The main algorithms used in the `seqHMM` package

Jouni Helske
University of Jyväskylä, Finland

November 8, 2017

1 Introduction

This vignette contains the descriptions of the main algorithms used in the `seqHMM` (Helske and Helske, 2017) package. First, a forward-backward algorithm is presented, followed by a Viterbi algorithm, and the derivations of the gradients for the numerical optimisation routines.

2 Forward–Backward Algorithm

Following Rabiner (1989), the *forward variable*

$$\alpha_{it}(s) = P(\mathbf{y}_{i1}, \ldots, \mathbf{y}_{it}, z_t = s|\mathcal{M})$$

is the joint probability of partial observation sequences for subject i until time t and the hidden state s at time t given the model $\mathcal{M}$. Let us denote $b_s(\mathbf{y}_{it}) = b_s(y_{it1}) \cdots b_s(y_{itC})$, the joint emission probability of observations at time t in channels $1, \ldots, C$ given hidden state s. The forward variable can be solved recursively for subject $i = 1, \ldots, N$:

1. Initialization: For $s = 1, \ldots, S$, compute $\alpha_{i1}(s) = \pi_s b_s(\mathbf{y}_{i1})$
2. Recursion: For $t = 1, \ldots, T-1$, and $r = 1, \ldots, S$, compute $\alpha_{i(t+1)}(r) = \left[\sum_{s=1}^{S} \alpha_{it}(s) a_{sr}\right] b_r(\mathbf{y}_{i(t+1)})$
3. Termination: Compute the likelihood $P(Y_i|\mathcal{M}) = \sum_{s=1}^{S} \alpha_{iT}(s)$

The *backward variable*

$$\beta_{it}(s) = P(\mathbf{y}_{i(t+1)}, \ldots, \mathbf{y}_{iT}|z_t = s, \mathcal{M})$$

is the joint probability of the partial observation sequence after time t and hidden state s at time t given the model $\mathcal{M}$. For subject $i = 1, \ldots, N$, the backward variable can be computed as

1. Initialization: For $s = 1, \ldots, S$, set
   $\beta_{iT}(s) = 1$

2. Recursion: For $t = T-1, \ldots, 1$, and $s = 1, \ldots, S$, compute
   $\beta_{it}(s) = \left[\sum_{r=1}^{S} a_{sr}\right] b_s(\mathbf{y}_{i(t+1)})\beta_{i(t+1)}(r)$

In practice the forward-backward algorithm is prone to numerical instabilities. Typically we scale the forward and backward probabilities, as follows (Rabiner, 1989). For subject $i = 1, \ldots, N$,

1. Initialization: For $s = 1, \ldots, S$, compute

$$\begin{aligned} \alpha_{i1}(s) &= \pi_s b_s(\mathbf{y}_{i1}), \\ c_t &= 1/\sum_{s=1}^{S} \alpha_{i1}(s), \\ \hat{\alpha}_{i1}(s) &= c_t \alpha_{i1}(s) \end{aligned} \tag{1}$$

2. Recursion: For $t = 1, \ldots, T-1$, and $r = 1, \ldots, S$, compute

$$\begin{aligned} \alpha_{i(t+1)}(r) &= \left[\sum_{s=1}^{S} \alpha_{it}(s) a_{sr}\right] b_r(\mathbf{y}_{i(t+1)}) \\ c_t &= 1/\sum_{s=1}^{S} \alpha_{i(t+1)}(s), \\ \hat{\alpha}_{i(t+1)}(s) &= c_t \alpha_{i(t+1)}(s) \end{aligned} \tag{2}$$

3. Termination: Compute the log-likelihood
   $\log P(Y_i|\mathcal{M}) = -\sum_{t=1}^{T} c_t$

The scaling factors c_t from the forward algorithm are commonly used to scale also the backward variables, although other scaling schemes are possible as well. In `seqHMM`, the scaled backward variables for subject $i = 1, \ldots, N$ are computed as

