Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: Day Month Year
Manuscript Category

OXFORD

Subject Section

Consensus clustering for Bayesian mixture models

Stephen Coleman [1,*], Paul DW Kirk [1, 2] and Chris Wallace [1,2*]

[1]MRC Biostatistics Unit, University of Cambridge, Cambridge, CB2 0SR, United Kingdom and
[2]Department of Medicine, University of Cambridge, Cambridge, CB2 0AW, United Kingdom.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Bayesian mixture models have attractive features and been successfully applied in a diverse range of settings. Inference of these models is normally performed using Markov-chain Monte Carlo (MCMC) methods. In high dimensions MCMC methods often explore a limited range of partitions, with a lack of overlap between chains (i.e. a lack of convergence) frequently present.
Results: We extend the ensemble method, Consensus clustering (CC), to Bayesian mixture models. We compare CC to inference performed using MCMC methods and also to `mclust`, a popular mixture model R package. We show that CC can be extended to Bayesian integrative clustering models. CC is then demonstrated on real datasets in both the single dataset and multiple dataset context. CC is shown to capture more modes in the clustering distribution than either the maximum-likelihood estimate (MLE) or any individual Markov chain. CC also reduces the computation time required when a parallel environment is present compared to Bayesian inference.
Availability: None?
Contact: stephen.coleman@mrc-bsu.cam.ac.uk
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

From defining a taxonomy of disease to creating molecular sets, grouping items can help us to understand and make decisions about complex biological problems. For example, clustering patients based upon disease characteristics and personal omics data allows the reaction and progression of individuals within a cluster to inform treatment decisions about other members of the same group. In another setting, defining gene regulatory networks can provide valuable insights into the causal mechanisms driving molecular products and may be used in diagnosis or for drug targets (Emmert-Streib *et al.*, 2014).

Clustering data is about maximising the quantity of relevant data for each individual within a specific analysis, reducing from the complexity of the entire set without contracting to the individual level. The clustering approximates complex variation within the dataset, enabling local downstream analysis and decisions upon each group rather than at the global level.

The act of identifying such groups is referred to as "cluster analysis". Traditional methods such as k-means clustering (Lloyd, 1982; Forgy, 1965) or hierarchical clustering condition upon a user inputted choice of K, the number of occupied clusters present. Normally different choices of K are compared under some metric such as silhouette or based upon the within-cluster sum of squared errors as a function of K. There exists an alternative school of clustering, model-based clustering, which embeds the cluster analysis within a formal, statistical framework. This means that choice of K is a model selection problem with all the associated literature.

In many analyses or decision making processes, understanding how certain the clustering is van be vital. For example, in clustering patients there might be individuals with almost equal probability of being allocated between a number of clusters. Knowing which individuals have uncertain membership could strongly influence decisions about treatment. However in many cluster analyses only a point clustering is estimated and thus no one is no wiser about which individuals are boundary members of clusters. Bayesian mixture models provide an uncertainty quantification that allows one to include this uncertainty in a formal manner in downstream analyses and decisions. These models and their extensions have a history of successful application to a diverse range of biological problems such as finding clusters of gene expression profiles (Medvedovic and Sivaganesan, 2002), cell types in flow cytometry (Chan *et al.*, 2008; Hejblum *et al.*, 2019) or scRNAseq experiments (Prabhakaran *et al.*, 2016), and predicting protein localisation (Crook *et al.*, 2018).

One might believe that the number of occupied clusters present, K, is a random variable that should be inferred from the data. Bayesian mixture models can treat K in this way, thereby incorporating uncertainty about K into the the allocation uncertainty and reducing some of the subjectivity of choosing K. The most common models for are the mixture of finite mixtures (Miller and Harrison, 2018, frequently implemented using Reversible Jump MCMC moves via Richardson and Green, 1997), the Dirichlet process (Ferguson, 1973; Neal, 2000) or the overfitted mixture model (Van Havre *et al.*, 2015). In these models the number of components and the component parameters are modelled jointly, in contrast with many clustering methods. Furthermore, if there is a prior information about K, this may be included due to the Bayesian nature of the model.

We write the basic mixture model for independent items $X = (x_1, \ldots, x_N)$ as

$$x_i \sim \sum_{k=1}^{K} \pi_k f(x_i|\theta_k) \qquad \text{independently for } i = 1, \ldots, N \quad (1)$$

where $f(\cdot|\theta)$ is some family of densities parametrised by θ. A common choice is the Gaussian density function, with $\theta = (\mu, \sigma^2)$. K, the number of subgroups in the population, $\{\theta_k\}_{k=1}^{K}$, the component parameters, and $\pi = (\pi_1, \ldots, \pi_K)$, the component weights are the objects to be inferred. In the context of *clustering*, such a model arises due to the belief that the population from which the random sample under analysis has been drawn consists of K unknown groups proportional to π. In this setting it is natural to include a latent *allocation variable*, $c = (c_1, \ldots, c_N)$, to indicate which group each item is drawn from. The model is then

$$\begin{aligned} p(c_i = k) &= \pi_k \quad \text{for } k = 1, \ldots, K, \\ x_i|c_i &\sim f(x_i|\theta_k) \quad \text{independently for } i = 1, \ldots, N. \end{aligned} \quad (2)$$

