EBSeq: improving mixing computations for multi-group differential expression analysis

Xiuyu Ma
University of Wisconsin Madison

Christina Kendziorski
University of Wisconsin Madison

Michael A. Newton
University of Wisconsin Madison

ABSTRACT

EBSeq is a Bioconductor package designed to calculate empirical-Bayesian inference summaries from sequence-based gene-expression (RNA-Seq) data. It produces gene or isoform-specific scores that measure various patterns of differential expression among a set of sample groups, and is most commonly deployed to measure differential expression between two groups. Its use of local posterior probabilities from a fitted mixture model provides the data analyst a direct way to score the false discovery rate of any reported list of genes, and it is one of the only tools that can address local false discovery rates when analyzing multiple sample groups. Contemporary applications have increasing numbers of sample groups, and the algorithms deployed in **EBSeq** are neither space nor time efficient in this important case. We describe a version update utilizing code improvements and novel pruning and clustering algorithms in order to reduce the complexity of mixture computations. The algorithms are supported by a theoretical analysis and tested empirically on a variety of benchmark and synthetic data sets.

Keywords: Empirical Bayes, hierarchical model, Bayes factor, pruning, local false discovery rate

Introduction

[1] Since their introduction over twenty years ago, technologies to measure genome-wide gene expression have revolutionized science and medicine. The resulting data have had a major impact on statistical sciences as well, by introducing challenges arising from "small n, large p" datasets. One of the central statistical challenges has been the differential expression problem: namely, how do we identify genes whose expression levels vary significantly across biological conditions? Dozens of methods have been developed toward this end and a few have endured. For RNA-Seq data, the empirical Bayesian hierarchical modeling approach, encoded in R package **EBSeq** has a number of advantages owing to how it captures variation characteristics of genes and isoforms and how it scores differential expression over two or more conditions (Leng, Dawson, Thomson, Rissman, Smits, Haag, Gould, Stewart, and Kendziorski 2013). It has proven useful in hundreds of studies including studies of development (Louro, Marques, Manchado, Power, and Campinho 2020; Sanders and Cartwright 2015; Yoon, Ringeling, Vissers, Jacob, Pokrass, Jimenez-Cyrus, Su, Kim, Zhu, Zheng *et al.* 2017; Sabbagh, Heng, Luo, Castanon, Nery, Rattner, Goff, Ecker, and Nathans 2018), viral transcription (Newhouse, Hofmeister, and Balakrishnan 2017; O'Grady, Baddoo, and Flemington 2017; Baños-Lara, Zabaleta, Garai, Baddoo, and Guerrero-Plata 2018; Zhang, Zeng, Zhou, Irving, Li, Shi, and Wang 2017), and cancer (Lee, Ahn, Jeong, Pak, Moon, and Kim 2020; Song, Tang, Li, Liu, and Zhou 2018; Son, Jeong, Park, No, Lee, Lee, and Chung 2017; Yang, Lv, Lv, Liu, Dong, Yao, Dai, Zhang, and Wang 2016). **EBSeq** has been deployed in Bioconductor since 2013 (Leng and Kendziorski 2019; Huber, Carey, Gentleman, Anders, Carlson, Carvalho, Bravo, Davis, Gatto, Girke, Gottardo, Hahne, Hansen, Irizarry, Lawrence, Love, MacDonald, Obenchain, Ole's, Pag'es, Reyes, Shannon, Smyth, Tenenbaum, Waldron, and Morgan 2015).

The most common use of **EBSeq** is to score differential expression between two biological conditions. The package's multi-condition feature is less frequently deployed; but recently it has been recognized that **EBSeq** multi-condition scores are uniquely suited to characterizing multiple cellular subtypes from the growing body of single-cell RNA-seq data (Ma, Korthauer, Kendziorski, and Newton 2019). The work reported here is motivated by the need to improve computational performance of the multi-condition calculations within the original system, say **EBSeq.v1**, which at the time of writing is version 1.26.0 at Bioconductor. In addition to some basic code improvements, we deploy in **EBSeq.v2** a new algorithmic approach to determining multi-condition differential expression scores.

With samples from K biological conditions, **EBSeq.v1** calculates posterior probability scores for various patterns of differential expression among these conditions. Typically the null pattern, in which expected expression is the same across all groups, receives the highest score, on the average over genes; the software can consider many patterns. A computational bottleneck arises if we ask the code to consider all possible patterns, which are equal in number to the Bell number, B_K, of partitions of K objects (Gardner 1978). For even moderate K, the memory and time costs of **EBSeq.v1** become excessive. Section 2 describes an alternative pruning/clustering algorithm which leverages the finding that many of the differential expression patterns will have small mixing rates; and we can know this without fitting the full model. By identifying patterns that are probably inconsequential to the final

[1] This manuscript version is a technical report from the University of Wisconsin Department of Biostatistics and Medical Informatics, and was posted to bioRxiv June 19, 2020.

inferences, we remove them from other compute-intensive parts of the code and improve the overall operating characteristics, as demonstrated in a battery of tests in Section 3.

