Available online at www.sciencedirect.com
ScienceDirect
Computational Statistics and Data Analysis 52 (2008) 4147–4160
COMPUTATIONAL STATISTICS & DATA ANALYSIS
www.elsevier.com/locate/csda

Direct maximization of the likelihood of a hidden Markov model

Rolf Turner*

Department of Mathematics and Statistics, University of New Brunswick, Fredericton, Canada
Starpath Project, University of Auckland, New Zealand

Received 25 September 2006; received in revised form 23 January 2008; accepted 23 January 2008
Available online 21 February 2008

Abstract

Ever since the introduction of hidden Markov models by Baum and his co-workers, the method of choice for fitting such models has been maximum likelihood via the EM algorithm. In recent years it has been noticed that the gradient and Hessian of the log likelihood of hidden Markov and related models may be calculated in parallel with a filtering process by which the likelihood may be calculated. Various authors have used, or suggested the use of, this idea in order to maximize the likelihood directly, without using the EM algorithm.

In this paper we discuss an implementation of such an approach. We have found that a straightforward implementation of Newton's method sometimes works but is unreliable. A form of the Levenberg–Marquardt algorithm appears to provide excellent reliability. Two rather complex examples are given for applying this algorithm to the fitting of hidden Markov models. In the first a better than 6-fold increase in speed over the EM algorithm was achieved. The second example turned out to be problematic (somewhat interestingly) in that the maximum likelihood estimator appears to be inconsistent. Whatever its merit, this estimator is calculated much faster by Levenberg–Marquardt than by EM.

We also compared the Levenberg–Marquardt algorithm, applied to the first example, with a generic numerical maximization procedure. The Levenberg–Marquardt algorithm appeared to perform almost three times better than the generic procedure, even when analytic derivatives were provided, and 19 times better when they were not provided.

1. Introduction, notation and terminology

Hidden Markov models were introduced by Baum and various co-authors in a series of papers (Baum and Petrie, 1966; Baum and Egon, 1967; Baum et al., 1970), in the late sixties and early seventies. The likelihood of such models is subtle to compute; Baum and his co-workers developed a filtering method via a recursively calculated sequence of "forward probabilities".

They then introduced a procedure, dubbed the "Baum–Welch algorithm", for maximizing the likelihood. This algorithm is in effect one of the earliest instances of the EM algorithm. The associated ideas were placed in a completely general context, and the name "EM algorithm" was coined in the widely cited paper Dempster et al. (1977).

* Corresponding address: Starpath Project, University of Auckland, New Zealand. Tel.: +64 9 623 8899; fax: +64 9 623 8953.
E-mail address: r.turner@auckland.ac.nz.

0167-9473/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.csda.2008.01.029

Baum and his co-workers defined hidden Markov models in the context of observation sequences indexed by *discrete* time and corresponding sequences of states from a discrete state space. Since the initial work of Baum et al. the concept has been considerably generalized to encompass models involving both continuous time and continuous state spaces. See Chapter 1 of Cappé et al. (2005) for a description of these generalizations and of the relationship of hidden Markov models to "state space" models. In this paper we deal only with discrete time hidden Markov models having discrete state spaces. See the tutorial paper by Rabiner (1989) and the paper by Juang and Rabiner (1991) for more background on such hidden Markov models and the use of the EM algorithm to fit them.

A hidden Markov model in this restricted sense comprises one or more sequences of observations $y_1, y_2, \ldots, y_n$ corresponding to an underlying but unobserved sequence of states $s_1, s_2, \ldots, s_n$ of a Markov chain. The (assumed finite) state space of the Markov chain may without loss of generality be taken to be $\{1, 2, \ldots, K\}$ for some K. We denote the transition probability matrix of the chain by $P = [p_{ij}]$ and the initial state distribution by $\boldsymbol{\pi} = (\pi_1, \pi_2, \ldots, \pi_K)^{\mathsf{T}}$. We shall refer to the entries of P and $\boldsymbol{\pi}$ as the "Markovian parameters" of the model.

The distribution of each of the observations depends on the hidden underlying state of the chain. The probability (density) function of y_t given that $s_t = j$ is denoted by $f_j(y)$. The functions f_j will generally depend upon a vector of parameters $\boldsymbol{\phi}$ which we shall call the "model parameters". We denote the totality of all parameters (Markovian and model) by $\boldsymbol{\theta}$.

The "forward probabilities" referred to previously are defined by $\alpha_t(j) = P(y_1, \ldots, y_t, s_t = j)$. They can be calculated recursively by

$$\alpha_1(j) = \pi_j f_j(y_1); \qquad \alpha_t(j) = f_j(y_t) \sum_{i=1}^{K} \alpha_{t-1}(i) p_{ij} \quad \text{for } t > 1.$$

The likelihood of the observed data $y_1, \ldots, y_n$ can be expressed in terms of the forward probabilities as

$$L(y_1, \ldots, y_n) = \sum_{i=1}^{K} \alpha_n(i). \tag{1}$$

2. Fitting hidden Markov models

The EM algorithm has been the tool used, in the large majority of instances, to fit hidden Markov models ever since their introduction. An early exception to this rule is the paper by Zucchini and Guttorp (1991) which made use of a general numerical algorithm to maximize the likelihood. Another exception is the work of Altman and Petkau (2005). Campillo and Le Gland (1989) compare the EM algorithm with a direction maximization approach for a related continuous time model.

Collings and Rydén (1998) present a "gradient algorithm" for fitting hidden Markov models. This algorithm takes a single recursive step toward maximizing the likelihood with each new datum received, rather than iterating to convergence for a complete data set. (It is focused, as the title indicates, on processing data in an "on-line" context.) The authors develop their procedure only in the context of Gaussian distributed observations, although they assert that it can be generalized to arbitrary distributions.

Lystig and Hughes (2002), Khan (2002), and Cappé and Moulines (2005), all developed independently the recursive procedure for calculating derivatives of the log likelihood of a hidden Markov model. The latter reference expresses results in a framework encompassing both discrete and continuous state spaces. In addition Cappé et al. (2005) present two ways of deriving the gradient equations, one of which is the so-called sensitivity equations approach, which is effectively the approach used in the current paper. The second is based on a recursive realization of smoothing which may also be used to compute the observed information. Cappé et al. also compare the results obtained by the EM algorithm and a quasi-Newton algorithm which uses the gradient of the log-likelihood on a simple example.

Lystig and Hughes also observed that the availability of the gradient and Hessian makes it possible to attack the problem of maximizing the likelihood via a direct approach. They did not however implement such an approach.

The EM algorithm appears to work dependably in a broad selection of examples of fitting HMMs, perhaps surprisingly so considering the complexity of the likelihood being maximized. However it has the disadvantage of often being very slow to converge. This slowness seems to be due in part to the flatness of the likelihood surface near its maximum.

