and the parameter $\zeta^{(t)} (0 \leq \zeta^{(t)} \leq 1)$ is a time-dependent forgetting factor, with boundary condition $\prod_{s=t+1}^{t} \zeta^{(s)} = 1$. The normalising factor is a learning rate:

$$\eta^{(t)} = \left(\sum_{\tau=1}^{t} \prod_{s=\tau+1}^{t} \zeta^{(s)} \right)^{-1} \tag{5}$$

The following classical proposition allows us to map the forgetting factor representation of equation (4) to a recursive (on-line) estimator of the sufficient statistics of $f(\mathbf{x})$.

Proposition 1 *(Sato and Ishii 1998) The forgetting factor representation of equation (4) is equivalent to the following on-line update*

$$\langle f(x)\rangle_c^{(t)} = \langle f(x)\rangle_c^{(t-1)} + \eta^{(t)}\left(f(x^{(t)})\xi_c^{(t)} - \langle f(x)\rangle_c^{(t-1)}\right) \tag{6}$$

and the learning rate is related to the forgetting factor as follows

$$\eta^{(t)} = \frac{\eta^{(t-1)}}{\zeta^{(t)} + \eta^{(t-1)}}$$

Proof: See Appendix A.

This on-line estimator only requires that we set the learning rate. There is no need to set the forgetting factor. The estimator can also be written as a binary decision process

$$\langle f(\mathbf{x})\rangle_c^{(t)} = \left(1-\eta^{(t)}\right)\langle f(\mathbf{x})\rangle_c^{(t-1)} + \eta^{(t)}\left(f(\mathbf{x}^{(t)})\xi_c^{(t)}\right)$$

The current estimate is a weighted sum of the previous estimate and the expectation of the current data point.

Using this recursion, the on-line EM for mixtures of multinomials at time t is:

- **E step**: Compute the distribution of the latent variables:

$$\xi_c^{(t)} = \frac{\lambda_c \prod_{a=1}^{n_a} \varphi_{a,c}^{x_a^{(t)}}}{\sum_{c'=1}^{n_c} \lambda'_c \prod_{a=1}^{n_a} \varphi_{a,c'}^{x_a^{(t)}}}$$

- **M step**: Update the estimates in terms of weighted-means:

$$\begin{aligned}
\langle 1\rangle_c^{(t)} &= \langle 1\rangle_c^{(t-1)} + \eta^{(t)}\left(\xi_c^{(t)} - \langle 1\rangle_c^{(t-1)}\right) \\
\langle x_a\rangle_c^{(t)} &= \langle x_a\rangle_c^{(t-1)} + \eta^{(t)}\left(x_a^{(t)}\xi_c^{(t)} - \langle x_a\rangle_c^{(t-1)}\right) \\
\widehat{\lambda}_c^{(t)} &= \langle 1\rangle_c^{(t)} \\
\widehat{\varphi}_{a,c}^{(t)} &= \frac{\langle x_a\rangle_c^{(t)}}{\sum_a \widehat{\varphi}_{a,c}^{(t)}}
\end{aligned}$$

In the remainder, we refer to this algorithm as Sato's on-line EM (SOEM).

The learning rates used in SOEM are often of the type $a/(bt+c)$ where the constants a, b and c are chosen by trial and error. In the following section, we show how one can use Bayesian theory to circumvent this time consuming task.

4 Bayesian On-Line EM

When we observe the first observation, the Dirichlet posteriors are:

$$\begin{aligned}\lambda|(x^{(1)},\alpha) &\sim \mathcal{D}_{n_c}\left(\alpha_1^{(0)}+\mathbb{I}_1(z^{(1)}),\cdots,\alpha_{n_c}^{(0)}+\mathbb{I}_{n_c}(z^{(1)})\right)\\ \varphi_c|(x^{(1)},\beta) &\sim \mathcal{D}_{n_a}\left(\beta_{1,c}^{(0)}+x_1^{(1)}\mathbb{I}_c(z^{(1)}),\cdots,\beta_{n_a,c}^{(0)}+x_{n_a}^{(1)}\mathbb{I}_c(z^{(1)})\right)\end{aligned}$$

Since the indicator variables are unknown, we replace them with their expectations as follows:

$$\begin{aligned}\lambda|(x^{(1)},\alpha) &\sim \mathcal{D}_{n_c}\left(\alpha_1^{(0)}+\xi_1^{(1)},\cdots,\alpha_{n_c}^{(0)}+\xi_{n_c}^{(1)}\right)\\ \varphi_c|(x^{(1)},\beta) &\sim \mathcal{D}_{n_a}\left(\beta_{1,c}^{(0)}+x_1^{(1)}\xi_c^{(1)},\cdots,\beta_{n_a,c}^{(0)}+x_{n_a}^{(1)}\xi_c^{(1)}\right)\end{aligned}$$