EBSeq.v2 improves the performance of **EBSeq.v1**. In addition to improved handling of group partitions indicated above and discussed in Section 2, the core code is converted to C++ and adopts open-source, peer-reviewed, and fast libraries **Eigen** and **Boost** for internals (Guennebaud, Jacob *et al.* 2010; Boost 2015). We also modify the EM algorithm by changing how the hyperparameters are recomputed in each cycle. We substitute the Nelder-Mead optimization (`optim` from package **stats**) with a single gradient step within each EM update. The overall effect of these changes is to dramatically improve the performance of **EBSeq** in the multi-group setting. Section 3 presents a variety of simulated and empirical examples and a discussion follows in Section 4.

METHODS

The statistical problem

For each inference unit in the system we have real-valued measurements $X = \{X_i\}$, for a sample index $i = 1, 2, \cdots, n$, as well as discrete sample labels, say y_i, taking values in a label set $\{1, 2, \cdots, K\}$. The labels refer to different sampling groups, or conditions, that underlie the measurements, and we are especially interested in the case when K exceeds 2. In applications of interest we expect K to be small compared to n, perhaps in the tens when n is in the hundreds to thousands. Statistical inference is focused on evaluating hypotheses about the unknown mean values $\mu_k = E(X_i|y_i = k)$. For example, in the case of $K = 4$ groups, we have $B_4 = 15$ different patterns of equality and inequality among the group means, a few of which are:

$$\begin{aligned} H_0 &: \mu_1 = \mu_2 = \mu_3 = \mu_4 \\ H_1 &: \mu_1 \neq \mu_2 = \mu_3 = \mu_4 \\ H_2 &: \mu_1 \neq \mu_2 \neq \mu_3 = \mu_4, \mu_1 \neq \mu_3. \end{aligned}$$

The set of partitions of $\{1, 2, \cdots, K\}$ is in one-one correspondence with the set of such hypotheses regarding equalities and inequalities among the means. More specifically, for a partition $\pi = \{b\}$ composed of mutually disjoint blocks $b \subset \{1, 2, \cdots, K\}$, we say that mean vector $\mu = (\mu_k)$ satisfies pattern π if $\mu_j = \mu_k$ whenever $j, k \in b$ for some $b \in \pi$ and also if $\mu_j \neq \mu_k$ whenever j and k are in different blocks. Thus, for example, H_0 above corresponds to $\pi = \{\{1, 2, 3, 4\}\}$, and H_1 corresponds to $\pi = \{\{1\}, \{2, 3, 4\}\}$. In the remainder, we use notation M_π to denote the the constraint (i.e., hypothesis) on the mean vector associated with partition π:

$$M_\pi = \left\{ \mu \in \mathbb{R}^K : \mu_j = \mu_k \iff j, k \in b,\, b \in \pi \right\}.$$

Notice that any mean vector μ is an element of exactly one of these subsets, and, considering all partitions,

$$\mathbb{R}^K = \bigcup_\pi M_\pi.$$

The use of partition/pattern structures for expected values appears in many statistical problems. They are a central feature of Dirichlet-process mixture computations (MacEachern 1998), various Bayesian clustering algorithms (Quintana and Iglesias 2003; Heller and Ghahramani 2005), as well as empirical Bayesian methodologies for genomic applications, such as the **EBSeq** tool introduced earlier, and a similar tool for microarray-based data, **EBarrays** (Kendziorski, Newton, Lan, and Gould 2003; Yuan, Newton, Sarkar, and Kendziorski 2019).

In empirical Bayesian applications, there are very many inference units (genes), and for each sample i there is a measurement on every unit. The methodology entails both discrete and continuous mixing over the parameter space in order to deliver posterior probability scores for each unit: $p_{\text{local},\pi} = P(M_\pi | X, y)$. We use the subscript 'local' to remind us that the the probability is local to the particular unit (e.g., gene), in the same way that the local false discovery rate is local to each testing unit in large-scale inference (**?**).