Another problem with using the EM algorithm is the fact that it provides only point estimates of the parameters with no obvious means of assessing the variability of these estimates. Louis (1982) developed a procedure for calculating the observed Fisher Information matrix in the EM algorithm context. This procedure involves expressing the observed Fisher Information matrix in the form $\mathcal{I} = \mathcal{I}_1 - \mathcal{I}_2$ where $\mathcal{I}_1$ may be interpreted as the "total information" and $\mathcal{I}_2$ as the "missing information".

The calculation of $\mathcal{I}_2$ however, poses difficulties in general. This calculation involves taking expectations (with respect to the hidden states) which involve "4th-order" probabilities of the form

$$P(s_t = i, s_{t+1} = j, s_r = k, s_{r+1} = \ell \mid y_1, \ldots, y_n)$$

where t and r are two unrelated times. Determining these probabilities appears to be an intractable problem. Oakes (1999) comments on this difficulty, and presents an alternative approach to calculating the observed information matrix.

3. The direct approach

In more detail, the insight obtained in Lystig and Hughes (2002), Khan (2002), and Cappé and Moulines (2005) is that by differentiating the expression (1) we find that the gradient is equal to

$$G = \frac{\partial \ln L}{\partial \boldsymbol{\theta}^{\mathsf{T}}} = \frac{1}{L} \sum_{i=1}^{K} \frac{\partial \alpha_n(i)}{\partial \boldsymbol{\theta}^{\mathsf{T}}} \tag{2}$$

and the Hessian is equal to

$$H = \frac{\partial^2 \ln L}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathsf{T}}} = \frac{1}{L} \sum_{i=1}^{K} \frac{\partial^2 \alpha_n(i)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathsf{T}}} - \frac{1}{L^2} \left(\sum_{i=1}^{K} \frac{\partial \alpha_n(i)}{\partial \boldsymbol{\theta}} \right) \left(\sum_{i=1}^{K} \frac{\partial \alpha_n(i)}{\partial \boldsymbol{\theta}^{\mathsf{T}}} \right). \tag{3}$$

To evaluate this expression we just need to be able to calculate

$$\frac{\partial \alpha_n(i)}{\partial \boldsymbol{\theta}} \quad \text{and} \quad \frac{\partial^2 \alpha_n(i)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathsf{T}}}.$$

These derivatives can be obtained recursively. The updating equations for them are obtained by (twice) differentiating the updating equation for the $\alpha_t(i)$. The calculations get mildly messy, but amount to nothing more than elementary calculus. The complete details may be found in Khan (2002) and in Lystig and Hughes (2002).

4. Normalizing constants

Recall (see Rabiner, 1989) that in updating the $\alpha_t(j)$ numerical issues come into play. To avoid "underflow" (or possibly overflow in the case of continuously distributed observations) the values of $\alpha_t(j)$ must be systematically rescaled. The most obvious way to effect the rescaling is to renormalize the $\alpha_t(j)$ to sum to 1 over j. The details of the rescaling are discussed in Rabiner (1989), and in the context of recursive differentiation of the $\alpha_t(j)$, in Khan (2002) and Lystig and Hughes (2002). Generalizations of these ideas, including the treatment of continuous state spaces, are to be found in Collings and Rydén (1998), Cappé and Moulines (2005) and Cappé et al. (2005). The following treatment is essentially that of Khan (2002).

The renormalized values of the $\alpha_t(j)$, denoted by $\alpha_t^*(j)$, are calculated by setting

$$\alpha_1'(j) = \alpha_1(j), \quad j = 1, \ldots, K$$

$$\alpha_t'(j) = f_j(y_t) \sum_{i=1}^{K} \alpha_{t-1}^*(i) p_{ij}, \quad j = 1, \ldots, K, t > 1$$

$$c_t = \sum_{i=1}^{K} \alpha_t'(i)$$

$$\alpha_t^*(j) = \alpha_t'(j)/c_t.$$

It is easily seen that $\alpha_t(j) = c_1 c_2 \dots c_t \alpha_t^*(j)$. Notice that the likelihood of the model and data is

$$L = \sum_{i=1}^{K} \alpha_n(i) = c_1 c_2 \dots c_n \sum_{i=1}^{K} \alpha_n^*(i) = c_1 c_2 \dots c_n.$$

For convenience set $C_t = c_1 c_2 \dots c_t$.

The recursions leading to expressions for the gradient and Hessian of the log-likelihood of the model must then be expressed in terms of the $\alpha_t^*(j)$. In so doing it is convenient to define

$$a_t(j) = \frac{1}{C_{t-1}} \frac{\partial \alpha_t(j)}{\partial \boldsymbol{\theta}^{\mathsf{T}}}$$
$$B_t(j) = \frac{1}{C_{t-1}} \frac{\partial^2 \alpha_t(j)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathsf{T}}}.$$

The recursion relations can then be expressed solely in terms of $\alpha_t^*(j)$, $a_t(j)$, and $B_t(j)$. Details of these recursions are given in Khan (2002) and Lystig and Hughes (2002), with generalizations to be found in Collings and Rydén (1998), Cappé and Moulines (2005) and Cappé et al. (2005). For completeness these recursions are explicitly stated, in terms of the notation used in this paper, in Appendix A.

The derivatives in which we are interested are then expressed as

$$\frac{\partial \alpha_n(i)}{\partial \boldsymbol{\theta}^{\mathsf{T}}} = C_{n-1} a_n(i) \quad \text{and} \quad \frac{\partial^2 \alpha_n(i)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathsf{T}}} = C_{n-1} B_n(i).$$

Plugging these into the expressions (2) and (3) for G and H we get

$$G = \frac{C_{n-1}}{L} \sum_{i=1}^{K} a_n(i) = \frac{1}{c_n} \sum_{i=1}^{K} a_n(i)$$

and

$$\begin{aligned} H &= \frac{C_{n-1}}{L} \sum_{i=1}^{K} B_n(i) + \frac{C_{n-1}^2}{L^2} \left(\sum_{i=1}^{K} a_n(i) \right) \left(\sum_{i=1}^{K} a_n^{\mathsf{T}}(i) \right) \\ &= \frac{1}{c_n} \sum_{i=1}^{K} B_n(i) + \frac{1}{c_n^2} \left(\sum_{i=1}^{K} a_n(i) \right) \left(\sum_{i=1}^{K} a_n^{\mathsf{T}}(i) \right). \end{aligned}$$

5. Handling the transition probability matrix

Since the entries of the transition probability matrix $P = [p_{ij}]$ sum to 1 along rows there is some redundancy in the "natural" specification of the Markovian parameters. When the EM algorithm is used this redundancy causes no problem; the M step for the EM algorithm leads (via Lagrange multipliers) to expressions for the p_{ij} which satisfy the row sum constraint. When using a direct approach to maximizing the likelihood we require a parameterization of the p_{ij} which imposes this constraint "smoothly".

