An Introduction to the R package HMMmlselect

Yang Chen[*], Chu-Lan Kao[†]

February 5, 2019

This article is an introduction to the R package `HMMmlselect` for estimating the order (i.e., the number of states) of hidden Markov models (HMM). Section 1 briefly describes the theoretical framework of the marginal Likelihood method as proposed in Chen et al. [2017+]. Section 2 states R functions contained in the package. Section 3 demonstrates usage of the package.

1 HMM Order Selection via Marginal Likelihood

A hidden Markov model (HMM) has two major components: an underlying (unobserved) ergodic Markov chain, denoted by $\mathbf{X} = \{X_i, i \geq 0\}$, and an observed trajectory, denoted by $\mathbf{Y} = \{Y_i, i \geq 1\}$. Let K be the number of hidden states, then each X_i takes a value in $\mathcal{X}_K = \{1, \cdots, K\}$. The transition matrix of $\mathbf{X}$ is denoted by $Q_K = \{q_{kl}, 1 \leq k, l \leq K\} \in \mathcal{Q}_K$, i.e., $q_{kl} = P(X_{i+1} = l | X_i = k)$ for all $i \geq 0$. Conditioning on $\mathbf{X}$, $\mathbf{Y}$ are independent random variables on $\mathcal{Y}$, and the distribution of Y_i given $X_i = k$ is $f(\cdot|\boldsymbol{\theta}_k)$ for $i \geq 1$ and $k \in \mathcal{X}_K$, where $\boldsymbol{\theta}_k \in \Theta$. Denote the model parameters by $\boldsymbol{\phi}_K = (Q_K; \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K) \in \mathcal{Q}_K \times \Theta^K = \Phi_K$.

In an HMM with n observations, we observe $\mathbf{y}_{1:n} = \{y_1, y_2, \cdots, y_n\} \in \mathcal{Y}^n$ but not the underlying process $\mathbf{x}_{1:n} = \{x_1, x_2, \cdots, x_n\}$. The likelihood of $\mathbf{y}_{1:n}$ given the parameters $\boldsymbol{\phi}_K$, after integrating out the hidden states $\mathbf{x}_{1:n}$, is

$$p(\mathbf{y}_{1:n}|\boldsymbol{\phi}_K) = \sum_{\mathbf{x}_{1:n} \in \mathcal{X}_K^n} \left[\prod_{k=1}^{K} \left\{ \prod_{i:x_i=k} f(y_i|\boldsymbol{\theta}_k) \right\} \times \left\{ \prod_{i=1}^{n} q_{x_{i-1}x_i} \right\} \right], \tag{1.1}$$

where $\mathcal{X}_K^n$ denotes the product space of n copies of $\mathcal{X}_K$.

[*] Department of Statistics, University of Michigan, Ann Arbor, MI 48109, U.S.A, ychenang@umich.edu

[†] Institute of Statistics, National Chiao Tung University, Hsinchu, Taiwan, chulankao@gmail.com

Let $p_0(\boldsymbol{\phi}_K)$ be the prior distribution for $\boldsymbol{\phi}_K$, then we estimate the number of hidden states by the K that maximize the marginal likelihood, after integrating out the parameters,

$$\hat{K}_n := \text{arg max}_{K\geq 1}\{p_K(\mathbf{y}_{1:n})\} = \text{arg max}_{K\geq 1}\left\{\int_{\Phi_K} p(\mathbf{y}_{1:n}|\boldsymbol{\phi}_K)p_0(\boldsymbol{\phi}_K)d\boldsymbol{\phi}_K\right\}.$$

Computation. Since the marginal likelihood $p_K(\mathbf{y}_{1:n})$ is the normalizing constant of the posterior distribution $p(\boldsymbol{\phi}_K|\mathbf{y}_{1:n}) = p(\mathbf{y}_{1:n}|\boldsymbol{\phi}_K)p_0(\boldsymbol{\phi}_K)/p_K(\boldsymbol{y}_{1:n})$ and the un-normalized posterior likelihood $p(\mathbf{y}_{1:n}|\boldsymbol{\phi}_K)p_0(\boldsymbol{\phi}_K)$ can be evaluated at any $\boldsymbol{\phi}_K$ using the forward algorithm [Baum and Petrie, 1966, Baum et al., 1970], we can use the importance sampling (or reciprocal importance sampling) strategy to estimate the marginal likelihood [Geweke, 1989, Oh and Berger, 1993, Newton and Raftery, 1994, Gelfand and Dey, 1994, Chen and Shao, 1997, DiCiccio et al., 1997, Neal, 2005, Steele et al., 2006, Ionides, 2008].

