

# Bayesian Model Averaging Across Model Spaces via Compact Encoding

**Ke Yin**

Department of Computer Science  
SUNY – Albany  
Albany, NY 12222  
ke@cs.albany.edu

**Ian Davidson\***

Department of Computer Science  
SUNY – Albany  
Albany, NY 12222  
davidson@cs.albany.edu

## Abstract

*Bayesian Model Averaging (BMA) is well known for improving predictive accuracy by averaging inferences over all models in the model space. However, Markov chain Monte Carlo (MCMC) sampling, as the standard implementation for BMA, encounters difficulties in even relatively simple model spaces. We introduce a minimum message length (MML) coupled MCMC methodology, which not only addresses these difficulties but has additional benefits. We illustrate the methodology with a mixture component model example (clustering) and show that our approach produces more interpretable results when compared to Green’s popular reverse jump sampling across model sub-spaces technique. The MML principle mathematically embodies Occam’s razor by assigning penalized prior probabilities to complicated models. We find that BMA prediction based on sampling across multiple sub-spaces of different complexity makes much improved predictions compared to the single best (shortest) model.*

## 1. Introduction

Bayesian model averaging (BMA) removes the model uncertainty by making inferences from all possible models in the considered model spaces weighted by their posterior probabilities. The removal of uncertainty decreases the associated risk of making predictions from only a single model hence improves the prediction accuracy [9]. The standard BMA implementation involves running a Markov chain that has the posterior distribution of the models as its stationary distribution typically using either the Gibbs’ or Metropolis-Hasting’s algorithm. However, the method encounters difficulties [6] in even simple model spaces, such as mixture models. Not only are the full conditional distributions difficult to derive and sample from, but mixing can also become slow, which can be exacerbated if the data has relatively high dimensions. Thus, powerful as BMA is, these difficulties prevented its wide application in the machine learning and statistical inference problems.

The Minimum Message Length (MML) principle [1] evaluates the quality of a model by the total message length to encode both the model and the data  $D$ . The message length of a model  $\theta$  has a simple relationship with the posterior probability of the model that  $p(\theta|D) \propto \exp(-\text{MsgLen}(\theta) - \text{MsgLen}(D|\theta))$  where the information length is measured in nits. This property allows sampling models according to their posterior probabilities calculated from the message lengths of the models. In this paper we show that the MML-MCMC sampler has significant additional benefits over MCMC with traditional estimators, in particular:

- The full conditional distribution defined by the message length distribution is easier to derive and sample from.
- MML mathematically embodies Occam’s razor by associating the model’s prior probability with its complexity.
- MML uses message length as the universal metric to evaluate the quality of models, allowing easily sampling across different model subspaces or even fundamentally different model spaces.
- The MML discretization of the model space into regions with each region having a representative model greatly reduces the size of the model space, resulting in more efficient sampling.

In this paper, we illustrate the MML coupled MCMC framework with mixture model as an example although the approach is applicable to other model families. Bayesian inference from Gaussian mixtures with unknown

number of components was first addressed by Richardson and Green [5] using their reversible jump method. Though their approach is readily applicable to univariate problems, to our knowledge it has not been successfully demonstrated on high dimensional data. There exist work that attempts to bypass the problem of estimating the number of components by assuming infinite models using the Dirichlet process methods [10][13]. The method relies on the mathematical convenience of a conjugate prior and we intend to compare this approach to ours in future work.

We start by introducing the MML principle and its formalization for mixture models. We then formally propose the five kernel moves in the MML-MCMC sampler and prove its convergence to the MML defined posterior. The convergence is then empirically verified via chain diagnosis on an artificially generated Gaussian dataset. Next, we compare MML-MCMC sampler with the reversible jump sampler [5] and illustrate that MML-MCMC sampler finds more interpretable posterior distribution of  $k$  by adopting the automatic prior in the MML principle. Finally, we empirically explore the predictive capability of BMA with a number of standard machine learning and statistical datasets. This work is an extension of our earlier work [12] to include jumping between model spaces of varying complexity. To our knowledge, it is the first work to address model averaging across model spaces of varying dimension for predictive purposes.

## 2. Minimum Message Length Principle

The minimum message length principle was first proposed by Wallace and Boulton [2] using Shannon's information encoding scheme. The principle measures both the model complexity and the model's fit to the data in nits of information. The total message length equals to the sum of two parts: the message length to encode the model and the message length to encode the data in the presence of the model.

$$MsgLen(D, \theta) = MsgLen(\theta) + MsgLen(D | \theta) \quad (1)$$

One can only encode the model to some finite precision otherwise the message length will be infinite. For this reason, the model space must be discretized into a number of cells. All the models contained in each cell are considered to be uniformly distributed throughout it and the cell is represented by the model in the center. The larger the cell volume is, the less number of cells will there be in the model space, and the less information required specifying a particular model. However, larger cell volumes also lead to inaccurate model specifications resulting in longer expected message length in the second part of the message. For a model with  $n_p$  parameters, the optimal volume of a cell  $V$  that minimizes the total expected message length is determined [4] by

$$V = \sqrt{\frac{1}{(\kappa_{n_p})^{n_p} \det(F(\theta))}} \quad (2)$$

Here,  $\kappa_d$  is the lattice constant [7], and  $F(\theta)$  is the expected Fisher information. Using the optimal cell volume produces a message length of:

$$MsgLen = \frac{n_p}{2} \ln(\kappa_{n_p}) - \ln(p(\theta)) + \frac{1}{2} \ln(\det(F(\theta))) + L(\theta) + \frac{n_p}{2} \quad (3)$$