These Dirichlet distributions are updated recursively. That is, $p(\lambda|x^{(t)},\alpha^{(t)})$ is Dirichlet with parameters $\alpha_c^{(t)}=\alpha_c^{(t-1)}+\xi_c^{(t)}$ and $p(\varphi_c|x^{(t)},\beta)$ is Dirichlet with parameters $\beta_{a,c}^{(t)}=\beta_{a,c}^{(t-1)}+x_a^{(t)}\xi_c^{(t)}$. The corresponding posterior mean estimates are:

$$\widehat{\lambda}_c^{(t)} = \frac{\alpha_c^{(t-1)}+\xi_c^{(t)}}{\alpha_0+\sum_{k=1}^{t}\sum_{c=1}^{n_c}\xi_c^{(k)}} = \frac{\alpha_c^{(t)}}{\alpha_0+t} \tag{7}$$

$$\widehat{\varphi}_{a,c}^{(t)} = \frac{\beta_{a,c}^{(t-1)}+x_a^{(t)}\xi_c^{(t)}}{\beta_{0,c}+\sum_{k=1}^{t}\sum_{a'=1}^{n_a}x_{a'}^{(k)}\xi_c^{(k)}} = \frac{\beta_{a,c}^{(t)}}{\beta_{0,c}+\sum_{k=1}^{t}\sum_{a'=1}^{n_a}x_{a'}^{(k)}\xi_c^{(k)}} \tag{8}$$

where $\alpha_0=\alpha_1^{(0)}+\cdots+\alpha_{n_c}^{(0)}$ and $\beta_{0,c}=\beta_{1,c}^{(0)}+\cdots+\beta_{n_a,c}^{(0)}$. At time $t-1$, we have

$$\widehat{\lambda}_c^{(t-1)} = \frac{\alpha_c^{(t-1)}}{\alpha_0+t-1}$$

Substituting this expression into Equation (7) gives us the following update equation for λ:

$$\widehat{\lambda}_c^{(t)} = \widehat{\lambda}_c^{(t-1)} + r_t\left(\xi_c^{(t)}-\widehat{\lambda}_c^{(t-1)}\right) \tag{9}$$

where the learning rate is:

$$r_t = \frac{1}{\alpha_0 + t} \tag{10}$$

This recurrence is the same as in SOEM, but in this case the learning rate is readily available in terms of the hyperparameters of the Dirichlet prior.

To shorten the notation, we define the quantities $h_c^{(k)} \triangleq \sum_{a'=1}^{n_a} x_{a'}^{(k)} \xi_c^{(k)}$ and $g_c^{(t)} \triangleq (\beta_{0,c} + \sum_{k=1}^{t} h_c^{(k)})^{-1}$, then Equation (8) can be written as follows at times t and $t-1$:

$$\begin{aligned} \widehat{\varphi}_{a,c}^{(t)} &= g_c^{(t)} \left(\beta_{a,c}^{(t-1)} + x_a^{(t)} \xi_c^{(t)} \right) \\ \widehat{\varphi}_{a,c}^{(t-1)} &= g_c^{(t-1)} \left(\beta_{a,c}^{(t-2)} + x_a^{(t-1)} \xi_c^{(t-1)} \right) \end{aligned}$$

Combining these equations, we obtain a recursive estimator for φ

$$\begin{aligned} \widehat{\varphi}_{a,c}^{(t)} &= \frac{g_c^{(t)}}{g_c^{(t-1)}} \widehat{\varphi}_{a,c}^{(t-1)} + g_c^{(t)} x_a^{(t)} \xi_c^{(t)} \\ &= \widehat{\varphi}_{a,c}^{(t-1)} + g_c^{(t)} \left(x_a^{(t)} \xi_c^{(t)} - h_c^{(t)} \widehat{\varphi}_{a,c}^{(t-1)} \right) \end{aligned}$$

Let the adaptive learning rate be

$$\eta_c^{(t)} = \frac{h_c^{(t)}}{\beta_{0,c} + \sum_{k=1}^{t} h_c^{(k)}} \tag{11}$$

Then,

$$\widehat{\varphi}_{a,c}^{(t)} = \widehat{\varphi}_{a,c}^{(t-1)} + \eta_c^{(t)} \left(\frac{x_a^{(t)} \xi_c^{(t)}}{\sum_{a'} x_{a'}^{(t)} \xi_c^{(t)}} - \widehat{\varphi}_{a,c}^{(t-1)} \right) \tag{12}$$

The Bayesian online EM (BOEM) algorithm has the same E step as SOEM, but the M step is given by equations (9) and (12). The learning rates for both recursions are automatically given in terms of the Dirichlet priors. The learning rate for φ also has the nice property that it adapts with the data.

