Non-parametric clustering over user features and latent behavioral functions with dual-view mixture models

Alberto Lumbreras, Julien Velcin, Marie Guégan & Bertrand Jouve

Computational Statistics

ISSN 0943-4062
Volume 32
Number 1

Comput Stat (2017) 32:145-177
DOI 10.1007/s00180-016-0668-0

Springer

Comput Stat (2017) 32:145–177
DOI 10.1007/s00180-016-0668-0

ORIGINAL PAPER

Non-parametric clustering over user features and latent behavioral functions with dual-view mixture models

Alberto Lumbreras[1] · **Julien Velcin**[2] · **Marie Guégan**[1] · **Bertrand Jouve**[3,4]

Received: 27 May 2015 / Accepted: 15 June 2016 / Published online: 1 July 2016

Abstract We present a dual-view mixture model to cluster users based on their features and latent behavioral functions. Every component of the mixture model represents a probability density over a feature view for observed user attributes and a behavior view for latent behavioral functions that are indirectly observed through user actions or behaviors. Our task is to infer the groups of users as well as their latent behavioral functions. We also propose a non-parametric version based on a Dirichlet Process to automatically infer the number of clusters. We test the properties and performance of the model on a synthetic dataset that represents the participation of users in the threads of an online forum. Experiments show that dual-view models outperform single-view ones when one of the views lacks information.

✉ Alberto Lumbreras
alberto.lumbreras@technicolor.com

Julien Velcin
julien.velcin@univ-lyon2.fr

Marie Guégan
marie.guegan@technicolor.com

Bertrand Jouve
jouve@univ-tlse2.fr

1 Technicolor, 975 Avenue des Champs Blancs, 35576 Cesson-Sévigné, France

2 Laboratoire ERIC, Université de Lyon, 5, Avenue Pierre Mendès France, 69676 Bron, France

3 FRAMESPA - UMR 5136, CNRS, Université de Toulouse, 5 Allée Antonio Machado, 31058 Toulouse, Cedex 9, France

4 IMT - UMR 5219, CNRS, Université de Toulouse, 118 Route de Narbonne, 31062 Toulouse, Cedex 9, France

Keywords Multi-view clustering · Model-based clustering · Dirichlet Process (DP) · Chinese Restaurant Process (CRP)

1 Introduction

We consider the problem of clustering users over both their observed features and their latent behavioral functions. The goal is to infer the behavioral function of each user and a clustering of users that takes into account both their features and their behavioral functions. In our model, users in the same cluster are assumed to have similar features *and* behavioral functions, and thus the inference of the clusters depends on the inference of the behavioral functions, and vice versa.

Latent behavioral functions are used to model individual behaviors such as pairwise choices over products, reactions to medical treatments or user activities in online discussions. The inclusion of latent behavioral functions in clustering methods has several potential applications. On the one hand, it allows richer clusterings in settings where users, besides being described through feature vectors, also perform some observable actions that can be modelled as the output of a latent function. On the other hand, by assuming that users in the same cluster share similar latent functions, it may leverage inference of these functions. In the case, for instance, of recommender systems, this may be used to alleviate the *cold-start problem*—the fact that we cannot infer user preferences until they have interacted with the system for a while—if we have a set of features describing user profiles. In that context, a user with low activity will be given the same cluster as those users with similar features. Then, the inference of its behavioral function (e.g.: its preferences) will be based on the behavioral functions of the users in the same cluster.

One of the difficulties in dealing with latent behavioral functions is that, since these functions are latent, they are not representable in a feature-like way and therefore traditional clustering methods are not directly applicable. Our approach is to think of features and behaviors as two different *views* or representations of users, and to find the partition that is most consensual between the different views. In this sense, this is a multi-view clustering problem (Bickel and Scheffer 2004). However, the clustering in one of the views depends on latent variables that need to be inferred. In this regard, it has similarities to Latent Dirichlet Allocation when used for clustering (e.g., cluster together documents that belong to the same latent topics).

Multi-view clustering ranges from Kumar et al. (2011), which finds a consensual partition by co-training (Blum and Mitchell 1998), to Greene and Pádraig (2009), which proposes a two-step multi-view clustering that allows both consensual groups and groups that only exist in one of the views. In Niu et al. (2012), the multi-view approach is presented as the problem of finding multiple cluster solutions for a single description of features.

In this paper, we extend the idea of multi-view clustering to deal with cases where one of the views comprises latent functions that are only indirectly observed through their outputs. The proposed method consists of a dual-view mixture model where every component represents two probability densities: one over the space of features and the other over the space of latent behavioral functions. The model allows to infer both the

clusters and the latent functions. Moreover, the inference of the latent functions allows to make predictions on future user behaviors. The main assumption is that users in the same cluster share both similar features and similar latent functions and that users with similar features and behaviors are in the same cluster. Under this assumption, we show that this dual-view model requires less examples than single-view models to make good inferences.

The idea of using similarities in one view to enhance inference in the other view is not new. In bioinformatics, Eisen et al. (1998) found evidence suggesting that genes with similar DNA microarray expression data may have similar functions. Brown et al. (2000) exploit this evidence to train a Support Vector Machine (SVM) for each functional class and predict whether an unclassified gene belongs to that class given its expression data as an input. Pavlidis et al. (2002) extend the method of Brown et al. by also exploiting evidence that similar phylogenetic profiles between genes suggested a same functional class as well (Pellegrini et al. 1999). Pavlidis et al. propose a multi-view SVM method that uses both types of gene data as input.

More recently, Cheng et al. (2014) applied multi-view techniques to predict user labels in social networks such as LinkedIn (e.g., engineer, professor) or IMdB (e.g., directors, actors). Their method lies in the maximization of an objective function with a co-regularization that penalizes predictions of different labels for users that are similar either in terms of profile features or in terms of graph connectivity.

In the context of preference learning, Bonilla et al. (2010) also work with the assumption that similar users may have similar preferences, and models this assumption via a Gaussian Process prior over user utility functions. This prior favors utility functions that account for user similarity and item similarity. To alleviate the computational cost of this model, Abbasnejad et al. (2013) propose an infinite mixture of Gaussian Processes that generates utility functions for groups of users assuming that users in each community share one single utility function. The main difference between our model and Abbasnejad et al. (2013) in that their model clusters users only based on their utility functions, while ours considers user features as well. In short, ours is a multi-view clustering model that also infers latent behavioral functions, while theirs is a single-view model focused on the inference of latent functions.

The paper is structured as follows: first, we briefly recall mixture models. Second, we present our model as an extension of classic mixture models. The description of the model ends up with a generalization to an infinite number of clusters, which makes the model non-parametric. We finally describe an application to cluster users in online forums and end the paper with experiments on synthetic data to demonstrate the properties of the model.

2 Model description

In this section, we introduce our model through three sequential steps. First, we start with a simple mixture model. Second, we extend the mixture model to build a *dual-view* mixture model. And third, we extend the *dual-view* mixture model so that the number of clusters is automatically inferred.

2.1 Mixture models

When a set of observations $x_1, x_2, \ldots, x_n$ cannot be properly fitted by a single distribution, we may get a better fit by considering that different subsets of observations come from different component distributions. Then, instead of a unique set of parameters $\boldsymbol{\theta}$ of a single distribution, we need to infer K sets of parameters $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_K$ of K components and the assignments $z_1, \ldots, z_n$ of individual observations to one of these components. The model can be expressed as follows:

$$\begin{aligned} x_i | z_i, \boldsymbol{\theta}_{z_i} &\sim F(\boldsymbol{\theta}_{z_i}) \\ z_i &\sim \text{Discrete}(\boldsymbol{\pi}) \end{aligned} \tag{1}$$

where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$ contains the probability of belonging to each component and F is the likelihood function over the observations. In Bayesian settings it is common to add priors over these parameters, resulting in a model such as:

$$\begin{aligned} x_i | z_i, \boldsymbol{\theta}_{z_i} &\sim F(\boldsymbol{\theta}_{z_i}) \\ \boldsymbol{\theta}_j &\sim G_0 \\ z_i | \boldsymbol{\pi} &\sim \text{Discrete}(\boldsymbol{\pi}) \\ \boldsymbol{\pi} &\sim \text{Dirichlet}(\alpha) \end{aligned} \tag{2}$$

where G_0 is the *base distribution* and α the *concentration parameter*. Mixture models are mostly used for *density estimation*. Nonetheless, inference over $\mathbf{z}$ allows to use them as *clustering* methods. In this case, every component is often associated to a cluster.

2.2 Dual-view mixture model

In this section, we present an extension of mixture models to account both for features and latent behavioral functions. We denote by *behavioral functions* any function which, if known, can be used to predict the behavior of a user in a given situation. In the context of preference learning (Bonilla et al. 2010; Abbasnejad et al. 2013) or recommender systems (Cheung et al. 2004), behavioral functions may indicate hidden preference patterns, such as utility functions over the items, and the observed behavior may be a list of pairwise choices or ratings. In the context of online forums, behavioral functions may indicate the reaction of a user to a certain type of post and the observed behavior may be the set of replies to different posts. In general, behavioral functions are linked to observed behaviors through a likelihood function $p(y|f)$ where y represents an observation and f the latent behavioral function.

Let $\mathbf{a}_u$ be the set of (observed) features of user u. Let f_u be a (latent) function of user u. Let y_u be the (observed) outcome of f_u. By slightly adapting the notation from last section we can describe the variables of our dual model as follows:

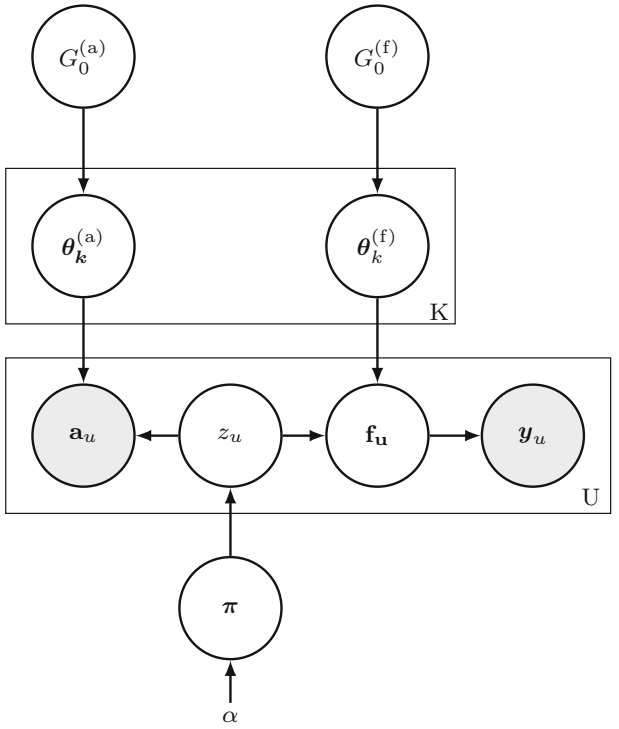

Fig. 1 Graphical model of the generative process for U users and K clusters. *Shaded circles* represent observations and *white circles* represent latent variables. Views are connected through the latent assignments $\mathbf{z}$. A user u draws a feature vector $\mathbf{a}_u$ and a behavior $\mathbf{f}_u$ from the cluster indicated by z_u (the u-th element of $\mathbf{z}$)

$$
\begin{aligned}
\mathbf{a}_u | z_u, \boldsymbol{\theta}^{(a)}_{z_u} &\sim F^{(a)}(\boldsymbol{\theta}^{(a)}_{z_u}) \\
f_u | z_u, \boldsymbol{\theta}^{(f)}_{z_u} &\sim F^{(f)}(\boldsymbol{\theta}^{(f)}_{z_u}) \\
y_u | f_u &\sim p(y_u | f_u) \\
\boldsymbol{\theta}^{(a)}_j &\sim G_0^{(a)} \\
\boldsymbol{\theta}^{(f)}_j &\sim G_0^{(f)} \\
z_u | \boldsymbol{\pi} &\sim \text{Discrete}(\boldsymbol{\pi}) \\
\boldsymbol{\pi} &\sim \text{Dirichlet}(\alpha)
\end{aligned}
\tag{3}
$$

where we use the superindex (a) for elements in the *feature view* and the superindex (f) for elements in the latent functions view, henceforth *behavior view*. Otherwise, the structures are similar except for y_u, which represents the set of observed behaviors for user u. The corresponding Probabilistic Graphical Model is shown in Fig. 1.

Every component has two distributions: one for features and one for latent behavioral functions. Latent behavioral functions are not directly observable, but they may

be inferred through some observations if we have a likelihood function of observations given the latent functions.