The discrete mixing further involves probabilities estimated for the whole system, $p_{\text{global},\pi}$. By the use of maximum likelihood estimation, the fitted probabilities satisfy:

$$p_{\text{global},\pi} = \text{mean}_{\text{units}} \left(p_{\text{local},\pi} \right)$$

The conditional independence assumptions behind **EBSeq** induce a product-partition form on $p_{\text{local},\pi}$: with $X_b = \{X_i : y_i = k, \text{and } k \in b\}$ denoting all n_b realized measurements on a given unit (gene) from samples i whose condition status y_i maps them to a block b of partition π,

$$p_{\text{local},\pi} \propto p_{\text{global},\pi} \prod_{b \in \pi} f(X_b). \tag{0.0.1}$$

The proportionality is resolved by summing over all B_K partitions, which presents a computational challenge even for moderate K. In **EBarrays**, the joint predictive mass $f(x)$ takes either a compound Gamma form or a log-normal form; our focus is to improve **EBSeq**, which brings empirical Bayes methodology to RNA-Seq data. In **EBSeq**, $f(x)$ takes a special form as a Beta mixture of Negative Binomial (NB) mass functions. These distributional choices respond to properties and variance characteristics of RNA-Seq data, and also lead to a convenient closed-form predictive mass function:

$$f(x) = \left[\prod_{i=1}^{m} \binom{x_i + \gamma - 1}{x_i} \right] \frac{\text{Beta}(\alpha + m\gamma, \beta + \Sigma_{i=1}^{m} x_i)}{\text{Beta}(\alpha, \beta)} \tag{0.0.2}$$

EBSeq.v1 sets hyperparameters α, β, and γ as well as mixing rates $p_{\text{global},\pi}$ using both local data on the unit and global data on from all units. See Appendix A for additional details. A gene scores highly for pattern π if its local posterior probability $p_{\text{local},\pi}$ as in (0.0.1) is high, as evaluated through the density (0.0.2). Noting the cancellation of factorial terms, we see that these local scores are determined as a special transform of total expression within each block.

Pruning

EBSeq.v1 fits a mixture over all B_K partitions. Even if this were computationally feasible for moderately large K, we expect in applications that very many, perhaps most, of these

partitions would contribute a negligible amount to the fitted model. We propose a pruning algorithm that uses filtering statistics to identify partitions that are likely to carry most of the mixture mass without the need to fit the mixture over all partitions. The algorithm works by selecting probable partitions at each unit and taking their union as our global pool of partitions possibly of much smaller size than B_K.

Consider first a single inference unit (gene) and the group sample means $\hat{\mu} = (\hat{\mu}_1, \cdots, \hat{\mu}_K)$:

$$\hat{\mu}_k = \frac{\sum_{i=1}^n X_i 1[y_i = k]}{\sum_{i=1}^n 1[y_i = k]}$$

and let $r = (r_1, \cdots, r_K)$ be the rank of $\hat{\mu}$, i.e. r_k is the position of $\hat{\mu}_k$ in the permutation rearranging $\hat{\mu}$ into ascending order. In case of ties we break them according to values of group labels in y. We use an overlap concept that was also used in (Dahl 2009) to trim partitions in a modal clustering application.

Definition 1. *Two sets E_1 and E_2 of finite and integer elements overlap if E_1 contains a number between the smallest and largest numbers of E_2, or vice versa.*

For example, $E_1 = \{1,3\}$ overlaps with $E_2 = \{2\}$, but $E_1 = \{1,2\}$ and $E_2 = \{3\}$ do not overlap. Relative to a partition $\pi = \{b\}$, consider sets of indices $A_r(b) = \{r_k, k \in b\}$.

Definition 2. *Compatibility: A partition π is compatible with an empirical rank r if either π contains only one block or if for any two different blocks $b_1, b_2 \in \pi$, $A_r(b_1), A_r(b_2)$ do not overlap.*

For example with $K = 4$, and the empirical ordering $\hat{\mu}_1 < \hat{\mu}_2 < \hat{\mu}_3 < \hat{\mu}_4$ corresponds to rank $r = (1,2,3,4)$. The partitions compatible with r, and their corresponding hypotheses, are

$$\begin{aligned}
\{\{1,2,3,4\}\} &\quad \mu_1 = \mu_2 = \mu_3 = \mu_4 \\
\{\{1\},\{2,3,4\}\} &\quad \mu_1 \neq \mu_2 = \mu_3 = \mu_4 \\
\{\{1,2\},\{3,4\}\} &\quad \mu_1 = \mu_2 \neq \mu_3 = \mu_4 \\
\{\{1,2,3\},\{4\}\} &\quad \mu_1 = \mu_2 = \mu_3 \neq \mu_4 \\
\{\{1,2\},\{3\},\{4\}\} &\quad \mu_1 = \mu_2 \neq \mu_3 \neq \mu_4 \\
\{\{1\},\{2,3\},\{4\}\} &\quad \mu_1 \neq \mu_2 = \mu_3 \neq \mu_4, \mu_1 \neq \mu_4 \\
\{\{1\},\{2\},\{3,4\}\} &\quad \mu_1 \neq \mu_2 \neq \mu_3 = \mu_4, \mu_1 \neq \mu_3 \\
\{\{1\},\{2\},\{3\},\{4\}\} &\quad \text{All } \mu\text{s are distinct,}
\end{aligned}$$