A "logistic style" parameterization which satisfies this desideratum is:

$$p_{ij} = \begin{cases} \dfrac{e^{\rho_{ij}}}{1 + \Delta_i} & 1 \le j \le K - 1 \\ \dfrac{1}{1 + \Delta_i} & j = K \end{cases}$$

where $\Delta_i = \sum_{j=1}^{K-1} e^{\rho_{ij}}$.

In our recursion expressions we need the derivatives of the p_{ij} with respect to the parameters $\boldsymbol{\theta}$, effectively now with respect to the ρ_{ij}. These derivatives (of the p_{ij} in terms of the ρ_{ij}) are mildly intricate. However it is possible to quarantine the intricacy; the calculations need only be done once. In software terms a procedure can be written "once and for all" to do this part of the calculation. The procedure can then be used as a module that plugs into the "driver"

being used to fit the (arbitrary) hidden Markov model of current interest. Explicit expressions for the derivatives of the p_{ij} with respect to the ρ_{ij} are given in Appendix B.

6. Handling the initial state probability distribution

As noted by Cappé et al. (2005, page 361), there is usually no hope of estimating the initial state distribution consistently, "as there is only one random variable ...(that is not even observed!) drawn from this [distribution]". Cappé et al. point out that it is "typical" to assume that the initial state distribution is the stationary (unique steady state) distribution of the chain. Under this assumption the initial state distribution $\boldsymbol{\pi}$ is determined by

$$P^{\mathsf{T}}\boldsymbol{\pi} = \boldsymbol{\pi}. \tag{4}$$

It is interesting to note that if $\boldsymbol{\pi}$ is treated as a "free" variable, the (inconsistent) maximum likelihood estimate will necessarily be obtained by setting one of the π_i equal to unity and all the others to zero. The index i to be chosen is the one for which $L(y_1, \ldots, y_n \mid s_1 = i)$ is maximal.

The expression for $\alpha_1(i)$, i.e. $\pi_i f_i(y_1)$, has to be differentiated twice with respect to $\boldsymbol{\theta}$ in order to start the recursions for the derivatives of the $\alpha_n(i)$. This in turn requires that we obtain the derivatives of $\boldsymbol{\pi}$ with respect to $\boldsymbol{\theta}$. Under the assumption that $\boldsymbol{\pi}$ is the stationary distribution, we achieve our goal by differentiating (4) with respect to θ_i to get

$$\frac{\partial P^{\mathsf{T}}}{\partial \theta_i}\boldsymbol{\pi} + P^{\mathsf{T}}\frac{\partial \boldsymbol{\pi}}{\partial \theta_i} = \frac{\partial \boldsymbol{\pi}}{\partial \theta_i}.$$

Collecting terms gives

$$(I - P^{\mathsf{T}})\frac{\partial \boldsymbol{\pi}}{\partial \theta_i} = \frac{\partial P^{\mathsf{T}}}{\partial \theta_i}\boldsymbol{\pi}.$$

This is a singular system since the rows of $I - P^{\mathsf{T}}$ sum to 0, but by invoking the constraint $\mathbf{1}^{\mathsf{T}}\boldsymbol{\pi} = 1$ and differentiating $\mathbf{1}^{\mathsf{T}}P^{\mathsf{T}}\boldsymbol{\pi} = \mathbf{1}^{\mathsf{T}}\boldsymbol{\pi} = 1$ with respect to θ_i we find that

$$\mathbf{1}^{\mathsf{T}}\frac{\partial P^{\mathsf{T}}}{\partial \theta_i}\boldsymbol{\pi} = 0 \quad \text{as well.}$$

We thus can form A equal to $I - P^{\mathsf{T}}$ with its last row replaced by $\mathbf{1}^{\mathsf{T}}$ and v equal to $\frac{\partial P^{\mathsf{T}}}{\partial \theta_i}\boldsymbol{\pi}$ with its last entry replaced by 0. Then we can solve

$$A\frac{\partial \boldsymbol{\pi}}{\partial \theta_i} = v$$

(since A is non-singular) to obtain $\frac{\partial \boldsymbol{\pi}}{\partial \theta_i}$. Second derivatives are obtained by repeating the procedure; I omit the details.

Another issue arises when we assume that $\boldsymbol{\pi}$ is the stationary distribution and seek to maximize the likelihood via the EM algorithm. The M-step of this algorithm consists in maximizing an "auxiliary function" $Q(\theta, \theta')$ with respect to θ where θ' is the current estimate of the parameter vector. In the hidden Markov model context the auxiliary function has the form

$$Q(\theta, \theta') = \sum_{k=1}^{K} \gamma_1(k) \ln \pi_k + \sum_{t=1}^{n-1}\sum_{i=1}^{K}\sum_{j=1}^{K} \xi_t(i, j) \ln p_{ij} + \sum_{t=1}^{n}\sum_{k=1}^{K} \gamma_t(k) \ln f_k(y_t).$$

A problem arises when the model is fitted under the constraint given by (4); our experience is that maximizing Q subject to this constraint is effectively impossible, although Cappé et al. indicate (Cappé et al., 2005, page 370) that it can be done. We have also formed the impression that the usual practice (which we follow) is to ignore the initial term $\sum_{k=1}^{K} \gamma_1(k) \ln \pi_k$ on the grounds that it is asymptotically negligible (see for example Billingsley (1961, p. 4)). This practice gave rise to an initially mysterious apparent anomaly in the fitting of a model in one example. (See Section 9.1.)

7. The Levenberg–Marquardt algorithm

Having available the gradient and Hessian of the log-likelihood permits approaching the task of maximizing the likelihood by solving $G = 0$ for $\boldsymbol{\theta}$ via Newton's method. In practice however it appears that in a majority of cases the Hessian eventually becomes singular when Newton's method is applied and as a result the method breaks down. An attractive alternative to Newton's method, for difficult optimization problems, is the Levenberg–Marquardt algorithm. (See Thisted (1988, p. 207 ff.), Press et al. (1994, p. 678 ff.), and Kennedy and Gentle (1980, pp. 442 ff., 483 ff).)

The Levenberg–Marquardt algorithm is usually expressed in terms of non-linear least squares problems. However it applies to more general optimization problems; Kennedy and Gentle (1980) describe the Levenberg–Marquardt algorithm in these general terms. Basically this algorithm is a judicious combination of Newton's method and the method of steepest ascent/descent. In Newton's method the update of $\boldsymbol{\theta}$ is equal to

$$\boldsymbol{\theta} - H^{-1}G$$

where H and G are evaluated at the current value of $\boldsymbol{\theta}$. In the method of steepest ascent/descent the update of $\boldsymbol{\theta}$ is equal to

$$\boldsymbol{\theta} + \epsilon \times G$$

where ϵ is an appropriate step size (positive for steepest ascent, negative for steepest descent).

