Biostatistics (2023) **24**, 1, *pp.* 68–84
doi: 10.1093/biostatistics/kxab025
Advance Access publication on August 7, 2021

A sparse negative binomial mixture model for clustering RNA-seq count data

YUJIA LI[†], TANBIN RAHMAN[†]

Department of Biostatistics, University of Pittsburgh, Pittsburgh, PA 15261, USA

TIANZHOU MA

Department of Epidemiology and Biostatistics, University of Maryland, College Park, MD 20742, USA

LU TANG, GEORGE C. TSENG[*]

Department of Biostatistics, University of Pittsburgh, Pittsburgh, PA 15261, USA

ctseng@pitt.edu

Summary

Clustering with variable selection is a challenging yet critical task for modern small-n-large-p data. Existing methods based on sparse Gaussian mixture models or sparse K-means provide solutions to continuous data. With the prevalence of RNA-seq technology and lack of count data modeling for clustering, the current practice is to normalize count expression data into continuous measures and apply existing models with a Gaussian assumption. In this article, we develop a negative binomial mixture model with lasso or fused lasso gene regularization to cluster samples (small n) with high-dimensional gene features (large p). A modified EM algorithm and Bayesian information criterion are used for inference and determining tuning parameters. The method is compared with existing methods using extensive simulations and two real transcriptomic applications in rat brain and breast cancer studies. The result shows the superior performance of the proposed count data model in clustering accuracy, feature selection, and biological interpretation in pathways.

Keywords: Cluster analysis; Feature selection; Gaussian mixture model; Sparse K-means.

1. Introduction

Cluster analysis is a powerful exploratory tool for high-dimensional data. In omics applications, many classical methods such as K-means clustering, hierarchical clustering, self-organizing map (SOM), and model-based clustering have been widely used. In transcriptomic data measured in microarray experiments, for example, genes can be clustered into gene modules that suggest coregulated or coexpressed genes with related biological functions. In many complex diseases, patients can be clustered to identify novel disease subtypes with distinct disease mechanism or drug responses, which often forms the basis for personalized medicine and such sample clustering is the focus of this article. When clustering such high-dimensional

[*]To whom correspondence should be addressed.
[†]Yujia Li and Tanbin Rahman contributed equally to this work.

data, methods such as hierarchical clustering and SOM are heuristic in nature while model-based clustering assumes data to come from a mixture distribution of multiple clusters. Although the heuristic clustering algorithms are easy to implement and popular, they lack formal inference. Model-based clustering, on the other hand, imposes distributional assumptions on data observations, and can allow rigorous inference, biological interpretation, and prediction of future samples. For microarray data, model-based clustering has been found with superior performance compared to heuristic methods such as hierarchical clustering or SOM (Thalamuthu *and others*, 2006).

When clustering patients in omics data with thousands of genes, it is biologically reasonable to assume that only a small subset of genes (e.g., 50–200 genes) are informative (i.e., relevant to sample clustering). For this purpose, Pan and Shen (2007) proposed a Gaussian mixture model-based clustering with lasso penalty. Witten and Tibshirani (2010) proposed a sparse K-means algorithm extended from K-means for feature selection. Fop and Murphy (2018) provided thorough reviews of variable selection methods of clustering high-dimensional data. These methods can serve well for clustering continuous transcriptomic data from the gradually outdated microarray platforms. In the past 10 years, the rapid development of RNA sequencing (RNA-seq) technology has revolutionized the transcriptomic research. Unlike the continuous fluorescent measurements from microarray experiments, one critical feature of RNA-seq is the (discrete) count-based data after the alignment of millions of sequencing reads. In the literature, a common practice is to transform RNA-seq count data into continuous normalized values (e.g., transcripts per million (TPM) or counts per million (CPM) values) and directly apply methods that have been developed for microarray data. This leads to a significant loss of information, particularly for genes with lower counts. Methods directly modeling count data are expected to better fit the experimental data generation process, capture essential data characteristics, and thus perform better in clustering accuracy and gene selection.

In the literature, Si *and others* (2014) proposed a count-based model for clustering genes without considering variable selection. Witten (2011) proposed to cluster RNA-seq data by hierarchical clustering where the dissimilarity matrix is calculated by the Poisson assumption. Dey *and others* (2017) proposed grade of membership models, a soft clustering approach allowing each sample to have membership in multiple clusters. These two methods neither consider overdispersion of count data nor achieve gene selection. In this article, we focus on the problem of clustering samples with variable (gene) selection using transcriptomic data from RNA-seq. The data are count-based and usually contain $\sim$50–200 samples and $\sim$5000–10 000 genes after proper gene filtering, which necessitates effective variable selection while performing clustering. Our approach directly deals with the count data using a mixture of negative binomial models via GLM framework, without loss of information from transformation to continuous data. Further, we perform the variable selection by lasso or fused lasso penalty to shrink the cluster-specific means of each feature towards its global mean across all clusters or towards common means of subsets of clusters. The aforementioned existing methods and our proposed model belong to the finite mixture models for (frequentist) model-based cluster analysis. A potential limitation of this approach is the separate point estimations of the number of clusters, variable selection, and cluster assignment performed at different stages, rather than a unified model. Bayesian clustering models, in contrast, incorporate uncertainties at these different levels within a single unified framework (Binder, 1978; Richardson and Green, 1997). On the other hand, Bayesian methods face computational challenges in high-dimensional variable selection (usually only applicable to up to hundreds of genes), potentially can be sensitive to prior specification, and still require a justifiable procedure to summarize the simulated posteriors into the final decision of the parameters (Wade *and others*, 2018). Since development and a full comparison of the Bayesian counterpart of finite mixture models are beyond the scope of this article, we will only evaluate and compare popular frequentist approaches in this article (see discussion in the final section).

The article is structured as follows. In Section 2, we first summarize two existing count-based methods without variable selection (Section 2.1.1). Next, we review two existing methods with variable selection using continuous normalized values, sparse Gaussian mixture clustering and sparse K-means (Section

2.1.1), and then propose the sparse negative binomial mixture clustering model (Section 2.2). Specifically, we propose two versions of clustering models with lasso (Section 2.2.1) or fused lasso penalty (Section 2.2.2). Bayesian information criterion (BIC) for model selection (Section 2.3) and performance benchmarks (Section 2.4) will be presented. Section 3 will cover extensive simulations to benchmark and justify the improved performance of the proposed method. In Section 4, two real applications using RNA-seq data from rat brain and breast cancer subtype studies will be evaluated to demonstrate improved clustering accuracy and gene selection. Section 5 contains the final conclusion and discussion.

2. EXISTING AND PROPOSED METHODS

To simplify discussion hereafter, we will abbreviate the two existing count-based methods without variable selection (i.e., Witten, 2011; Dey *and others*, 2017) as "PoiClaClu" and "CountClust," respectively according to the names of their R packages. Also, we abbreviate the sparse Gaussian clustering model as "sgClust" and abbreviate the sparse K-means method as "sKmeans." Our proposed method, the sparse negative binomial model-based clustering with lasso or fused lasso penalty, is abbreviated as "snbClust.lasso" and "snbClust.fused," respectively. Throughout the article, we assume the raw sequencing reads from RNA-seq experiment are properly preprocessed, aligned, and summarized. Denote by y_{ij} the observed counts for gene j ($1 \le j \le G$) in sample i ($1 \le i \le n$). Our proposed snbClust model, PoiClaClu, and CountClust will utilize the count data as input. For sgClust and sKmeans, Gaussian assumption is explicitly or implicitly assumed and only continuous input data are allowed. We will generate log-transformed (base 10) CPM values using the edgeR package (Robinson *and others*, 2010). The resulting log-CPM continuous values are denoted as x_{ij} and are the input data for sgClust and sKmeans.