while, for example, $\mu_1 = \mu_3 \neq \mu_2 = \mu_4$, i.e., $\{\{1,3\},\{2,4\}\}$, is not compatible with r. In fact, for the set $\mathcal{C}_g$ of partitions π that are compatible with the empirical ranks at unit g, we find:

Lemma 1. *The cardinality of $\mathcal{C}_g$ is $|\mathcal{C}_g| = 2^{K-1}$.*

Compatible partitions vary from unit to unit, and over the full system there may still be close to B_K different partitions that are compatible for at least one unit. But compatibility is a useful first property to consider as we filter the total number of partitions to a manageable number.

Sampling theory offers one argument in support of considering compatible partitions when the total number of samples n is large compared to the number of groups K. Then, deviations $\hat{\mu} - \mu$ tend to be relatively small. For the pattern M_π corresponding to μ, π is not compatible with the empirical ranks only if for some pair of groups j and k, we have $\mu_j < \mu_k$ and also $\hat{\mu}_j > \hat{\mu}_k$. By the law of large numbers, this event is increasingly improbable as n increases.

We investigate compatibility empirically by running **EBSeq.v1** in three example data sets where K is sufficiently small that computations are feasible: GSE45719 (Deng, Ramsköld, Reinius, and Sandberg 2014), GSE57872 (Patel, Tirosh, Trombetta, Shalek, Gillespie, Wakimoto, Cahill, Nahed, Curry, Martuza *et al.* 2014), GSE74596 (Engel, Seumois, Chavez, Samaniego-Castruita, White, Chawla, Mock, Vijayanand, and Kronenberg 2016). For each data set, Table 1 reports

$$p_{\text{compatible}} = \text{mean}_g \left(\sum_{\pi \in \mathcal{C}_g} p_{\text{local},\pi} \right),$$

which measures the posterior probability mass of compatible partitions in a fitted model. Compatible partitions cover most of the mixture mass in these cases. Table 1 also reports $N_{95\%}$, which measures the concentration of local posterior probability mass over the full set of partitions. Specifically, at each unit g we consider the most probable partitions and we count how few of these are required to capture at least 95% of the local probability mass, we then average over units to get $N_{95\%}$. We also keep track of $N_{c,95\%}$, which measures the average number of compatible partitions within the set capturing at least 95% local mass. Notice that $N_{c,95\%}$ and $N_{95\%}$ are much smaller than $|\mathcal{C}_g| = 2^{K-1}$, but further pruning is typically necessary to have a manageable number of partitions when considering all units at once.

Data Set	N_{samples}	N_{units}	K	$p_{\text{compatible}}$	B_K	$N_{95\%}$	$N_{c,95\%}$
GSE45719	110	27083	4	95%	15	4.9	3.56
GSE74596	114	23337	6	82%	203	13.11	6.57
GSE52529	143	22112	7	85%	877	40.5	21.2

Table 1: Empirical properties of partitions in several data sets. $N_{\text{samples}}, N_{\text{units}}$ are numbers of samples and units

A key observation is that some pairs of groups present strong information that we can use to filter partitions. For example, at typical levels of variation seen in RNA-Seq data, observing $\hat{\mu}_j = 10, \hat{\mu}_k = 1000$ at two groups would suggest $\mu_j \neq \mu_k$, and we may not lose much by dropping partitions π asserting the opposite: $\mu_j = \mu_k$. Such information is measured by a Bayes factor comparing differential mean with equivalent mean, namely

$$D_{j,k} = \frac{P(X_{\{j,k\}} | \mu_j \neq \mu_k, y)}{P(X_{\{j,k\}} | \mu_j = \mu_k, y)} = \frac{f(X_{\{j\}}) f(X_{\{k\}})}{f(X_{\{j,k\}})} \tag{0.0.3}$$

where $X_{\{j\}} = \{X_i : y_i = j\}$ denotes all measurements from any group j, $X_{\{j,k\}}$ represents vector of all expression values at a given unit for samples from two groups j and k, and f

is the predictive density function from (0.0.2). If the two-group Bayes factor is sufficiently extreme (small or large), we are guided about partitions that may be dropped or included before doing a full mixture computation.