The model can also be expressed in terms of a generative process:

- For every component k:
  - Draw feature and function parameters from their base distributions $\boldsymbol{\theta}_k^{(\mathrm{a})} \sim G_0^{(\mathrm{a})}$ and $\boldsymbol{\theta}_k^{(\mathrm{f})} \sim G_0^{(\mathrm{f})}$.
- Draw the mixture proportions $\boldsymbol{\pi} \sim \mathrm{Dirichlet}(\alpha)$.
- For every user u:
  - Draw a component assignment $z_u \sim \mathrm{Multinomial}(\boldsymbol{\pi})$.
  - Draw user features $\mathbf{a}_u \sim F^{(\mathrm{a})}(\boldsymbol{\theta}_{z_u}^{(\mathrm{a})})$.
  - Draw a user latent function $f_u \sim F^{(\mathrm{f})}(\boldsymbol{\theta}_{z_u}^{(\mathrm{f})})$.
  - Draw a set of observed behaviors $y_u \sim p(y_u|f_u)$.

Left and right branches of Fig. 1 correspond to the *feature view* and the *behavior view*, respectively. Note that every component contains two sets of parameters, one for features and one for behaviors, so that the two views can be generated from the same component. This encodes our prior assumption that users who belong to the same cluster should be similar in both views.

Given the user assignments $\mathbf{z}$, variables in one view are conditionally independent from variables in the other view. That means their inferences can be considered separately. However, inference of $\mathbf{z}$ uses information from both views. The conditional probability of $\mathbf{z}$ given all the other variables is proportional to the product of its prior and the likelihood of both views:

$$p(\mathbf{z}|\cdot) \propto p(\mathbf{z}|\boldsymbol{\pi})\, p(\mathbf{a}|\boldsymbol{\theta}^{(\mathrm{a})}, \mathbf{z})\, p(\mathbf{f}|\boldsymbol{\theta}^{(\mathrm{f})}, \mathbf{z}) \tag{4}$$

The information given by each view is conveyed through the likelihood factors $p(\mathbf{a}|\boldsymbol{\theta}^{(\mathrm{a})}, \mathbf{z})$ and $p(\mathbf{f}|\boldsymbol{\theta}^{(\mathrm{f})}, \mathbf{z})$. The ratio between the conditional probability of a partition $\mathbf{z}$ and the conditional probability of a partition $\mathbf{z}'$ is:

$$\frac{p(\mathbf{z}|\cdot)}{p(\mathbf{z}'|\cdot)} = \frac{p(\mathbf{z}|\boldsymbol{\pi})}{p(\mathbf{z}'|\boldsymbol{\pi})} \frac{p(\mathbf{a}|\boldsymbol{\theta}^{(\mathrm{a})}, \mathbf{z})}{p(\mathbf{a}|\boldsymbol{\theta}^{(\mathrm{a})}, \mathbf{z}')} \frac{p(\mathbf{f}|\boldsymbol{\theta}^{(\mathrm{f})}, \mathbf{z})}{p(\mathbf{f}|\boldsymbol{\theta}^{(\mathrm{f})}, \mathbf{z}')} \tag{5}$$

where the contribution of each view depends on how much more likely $\mathbf{z}$ is over the other assignments in that view. An extreme case would be a uniform likelihood in one of the views, meaning that all partitions $\mathbf{z}$ are equally likely. In that case, the other view leads the inference.

The two views provide reciprocal feedback to each other through $\mathbf{z}$. This means that if one view is more confident about a given $\mathbf{z}$, it will not only have more influence on $\mathbf{z}$ but also it will force the other view to re-consider its beliefs and adapt its latent parameters to fit the suggested $\mathbf{z}$.

Note also that inference of latent behavioral functions may be used for prediction of future behaviors.

2.3 Infinite number of clusters

So far, we have considered the number of components K to be known. Nonetheless, if we let $K \to \infty$ and marginalize over the mixture weights $\boldsymbol{\pi}$, it becomes a non-parametric model with a Dirichlet Process (DP) based on a Chinese Restaurant Process (CRP) prior over the user assignments, which automatically infers the number of *active* of components (see the "Appendix" for the full derivation). Since we integrate out $\boldsymbol{\pi}$, user assignments are not independent anymore. Instead, the probability of a user u to be assigned to a non-empty (active) component k, given the assignments of all other users $\mathbf{z}_{-u}$, is:

$$p(z_u = k|\mathbf{z}_{-u}) \propto n_k \qquad \text{for } k = 1, \ldots, c \tag{6}$$

where c denotes the number of non-empty components and n_k is the number of users already assigned to the k-th component. The probability of assigning a user u to an empty (non-active) component, that would be labelled as $c + 1$, is:

$$p(z_u = k|\mathbf{z}_{-u}) \propto \alpha \qquad \text{for } k = c + 1 \tag{7}$$

These two equations reflect a generative process that assigns users to clusters in *a rich get richer* manner. The more users in a component, the more attractive this component becomes. Empty components also have a chance of being filled. Despite the appearance of these equations, the idea behind the inference of $\mathbf{z}$ remains the same. The only differences between the finite and the infinite cases are the number of components and the probabilities to be assigned to each component.

3 Application to role detection in online forums

The above model provides a general framework that can be adapted to many scenarios. In this section, we apply our model to the clustering of users in online forums. Clustering users in online communities may be used for latent *role detection*. Although clustering based on user features may provide interesting insights, we think that the notion of *role* should include information that allows to predict behaviors. After all, this is what roles are useful for. We expect that a person holding a role behaves according to their role.

Let us specify the two views of our model for this scenario, given U users who participate in a forum composed of T discussion threads. For the feature view, we describe every user through a feature vector $\mathbf{a}_u = (a_{u1}, a_{u2}, \ldots, a_{uD})^T$ that will typically contain features such as centrality metrics or number of posts. For the behavior view, we define a latent behavioral function that we call *catalytic power* and denote by b_u, which represents the ability of a user to promote long discussions; we refer to $\mathbf{b}$ as the vector of all user catalytic powers. Let the *participation vector* of the discussion thread $\mathbf{p}_t = (p_{1t}, \ldots, p_{Ut})^T$ be a binary array indicating which users participated among the first m posts of the thread t. Assuming that the dynamic of a discussion is strongly conditioned by the first participants, we model the final length of a thread y_t:

$$y_t \sim \mathcal{N}(\mathbf{p}_t^T \mathbf{b}, s_{\mathrm{y}}^{-1})$$

where $\mathbf{p}_t^T \mathbf{b}$ is the cumulated catalytic power of users who participated in its first m posts and s_{y} represents the precision (inverse of the variance) of the unknown level of noise.

If the assumption that there exist groups of users sharing similar features and similar catalytic power holds, then our model will not only find a clustering based on features and behaviors (catalytic powers), but it will also exploit feature information to infer catalytic powers and, vice versa, the inferred catalytic powers will be treated by the model as an extra feature dimension.

Note that, unlike the model presented in Fig. 1, the observed output y_t is common to all users who participated in the first m posts of thread t. Moreover, y_t depends on the observed participations $\mathbf{p_t}$. We also defined the noise factor s_{y} which was not explicitly present in the general model. The graphical model would be similar to that of Fig. 1 but with the thread lengths $\mathbf{y}$, the participation matrix $\mathbf{P}$ and the noise s_{y} out of the users plate. In the remaining of this section we provide more details about the components of the two views.

3.1 Feature view

In this section we specify the component parameters $\boldsymbol{\theta}_k^{(\mathrm{a})}$ and the base distribution $G_0^{(\mathrm{a})}$ of the feature view. Our choice of priors follows that of the Infinite Gaussian Mixture Model (IGMM) as described by Rasmussen (2000) and Görür and Rasmussen (2010).

The feature vectors are assumed to come from a mixture of Gaussian distributions:

$$\boldsymbol{a}_u \sim \mathcal{N}\left(\boldsymbol{\mu}_{z_u}^{(\mathrm{a})}, \left(\mathbf{S}_{z_u}^{(\mathrm{a})}\right)^{-1}\right) \tag{8}$$

where the mean $\boldsymbol{\mu}_{z_u}^{(\mathrm{a})}$ and the precision $\mathbf{S}_{z_u}^{(\mathrm{a})}$ are component parameters common to all users assigned to the same component. The component parameters are given Normal and Wishart priors:

$$\boldsymbol{\mu}_k^{(\mathrm{a})} \sim \mathcal{N}\left(\boldsymbol{\mu}_0^{(\mathrm{a})}, \left(\mathbf{R}_0^{(\mathrm{a})}\right)^{-1}\right) \tag{9}$$

$$\mathbf{S}_k^{(\mathrm{a})} \sim \mathcal{W}\left(\beta_0^{(\mathrm{a})}, \left(\beta_0^{(\mathrm{a})} \mathbf{W}_0^{(\mathrm{a})}\right)^{-1}\right) \tag{10}$$

where the mean $\boldsymbol{\mu}_0^{(\mathrm{a})}$, the precision $\mathbf{R}_0^{(\mathrm{a})}$, the covariance $\mathbf{W}_0^{(\mathrm{a})}$, and the degrees of freedom $\beta_0^{(\mathrm{a})}$ are hyperparameters common to all components. The hyperparameters themselves are given non-informative priors centered at the observed data

$$\boldsymbol{\mu}_0^{(\mathrm{a})} \sim \mathcal{N}(\boldsymbol{\mu_a}, \boldsymbol{\Sigma_a}) \tag{11}$$

$$\mathbf{R}_0^{(\mathrm{a})} \sim \mathcal{W}\left(D, (D\boldsymbol{\Sigma_a})^{-1}\right) \tag{12}$$

$$\mathbf{W}_0^{(\mathrm{a})} \sim \mathcal{W}\left(D, \frac{1}{D}\boldsymbol{\Sigma}_{\boldsymbol{a}}\right) \tag{13}$$

$$\frac{1}{\beta_0^{(\mathrm{a})} - D + 1} \sim \mathcal{G}\left(1, \frac{1}{D}\right) \tag{14}$$

where $\boldsymbol{\mu}_{\boldsymbol{a}}$ and $\boldsymbol{\Sigma}_{\boldsymbol{a}}$ are the mean and covariance of all the features vectors and $\mathcal{G}$ is the Gamma distribution. Note that the expectation of a random matrix drawn from a Wishart distribution $X \sim \mathcal{W}(\upsilon, W)$ is $\mathbb{E}[\mathbf{X}] = \upsilon\mathbf{W}$. Our parametrization of the Gamma distribution corresponds to a one-dimensional Wishart. Its density function is therefore given by $\mathcal{G}(\alpha, \beta) \propto x^{\alpha/2-1}\exp(-\frac{x}{2\beta})$ and the expectation of a random scalar drawn from a Gamma distribution $x \sim \mathcal{G}(\alpha, \beta)$ is $\mathbb{E}[x] = \alpha\beta$.

As pointed out in Görür and Rasmussen (2010), this choice of hyperparameters, which is equivalent to scaling the data and using unit priors, makes the model invariant to translations, rotations, and rescaling of the data. Conjugate priors are chosen whenever possible to make the posteriors analytically accessible. As for $\beta_0^{(\mathrm{a})}$, the prior in Eq. 14 guarantees that the degrees of freedom in the Wishart distribution in Eq. 10 are greater than or equal to $D - 1$. The density $p(\beta_0^{(\mathrm{a})})$ is obtained by a simple transformation of variables (see "Appendix").

3.2 Behavior view

In this section we specify the component parameters $\boldsymbol{\theta}_k^{(\mathrm{f})}$ and the base distribution $G_0^{(\mathrm{f})}$ of the behavior view. Our choice corresponds to a Bayesian linear regression where coefficients are drawn not from a single multivariate Gaussian but from a *mixture* of one-dimensional Gaussians.