The joint model can then be written

$$p(X, c, K, \pi, \theta) = p(X|c, \pi, K, \theta)p(\theta|c, \pi, K)p(c|\pi, K)p(\pi|K)p(K)$$

An assumption that is frequently used is that the density of each feature is independent, with $\theta_k = (\theta_{k1}, \ldots, \theta_{kP})$ for all $k = 1, \ldots, K$. Furthermore, conditional independence is assumed between certain parameters such that the model reduces to

$$p(X, c, \theta, \pi, K) = \prod_{i=1}^{N} p(x_i|c_i, \theta_{c_i}) \prod_{i=1}^{N} p(c_i|\pi, K)p(\pi|K)p(\theta|K)p(K). \quad (3)$$

Bayesian mixture models have been extended to the multiple dataset context where they are used to perform integrative clustering. This means that as much pertinent information as possible can be included in the joint model. Multiple Dataset Integration (**MDI**, Kirk *et al.*, 2012) is an example of such a model where dataset specific clusterings are learnt, informed by common information. This is captured by the different prior on the allocation

$$p(c_{n1}, \ldots, c_{nL}) \propto \prod_{l=1}^{L} \pi_{c_{nl}l} \prod_{i=1}^{L-1} \prod_{j=i+1}^{L} (1 + \phi_{ij}\mathbb{I}(c_{ni} = c_{nj})) \quad (4)$$

for L datasets and $n = 1, \ldots, N$. ϕ_{ij} is the parameter defined by the correlation of the clusterings for the i^{th} and j^{th} datasets. As this increases more mass is placed on the common partition for i and j. Conversely, as $\phi_{ij} \to 0$ we have independent mixture models. In this way the correlation of the clustering between each pair of datasets controls how information is shared across each clustering, with more correlated datasets influencing each other and uncorrelated datasets remaining independent.

Bayesian inference of mixture models and their extensions is normally performed using Markov-chain Monte Carlo (MCMC) methods. However, MCMC methods in general are known to display problematic behaviour in high-dimensions (Robert *et al.*, 2018; Yao *et al.*, 2020; Chandra *et al.*, 2020). One of these problems is the difficulty in exploring a range of modes in the posterior distribution of the clustering. This means that there is a lack of convergence across chains and the distribution in any single chain is highly concentrated upon a very small number (often one) possible clustering(s). Another problem that emerges in large or high-dimensional datasets is that individual iterations of the MCMC sampler become slow and a long chain can become infeasible.

An alternative approach to clustering is through the use of ensembles of models. Ensembles are often better at representing modes within parameters than the individual learners (Ghaemi *et al.*, 2011). Ensembles also offer reductions in computational runtime because most ensemble methods enable use of a parallel environment to improve computation speed (Ghaemi *et al.*, 2009). Consensus clustering (Monti *et al.*, 2003) is an ensemble method for cluster analysis, previously implemented using k-means clustering in the R package `ConsensusClusterPlus` (Wilkerson *et al.*, 2010). This flavour of consensus clustering has been applied to cancer subtyping (Lehmann *et al.*, 2011; Verhaak *et al.*, 2010) and identifying subclones in single cell analysis (Kiselev *et al.*, 2017). Consensus clustering applies R independent runs of the clustering algorithm to peturbed versions of the dataset and combines the R final partitions in a *Consensus matrix* (CM) to infer a final clustering. The consensus matrix is a symmetric matrix with the $(i, j)^{th}$ entry being the proportions of model runs for which the i^{th} and j^{th} items are clustered together. For a single partition the *coclustering matrix* represents this information, being a binary matrix with the $(i, j)^{th}$ entry indicating if items i and j are allocated to the same cluster.

Data: $X = (x_1, \ldots, x_N)$
Input: A resampling scheme *Resample*
A clustering algorithm *Cluster*
Number of resampling iterations S
Set of cluster numbers to try $\mathcal{K} = \{K_1, \ldots, K_{max}\}$
Output: A predicted clustering, $\hat{Y}$
The predicted number of clusters present $\hat{K}$
begin
  for $K \in \mathcal{K}$ **do**
    `/* initialise an empty Consensus Matrix */`
    $\mathbf{M}^{(K)} \leftarrow \mathbf{0}_{N \times N}$;
    for $s = 1$ **to** S **do**
      $X^{(s)} \leftarrow Resample(X)$;
      `/* Cluster the peturbed dataset, represented in a coclustering matrix */`
      $\mathbf{B}^{(s)} \leftarrow Cluster(X^{(s)}, K)$;
      $\mathbf{M}^{(K)} \leftarrow \mathbf{M}^{(K)} + \mathbf{B}^{(s)}$;
    end
    $\mathbf{M}^{(K)} \leftarrow \frac{1}{S}\mathbf{M}^{(K)}$;
  end
  $\hat{K} \leftarrow$ best $K \in \mathcal{K}$ based upon all $\mathbf{M}^{(K)}$;
  $\hat{Y} \leftarrow$ partition X based upon $\mathbf{M}^{(\hat{K})}$;
end

Algorithm 1: Consensus Clustering algorithm

We extend consensus clustering to Bayesian mixture models. We show via simulation that ensembles consisting of short chains are sufficient to

uncover meaningful structure in a number of scenarios including some within which a Gibbs sampler becomes trapped in individual modes for any reasonable length of runtime. The chains are both short and independent, thus their individual runtime is far shorter than the chains traditionally used for Bayesian inference and may also be run in parallel. This means that consensus clustering of Bayesian mixture models offers significant reductions in runtime without sacrifices in performance. As the ensemble can describe multiple modes, the uncertainty present in the consensus matrix can be more representative of the data than the individual modes captured by any single chain.