2.1. *Existing methods using count or continuous input data*

2.1.1. *Methods using count input data:* To our knowledge, PoiClaClu and CountClust are the only two methods based on count data for clustering RNA-seq samples. Witten (2011) proposed to cluster RNA-seq data by hierarchical clustering where the dissimilarity matrix is calculated by the Poisson assumption. They assumed $y_{ij} \sim \text{Poisson}\left(u_{ij}d_{ij}\right)$ and $u_{ij} = s_i g_j$, where s_i is the pre-estimated normalization size factor of the ith sample, g_j is gene-specific expression level and d_{ij} is the class-specific scaling factor for sample i and gene j. For each pair of samples i and i', under the null hypothesis $H_0 : d_{ij} = d_{i'j} = 1, j = 1, \ldots, p$, the resulting modified log-likelihood ratio statistic $\sum_{j=1}^{p}\left(\hat{u}_{ij} + \hat{u}_{i'j} - \hat{u}_{ij}\hat{d}_{ij} - \hat{u}_{i'j}\hat{d}_{i'j} + y_{ij}\log\hat{d}_{ij} + y_{i'j}\log\hat{d}_{i'j}\right)$ can be used as a measure of dissimilarity, where $\hat{d}_{ij} = \frac{y_{ij}+\beta}{\hat{u}_{ij}+\beta}$ and $\hat{d}_{i'j} = \frac{y_{i'j}+\beta}{\hat{u}_{i'j}+\beta}$ are posterior means for d_{ij} and $d_{i'j}$ under Gamma(β, β) priors. Hierarchical clustering can be applied based on the estimated dissimilarity matrix. This method is implemented in R package "PoiClaClu."

Dey *and others* (2017) proposed grade of membership models, a soft clustering approach allowing each sample to have membership in multiple clusters. Specifically, they assume $(y_{i1}, y_{i2}, \ldots, y_{iG}) \sim$ multinomial $(y_{i+}, p_{i1}, p_{i2}, \ldots, p_{iG})$, where $y_{i+} = \sum_{j=1}^{G} y_{ij}$ and $p_{ij} = \sum_{k=1}^{K} q_{ik}\theta_{kj}$. θ_{kj} is a probability vector for each cluster k and q_{ik} is the proportion of its reads coming from cluster k for sample i. Although the original method only aims to derive soft-assignment vector $q_{i1}, q_{i2} \ldots, q_{iK}$, we can perform a hard assignment for evaluation by assigning sample i to cluster k with the highest probability. The method can be implemented in R package "CountClust."

2.1.2. *Methods using continuous input data:* Sparse Gaussian model-based clustering model (sgClust). Pan and Shen (2007) proposed a penalized likelihood approach by extending conventional Gaussian mixture models with a penalty term for feature selection. By assuming zero mean for each gene vector, the penalty term is simply the sum of l_1-norm of all cluster means in all genes. Specifically,

$\log L(\theta; x) = \sum_{i=1}^{n} \log[\sum_{k=1}^{K} p_k f_k(x_i; \theta_k)] - \lambda b(\theta)$ is the likelihood to be maximized, where $f_k(x_i; \theta_k)$ is the density function of multivariate normal distribution for cluster k with cluster means and variances $\theta_k = \{\mu_k, \Sigma_k\}$ and $x_i = (x_{i1}, \ldots, x_{iG})$, p_k is the mixing probability of the kth cluster and $b(\theta) = \sum_{j=1}^{G} \sum_{k=1}^{K} |\mu_{jk}|$ is the penalty term for regularization. For simplicity, we note that this method assumes diagonal (i.e., independence across genes) and equal covariance matrices across all clusters (i.e., $\Sigma_k = \text{diag}\,(\sigma_1, \sigma_2, \ldots, \sigma_G)\,, \forall k$). For a give gene j, if $u_{jk} = 0$ for all k $(1 \leq k \leq K)$, then the jth gene does not contribute to the clustering. λ is the tuning parameter controlling the number of genes contributing to the clustering, which is determined by BIC (see Section 2.3 for details). In real applications, each gene vector is standardized to zero mean before applying the method. Since no R package is available to the best of our knowledge, we wrote the R functions to carry out the algorithm and included it in our R package.

Sparse K-means clustering (sKmeans). K-means clustering is a classical, efficient and effective clustering algorithm that seeks to minimize the within cluster sum-of-squares (WCSS). The method is related to Gaussian mixture model-based clustering with equal and spherical covariance matrices in each cluster (Tseng, 2007). In calculating distances for WCSS, traditional K-means adopts equal contribution from all gene features. In genomic applications, however, the input data set contains thousands of genes and biologically only a small set of "informative genes" are relevant to sample clustering. Witten and Tibshirani (2010) proposed a sparse K-means approach to allow feature selection and to improve clustering performance. While K-means minimizes the WCSS, sparse K-means equivalently seeks to maximize the between cluster sum of squares with gene-specific weight w_j for gene j and an l_1 lasso penalty on w_j. Specifically, sparse K-means seeks to maximize $\sum_{j=1}^{G} w_j \cdot BCSS_j = \sum_{j=1}^{G} w_j \cdot (TSS_j - WCSS_j)$, subject to $||w||^2 \leq 1$, $||w||_1 \leq s$, and $w_j \geq 0, \forall j$. Here, $TSS_j = \frac{1}{n} \sum_{i=1}^{n} \sum_{i'=1}^{n} d_j(x_i, x_{i'})$ is the total sum-of-squares, $WCSS_j = \sum_{k=1}^{K} \frac{1}{n_k} \sum_{i,i' \in C_k} d_j(x_i, x_{i'})$ is the within cluster sum-of-squares for gene j, and $d_j(x_i, x_{i'}) = (x_{ij} - x_{i'j})^2$. $||w||_1$ and $||w||^2$ are l_1 and l_2 norms of weight w. The l_1 regularization shrinks most of the feature weights w_j to 0 and realizes variable selection (i.e., not contributing to the clustering). Note that s is the tuning parameter to control feature selection (i.e., sparsity) and is chosen by gap statistic in the original paper. In this article, the method is implemented using the R package "sparcl."

2.2. *Sparse negative binomial clustering with varying library size (snbClust)*

In the literature, negative binomial model has been widely used for RNA-seq differential expression analysis due to its better model fitness than the Poisson model with an additional over-dispersion parameter. We assume $y_{ij}|C_i = k \sim NB(\mu_{ijk}, \phi_j)$ and $\log(\mu_{ijk}) = \log(s_i) + \beta_{jk}$, where C_i is the cluster assignment for the ith sample, s_i is the normalization size factor of the i-th sample, a priori estimated by edgeR package to control for the library size variation among samples, β_{jk} is the cluster mean of the kth cluster for the jth gene on the log scale after controlling for the library size variation and ϕ_j is the dispersion parameter for the jth gene. Here, the negative binomial model is parameterized as $E(y_{ij}|C_i = k) = \mu_{ijk}$ and $Var(y_{ij}|C_i = k) = \mu_{ijk} + \frac{\mu_{ijk}^2}{\phi_j}$ (higher dispersion parameter ϕ_j means lower variance). Let $\vec{y}_i = (y_{i1}, y_{i2}, \ldots, y_{iG})$ be the observed counts in sample i with G features. The penalized log-likelihood is given by,

$$\log L(\Theta_1) = \sum_{i=1}^{n} \log[\sum_{k=1}^{K} p_k f_k^{(nb)}(\vec{y}_i; s_i \exp(\vec{\beta}_k), \vec{\phi})] - \lambda h(\beta), \tag{2.1}$$

where $\Theta_1 = \{(p_k, \vec{\beta}_k), \vec{\phi}; k = 1, \ldots, K\}$ is the set of all unknown parameters, $f_k^{(nb)}(\vec{y}_i; s_i \exp(\vec{\beta}_k), \vec{\phi})$ is the density function of NB$(s_i \exp(\vec{\beta}_k), \vec{\phi})$ with $\vec{\beta}_k = (\beta_{1k}, \beta_{2k}, \ldots \beta_{Gk})$ being the cluster means of cluster k, $\vec{\phi} = (\phi_1, \ldots, \phi_G)$ is the vector of gene-specific dispersion parameters and p_k is the probability of belonging to cluster k.