1. Initialization: For $s = 1, \ldots, S$, compute

$$\hat{\beta}_{iT}(s) = 1 \tag{3}$$

2. Recursion: For $t = T-1, \ldots, 1$, and $r = 1, \ldots, S$, compute

$$\begin{aligned} \beta_{it}(s) &= \left[\sum_{r=1}^{S} a_{sr}\right] b_s(\mathbf{y}_{i(t+1)})\beta_{i(t+1)}(r), \\ \hat{\beta}_{it}(s) &= c_t \beta_{it}(s) \end{aligned} \tag{4}$$

Most of the times this scaling method described works well, but in some ill-conditioned cases it is possible that the default scaling still produces underflow in backward algorithm. For these cases, `seqHMM` also supports the computation of the forward and backward variables in log-space. Although numerically more stable, the algorithm is somewhat slower due repeated use of log-sum-exp trick.

3 Viterbi Algorithm

We define the score

$$\delta_{it}(s) = \max_{z_{i1} z_{i2} \cdots z_{i(t-1)}} P(z_{i1} \cdots z_{it} = s, \mathbf{y}_{i1} \cdots \mathbf{y}_{it} | \mathcal{M}),$$

which is the highest probability of the hidden state sequence up to time t ending in state s. By induction we have

$$\delta_{i(t+1)}(r) = \left[\max_s \delta_{it}(s) a_{sr} \right] b_r(\mathbf{y}_{i(t+1)}). \tag{5}$$

We collect the arguments maximizing Equation 5 in an array $\psi_{it}(r)$ to keep track of the best hidden state sequence. The full Viterbi algorithm can be stated as follows:

1. Initialization
   $\delta_{i1}(s) = \pi_s b_s(\mathbf{y}_{i1}), s = 1, \ldots, S$
   $\psi_{i1}(s) = 0$

2. Recursion
   $\delta_{it}(r) = \max_{s=1,\ldots,S}(\delta_{i(t-1)}(s) a_{sr}) b_h(\mathbf{y}_{it})$,
   $\psi_{it}(s) = \arg\max_{s=1,\ldots,S}(\delta_{i(t-1)}(s) a_{sr}), s = 1, \ldots, S; t = 2, \ldots, T$

3. Termination
   $\hat{P} = \max_{s=1,\ldots,S}(\delta_{iT}(s))$
   $\hat{z}_{iT} = \arg\max_{s=1,\ldots,S}(\delta_{iT}(s))$

4. Sequence backtracking
   $\hat{z}_{it} = \psi_{i(t+1)}(\hat{s}_{i(t+1)}), t = T-1, \ldots, 1.$

To avoid numerical underflow due to multiplying many small probabilities, the Viterbi algorithm can be straighforwardly computed in log space, i.e., calculating $\log(\delta_{it}(s))$.

4 Gradients

Following Levinson, Rabiner, and Sondhi (1983), by using the scaled forward and backward variables we have

$$\frac{\partial \log P(Y_i | \mathcal{M})}{\partial \pi_s} = b_s(\mathbf{y}_{i1}) \hat{\beta}_{i1}(s),$$

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial a_{sr}} = \sum_{t=1}^{T-1} \hat{\alpha}_{it}(s) b_r(\mathbf{y}_{i(t+1)}) \hat{\beta}_{i(t+1)}(r),$$

and

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial b_{rc}(m)} = \sum_{t:y_{itc}=m} \sum_{s=1}^{S} \alpha_{1t}(s) a_{sr} \hat{\beta}_{i(t+1)}(r) + \mathrm{I}(y_{i1c} = m) \pi_r \hat{\beta}_{i1}(r).$$

In the direct numerical optimization algorithms used by `seqHMM`, the model is parameterised using unconstrained parameters $\pi'_s, a'_{sr}, b'_{rc}(m)$ such that $a_{sr} = \exp(a'_s r)/\sum_{k=1}^{S} \exp(a'_s k)$, and similarly for emission and initial probabilities. This leads to

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial \pi'_s} = \frac{\partial \log P(Y_i|\mathcal{M})}{\partial \pi_s} \pi_s(1 - \pi_s)$$