We then apply our algorithm to the multiple dataset setting and the extension of Bayesian mixture models, Multiple Dataset Integration (MDI). We show on three datasets from the original MDI paper that Consensus clustering performs similarly to Bayesian inference of this model, and then using more modern, larger data that Consensus clustering enables implementation of such models in scenarios where the problem of long runtimes and poor mixing previously discouraged this.

2 Methods

2.1 Consensus clustering for Bayesian mixture models

We extend CC to use Bayesian mixture models as the underlying model. This offers the ability to include a prior distribution on parameters and inference of the number of occupied clusters present, maintaining two of the key attractions of Bayesian model-based clustering while losing the asymptotic guarantees of Bayesian inference. In this case the algorithm is simplified as it is not necessary to try a range of possible clusters present. Furthermore, the dataset is not perturbed as described in algorithm 1 for two reasons. First, the overfit mixture can capture individuals in singletons and thus is more robust to outliers than k-means. Secondly, the MCMC method driving each model offers diversity of partitions when combined with different initialisations. The method is described in algorithm 2.

Data: $X = (x_1, \ldots, x_N)$
Input: A Bayesian mixture model with membership vector
$c = (c_1, \ldots, c_N)$
A clustering algorithm that generates samples *Cluster*
The number of chains to run, S
The number of iterations within each chain, R
Output: A predicted clustering, $\hat{Y}$
The consensus matrix $\mathbf{M}$
begin
```
/* initialise an empty Consensus Matrix */
```
$\mathbf{M} \leftarrow \mathbf{0}_{N \times N}$;
for $s = 1$ **to** S **do**
```
/* set the random seed controlling
   initialisation and MCMC moves      */
```
$set.seed(s)$;
```
/* initialise a random partition on X
   drawn from the prior distribution  */
```
$Y_{(0,s)} \leftarrow Initialise(X)$;
for $r = 1$ **to** R **do**
```
/* generate a markov chain for the
   membership vector                  */
```
$Y_{(r,s)} \leftarrow Cluster(c, r)$;
end
```
/* create a coclustering matrix from the
   R^th sample                        */
```
$\mathbf{B}^{(s)} \leftarrow Y_{(R,s)}$;
$\mathbf{M} \leftarrow \mathbf{M} + \mathbf{B}^{(s)}$;
end
$\mathbf{M} \leftarrow \frac{1}{S}\mathbf{M}$;
$\hat{Y} \leftarrow$ partition X based upon $\mathbf{M}$;
end

Algorithm 2: Consensus Clustering for Bayesian mixture models

2.2 Predicting a clustering from CMs

We use the `maxpear` function (Fritsch *et al.*, 2009) from the R package `mcclust` (Fritsch, 2012) to infer a point clustering from CMs. This function was designed to perform inference upon the posterior similarity matrix (**PSM**) from the samples of a single long chain (this is analogous to a CM, except the partitions are all drawn from a single Markov chain), predicting a clustering that has maximum posterior expected adjusted Rand index (**ARI**, Hubert and Arabie, 1985) with the true clustering. In the context of the CM, the function does not have this interpretation. However, the method produces a kind of "average clustering" based upon all sampled partitions. We believe this is a sensible method for predicting a point clustering from the CM, effectively averaging over each learner in the ensemble.

2.3 Data generating mechanism

The data generating model is a finite mixture model (as per equation 2) with independent features. Within this model there exist "irrelevant features" (Law *et al.*, 2003) that have global parameters rather than component specific parameters. The generating model has a fixed number of clusters chosen by us and is thus, extending from equation 3,

$$p(X, c, \theta, \pi|K) = \prod_{i=1}^{N}\prod_{p=1}^{P} p(x_{ip}|c_i, \theta_{c_i p})^{(1-\phi_p)} p(x_{ip}|\theta_p)^{\phi_p} \times \prod_{i=1}^{N} p(c_i|\pi, K) p(\pi|K) p(\theta|K)$$

with $\phi_p = 1$ indicating that the p^{th} feature is relevant. A description of the algorithm implementing this is given in algorithm **??**.

In the simulation study described here, the model is a mixture of *Gaussian* distributions and thus $\theta_{kp} = (\mu_{kp}, \sigma^2_{kp})$. The prior distributions used on the mixture parameters are

$$\pi \sim \text{Dirichlet}(\alpha, \ldots, \alpha), \qquad \mu_{kp} \sim \mathcal{N}(\xi, \kappa), \qquad \sigma^2 \sim \Gamma^{-1}(a, b).$$

The total number of occupied and empty components is set to $K_{max} = 50$. This and the choice of priors are the defaults in the software provided by Mason *et al.* (2016).