In the penalty term, λ is the tuning parameter and lasso penalty or fused lasso penalty can be used in $h(\beta)$. For lasso penalty, $h(\beta) = h_0(\beta) = \sum_{k=1}^{K}\sum_{j=1}^{G} |\beta_{jk} - \beta_j^*|$ with β_j^* being the maximum likelihood estimation (MLE) of the global log-scaled mean of the jth gene assuming no cluster effect after controlling for the library size variation (see Section 2.2.1 for the estimation of β_j^*). The formulation is similar to sgClust in Section 2.1.1, but we note that unlike the Gaussian model in sgClust, the count data cannot be standardized in each gene vector. The subtraction of overall global cluster mean β_j^* for each gene j in $h_0(\beta)$ is necessary. For fused lasso penalty, $h(\beta) = h_1(\beta) = \sum_{j=1}^{G}\sum_{1\le k<k'\le K} |\beta_{jk} - \beta_{jk'}|$. Maximization of equation 2.1 can be achieved by using expectation-maximization (EM) algorithm (Dempster *and others*, 1977) for both lasso and fused lasso penalty. Here, we introduce a latent variable $z_{ik} = I\{i \in C_k\}$ as the indicator function of cluster assignment for sample i to be assigned to cluster k and the problem becomes maximizing the following complete penalized log-likelihood:

$$\log L_c(\Theta_2) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}[\log(p_k) + \sum_{j=1}^{G} \log(f_k^{(nb)}(y_{ij}; \exp(\log(s_i) + \beta_{jk}), \phi_j))] - \lambda h(\beta), \tag{2.2}$$

where $\Theta_2 = \{(p_k, \vec{\beta}_k, \vec{z}_k); k = 1, \ldots, K)\}$ and $\vec{z}_k = (z_{1k}, \ldots, z_{nk})$. Section 2.2.1 will elaborate the model with lasso penalty and Section 2.2.2 will introduce the model with fused lasso penalty and discuss the pros and cons.

REMARK For the gene-specific dispersion parameters ϕ_j's, they are estimated by the edgeR package for each data set and plugged into the model. For simplicity, $\vec{\phi}$ will be ignored as we introduce the algorithms below.

2.2.1. *SnbClust with lasso penalty (snbClust.lasso)* SnbClust with lasso penalty seeks to maximize 2.2 with $h_0(\beta) = \sum_{k=1}^{K}\sum_{j=1}^{G} |\beta_{jk} - \beta_j^*|$. In the literature, McLachlan (1997) discussed the estimation of a mixture of generalized linear models using iteratively reweighted least square algorithm (IRLS). Friedman *and others* (2010) proposed the estimation of a generalized linear model with convex penalties for variable selection using a coordinate descent algorithm. For the optimization of 2.2, we combined the above two ideas to derive a new EM algorithm to estimate the parameters in a mixture of generalized linear models with convex penalties. The pseudocode of the optimization of snbClust.lasso is given in Algorithm 1 in the Appendix A of the Supplementary material available at *Biostatistics* online and detailed steps are described below.

We first pre-estimate β_j^* (i.e., the global mean of feature j) and consider it known during the EM algorithm. β_j^* is estimated by maximizing $\sum_{i=1}^{n}\sum_{j=1}^{G} \log f(y_{ij}; \exp(\log(s_i) + \beta_j))$ using IRLS assuming no clustering effect. Once the vector β_j^* is estimated, we carry out the EM algorithm as follows. We use a generic notation $\Theta_2^{(m)}$ to represent the parameter estimates at iteration m. The E-step yields:

$$Q(\Theta_2;\Theta_2^{(m)}) = E_{\Theta_2^{(m)}}(\log L_{c,\Theta_2}|Y) = \sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}^{(m+1)}[\log p_k + \sum_{j=1}^{G}\log f(y_{ij};\exp(\log(s_i)+\beta_{jk}))] - \lambda\sum_{j}^{G}\sum_{k}^{K}|\beta_{jk}-\beta_j^*|,$$

where

$$z_{ik}^{(m+1)} = \frac{p_k^{(m)}\prod_{j=1}^{G} f_k^{(nb)}(y_{ij};\exp(\log(s_i)+\beta_{jk}^{(m)}))}{\sum_{l=1}^{K} p_l^{(m)}\prod_{j=1}^{G} f_l^{(nb)}(y_{ij};\exp(\log(s_i)+\beta_{jl}^{(m)}))}. \tag{2.3}$$

In the M-step, the updating function of p is given by $p_k^{(m+1)} = \sum_{i=1}^{n} z_{ik}^{(m+1)}/n$. The updating function of β cannot be easily derived by maximizing the above Q function. We can solve it by using IRLS algorithm, a similar idea recently applied in Wang *and others* (2016) under a regression setting. Suppose t is the current iteration of IRLS, we will repeat the following four steps for each gene respectively until convergence and return the final estimates of β_{jk} as $\beta_{jk}^{(m+1)}$:

(1) Calculate $w_{ijk}^{(t+1)} = \mu_{ijk}^{(t)}/(1+\phi_j^{-1}\mu_{ijk}^{(t)})$
(2) Update $\tau_{ijk}^{(t+1)} = \log(s_i) + \beta_{jk}^{(t)} + (y_{ij}-\mu_{ijk}^{(t)})/\mu_{ijk}^{(t)}$
(3) Solve $\beta_{jk}^{(t+1)} = \text{argmin}\ \frac{1}{2}\sum_i z_{ik}^{(m+1)} w_{ijk}^{(t+1)}(\tau_{ijk}^{(t+1)} - \log(s_i) - \beta_{jk})^2 + \lambda|\beta_{jk}-\beta_j^*|$
(4) Update $\mu_{ijk}^{(t+1)} = \exp(\beta_{jk}^{(t+1)} + \log(s_i))$

The solution in step 3 is given by:

$$\beta_{jk}^{(t+1)} = \beta_j^* + \text{sign}(\tilde{\beta}_{jk}-\beta_j^*)[|\frac{\sum_i z_{ik}^{(m+1)} w_{ijk}^{(t+1)}(\tau_{ijk}^{(t+1)} - \log(s_i)) - \lambda\ \text{sign}(\tilde{\beta}_{jk}-\beta_j^*)}{\sum_i z_{ik}^{(m+1)} w_{ijk}^{(t+1)}}| - |\beta_j^*|]_+, \tag{2.4}$$

where $\tilde{\beta}_{jk} = \sum_{i=1}^{n} z_{ik}^{(m+1)} w_{ijk}^{(t+1)}(\tau_{ijk}^{(t+1)} - \log(s_i))/\sum_{i=1}^{n} z_{ik}^{(m+1)} w_{ijk}^{(t+1)}$ is the estimate of β_{jk} without penalization and f_+ is the soft-thresholding function which takes the value f if $f_+ > 0$ and 0 otherwise. Once we obtain the estimates $\beta_{jk}^{(m+1)}$ from the IRLS algorithm, we can continue to iteratively carry out E step and M step until convergence to obtain the final maximum penalized likelihood estimate (MPLE).

2.2.2. *SnbClust with fused lasso penalty (snbClust.fused)* In lasso penalty, $h_0(\beta) = \sum_{k=1}^{K}\sum_{j=1}^{G}|\beta_{jk}-\beta_j^*|$ is used to shrink the β_{jk} towards β_j^* for every feature j. The natural downside of lasso penalty is that grouping can only occur at β_j^* but not elsewhere. When the penalty is not strong enough to shrink all β_{jk} to β_j^* for feature j, all β_{jk}'s tend to obtain distinct values. In other words, the lasso penalty cannot conclude for cluster-specific genes such as $(\hat{\beta}_{j1},\hat{\beta}_{j2},\hat{\beta}_{j3},\hat{\beta}_{j4}) = (5.96, 2.22, 2.22, 2.22)$ but instead will generate estimates like $(\hat{\beta}_{j1},\hat{\beta}_{j2},\hat{\beta}_{j3},\hat{\beta}_{j4}) = (5.96, 2.28, 2.24, 2.15)$. To accommodate this issue, we alternatively apply a fused lasso penalty $h_1(\beta) = \sum_{j=1}^{G}\sum_{1\le k<k'\le K}|\beta_{jk}-\beta_{jk'}|$, so that the distance between each pair of β_{jk} and $\beta_{jk'}$ can be shrunken toward zero.