The detailed procedure for estimating the marginal likelihood $p_K(\mathbf{y}_{1:n})$ is as follows.

1. Obtain posterior samples. Sample from $p(\boldsymbol{\phi}_K|\mathbf{y}_{1:n})$ using a preferred Markov chain Monte Carlo (MCMC) algorithm (see Liu [2001] and references therein), and denote the samples by $\{\boldsymbol{\phi}_K^{(i)}\}_{i=1}^N$ (where N is often a few thousand).

2. Find a "good" importance function. Construct the importance function $g(\cdot)$ by fitting a Gaussian mixture with R package `Mclust` [Fraley and Raftery, 2006] or by simply fitting a multivariate Gaussian using samples $\{\boldsymbol{\phi}_K^{(i)}\}_{i=1}^N$.

3. Choose a finite region. Choose Ω_K to be a bounded subset of the parameter space such that $\int_{\Omega_K} g(\cdot) \approx 0.5$. This is achieved through finding an appropriate finite region for each mixing component of $g(\cdot)$, avoiding the tail parts.

4. Define $p(\mathbf{y}_{1:n}, \boldsymbol{\phi}) = p(\mathbf{y}_{1:n}|\boldsymbol{\phi})p_0(\boldsymbol{\phi})$. Estimate $p_K(\boldsymbol{y}_{1:n})$ using either way as follows:

   - Reciprocal importance sampling. Approximate $p_K(\mathbf{y}_{1:n})$ by

     $$\hat{p}_K^{(RIS)}(\boldsymbol{y}_{1:n}) = \left[\frac{1}{N\int_{\Omega_k} g(\cdot)}\sum_{i=1}^{N}\frac{g(\boldsymbol{\phi}_K^{(i)})}{p(\mathbf{y}_{1:n}, \boldsymbol{\phi}_K^{(i)})} I_{\boldsymbol{\phi}_K^{(i)}\in\Omega_K}\right]^{-1}, \qquad (1.2)$$

     where $I_{\boldsymbol{\phi}_K^{(i)}\in\Omega_K} = 1$ if $\boldsymbol{\phi}_K^{(i)} \in \Omega_K$ and zero otherwise.

   - Importance sampling.

     (a) Draw M independent samples from $g(\cdot)$, denoted by $\{\boldsymbol{\psi}_K^{(j)}\}_{1\leq j\leq M}$.

(b) Approximate $p_K(\mathbf{y}_{1:n})$ by

$$\hat{p}_K^{(IS)}(\boldsymbol{y}_{1:n}) = \frac{1}{MP_\Omega}\sum_{j=1}^{M}\frac{p(\mathbf{y}_{1:n},\boldsymbol{\psi}_K^{(j)})}{g(\boldsymbol{\psi}_K^{(j)})} I_{\boldsymbol{\psi}_K^{(j)}\in\Omega_K}, \tag{1.3}$$

where $I_{\boldsymbol{\psi}_K^{(j)}\in\Omega_K} = 1$ if $\boldsymbol{\psi}_K^{(j)} \in \Omega_K$ and zero otherwise; $P_\Omega = \#\mathcal{S}/N$, where $\mathcal{S} = \{i : \boldsymbol{\phi}_K^{(i)} \in \Omega_K; 1 \le i \le N\}$ and $\#\mathcal{S}$ denotes its cardinality.

Choice of Priors. We choose independent, conjugate priors for each $\boldsymbol{\theta}_k$ and Q_K based on two principles: first is to be flat/noninformative and second is to be conformal to the scale of the data. The hyper parameters are set to be fixed constants, see Chen et al. [2017+] for more details. The choice of priors is already implemented in the R package thus the user does not need to supply it unless scientific knowledge indicates certain hyper parameter values, which could be set manually by adding corresponding fields to the optional input list.

2 R Implementation

Order Selection. The function that estimates the order of a Gaussian HMM is

```
results = HMMmlselect ( y , optionalfit = list ( ) )
```

This function only requires the observed HMM trace (i.e., time-series) $\boldsymbol{y}_{1:n}$, a column vector, and a list named `optionalfit`, which contains optional inputs and its default is an empty list, see Appendix A for the available optional inputs and their default values.

It returns (1) the estimated number of hidden states using the marginal likelihood method, (2) the marginal likelihood values corresponding to $2, 3, \ldots$ number of hidden states, and (3) the fitted model parameters given the estimated number of hidden states.

Visualization. The estimation results can be visualized using

```
PlotHMM ( y , results )
```

The input `results` is the output from function `HMMmlselect`.