2.4 Performance quantification

We use the ARI as our metric for the quality of the point clustering inferred by each method, comparing this estimate with the generating labels. This is a measure of "predictive performance", the ability of the methods to infer a single partition that and its similarity to the truth. We also attempt to summarise the uncertainty quantification from each by computing the Frobenius Norm between the true coclustering matrix and the

- consensus matrix for consensus clustering,
- posterior similarity matrix for the Bayesian inference, and
- coclustering matric for `mclust`.

The Frobenius Norm will provide some information if the above matrices correspond at all to the true coclustering matrix, but if no method has performed well then this Norm will reward the *singleton solution* wherein all items are allocated to individual clusters. This means that a visual

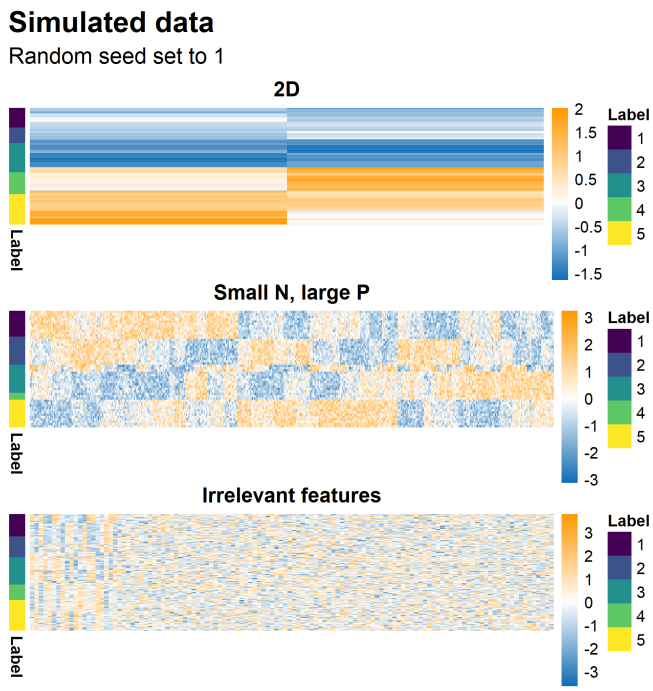

Fig. 1. Example of generated datasets. The low-dimensional dataset should enable proper mixing of chains in the MCMC and is less likely to have a local mode that traps `mclsut`. The small N, large P case has clear structure (observable by eye). This is intended to highlight the problems of poor mixing due to high dimensions even when the generating labels are quite identifiable. In the irrelevant features case the structure is clear in the relevant features (on the lefthand side of this heatmap). This setting is intended to test how sensitive each approach is to noise.

inspection of the PSMs and CMs is also required. As `mclust` provides only a point estimate the ARI between this and the truth will contain the required information.

The runtime of each MCMC chain is calculated using the terminal command `time`, measured in milliseconds.

2.5 Bayesian model convergence

To assess within-chain convergence of our Bayesian inference we use the Geweke Z-score statistic (Geweke *et al.*, 1991). Of the chains that appear to behave properly we then asses across-chain convergence using $\hat{R}$ (Gelman *et al.*, 1992) and the recent extension provided by Vats and Knudson (2018).

If a chain has reached its stationary distribution the Geweke Z-score statistic is expected to be normally distributed. Normality is tested for using a Shapiro-Wilks test (Shapiro and Wilk, 1965). If a chain fails this test (i.e. the associated p-value is less than 0.05), we assume that is has not achieved stationarity and is excluded from the remainder of the analysis.

The Vats and Knudson extension of $\hat{R}$ is a summary statistic for the entire chain; this is the primary indicator of failure for convergence, but a visualisation of the original $\hat{R}$ diagnostic is also considered.

3 Examples

3.1 Simulations

We compare consensus clustering of Bayesian mixture models to a traditional inference using several long chains of 1 million iterations (thinning to every thousandth) and an maximum-likelihood estimator as implement in the R package `mclust` (Scrucca *et al.*, 2016). These are compared within a range of simulations described in table 2. In this article we describe the results of three of these in more detail with the remainder described in the supplementary material.

We test different scenarios that change various parameters in this model. The scenarios tested and there defining parameters are shown in table 2. In this case the number of relevant features (P_s) is $\sum_p \phi_p$, and $P_n = P - P_s$.

Table 1. Parameters defining the simulation scenarios as used in generating data and labels. Results for the Simple 2D, the first Small N, large P and final Irrelevant features scenarios are shown in this report, please see the supplementary material for additional results. The number of relevant features (P_s) is $\sum_p \phi_p$, and $P_n = P - P_s$.

Scenario	N	P_s	P_n	K	Δ_μ	σ^2	π
2D	100	2	0	5	3.0	1	$(\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5})$
Irrelevant features	200	20	100	5	1.0	1	$(\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5})$
Small N, large P	50	500	0	5	1.0	1	$(\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5})$

Table 2. Parameters defining the simulation scenarios as used in generating data and labels.

The examples included in this article are

- a low-dimensional dataset,
- a wide dataset representative of the *small N, large P* paradigm prevalent in genetics, and
- a dataset with a large number of irrelevant features.