5 Asymptotic Convergence of BOEM

In this section, we show that the BOEM parameter estimates converge with probability 1 to the batch EM estimates with an infinite number of data. Specifically, we show that $\widehat{\lambda}_c \xrightarrow{a.s.} \mathbb{E}[\xi]$ and $\widehat{\varphi}_{a,c} \xrightarrow{a.s.} \mathbb{E}\left[x_a \xi_c / h_c\right]$, where the

expectations are taken with respect to the data $\mathbf{x}$ and ξ is a function evaluated at the optimal parameters, $\xi = \xi(\mathbf{x}, \lambda^\star, \varphi^\star)$.

The proof follows from standard results in stochastic approximation theory (Bertsekas and Tsitsiklis 1996, Chapter 4). First, we show that the learning rates r and η satisfy some conditions necessary for convergence.

Lemma 1 *Let $\epsilon < h_c^{(t)} < B$ for a small constant ϵ and some constant B.* [1] *Then the learning rates $r^{(t)}$ and $\eta_c^{(t)}$, satisfy the following Properties:*

1. $\lim_{t\to\infty} r^{(t)} = 0$ *and* $\lim_{t\to\infty} \eta_c^{(t)} = 0$
2. $\sum_{t=1}^{\infty} r^{(t)} = \infty$ *and* $\sum_{t=1}^{\infty} \eta_c^{(t)} = \infty$
3. $\sum_{t=1}^{\infty} (r^{(t)})^2 < \infty$ *and* $\sum_{t=1}^{\infty} (\eta_c^{(t)})^2 < \infty$

Proof: See Appendix B.

An intuitive discussion on why these conditions on the learning rates are required is provided in (Bertsekas and Tsitsiklis 1996, Section 4.1).

It is important to note for the remainder of the proof that, by definition, $\widehat{\lambda}_c$, $\widehat{\varphi}_{a,c}$, ξ_c and $x_a \xi_c / h_c$ are upperbounded by 1 because the data consists of finite counts and all quantities are appropriately normalized in the algorithm presented in the previous section. Hence, there is no need for projection operators.

The parameter update equations in BOEM can be interpreted as small stepwise iterations of the fixed point $\lambda_c^\star = \mathbb{E}[\xi_c]$ and $\varphi_{a,c}^\star = \mathbb{E}\left[x_a \xi_c / h_c\right]$. For notational brevity we summarise both fixed point equations by introducing the quantities $\theta \triangleq \{\xi, \varphi\}$, $\gamma = \{r, \eta\}$ and $\phi \triangleq \{\xi_c, x_a \xi_c / h_c\}$. Hence, for the remainder of this section, we only concentrate on the single fixed point equation:

$$\begin{aligned} \theta^\star &= \mathbb{E}[\phi] \\ &= (1-\gamma)\theta^\star + \gamma \mathbb{E}[\phi] \end{aligned}$$

where again we emphasize that $\mathbb{E}[\phi]$ is a function of the optimal parmaters $\theta^\star$. Since we only gather one observation at each time step, the expectation in the fixed point iteration is approximated by a single sample $\phi^{(t)} =$

[1] The upperbound is trivially satisfied as h is given by finite counts, while the lowerbound is simply a requirement that there should be no empty clusters. This assumption is not overly restrictive in practice as we can enforce a small lowerbound ϵ at the level of machine precision or, after sufficient steps, we can eliminate empty clusters by pruning.

$\phi(\mathbf{x}^{(t)}, \theta^{(t)})$. (We could use more samples in a Monte Carlo fashion, but this is unnecessary.) The fixed point equation leads to the following update equation for the parameters:

$$\begin{aligned} \theta^{(t+1)} &= \left(1 - \gamma^{(t)}\right) \theta^{(t)} + \gamma^{(t)} \phi^{(t+1)} \\ &= \theta^{(t)} + \gamma^{(t)} \left(\phi^{(t+1)} - \theta^{(t)}\right) \end{aligned} \tag{13}$$

This update can be re-written as a *Robbins-Monro stochastic approximation*

$$\theta^{(t+1)} = \left(1 - \gamma^{(t)}\right) \theta^{(t)} + \gamma^{(t)} \mathbb{E}\left[\phi\right] + \gamma^{(t)} \left(\phi^{(t+1)} - \mathbb{E}\left[\phi\right]\right) \tag{14}$$

where $\mathbb{E}[\phi] - \theta$ is the mean field and $e^{(t+1)} \triangleq \phi^{(t+1)} - \mathbb{E}\left[\phi\right]$ is the stochastic approximation error.