In the above equation  $\theta$  is no longer a continuous variable. It may only change with the unit of the cell volume. Rewriting equation (3) yields

$$MsgLen = -\ln(p(\theta)) - \ln\left(\sqrt{\frac{1}{(\kappa_{n_p})^{n_p} \det(F(\theta))}}\right) + L(\theta) + \frac{n_p}{2} = -\ln(p(\theta) \cdot V) - \ln(p(D | \theta)) + \frac{n_p}{2} \quad (4)$$

This expression is very close to the logarithm of the famous Bayesian theorem shown below, less the probability of the data which is a constant.

$$-\ln p((\theta | D)) = -\ln p(\theta) - \ln(p(D | \theta)) + \ln(p(D)) \quad (5)$$

We see that the first part of the message length serves as the prior probability for the model. This suggests more complicated models that take more information to encode should have less prior probability compared to simpler models. This is consistent with and quantifies the Occam's razor philosophy to choose simpler models when the fitness of the data is equivalent. Later we will see this automatic prior yields more interpretable results in section 7 compared to uniform prior over models with different complexity.

The discretization in MML has introduced imprecision on  $\theta$ . However, it can be shown that this imprecision is no greater than the difference between the estimator and the true value of  $\theta$  [1]. In other words, the MML defined optimal cell volume equals to the inherent imprecision due to the distribution of the data. Thus, we have

$$MsgLen = -\ln p(\theta) - \ln p((\theta | D)) + \frac{n_p}{2} - \ln(p(D)) \quad (6)$$

Since  $\frac{n_p}{2}$  is constant, we obtain the transformation between the message length (measured in nits) and the posterior distribution.

$$p(\theta | D) \propto e^{-MsgLen} \quad (7)$$

The discretization of the parameter space yields much less models to consider, making the sampling more efficient. Moreover, the message length distribution can be truncated by a constant amount  $c$  while still transforms into the same posterior distribution. Let  $\Theta = \{\theta_1, \theta_2, \dots, \theta_n\}$  be a set of models and let  $M = \{m_1, m_2, \dots, m_n\}$  be the corresponding two part message lengths of the models. Then both message length distributions  $M = \{m_1, m_2, \dots, m_n\}$  and  $M' = \{m_1 - c, m_2 - c, \dots, m_n - c\}$  will transform into the equivalent posterior distributions of  $\Theta$ . The proof is straightforward since

$$p(\theta_i | D) = \frac{e^{-m_i}}{\sum_{i=1}^n e^{-m_i}} = \frac{e^{-(m_i - c)}}{\sum_{i=1}^n e^{-(m_i - c)}} \quad (8)$$

This result makes the message length calculation significantly less cumbersome when trying to sample from the transformed distribution. Instead of calculating the absolute message length for each model, one only needs to calculate their relative differences.

### 3. Minimum Message Length in Mixture Component Model

A mixture component model ( $M$ ) with  $d$  dimensions and  $n$  instances can be specified as  $M = \{w, p, k, \theta\}$ , where

- |                                |  |
|--------------------------------|--|
| $w = \{w_1, w_2, \dots, w_n\}$ | The assignments of the instances, each $w_i = \{1, 2, \dots, k\}$ $i = 1, 2, 3, \dots, n$  |
| $p = \{p_1, p_2, \dots, p_k\}$ | The relative weight of each component $p_i \geq 0$ $\sum_{i=1}^k p_i = 1$  |
| $k = 1, 2, \dots, K$           | Number of components, $K$ is the maximum possible value for $k$ .  |
| $\theta = \{\mu, \sigma\}$     | The $k \times d$ parameter matrices of the mixture component model. $\{\mu_{ij}, \sigma_{ij}\}$ stands for the mean and standard deviation for the $j$ th dimension in the $i$ th component. |

The full encoding of the mixture model consists of two parts. The first part of the message encodes the independent parameters  $k, p, \theta$  according to their prior distributions. The second part of the message encodes the assignments  $w$  and the data. Given the knowledge of  $p$ , the optimal coding dictionary encodes the assignment to component  $j$  with  $-\ln p_j$  nits. Then the instances are encoded with knowledge of  $w$  and  $\theta$ . An instance  $x$  with  $d$  dimensions will take  $-\sum_{m=1}^d \ln f(x_m | \mu_{w(x),m}, \sigma_{w(x),m})$  nits to encode. In summary,

$$MsgLen = -\ln(h(k, p, \theta)) - n \sum_{j=1}^k p_j \ln p_j - \sum_{i=1}^n \sum_{m=1}^d \ln f(x_{i,m} | \mu_{w(i),m}, \sigma_{w(i),m}) + \frac{1}{2} \ln(\det F(k, p, \theta)) + \frac{n_p}{2} \ln(\kappa_{n_p}) + \frac{n_p}{2} \quad (9)$$

An expansion similar to that of Baxter and Oliver [3] gives the full message length expansion.

$$MsgLen = k \sum_{m=1}^d \ln(2\sigma_{pop,m}^2) - \ln(k-1)! + \sum_{j=1}^k \sum_{m=1}^d \ln \frac{\sqrt{2p_j}}{(\sigma_{j,m})^2} + \frac{3k-1}{2} \ln n + n \sum_{j=1}^k p_j \ln p_j \quad (10)$$

$$- \frac{nd}{2} \ln(2\pi) + \sum_{i=1}^n \sum_{m=1}^d \ln \sigma_{w(i),m} + \sum_{i=1}^n \sum_{m=1}^d \frac{(x_{i,m} - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} - \ln k! + \frac{2dk + k - 1}{2} (\ln(\kappa_{n_p}) + 1)$$

Each different parameters tuple  $\{k, p, \theta, w\}$  corresponds to a model in the model space and its message length is calculated by equation (10) from which the joint distribution  $p(k, p, \theta, w, D)$  can be derived. Also the full conditional distribution where some of the parameters are known, such as  $p(\theta | k, p, w, D)$ , can be conveniently transformed from the same message length distribution only by fixing the given parameters as constants.