The optimization of snbClust.fused is very similar to snbClust.lasso except for step 3 of IRLS. Instead of optimizing $\beta_{jk}^{(t+1)} = \text{argmin}\ \frac{1}{2}\sum_i z_{ik}^{(m+1)} w_{ijk}^{(t+1)} (\tau_{ijk}^{(t+1)} - \log(s_i) - \beta_{jk})^2 + \lambda|\beta_{jk} - \beta_j^*|$ in snbClust.lasso, now we are optimizing:

$$\beta_{jk}^{(t+1)} = argmin \frac{1}{2}\sum_i z_{ik}^{(m+1)} w_{ijk}^{(t+1)} (\tau_{ijk}^{(t+1)} - \log(s_i) - \beta_{jk})^2 + \lambda \sum_{1\le k<k'\le K} |\beta_{jk} - \beta_{jk'}|. \tag{2.5}$$

Equation 2.5 does not have a closed form solution as in 2.4, and we develop an alternating direction method of multipliers (ADMM) algorithm (Boyd *and others*, 2011) for the optimization. By introducing an auxiliary variable $\{\delta_{jkk'}, 1 \le k < k' \le K\}$, where $\delta_{jkk'} = \beta_{jk} - \beta_{jk'}$, we can reformulate the problem as minimizing

$$L_0(\vec{\beta}_j, \vec{\delta}_j) = \frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}^{(m)} w_{ijk}^{(t+1)} (\tau_{ijk}^{(t+1)} - \log(s_i) - \beta_{jk})^2 + \sum_{k<k'} \lambda|\delta_{jkk'}|$$

$$\text{subject to } \beta_{jk} - \beta_{jk'} - \delta_{jkk'} = 0.$$

The augmented Lagrangian is

$$L(\vec{\beta}_j, \vec{\delta}_j, \vec{u}_j) = L_0(\vec{\beta}_j, \vec{\delta}_j) + \sum_{k<k'} u_{jkk'}(\beta_{jk} - \beta_{jk'} - \delta_{jkk'}) + \frac{\nu}{2}\sum_{k<k'}(\beta_{jk} - \beta_{jk'} - \delta_{jkk'})^2,$$

where the dual variable $\vec{u}_j = \{u_{jkk'}, k < k'\}$ are Lagrange multipliers and ν is a hyperparameter controlling the convergence rate of ADMM. For a given value of $\vec{\delta}_j^{(b)}$ and $\vec{u}_j^{(b)}$ at step b, the iteration goes as

$$\vec{\beta}_j^{(b+1)} = \arg\min_{\vec{\beta}_j} L(\vec{\beta}_j, \vec{\delta}_j^{(b)}, \vec{u}_j^{(b)}) \tag{2.6}$$

$$\vec{\delta}_j^{(b+1)} = \arg\min_{\vec{\delta}_j} L(\vec{\beta}_j^{(b+1)}, \vec{\delta}_j, \vec{u}_j^{(b)}) \tag{2.7}$$

$$\vec{u}_{jkk'}^{(b+1)} = u_{jkk'}^{(b)} + \nu(\beta_{jk}^{(b+1)} - \beta_{jk'}^{(b+1)} - \delta_{jkk'}^{(b+1)}) \tag{2.8}$$

In 2.6, the problem is equivalent to minimizing

$$\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{K} z_{ik}^{(m)} w_{ijk}^{(t+1)} (\tau_{ijk}^{(t+1)} - \log(s_i) - \beta_{jk})^2 + \frac{\nu}{2}\sum_{k<k'}(\beta_{jk} - \beta_{jk'} - \delta_{jkk'}^{(b)} + \frac{u_{jkk'}^{(b)}}{\nu})^2.$$

Some algebra shows that we can rewrite the above as

$$\frac{1}{2}(q_j - A\vec{\beta}_j)^T W_j (q_j - A\vec{\beta}_j) + \frac{\nu}{2}(B\vec{\beta}_j - r_j)^T (B\vec{\beta}_j - r_j),$$

where $q_j = (\tau_{1j1} - \log(s_1), \ldots, \tau_{1jK} - \log(s_1), \ldots, \tau_{nj1} - \log(s_n), \ldots, \tau_{njK} - \log(s_n))$ is an $(nK \times 1)$ vector, $W_j = \text{diag}\{(z_{11}w_{1j1}, \ldots, z_{1K}w_{1jK}, \ldots, z_{n1}w_{nj1}, \ldots, z_{nK}w_{njK})\}$ is an $(nK \times nK)$ diagonal matrix, and $r_j = \delta_j - \nu^{-1}u_j$ is a $(\binom{K}{2} \times 1)$ vector. Matrices A and B are transformation matrix that expands β to the corresponding dimensions. We can derive the updating step for $\vec{\beta}_j^{(b+1)}$ as

$$\vec{\beta}_j^{(b+1)} = (A^T W_j A + \nu B^T B)^{-1}(A^T W_j q_j + \nu B^T r_j).$$

In (2.7), the problem is equivalent to minimizing $\frac{\nu}{2}(\delta_{jkk'} - \zeta_{jkk'})^2 + \rho(|\delta_{jkk'}|, \lambda)$, where $\zeta_{jkk'} = \beta_{jk}^{(t+1)} - \beta_{jk'}^{(t+1)} + u_{jkk'}^{(t)}/\nu$. The updating function is

$$\delta_{jkk'}^{(b+1)} = S\left(\zeta_{jkk'}, \lambda/\nu\right), \text{ for } 1 \leq k < k' \leq K,$$

where $S(z,t) = \text{sgn}(z)(|z| - t)_+$. The pseudocode of snbClust.fused is give in Algorithm 2 (Web Appendix A of the Supplementary material available at *Biostatistics* online). Compared with snbClust.lasso, snbClust.fused requires another ADMM loop which increases the computing time.

2.3. *Model selection*

For snbClust, sgClust, and sKmeans, the number of clusters K and sparsity parameter λ must be pre-estimated. BIC is a popular method to determine the tuning parameter by minimizing the criterion. A modified version of the BIC was introduced by Pan and Shen (2007) for the sgClust model. Here, we propose a similar BIC approach for estimating K and λ simultaneously: $BIC = -2\log L(\hat{\Theta}_1) + \log(n)d_e$, where $\hat{\Theta}_1$ is the MPLE calculated from Sections 2.2.1 and 2.2.2, given K and $d_e = (K-1) + KG - q$ is the effective number of parameters. In determining d_e, the first term $K-1$ refers to the number of parameters in the mixing probabilities with constraint $\sum p_k = 1$, the second term KG is the number of parameters in cluster means. Finally, q refers to the number of estimates (among the $K \cdot G$ cluster mean parameters) which are either shrunken to the global mean (snbclust.lasso) or to the common value (snbClust.fused). The dispersion parameters are pre-estimated by edgeR package, therefore they are considered known and thus not included in the BIC. As for sKmeans, gap statistic was proposed in the original paper and software package "sparcl" is used for selecting the sparsity parameter s, while the number of cluster K cannot be determined easily, similar to CountClust and PoiClaClu. How to determine K for sKmeans, CountClust, and PoiClaClu is beyond the scope of this article, and we provide the true K when evaluating these three methods (see Section 2.4 for details).

Remark: When fitting snbClust.lasso, BIC will be used to simultaneously estimate K and λ. For snbClust.fused, since the computational burden is heavy, we recommend to first fit snbClust.lasso for estimating K. With an estimated K, a sequence of λ can be fitted to snbClust.fused, where the best model is chosen by modified BIC above.

2.4. *Benchmarks for evaluation*

To benchmark the performance of selecting K, we will compare sgClust, snbClust.lasso, and snbClust.fused using BIC. We will also use BIC to evaluate the performance of feature selection (i.e., selecting λ) although as we will see later in Section 3, BIC can only perform well in selecting K in simulations but not for λ. For sKmeans, CountClust, and PoiClaClu, BIC cannot be used and how to select K for these three methods is out of the scope of this article. To fairly compare clustering accuracy and feature selection performance of all the methods above, we will evaluate based on the underlying true K.

In the high-dimensional clustering problem we consider here, clustering performance is first benchmarked by the clustering accuracy using adjusted Rand index (ARI) when the true cluster labels are known in simulations and real applications. We first compare ARI under the respectively selected tuning parameter (i.e., s for sKmeans and λ for sgClust, snbClust.lasso, and snbClust.fused) using gap statistic or BIC. Since the feature selection performance may vary for different methods and data compatibility to the models, we next evaluate ARI (plotted on y-axis) under different number of selected genes (plotted on x-axis) as an alternative approach. We next consider performance on feature (variable) selection. In simulation, since the true cluster-predictive features are known, we use Jaccard index of genes from the best model selected by BIC, as well as the receiver operating characteristic curve and its area under curve

(AUC), for evaluation. In real data, the true cluster-predictive features are unknown so we perform pathway enrichment analysis using Fisher's exact test under different number of genes, selected from models using different degrees of sparsity (λ), to evaluate statistical significance of biological annotation on selected features.