The key to our proof is to introduce a Lyapunov potential function $f\left(\theta^{(t)}\right)$ that measures how far the current estimate is from the true expectation. We can choose any function that is unbounded away from the true expectation and whose minimum coincides with the true expectation. In our case, we choose the following quadratic Lyapunov function:

$$f\left(\theta^{(t)}\right) = \frac{1}{2} \left\| \theta^{(t)} - \mathbb{E}[\phi] \right\|^2$$

We will show that our algorithm descends on this quadratic function and hence $\theta^{(t)}$ converges to $\mathbb{E}[\phi]$. Since $\mathbb{E}[\phi]$ is a function of only the optimal parameters $\theta^\star$, the gradient of the Lyapunov function with respect to $\theta^{(t)}$ is $\nabla f(\theta^{(t)}) = \theta^{(t)} - \mathbb{E}[\phi]$. This expression for the gradient, allows us to rewrite Equation (14) as follows:

$$\begin{aligned} \theta^{(t+1)} &= \theta^{(t)} - \gamma^{(t)} \left(\nabla f\left(\theta^{(t)}\right) - e^{(t+1)}\right) \\ &= \theta^{(t)} + \gamma^{(t)} s^{(t+1)} \end{aligned}$$

where $s^{(t+1)}$ is the search direction. To descend on $f\left(\theta^{(t)}\right)$, we require

$$\nabla f\left(\theta^{(t)}\right) \mathbb{E}\left[s^{(t+1)} | \mathcal{F}^{(t)}\right] \leq 0$$

That is, $s^{(t+1)}$ must be a direction of gradient descent. Here, the history of the algorithm is described by the increasing family of Sigma-fields $\mathcal{F}^{(t)} \triangleq \{\theta^{(1)} \ldots \theta^{(t)}, \phi^{(1)} \ldots \phi^{(t)}, \gamma^{(1)} \ldots \gamma^{(t)}\}$. To avoid the problem of $s^{(t+1)}$ becoming orthogonal to $\nabla f\left(\theta^{(t)}\right)$, we require a stronger condition:

$$c \nabla f\left(\theta^{(t)}\right) \mathbb{E}\left[s^{(t+1)} | \mathcal{F}^{(t)}\right] \leq - \left\| \nabla f\left(\theta^{(t)}\right) \right\|^2 \qquad \forall t$$

where c is a positive constant. The following lemma establishes these conditions for BOEM.

Lemma 2 *There exist positive constants c, k_1 and k_2 such that the BOEM algorithm satisfies the following conditions:*

1. $\forall t$, $c\nabla f\left(\theta^{(t)}\right) \cdot \mathbb{E}\left[s^{(t+1)}|\mathcal{F}^{(t)}\right] \leq -\left\|\nabla f\left(\theta^{(t)}\right)\right\|^2$
2. $\forall t$, $\mathbb{E}\left[\left\|s^{(t+1)}\right\|^2|\mathcal{F}^{(t)}\right] \leq k_1 + k_2\left\|\nabla f\left(\theta^{(t)}\right)\right\|^2$

Proof: *See Appendix C.*

We can now state our convergence result.

Proposition 2 *Under the conditions of Lemmas 1 and 2 and the Lipschitz condition $\|\nabla f(\theta) - \nabla f(\mathbb{E}(\phi))\| \leq L\|\theta - \mathbb{E}(\theta)\|$ for some constant L, we have with probability 1:*

1. *The sequence $f(\theta^{(t)})$ converges.*
2. $\lim_{t\to\infty} \nabla f(\theta^{(t)}) = 0$.
3. *Every limit point of $\theta^{(t)}$ is a stationary point of f. Hence, $\theta^{(t)}$ converges almost surely to $\mathbb{E}(\phi)$.*

The proof of this proposition follows from the super-martingale convergence theorem (Bertsekas and Tsitsiklis 1996, Section 4.2.3). The Lipschitz continuity condition on ∇f is satisfied if f is twice differentiable and the Hessian $\nabla^2 f$ is bounded over the search space.

It is interesting to note that $\mathbb{E}[\|\gamma^{(t)}e^{(t+1)}\|^2|\mathcal{F}^{(t)}] = (\gamma^{(t)})^2\mathbb{E}[\|e^{(t+1)}\|^2|\mathcal{F}^{(t)}]$ is upperbounded. Moreover, let

$$M^{(t)} \triangleq \sum_{k=1}^{t} \gamma^{(k)} e^{(k)}$$

We have

$$\begin{aligned}\mathbb{E}\left[M^{(t)}|\mathcal{F}^{(t-1)}\right] &= \mathbb{E}\left[\gamma^{(t)}e^{(t)}\middle|\mathcal{F}^{(t-1)}\right] + \mathbb{E}\left[M^{(t-1)}|\mathcal{F}^{(t-1)}\right] \\ &= 0 + M^{(t-1)}\end{aligned}$$

Hence $M^{(t)}$ is a martingale with bounded second moments, and by the martingale convergence theorem it converges to a limit point. It follows

that the term $\gamma^{(t)}e^{(t)}$ vanishes to zero with probability 1. Consider our stochastic approximation

$$\theta^{(t+1)} = \theta^{(t)} + \gamma^{(t)}\left(\theta^{(t)} - \mathbb{E}[\phi]\right).$$

In the limit of $\gamma^{(t)}$ going to zero, the stability of the BOEM algorithm is governed by the following differential equation

$$\frac{d\theta}{dt} = \theta - \mathbb{E}[\phi]$$

6 Experiments: BOEM in the Wild

While we have established powerful theoretic capabilities of BOEM in Section 5, if we are to expect BOEM to be used in the real world, it is also necessary to show that there are, indeed, real-world situations in which it should be used.