#### 4. MML Coupled Markov Chain Monte Carlo

The standard approach to sampling from the posterior distribution is by running a Markov chain that has the posterior distribution as its stationary distribution. In the chain, the state at time  $t$ ,  $M^t = \{w^t, p^t, k^t, \theta^t\}$ , will only depend on the state at time  $t-1$ ,  $M^{t-1} = \{w^{t-1}, p^{t-1}, k^{t-1}, \theta^{t-1}\}$ . The iteration from time  $t-1$  to  $t$ , usually called a sweep, consists of sampling each parameter conditionally on all other remaining parameters. In the MML-MCMC sampler for mixture component model, there are five moves in each sweep.

- Sampling  $w$  from  $p(w | k, p, \theta, D)$
- Sampling  $p$  from  $p(p | k, w, \theta, D)$
- Death and Re-Birth of a component
- Split and Combine of a component
- Sampling  $\theta$  from  $p(\theta | k, p, w, D)$

The first, second and fifth steps sample from the conditional distribution using Gibbs algorithm while the third and fourth steps uses a Metropolis-Hasting algorithm to handle empty components and jump across subspaces. We will formalize each move in detail in the rest part of this section.

##### 4.1 Sampling $w$

The assignment  $w(x)$  of an instance  $x$  will affect the message length in the following way. Firstly the instance assignment  $w(x)$  to a component must be encoded with  $\ln(p_{w(x)})$  nits. Then the data point itself must be encoded using  $\theta_{w(x)}$  which will take  $-\ln(x | \theta_{w(x)})$  nits. We only consider the message length parts that are functions of  $w(x)$  in equation (10) since all other parts of message length are constants and can be ignored in calculating the full conditional distribution. The truncated message length distribution that an instance  $x$  is assigned to component  $j$  is:

$$MsgLen(x, j) = -\ln(p_j) - \ln f(x | \theta_j) = -\ln(p_j) + \frac{d}{2} \ln(2\pi) + \sum_{m=1}^d \ln \sigma_{j,m} + \sum_{m=1}^d \frac{(x_m - \mu_{j,m})^2}{2\sigma_{j,m}^2} \quad (11)$$

The distribution consists of  $k$  message lengths each corresponds to the  $k$  possible assignments for the instance. This message length distribution can be transformed into the full conditional distribution.

$$p(w(x) = j | p, k, \theta, D) = \frac{e^{-MsgLen(x, j)}}{\sum_{j=1}^k e^{-MsgLen(x, j)}} \quad (12)$$

The instance is stochastically assigned to one of the components according this distribution. This stochastic assignment process repeats for each instance in the dataset.

## 4.2 Sampling $p$

We are to sample  $p$  from the full conditional distribution  $p(p | k, w, \theta, D)$ . Here,  $p$  can be viewed as a multinomial distribution with uniform priors whose  $p.d.f$  is given by

$$f(p_1, p_2, \dots, p_{k-1}) = \frac{n!}{\prod_{j=1}^k (n_j!)} \prod_{j=1}^k p_j^{n_j}, \text{ where } p_k = 1 - \sum_{j=1}^{k-1} p_j, n_k = n - \sum_{j=1}^{k-1} n_j \quad (13)$$

We sample  $p$  using a message length defined distribution derived from the equation below.

$$MsgLen = -\ln(h(p)) - n \sum_{j=1}^k p_j \ln p_j + \frac{1}{2} \ln(\det F(p)) = -\ln(k-1)! - \ln(n!) + \sum_{j=1}^k (n_j!) - \sum_{j=1}^k n_j \ln(p_j) + \frac{1}{2} \ln \frac{n^{k-1}}{\prod_{j=1}^k p_j} \quad (14)$$

The optimal cell volume for the  $k-1$  parameters that minimize the expected message length is given by

$$V = \sqrt{\frac{1}{(\kappa_{k-1})^{k-1} \det(F(\hat{p}))}} = \sqrt{\frac{\prod_{j=1}^k \hat{p}_j}{(\kappa_{k-1} n)^{k-1}}} \quad (15)$$

We approximate the volume of the cell by a hypercube. The width of the  $j$ th ( $j=1$  to  $k-1$ ) dimension of the cube  $s_j$  equals

$$s_j = \sqrt{\frac{\hat{p}_j}{\kappa_{k-1} n} (\hat{p}_k)^{\frac{1}{2(k-1)}}} \quad (16)$$

We sample each  $p_j$  ( $1 \leq j \leq k-1$ ) around the maximal likelihood estimator  $\hat{p}_j = \frac{n_j}{n}$  with the cell width

of  $\sqrt{\frac{\hat{p}_j}{\kappa_{k-1} n} (\hat{p}_k)^{\frac{1}{2(k-1)}}}$ . We truncate the message length distribution by the message length at  $\hat{p}_j$ . The truncated message length distribution is specified by

$$\begin{aligned} MsgLen(p_j) &= -n_j \ln p_j - n_k \ln p_k + \frac{1}{2} \ln \det(F(p)) - \left( -n_j \ln \hat{p}_j - n_k \ln \hat{p}_k + \frac{1}{2} \ln \det(F(\hat{p})) \right) \\ &= \frac{1}{2} \ln \frac{\hat{p}_j \hat{p}_k}{p_j p_k} + n_j \ln \frac{\hat{p}_j}{p_j} + n_k \ln \frac{\hat{p}_k}{p_k} = \left( n_j + \frac{1}{2} \right) \ln \frac{\hat{p}_j}{p_j} + \left( n_k + \frac{1}{2} \right) \ln \frac{\hat{p}_k}{p_k} \end{aligned} \quad (17)$$

We sample the value of  $p_j$  using the fully conditional distribution transformed from this message length distribution. And the sampling repeats for all  $j$  values from  $1$  to  $k-1$ .