3. SIMULATION

3.1. *Simulation settings*

In this section, we conduct four simulations to show the advantages of snbClust.lasso and snbClust.fused while comparing it to other methods. In simulation 1, we assume that all genes are informative and all samples have equal library sizes. No variable selection is performed so we only assess the clustering performance. In simulation 2, we assume only a proportion of genes are informative and assess both the clustering accuracy and variable selection performance. In simulation 3, we perform additional sensitivity analysis by simulating gene–gene dependency structure to examine whether the performance would be affected and whether the gene independence assumption is valid in general. In simulation 4, we assume that each informative gene can only distinguish one cluster versus the rest so that conceptually snbClust.fused can be favored in this scenario over snbClust.lasso. We repeat 100 times for simulation 1 and 50 times for simulations 2, 3, and 4 and evaluate the averaged results.

To mimic real data structure, we extract the main characteristics of The Cancer Genome Atlas (TCGA) breast cancer RNA-seq data, which is also used in the second real data example in Section 4.2, to perform the simulation. The data set contains 610 female patients. We first compute the mean counts of each gene over all samples and obtain an empirical distribution of mean counts, which will be used to simulate baseline expression levels in all the four simulations. Since RNA-seq data are usually skewed with many highly expressed house-keeping genes which are irrelevant to cluster analysis, we exclude the top 30% mean counts when forming the empirical distribution. In addition, we also pre-estimate the gene-specific dispersion parameter $\vec{\phi}$ from the data using edgeR package and use it to simulate the data sets in simulation 1–4 above. The details of each simulation setting is shown in Appendix B of the Supplementary material available at *Biostatistics* online.

Note that in the simulation, the dispersion parameter $\vec{\phi}$ estimated from the TCGA data is only used to simulate the data sets. For each simulated data set, dispersion parameter $\vec{\phi}$ and library size normalization factors will be re-estimated by edgeR and then plugged into the snbClust, for a fair evaluation.

3.2. *Simulation results*

Figure 1 shows the mean and standard error of ARI values over 50 replications for the five methods in simulation 1. Here, the purpose is to evaluate whether using negative binomial distribution to model the count data outperforms other Gaussian-based methods (Kmeans or gClust) and other count-based models without considering overdispersion (PoiClaClu or CountClust) in a simple situation. We consider all the genes to be informative. Therefore, only clustering performance in terms of ARI is assessed in this case. Compared to other four methods, nbClust has better clustering performance (larger ARI) and the advantage is consistent as we vary the minimal effect size γ.

In simulation 2, we first evaluate the performance of estimating K and conclude that snbClust.lasso generally has better performance than sgClust (Table S1 (A) of the Supplementary material available at *Biostatistics* online). Next, given $K = 3$, we evaluate how the performance varies when there are noninformative genes as well as varying effect size γ. The clustering performance is measured using ARI as before while the variable selection is assessed using AUC, Jaccard index, and the number of genes in the best model selected from BIC. For PoiClaClu and CountClust, only ARI is measured since these two methods do not achieve variable selection. The result for this simulation scheme is summarized

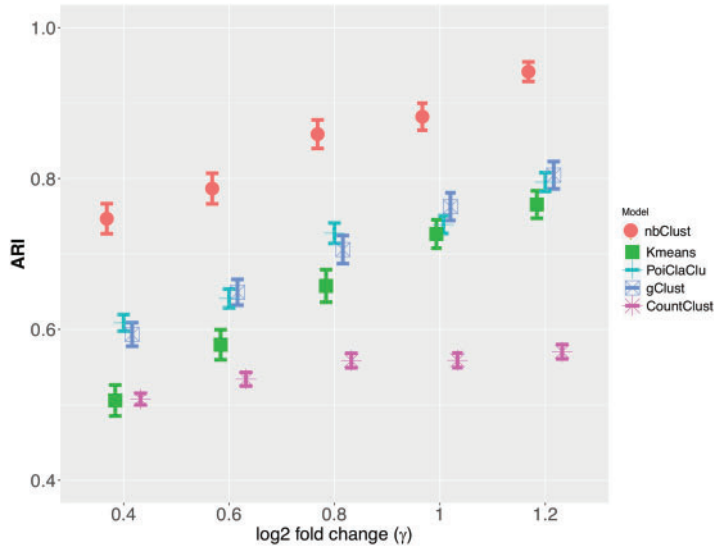

Fig. 1. ARI by effect size γ for simulation scheme 1 when no feature selection is needed. nbClust, Kmeans, and gClust are named after their respective methods without "s" since no variable sparsity is pursued.

in Figure 2 and Figure S1 of the Supplementary material available at *Biostatistics* online. Figure 2(A) and (B) and Figure S1(A) and (B) of the Supplementary material available at *Biostatistics* online show the comparison of performance between the six methods when the variation of library size is moderate (normalization size factor varies from 0.90 to 1.10). The ARI value of snbClust.lasso and snbClust.fused is much higher on average compared to other four methods. The variable selection performance in terms of Jaccard index and AUC is also higher for snbClust.lasso and snbClust.fused compared to sKmeans and sgClust. When the signal strength γ increases, we observe improved performance as expected. A similar trend is observed in the presence of high level of library size variation (normalization size factor varies from 0.70 to 1.30) shown in Figure 2(C) and (D) and Figure S1(C) and (D) of the Supplementary material available at *Biostatistics* online. We also compare ARI under different number of selected genes by tuning the sparsity parameter. For each effect size level γ (log2 fold change), we randomly select three simulation results in Figure S5 of the Supplementary material available at *Biostatistics* online, where snbClust.lasso and snbClust.fused consistently have better ARI across different number of genes, especially when effect size is moderate.

Table S1(B) of the Supplementary material available at *Biostatistics* online shows the performance of estimating K for snbClust.lasso and sgClust in Simulation scheme 3, where snbClust.lasso has clear advantage. Figure S2 of the Supplementary material available at *Biostatistics* online shows the results for varying gene dependence correlation α given $K = 3$. As we can see, the performance of snbClust.lasso and snbClust.fused remains relatively stable even when α increases up to 0.75, partially justifying robustness of the gene-gene independence assumption in our model. Intuitively, in high dimensional space, data points are much better separated and ignoring gene dependence structure may less impact the clustering performance, similar to the phenomenon of blessings of dimensionality described in Donoho (2000).

Figure 3 shows the performance of snbClust.lasso and snbClust.fused in Simulation scheme 4. It shows clear advantage of snbClust.fused over snbClust.lasso when each informative gene is a single-cluster-specific gene (i.e., only distinguish one cluster versus the rest), in terms of higher clustering accuracy ARI (Figure 3(A) and (C)) and feature selection Jaccard index (Figure 3(B) and (D)) and AUC (Figure S3(B)

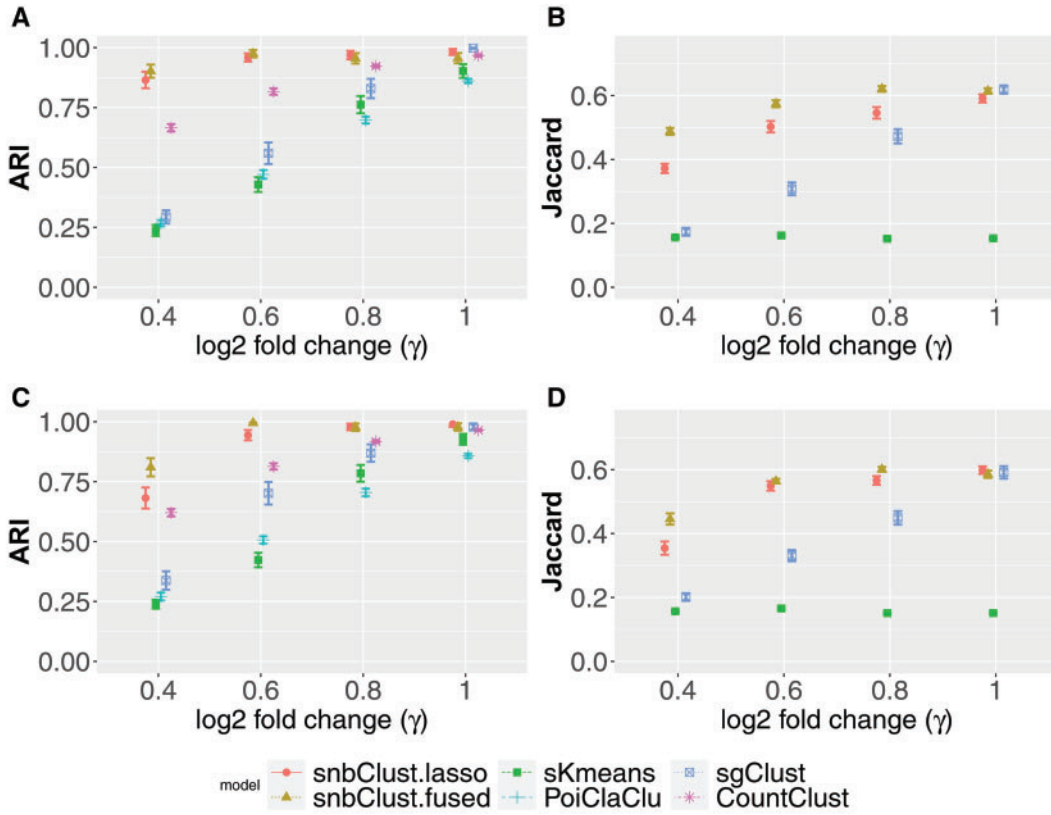

Fig. 2. Clustering accuracy by ARI and feature selection accuracy by Jaccard index for simulation scheme 2. (A, B) The result of low library size variation (normalization size factor varies from 0.90 to 1.10); (C, D) The result of high library size variation (normalization size factor varies from 0.70 to 1.30.

and (D) of the Supplementary material available at *Biostatistics* online). The advantage of snbClust.fused becomes large when K is large ($K = 5$), as expected.