The thread lengths are assumed to come from a Gaussian distribution whose mean is determined by the catalytic power of users who participated in the first posts and whose variance represents the unknown level of noise:

$$y_t \sim \mathcal{N}\left(\mathbf{p}_t^T\mathbf{b}, s_{\mathrm{y}}^{-1}\right) \tag{15}$$

where the precision s_{y} is given a Gamma prior centered at the sample precision σ_0^{-2}:

$$s_{\mathrm{y}} \sim \mathcal{G}\left(1, \sigma_0^{-2}\right) \tag{16}$$

The power coefficients b_u come from a mixture of Gaussians:

$$b_u \sim \mathcal{N}\left(\mu_{z_u}^{(\mathrm{f})}, \left(s_{z_u}^{(\mathrm{f})}\right)^{-1}\right) \tag{17}$$

where the mean $\mu_{z_u}^{(\mathrm{f})}$ and the precision $s_{z_u}^{(\mathrm{f})}$ are component parameters common to all users assigned to the same component z_u. The component parameters are given Normal and Gamma priors:

$$\mu_k^{(\mathrm{f})} \sim \mathcal{N}\left(\mu_0^{(\mathrm{f})}, \left(r_0^{(\mathrm{f})}\right)^{-1}\right) \tag{18}$$

$$s_k^{(\mathrm{f})} \sim \mathcal{G}\left(\beta_0^{(\mathrm{f})}, \left(\beta_0^{(\mathrm{f})} w_0^{(\mathrm{f})}\right)^{-1}\right) \tag{19}$$

where the mean $\mu_0^{(\mathrm{f})}$, the precision $r_0^{(\mathrm{f})}$, the variance $w_0^{(\mathrm{f})}$, and the degrees of freedom $\beta_0^{(\mathrm{f})}$ are hyperparameters common to all components. Because the coefficients are not observed, we cannot center the hyperparameters in the data as we did in the feature view. Instead, we use their Maximum Likelihood Estimates, computed as $\hat{\mathbf{b}} = (\mathbf{P}\mathbf{P}^{\mathbf{T}} + \lambda\mathbf{I})^{-1}\mathbf{P}^{\mathbf{T}}\mathbf{y}$, with a regularization parameter $\lambda = 0.01$, where $\mathbf{P}$ is the participation matrix $\mathbf{P} = \{\mathbf{p_1}, \ldots, \mathbf{p_T}\}$. Then the hyperparameters are given non-informative priors centered at $\hat{\mathbf{b}}$:

$$\mu_0^{(\mathrm{f})} \sim \mathcal{N}\left(\mu_{\hat{b}}, \sigma_{\hat{b}}^2\right) \tag{20}$$

$$r_0^{(\mathrm{f})} \sim \mathcal{G}\left(1, \sigma_{\hat{b}}^{-2}\right) \tag{21}$$

$$w_0^{(\mathrm{f})} \sim \mathcal{G}\left(1, \sigma_{\hat{b}}^2\right) \tag{22}$$

$$\frac{1}{\beta_0^{(\mathrm{f})}} \sim \mathcal{G}\,(1, 1) \tag{23}$$

where $\mu_{\hat{b}}$ and $\sigma_{\hat{b}}^2$ are the mean and the variance of the Maximum Likelihood Estimators $\hat{\mathbf{b}}$ of the coefficients. Note that this choice is data-driven and, at the same time, the risk of overfitting is reduced since hyperparameters are high in the model hierarchy.

3.3 Shared parameters

As for the common variables $\mathbf{z}$ and α, $\mathbf{z}$ is given a Chinese Restaurant Process prior:

$$p(z_u = k|\mathbf{z}_{-u}, \alpha) \propto \begin{cases} n_k & \text{for } k = 1, \ldots, c \\ \alpha & \text{for } k = c + 1 \end{cases} \tag{24}$$

where c denotes the number of non-empty components before the assignment of z_u and n_k is the number of users already assigned to the k-th component. The concentration parameter α is given a vague inverse gamma prior:

$$\frac{1}{\alpha} \sim \mathcal{G}(1, 1)$$

4 Inference

The latent parameters of our model can be inferred by using Gibbs sampling, and sequentially taking samples of every variable given the others. Conditional distribu-

tions are detailed in the appendices. A single iteration of the Gibbs sampler goes as follows:

- Sample component parameters $\mathbf{S}_k^{(\mathrm{a})}$, $\boldsymbol{\mu}_k^{(\mathrm{a})}$ conditional on the indicators $\mathbf{z}$ and all the other variables of the two views.
- Sample hyperparameters $\boldsymbol{\mu}_0^{(\mathrm{a})}$, $\mathbf{R}_0^{(\mathrm{a})}$, $\mathbf{W}_0^{(\mathrm{a})}$, $\beta_0^{(\mathrm{a})}$ conditional on the indicators $\mathbf{z}$ and all the other variables of the two views.
- Sample component parameters $s_k^{(\mathrm{f})}$, $\mu_k^{(\mathrm{f})}$ conditional on the indicators $\mathbf{z}$ and all the other variables of the two views.
- Sample hyperparameters $\mu_0^{(\mathrm{f})}$, $r_0^{(\mathrm{f})}$, $w_0^{(\mathrm{f})}$, $\beta_0^{(\mathrm{f})}$ conditional on the indicators $\mathbf{z}$ and all the other variables of the two views.
- Sample coefficients $\mathbf{b}$ conditional on the indicators $\mathbf{z}$ and all the other variables of the two views.
- Sample s_{y} conditional on the indicators $\mathbf{z}$ and all the other variables of the two views.
- Sample indicators $\mathbf{z}$ conditional on all the variables of the two views.

Since we use conjugate priors for almost all the variables, their conditional probabilities given all the other variables are analytically accessible. The degrees of freedom $\beta_0^{(\mathrm{a})}$, $\beta_0^{(\mathrm{f})}$ and the concentration parameter α can be sampled by Adaptive Rejection Sampling (Gilks and Wild 1992), which exploits the log-concavity of $p(\log \beta_0^{(\mathrm{a})}|\cdot)$, $p(\log \beta_0^{(\mathrm{f})}|\cdot)$ and $p(\log \alpha|\cdot)$ (see "Appendix"). As for the sampling of $\mathbf{z}$, the conditional probability of assigning user u to an active component k is proportional to the prior times the likelihoods:

$$p\left(z_u = k|\mathbf{z}_{-\mathbf{u}}, \alpha, \cdot\right) \propto n_k\, p\left(\mathbf{a}_u|\boldsymbol{\mu}_k^{(\mathrm{a})}, \mathbf{S}_k^{(\mathrm{a})}\right) p\left(b_u|\mu_k^{(\mathrm{f})}, s_k^{(\mathrm{f})}\right) \qquad \text{for } k = 1, \ldots, c \tag{25}$$

and for the conditional probability of assigning z_u to a non-active component:

$$\begin{aligned} p\left(z_u = k|\mathbf{z}_{-u}, \alpha, \cdot\right) \propto \alpha \int & p\left(\mathbf{a}_u|\boldsymbol{\mu}_k^{(\mathrm{a})}, \mathbf{S}_k^{(\mathrm{a})}\right) p\left(\boldsymbol{\mu}_k^{(\mathrm{a})}\right) p\left(\mathbf{S}_k^{(\mathrm{a})}\right) \mathrm{d}\boldsymbol{\mu}_k^{(\mathrm{a})} \mathrm{d}\mathbf{S}_k^{(\mathrm{a})} \\ \times & \int p\left(b_u|\mu_k^{(\mathrm{f})}, s_k^{(\mathrm{f})}\right) p\left(\mu_k^{(\mathrm{f})}\right) p\left(s_k^{(\mathrm{f})}\right) \mathrm{d}\mu_k^{(\mathrm{f})} \mathrm{d}s_k^{(\mathrm{f})} \quad \text{for } k = c+1 \end{aligned} \tag{26}$$

Unfortunately, these integrals are not analytically tractable because the product of the factors does not give a familiar distribution. Neal (2000) proposes to create m auxiliary empty components with parameters drawn from the base distribution, and then computing the likelihoods of $\mathbf{a}$ and $\mathbf{b}$ given those parameters. The higher the m, the closer we will be to the real integral and the less autocorrelated the cluster assignments will be. However, the equilibrium distribution of the Markov Chain is exactly correct for any value of m. To speed up the computations, m is usually small. For our experiments, we chose $m = 3$. That is, we generate 3 additional empty tables each one with its own parameters $\boldsymbol{\mu}'$, $\mathbf{S}'$, μ', s'. We also run some experiments with $m = 4$ and $m = 5$, without observing significant differences neither in the clustering

nor in the predictions, while it significantly increased the computational time. See Neal (2000) for a more systematic study on the effect of m.

4.1 Predictive distribution

We are also interested in the ability of the model to predict new thread lengths. The posterior predictive distribution over the length of a new thread is:

$$p(y_*|\mathbf{p}_*,\mathbf{P},\mathbf{y}) = \int_{\boldsymbol{\theta}} p(y_*|\mathbf{p}_*,\boldsymbol{\theta})p(\boldsymbol{\theta}|\mathbf{y},\mathbf{P})\mathrm{d}\boldsymbol{\theta} \tag{27}$$

where $\mathbf{p}_*$ is the participation vector of the new thread, and y_* its predicted length. If we have samples $\boldsymbol{\theta}^{(1)},\ldots,\boldsymbol{\theta}^{(N)}$ from the posterior $p(\boldsymbol{\theta}|\mathbf{y},\mathbf{P})$, we can use them to approximate the predictive distribution:

$$p\left(y_*|\mathbf{p}_*,\mathbf{P},\mathbf{y}\right) \approx \frac{1}{N}\sum_{i=1}^{N} p\left(y_*|\mathbf{p}_*,\boldsymbol{\theta}^{(i)}\right) \tag{28}$$

where $\boldsymbol{\theta}^{(i)}$ are the i-th samples of $\mathbf{b}$ and σ_{y}.

5 Experiments

We generated three scenarios to study in which situations dual-view models outperform single-view ones. The data reproduces the scenario of online forums presented in Sect. 3. In the first scenario, users belong to five clusters and those who belong to the same cluster in one view also share the same cluster in the other view (*agreement between views*). In the second scenario, users belong to five clusters in the behavior view but two of the clusters are completely overlapped in the feature view (*disagreement between views*). In order to reproduce a more realistic clustering structure, in the last scenario user features and coefficients are taken from the *iris dataset*.

We will see in Sect. 5.6 that the main bottleneck of the model is the sampling of coefficients $b_1,\ldots,b_U$ since they are obtained by sampling from a U-dimensional Gaussian distribution that requires, at each Gibbs iteration, inverting a $U \times U$ matrix to get its covariance. This issue would disappear if the inference of the behavioral function parameters for a user were independent from the parameters of the other users. In this paper, we use the *iris* dataset to demonstrate the properties of the model as a whole, without making any statement on the convenience of the presented behavioral functions.

5.1 Compared models

We compared two dual-view models and one single-view model. We call them dual-fixed, dual-DP and single. The `dual-fixed` corresponds to the present model where

Table 1 Compared and related models

	Features	Behaviors	Clusters
Dual-DP	Yes	Yes	∞
Dual-fixed	Yes	Yes	Fixed
Single	Yes	Yes	1
IGMM	Yes	No	∞
GMM	Yes	No	Fixed
–	Yes	No	1
Latent-IGMM	No	Yes	∞
Latent-GMM	No	Yes	Fixed
Single	No	Yes	1

Both single models are the same since if the number of clusters is fixed to one they cannot use the feature view. The row marked as – corresponds to a model that has no interest in this context since it simply finds the Gaussian distribution that best fits the observed features and makes neither clustering nor predictions

the number of clusters is set to the ground truth (five clusters). The `dual-DP` corresponds to the present model where the number of clusters is also inferred (Sect. 2.3). The `single` model corresponds to a Bayesian linear regression over the coefficients **b**, which is equivalent to the behavior view specified in Sect. 3.2 where the number of clusters is set to one (that is, no clusters at all) and therefore there is no information flowing from the feature view to the behavior view; this model can only learn the latent coefficients **b**.

Table 1 presents these three models as well as other related models that appear when blinding the models from one of the views. Note that we left out of the analysis those models that use clustering but are blinded of one view. The first two of these (IGMM and GMM), are regular clustering methods over feature vectors; we discarded them because they do not make inferences on latent behaviors. The last two (we call them latent-IGMM and latent-GMM) are Bayesian linear regressions where coefficients are assumed to come from a mixture of Gaussians; because these are in practice very similar to a simple Bayesian linear regression (they can be seen as Bayesian linear regressions with priors that tend to create groups of coefficients), we chose to benchmark only against the `single` model.

Posterior distributions of parameters are obtained by Gibbs sampling. We used the `coda` package (Plummer et al. 2006, 2015) in R (R Core Team 2016) to examine the traces of the Gibbs sampler. For the convergence diagnostics, we used Geweke's test available in the same package. After some initial runs and examinations of the chains to see how long it took to converge to the stationary distribution for the dual models, we observed that convergence for all the models is usually reached before 5000 samples. Since we run a large number of automatized experiments, we set a conservative number of 30,000 samples for every run, from which the first 15,000 are discarded as burn-in samples. For the first two experiments we initialized our samplers with all users in the same cluster. For the *iris* experiment we used the result of a k-means with 10 clusters over the feature view. We did not systematically benchmark

the two initialisation strategies. Nevertheless, this second strategy is, in general, more convenient in order to arrive to the true target distribution within less iterations.