### 4.3 Death and Rebirth

Completely empty components and components that contain only single instance require special attention, as they usually indicate a redundant part of the model that increases the message length to specify the model yet will not help compress the data. To increase the mixing rate and shorten the total message length, better parameters can be proposed to help encode the data. The death and rebirth moves involve destroying an empty component or a nearly empty component and reinitialize it with new parameters, which will hopefully encode the data more efficiently.

As we adopt uniform priors for  $\theta$  and  $p$ , their message length is always the same no matter what values we encode. Reinitializing the parameter values for the empty component will not change the total message length since no instance is encoded with the empty component. Also, we maintain a symmetrical parameter proposal function by reinitialize the parameters randomly so that the detail balance will be satisfied [10]. If a component contains one instance, we cannot always reinitialize the component since changes of the parameters of the component will affect the message length to encode the single instance in that component. The increase of the message length is  $-\ln f(x|\theta_{new})$  since it takes 0 nits to encode the instance previously (the parameters of the component contains all information about the instance). Thus the probability of accepting the move is  $f(x|\theta_{new})$ . If the dimensionality of the data is high, then the acceptance rate becomes very low. One way of overcoming this difficulty is by only reinitializing the parameter only in one random dimension.

### 4.4 Split and Combine

This is the move that enables sampling across subspaces with different  $k$  values. At each split and combine move, we will stochastically determine whether to attempt to split or to combine. When  $k=1$ , we always attempt the split move and when  $k$  equals the maximum number of components  $K$ , we always attempt the combination move. At all other  $k$  values, both the probability to split and the probability to combine will equal to 0.5.

#### 4.4.1 Split

First a random component is chosen as the split candidate. Then we choose the dimension with the largest standard deviation to generate the split pivot. The split pivot  $sp$  is randomly generated from  $[\mu - \sigma, \mu + \sigma]$  in the chosen dimension and all instances inside the component are divided into two groups: those larger than the pivot and those smaller in the chosen dimension. All the parameters of the two groups are estimated using the maximum likelihood estimator. Let  $s$  be the candidate component proposed to split into  $s_1$  and  $s_2$ , the change of the message length after the proposed split can be calculated from equation (10):

$$\begin{aligned} \Delta MsgLen = & \sum_{m=1}^d \ln(2\sigma_{pop,m}) - \ln k - \frac{2d+1}{2} (\ln(\kappa_{n_p}) + 1) + \sum_{m=1}^d \left( \ln \frac{\sqrt{2p_{s_1}}}{(\sigma_{s_1,m})^2} + \ln \frac{\sqrt{2p_{s_2}}}{(\sigma_{s_2,m})^2} - \ln \frac{\sqrt{2p_s}}{(\sigma_{s,m})^2} \right) \\ & + \frac{3}{2} \ln n + n_s (p_{s_1} \ln p_{s_1} + p_{s_2} \ln p_{s_2} - p_s \ln p_s) - n_s \sum_{m=1}^d (p_{s_1} \ln \sigma_{s_1,m} + p_{s_2} \ln \sigma_{s_2,m} - p_s \ln \sigma_{s,m}) \\ & - \left( \sum_{x \in C(s_1)} \sum_{m=1}^d \frac{(x_m - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} + \sum_{x \in C(s_2)} \sum_{m=1}^d \frac{(x_m - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} - \sum_{x \in C(s)} \sum_{m=1}^d \frac{(x_m - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} \right) - \ln(k+1) \end{aligned} \quad (18)$$

The move is subject to a Metropolis test and will pass the test with a probability of  $\min\{1, e^{-\Delta MsgLen}\}$ .

It is also important that the split proposal function and the combination proposal function be symmetrical. That is, if  $s$  splits into  $s_1$  and  $s_2$ , then the probability of choosing  $s$  for the split attempt should equals the probability of choosing  $s_1$  and  $s_2$  for the combination attempt. To achieve this, we enforce that after the split, **at least** one of the two newly produced components must be the most adjacent component of the other. Here, when we say component  $j1$  is the most adjacent component to  $j2$ , we mean  $j1$  has the shortest Euclidean distance to  $j2$ . Note that this property is not mutual, that is,  $j1$  is most adjacent to  $j2$  does not follow that  $j2$  is

most adjacent to  $jI$ . If none of the two components is most adjacent to the other, the split attempt is unconditionally rejected. How this proposal function enforces symmetry we discuss in greater detail in the section on proving convergence.