The first of these is expected to be the setting where `mclust` and the individual long chains behave well. In the other simulations increasing dimensionality means that mixing problems can emerge and the chains become liable to being trapped in individual modes. Within each simulation 100 datasets are generated. To each of these datasets, the following are applied

- `mclust` (for a range of possible K),
- 10 chains of 1 million iterations, thinning to every thousandth sample for the overfitted Bayesian mixture model, and
- a variety of consensus clustering ensembles defined by inputs of S chains and R iterations within each chain (see algorithm 2) with $S \in \{1, 10, 30, 50, 100\}$ and $R \in \{1, 10, 100, 1000, 10000\}$.

In theory we would expect the Bayesian chains to explore a common posterior distribution, but the practice sees chains become trapped in distinct modes in different scenarios. We believe that the mode within which the greatest number of chains become trapped would be that which is used to perform the inference in a real application (and longer chains did not solve the problem). As part of a pipeline where such analyses have to happen $12 \times 100 = 1,200$ times, we pool the Bayesian samples into a single PSM as manually assessing which mode is dominant. As visual inspection of the PSMs indicates that the disagreement between modes tends to be one of

- several clusters are merged or
- a cluster is represented by two or more components,

each of the modes tends to have large overlap with all others. This means that the clustering inferred from the PSM created from the samples pooled across all stationary chains will represent the most popular mode and the ARI will not be unduly inflated. The Frobenius norm between the PSM and the true coclustering will be less representative of a single chain as more modes will be represented and therefore greater uncertainty than any single chain. This is inline with our previous assertion that the Frobenius statistic is more an indicator that should be used along with visual assessment of

examples of the PSMs and CMs from a simulation rather than sufficiently informative in and of itself.

3.2 Yeast data

We analyse the 551 yeast genes considered in the original MDI paper (Kirk *et al.*, 2012) in an integrative setting. This consists of three datasets

- ChIP data from Harbison *et al.* (2004),
- binary PPI data obtained from BioGRID (Stark *et al.*, 2006) and
- a gene expression time course dataset from Granovskaia *et al.* (2010).

The clustering in these datasets is modelled using MDI. We use 5 chains of 625,000 iterations (the number of iterations performed in 36 hours on the slowest chain) and consensus clustering of 100 chains of 500 iterations. The ChIP and PPI data are modelled using mixtures of multinomial distributions and the time course data by a mixture of Gaussian processes (with dataset density choice following the original paper by Kirk *et al.*, 2012).

4 Results

4.1 Simulations

A summary of the results for a selection of the simulation scenarios are shown in table 3 and figure 2. In the strong signal scenarios (i.e. noise free and clearly distinguishable generating clusters), `mclust` performs very well, correctly identifying the true structure However, a large number of irrelevant features completely erodes the ability of `mclust` to uncover meaningful structure. The pooled samples from multiple long chains performs consistently very well. This is not surprising as the pooling effect means that any multi-modality present in the data does not present any degree of problem. In this case the pooled samples, themselves a consensus of 10 models, acts as an upper bound on the more practical implementations of consensus clustering. Consensus clustering does consistently uncover structure in the data. With sufficiently large ensembles and chain depth, consensus clustering is close to the pooled Bayesian samples in predictive performance. In terms of the Frobenius norm, many of the consensus clustering results have significant overlap with the pooled long chains.

Table 3. Mean ARI for 100 datasets within three simulation scenarios between the generating labels and the predicted clustering for a subset of methods. CC(r, s) indicates consensus clustering using the r^{th} sample from s chains.

Model	2D	Small N, large P	Irrelevant features
Mclust	0.970	1.000	0.000
Bayesian (Pooled)	0.669	0.999	0.946
CC(10, 10)	0.362	0.997	0.385
CC(100, 10)	0.844	0.346	0.999
CC(100, 50)	0.873	0.517	0.999
CC(100, 100)	0.879	0.576	0.999
CC(10000, 10)	0.348	0.999	0.935
CC(10000, 100)	0.570	0.999	0.944

It can be seen from table 3 and figures 2 that

Figure 3 shows an example of different chains becoming trapped in different modes and failing to explore a common partition space. In the simulations shown here the overlap between the modes and signal in the data is clear enough that one can pick the true clustering structure and select the chain that best represents this, but in other

Figure 4 reveals the gains in computation runtime achieved by consensus clustering when a parallel environment is available. If a single iteration

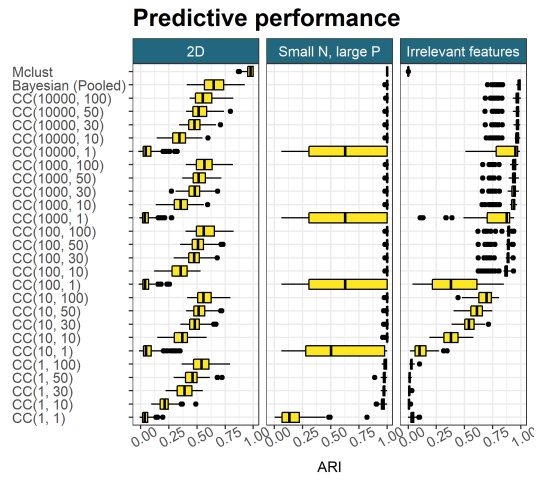

Fig. 2. Model performance in a subset of simulation scenarios.