The following experiments show the performance of batch EM, Sato's online EM (SOEM) and BOEM. All code was implemented in C++ (an updated GPL C++ implementation of batch EM, SOEM, and BOEM is available from the author's web site, *http://www.cs.ubc.ca/∼ebrochu.*) using the high-performance UBLAS library for linear algebra operations. Experiments were conducted on an Intel 2.66 GHz Pentium 4.

6.1 Experiments on Synthetic Cases

Synthetic data sets often prove useful as a controlled environment in which data can easily be manipulated to expose the utility of an algorithm. Our first experiments involve such data.

Our synthetic data sets are simulations of i.i.d. documents. Let n_c be the number of clusters, in the underlying n_a-dimensional Multinomial mixture, where each entry of the matrix $\varphi_{a,c}$ is drawn from a Uniform distribution $\mathcal{U}(0,1)$ and normalized by $\sum_a \varphi_{a,c}$. The mixture weights λ are determined similarly, by sampling each from $\mathcal{U}(0,1)$ and normalizing over $\sum_c \lambda_c$. The documents can then be generated by sampling a generating distribution c from λ and then sampling n 'words' from the Multinomial distribution φ_c.

6.1.1 Log-likelihood comparison

The synthetic data set consists of 10000 data from 5 clusters of 1000 dimensions. The number of 'words' in each datum is normally distributed with a

mean of 5000 and variance of 2500 (documents of fewer than 100 words are discarded and new documents generated in their place).

There is no automatic means to select the learning rate parameters κ and t_0 for SOEM, but after several trials, we found $\kappa = 0.7$ and $t_0 = 5000$ gave the best performance on this data set. Because the goal of this experiment was to maximize the log-likelihood, rather than to regularize or find the underlying distributions, the Bayesian hyperparameters of BOEM were set to $\alpha = 1, \beta = 1$, which eliminates the prior – that is, it causes our MAP model to collapse to the ML case.

50 trials were conducted for each algorithm on the same data set, with randomly-chosen initialization parameters for φ. The log-likelihood was computed at regular intervals. As Figure 1 shows, BOEM and SOEM have similar rates of convergence, and both stabilize after a few thousand data, but BOEM finds, on average, significantly superior log-likelihoods.

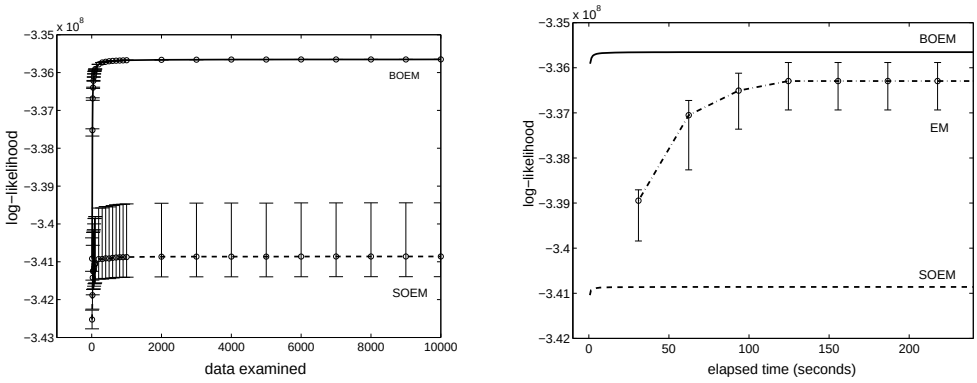

Figure 1: *[Left] Log-likelihood of the ML version of BOEM (solid) and SOEM (broken) on a synthetic data set of 10000 documents. BOEM and SOEM arrive at good approximations quickly, but BOEM does significantly better on average, and it does not require setting tricky forgetting factor parameters.[Right] Comparison of SOEM and BOEM to batch EM on the same problem.*

SOEM makes rapid estimates of φ during the very early time steps, which then are slowly improved on. In BOEM, the learning rate starts out slower and changes more slowly. SOEM is therefore more sensitive to the first few data examined, and if these give a poor clustering, it may be very difficult for the algorithm to escape even shallow local minima (often prevalent in high-dimensional data) that BOEM, by nature of its more even steps, is able to quickly escape. We also tried batch EM on this problem and report the results in Figure 1. Clearly, batch EM is not as well suited as BOEM

to analysis of data sets of even this modest size. In general, the differences between batch and online are only magnified as the data set grows in size.

6.1.2 Bayesian performance: BOEM vs batch

In a second experiment, we assessed the regularisation performance of the MAP EM and BOEM algorithms. we repeated the synthetic experiment but on this occasion generated the data from two mixture components of weights $\{0.8, 0.2\}$. The algorithms were then trained with $n_c = 6$ and $\beta = 5$. The attained log-posterior values are shown in Figure 2. Here BOEM outperforms the batch MAP EM algorithm again. Figure 2 also shows that BOEM computes the right number clusters. The batch EM algorithm also estimates the right number of clusters and parameters when using the shrinkage regulariser.