#### 4.4.2 Combine

We pick the two candidate components to combine by choosing the first candidate component randomly and enforce its most adjacent component as the second candidate component. The message length change for a combination step is very similar to the reverse as that of the split step.

$$\begin{aligned} \Delta \text{MsgLen} = & -\sum_{m=1}^d \ln(2\sigma_j) + \ln(k-1) + \frac{2d+1}{2} (\ln(\kappa_{n_p}) + 1) - \sum_{m=1}^d \left( \ln \frac{\sqrt{2p_{s1}}}{(\sigma_{s1,m})^2} + \ln \frac{\sqrt{2p_{s2}}}{(\sigma_{s2,m})^2} - \ln \frac{\sqrt{2p_s}}{(\sigma_{s,m})^2} \right) \\ & - \frac{3}{2} \ln n - n_s (p_{s1} \ln p_{s1} + p_{s2} \ln p_{s2} - p_s \ln p_s) + n_s \sum_{m=1}^d (p_{s1} \ln \sigma_{s1,m} + p_{s2} \ln \sigma_{s2,m} - p_s \ln \sigma_{s,m}) \\ & + \left( \sum_{x \in C(s1)} \sum_{m=1}^d \frac{(x_m - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} + \sum_{x \in C(s1)} \sum_{m=1}^d \frac{(x_m - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} - \sum_{x \in C(s)} \sum_{m=1}^d \frac{(x_m - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} \right) - \ln k \end{aligned} \quad (19)$$

The combination move is accepted with the Metropolis probability  $\min\{1, e^{-\Delta \text{MsgLen}}\}$ . In the next section, we will see this is the symmetrical move for the split move.

#### 4.5 Sampling $\theta$

The truncated message length in sampling  $\theta$  from the conditional distribution  $p(\theta | k, p, w, D)$ , by fixing all given parameters as constants, is

$$\text{MsgLen} = \sum_{j=1}^k \sum_{m=1}^d \ln \frac{1}{(\sigma_{j,m})^2} + \sum_{i=1}^n \sum_{m=1}^d \ln \sigma_{w(i),m} + \sum_{i=1}^n \sum_{m=1}^d \frac{(x_{i,m} - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} \quad (20)$$

Because there are  $k$  components and in each component there are  $d$  independently Gaussian distributions, the total number of Gaussian distribution is  $k \times d$ . The message length for the  $j$ th component and  $m$ th variable is:

$$\text{MsgLen}(\sigma_{j,m}, \mu_{w(i),j}) = n \frac{1}{(\sigma_{j,m})^2} + n_j \ln \sigma_{w(i),m} + \sum_{i=1}^{n_j} \frac{(x_{i,m} - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} = (n_j - 2) \ln \sigma_{w(i),m} + \sum_{i=1}^{n_j} \frac{(x_{i,m} - \mu_{w(i),j})^2}{2\sigma_{w(i),j}^2} \quad (21)$$

We sample  $\sigma_{j,m}, \mu_{w(i),j}$  from the full conditional distribution transformed from the above message length calculations. This step is completed for each component and each attribute.

### 5. Proof of Convergence

If the stationary distribution of a Markov chain is to converge towards the posterior distribution, the chain must satisfy the following three conditions [6].

- The chain must be aperiodic, which means there should **not** be any positive integer,  $T$ , that satisfies  $M^t = M^{t+T}$ ,  $\{w^t, p^t, k^t, \theta^t\} = \{w^{t+T}, p^{t+T}, k^{t+T}, \theta^{t+T}\}$  for **all**  $t \geq t_0$ . If there is such a  $T$  then the chain is periodic and the period of the chain would be  $T$ .
- The chain must be irreducible, that is, the transition time takes from any two states  $M(w, p, k, \theta)$  and  $M(w', p', k', \theta')$  must not be infinity.
- The chain must satisfy the detail balance condition for every move, which is given by

$$p(w, p, k, \theta) p(w, p, k, \theta \rightarrow w', p', k', \theta') = p(w', p', k', \theta') p(w', p', k', \theta' \rightarrow w, p, k, \theta) \quad (22)$$

Given any state at time  $t$ , the probability the next state is the same state is greater than zero. Thus it is possible that the chain will stay at one state for arbitrary number of iterations and move to other states. The chain can only repeat the history with a probability less than 1. Therefore, the chain can never get into the deterministic cycle that satisfies the periodicity condition. The chain is also irreducible as the transition probability between any two states is greater than zero thus it will not take infinite time for the transition to take place. We now prove that detail balance holds for the five moves.

In the five moves the parameters are sampled from the full conditional distributions transformed from the message length distribution. In move 1, 2 and 5 where Gibbs sampling is used, it is easy to see the detail balance holds since

$$p(\theta_1)p(\theta_1 \rightarrow \theta_2) = \frac{e^{-m_1}}{Z} \cdot \frac{e^{-m_2}}{Z} = p(\theta_2)p(\theta_2 \rightarrow \theta_1) \quad (23)$$

Here  $\theta$  stands for whichever parameter being sampled in the move and  $Z$  is the normalization factor  $\sum_i e^{-m_i}$ .

In move 3 and 4 where Metropolis algorithm is used,

$$p(\theta_1)p(\theta_1 \rightarrow \theta_2) = \frac{e^{-m_1}}{Z} \min\{e^{-(m_2-m_1)}, 1\} = \frac{e^{-m_2}}{Z} \min\{e^{-(m_1-m_2)}, 1\} = p(\theta_2)p(\theta_2 \rightarrow \theta_1) \quad (24)$$

But it remains to prove that in the Metropolis test, the proposal function is symmetrical. In the death and rebirth move, since these parameters are randomly reinitialized, the probability of a proposal is independent of the current state and is obviously symmetrical. In the split/combine move, the probability of proposing a component as the split candidate equals  $1/k$ . The proposed component  $j$  is split into  $j1$  and  $j2$ . We also enforce that **at least one** component resulted from the split must be the most adjacent of the other. Without losing generality, let  $j2$  be the most adjacent component of  $j1$ . In the combination step, the probability of choosing  $j1$  and  $j2$  equals to the sum probability of choosing  $j1$  (which will automatically choose  $j2$  subsequently) and the probability of choosing  $j2$  and then  $j1$  as its most adjacent component. Let  $p(j1, j2)$  be the probability of proposing components  $j1$  and  $j2$  as the combination candidates and  $p(j)$  be the probability of proposing component  $j$  as the split candidate, then

$$p(j1, j2) = \frac{1}{k+1} + \frac{1}{k+1} \frac{1}{k} = \frac{1}{k} = p(j) \quad (25)$$

The proposing function is symmetrical and the detail balance holds.