In restricting to compatible partitions $\mathcal{C}_g$, the two-group Bayes factors are relevant to pairs of groups that are adjacent after ranking by empirical mean. Let o_j be the antirank representing the group label having jth smallest sample mean. Then each $\pi \in \mathcal{C}_g$ is set by filling either "$\neq$" or "$=$" into the slots of equation (0.0.4),

$$\mu_{o_1} _ \mu_{o_2} _ \cdots _ \mu_{o_K}. \tag{0.0.4}$$

These $K-1$ slots may be assessed using two-group Bayes factors. To identify a unit specific set of pruned partitions $\mathcal{S}_g$, we consider three assignment states for each slot in (0.0.4): (equivalent, differential, and uncertain) based on the Bayes factor and a user-specified threshold $t_1 > 0$.

$$\begin{aligned} \log(D_{o_j,o_{j+1}}) > t_1 &\Longrightarrow \mu_{o_j} \neq \mu_{o_{j+1}} \\ \log(D_{o_j,o_{j+1}}) < -t_1 &\Longrightarrow \mu_{o_j} = \mu_{o_{j+1}} \\ -t_1 < \log(D_{o_j,o_{j+1}}) < t_1 &\Longrightarrow \text{uncertain} \end{aligned}$$

Those partitions consistent with the filled-in (0.0.4) constitute a restricted (i.e., pruned) set of partitions $\mathcal{S}_g \subset \mathcal{C}_g$. The user-specified threshold t_1 gauges the size of $\mathcal{S}_g$, with larger t_1 being more inclusive and smaller values being more restrictive. We take $\mathcal{S} = \cup_g \mathcal{S}_g$ to be the selected set of partitions to be used in the full model-fitting computation. Pseudo-code for the resulting `pruning` tool is in Algorithm 1. For Bayes-factor evaluation, we set hyperparameters (α, β) to default values and estimate γ; numerical experiments in Appendix C.2 show that $\mathcal{S}$ is robust to the choice of these hyperparameters.

To investigate `pruning`, we run it on two data sets that are too large for **EBSeq.v1**: Hrvatin (Hrvatin, Hochbaum, Nagy, Cicconet, Robertson, Cheadle, Zilionis, Ratner, Borges-Monroy, Klein *et al.* 2018), Retina (Shekhar, Lapan, Whitney, Tran, Macosko, Kowalczyk, Adiconis, Levin, Nemesh, Goldman *et al.* 2016), and we set the threshold $t_1 = 1$. We say $N_{\mathrm{UC},g}$ is the number of uncertain positions at unit g and take $N_{\mathrm{selected}} = |\mathcal{S}|$. Table 2 shows that units

Data Set	N_{samples}	N_{units}	K	mean $N_{\mathrm{UC},g}$	max $N_{\mathrm{UC},g}$	N_{selected}	B_K (million)
Hrvatin	2164	17228	12	0.016	4	521	4.2
Retina	15551	22156	14	0.008	4	376	190.9

Table 2: Properties of two-group Bayes factor filtering in two example data sets

$N_{\mathrm{UC},g}$ is quite small for this threshold setting. Further, we provide users an option to bound the number of uncertainty positions at each unit in order to reduce complexity; details can be found in the Appendix C.2.

We also check `pruning` in the smaller examples from Table 1. Here we calculate the overall mixture mass (from the full model and **EBSeq.v1**) that is associated with the selected set

Data Set	N_{selected}	B_K	$p_{\mathcal{S}}$
GSE45719	15	15	1
GSE74596	126	203	0.99
GSE52529	227	877	0.88

Table 3: Empirical properties of selected partitions in three example data sets.

of partitions: $p_{\mathcal{S}} = \sum_{\pi \in \mathcal{S}} p_{\text{global},\pi}$. Table 3 confirms that `pruning` finds the dominant mixture components.

Since `pruning` utilizes pairwise comparisons between sample groups in order to select partitions for full model fitting, it is reassuring to know that this pairwise information is sufficient to detect differences that would be found had we accounted for all partitions of all groups. We find:

Theorem 1. *Let hyperparameters α, β, γ be fixed positive integers. If empirical means $\hat{\mu}_{o_j}$ and $\hat{\mu}_{o_{j+1}}$ of adjacent groups o_j and o_{j+1} are sufficiently distinct, then the partition π^* that maximizes $P(X|M_\pi, y)$ over all B_K partitions must place o_j and o_{j+1} into different blocks.*

Conditions assuring that $\hat{\mu}_{o_j}$ and $\hat{\mu}_{o_{j+1}}$ are sufficiently distinct are presented in Appendix B. This finding confirms that the use of pairwise information, such as in `pruning`, conveys properties about the optimal solution one would compute in considering all B_K partitions, without going to the trouble of doing a full survey.