5.2 Metrics

We used two metrics for evaluation, one for clustering (within dual models) and one for predictions of thread lengths (within the three models).

Metric for clustering Clustering by mixtures models suffers from identifiability. The posterior distributions of $\mathbf{z}$ has $k!$ reflections corresponding to the $k!$ possible relabelling of the k components. Due to this, different MCMC samples of $\mathbf{z}$ may come from different reflections making it hard to average the samples. A common practice is to summarize the pairwise posterior probability matrix of clustering, denoted by $\hat{\pi}$, that indicates the probability of every pair of users to be in the same component (no matter the label of the component). In order to obtain a full clustering $\mathbf{z}$ from $\hat{\pi}$, Dahl (2006) proposes a *least-squares model-based clustering* which consists of choosing as $\mathbf{z}$ the sample $\mathbf{z}^{(i)}$ whose corresponding pairwise matrix has the smaller least-squares distance to $\hat{\pi}$:

$$\mathbf{z}_{LS} = \underset{\mathbf{z}\in\mathbf{z}^{(1)},\ldots,\mathbf{z}^{(N)}}{\arg\min} \sum_{i}^{U}\sum_{j}^{U}(\delta_{i,j}(\mathbf{z}) - \hat{\pi})^2 \tag{29}$$

where $\delta_{i,j}(\mathbf{z})$ indicates whether i and j share the same component in $\mathbf{z}$. Finally, to assess the quality of the proposed clustering $\mathbf{z}_{LS}$ we use the *Adjusted Rand Index*, a measure of pair agreements between the true and the estimated clusterings.

Metric for predictions For predictions, we use the *negative loglikelihood*, which measures how likely the lengths are according to the predictive posterior:

$$p\left(y^{\text{test}}|\mathbf{p}_t^{\text{test}}, \mathbf{P}^{\text{train}}, \mathbf{y}^{\text{train}}\right) \tag{30}$$

and that can be approximated from Eq. 28. Negative loglikelihoods are computed on test sets of 100 threads.

5.3 Agreement between views

To set up the first scenario, for a set of U users and five clusters we generated an assignment z_u to one of the clusters so that the same number of users is assigned to every cluster. Once all assignments $\mathbf{z}$ had been generated, we generated the data for each of the views. For the feature view, every user was given a two-dimensional feature vector $\mathbf{a}_u = (a_{u_1}, a_{u_2})^T$ drawn independently from:

$$\mathbf{a}_u \sim \mathcal{N}\left(\boldsymbol{\mu}_{z_u}, \Sigma_a\right) \tag{31}$$

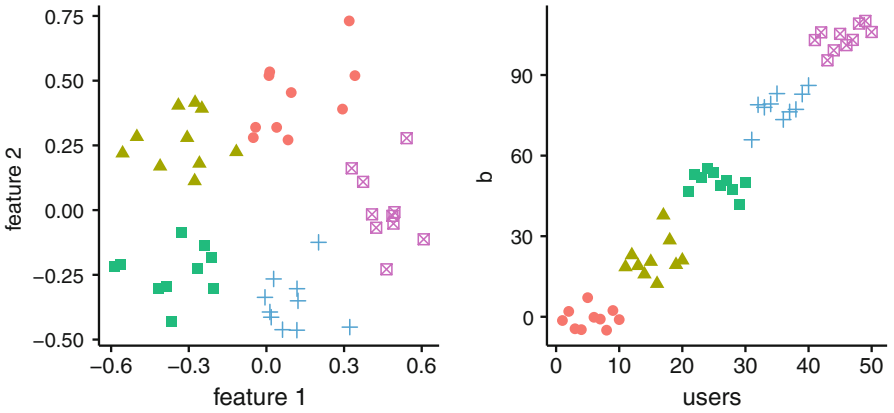

Fig. 2 Dataset for agreement between the views. User features (*left*) and user coefficients (*right*). Every group (*shape*) of users has a well differentiated set of features and coefficients

where $\mu_{z_u} = \left(\cos(2\pi \frac{z_u}{5}), \sin(2\pi \frac{z_u}{5})\right)^T$ for $z_u = 1, \ldots, 5$ (see Fig. 2). For the behavior view, every user was given a coefficient drawn independently from:

$$b_u \sim \mathcal{N}(-50 + 25z_u, \sigma^2) \qquad \text{for } z_u = 1, \ldots, 5 \tag{32}$$

where coefficients for users in the same cluster are generated from the same Normal distribution and the means of these distributions are equidistantly spread in a [−200,200] interval (see Fig. 2). To simulate users participations in a forum we generated, for each user u, a binary vector $\mathbf{p}_u = (p_{u1}, \ldots, p_{uT})^T$ of length T that represents in which threads the user participated among the first m posts. We supposed each user participated in average in half the threads.

$$p_{ut} \sim \text{Bernoulli}(0.5) \tag{33}$$

Finally, we assumed that the final length of a thread is a linear combination of the coefficients of users who participated among the first posts:

$$y_t \sim \mathcal{N}(\mathbf{p}_t^T \mathbf{b}, \sigma_y) \tag{34}$$

If both views agree and there is enough data for both of them, we expect dual models to find the true clusters and true latent coefficients, and the single model to find also the true latent coefficients. In this case, the feature view brings no competitive advantage when there is enough information in the behavior view (and conversely, dual models should not outperform simple IGMM and GMM for the clustering task since there is enough information in the feature view).

On the contrary, when one of the views lacks information, then dual-view models should outperform single-view ones. In our model, the lack of information may come either from having too few threads to learn from or from having a high ratio of users versus threads since we have too many user behavior coefficients to learn.

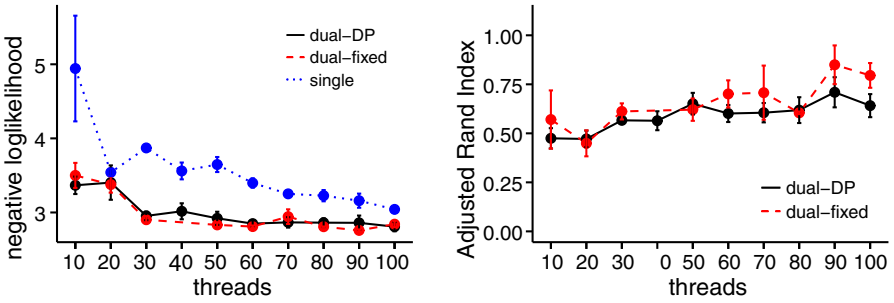

Fig. 3 Results for agreement between the views. Comparison of models under different threads/users ratios (50 users and variable number of threads). Means and standard errors over 5 runs

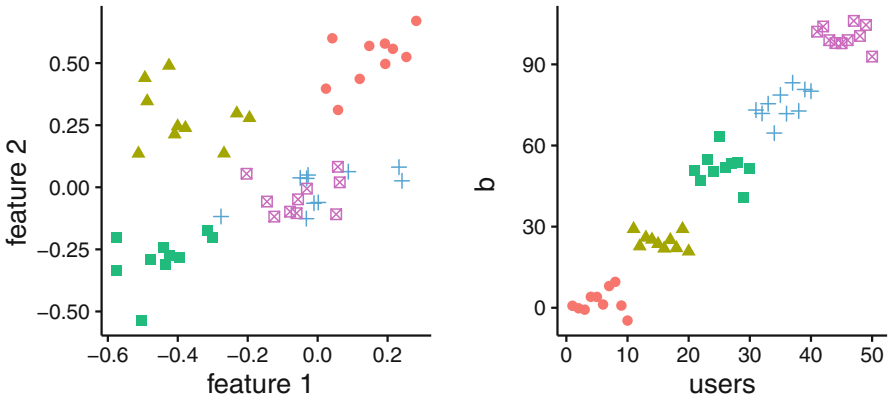

Fig. 4 Dataset for disagreement between the views. User features (*left*) and user coefficients (*right*). Two of the groups (*shapes*) of users have similar features but different coefficients

Figure 3 shows how increasing the threads vs users ratio affects the accuracy of each model. When the number of threads is too low with respect to the number of users neither view has enough information and thus the three models make bad predictions the inference is difficult for the three models. Yet, dual-view models need less threads than the single model to make good inferences. When the number of threads is high enough, the three models tend to converge.

The number of users and threads in the experiments ranges from 10 to 100 users and from 10 to 100 threads. We avoided larger numbers to prevent the experiments from taking too long. 30,000 iterations of the Gibbs sampler described in Sect. 4 for 50 users and 100 threads take around 3 h in a Pentium Intel Core i7-4810MQ @2.80GHz. Nevertheless, the ratio users/threads remains realistic. In the real forums that we analyzed from www.reddit.com a common ratio is 1/10 for a window of 1 month.

5.4 Disagreement between views

If each view suggests a different clustering **z**, dual models should find a consensus between them (recall Eq. 4). We generated a new dataset (Fig. 4) where there are

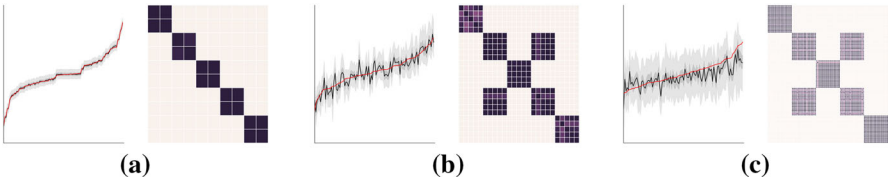

Fig. 5 Posteriors for DP-dual when the two views see a different number of clusters. **a** 10 users and 100 threads. **b** 25 users and 25 threads. **c** 100 users and 10 threads. Figures on the *left*: examples of posterior distributions of thread length predictions over 100 test threads with 50 and 95 % credible intervals in test set with 50 users and 10, 50 and 100 threads. x-axis correspond to threads sorted by their (true) length while y-axis correspond to predicted thread lengths. True lengths are plotted in *red* (*smoothest lines*). Figures on the *right*: examples of posterior pairwise clustering matrices $\hat{\pi}$. x-axes and y-axes correspond to the users. A *dark point* means a high probability of being in the same cluster. The worst case is **c**, which makes a similar clustering to **b** but worse predictions, because the feature view receives misguiding information from the behavior view and the number of threads is not high enough to compensate for it (color figure online)

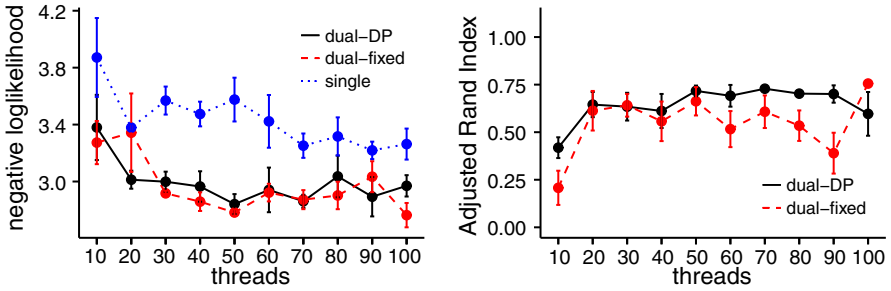

Fig. 6 Results for disagreement between the views. Comparison of models under different threads/users ratios (50 users and variable number of threads) when the two views see a different number of clusters. Means and standard errors over 5 runs

four clusters according to the feature view and five clusters according to the behavior view.

Figure 5 shows the posterior distributions (over thread lengths and over pairwise clustering) when (a) the behavior view has more information (b) both views lack data (c) the feature view has more information. By *having more information* we mean that a view dominates the inference over the posterior of the clustering **z**.

(a) Lack of information in feature view If the number of users is low but they participated in a sufficient number of threads, the behavior view (which sees five clusters) will be stronger than the feature view (which sees four clusters) and will impose a clustering in five groups. User coefficients (used for thread length predictions) are also well estimated since the number of threads is high enough to learn them despite the lack of help from the other view (Fig. 5a).

(b) Lack of information in both views In the middle ground where neither view has enough evidence, the model recovers four clusters and the predictive posterior over thread lengths gets wider though still making good inferences (Fig. 5b).

(c) Lack of information in behavior view If the number of users is high and the number of threads is low, the feature view (four clusters) will have more influence in the posterior than the behavior view (five clusters), (Fig. 5c). Predictions get worse because the model imposes a four clusters prior over coefficients that are clearly grouped in five clusters.