In summary, the proposed MML coupled MCMC sampler consists of five moves. The Markov chain grown by the sampler is aperiodic, irreducible and detail balanced therefore its stationary distribution converges to the MML defined posterior distribution of the models.

## 6. MCMC Diagnosis

In this section, we empirically diagnose the MML coupled MCMC sampler to verify the convergence and analyze the mixing rate. A four component two dimension Gaussian mixture (Figure 1) dataset is used for the purpose. We run the chain for 200,000 iterations with the first 50,000 as the burn-in period. We diagnose the convergence by comparing the posterior probabilities of  $k$  among different segments of the chain, each with a size of 50,000 iterations. If the chain has converged then the posterior probabilities should be constant across the segments. The probability of a particular value of  $k$  is the number of iterations the sampler stays in the given model sub-space. The comparison results are presented in Figure 2. Trace plots for  $k$  (Figure 3) indicates efficient mixing across the subspaces of different  $k$ , with split move and the combine move have an acceptance rate of 11.2% and 11.3% respectively.

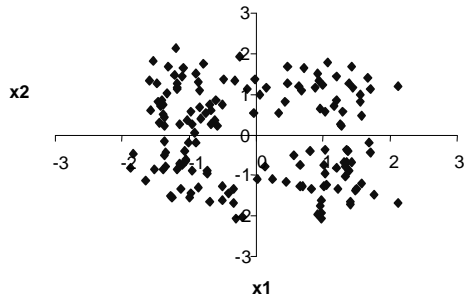


Figure 1. Four components two-dimension Gaussian Mixture

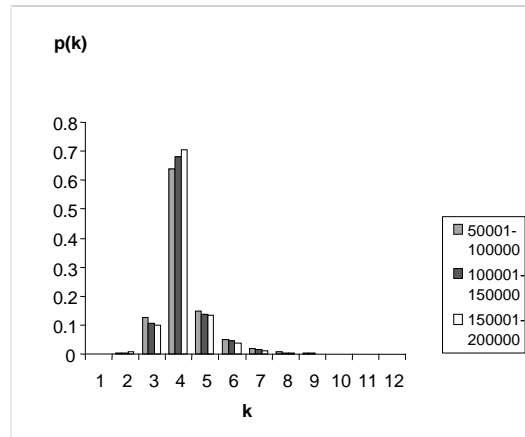


Figure 2. Posterior distribution of  $k$  among different time segments

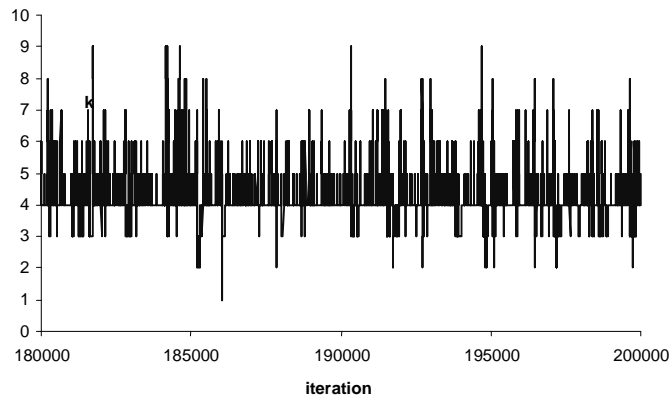


Figure 3. Typical trace plot of  $k$  (iteration 180001-200000)

## 7. Comparisons with Green’s Reversible Jump Sampling

The sampler is capable of jumping across subspaces with different  $k$  values and the posterior distribution of  $k$  is the relative frequency the sampler stays in each subspace. This distribution of  $k$  reflects the belief in how many components should there be in the data. In this section we apply the MML coupled MCMC sampler on three univariate datasets used by Richardson and Green (Figure 4) in their paper on reversible jump sampling [5]. We compare the posterior distribution of  $k$  found by MML-MCMC sampler with that found by the reversible jump sampler. The comparison results are summarized in table 1.

Table 1 comparing the distribution of  $k$  on enzyme, acidity and galaxy datasets

	$k$	1	2	3	4	5	6	7	8	Split	Combine
Enzyme	MML	0.000	0.989	0.010	0.001	0.000	0.000	0.000	0.000	0.46%	0.46%
	RG	0.000	0.024	0.290	0.317	0.206	0.095	0.041	0.017	8%	4%
Acidity	MML	0.003	0.989	0.007	0.001	0.000	0.000	0.000	0.000	1.1%	1.1%
	RG	0.000	0.082	0.244	0.236	0.172	0.118	0.069	0.037	14%	7%
Galaxy	MML	0.060	0.746	0.174	0.018	0.002	0.000	0.000	0.000	9.50%	10.60%
	RG	0.000	0.000	0.061	0.128	0.182	0.199	0.16	0.109	11%	18%

MML-MCMC sampler finds quite different  $k$  distributions from the reversible jump sampler. This great difference, however, does not invalidate either approach. In Richardson's work, they adopt a uniform prior distribution on  $k$ , that is, the complexity of the model is not considered as part of the problem. In minimum message length principle, larger  $k$  values, which imply more complicated model, are penalized as they require more information to encode. However, upon examining the histogram of the datasets visually, the posterior distribution of  $k$  found by MML-MCMC is more consistent with human cognitions. For example, the reversible jump sampler suggests the best  $k$  values for the three problems are 4, 4 and 6 respectively while MML-MCMC suggests 2, 2 and 2. The smaller acceptance rate on split and combine for MML-MCMC in the enzyme and galaxy data does not suggest better mixing rate for reversible jump sampler since it finds much more diffused posterior distribution of  $k$  than MML-MCMC sampler.