Crowding

In contrast to `pruning`, we may also seek to prevent scenarios in which too few patterns are selected. This happens when `pruning` creates a chain of equality states even though groups at either end of the chain are significantly different. Then, `pruning` would select

$$\mu_{o_j} = \mu_{o_{j+1}} = \mu_{o_{j+2}} \ldots = \mu_{o_k},$$

if the two-group Bayes factors favor equality at all adjacent pairs, even though the empirical difference between head and tail of the chain may be large. For example, if $D_{o_j,o_{j+1}}, D_{o_{j+1},o_{j+2}}$ are both near zero and favor equivalent means, then `pruning` selects $\mu_{o_j} = \mu_{o_{j+1}} = \mu_{o_{j+2}}$. However $D_{o_j,o_{j+2}}$ may support uncertain or differential states between these non-adjacent groups. To address this crowding phenomenon, `equalHandle` is an algorithm to break down equality chains when necessary. It adopts an idea from agglomerative hierarchical clustering by iteratively checking the state of adjacent groups with the smallest Bayes factor along the chain. If the state is equal, `equalHandle` combines adjacent groups to form a new, larger group. Otherwise it breaks the chain. Figure 1 demonstrates the algorithm on an equality chain of length 5; see pseudo-code for `equalHandle` in Algorithm 2.

EBSeq.v2

EBSeq.v2 works as follows: a pool of partitions is compiled by running `pruning` and `equalHandle` over all units; then the EM algorithm is applied to fit a discrete mixture

Algorithm 1 PRUNING

Input:

expression data at one gene across all samples X_g

group labels $y = (y_i)$

threshold t_1 for Bayes factor

threshold t_2 to filter small mean values

threshold K^* for number of uncertainty positions

hyperparameters $(\alpha, \beta_g, \gamma_g)$

Output: $\mathcal{S}_g$: pruned partitions for gene g

procedure PRUNING($X_g, y, t_1, t_2, K^*, \alpha, \beta_g, \gamma_g$)
  group sample means: $\hat{\mu} \leftarrow X_g, y$
  empirical ordering of $\hat{\mu}$: $o \leftarrow \hat{\mu}$
  groups sizes: n_o
  initialize state between adjacent groups: $ST, ST_j \leftarrow$ uncertain
  uncertain positions: $UC \leftarrow \{\}$
  for o_j in o_1 to o_{K-1} **do**
    if $\max(\hat{\mu}_{o_j}, \hat{\mu}_{o_{j+1}}) < t_2$ **then**
      $ST_j \leftarrow =$
    else
      $D_{o_j,o_{j+1}} \leftarrow$ calculate two-group Bayes factor from$(\hat{\mu}_{o_j}, \hat{\mu}_{o_{j+1}}, n_{o_j}, n_{o_{j+1}}, \alpha, \beta_g, \gamma_g)$
      if $\log(D_{o_j,o_{j+1}}) > t_1$ **then**
        $ST_j \leftarrow \neq$
      else if $\log(D_{o_j,o_{j+1}}) < -t_1$ **then**
        $ST_j \leftarrow =$
      else
        $UC \leftarrow UC \cup \{o_j\}$
      end if
    end if
  end for
  $N_{UC} \leftarrow \min(K^*, |UC|)$
  $UC^* \leftarrow$ positions with smallest N_{UC} absolute value of log Bayes factors in UC.
  $\mathcal{S}_g \leftarrow$ use ST and UC^* to generate
end procedure

Algorithm 2 EQUALHANDLE

Input:

expression data at one gene X_g

group labels $y = (y_i)$

positions for chain of equality states $\tilde{o} = (o_j, ...o_k)$

vector of Bayes factors for adjacent elements along the chain $D_{\tilde{o}} = (D_{o_j,o_{j+1}}, ...D_{o_{k-1},o_k})$

threshold t_1 for Bayes factor

Output: updated $\mathcal{S}_g$

procedure EQUALHANDLE(X_g, y, D_e, e, t_1)
  while $\min|\log(D_e)| > t_1$ **do**
    pick j^* such that $D_{o_{j^*},o_{j^*+1}} = \min(D_{\tilde{o}})$
    update $\tilde{o}$ by merging group o_{j^*} and o_{j^*+1}
    update $D_{\tilde{o}}$ by recalculating Bayes factor between the merged group and its adjacent groups.

  end while
  if $|\tilde{o}| > 1$ **then**
    mark state between groups that are not merged as uncertain
  else
    return
  end if
end procedure