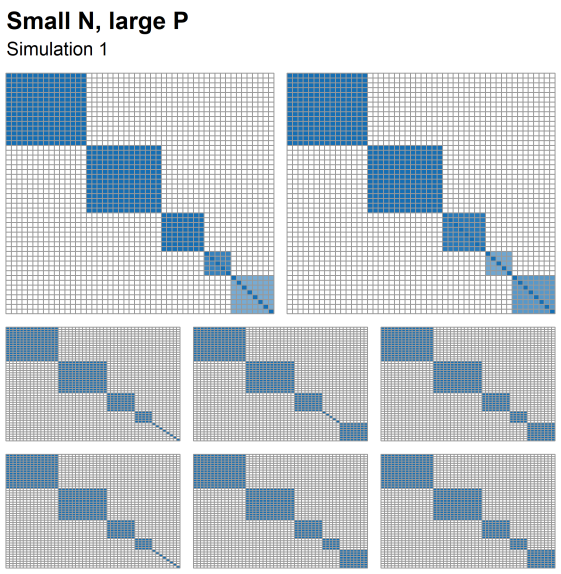

Fig. 3. In the first row the PSM of the pooled Bayesian samples is compared to the CM for CC(100, 50), with a common ordering of rows and columns in both heatmaps. There is very little difference between these two objects. In the next two rows the PSMs constructed from 6 of the long chains that passed the stationarity test are displayed. Three different modes emerge across the different chains.

requires t seconds, running 100 chains of length 100 using 8 cores takes $1,300t$ compared to 5 chains of 10,000 iterations taking $10,000t$.

4.2 Yeast

We performed a Gene Ontology (GO) enrichment analysis of our inferred clusters using the `Bioconductor` packages `biomaRt` (Durinck *et al.*, 2005, 2009) and `clusterProfiler` (Yu *et al.*, 2012). The long chains

5 Discussion

In this article, we have extended Consensus clustering to Bayesian mixture models. This overcomes the problem of mixing for MCMC based clustering methods and also improves the speed at which an analysis can

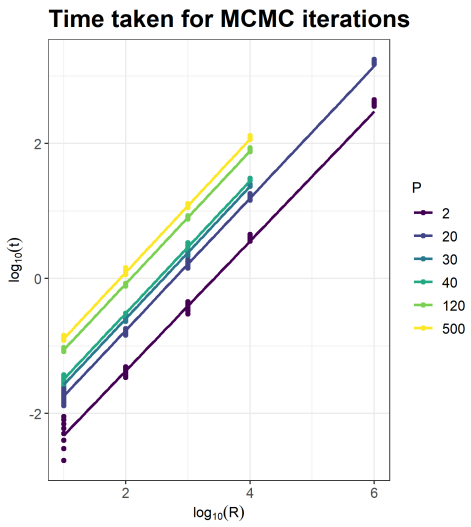

Fig. 4. The time taken for different numbers of iterations of MCMC moves. The relationship between chain length, R, and the time taken is linear, with a change of intercept for different dimensions.

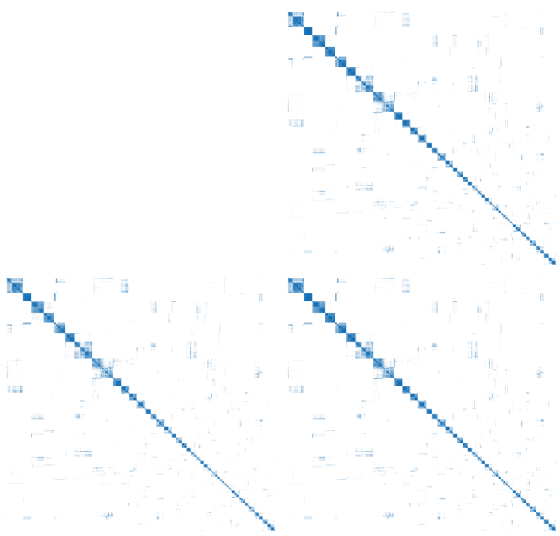

Fig. 5. The chain length, R, and ensemble size S, were decided by comparing the consensus matrices for different values of (R, S). Here we show an example for the time course data where $R = 100$ in the top row and $R = 500$ in the second, and $S = 50$ in the first column and $S = 100$ in the second. These matrices appear identical, so we stop increasing the chain length or ensemble size.

be performed. The proposed method has demonstrated good performance on simulation studies, having similar ability to uncover structure as more traditional approaches. It has better ability to represent several modes in the data than individual chains and is significantly faster, while retaining the ability to infer K, the number of occupied components present.

We expect that our method will be useful to researchers analysing high-dimensional data where the runtime of MCMC methods becomes too onerous and multi-modality is more likely to be present.