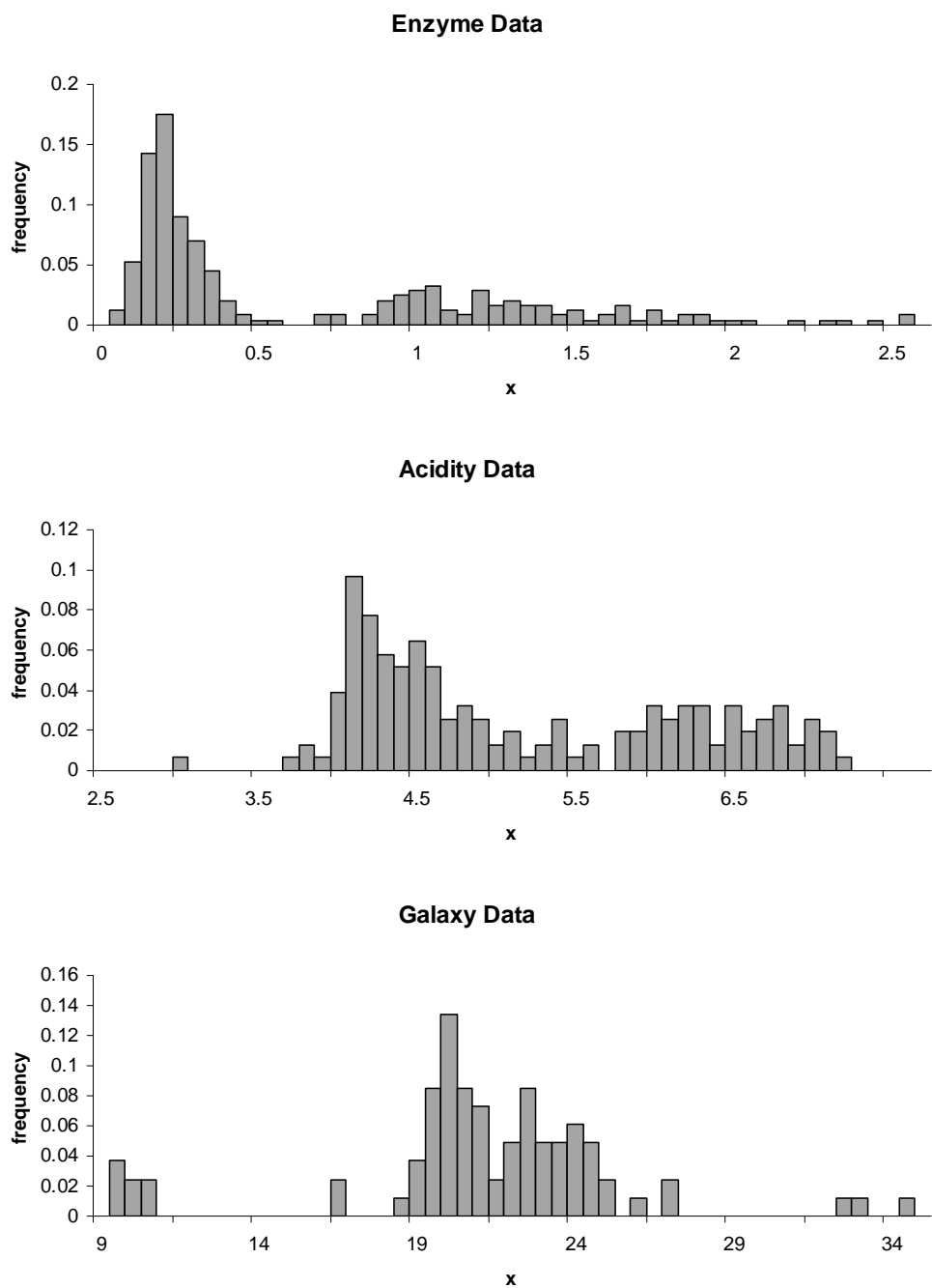


Figure 4. The instance density distribution for Enzyme, Acidity and Galaxy datasets

## 8. Bayesian Model Averaging and Classification

As previously stated, MML-MCMC sampler automatically adopts a prior distribution of the models that penalizes overly complicated models. In this section we empirically show that the MML-MCMC sampler when making predictions via Bayesian model averaging performs significantly better than predictions made from a single model. We show our results on a number benchmark machine learning and statistics dataset. We compare the prediction capability between the single best model found by EM and the Bayesian averaged model calculated from the MML-MCMC sampler. It should be noted that some of these datasets have high dimensions and to our best knowledge, no previous sampler may sample across subspaces for data with high dimension efficiently.

The first dataset we use is the handwritten digit dataset. This data set consists of sixteen attributes and a dependent attribute representing the digit (0 through 9). The sixteen attributes represent 8 x-y coordinates the pen took while writing the digit. To reduce the computational complexity, we reduce the digit recognition problem into five smaller problems. Instead of trying to identify a digit from all 10 possible classes, we paired digits similar in shape into fives groups. The accuracies are given in Table 2. The predictions accuracies are based on 100 trials with 70% training and 30% testing set.