In order to compare between the performance in case of agreement and the performance in case of disagreement, we repeated the experiments of the last section with the current configuration. Figure 6 shows the performance of the models for 50 users and a different number of threads. While predictions and clustering improve with the number of threads, clustering with a small number of threads is worse in case of disagreement since the feature view imposes its 4 clusters vision. To recover the five clusters we would need either more threads or less users.

For the predictions, the dual models still outperform the single one because the feature view mostly agrees with the behavior view except for the users in one of the clusters. If all user features were in the same cluster, (no clustering structure) the performance of the predictions would be similar for the three models since the feature view would add no extra information. If we randomize the features so that, for instance, there are five clusters in the feature view that are very different from the clusters in the behavior view, we may expect the dual-view models to give worse predictions than the single-view one in those cases where they now perform better. In those cases, dual-models would be getting noise in the feature view (or very bad priors) and only a large enough number of threads could compensate for it.

5.5 Iris dataset

To reproduce a more realistic clustering structure we performed a last experiment based on the *iris* dataset. We used the *iris* data available in R, which corresponds to the original dataset reported in Anderson (1935). In our experiment, features correspond to three of the features of the *iris* dataset (Fig. 7). We chose three out of the four

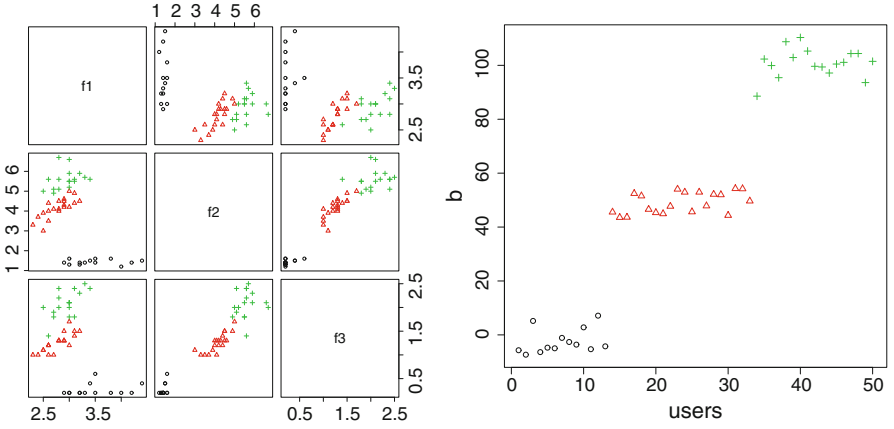

Fig. 7 Iris dataset. User features (*left*) and synthetic user coefficients (*right*)

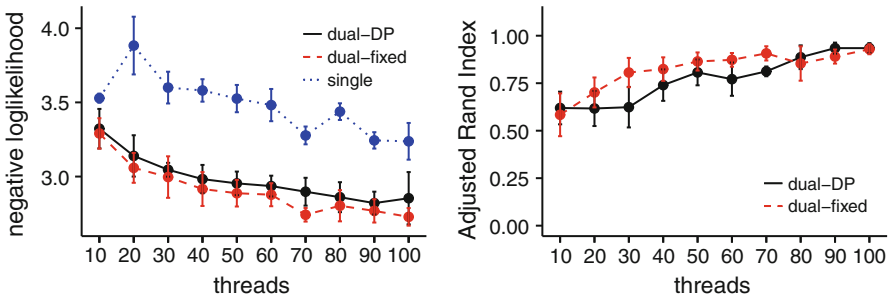

Fig. 8 Results for the iris dataset. Comparison of models under different threads/users ratios (50 users and variable number of threads). Means and standard errors over 5 runs

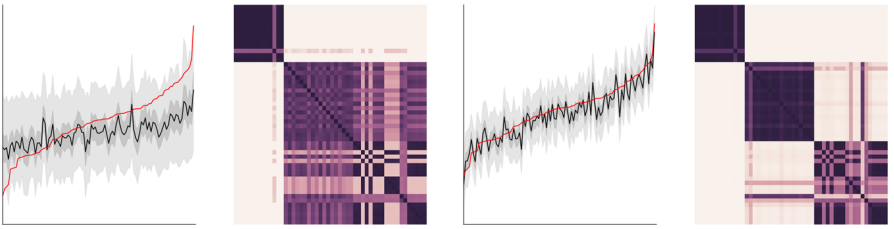

Fig. 9 Predictive posteriors given by the `DP-dual` model over thread lengths and pairwise clustering matrices with 50 users and a training set of 10 threads (*left*) and 100 threads (*right*). *Shadowed* areas indicate 95 and 50 % credible intervals. True lengths are plotted in *red* (*smoothest lines*). As the number of thread increases, the model starts to see the three cluster structure as well as to make better predictions (color figure online)

features (sepal width, petal length and petal width) as well as a random subset of 50 observations so that the clustering task is harder if we only use the feature view. The coefficients of the behavior view are similar to those used in the former experiments. The selected observations are assigned to the same cluster than in the *iris* dataset (species). We run a standard EM-GMM, from the R package `mclust` (Fraley et al. 2012), over the features to have an idea of what we should expect from our model when there are almost no threads and the coefficients are difficult to infer. We also run the same EM-GMM over the features and the true coefficients to have an idea of what we should expect from our model when the number of threads is high enough to make a good inference of the coefficients. This gave us an ARI of 0.48 and 0.79, respectively. Indeed, starting nearer to 0.48 when the number of threads is small, our model gets closer to 0.79 as we keep adding threads (Fig. 8). Of course, the inference of the coefficients and thus the predictions over the test set also improve by increasing the number of threads. Since the single model does not take profit of the feature view, it needs more threads to reach the same levels than its dual counterparts. Figure 9 shows two examples of confusion matrices and predicted lengths for 10 and 100 threads.

Figure 10 shows the histogram of the number of clusters within the MCMC chain and the distribution of cluster sizes. The model infers three clusters but it also places some probability over a two clusters structure due to the closeness of two of the clusters in the feature view.

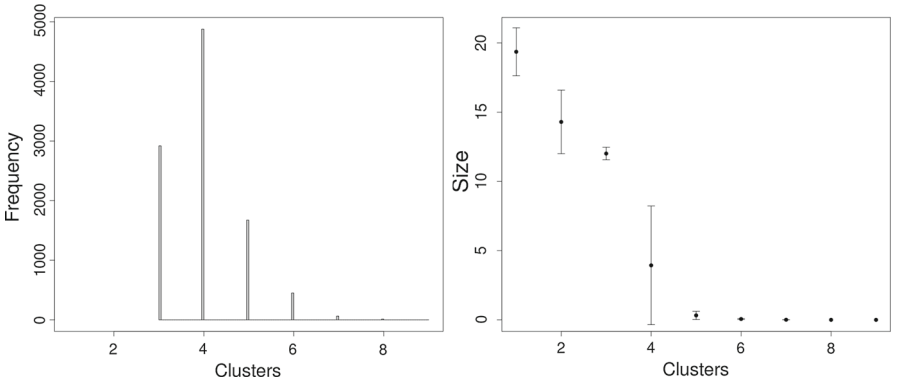

Fig. 10 *Left* histogram of the number of active clusters estimated by the `DP-model` in the *iris* scenario with 50 threads. *Right* mean and standard deviations of the number of users assigned to each cluster during the chain. Most users are assigned to the three major clusters and a small group of users is assigned to a fourth cluster

5.6 Computational cost

We analyzed the computational cost of the `dual-DP` model since it is the most complex of the three compared. Unlike the `single` model, it makes inferences in the two views, meaning about twice the number of variables. And unlike the `fixed-dual`, it has to infer the number of cluster and does it by creating m empty candidate clusters every time we sample a cluster assignment for a user at each iteration of the Gibbs sampler. This means creating $U \times iterations \times m$ empty clusters and computing, as many times, whether a user belongs to one of these auxiliary clusters (m possible extra clusters at each iteration), which makes it the slowest of the three models in terms of time per iteration.

We look at the autocorrelation time to estimate the distance between two uncorrelated samples:

$$\tau = 1 + 2 \sum_{n=1}^{1000} |\rho_n|$$

where ρ_n is the autocorrelation at lag n. The variable with the higher autocorrelation is $\mu_0^{(\mathrm{f})}$ which has an autocorrelation time of 79. Since we drop the first 15,000 samples for burn-in, we get an Effective Sample Size of 189 independent samples.

The bottlenecks of the algorithm are computing the likelihoods of the features and the coefficients given a cluster assignment, and the sampling of the coefficients. The sampling of the coefficients is relatively slow because it implies sampling from a multivariate Gaussian distribution with a $U \times U$ covariance matrix (see "Appendix"). This is due to the fact that the coefficient of each user depends on the coefficients of the users who have co-participated in the same thread. Note that this bottleneck would disappear in other scenarios where the inference of the behavioral function of a user is independent from the other users.

5.7 Summary of the experiments

To summarize, the experiments show on the one hand that dual-view models outperform single-view ones when users can be grouped in clusters that share similar features and latent behavioral functions and, on the other hand, that even when this assumption is not true, as long as there is enough data, the inference will lead to a consensual partition and a good estimation of latent functions. Indeed, each view acts as a prior, or a regularization factor, on the other view. Good priors improve inference, while bad priors misguide the inferences unless there is sufficient amount of evidence to ignore the prior.

6 Conclusions

We presented a dual-view mixture model to cluster users based on features and latent behavioral functions. Every component of the mixture model represents a probability density over two *views*: a feature view for observed user attributes and a behavior view for latent behavioral functions that are indirectly observed through user actions or behaviors. The posterior distribution of the clustering represents a consensual clustering between the two views. Inference of the parameters in each view depends on the other view through this common clustering, which can be seen as a proxy that passes information between the views. An appealing property of the model is that inference on latent behavioral functions may be used to make predictions of users future behaviors. We presented two versions of the model: a parametric one where the number of clusters is treated as a fixed parameter and a nonparametric, based on a Dirichlet Process, where the number of clusters is also inferred.

We have adapted the model to a hypothetical case of online forums where behaviors correspond to the ability of users to generate long discussions. We clustered users and inferred their behavioral functions in three datasets to understand the properties of the model. We inferred the posteriors of interest by Gibbs sampling for all the variables but two of them which were inferred by Adapted Rejection Sampling. Experiments confirm that the proposed dual-view model is able to learn with less instances than its single-view counterpart due to the fact that dual-view models use more information. Moreover, inferences with the dual-view model based on a Dirichlet Process are as good as inferences with the parametric model even if the latter knows the true number of clusters.

In our future research we plan to adapt and apply the model to more realistic tasks such as learning users preferences based on choices and users features. Particularly, we would like to compare our model to that of Abbasnejad et al. (2013) in the *sushi* dataset (Kamishima and Akaho 2009). Also, it might be interesting to consider latent functions at a group level, that is, that users in the same cluster share *exactly* the same latent behavior. Not only it would reduce the computational cost but, if we have few data about every user, a group-level inference may also be more grounded and statistically sound.

Finally, in order to apply the model to large scale data we will also explore alternative and faster inference methods such as Bayesian Variational Inference.

Appendix

Chinese Restaurant Process

In this section we recall the derivation of a Chinese Restaurant Process. Such a process will be used as the prior over cluster assignments in the model. This prior will then be updated through the likelihoods of the observations through the different views.