Simulation. The function that generates an HMM trace is

```
y = HMMsim ( n , optionalsim = list ( ) )
```

This function requires specifying the number of observations for the simulated HMM, `n`, and an list named `optionalsim`, which contains optional inputs and its default value is an empty list, see Appendix B for the available optional inputs and their default values.

3 Numerical Demonstration

Please run `'demon.R'` for annotated numerical examples that illustrate the usage of the functions in the package and the corresponding results.

'demon.R'

```
# Example 1: simple example with all tuning parameters set as default values

obs = HMMsim ( n = 200 )$obs # simulate a 200 observations HMM

results = HMMmlselect ( y = obs ) # perform order selection and estimation

PlotHMM ( y = obs, results ) # visualize the results, see figure 1



# Example 2: manually setting values for hyper parameters

of = list ( priors = list (mu_prior_mean = 1:3,
             mu_prior_sd = rep (0.2, 3)),
             nu = 2, s2 = 0.1,
             P_prior = matrix(1, 3, 3),
             pi_prior = rep(1, 3) )

results = HMMmlselect ( y = obs , optionalfit = of )
```

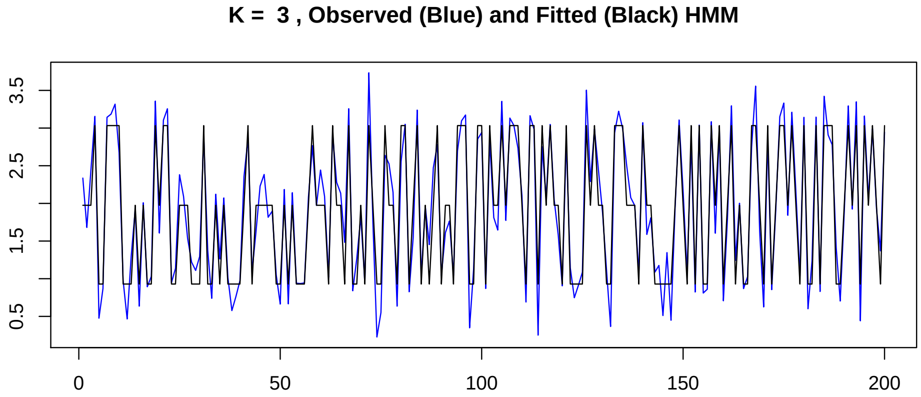

Figure 1: A fitted trace after determining the number of states using HMMmlselect package.

```
# Example 3: use Gaussian mixture model marginal likelihood

results = HMMmlselect ( y = obs, list ( boolHMM = FALSE ) )
```

```
# Example 4: use BIC to determine the order of HMM

resultsBIC = RobustBIC ( y = obs )
```

References

L. E. Baum and T. Petrie. Statistical inference for probabilistic functions of finite state Markov chains. *Ann. Math. Stat.*, 37(6):1554–1563, 1966.

L. E. Baum, T. Petrie, G. Soules, and N. Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Stat.*, 41(1): 164–171, 1970.

M.-H. Chen and Q.-M. Shao. On Monte Carlo methods for estimating ratios of normalizing constants. *Ann. Statist.*, 25:1563–1594, 1997.

Yang Chen, Cheng-Der Fuh, Chu-Lan Kao, and Samuel Kou. Determine the number of states in hidden markov models via marginal likelihood. *submitted*, 2017+.

T. J. DiCiccio, R. E. Kass, A. E. Raftery, and L. Wasserman. Computing Bayes factors by combining simulation and asymptotic approximations. *J. Amer. Statist. Assoc.*, 92(439): 903–915, 1997.

Chris Fraley and Adrian E Raftery. Mclust version 3: an r package for normal mixture modeling and model-based clustering. Technical report, WASHINGTON UNIV SEATTLE DEPT OF STATISTICS, 2006.

A. E. Gelfand and D. K. Dey. Bayesian model choice: Asymptotics and exact calculations. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 56(3):501–514, 1994.

J. Geweke. Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 57:1317–1339, 1989.

E. L. Ionides. Truncated importance sampling. *J. Comput. Graph. Statist.*, 17(2):295–311, 2008.

J. S. Liu. *Monte Carlo Strategies in Scientific Computing.* Springer-Verlag New York, Inc., 2001.

R. M. Neal. Estimating ratios of normalizing constants using linked importance sampling. *Technical Report No. 0511, Department of Statistics, Univ. Toronto*, 2005.

M. A. Newton and A. E. Raftery. Approximate Bayesian inference with the weighted likelihood bootstrap. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 56(1):3–48, 1994.

M.-S. Oh and J. O. Berger. Integration of multimodal functions by Monte Carlo importance sampling. *J. Amer. Statist. Assoc.*, 88(422):450–456, 1993.