4. Real data application

4.1. *Multiple brain regions of rat*

In the first example, we apply our method to an RNA-seq data set studying the brain tissues of HIV transgenic rat from Gene Expression Omnibus database (Li *and others*, 2013). RNA samples from three brain regions (hippocampus, striatum, and prefrontal cortex) are sequenced for both control strains and HIV infected strains. Only the 36 control strains (12 samples in each brain region) are used here to see whether samples from the three brain regions can be correctly identified ($K = 3, n_1 = n_2 = n_3 = 12$). After standard preprocessing and filtering out genes with mean counts smaller than 10 based on the guidance in edgeR, 10 280 genes are retained for clustering analysis. In this application, the true cluster labels (brain regions) are known and ARI can be used to evaluate clustering accuracy. However, the true informative genes are unknown and the Jaccard index and AUC cannot be calculated to assess feature selection accuracy for methods with variable selection, as in simulation. Instead, we obtain results of a sequential number of selected genes (around 50–1000) by varying the tuning parameter λ and compare the ARI curves. Finally,

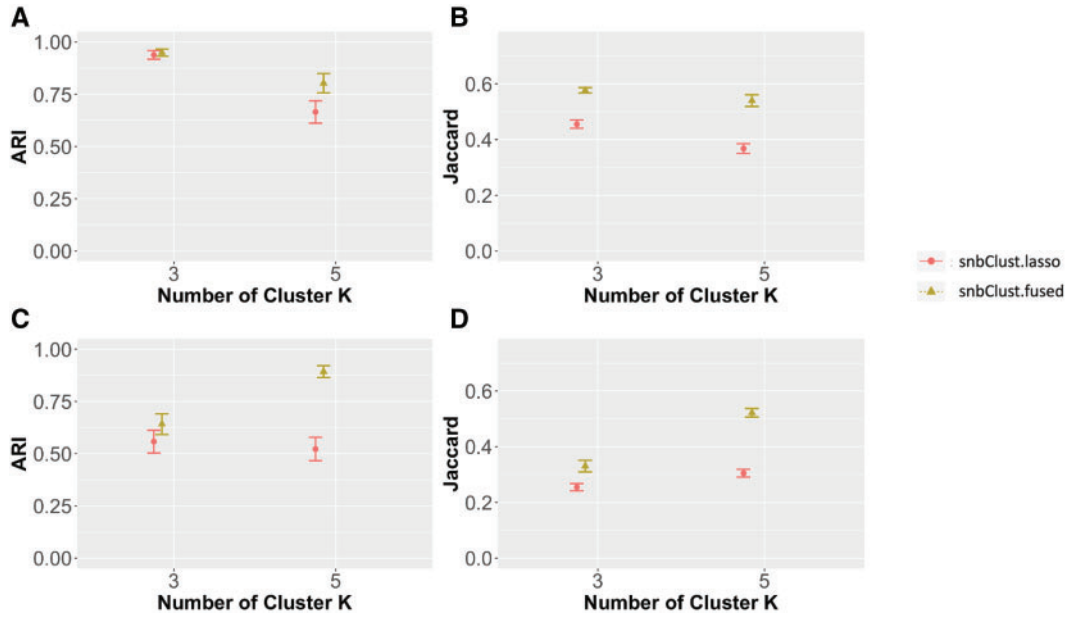

Fig. 3. Clustering accuracy by ARI and feature selection accuracy by Jaccard index for simulation scheme 4. Figures 2(A) and (B) is the result of simulation with large effect size and large variance; Figures 2(C) and (D) is the result of simulation with small effect size and small variance.

we perform pathway enrichment analysis by using Fisher's exact test based on the Gene Ontology (GO), Kyoto Encyclopedia of Genes and Genomes (KEGG), and Reactome pathway databases to assess the biological relevance of selected genes from models using different sparsity parameter λ.

SnbClust.lasso correctly chooses $K = 3$ using BIC while sgClust chooses $K = 4$, which shows superior performance of snbClust.lasso for estimating K. For a fair comparison, we input $K = 3$ for all the methods to evaluate the clustering accuracy and feature selection. Since gap statistic and BIC fail to select a reasonable number of genes (10 280 for sKmeans, 9846 for sgClust, 10 280 for snbClust.lasso, and 10 280 for snbClust.fused, with ARI = 1 for all four methods), we examine the clustering accuracy using different numbers of genes from models of different sparsity (by tuning λ or s). Figure 4(A) shows the ARI values under different gene selection for snbClust.lasso, snbClust.fused, sgClust, and sKmeans. SnbClust.lasso, snbClust.fused and sKmeans all demonstrate a perfect clustering performance (ARI = 1) when more than 20 genes are selected while sgClust performs poorly with at most around ARI = 0.55. PoiClaClu achieves ARI = 1 using all the genes while CountClust has ARI = 0 since the first cluster has the highest posterior probability for all the samples. To distinguish performance of snbClust.lasso, snbClust.fused and sKmeans further, we randomly subsample the sequencing counts to mimic shallower sequencing experiments, which is commonly encountered to save experimental cost. Figure 4(B) and (C) show the ARI results when we downsampled the sequencing reads to only 50% and 20% of their original total reads. At 50% subsampling sKmeans requires more than 30 selected genes to achieve perfect ARI whereas snbClust.lasso and snbClust.fused only need 20 and 6, respectively. When sequencing depth is further reduced to 20%, sKmeans needs 70 genes to achieve ARI = 1 whereas snbClust.lasso only needs around 30 genes and snbClust.fused only needs 12 genes. The performance for sgClust has been found to be universally worse than the other two methods.

To examine biological interpretation and functional annotation of selected genes, Figure 4(D) shows the number of enriched pathways under false discovery rate (FDR) = 0.05 when different numbers of genes

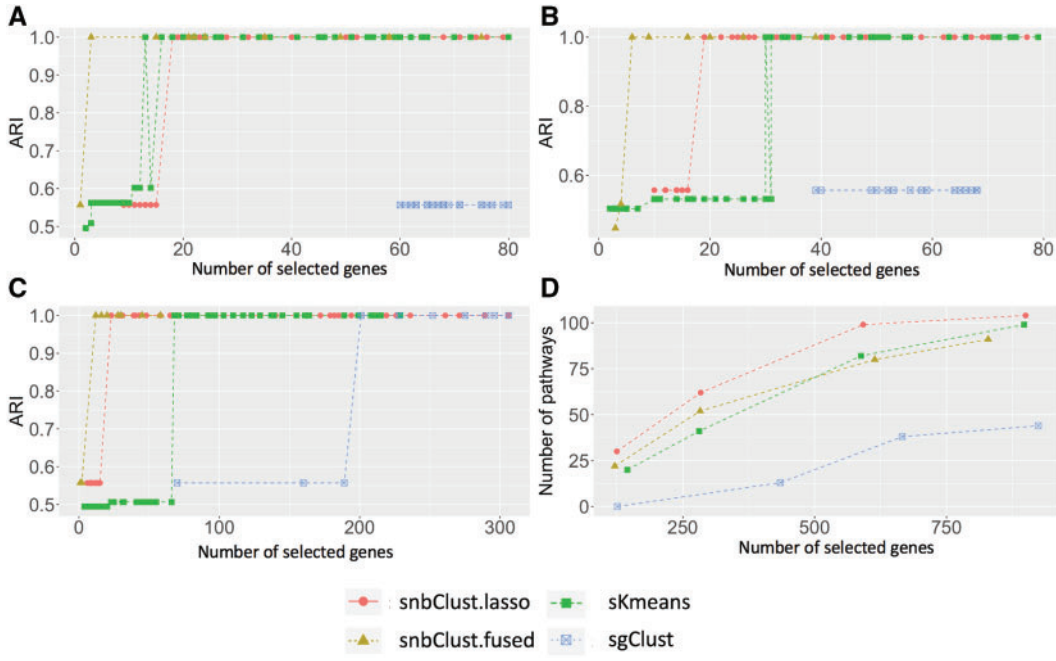

Fig. 4. (A) The ARI values under different gene selection for snbClust, sgClust, and sKmeans are shown. (B, C) The clustering performance under 50% and 20% downsampling, respectively are shown. (D) The number of enriched pathways under FDR = 0.05 when different numbers of genes (by tuning λ) are selected.