Imagine that every user u belongs to one of K clusters. z_u is the cluster of user u and $\mathbf{z}$ is a vector that indicates the cluster of every user. Let us assume that z_u is a random variable drawn from a multimomial distribution with probabilities $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_K)$. Let us also assume that the vector $\boldsymbol{\pi}$ is a random variable drawn from a Dirichlet distribution with a symmetric concentration parameter $\boldsymbol{\alpha} = (\alpha/K, \ldots, \alpha/K)$. We have:

$$\begin{aligned} z_u|\boldsymbol{\pi} &\sim \text{Multinomial}(\boldsymbol{\pi}) \\ \boldsymbol{\pi} &\sim \text{Dirichlet}(\boldsymbol{\alpha}) \end{aligned}$$

The marginal probability of the set of cluster assignments $\mathbf{z}$ is:

$$\begin{aligned} p(\mathbf{z}) &= \int \prod_{u=1}^{U} p(z_u|\boldsymbol{\pi})\, p(\boldsymbol{\pi}|\boldsymbol{\alpha}) \mathrm{d}\boldsymbol{\pi} \\ &= \int \prod_{i=1}^{K} \pi_i^{n_i} \frac{1}{B(\boldsymbol{\alpha})} \prod_{j=1}^{K} \pi_j^{\alpha/K-1} \mathrm{d}\boldsymbol{\pi} \\ &= \frac{1}{B(\boldsymbol{\alpha})} \int \prod_{i=1}^{K} \pi_i^{\alpha/K+n_i-1} \mathrm{d}\boldsymbol{\pi} \end{aligned}$$

where n_i is the number of users in cluster i and B denotes the Beta function. Noticing that the integrated factor is a Dirichlet distribution with concentration parameter $\boldsymbol{\alpha}+\mathbf{n}$ but without its normalizing factor:

$$\begin{aligned} p(\mathbf{z}) &= \frac{B(\boldsymbol{\alpha}+\mathbf{n})}{B(\boldsymbol{\alpha})} \int \frac{1}{B(\boldsymbol{\alpha}+\mathbf{n})} \prod_{i=1}^{K} \pi_i^{\alpha/K+n_i-1} \mathrm{d}\boldsymbol{\pi} \\ &= \frac{B(\boldsymbol{\alpha}+\mathbf{n})}{B(\boldsymbol{\alpha})} \end{aligned}$$

which expanding the definition of the Beta function becomes:

$$p(\mathbf{z}) = \frac{\prod_{i=1}^{K} \Gamma(\alpha/K+n_i)}{\Gamma\left(\sum_{i=1}^{K} \alpha/K+n_i\right)} \frac{\Gamma\left(\sum_{i=1}^{K} \alpha/K\right)}{\prod_{i=1}^{K} \Gamma(\alpha/K)} = \frac{\prod_{i=1}^{K} \Gamma(\alpha/K+n_i)}{\Gamma(\alpha+U)} \frac{\Gamma(\alpha)}{\prod_{i=1}^{K} \Gamma(\alpha/K)} \tag{35}$$

where $U = \sum_{i=1}^{K} n_i$. Note that marginalizing out $\boldsymbol{\pi}$ we introduce dependencies between the individual clusters assignments under the form of the counts n_i. The conditional distribution of an individual assignment given the others is:

$$p(z_u = j|\mathbf{z}_{-\mathbf{u}}) = \frac{p(\mathbf{z})}{p(\mathbf{z}_{-u})} \tag{36}$$

To compute the denominator we assume cluster assignments are exchangeable, that is, the joint distribution $p(\mathbf{z})$ is the same regardless the order in which clusters are assigned. This allows us to assume that z_u is the last assignment, therefore obtaining $p(\mathbf{z}_{-\mathbf{u}})$ by considering how Eq. 35 before z_u was assigned to cluster j.

$$p(\mathbf{z}_{-u}) = \frac{\Gamma(\alpha/K + n_j - 1)\prod_{i\neq j}\Gamma(\alpha/K + n_i)}{\Gamma(\alpha + U - 1)} \frac{\Gamma(\alpha)}{\prod_{i=1}\Gamma(\alpha/K)} \tag{37}$$

And finally plugging Eqs. 37 and 35 into Eq. 36, and cancelling out the factors that do not depend on the cluster assignment z_u, and finally using the identity $a\Gamma(a) = \Gamma(a+1)$ we get:

$$p(z_u = j|\mathbf{z}_{-u}) = \frac{\alpha/K + n_j - 1}{\alpha + U - 1} = \frac{\alpha/K + n_{-j}}{\alpha + U - 1}$$

where n_{-j} is the number of users in cluster j before the assignment of z_u.

The Chinese Restaurant Process is the consequence of considering $K \to \infty$. For clusters where $n_{-j} > 0$, we have:

$$p(z_u = j \text{ s.t } n_{-j} > 0|\mathbf{z}_{-u}) = \frac{n_{-j}}{\alpha + U - 1}$$

and the probability of assigning z_u to any of the (infinite) empty clusters is:

$$p(z_u = j \text{ s.t } n_{-j} = 0|\mathbf{z}_{-\mathbf{u}}) = \lim_{K\to\infty}(K - p)\frac{\alpha/K}{\alpha + U - 1} = \frac{\alpha}{\alpha + U - 1}$$

where p is the number of non-empty components. It can be shown that the generative process composed of a Chinese Restaurant Process were every component j is associated to a probability distribution with parameters $\boldsymbol{\theta}_j$ is equivalent to a Dirichlet Process.

Conditionals for the feature view

In this appendix we provide the conditional distributions for the feature view to be plugged into the Gibbs sampler. Note that, except for $\beta_0^{(a)}$, conjugacy can be exploited in every case and therefore their derivations are straightforward and well known. The derivation for $\beta_0^{(a)}$ is left for another section:

Component parameters

Components means $p(\boldsymbol{\mu}_k^{(a)}|\cdot)$*:*

$$\begin{aligned} p(\boldsymbol{\mu}_k^{(\mathrm{a})}|\cdot) &\propto p\left(\boldsymbol{\mu}_k^{(\mathrm{a})}|\boldsymbol{\mu}_0^{(\mathrm{a})}, \left(\mathbf{R}_0^{(\mathrm{a})}\right)^{-1}\right) \prod_{u\in k} p\left(\mathbf{a}_u|\boldsymbol{\mu}_k^{(\mathrm{a})}, \mathbf{S}_k^{(\mathrm{a})}, \mathbf{z}\right) \\ &\propto \mathcal{N}\left(\boldsymbol{\mu}_k^{(\mathrm{a})}|\boldsymbol{\mu}_0^{(\mathrm{a})}, \left(\mathbf{R}_0^{(\mathrm{a})}\right)^{-1}\right) \prod_{u\in k} \mathcal{N}\left(\mathbf{a}_u|\boldsymbol{\mu}_k^{(\mathrm{a})}, \mathbf{S}_k^{(\mathrm{a})}\right) \\ &= \mathcal{N}(\boldsymbol{\mu}', \boldsymbol{\Lambda}') \end{aligned}$$

where:

$$\begin{aligned} \boldsymbol{\Lambda}' &= \mathbf{R}_0^{(\mathrm{a})} + n_k \mathbf{S}_k^{(\mathrm{a})} \\ \boldsymbol{\mu}' &= \boldsymbol{\Lambda}'^{-1}\left(\mathbf{R}_0^{(\mathrm{a})}\boldsymbol{\mu}_0^{(\mathrm{a})} + \mathbf{S}_k^{(\mathrm{a})} \sum_{u\in k} \mathbf{a}_u\right) \end{aligned}$$

Components precisions $p(\mathbf{S}_k^{(a)}|\cdot)$*:*

$$\begin{aligned} p(\mathbf{S}_k^{(\mathrm{a})}|\cdot) &\propto p\left(\mathbf{S}_k^{(\mathrm{a})}|\beta_0^{(\mathrm{a})}, \mathbf{W}_0^{(\mathrm{a})}\right) \prod_{u\in k} p\left(\mathbf{a}_u|\boldsymbol{\mu}_k^{(\mathrm{a})}, \mathbf{S}_k^{(\mathrm{a})}, \mathbf{z}\right) \\ &\propto \mathcal{W}\left(\mathbf{S}_k^{(\mathrm{a})}|\beta_0^{(\mathrm{a})}, (\beta_0^{(\mathrm{a})}\mathbf{W}_0^{(\mathrm{a})})^{-1}\right) \prod_{u\in k} \mathcal{N}\left(\mathbf{a}_u|\boldsymbol{\mu}_k^{(\mathrm{a})}, \mathbf{S}_k^{(\mathrm{a})}\right) \\ &= \mathcal{W}(\beta', \mathbf{W}') \end{aligned}$$

where:

$$\begin{aligned} \beta' &= \beta_0^{(\mathrm{a})} + n_k \\ \mathbf{W}' &= \left[\beta_0^{(\mathrm{a})}\mathbf{W}_0^{(\mathrm{a})} + \sum_{u\in k} (\mathbf{a}_u - \boldsymbol{\mu}_k^{(\mathrm{a})})(\mathbf{a}_u - \boldsymbol{\mu}_k^{(\mathrm{a})})^T\right]^{-1} \end{aligned}$$

Shared hyper-parameters

Shared base means $p(\boldsymbol{\mu}_0^{(a)}|\cdot)$*:*

$$\begin{aligned} p(\boldsymbol{\mu}_0^{(\mathrm{a})}|\cdot) &\propto p\left(\boldsymbol{\mu}_0^{(\mathrm{a})}|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a\right) \prod_{k=1}^{K} p\left(\boldsymbol{\mu}_k^{(\mathrm{a})}|\boldsymbol{\mu}_0^{(\mathrm{a})}, \mathbf{R}_0^{(\mathrm{a})}\right) \\ &\propto \mathcal{N}\left(\boldsymbol{\mu}_0^{(\mathrm{a})}|\boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a\right) \prod_{k=1}^{K} \mathcal{N}\left(\boldsymbol{\mu}_k^{(\mathrm{a})}|\boldsymbol{\mu}_0^{(\mathrm{a})}, \left(\mathbf{R}_0^{(\mathrm{a})}\right)^{-1}\right) \\ &= \mathcal{N}\left(\boldsymbol{\mu}', \boldsymbol{\Lambda}'^{-1}\right) \end{aligned}$$

where:

$$\begin{aligned}\boldsymbol{\Lambda}' &= \boldsymbol{\Lambda}_{\boldsymbol{a}} + K\mathbf{R}_0^{(\mathrm{a})} \\ \boldsymbol{\mu}' &= \boldsymbol{\Lambda}'^{-1}\left(\boldsymbol{\Lambda}_{\boldsymbol{a}}\boldsymbol{\mu}_{\boldsymbol{a}} + K\mathbf{R}_0^{(\mathrm{a})}\overline{\boldsymbol{\mu}_k^{(\mathrm{a})}}\right)\end{aligned}$$

Shared base precisions $p(\mathbf{R}_0^{(a)}|\cdot)$*:*

$$\begin{aligned}p(\mathbf{R}_0^{(\mathrm{a})}|\cdot) &\propto p\left(\mathbf{R}_0^{(\mathrm{a})}|D, \boldsymbol{\Sigma}_{\boldsymbol{a}}^{-\mathbf{1}}\right)\prod_{k=1}^{K} p\left(\boldsymbol{\mu}_k^{(\mathrm{a})}|\boldsymbol{\mu}_0^{(\mathrm{a})}, \mathbf{R}_0^{(\mathrm{a})}\right) \\ &\propto \mathcal{W}\left(\mathbf{R}_0^{(\mathrm{a})}|D, (D\boldsymbol{\Sigma}_{\boldsymbol{a}})^{-1}\right)\prod_{k=1}^{K}\mathcal{N}\left(\boldsymbol{\mu}_k^{(\mathrm{a})}|\boldsymbol{\mu}_0^{(\mathrm{a})}, \left(\mathbf{R}_0^{(\mathrm{a})}\right)^{-1}\right) \\ &= \mathcal{W}(\upsilon', \boldsymbol{\Psi}')\end{aligned}$$

where:

$$\begin{aligned}\upsilon' &= D + K \\ \boldsymbol{\Psi}' &= \left[D\boldsymbol{\Sigma}_{\boldsymbol{a}} + \sum_k\left(\boldsymbol{\mu}_k^{(\mathrm{a})} - \boldsymbol{\mu}_0^{(\mathrm{a})}\right)\left(\boldsymbol{\mu}_k^{(\mathrm{a})} - \boldsymbol{\mu}_0^{(\mathrm{a})}\right)^T\right]^{-1}\end{aligned}$$

Shared base covariances $p(\mathbf{W}_0^{(a)}|\cdot)$*:*

$$\begin{aligned}p(\mathbf{W}_0^{(\mathrm{a})}|\cdot) &\propto p\left(\mathbf{W}_0^{(\mathrm{a})}|D, \frac{1}{D}\boldsymbol{\Sigma}_{\boldsymbol{a}}\right)\prod_{k=1}^{K} p\left(\mathbf{S}_k^{(\mathrm{a})}|\beta_0^{(\mathrm{a})}, \left(\mathbf{W}_0^{(\mathrm{a})}\right)^{-1}\right) \\ &\propto \mathcal{W}\left(\mathbf{W}_0^{(\mathrm{a})}|D, \frac{1}{D}\boldsymbol{\Sigma}_{\boldsymbol{a}}\right)\prod_{k=1}^{K}\mathcal{W}\left(\mathbf{S}_k^{(\mathrm{a})}|\beta_0^{(\mathrm{a})}, \left(\beta_0^{(\mathrm{a})}\mathbf{W}_0^{(\mathrm{a})}\right)^{-1}\right) \\ &= \mathcal{W}(\upsilon', \boldsymbol{\Psi}')\end{aligned}$$

where:

$$\begin{aligned}\upsilon' &= D + K\beta_0^{(\mathrm{a})} \\ \boldsymbol{\Psi}' &= \left[D\boldsymbol{\Sigma}_{\boldsymbol{a}}{}^{-1} + \beta_0^{(\mathrm{a})}\sum_{k=1}^{K}\mathbf{S}_k^{(\mathrm{a})}\right]^{-1}\end{aligned}$$

Shared base degrees of freedom $p(\beta_0^{(a)}|\cdot)$*:*

$$\begin{aligned}p\left(\beta_0^{(\mathrm{a})}|\cdot\right) &\propto p\left(\beta_0^{(\mathrm{a})}\right)\prod_{k=1}^{K} p\left(\mathbf{S}_k^{(\mathrm{a})}|\mathbf{W}_0^{(\mathrm{a})}, \beta_0^{(\mathrm{a})}\right) \\ &= p\left(\beta_0^{(\mathrm{a})}|1, \frac{1}{D}\right)\prod_{k=1}^{K}\mathcal{W}\left(\mathbf{S}_k^{(\mathrm{a})}|\mathbf{W}_0^{(\mathrm{a})}, \beta_0^{(\mathrm{a})}\right)\end{aligned}$$

where there is no conjugacy we can exploit. We may sample from this distribution with Adaptive Rejection Sampling.