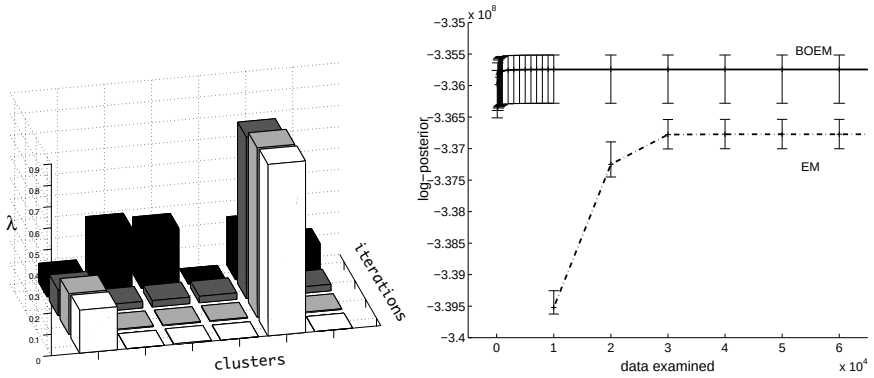

Figure 2: *[Left] λ evolution of a typical run of BOEM over synthetic data generated from two mixture components of weights $\{0.8, 0.2\}$. BOEM starts with 6 clusters, but eventually finds the right number. [Right] Performance using Bayesian regularization. Log-posterior over 50 trials of BOEM and Batch EM with $\alpha = 1$ and $\beta = 5$, on the set of 10000 synthetic data.*

6.2 Real World Data: The Gutenberg Corpus

To evaluate BOEM in a real world situation, we use data from Project Gutenberg. Project Gutenberg is an online repository of public domain texts, available at *http://www.gutenberg.net*. We randomly selected English-language 10000 documents from the corpus, consisting of novels and plays from various time periods and nationalities, political and philosophical writing, poetry, journalism and the like, to create a heterogeneous test set. We

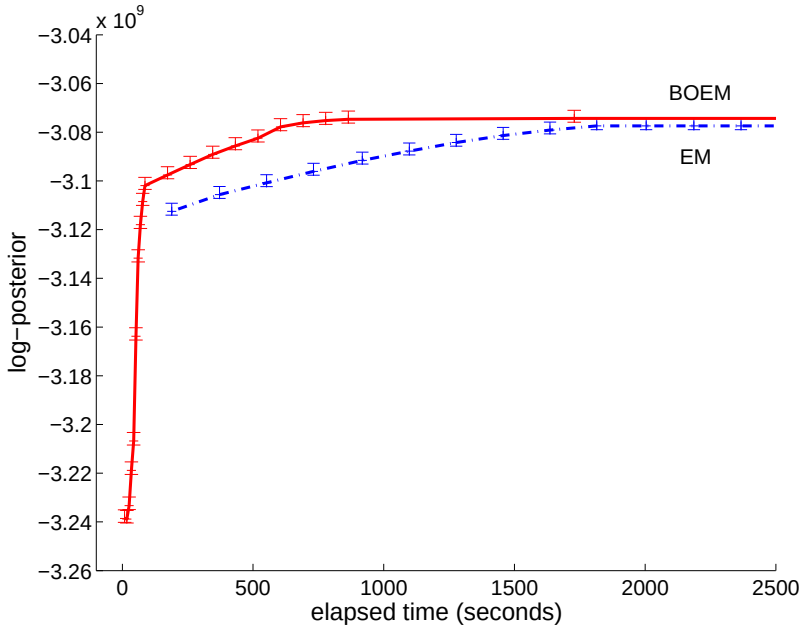

Figure 3: *Log-posterior of BOEM and batch EM on the Project Gutenberg data.*

removed any terms that appeared fewer than 5 times in the data set, or that appeared in only one document. This resulted in $n_a = 4322$ terms.

We ran batch EM and BOEM 50 times each on the entire data set of 10000 documents to compute the MAP estimates. The results may be seen in Figure 3. BOEM and batch eventually achieve similar results. BOEM performs slightly better, but it would seem its performance advantage is limited on the noisier, higher-dimensional data set. However, BOEM achieves good results in a fraction of the time batch takes.

7 Conclusions and Further Work

We have shown that BOEM outperforms batch EM and other online EM variants when applied to mixtures of multinomials. BOEM is fast, leads to a dramatic reduction in storage and typically results in better MAP estimates. Future work will involve extending the algorithm to LDA models (Blei, Ng and Jordan 2002) and sparse classifiers (Tipping 2001).

Acknowledgements

We are very indebted to Christophe Andrieu for substantial comments on the manuscript and convergence proof. We would also like to thank Mike Klaas for his corrections.