The Levenberg–Marquardt algorithm combines these ideas by making the update of $\boldsymbol{\theta}$ equal to

$$\boldsymbol{\theta} - (H + \tau I)^{-1}G \tag{5}$$

where τ is of the order of $1/\epsilon$. Note that as $\tau \to 0$ the algorithm tends to Newton's method, and as $\tau \to \infty$ the algorithm effectively becomes the method of steepest ascent/descent (with a small step size). Details of the Levenberg–Marquardt algorithm may be found in Press et al. (1994, p. 679). The essential idea is that after a step given by (5) is taken, the objective function is checked to see whether it has improved. If it has, τ is decreased (multiplied by, say $1/10$) and another step is taken. If the objective function has not improved then τ is increased (multiplied by, say, 10) and the same step is re-taken.

In the more common non-linear least squares formulation of the Levenberg–Marquardt algorithm the "method of scoring" is applied, rather than Newton's method. That is, the Hessian is replaced by its expected value which is the matrix of twice the sums of squares and cross-products of first derivatives of the non-linear function whose parameters are being optimized. This expected value is guaranteed to be positive definite. In the general optimization setting this option is not available. In the current setting, which involves maximization, it is necessary to make sure that the matrix used is *negative* definite. To arrange this, it suffices to employ the simple expedient of subtracting the largest eigenvalue of the Hessian, before subtracting τ, if this eigenvalue is indeed positive.

This of course requires that we calculate the eigenvalues of the Hessian at each step, which could be computationally expensive. Despite this expense, this implementation of the Levenberg–Marquardt algorithm seems to be affordable.

8. Stopping criteria

A number of different stopping criteria may be applied to both the EM algorithm and the Levenberg–Marquardt algorithm. These include criteria based on

1. the change in the value of the log-likelihood,
2. the sum of squares of the changes in the values of the parameter estimates, and
3. the maximum of the absolute values of the changes in the parameter estimates.

For the Levenberg–Marquardt algorithm it is also appropriate to consider a criterion based upon some measure of the "size" of the gradient vector.

In order to make comparisons between algorithms "as valid as possible" the stopping criterion used must be applicable in all cases and must be structured so as to apply in all cases in as similar a manner as possible. In order to be able to make a valid comparison with a "general purpose" optimization routine (see Section 10) we chose to use a

stopping criterion base on the change in the value of the log-likelihood. Explicitly set $\mathcal{L}^{(0)}$ equal to the current value of the log likelihood and $\mathcal{L}^{(1)}$ equal to the updated value and define

$$\mathcal{C} = \mathcal{L}^{(1)} - \mathcal{L}^{(0)}. \tag{6}$$

An algorithm is deemed to have converged if

$$\mathcal{C} < (\mathcal{C} + \epsilon) \times \epsilon \tag{7}$$

where ϵ is the stopping tolerance. (Note: This structure for the stopping criterion was based on the stopping criterion used in the R function `optim()` (R Development Core Team, 2005; see page 16). The default value of ϵ was taken to be the square root of the built-in R constant `.Machine$double.eps`. This constant is effectively the smallest positive value that the computer in use is capable of handling. Using its square root as a tolerance for "differing from 0" is standard practice in R programming. (See for example the documentation on the function `all.equal()` in the R language, R Development Core Team (2005).)

It is worth remarking that the software has been written to make use of three other stopping criteria. If we set $\boldsymbol{\theta}^{(0)}$ equal to the current value of the vector of parameter estimates and $\boldsymbol{\theta}^{(1)}$ equal to the updated value, and let G be the updated value of the gradient vector (in the case of the Levenberg–Marquardt algorithm) we may define

$$\mathcal{C} = \left(\sum_{i=1}^{d} (\theta_i^{(1)} - \theta_i^{(0)})^2 \right)^{1/2}$$
$$\text{or } \mathcal{C} = \max\{|\theta_i^{(1)} - \theta_i^{(0)}|, i = 1, \ldots, d\}$$
$$\text{or } \mathcal{C} = \max\{|G_i|, i = 1, \ldots, d\}$$

where d is the dimension of the parameter vector $\boldsymbol{\theta}$. We re-emphasize here that the last definition of $\mathcal{C}$ makes sense for the Levenberg–Marquardt algorithm but not for the EM algorithm.

9. Examples

I now discuss in some detail two examples of applying the Levenberg–Marquardt algorithm to problems of fitting hidden Markov models.

9.1. Example 1: The Sydney coliform count data

This data set (see Turner et al. (1998)) consists of faecal coliform counts observed between 1989 and 1992 in the coastal waters near Sydney, Australia. Observations were made at four depths and seven locations consisting of three sewage outfall locations and four control locations. A subset of these data was analyzed in Turner et al. (1998).

These data were assumed to conform to a two-state ("clean" and "dirty") hidden Markov model. The states were related to a phenomenon called the "thermocline". Corresponding to each location–depth combination ("cell" of the model) there were a number of observations which were massaged to conform to a sequence of 194 dates at a one week separation. Most of the observations in the resulting sequences – about 73% – were actually missing. Turner et al. assumed serial dependence, modelled by a hidden Markov chain, within cells. They assumed that there is *independence* between cells. The cell means were modelled by a generalized linear (log linear Poisson) model with first-order terms only:

$$y \sim \text{mean} + \text{locn} + \text{depth} + \text{state}.$$

The data were transformed, prior to analysis, to make them more like Poisson counts.

The key idea is that (by assumption) "state" is one of the predictors in the model, which enters in the same log linear manner as all of the other predictors. To take account of "state" each observation in the data set is replicated twice (in general K times), one for each state. In this model the probability function for an observation y is

$$f(y) = \frac{\mathrm{e}^{-\lambda}\lambda^{y}}{y!}$$

where

$$\lambda = \mathbb{E}(y) = \mathrm{e}^{x^{\mathsf{T}}\phi}$$

and where in turn x is the vector of predictors corresponding to the model cell in which y is observed, and ϕ is the vector of model parameters. To calculate the gradient and Hessian of the log-likelihood of the model, we need the first and second derivatives of $\ln f(y)$ with respect to $\boldsymbol{\theta}$.

The derivatives of $f(y)$ with respect to the Markovian parameters are obviously all 0, so we need only concern ourselves with derivatives with respect to ϕ.