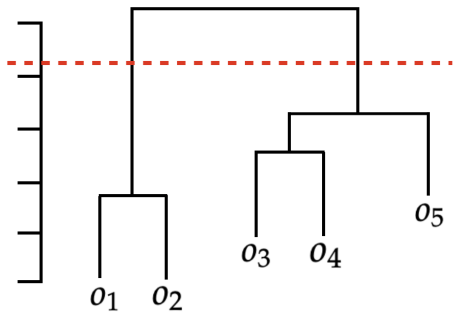

Figure 1: `equalHandle` illustration: $o_1, ..., o_5$ are groups ordered along a chain of equalities suggested by `pruning`. The algorithm sequentially merges adjacent groups having the smallest Bayes factor (strongest evidence for having equal mean) to build a dendrogram. The red line is a threshold where we break the chain, in this case giving two clusters: groups within the same cluster are inferred to have same mean, while the state between clusters remains uncertain, thus adding to the number of possible partitions.

over this selected pool. Compared to **EBSeq.v1**, we also simplify the optimization within the M-step. **EBSeq.v1** uses `optim` to update the hyperparameters (α, β). There is a closed formula for updated mixing rates but hyperparameter optimization by `optim` is more compute intensive. **EBSeq.v2** uses one gradient step toward the optimal solution within each EM iteration. Numerical experiments in Section 3 demonstrate the two methods produce comparable results. Further, **EBSeq.v1** is implemented primarily in R, which, like any other interpreted language, is relatively slow for intensive computing. We deploy core components of **EBSeq.v2** in C++ for efficiency.

Some applications may require flexibility to further reduce the space of partitions. For example, low intrinsic dimensionality of expression data may effectively reduce the number of relevant partitions (Lopez, Regier, Cole, Jordan, and Yosef 2018). In **EBSeq.v2**, we provide an option for users to prune more aggressively, by iteratively removing partitions having small $p_{\text{global},\pi}$. The complete framework is presented as Algorithm 3.

Algorithm 3 EBSEQ.v2

Input:

GENES by SAMPLES expression data $\{X_{g,i}\}$

sample group labels $y = (y_i)$

threshold t_1 for Bayes factor

threshold t_2 to filter small mean values

threshold t_3 for further trimming

threshold K^* for number of uncertainty positions

minimum length L_e of equality chain for doing equal-Handle

initial values for hyper parameters $\alpha^{(0)}, \beta^{(0)} = (\beta_g^{(0)})$

Output:

selected partitions $\mathcal{S}$

GENES $\times$ $\#\mathcal{S}$ matrix of local posterior probabilities $\{p_{\text{local},\pi}\}$

procedure EBSEQ2($X_{g,c}, y, t_1, t_2, t_3, K^*, L_e$)
  for g in GENES **do**
    Estimate γ_g
    $S_g \leftarrow$ run `pruning`
    if there is equality chain with length $> L_e$ **then**
      updated $\mathrm{S}_g \leftarrow$ run `equalHandle`
    end if
  end for
  $\mathcal{S} \leftarrow \cup_g \mathcal{S}_g$
  $p^{(0)}_{\text{global},\pi} \leftarrow \frac{1}{|\mathcal{S}|}$, uniform initialization
  $t = 0$, number of iterations.
  repeat
    if $t > 0$ **then**
      if $p^{(t)}_{\text{global},\pi} < t_3$ **then**
        $p^{(t)}_{\text{global},\pi} \leftarrow 0$
      end if
    end if
    E-step: $p^{(t+1)}_{\text{local},\pi} \leftarrow \text{constant} \times p^{(t)}_{\text{global},\pi} \times P(X|M_{g,\pi}, y, \alpha^{(t)}, \beta^{(t)})$
    M-step: $\alpha^{(t+1)}, \beta^{(t+1)}$ updated by one step gradient ascent.
    M-step: $p^{(t+1)}_{\text{global},\pi} \leftarrow \text{mean}\left(p^{(t+1)}_{\text{local},\pi}\right)$
    $t = t + 1$
  until convergence criterion is met
end procedure

RESULTS

We compare package versions **EBSeq.v2** and **EBSeq.v1** on benchmark data sets where the number of groups K is small. We go on to assess the performance of **EBSeq.v2** in both synthetic and empirical data sets where K is larger.