Appendix A: Proof of Proposition 1

The proof begins with the forgetting factor representation

$$\begin{aligned}
\langle f(\mathbf{x})\rangle_c^{(t)} &= \eta^{(t)} \sum_{\tau=1}^{t} \left(\prod_{s=\tau+1}^{t} \zeta^{(s)} \right) f(\mathbf{x}^{(\tau)}) \xi_c^{(\tau)} \\
&= \eta^{(t)} \zeta^{(t)} \sum_{\tau=1}^{t} \left(\prod_{s=\tau+1}^{t-1} \zeta^{(s)} \right) f(\mathbf{x}^{(\tau)}) \xi_c^{(\tau)} \\
&= \eta^{(t)} \zeta^{(t)} \sum_{\tau=1}^{t-1} \left(\prod_{s=\tau+1}^{t-1} \zeta^{(s)} \right) f(\mathbf{x}^{(\tau)}) \xi_c^{(\tau)} + \eta^{(t)} f(\mathbf{x}^{(t)}) \xi_c^{(\tau)}
\end{aligned}$$

Since

$$\langle f(\mathbf{x})\rangle_c^{(t-1)} = \eta^{(t-1)} \sum_{\tau=1}^{t-1} \left(\prod_{s=\tau+1}^{t-1} \zeta^{(s)} \right) f(\mathbf{x}^{(\tau)}) \xi_c^{(\tau)}$$

we have:

$$\begin{aligned}
\langle f(\mathbf{x})\rangle_c^{(t)} &= \frac{\eta^{(t)}}{\eta^{(t-1)}} \zeta^{(t)} \langle f(\mathbf{x})\rangle_c^{(t-1)} + \eta^{(t)} f(\mathbf{x}^{(t)}) \xi_c^{(t)} \\
&= \langle f(\mathbf{x})\rangle_c^{(t-1)} + \eta^{(t)} \left\{ f(\mathbf{x}^{(t)}) \xi_c^{(t)} + \left(\frac{\zeta^{(t)}}{\eta^{(t-1)}} - \frac{1}{\eta^{(t)}} \right) \langle f(\mathbf{x})\rangle_c^{(t-1)} \right\}
\end{aligned} \tag{15}$$

From equation (5), we know that

$$\begin{aligned}
\eta^{(t)} &= \left(\sum_{\tau=1}^{t} \prod_{s=\tau+1}^{t} \zeta^{(s)} \right)^{-1} = \left(1 + \zeta^{(t)} \sum_{\tau=1}^{t-1} \prod_{s=\tau+1}^{t-1} \zeta^{(s)} \right)^{-1} \\
&= \frac{1}{\zeta^{(t)} \frac{1}{\eta^{(t-1)}} + 1} = \frac{\eta^{(t-1)}}{\zeta^{(t)} + \eta^{(t-1)}}
\end{aligned}$$

Thus we get:

$$\frac{1}{\eta^{(t)}} = 1 + \frac{\zeta^{(t)}}{\eta^{(t-1)}} \tag{16}$$

Combining equations (15) and (16), we have:

$$\langle f(\mathbf{x})\rangle_c^{(t)} = \langle f(\mathbf{x})\rangle_c^{(t-1)} + \eta^{(t)}\left(f(\mathbf{x}^{(t)})\xi_c^{(t)} - \langle f(\mathbf{x})\rangle_c^{(t-1)}\right)$$

□

Appendix B: Proof of Lemma 1

Since $r^{(t)} = \frac{1}{\alpha_0+t}$, $\lim_{t\to\infty} r^{(t)} = 0$. The series $\sum_{t=1}^{\infty} r^{(t)}$ is a diverging harmonic series and $\sum_{t=1}^{\infty}(r^{(t)})^2$ converges. Next, we focus on $\eta_c^{(t)}$. We have:

$$\eta_c^{(t)} = \frac{h_c^{(t)}}{\beta_{0,c} + \sum_{k=1}^{t} h_c^{(k)}}$$

Since $\epsilon < h_c^{(t)} < B$, we also have harmonic series for $\eta_c^{(t)}$, say $\eta_c^{(t)} = \frac{k_1}{\beta_{0,c}+k_2 t}$ where $\epsilon < k_1, k_2 < B$. Consequently the three properties are also satisfied.