Could the early stopping of the MCMC algorithm be a form of regularisation, making the model more robust to misspecification? In line with early stopping from Deep Learning (see Morgan and Bourlard, 1990),

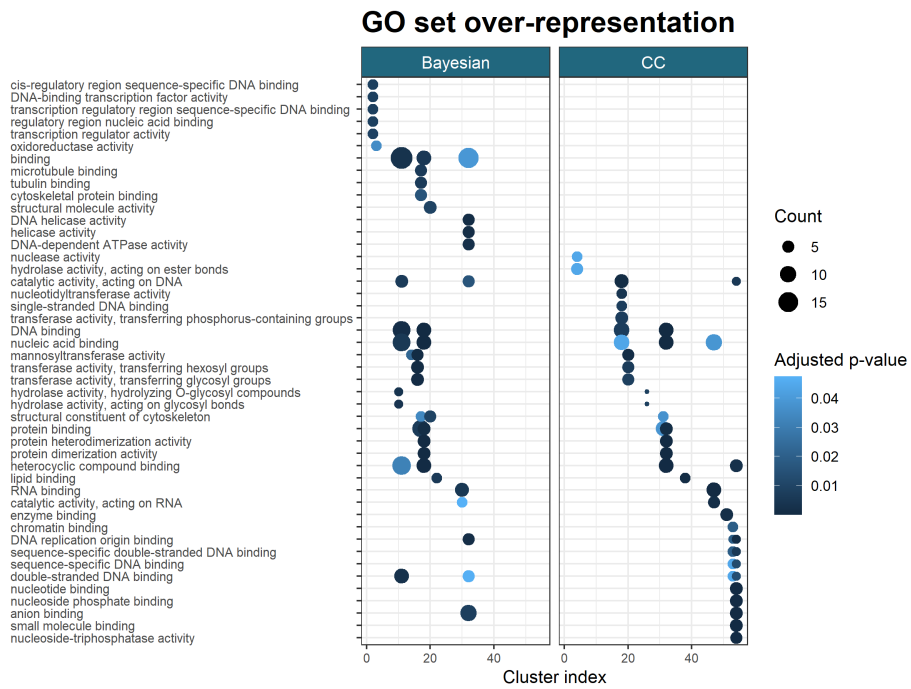

Fig. 6. GO term over representation in the clusters using the molecular function ontology.

regularisation and misspecification idea from Miller and Dunson (2018); Cai *et al.* (2020).

6 Conclusion

(Table **??**) Text Text Text Text Text Text Text Text Text

1. this is item, use enumerate
2. this is item, use enumerate
3. this is item, use enumerate

Acknowledgements

Text Text Text Text Text Text Text Text. nt to know about text text text text

Funding

This work has been supported by the... Text Text Text Text.

References

Cai, D. *et al.* (2020). Finite mixture models are typically inconsistent for the number of components. *arXiv preprint arXiv:2007.04470*.

Chan, C. *et al.* (2008). Statistical mixture modeling for cell subtype identification in flow cytometry. *Cytometry Part A: The Journal of the International Society for Analytical Cytology*, **73**(8), 693–701.

Chandra, N. K. *et al.* (2020). Bayesian clustering of high-dimensional data. *arXiv preprint arXiv:2006.02700*.

Crook, O. M. *et al.* (2018). A bayesian mixture modelling approach for spatial proteomics. *PLoS computational biology*, **14**(11), e1006516.

Durinck, S. *et al.* (2005). Biomart and bioconductor: a powerful link between biological databases and microarray data analysis. *Bioinformatics*, **21**(16), 3439–3440.

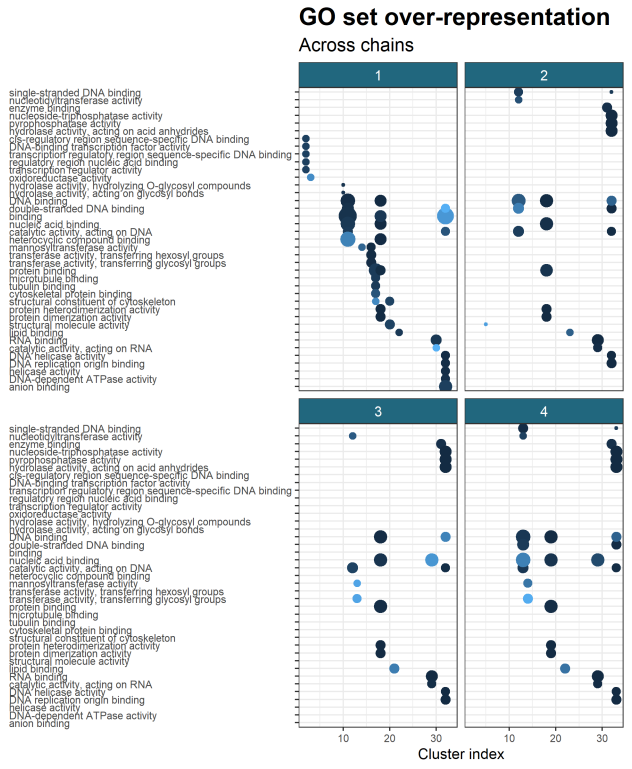

Fig. 7. The enriched GO terms disagree across chains.

Durinck, S. *et al.* (2009). Mapping identifiers for the integration of genomic datasets with the r/bioconductor package biomart. *Nature protocols*, **4**(8), 1184.

Emmert-Streib, F. *et al.* (2014). Gene regulatory networks and their applications: understanding biological and medical problems in terms of networks. *Frontiers in cell and developmental biology*, **2**, 38.

Ferguson, T. S. (1973). A bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230.

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *biometrics*, **21**, 768–769.

Fritsch, A. (2012). *mcclust: Process an MCMC Sample of Clusterings*. R package version 1.0.