from models of different sparsity (by tuning λ) are selected. Compared to sKmeans and sgClust methods, snbClust.lasso consistently detects more enriched pathways, implying the better functional association of selected genes by snbClust. Table S2 of the Supplementary material available at *Biostatistics* online shows the union of pathways detected at FDR = 0.05 using the top 284, 283, 435, and 281 selected genes (62 pathways for snbClust.lasso, 52 pathways for snbClust.fused, 13 for sgClust, and 41 for sKmeans). The result finds many neural development, synapse function and metal ion transport pathways that are known to be actively and differentially expressed in different brain regions.

4.2. *Breast cancer data set*

Next, we apply the methods above to the Cancer Genome Atlas (TCGA) breast cancer data set. The data set contains 610 female patients with four different subtypes of breast cancer: Basal (116 subjects), Her2 (63 subjects), LumA (257 subjects), and LumB (174 subjects). After standard preprocessing and using the criteria of filtering out genes with mean count less than 5 and variance less than the median variance, 8789 genes are retained. LumA and LumB expression patterns were known to be similar, hence, three clusters considered for evaluation are Basal, Her2 and LumA+LumB. The evaluation is performed similarly to the rat brain example. Both BIC of sgClust and snbClust.lasso estimate $K = 7$. For a fair comparison, we use $K = 3$ as input for all the methods. Similar to the rat brain example, gap statistic and BIC fail to select a reasonable number of genes: sKmeans selects 8322 genes with ARI = 0.372, sgClust selects 8667 genes with ARI=0.369, snbClust.lasso selects 8789 genes with ARI = 0.603, and snbClust.fused selects 8789 genes with ARI = 0.599. We further compare the clustering accuracy of different methods using different numbers of selected genes. As shown in Figure 5(A), snbClust.lasso reaches high clustering accuracy

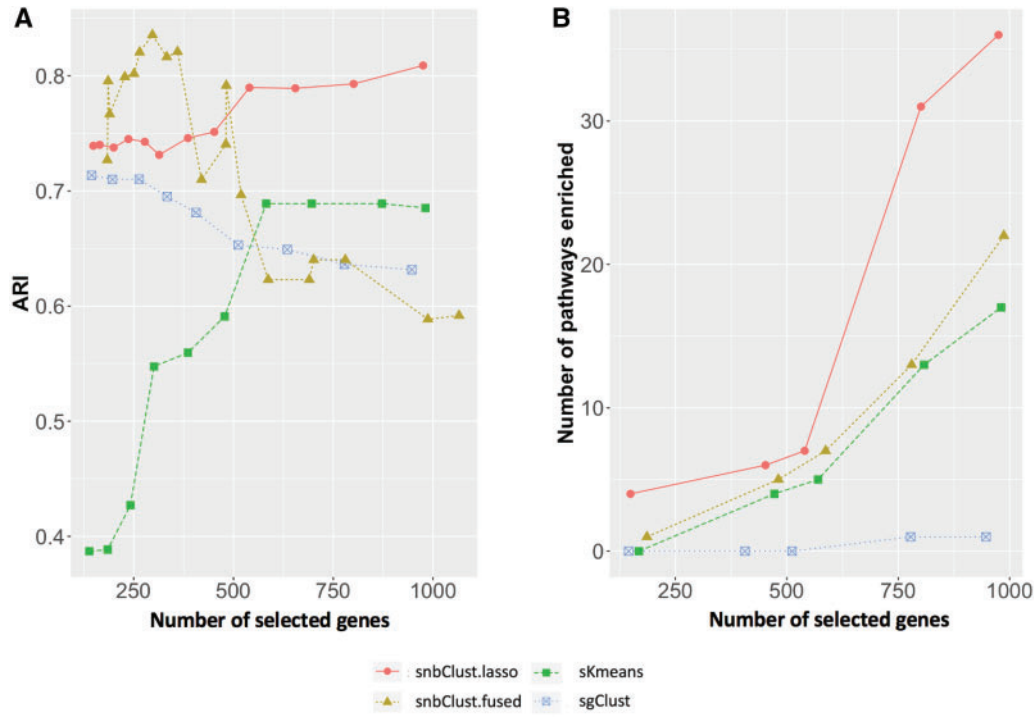

Fig. 5. Comparison of snbClust, skmeans, and sgClust model in Breast cancer data.

at 79% when 540 genes are selected and outperforms sgClust and sKmeans (accuracy at most ∼70%). SnbClust.fused has higher clustering accuracy (83% versus 74%) over snbClust.lasso when 250–400 genes are selected. Also, snbClust.fused has better performance than sgClust and sKmeans when about 500 genes are selected. In particular, performance of sgClust drops dramatically when the number of selected genes increases. PoiClaClu and CountClust show inferior performance than snbClust with ARI = 0.6 and ARI = 0.13, respectively, which shows the improved performance to model overdispersion and conduct variable selection. In terms of pathway analysis, snbClust.lasso and snbClust.fused also performs the best with larger number of enriched pathways compared to the other two methods when selecting 127–1000 top genes (Figure 5(B)). Specifically, we select similar numbers of selected genes (975, 987, 947, and 981) for snbClust.lasso, snbClust.fused, sgClust, and sKmeans and identify 36, 22, 1, and 17 enriched pathways at FDR = 0.05. Table S3 of the Supplementary material available at *Biostatistics* online outlines the union set of 46 pathways, which contains many pathways known related to cancer. For example, cell–cell adhesion (Farahani *and others*, 2014), ion channels (Biasiotta *and others*, 2016), calcium signaling (Cui *and others*, 2017), transmembrane transporter activity (Huang and Sadée, 2006) and ectoderm development are related to tumorigenesis. Epidermis development is related to the HER2-enriched subtype, which requires specific treatment such as Trastuzumab (Iqbal and Iqbal, 2014).

5. Discussion and conclusion

In this article, we proposed a sparse model-based clustering analysis with negative binomial mixture distribution using lasso (snbClust.lasso) and fused lasso penalty (snbClust.fused). Since RNA-seq data are known to be discrete, skewed and overdispersed, negative binomial is a more appropriate distribution

to capture the data characteristics, while normalizing counts to continuous and applying Gaussian-based models (e.g., sgClust and sKmeans), as well as modeling count data using Poisson or multinomial distribution (e.g., PoiClaClu and CountClust), lose information and efficiency. The extensive simulations and two real applications clearly confirm this intuition, where both snbClust.lasso and snbClust.fused outperform other methods in terms of clustering accuracy, gene selection and biological interpretation by pathway enrichment analysis.

One might wonder whether a better selected transformation can generate more Gaussian-like data and improve performance of sgClust and sKmeans. We fit the log(CPM) data to linear regression with true class label in each gene and generate qq-plots for the residuals. Figure S4 of the Supplementary material available at *Biostatistics* online shows qq-plots for residuals of all genes (Figure S4(a)), highly expressed genes (Figure S4(b) of the Supplementary material available at *Biostatistics* online), highly expressed genes with small overdispersion (i.e., large ϕ) (Figure S4(c) of the Supplementary material available at *Biostatistics* online) and lowly expressed genes with small overdispersion (Figure S4(d) of the Supplementary material available at *Biostatistics* online). The result confirms theoretical intuition that only genes with large counts and small overdispersion (i.e., Poisson-like) can be approximated well by Gaussian distribution (Figure S4(c) of the Supplementary material available at *Biostatistics* online). We also applied a global power transformation in Witten (2011) to eliminate overdispersion in the count data but performance of sgClust does not improve (Table S4 of the Supplementary material available at *Biostatistics* online), possibly due to gene-specific dispersion. Although we cannot prove it impossible, it is at least not easy to find a good transformation to accommodate both gene-specific overdispersion and genes with low counts to allow better fitting of the Gaussian residual assumption in sgClust.