**Table 2 Performance of EM and Bayesian model averaging on digit recognition dataset**

DIGITS	Number of Attributes	EM Error (%)	EM Standard Deviation	BMA Accuracy	BMA Standard Deviation	Error Reduction	Standard Deviation Reduction
0 & 2	16	0.3	0.7	0.1	0.5	67%	29%
1 & 7	16	20.1	4.4	7.3	3.2	64%	27%
3 & 8	16	6.3	3.0	3.5	2.7	44%	10%
4 & 9	16	12.3	13.6	5.3	2.5	57%	82%
5 & 6	16	18.4	4.6	0.3	0.9	98%	80%

We further test the prediction performance via another six benchmark datasets in UCI machine learning repository and the results are show in Table 3.

**Table 3 Performance of EM and Bayesian model averaging on UCI datasets**

Datasets	Number of Attributes	EM Error (%)	EM Standard Deviation	BMA Error (%)	BMA Standard Deviation	Error Reduction	Standard Deviation Reduction
Wine	13	5.4	3.0	4.0	2.6	25.9%	13.3%
Iris	4	8.7	5.2	4.8	3.2	44.8%	38.5%
Pima	8	35	2.6	29.3	2.6	16.3%	0.0%
Glass	9	51.6	6.3	45.7	6.5	11.4%	-3.2%
Diabete	3	15.7	6.1	12.4	4.5	21.0%	26.2%
Shuttle	9	20.2	5.3	10.3	3.3	49.0%	37.7%

We see that the Bayesian averaged model from the MML-MCMC sampler makes much better predictions, both in terms of accuracy and variance. The removal of the model uncertainty occurs at two levels. Not only is the uncertainty of  $k$  value removed, but also the model uncertainty for a given  $k$ . These accuracies might not

be the highest among all possible classifiers available, such as C4.5 and neural network. However, we expect the Bayesian model averaged C4.5 performs better than plain C4.5 and the Bayesian model averaged neural network performs better than a plain neural network.

## 9. Conclusion

Minimum message length principle evaluates the quality of models by the information length to encode both the model and data. The different message lengths associated with different models constitutes a message length distribution which can be conveniently transformed into the posterior distribution or full conditional distributions. This allows building MML-MCMC samplers that are easier to mathematically and pragmatically work with. Furthermore, it inherits the benefit from the MML principle that enables computable connection between model complexity and the prior probability of the model. With all these conveniences and a newly proposed sampling algorithm with five moves, we introduced an MML-MCMC sampler for the mixture component models. The sampler can sample across subspaces with different components with good mixing rate even for high dimensional data. Also, it finds posterior distribution of  $k$  that is more consistent with human cognition. Thus the sampler can reliably estimate  $k$  on high dimensional data where estimation by visualization becomes less obvious to human. Moreover, such a sampler allows making Bayesian model averaging predictions which are empirically verified to be more accurate than the single best models.

Pragmatically, the work motivates the implementation of MML-MCMC sampler for other model families such as Hidden Markov Model (HMM) or a C4.5 decision tree, making inference by averaging the inference from all models in each model space, or further, in both model spaces together. MML-MCMC makes sampling across fundamentally different model spaces possible since the posterior is calculated from a universally computable metric, the message length. Such a sampler is a long-term goal of this work. The BMA predictor in this situation should also achieve improved accuracy and reduced variance.

Philosophically, the work justifies the consistency between Occam's razor and human cognition in a statistical light. By assigning smaller prior probability to more complicated models, we have obtained estimation on the number of components that is more interpretable to a human. This automatic prior obtained from the model complexity, now becomes the prior in the literal sense. That is, it is not acquired via repeated exposure to experience, but rather preexists in the learning faculty itself [11].

---

## References

- [1] C.S. Wallace, P.R. Freeman.: *Estimation and Inference by Compact Encoding*, Journal of the Royal Statistical Society. Series B (Methodological) 49 (1987) 240-265
- [2] C.S. Wallace and D.M. Boulton, *An Information Measure for Classification*, Computer Journal, 11:185-195, 1968
- [3] R.A. Baxter and J.J. Oliver. *Finding Overlapping Components with MML*, Statistics and Computing, 2000, 10, pp5-16.
- [4] J.J. Oliver and R.A. Baxter, *MML and Bayesianism: similarities and differences*, 1994, Dept. of Computer Science, Monash University, Clayton, Victoria 3168, Australia, Technical Report TR 206
- [5] S. Richardson and P. Green. *On Bayesian analysis of mixtures with an unknown number of components*, with discussion, Journal of the Royal Statistical Society, B, 59, 731-792 (1997).
- [6] W. Gilks, S. Richardson, and D. Spiegelhalter, *Markov Chain Monte Carlo in Practice*. Interdisciplinary Statistics. Chapman and Hall, 1996.
- [7] J.H. Conway and N.J.A Sloane, *Sphere Packings, Lattices and Groups*. Springer-Verlag, London, 1988.
- [8] A. R. Barron and T. M. Cover, *Minimum complexity density estimation*, IEEE Trans. Inform. Theory, vol. 37, pp. 1034-1054, July 1991.
- [9] T. Mitchell, *Machine Learning*, McGraw-Hill, 1997.

- [10] R. Neal, *Probabilistic Inference Using Markov Chain Monte Carlo Method*, Technical Report CRG-TR-93-1, Department of Computer Science University of Toronto, 1993.
- [11] I. Kant, *Critique of Pure Reason*, Cambridge University Press, 1998.
- [12] I. Davidson, K. Yin, *Message Length Estimators, Probabilistic Sampling and Optimal Prediction*, DIMACS Workshop on Complexity and Inference, Li, Vitanyi and Hansen, 2003.
- [13] C.E. Rasmussen, *The Infinite Gaussian Mixture Model*, in Advances in Neural Information Processing Systems, 14, 2000.