R. J. Steele, A. E. Raftery, and M. J. Emond. Computing normalizing constants for finite mixture models via incremental mixture importance sampling. *J. Comput. Graph. Statist.*, 15(3):712–734, 2006.

A Optional Inputs for HMMmlselect

The R command for specifying the optional inputs for function `HMMmlselect` is

```
of = list ( a = , b = , c = , ... ) # Caution: replace the a, b, c, ...
results = HMMmlselect ( y, optionalfit = of )
```

Users can set values for certain chosen fields by replacing the `a, b, c, ...` with corresponding field names listed as follows and simply ignoring the other unnecessary fields.

Name	Explanation	Default Value(s)
`Kfits`	Possible Number of States	$(2, 3, \ldots, 6)$
`RIS`	Using reciprocal importance sampling	FALSE
`IS`	Using importance sampling	TRUE
`RunParallel`	Using parallel computing	TRUE
`boolUseMclust`	Using `Mclust` package	FALSE
`priors`	A list of hyper parameters	Flat[a]
`boolHMM`	Use HMM marginal likelihood	TRUE [b]

[a] See Chen et al. [2017+] for details.

[b] Use Gaussian mixture marginal likelihood if FALSE.

The list of hyper parameters for `priors` is

```
list (mu_prior_mean = , # prior (Gaussian) mean for mu

      mu_prior_sd = ,  # prior standard deviation for mu

      nu = , s2 = ,  # degree of freedom and scale
                     # for scaled inverse prior of sigma

      P_prior = , # Dirichilet priors for rows of transition matrix

      pi_prior = ) # Dirichilet prior for first hidden state
```

B Optional Inputs for HMMsim

The R command for specifying the optional inputs for function `HMMsim` is

```
os = list ( a = , b = , c = , ... ) # Caution: replace the a, b, c, ...

y = HMMsim ( n, optionalsim = os )
```

Users can set values for certain chosen fields by replacing the `a, b, c, ...` with corresponding field names listed as follows and simply ignoring the other unnecessary fields.

Name	Explanation	Default Value(s)
`Ksim`	Number of States	3
`P`	Transition matrix	Flat $K \times K$
`mu`	Means of K states	$1, 2, \ldots, K$
`sigma`	Standard deviations of K states	$(0.1, \ldots, 0.1)$
`pi`	Distribution of first hidden state	Flat
`BoolWritetoFile`	Output simulated trace to file	FALSE
`Filenameoutput`	Filename to output simulated trace	'HMMtrace.txt'

C Other Functions

C.1 Robust BIC Computation

The Bayesian information criterion (BIC) is a widely adopted method for order selection of HMMs. We also implement this in the package. Robust BIC values results from maximum likelihood estimators (MLEs) obtained through multiple starting points and double-checked with posterior samples. The function that calculates the BIC values robustly is:

```
BICresults = RobustBIC ( y , optionalbic = list ( ) )
```

This function only requires the observed HMM trace y and returns the BIC values of $K = 2, 3, \ldots, 6$ hidden states. The optional input, `optionalbic`, is an empty list by default and can be specified as a list containing any of the following fields.

Name	Explanation	Default Value(s)
`Kfits`	Possible number of states	$2, 3, \ldots, 6$
`Nstart`	Number of starting points	50
`verbose`	Print details	FALSE

This function returns the estimated number of hidden states through minimizing the BIC, the BIC values of all the possible number of hidden states, and the fitted model parameters under the estimated number of hidden states under the BIC method.

C.2 HMM and Gaussian Mixture Fittings

The following function performs (a) HMM fitting through the Expectation-Maximization algorithm (METHOD = 1), (b) HMM fitting through the Markov chain Monte Carlo algorithm (METHOD = 2), and (c) Gaussian mixture model fitting through the Markov chain Monte Carlo algorithm (METHOD = 3).

```
results = HMMfit ( y , K, METHOD, optionalfit = list ( ) )
```

This function only requires the observed HMM trace y, the number of hidden states K, and the METHOD chosen. The optional input, `optionalfit`, is an empty list by default and can be specified as a list containing any of the following fields.

Name	Explanation	Default Value(s)
`Ngibbs`	Number of samples	5000
`Burnin`	Number of burnin	5000
`Thin`	Number of thinning	10
`Nstart`	Number of starting points	50
`verbose`	Print details	FALSE
`priors`	Same as in `HMMmlselect`	Flat

This function returns the estimated number of hidden states through maximizing the marginal likelihood, the marginal likelihood values of all the possible number of hidden states, and a summary of posterior distributions (together with posterior samples) of model parameters under the estimated number of hidden states under the marginal likelihood method.