□

Appendix C: Proof of Lemma 2

Proof: The noise term $e^{(t+1)} = \left(\phi^{(t+1)} - \mathbb{E}[\phi]\right)$ has zero mean and bounded variance. That is, it has moments $\mathbb{E}[e^{(t+1)}|\mathcal{F}^{(t)}] = 0$ and $\mathbb{E}[\|e^{(t+1)}\|^2|\mathcal{F}^{(t)}] \leq A + B\|\nabla f(\theta^{(t)})\|$ for some constants A and B. The second property is an immediate consequence of ξ and $x_a\xi/b$ being upperbounded. Condition 1, with $c = 1$ follows from:

$$\begin{aligned}\nabla f\left(\theta^{(t)}\right)\cdot\mathbb{E}\left[s^{(t+1)}|\mathcal{F}^{(t)}\right] &= \nabla f\left(\theta^{(t)}\right)\cdot\left(-\nabla f\left(\theta^{(t)}\right) + \mathbb{E}\left[e^{(t+1)}|\mathcal{F}^{(t)}\right]\right)\\ &= -\left\|\nabla f\left(\theta^{(t)}\right)\right\|^2\end{aligned}$$

since $\nabla f\left(\theta^{(t)}\right)$ is determined by $\mathcal{F}^{(t)}$ and can therefore be pulled out of the expectation. Condition 2 results from:

$$\begin{aligned}
\mathbb{E}\left[\left\|s^{(t+1)}\right\|^2 |\mathcal{F}^{(t)}\right] &= \left\|\nabla f\left(\theta^{(t)}\right)\right\|^2 - 2\nabla f\left(\theta^{(t)}\right) \cdot \mathbb{E}\left[e^{(t+1)}|\mathcal{F}^{(t)}\right] \\
&\quad + \mathbb{E}\left[\left\|e^{(t+1)}\right\|^2 |\mathcal{F}^{(t)}\right] \\
&\leq \left\|\nabla f\left(\theta^{(t)}\right)\right\|^2 + A + B\left\|\nabla f\left(\theta^{(t)}\right)\right\|^2 \\
&= k_1 + k_2\left\|\nabla f\left(\theta^{(t)}\right)\right\|^2
\end{aligned}$$

□

References

Baum, L. E., Petrie, T., Soules, G. and Weiss, N. (1970). A maximization technique occurring in statistical analysis of probabilistic functions of Markov chains, *Annals of Mathematical Statistics* **41**: 164–171.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*, Athena Scientific.

Blei, D. M., Ng, A. Y. and Jordan, M. I. (2002). Latent dirichlet allocation, *in* T. G. Dietterich, S. Becker and Z. Ghahramani (eds), *NIPS*, MIT Press, Cambridge, MA.

Brochu, E. and de Freitas, N. (2002). "Name That Song!": A Probabilistic Approach to Querying on Music and Text, *NIPS*, Vancouver, Canada.

Brochu, E., de Freitas, N. and Bao, K. (2003). The Sound of an Album Cover: Probabilistic Multimedia and IR, *AI-STATS*, Florida, USA.

Celeux, G. and Diebolt, J. (1992). A stochastic approximation type EM algorithm for the mixture problem, *Stochastics and stochastics reports* **41**: 127–146.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data using the EM algorithm, *Journal of the Royal Statistical Society Series B.*

de Freitas, N. and Barnard, K. (2001). Bayesian modelling of documents with images and text, TR 2001-15, Computer Science Department, UBC.

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

Jepson, A. D., Fleet, D. J. and El-Maraghi, T. (2003). Robust online appearance models for visual tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **25**: 1296–1311.

Kushner, H. J. and Yin, G. G. (1997). *Stochastic Approximation Algorithms and Applications*, Springer-Verlag.

Lipson, A., Kueck, H., Brochu, E. and de Freitas, N. (2003). Machine learning for computer games, *First International Digital Games Research Conference*.

McKendrick, A. G. (1926). Application of mathematics to medical problems, *Proceedings of the Edinburgh Mathematical Society* **44**: 98–130.

Petrovic, N., Jojic, N., Frey, B. J. and Huang, T. S. (2003). Real-time on-line learning of transformed hidden Markov models from video, *Artificial Intelligence and Statistics (AI-Stats 2003)*.

Sato, M. A. and Ishii, S. (1998). On-line EM algorithm for the normalized Gaussian network, *Neural Computation* **12**(2): 407–432.

Sato, M. A. (1999). Fast learning of on-line EM algorithm, *Technical report*, TR-H-281, ATR Human Information Processing Research Laboratories.

Sato, M. A. (2001). On-line model selection based on the variational Bayes, *Neural Computation* **13**(7): 1649–1681.

Sen, R. and Hansen, M. H. (2003). Predicting a Web user's next request based on log data, *Journal of Computational and Graphical Statistics*.

Smith, A. F. M. and Makov, U. E. (1978). A quasi-Bayes sequential procedure for mixtures, *Journal of the Royal Statistical Society, Series B* **40**(1): 106–112.

Tipping, M. E. (2001). Sparse Bayesian learning and the relevance vector machine, *Journal of Machine Learning Research* **1**: 211–244.

Titterington, D. M., Smith, A. F. M. and Makov, U. E. (1985). *Statistical Analysis of Finite Mixture Distributions*, John Wiley and Sons, San Diego.

Titterington, D. M. (1984). Recursive parameter estimation using incomplete data, *Journal of the Royal Statistical Society* **46**: 257–267.