The first derivative is

$$\frac{\partial \ln f(y)}{\partial \phi} = (y - \lambda)x$$

and the second is

$$\frac{\partial^2 \ln f(y)}{\partial \phi \partial \phi^{\mathsf{T}}} = -\lambda x x^{\mathsf{T}}.$$

The proposed hidden Markov model was fitted to the same subset of the Sydney coliform count data as was analyzed in Turner et al. (1998) using both direct maximization of the likelihood via the Levenberg–Marquardt algorithm and the EM algorithm. The stopping criterion used was that defined by Eqs. (6) and (7) using the default value of the numerical tolerance. (Although it is preferable to use a criterion defined in terms of the gradient vector for the Levenberg–Marquardt algorithm, the criterion used must be applicable to both algorithms in order to permit a fair comparison.) A rough heuristically appealing calculation (the details of which are fussy and of no great interest) was used to determine starting values.

The Levenberg–Marquardt procedure converged in 7 steps, the EM algorithm in 101. The estimated values agreed closely, although not exactly. An interesting phenomenon was revealed by the log-likelihood values. The final log-likelihood from the Levenberg–Marquardt algorithm was about 0.0758% smaller than that from the EM algorithm. This amount might be considered "appreciable" when thought of in terms of the stopping criterion tolerance value which is roughly 10^{-8}. Further investigation was undertaken: when the EM algorithm was re-started with the Levenberg–Marquardt estimates as starting values, the value of the log-likelihood *decreased* on the first EM step.

Such a decrease is "theoretically impossible" — there is however an explanation. It is the practice (discussed on Section 6) of ignoring the first term of the auxiliary function $Q(\theta, \theta')$ which gives rise to this anomaly. In the current example, the M-step taken at the re-start of the EM algorithm actually decreases the value of Q by about 0.02, whence a decrease in the log-likelihood is possible. The value of Q *without* the initial term, which is what is really being maximized by the M-step, increases (by about the same amount, i.e. 0.02).

The difference between the final estimates returned by the two procedures, and between the two final log-likelihoods, is statistically negligible, so the asymptotic negligibility of the first term of Q is borne out. However the fact remains that the EM algorithm (under the usual practice of ignoring the first term in the auxiliary function) does not quite maximize the likelihood. This is another argument in favour of using direct maximization methods.

9.2. Numerical experiments for example 1

To assess the relative speeds of the two algorithms I simulated 100 independent replicate data sets from the model which was fitted to the Sydney coliform count data. For each replicate, the model parameters were estimated using both algorithms, with the same stopping criterion as defined by Eqs. (6) and (7) the same tolerance (the square root of `.Machine$double.eps`), and the same starting values. The computation time for each algorithm, for each data set, was determined as the "user cpu time" as returned by the R (R Development Core Team, 2005) function `system.time()`. A histogram of the ratio of the computation times (EM over Levenberg–Marquardt) is shown in Fig. 1. This ratio had a population mean estimated to be in [6.49, 7.23] with 95% confidence. Although the histogram of the ratio exhibits no striking skewness, one might generally expect skewness in such data whence the median might be a more appropriate measure of location. A 95% confidence interval for the population median was calculated to be [6.25, 7.00].

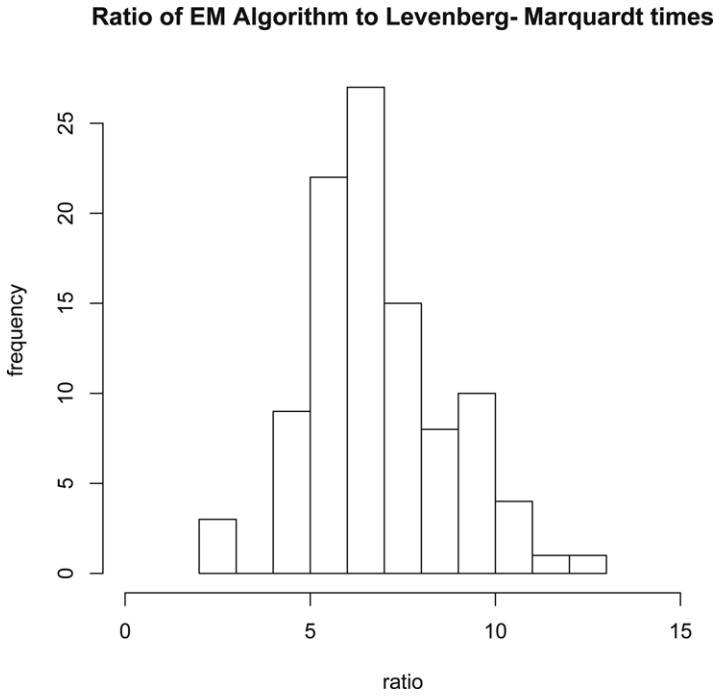

Fig. 1. Timing results for the Sydney coliform count example.

It is well known that maximum likelihood estimates of the parameters of a hidden Markov model may be very sensitive to the starting values used. As a quick check on whether there is a difference in this sensitivity between the EM algorithm and the Levenberg–Marquardt algorithm I refitted the model to the Sydney coliform count data using 20 random starting values chosen uniformly in the 10-dimensional "brick"

$$[0.05, 0.95]^2 \times [-0.6, 0.6]^7 \times [-3.5, -0.5]. \tag{8}$$

For all but one of these starting values both algorithms converged to the same parameter estimates as they did from the calculated starting values. In the one remaining case, both algorithms converged to an incorrect estimate with a substantially smaller log-likelihood.

9.3. Example 2: Multiple sclerosis lesion counts

Albert et al. (1994) studied sequences of counts, observed monthly and made by means of magnetic resonance imaging, of lesions in the brain and spinal cord of three multiple sclerosis patients. These data were further analyzed by Altman and Petkau (2005). A hidden Markov model (referred to by Altman and Petkau as the AMSF model) is formulated for these data by assuming that a given patient, at each observation time t is in a state of either "remission" or "relapse" where these are the states of a Markov chain. It is assumed that the transition probability matrix of the chain is

$$P = \begin{bmatrix} 1-\gamma & \gamma \\ \gamma & 1-\gamma \end{bmatrix}.$$

It is also assumed that the chain is stationary, i.e. in steady state when observation commences. This implies that the initial state probability distribution is $\boldsymbol{\pi} = (0.5, 0.5)^{\mathsf{T}}$, for any value of γ. The lesion count y_t for a given patient at time t is assumed to be Poisson distributed with mean equal to

$$\mu_t = \mu_0 \times \theta^{S_t} \quad \text{where } S_t = \sum_{j=1}^{t} Z_t \text{ where in turn } Z_t = \begin{cases} 1 & \text{if "relapse"} \\ -1 & \text{if "remission".} \end{cases} \tag{9}$$

Altman and Petkau also consider a few somewhat more complex models for these data sets, but for the purposes of this paper we restricted attention to the simpler model described above.