Conditionals for the behavior view

In this appendix we provide the conditional distributions for the behavior view to be plugged into the Gibbs sampler. Except for β_{b_0}, conjugacy can be exploited in every case and therefore their derivations straightforward and well known. The derivation for β_{b_0} is left for another section:

Users parameters

Users latent coefficient $p(b_u|\cdot)$*:*

Let $\mathbf{Z}$ be a $K \times U$ a binary matrix where $\mathbf{Z}_{k,u} = 1$ denotes whether user u is assigned to cluster k. Let $\mathbf{I}_{[\mathbf{T}]}$ and $\mathbf{I}_{[\mathbf{U}]}$ identity matrices of sizes T and U, respectively. Let $\boldsymbol{\mu}^{(\mathrm{f})} = (\mu_1^{(\mathrm{f})}, \ldots, \mu_K^{(\mathrm{f})})$ and $\mathbf{s}^{(\mathrm{f})} = (s_1^{(\mathrm{f})}, \ldots, s_K^{(\mathrm{f})})$ Then:

$$\begin{aligned} p\left(\mathbf{b}|\cdot\right) &\propto p\left(\mathbf{b}|\boldsymbol{\mu}^{(\mathrm{f})}, \mathbf{s}^{(\mathrm{f})}, \mathbf{Z}\right) p\left(\mathbf{y}|\mathbf{P}, \mathbf{b}\right) \\ &\propto \mathcal{N}\left(\mathbf{b}|\mathbf{Z}^T\boldsymbol{\mu}^{(\mathrm{f})}, \mathbf{Z}^T\mathbf{s}^{(\mathrm{f})}\mathbf{I}_{[\mathbf{U}]}\right)\mathcal{N}\left(\mathbf{y}|\mathbf{P}^T\mathbf{b}, \sigma_{\mathbf{y}}\mathbf{I}_{[\mathbf{T}]}\right) \\ &= \mathcal{N}\left(\boldsymbol{\mu}', \boldsymbol{\Lambda}'^{-\mathbf{1}}\right) \end{aligned}$$

where:

$$\begin{aligned} \Lambda' &= \mathbf{Z}^T\mathbf{s}^{(\mathrm{f})}\mathbf{I}_{[\mathbf{U}]} + \mathbf{P}\sigma_{\mathrm{y}}^{-2}\mathbf{I}_{[T]}\mathbf{P}^T \\ \boldsymbol{\mu}' &= {}'^{-1}\left(\mathbf{Z}^T\mathbf{s}^{(\mathrm{f})}\mathbf{Z}^T\boldsymbol{\mu}^{(\mathrm{f})} + \mathbf{P}\sigma_{\mathrm{y}}^{-2}\mathbf{I}_{[\mathbf{T}]}\mathbf{y}\right) \end{aligned}$$

Component parameters

Components means $p(\mu_k^{(f)}|\cdot)$*:*

$$\begin{aligned} p\left(\mu_k^{(\mathrm{f})}|\cdot\right) &\propto p\left(\mu_k^{(\mathrm{f})}|\mu_0^{(\mathrm{f})}, \left(r_0^{(\mathrm{f})}\right)^{-1}\right)\prod_{u\in k} p\left(b_u|\mu_k^{(\mathrm{f})}, s_k^{(\mathrm{f})}, \mathbf{z}\right) \\ &\propto \mathcal{N}\left(\mu_k^{(\mathrm{f})}|\mu_0^{(\mathrm{f})}, \left(r_0^{(\mathrm{f})}\right)^{-1}\right)\prod_{u\in k}\mathcal{N}\left(b_u|\mu_k^{(\mathrm{f})}, s_k^{(\mathrm{f})}\right) \\ &= \mathcal{N}\left(\boldsymbol{\mu}', \boldsymbol{\Lambda}'^{-1}\right) \end{aligned}$$

where:

$$\boldsymbol{\Lambda}' = r_0^{(\mathrm{f})} + n_k s_k^{(\mathrm{f})}$$

$$\boldsymbol{\mu}' = \boldsymbol{\Lambda}'^{-1}\left(r_0^{(\mathrm{f})}\mu_0^{(\mathrm{f})} + s_k^{(\mathrm{f})}\sum_{u\in k} b_u\right)$$

Components precisions $p(s_k^{(f)}|\cdot)$*:*

$$\begin{aligned} p\left(s_k^{(\mathrm{f})}|\cdot\right) &\propto p\left(s_k^{(\mathrm{f})}|\beta_0^{(\mathrm{f})}, w_0^{(\mathrm{f})}\right)\prod_{u\in k} p\left(b_u|\mu_k^{(\mathrm{f})}, s_k^{(\mathrm{f})}, \mathbf{z}\right) \\ &\propto \mathcal{G}\left(s_k^{(\mathrm{f})}|\beta_0^{(\mathrm{f})}, \left(\beta_0^{(\mathrm{f})} w_0^{(\mathrm{f})}\right)^{-1}\right)\prod_{u\in k}\mathcal{N}\left(b_u|\mu_k^{(\mathrm{f})}, s_k^{(\mathrm{f})}\right) \\ &= \mathcal{G}\left(\upsilon', \psi'\right) \end{aligned}$$

where:

$$\begin{aligned} \upsilon' &= \beta_0^{(\mathrm{f})} + n_k \\ \psi' &= \left[\beta_0^{(\mathrm{f})} w_0^{(\mathrm{f})} + \sum_{u\in k}\left(b_u - \mu_k^{(\mathrm{f})}\right)^2\right]^{-1} \end{aligned}$$

Shared hyper-parameters

Shared base mean $p(\mu_0^{(f)}|\cdot)$*:*

$$\begin{aligned} p\left(\mu_0^{(\mathrm{f})}|\cdot\right) &\propto p\left(\mu_0^{(\mathrm{f})}|\mu_{\hat{b}}, \sigma_{\hat{b}}\right)\prod_{k=1}^{K} p\left(\mu_k^{(\mathrm{f})}|\mu_0^{(\mathrm{f})}, r_0^{(\mathrm{f})}\right) \\ &\propto \mathcal{N}\left(\mu_0^{(\mathrm{f})}|\mu_{\hat{b}}, \sigma_{\hat{b}}\right)\prod_{k=1}^{K}\mathcal{N}\left(\mu_k^{(\mathrm{f})}|\mu_0^{(\mathrm{f})}, \left(r_0^{(\mathrm{f})}\right)^{-1}\right) \\ &= \mathcal{N}\left(\mu', \sigma'^{-2}\right) \end{aligned}$$

where:

$$\begin{aligned} \sigma'^{-2} &= \sigma_{\hat{b}}^{-2} + K r_0^{(\mathrm{f})} \\ \mu' &= \sigma_{\hat{b}}^{2'}\left(\sigma_{\hat{b}}^{-2}\mu_{\hat{b}} + K r_0^{(\mathrm{f})}\overline{\mu_k^{(\mathrm{f})}}\right) \end{aligned}$$

Shared base precision $p(r_0^{(f)}|\cdot)$

$$\begin{aligned} p\left(r_0^{(\mathrm{f})}|\cdot\right) &\propto p\left(r_0^{(\mathrm{f})}|1,\sigma_{\hat{b}}^{-2}\right)\prod_{k=1}^{K} p\left(\mu_k^{(\mathrm{f})}|\mu_0^{(\mathrm{f})},r_0^{(\mathrm{f})}\right) \\ &\propto \mathcal{G}\left(r_0^{(\mathrm{f})}|1,\sigma_{\hat{b}}^{-2}\right)\prod_{k=1}^{K}\mathcal{N}\left(\mu_k^{(\mathrm{f})}|\mu_0^{(\mathrm{f})},\left(r_0^{(\mathrm{f})}\right)^{-1}\right) \\ &= \mathcal{G}\left(\upsilon',\psi'\right) \end{aligned}$$

where:

$$\begin{aligned} \upsilon' &= 1+K \\ \psi' &= \left[\sigma_{\hat{b}}^{-2}+\sum_{k=1}^{K}\left(\mu_k^{(\mathrm{f})}-\mu_0^{(\mathrm{f})}\right)^2\right]^{-1} \end{aligned}$$

Shared base variance $p(w_0^{(f)}|\cdot)$*:*

$$\begin{aligned} p(w_0^{(\mathrm{f})}|\cdot) &\propto p(w_0^{(\mathrm{f})}|1,\sigma_{\hat{b}})\prod_{r=1}^{K} p\left(s_k^{(\mathrm{f})}|\beta_0^{(\mathrm{f})},w_0^{(\mathrm{f})}\right) \\ &\propto \mathcal{G}(w_0^{(\mathrm{f})}|1,\sigma_{\hat{b}})\prod_{k=1}^{K}\mathcal{G}\left(s_k^{(\mathrm{f})}|\beta_0^{(\mathrm{f})},\left(\beta w_0^{(\mathrm{f})}\right)^{-1}\right) \\ &= \mathcal{G}(\upsilon',\psi') \\ \upsilon' &= 1+K\beta_0^{(\mathrm{f})} \\ \psi' &= \left[\sigma_{\hat{b}}^{-2}+\beta_0^{(\mathrm{f})}\sum_{k=1}^{K}s_k^{(\mathrm{f})}\right]^{-1} \end{aligned}$$

Shared base degrees of freedom $p(\beta_0^{(f)}|\cdot)$*:*

$$\begin{aligned} p\left(\beta_0^{(\mathrm{f})}|\cdot\right) &\propto p\left(\beta_0^{(\mathrm{f})}\right)\prod_{r=1}^{K} p\left(s_k^{(\mathrm{f})}|w_0^{(\mathrm{f})},\beta_0^{(\mathrm{f})}\right) \\ &= p\left(\beta_0^{(\mathrm{f})}|1,1\right)\prod_{r=1}^{K}\mathcal{G}\left(s_k^{(\mathrm{f})}|\beta_0^{(\mathrm{f})},\left(\beta_0^{(\mathrm{f})}w_0^{(\mathrm{f})}\right)^{-1}\right) \end{aligned}$$

where there is no conjugacy we can exploit. We will sample from this distribution with Adaptive Rejection Sampling.