There are three potential limitations in the current study. Firstly, the new count data model requires heavier computing than Gaussian-based models although still in an affordable range for general omics applications. To benchmark computing time, 70 choices of tuning parameter λ are performed in the rat brain application ($n = 36$ and $p = 10\ 280$) using 40 computing threads and average computing time for sgClust, snbClust.lasso, and snbClust.fused is 32.5 s, 1.2 min, and 10.9 min, respectively. For the breast cancer application ($n = 610$ and $p = 8789$), 30 choices of tuning parameter λ are performed using 30 computing threads. sgClust, snbClust.lasso and snbClust.fused require 4.8 min, 1.1 h, and 15.4 h, respectively. Similar to all optimization-based clustering algorithms, initial value plays an important role for successful clustering. Secondly, the new model does not consider gene correlation structure that may be prevalent among the genes (Zhou *and others*, 2009). Since the high dimensional data have feature number considerably larger than sample number and due to the complex structure of multivariate negative binomial distribution, incorporating correlation structure in the current model is not addressed in this article and will be a future direction. We, however, have performed sensitivity analysis to examine the impact of varying level of correlation structure. We find generally robust results in the clustering and feature selection using the current model with the gene independence assumption. Thirdly, BIC is an off-the-shelf criteria for parameter selection for model-based clustering while it only works to some extent. In simulations, BIC can estimate the number of clusters K correctly only when effect size is large (see Table S1 of the Supplementary material available at *Biostatistics* online) and will always include additional noise genes even if effect size is very large (see Figure S1 of the Supplementary material available at *Biostatistics* online). In real applications, BIC fails to select a reasonable λ in both data sets and fails to select the correct K in the TCGA study. While out of the scope of this article, resampling approaches are alternative ways to select K and λ and they often achieve better performance than traditional methods (Li *and others*, 2021). Extending resampling approaches to model selection of model-based clustering is a future direction.

As discussed in Section 1, Bayesian clustering methods, such as that in Tadesse *and others* (2005) and bclust in Nia and Davison (2012), provide a unified hierarchical framework to systematically incorporate multilevel uncertainties. Unfortunately, the bclust model is designed for an agglomerative hierarchical clustering setting, which is not comparable to finite mixture models in this article. The Gaussian-based

Bayesian clustering model in Tadesse *and others* (2005) is the only comparable method we have found in the literature. It, however, lacks a software package or programming code for implementation. As a result, Bayesian clustering methods are not evaluated in this article. Implementing and developing Bayesian clustering counterparts based on Gaussian or negative binomial mixture models and providing a fair comparison will be a future direction.

An R package to implement sgClust (which is not available from the original paper and existing R packages), snbClust.lasso, and snbClust.fused methods is available on https://github.com/YujiaLi1994/snbClust, along with all data and source code used in this article.

Supplementary material

Supplementary material is available online at http://biostatistics.oxfordjournals.org.

Acknowledgments

Conflict of Interest: None declared.

Funding

National Institutes of Health (NIH) (R01CA190766 and R21LM012752 to Y.L. and G.C.T.).

References

Biasiotta, A., D'Arcangelo, D., Passarelli, F., Nicodemi, E. M. and Facchiano, A. (2016). Ion channels expression and function are strongly modified in solid tumors and vascular malformations. *Journal of Translational Medicine* **14**, 285–285.

Binder, D. A. (1978). Bayesian cluster analysis. *Biometrika* **65**, 31–38.

Boyd, S., Parikh, N. and Chu, E. (2011). *Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers*. Hanover, MA, USA: Now Publishers Inc.

Cui, C., Merritt, R., Fu, L. and Pan, Z. (2017). Targeting calcium signaling in cancer therapy. *Acta Pharmaceutica Sinica B* **7**, 3–17.

Dempster, A. P., Laird, N. M. and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)* **39**, 1–22.

Dey, K. K., Hsiao, C. J. and Stephens, M. (2017). Visualizing the structure of rna-seq expression data using grade of membership models. *PLoS Genetics* **13**, e1006759.

Donoho, D. L. (2000). High-dimensional data analysis: the curses and blessings of dimensionality. *AMS math challenges lecture* **1**, 1–32.

Farahani, E., Patra, H. K., Jangamreddy, J. R., Rashedi, I., Kawalec, M., Rao Pariti, R. K., Batakis, P. and Wiechec, E. (2014). Cell adhesion molecules and their relation to (cancer) cell stemness. *Carcinogenesis* **35**, 747–759.

Fop, M. and Murphy, T. B. (2018). Variable selection methods for model-based clustering. *Statistics Surveys*, **12**, 18–65.

Friedman, J., Hastie, T. and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software* **33**, 1–22.

Huang, Y and Sadée, W. (2006). Membrane transporters and channels in chemoresistance and-sensitivity of tumor cells. *Cancer Letters* **239**, 168–182.

IQBAL, N. AND IQBAL, N. (2014). Human epidermal growth factor receptor 2 (HER2) in cancers: overexpression and therapeutic implications. *Molecular Biology International* **2014**, 852748.

LI, M. D., CAO, J., WANG, S., WANG, J., SARKAR, S., VIGORITO, M., MA, J. Z. AND CHANG, S. L. (2013). Transcriptome sequencing of gene expression in the brain of the HIV-1 transgenic rat. *PLoS One* **8**, e59582.

LI, Y., ZENG, X., LIN, C.-W. AND TSENG, G. C. (2021). Simultaneous estimation of cluster number and feature sparsity in high-dimensional cluster analysis. *Biometrics* (in press).

MCLACHLAN, G. J. (1997). On the EM algorithm for overdispersed count data. *Statistical Methods in Medical Research* **6**, 76–98.

NIA, V. P. AND DAVISON, A. C. (2012). High-dimensional Bayesian clustering with variable selection: the r package bclust. *Journal of Statistical Software* **47**, 1–22.

PAN, W. AND SHEN, X. (2007). Penalized model-based clustering with application to variable selection. *Journal of Machine Learning Research* **8**, 1145–1164.

RICHARDSON, S. AND GREEN, P. J. (1997). On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **59**, 731–792.

ROBINSON, M. D., MCCARTHY, D. J. AND SMYTH, G. K. (2010). edger: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics* **26**, 139–140.

SI, Y., LIU, P, LI, P. AND BRUTNELL, T. P. (2014). Model-based clustering for RNA-seq data. *Bioinformatics* **30**, 197–205.

TADESSE, M. G., SHA, N. AND VANNUCCI, M. (2005). Bayesian variable selection in clustering high-dimensional data. *Journal of the American Statistical Association* **100**, 602–617.

THALAMUTHU, A., MUKHOPADHYAY, I., ZHENG, X. AND TSENG, G. C. (2006). Evaluation and comparison of gene clustering methods in microarray analysis. *Bioinformatics* **22**, 2405–2412.

TSENG, G. C. (2007). Penalized and weighted k-means for clustering with scattered objects and prior information in high-throughput biological data. *Bioinformatics* **23**, 2247–2255.

WADE, S., GHAHRAMANI, Z. *and others*. (2018). Bayesian cluster analysis: point estimation and credible balls (with discussion). *Bayesian Analysis* **13**, 559–626.

WANG, Z., MA, S., ZAPPITELLI, M., PARIKH, C., WANG, C.-Y. AND DEVARAJAN, P. (2016). Penalized count data regression with application to hospital stay after pediatric cardiac surgery. *Statistical Methods in Medical Research* **25**, 2685–2703.

WITTEN, D. M. (2011). Classification and clustering of sequencing data using a poisson model. *The Annals of Applied Statistics* **5**, 2493–2518.

WITTEN, D. M. AND TIBSHIRANI, R. (2010). A framework for feature selection in clustering. *Journal of the American Statistical Association* **105**, 713–726.

ZHOU, H., PAN, W. AND SHEN, XG. (2009). Penalized model-based clustering with unconstrained covariance matrices. *Electronic Journal of Statistics* **3**, 1473–1496.

[*Received May 25, 2020; revised June 03, 2021; accepted for publication June 06, 2021*]