Fritsch, A. *et al.* (2009). Improved criteria for clustering based on the posterior similarity matrix. *Bayesian analysis*, **4**(2), 367–391.

Gelman, A. *et al.* (1992). Inference from iterative simulation using multiple sequences. *Statistical science*, **7**(4), 457–472.

Geweke, J. *et al.* (1991). *Evaluating the accuracy of sampling-based approaches to the calculation of posterior moments*, volume 196. Federal Reserve Bank of Minneapolis, Research Department Minneapolis, MN.

Ghaemi, R. *et al.* (2009). A survey: clustering ensembles techniques. *World Academy of Science, Engineering and Technology*, **50**, 636–645.

Ghaemi, R. *et al.* (2011). A review: accuracy optimization in clustering ensembles using genetic algorithms. *Artificial Intelligence Review*, **35**(4), 287–318.

Granovskaia, M. V. *et al.* (2010). High-resolution transcription atlas of the mitotic cell cycle in budding yeast. *Genome biology*, **11**(3), 1–11.

Harbison, C. T. *et al.* (2004). Transcriptional regulatory code of a eukaryotic genome. *Nature*, **431**(7004), 99–104.

Hejblum, B. P. *et al.* (2019). Sequential dirichlet process mixtures of multivariate skew t-distributions for model-based clustering of flow cytometry data. *The Annals of Applied Statistics*, **13**(1), 638–660.

Hubert, L. and Arabie, P. (1985). Comparing partitions. *Journal of classification*, **2**(1), 193–218.

Kirk, P. *et al.* (2012). Bayesian correlated clustering to integrate multiple datasets. *Bioinformatics*, **28**(24), 3290–3297.

Kiselev, V. Y. *et al.* (2017). Sc3: consensus clustering of single-cell rna-seq data. *Nature methods*, **14**(5), 483–486.

Law, M. H. *et al.* (2003). Feature selection in mixture-based clustering. In *Advances in neural information processing systems*, pages 641–648.

Lehmann, B. D. *et al.* (2011). Identification of human triple-negative breast cancer subtypes and preclinical models for selection of targeted therapies. *The Journal of clinical investigation*, **121**(7), 2750–2767.

Lloyd, S. (1982). Least squares quantization in pcm. *IEEE transactions on information theory*, **28**(2), 129–137.

Mason, S. A. *et al.* (2016). Mdi-gpu: accelerating integrative modelling for genomic-scale data using gp-gpu computing. *Statistical applications in genetics and molecular biology*, **15**(1), 83–86.

Medvedovic, M. and Sivaganesan, S. (2002). Bayesian infinite mixture model based clustering of gene expression profiles. *Bioinformatics*, **18**(9), 1194–1206.

Miller, J. W. and Dunson, D. B. (2018). Robust bayesian inference via coarsening. *Journal of the American Statistical Association*.

Miller, J. W. and Harrison, M. T. (2018). Mixture models with a prior on the number of components. *Journal of the American Statistical Association*, **113**(521), 340–356.

Monti, S. *et al.* (2003). Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine learning*, **52**(1-2), 91–118.

Morgan, N. and Bourlard, H. (1990). Generalization and parameter estimation in feedforward nets: Some experiments. In *Advances in neural information processing systems*, pages 630–637.

Neal, R. M. (2000). Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, **9**(2), 249–265.

Prabhakaran, S. *et al.* (2016). Dirichlet process mixture model for correcting technical variation in single-cell gene expression data. In *International Conference on Machine Learning*, pages 1070–1079.

Richardson, S. and Green, P. J. (1997). On bayesian analysis of mixtures with an unknown number of components. *Journal of the Royal Statistical Society: series B*, **59**(4), 731–792.

Robert, C. P. *et al.* (2018). Accelerating mcmc algorithms. *Wiley Interdisciplinary Reviews: Computational Statistics*, **10**(5), e1435.

Scrucca, L. *et al.* (2016). mclust 5: clustering, classification and density estimation using Gaussian finite mixture models. *The R Journal*, **8**(1), 289–317.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, **52**(3/4), 591–611.

Stark, C. *et al.* (2006). Biogrid: a general repository for interaction datasets. *Nucleic acids research*, **34**(suppl_1), D535–D539.

Van Havre, Z. *et al.* (2015). Overfitting bayesian mixture models with an unknown number of components. *PloS one*, **10**(7), e0131739.

Vats, D. and Knudson, C. (2018). Revisiting the gelman-rubin diagnostic. *arXiv preprint arXiv:1812.09384*.

Verhaak, R. G. *et al.* (2010). Integrated genomic analysis identifies clinically relevant subtypes of glioblastoma characterized by abnormalities in pdgfra, idh1, egfr, and nf1. *Cancer cell*, **17**(1), 98–110.

Wilkerson *et al.* (2010). Consensusclusterplus: a class discovery tool with confidence assessments and item tracking. *Bioinformatics*, **26**(12), 1572–1573.

Yao, Y. *et al.* (2020). Stacking for non-mixing bayesian computations: The curse and blessing of multimodal posteriors. *arXiv preprint arXiv:2006.12335*.

Yu, G. *et al.* (2012). clusterprofiler: an r package for comparing biological themes among gene clusters. *OMICS: A Journal of Integrative Biology*, **16**(5), 284–287.