Regression noise

Let the precision s_{y} be the inverse of the variance σ_{y}^2. Then:

$$\begin{aligned} p\left(s_{\mathrm{y}}|\cdot\right) &\propto p\left(s_{\mathrm{y}}|1,\sigma_0^{-2}\right)\prod_{t=1}^{T} p\left(y_t|\mathbf{p}^{\mathbf{T}}\mathbf{b},s_{\mathrm{y}}\right) \\ &\propto \mathcal{G}\left(s_{\mathrm{y}}|1,\sigma_0^{-2}\right)\prod_{t=1}^{T}\mathcal{N}\left(y_t|\mathbf{p}^{\mathbf{T}}\mathbf{b},s_{\mathrm{y}}\right) \\ &= \mathcal{G}\left(\upsilon',\psi'\right) \\ \upsilon' &= 1+T \\ \psi' &= \left[\sigma_0^2+\sum_{t=1}^{T}\left(y_t-\mathbf{p}^{\mathbf{T}}\mathbf{b}\right)^2\right]^{-1} \end{aligned}$$

Sampling $\boldsymbol{\beta}_{\mathbf{0}}^{(\mathbf{a})}$

For the feature view, if:

$$\frac{1}{\beta-D+1}\sim\mathcal{G}\left(1,\frac{1}{D}\right)$$

we can get the prior distribution of β by variable transformation:

$$\begin{aligned} p(\beta) &= \mathcal{G}\left(\frac{1}{\beta-D+1}\right)\left|\frac{\partial}{\partial\beta}\frac{1}{\beta-D+1}\right| \\ &\propto \left(\frac{1}{\beta-D+1}\right)^{-1/2}\exp\left(-\frac{D}{2(\beta-D+1)}\right)\frac{1}{(\beta-D+1)^2} \\ &\propto \left(\frac{1}{\beta-D+1}\right)^{3/2}\exp\left(-\frac{D}{2(\beta-D+1)}\right) \end{aligned}$$

Then:

$$p(\beta)\propto(\beta-D+1)^{-3/2}\exp\left(-\frac{D}{2(\beta-D+1)}\right)$$

The Wishart likelihood is:

$$\begin{aligned} \mathcal{W}\left(\mathbf{S}_k|\beta,(\beta\mathbf{W})^{-1}\right) &= \frac{\left(|\mathbf{W}|\,(\beta/2)^D\right)^{\beta/2}}{\Gamma_D\,(\beta/2)}|\mathbf{S}_k|^{(\beta-D-1)/2}\exp\left(-\frac{\beta}{2}\mathrm{Tr}\left(\mathbf{S}_k\mathbf{W}\right)\right) \\ &= \frac{\left(|\mathbf{W}|\,(\beta/2)^D\right)^{\beta/2}}{\prod_{d=1}^{D}\Gamma\left(\frac{\beta+d-D}{2}\right)}|\mathbf{S}_k|^{(\beta-D-1)/2}\exp\left(-\frac{\beta}{2}\mathrm{Tr}\left(\mathbf{S}_k\mathbf{W}\right)\right) \end{aligned}$$

We multiply both equations, the Wishart likelihood (its K factors) and the prior, to get the posterior:

$$p(\beta|\cdot) = \left(\prod_{d=0}^{D} \Gamma\left(\frac{\beta}{2} + \frac{d-D}{2}\right)\right)^{-K} \exp\left(-\frac{D}{2(\beta - D + 1)}\right)(\beta - D + 1)^{-3/2} \times \left(\frac{\beta}{2}\right)^{\frac{KD\beta}{2}} \prod_{k=1}^{K} (|\mathbf{S}_k||\mathbf{W}|)^{\beta/2} \exp\left(-\frac{\beta}{2}\mathrm{Tr}(\mathbf{S}_k\mathbf{W})\right)$$

Then if $y = \ln \beta$:

$$p(y|\cdot) = e^y \left(\prod_{d=0}^{D} \Gamma\left(\frac{e^y}{2} + \frac{d-D}{2}\right)\right)^{-K} \exp\left(-\frac{D}{2(e^y - D + 1)}\right)(e^y - D + 1)^{-3/2} \times \left(\frac{e^y}{2}\right)^{\frac{KDe^y}{2}} \prod_{k=1}^{K} (|\mathbf{S}_k||\mathbf{W}|)^{e^y/2} \exp\left(-\frac{e^y}{2}\mathrm{Tr}(\mathbf{S}_k\mathbf{W})\right)$$

and its logarithm is:

$$\ln p(y|\cdot) = y - K\sum_{d=0}^{D} \ln \Gamma\left(\frac{e^y}{2} + \frac{d-D}{2}\right) - \frac{D}{2(e^y - D + 1)} - \frac{3}{2}\ln\left(e^y - D + 1\right) + \frac{KDe^y}{2}(y - \ln 2) + \frac{e^y}{2}\sum_{k=1}^{K}(\ln(|\mathbf{S}_k||\mathbf{W}|) - \mathrm{Tr}(\mathbf{S}_k\mathbf{W}))$$

which is a concave function and therefore we can use Adaptive Rejection Sampling (ARS). ARS sampling works with the derivative of the log function:

$$\frac{\partial}{\partial y}\ln p(y|\cdot) = 1 - K\frac{e^y}{2}\sum_{d=1}^{D}\Psi\left(\frac{e^y}{2} + \frac{d-D}{2}\right) + \frac{De^y}{2(e^y - D + 1)^2} - \frac{3}{2}\frac{e^y}{e^y - D + 1} + \frac{KDe^y}{2}(y - \ln 2) + \frac{KDe^y}{2} + \frac{e^y}{2}\sum_{k=1}^{K}(\ln(|\mathbf{S}_k||\mathbf{W}|) - \mathrm{Tr}(\mathbf{S}_k\mathbf{W}))$$

where $\Psi(x)$ is the digamma function.

Sampling $\boldsymbol{\beta_0^{(\mathbf{f})}}$

For the behavior view, if

$$\frac{1}{\beta} \sim \mathcal{G}(1, 1)$$

the posterior of β is:

$$p(\beta|\cdot) = \Gamma\left(\frac{\beta}{2}\right)^{-K} \exp\left(\frac{-1}{2\beta}\right)\left(\frac{\beta}{2}\right)^{(K\beta-3)/2} \prod_{k=1}^{K} (s_k w)^{\beta/2} \exp\left(-\frac{\beta s_k w}{2}\right)$$

Then if $y = \ln \beta$:

$$p\,(y|\cdot) = e^y \Gamma\left(\frac{e^y}{2}\right)^{-K} \exp\left(\frac{-1}{2e^y}\right)\left(\frac{e^y}{2}\right)^{(Ke^y-3)/2} \prod_{k=1}^{K} (s_k w)^{e^y/2} \exp\left(-\frac{e^y s_k w}{2}\right)$$

and its logarithm:

$$\ln p\,(y|\cdot) = y - K \ln \Gamma\left(\frac{e^y}{2}\right) + \left(\frac{-1}{2e^y}\right) + \frac{Ke^y - 3}{2}(y - \ln 2) + \frac{e^y}{2}\sum_{k=1}^{K} (\ln (s_k w) - s_k w)$$

which is a concave function and therefore we can use Adaptive Rejection Sampling. The derivative is:

$$\frac{\partial}{\partial y} \ln p\,(y|\cdot) = 1 - K\Psi\left(\frac{e^y}{2}\right)\frac{e^y}{2} + \left(\frac{1}{2e^y}\right) + \frac{Ke^y}{2}(y - \ln 2) + \frac{Ke^y - 3}{2} + \frac{e^y}{2}\sum_{k=1}^{K} (\ln (s_k w) - s_k w)$$

where $\Psi(x)$ is the digamma function.

Sampling α

Since the inverse of the concentration parameter α is given a Gamma prior

$$\frac{1}{\alpha} \sim \mathcal{G}(1, 1)$$

we can get the prior over α by variable transformation:

$$p(\alpha) \propto \alpha^{-3/2} \exp(-1/(2\alpha))$$

Multiplying the prior of α by its likelihood we get the posterior:

$$p(\alpha|\cdot) \propto \alpha^{-3/2} \exp(-1/(2\alpha)) \times \frac{\Gamma(\alpha)}{\Gamma(\alpha + U)} \prod_{j=1}^{K} \frac{\Gamma(n_j + \alpha/K)}{\alpha/K}$$

$$\propto \alpha^{-3/2} \exp\left(-1/(2\alpha)\right) \frac{\Gamma(\alpha)}{\Gamma(\alpha+U)} \alpha^{K}$$
$$\propto \alpha^{K-3/2} \exp\left(-1/(2\alpha)\right) \frac{\Gamma(\alpha)}{\Gamma(\alpha+U)}$$

Then if $y = \ln \alpha$:

$$p(y|\cdot) = e^{y(K-3/2)} \exp(-1/(2e^{y})) \frac{\Gamma(e^{y})}{\Gamma(e^{y}+U)}$$

and its logarithm is:

$$\ln p(y|\cdot) = y(K-3/2) - 1/(2e^{y}) + \ln \Gamma(e^{y}) - \ln \Gamma(e^{y}+U)$$

which is a concave function and therefore we can use Adaptive Rejection Sampling. The derivative is:

$$\frac{\partial}{\partial y} \ln p(y|\cdot) = (K-3/2) + 1/(2e^{y}) + e^{y}\Psi(e^{y}) - e^{y}\Psi(e^{y}+U)$$

References

Abbasnejad E, Sanner S, Bonilla EV, Poupart P (2013) Learning community-based preferences via Dirichlet process mixtures of Gaussian processes. In: Proceedings of the 23rd international joint conference on artificial intelligence, IJCAI'13. AAAI Press, pp 1213–1219

Anderson E (1935) The irises of the Gaspe Peninsula. Bull Am Iris Soc 59:2–5

Bickel S, Scheffer T (2004) Multi-view clustering. In: Proceedings of the fourth IEEE international conference on data mining, ICDM'04. IEEE Computer Society, Washington, DC, USA, pp 19–26

Blum A, Mitchell T (1998) Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on computational learning theory, COLT'98. ACM, New York, NY, USA, pp 92–100

Bonilla EV, Guo S, Sanner S (2010) Gaussian process preference elicitation. In: Lafferty JD, Williams CKI, Shawe-Taylor J, Zemel RS, Culotta A (eds) Advances in neural information processing systems 23. Curran Associates, Inc, Red Hook, pp 262–270

Brown MPS, Grundy WN, Lin D, Cristianini N, Sugnet CW, Furey TS, Ares M, Haussler D (2000) Knowledge-based analysis of microarray gene expression data by using support vector machines. Proc Natl Acad Sci USA 97(1):262–267

Cheng Y, Agrawal A, Choudhary A, Liu H, Zhang T (2014) Social role identification via dual uncertainty minimization regularization. In: IEEE international conference on data mining. IEEE, pp 767–772

Cheung KW, Tsui KC, Liu J (2004) Extended latent class models for collaborative recommendation. IEEE Trans Syst Man Cybern Part A Syst Hum 34(1):143–148

Dahl DB (2006) Model-based clustering for expression data via a Dirichlet process mixture model. In: Do KA, Müller P, Vannucci M (eds) Bayesian inference for gene expression and proteomics. Cambridge University Press, Cambridge, pp 201–218

Eisen MB, Spellman PT, Brown PO, Botstein D (1998) Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci USA 95(25):14863–14868

Fraley C, Raftery AE, Murphy TB, Scrucca L (2012) mclust version 4 for R: normal mixture modeling for model-based clustering, classification, and density estimation. http://cran.r-project.org/package=mclust

Gilks W, Wild P (1992) Adaptive rejection sampling for Gibbs sampling. Appl Stat 41(2):337–348

Görür D, Rasmussen CE (2010) Dirichlet process Gaussian mixture models: choice of the base distribution. J Comput Sci Technol 25:653–664

Greene D, Pádraig C (2009) Multi-view clustering for mining heterogeneous social network data. In: Workshop on information retrieval over social networks, 31st European conference on information retrieval

Kamishima T, Akaho S (2009) Efficient clustering for orders. In: Zighed DA, Tsumoto S, Ras ZW, Hacid H (eds) Mining complex data. Springer, Berlin, pp 261–279

Kumar A, Rai P, Daume H (2011) Co-regularized multi-view spectral clustering. In: Shawe-Taylor J, Zemel RS, Bartlett PL, Pereira F, Weinberger KQ (eds) Advances in neural information processing systems 24. Curran Associates, Inc, Red Hook, pp 1413–1421

Neal RM (2000) Markov chain sampling methods for Dirichlet process mixture models. J Comput Graph Stat 9(2):249–265

Niu D, Dy JG, Ghahramani Z (2012) A nonparametric Bayesian model for multiple clustering with overlapping feature views. In: Proceedings of the 15th international conference on artificial intelligence and statistics. JMLR, pp 814–822

Pavlidis P, Weston J, Cai J, Noble WS (2002) Learning gene functional classifications from multiple data types. J Comput Biol 9(2):401–411

Pellegrini M, Marcotte EM, Thompson MJ, Eisenberg D, Yeates TO (1999) Assigning protein functions by comparative genome analysis: protein phylogenetic profiles. Proc Nat Acad Sci USA 96(8):4285–4288

Plummer M, Best N, Cowles K, Vines K (2006) CODA: convergence diagnosis and output analysis for MCMC. R News 6(1):7–11

Plummer M, Best N, Cowles K, Vines K, Sarkar D, Bates D, Almond R, Magnusson A (2015) coda: Output analysis and diagnostics for MCMC. R package version 0.18-1. http://cran.r-project.org/package=coda

R Core Team (2016) R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/

Rasmussen CE (2000) The infinite Gaussian mixture model. In: Solla SA, Leen TK, Müller K (eds) Advances in neural information processing systems 12. MIT Press, Cambridge, pp 554–560