This model as it stands does not constitute a hidden Markov model in as much as the states "remission" and "relapse" do not determine the distribution of y_t. To overcome this difficulty Altman and Petkau defined a new Markov chain whose states consist of pairs (S_{t-1}, S_t) where S_t is as given in (9), with S_0 defined to equal 0. Note that S_t must be equal to $S_{t-1} \pm 1$. The transition probabilities of this Markov chain are

$$
\begin{aligned}
&P\{(S, S+1) \mapsto (S+1, S)\} = \gamma \\
&P\{(S, S+1) \mapsto (S+1, S+2)\} = 1-\gamma \\
&P\{(S, S-1) \mapsto (S-1, S)\} = \gamma \\
&P\{(S, S-1) \mapsto (S-1, S-2)\} = 1-\gamma.
\end{aligned}
$$

The pairs (S_{t-1}, S_t) may be (arbitrarily) mapped to the sequence of integers $1, 2, \ldots, K$ where $K = 4n-2$ and where n is the number of observations. A convenient mapping (for computer coding purposes) is

$$
\begin{aligned}
&(S, S+1) \mapsto 2n + 2S + 2 \\
&(S, S-1) \mapsto 2n + 2S - 1.
\end{aligned} \tag{10}
$$

The parameters of the model are $\mu_0 \geq 0$, $\theta \geq 1$ (for identifiability) and γ, $0 \leq \gamma \leq 1$. To avoid difficulties with the constraints that the parameters must satisfy, Altman and Petkau found it convenient to reparameterize the model in terms of

$$
\mu_0^* = \ln \mu, \qquad \theta^* = \ln(\theta - 1) \quad \text{and} \quad \gamma^* = \ln(\gamma/(1-\gamma))
$$

and we followed the same convention.

To implement the recursive procedure for calculating the gradient and Hessian of the log-likelihood of the model one needs the first and second partial derivatives, with respect to the (transformed) parameters of the model, of the transition probabilities and of the probability function for the observations. The derivatives of the transition probabilities are

$$
\begin{aligned}
&\frac{\partial p_{ij}}{\partial \gamma^{*2}} = \zeta \gamma (1-\gamma) \\
&\frac{\partial^2 p_{ij}}{\partial \gamma^{*2}} = \zeta \gamma (1-\gamma)(1-2\gamma)
\end{aligned}
$$

where $\zeta = 1$ if i and j are of opposite parity, and $\zeta = -1$ if i and j are of the same parity. Here i and j are the integer representations of the states, in the set $\{1, 2, \ldots, K = 4n-2\}$ under the mapping defined in (10). The foregoing assumes that a transition from i to j is possible, i.e. p_{ij} is either equal to γ or $1-\gamma$. Otherwise (i.e. when $p_{ij} = 0$) the derivatives are of course both equal to 0. The derivatives of the transition probabilities with respect to μ_0^* and θ^* are likewise all equal to 0.

The derivatives of the probability function are

$$
\begin{aligned}
&\frac{\partial f(y)}{\partial \mu_0^*} = (y-\lambda) f(y) \\
&\frac{\partial f(y)}{\partial \theta^*} = \frac{S_t(\theta-1)(y-\lambda) f(y)}{\theta} \\
&\frac{\partial^2 f(y)}{\partial \mu_0^{*2}} = ((y-\lambda) + (y-\lambda)^2 - y) f(y) \\
&\frac{\partial^2 f(y)}{\partial \mu_0^* \partial \theta^*} = \frac{S_t(\theta-1)((y-\lambda)^2 - \lambda) f(y)}{\theta} \\
&\frac{\partial^2 f(y)}{\partial \theta^{*2}} = \frac{S_t(\theta-1)(y-\lambda) f(y)}{\theta} + \frac{S_t(\theta-1)^2 (S_t(y-\lambda)^2 - (y-\lambda) - S_t\lambda) f(y)}{\theta^2}
\end{aligned}
$$

$$\text{where } \lambda = \mu_t = \mu_0 \theta^{S_t} \quad \text{and} \quad f(y) = \frac{e^{-\lambda}\lambda^y}{y!}.$$

The derivatives of $f(y)$ with respect to γ^* are obviously all equal to 0.

Maximizing the likelihood for this model via the EM algorithm involves certain intricacies in the M-step. To describe the procedure I need to remind the reader of some (fairly standard) notation used in the analysis of hidden Markov models. In this notation, as set out for example in Rabiner (1989), $\gamma_t(i)$ is used to represent the probability (given the observations, and calculated on the basis of the current parameter estimates) of the hidden chain's being in state i at time t.

Similarly $\xi_t(i, j)$ is used to represent the probability (given the observations, and calculated on the basis of the current parameter estimates) of a transition from state i at time t to state j at time $t + 1$. These quantities are readily calculated from the "forward and backward" probabilities $\alpha_t(i)$ and $\beta_t(i)$. See Rabiner (1989) for details. The array $\gamma_t(i)$ defined above is not to be confused with the scalar γ which determines the transition probabilities.

The M-step for γ is straightforward; the update is

$$\hat{\gamma} = \frac{\sum_{i-j=1 (\text{mod } 2)} \omega_{i,j}}{\sum_{ij} \omega_{ij}} \quad \text{where } \omega_{ij} = \sum_{t=1}^{n-1} \xi_t(i, j).$$

The M-step for μ_0 and θ can be accomplished by recognizing that the portion of the log-likelihood which involves μ_0 and θ has the form of a weighted Poisson log-likelihood with weights equal to the $\gamma_t(i)$ discussed above.

The three data sets which were analyzed by Albert et al. (1994) and Altman and Petkau (2005) were kindly made available to us by Professor Altman. For the purpose of experimenting with the Levenberg–Marquardt algorithm and comparing it with the EM algorithm, we focused on the data set for patient number 2. We used the same stopping criteria as in Example 1, i.e. as defined by Eqs. (6) and (7). Again a rough heuristically appealing procedure, the details of which we omit, was used to determine starting values. When the model was fitted by the EM algorithm, it converged in 16 steps. The Levenberg–Marquardt algorithm converged in 9 steps. The parameter estimates and the log-likelihoods were virtually identical. The transformed parameter estimates agreed (to 3 decimal places) with the estimates quoted in Altman and Petkau (2005, page 2340). The corresponding log-likelihood of the fit was −72.029.

9.4. Numerical experiments for example 2

As we did for Example 1, we simulated 100 replicate data sets from the model that was fitted to the real data. The same procedures were followed as for Example 1. Not surprisingly (given the small sample size) the results of this experiment revealed that the parameter estimation process seems not to be consistent in this context. (A Hotelling's T^2 test of the equality of the sample mean to the true mean yielded p-values of 9×10^{-5} for the EM algorithm and 3×10^{-4} in the case of the the Levenberg–Marquardt algorithm.)