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial a'_{sr}} = \frac{\partial \log P(Y_i|\mathcal{M})}{\partial a_{sr}} a_{sr}(1 - a_{sr}),$$

and

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial b'_{rc}(m)} = \frac{\partial \log P(Y_i|\mathcal{M})}{\partial b_{rc}(m)} b_{rc}(m)(1 - b_{rc}(m)).$$

4.1 MHMM case

For mixture HMM with K clusters, we define a full model with $S = S^1 + \cdots + S^K$ states in a block form with $\pi_i = (w_{i1}\pi^1, \ldots, w_{iK}\pi^K)^\top$, where π^k, $k = 1, \ldots, K$ is the vector of initial probabilities for the submodel $\mathcal{M}^k$ and $w_{ik} = \exp(\mathbf{x}_i^\top \gamma_k)/(1 + \sum_{j=2}^{K} \exp(\mathbf{x}_i^\top \gamma_j)$, with $\gamma_1 = 0$.

First note that the log-likelihood of the HMM for ith subject can be written as

$$P(Y_i|\mathcal{M}) = \sum_{s=1}^{S} \sum_{r=1}^{S} \alpha_t(s) a_{sr} b_r(\mathbf{y}_{i(t+1)}) \beta_{t+1}(r),$$

for any $t = 1, \ldots, T-1$. Thus for $t = 1$ we have

$$\begin{aligned} P(Y_i|\mathcal{M}) &= \sum_{s=1}^{S} \sum_{r=1}^{S} \alpha_1(s) a_{sr} b_r(\mathbf{y}_{i2}) \beta_2(r) \\ &= \sum_{s=1}^{S} \alpha_1(s) \sum_{r=1}^{S} a_{sr} b_r(\mathbf{y}_{i2}) \beta_2(r) \\ &= \sum_{s=1}^{S} \alpha_1(s) \beta_1(s) \\ &= \sum_{s=1}^{S} \pi_{is} b_s(\mathbf{y}_{i1}) \beta_1(s). \end{aligned} \tag{6}$$

Therefore the gradients for the unconstrained parameters $\pi_s^{k'}$ of the kth cluster are given as

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial \pi_s^{k'}} = \frac{\partial \log P(Y_i|\mathcal{M}^k)}{\partial \pi_s^k} \pi_s^k (1 - \pi_s^k) w_{ik}.$$

For γ^k, the gradients are of form

$$\frac{\partial \log P(Y_i|\mathcal{M})}{\partial \gamma^k} = \sum_{s=1}^{S} b_s(\mathbf{y}_{i1}) \hat{\beta}_1(s) \frac{\pi_{is}}{\partial \gamma_k}. \tag{7}$$

Now if state s belongs to cluster k, we have

$$\begin{aligned} \frac{\partial \pi_{is}}{\partial \gamma_k} &= \pi_s^k \frac{\partial}{\partial \gamma_k} \frac{\exp(\mathbf{x}_i^\top \gamma_k)}{\sum_{j=1}^{K} \exp(\mathbf{x}_i^\top \gamma_j))} \\ &= \pi_s^k \mathbf{x}_i^\top w_{ik}(1 - w_{ik}), \end{aligned} \tag{8}$$

and

$$\frac{\partial \pi_{is}}{\partial \gamma_k} = -\pi_s^h \mathbf{x}_i^\top w_{ih} w_{ik},$$

otherwise, where h is the index of cluster containing the state s.

References

Satu Helske and Jouni Helske. Mixture hidden Markov models for sequence data: the seqHMM package in R. Preprint ArXiv:1704.00543, 2017. URL `http://arxiv.org/abs/1704.00543`.

S. E. Levinson, L. R. Rabiner, and M. M. Sondhi. An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition. *Bell System Technical Journal*, 62(4):1035–1074, 1983. ISSN 1538-7305. doi: 10.1002/j.1538-7305.1983.tb03114.x. URL `http://dx.doi.org/10.1002/j.1538-7305.1983.tb03114.x`.

Lawrence Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989. doi: 10.1109/5.18626.