The *asymptotic* consistency of maximum likelihood estimates was shown by Leroux (1992). (See also the related works in Bickel and Ritov (1996), Bickel et al. (1998), Bickel et al. (2002) and Jensen and Petersen (1999).) However the sample size here, i.e. 26, is probably much too small for the asymptotic result to hold. This problem may also be compounded by the fact that the maximization procedure sometimes, possibly frequently, converges to a spurious local maximum rather than to the global maximum. A further worry is that the somewhat unorthodox nature of the model (in that the size of the state space depends upon the sample size) may invalidate the consistency result.

Whatever the truth of the matter it would be wise to treat any parameter estimates for this model with substantial caution. There is a tendency amongst statistical practitioners to accept, somewhat unquestioningly, maximum likelihood estimates as being "valid". The current example illustrates the dangers in this tendency. It is advisable to undertake simulation studies to assess the consistency of estimation procedures in any setting which is even mildly unorthodox. We might add here that we did some informal experimentation to investigate whether consistency of estimates would appear to hold for longer data series. When the length was increased to the order of 100 the series exhibited a distressing tendency to "blow up" to enormous values – more than 50,000 – of lesion counts. This phenomenon may indicate an inadequacy of the AMSF model for these lesion counts.

In view of the apparent inconsistency of the estimator obtained, one might question the relevance of timing results for algorithms implementing that estimator. Nevertheless we claim that these results give insight into the relative merits of the algorithms, and therefore briefly summarize the results as follows:

The population mean ratio of the computation times (EM algorithm over the Levenberg–Marquardt procedure) was estimated to be in [28.58, 61.36] with 95% confidence. The population median ratio was likewise estimated to be in [19.62, 29.70]. (In this case the histogram of the data was indeed highly skewed to the right, making the median a more appropriate measure of central tendency.) The implication is that the Levenberg–Marquardt procedure is a great deal faster than the EM algorithm in calculating the estimates of the parameters for the AMSF model, whatever the merits of such a calculation might be.

As for Example 1 we experimented with random starting values for the algorithms. Twenty random starting values were chosen uniformly from the "brick"

$$[1.95, 7.8] \times [1.25, 2.25] \times [0.6, 1]. \tag{11}$$

In this example the algorithms appear to be somewhat more sensitive to starting values than in Example 1. In 10 of the 20 cases the EM algorithm converged to a point at which the log-likelihood was less than -72.029; in the other 10 cases the log-likelihood was equal to -72.029. The Levenberg–Marquardt algorithm got to a lower log-likelihood in 7 of the 20 cases and to -72.029 in the other 13 cases. In no case did a random starting value lead to an improved value of the likelihood.

10. Comparison with a general purpose optimizer

It has been suggested that one might do just as well as or better than the Levenberg–Marquardt algorithm by invoking a general purpose optimization routine. I investigated this suggestion in the case of first example of the two examples by implementing the maximization via the `optim()` function in R. I did this in two ways: (i) using only numerically calculated derivatives (i.e. providing the algorithm only the values of the log-likelihood function) and (ii) providing the algorithm with analytically calculated values of the gradient of the log-likelihood.

Ratios of the computation times, Levenberg–Marquardt over `optim()`, were obtained as in the numerical experiments for example 1 (described in Section 9.2). For procedure (i) the 95% confidence interval for the mean of these timing ratios was [19.70, 21.35] and that for the median was [19.99, 21.50]. For procedure (ii) the race was much closer. The confidence interval for the mean was [2.90, 3.14] and that for the median was [2.90, 3.15].

Ratios of computation times were also investigated in the context of simulation for Monte Carlo inference (see Section 11) where the starting values can be chosen to be the "true" values, i.e. those fitted to the real data set. In this setting the confidence interval for the median ratio (with analytic derivatives provided to `optim()`) was [5.00, 5.16]. Thus the customized procedure is about three times faster than a general purpose procedure, when "rough" starting values are used and five times faster when "true" values are used as starting values, even when analytically calculated derivatives are provided. Since it is the analytical calculation of the derivatives that forms the bulk of the work in setting up the maximization procedure, it would appear to be worthwhile to customize.

11. Discussion and conclusions

In recent years it has been realized that the derivatives of the log-likelihood of a hidden Markov model may be calculated analytically. This in turn implies the possibility of maximizing the likelihood of such a model by means of a direct "gradient–Hessian" approach, without resorting to the EM algorithm. The obvious approach of using Newton's method is insufficiently robust to be practical. In this paper we have discussed an effective implementation of the direct approach using an adaptation of the Levenberg–Marquardt algorithm. In the experiments so far conducted the Levenberg–Marquardt algorithm appears to converge at least as dependable as the EM algorithm.

The Levenberg–Marquardt algorithm appears to be much faster than the EM algorithm: more than 6 times faster in the context of one of the two relatively complicated models which we used to study the performance of these algorithms. In the context of the other model it appears to be more than 19 times faster, although the value of the estimator calculated is dubious for the latter model.

Application of the Levenberg–Marquardt algorithm has the useful byproduct of automatically providing an estimate of the covariance matrix of the parameter estimates, and thus estimates of their precision. It also appears that once the

parameter values are sufficiently close to the solution, the Levenberg–Marquardt algorithm converges very quickly. This gives it an advantage in simulation studies where the true parameter values can be used as starting values. (The advantage will be particularly great with respect to the EM algorithm, but a five-fold advantage was found – see Section 10 – over a general purpose optimizer.) Such studies are necessary when analytic distributional results are questionable and it is desired to conduct Monte Carlo inference via parametric bootstrapping.

Regardless of the technique used, maximum likelihood estimation of the parameters of a hidden Markov model is always delicate in that the procedure is sensitive to the starting values used. Spurious local maxima will usually exist, and these may trap any maximization technique. For the first of the examples explored in this paper, the Levenberg–Marquardt and EM algorithms appear to be reasonably (and equally) robust, converging to the correct answer in 19 out of 20 instances of randomly chosen (and thereby generally bad) starting values.

Acknowledgements

The research was supported by the Natural Sciences and Engineering Research Council of Canada. The author thanks two anonymous referees whose comments led to substantial improvements in this paper.

Appendix A

$$\text{Recursion for } a_t(j): \quad a_t(j) = f_j(y_t) \sum_{i=1}^{K} \left(\alpha^*_{t-1}(i) \frac{\partial p_{ij}}{\partial \boldsymbol{\theta}^\mathsf{T}} + \frac{1}{c_{t-1}} a_{t-1}(i) p_{ij} \right) + \frac{\partial f_j(y_t)}{\partial \boldsymbol{\theta}^\mathsf{T}} \times \sum_{i=1}^{K} \alpha^*_{t-1}(i) p_{ij}$$

$$\begin{aligned}\text{Recursion for } B_t(j): \quad B_t(j) \;=\; & f_j(y_t) \sum_{i=1}^{K} \left(\alpha^*_{t-1}(i) \frac{\partial^2 p_{ij}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\mathsf{T}} + \frac{1}{c_{t-1}} a^\mathsf{T}_{t-1}(i) \frac{\partial p_{ij}}{\partial \boldsymbol{\theta}^\mathsf{T}} + \frac{\partial p_{ij}}{\partial \boldsymbol{\theta}} \frac{1}{c_{t-1}} a_{t-1}(i) \right. \\ & \left. + \frac{1}{c_{t-1}} B_{t-1}(i) p_{ij} \right) + \frac{\partial f_j(y_t)}{\partial \boldsymbol{\theta}} \times \sum_{i=1}^{K} \left(\alpha^*_{t-1}(i) \frac{\partial p_{ij}}{\partial \boldsymbol{\theta}^\mathsf{T}} + \frac{1}{c_{t-1}} a_{t-1}(i) p_{ij} \right) \\ & + \left[\sum_{i=1}^{K} \left(\alpha^*_{t-1}(i) \frac{\partial p_{ij}}{\partial \boldsymbol{\theta}} + \frac{1}{c_{t-1}} a^\mathsf{T}_{t-1}(i) p_{ij} \right) \right] \times \frac{\partial f_j(y_t)}{\partial \boldsymbol{\theta}^\mathsf{T}} \\ & + \frac{\partial^2 f_j(y_t)}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^\mathsf{T}} \times \sum_{i=1}^{K} \alpha^*_{t-1}(i) p_{ij}.\end{aligned}$$

Appendix B

First derivatives of the p_{ij}:

$$\frac{\partial p_{ij}}{\partial \rho_{i\ell}} = \frac{\mathrm{e}^{\rho_{ij}} (\delta_{j\ell} - \mathrm{e}^{\rho_{i\ell}})}{(1 + \Delta_i)^2}.$$

Note: $\frac{\partial p_{ij}}{\partial \rho_{i'\ell}} = 0$ for $i' \neq i$.

Second derivatives of the p_{ij}:

$$\frac{\partial^2 p_{ij}}{\partial \rho_{ik} \partial \rho_{i\ell}} = \frac{\mathrm{e}^{\rho_{ij}} ((a + \delta_{jk} b)(1 + \Delta_i) - 2\mathrm{e}^{\rho_{i\ell}} b)}{(1 + \Delta_i)^2} \quad \text{where}$$

$a = \delta_{j\ell} \mathrm{e}^{\rho_{ik}} - \delta_{k\ell} \mathrm{e}^{\rho_{i\ell}}$

$b = \delta_{j\ell}(1 + \Delta_i) - \mathrm{e}^{\rho_{i\ell}}$.

References

Albert, P.S., McFarland, H.F., Smith, M.E., Frank, J.A., 1994. Time series for modelling counts from a relapsing-remitting disease: Application to modelling disease activity in multiple sclerosis. Statist. Medicine 13, 453–466.

Altman, Rachel MacKay, Petkau, A. John, 2005. Application of hidden Markov models to multiple sclerosis lesion count data. Statist. Medicine 24, 2335–2344.

Baum, L.E., Egon, J.A., 1967. An inequality with applications to statistical estimation for probabilistic functions of a Markov process and to a model for ecology. Bull. Amer. Meteorol. Soc. 73, 360–363.

Baum, L.E., Petrie, T., 1966. Statistical inference for probabilistic functions of finite state Markov chains. Ann. Math. Statist. 37, 1554–1563.

Baum, L.E., Petrie, T., Soules, G., Weiss, N., 1970. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. Ann. Math. Statist. 41, 164–171.

Bickel, P.J., Ritov, Y., 1996. Inference in hidden Markov models I: Local asymptotic normality in the stationary case. Bernoulli 2, 199–228.

Bickel, P.J., Ritov, Y., Rydén, T., 1998. Asymptotic normality of the maximum likelihood estimator for general hidden Markov models. Ann. Statist. 26, 1614–1635.

Bickel, P.J., Ritov, Y., Rydén, T., 2002. Hidden Markov model likelihoods and their derivatives behave like i.i.d. ones. Ann. Inst. H. Poincaré Probab. Statist. 38, 825–846.

Billingsley, P., 1961. Statistical Inference for Markov Processes. The University of Chicago Press, Chicago.

Campillo, F., Le Gland, F., 1989. MLE for partially observed diffusions: Direct maximization vs. the EM algorithm. Stochastic Processes Appl. 33, 245–274.

Cappé, Olivier, Moulines, Eric, 2005. Recursive computation of the score and observed information matrix in hidden Markov models. In: IEEE Workshop on Statistical Signal Processing, July.

Cappé, Olivier, Moulines, Eric, Rydén, Tobias, 2005. Inference in Hidden Markov Models. Springer, New York.

Collings, I.B., Rydén, T., 1998. A new maximum likelihood gradient algorithm for on-line hidden Markov model identification. In: IEEE International Conference on Acoustics Speech and Signal Processing, May.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm (with discussion). J. Roy. Statist. Soc. 39, 1–38.

Jensen, J.L., Petersen, N.V., 1999. Asymptotic normality of the maximum likelihood estimator in state space models. Ann. Statist. 27, 514–535.

Juang, B.H., Rabiner, L.R., 1991. Hidden Markov models for speech recognition. Technometrics 33, 251–272.

Kennedy Jr., William, Gentle, J.E., 1980. Statistical Computing. Marcel Dekker, Inc., New York.

Khan, R. Nazim, 2002. Statistical modelling and analysis of ion channel data based on hidden Markov models and the EM algorithm. Ph.D. Thesis, University of Western Australia, Crawley, WA 6009.

Leroux, B.G., 1992. Maximum-likelihood estimation for hidden Markov models. Stochastic Processes Appl. 40, 127–143.

Louis, T.A., 1982. Finding the observed information matrix when using the EM algorithm. J. Roy. Statist. Soc. 44, 226–233.

Lystig, Theodore C., Hughes, J.P., 2002. Exact computation of the observed information matrix for hidden Markov models. J. Computat. Graphical Statist. 11 (3), 678–689.

Oakes, David, 1999. Direct calculation of the information matrix via the EM algorithm. J. Roy. Statist. Soc. 61, 479–482.

Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 1994. Numerical Recipes in FORTRAN: The Art of Scientific Computing, second ed. Cambridge University Press.

R Development Core Team, 2005. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0.

Rabiner, L.R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE 77, 257–286.

Thisted, R.A., 1988. Elements of Statistical Computing. Chapman and Hall, New York.

Turner, T. Rolf, Cameron, M.A., Thomson, P.J., 1998. Hidden Markov chains in generalized linear models. Canadian J. Statist. 26, 107–125.

Zucchini, W., Guttorp, P., 1991. A hidden Markov model for space–time precipitation. Water Resources Res. 27, 1917–1923.