Benchmarks with small K

We check **EBSeq.v2** on the benchmark data sets built into **EBSeq.v1**. Since $K \leq 3$ in these cases, we use no pruning in deploying **EBSeq.v2**, and so this tests speed and the effect of hyper-parameter optimization differences between the algorithms.

```
R> library(EBSeq)
R> library(EBSeq2)
R> data("GeneMat", package = "EBSeq")
```

This provides a simulated matrix of 1000 rows representing genes and 10 columns representing samples. The columns are ordered such that first and last five columns are corresponding to two different conditions. For both original and new packages, function `EBTest` allows user to get posterior inference of DE patterns among two groups.

```
R> Sizes <- EBSeq::MedianNorm(GeneMat)
R> EBOut <- EBSeq::EBTest(Data = GeneMat, Conditions =
+ as.factor(rep(c("C1","C2"),each = 5)), sizeFactors = Sizes, maxround = 5)
R> EBDERes <- EBSeq::GetDEResults(EBOut, FDR = 0.05)
R> EBOut2 <- EBSeq2::EBTest(Data = GeneMat, Conditions =
+ as.factor(rep(c("C1","C2"),each = 5)), uc = 1,
+ sizeFactors = Sizes, maxround = 50,
+ step1 = 1e-6, step2 = 1e-2, stopthre = 1e-5)
R> EBDERes2 <- EBSeq2::GetDEResults(EBOut2, FDR = 0.05)
```

The proposed method expects normalized data: we provide a median normalization (Love, Huber, and Anders 2014) in case the input data are not normalized. Basically, the function `MedianNorm` calculates the size factors that are further used for normalization and are passed in the `EBTest` function. **EBSeq.v2** inherits all the functions from version 1, but introduces some extra parameters. For example, we add `uc` to represent the number of uncertainty relation between groups' means for each gene; `step1` and `step2` are the step-sizes of gradient ascent for optimizing the hyper-parameters (α, β) of the beta prior; `maxround` and `stopthre` are parameters to determine the termination of the EM algorithm.

With two groups, the primary outputs of **EBSeq** (either version) are gene-specific local posterior probabilities of equivalent and differential expression – so-called EE and DE probabilities. EE corresponds to the partition π having a single block holding all samples (all samples have the same gene-specific expected value). Conditional upon the data, the expected number of EE genes on a list is the sum of the local posterior probabilities (i.e. local false discovery rates); hence we select genes where this EE pattern probability is low.

```
R> head(EBDERes2$PPMat)
```

```
      PPDE      PPEE
Gene_1 1 0.0000000e+00
Gene_2 1 7.826157e-47
Gene_3 1 1.162766e-218
Gene_4 1 2.243618e-33
Gene_5 1 3.526946e-229
Gene_6 1 1.250663e-09
```

The correlation between the PPDE under two algorithms is quite high:

```
R> PPDE1 <- EBDERes$PPMat[,2]
R> PPDE2 <- EBDERes2$PPMat[,1]
R>  ok <- which(!is.na(PPDE1))
R> round(cor(PPDE1[ok], PPDE2[ok]),3)
```

```
0.995
```

Consider a second example:

```
R> data("MultiGeneMat", package = "EBSeq")
```

For this dataset, we have 500 genes $n = 6$ samples and $K = 3$ groups (2 samples per group).

```
R> MultiSize <- EBSeq::MedianNorm(GeneMat)
R> Conditions <- c("C1","C1","C2","C2","C3","C3")
R> PosParti <- GetPatterns(Conditions)
R> MultiOut <- EBSeq::EBMultiTest(Data = MultiGeneMat, NgVector = NULL,
+ Conditions = Conditions, AllParti = Parti, sizeFactors = MultiSize, maxround = 5)
R> MultiPP <- EBSeq::GetMultiPP(MultiOut, FDR = 0.05)
R> MultiOut2 <- EBSeq2::EBMultiTest(Data = MultiGeneMat, Conditions =
+ Conditions, uc = 2, NgVector = rep(1,nrow(MultiGeneMat)),
+ sizeFactors = MultiSize, maxround = 50,
+ step2 = 1e-4, stopthre = 1e-5)
R> MultiPP2 <- EBSeq2::GetDEResults(EBOut2, FDR = 0.05)
```

Function `EBMultiTest` allows user to get posterior inference on DE patterns between multiple groups (more than 3). In the original package, function `GetPatterns` is required for generating all possible patterns that are input in function `EBMultiTest`. In the new package, we get patterns from our pruning algorithm and there is no need to input the set of patterns. At each gene, we consider the patterns π with maximum $p_{\text{local},\pi}$ under the two algorithms and use adjusted rand index (ARI) to measure their similarity.

```
R> MAP1 <- MultiPP$Patterns[match(MultiPP$MAP,rownames(MultiPP$Patterns)),]
R> MAP2 <- MultiPP2$Patterns[match(MultiPP2$MAP,rownames(MultiPP2$Patterns)),]
R> round(mean(sc[!is.na(sc)]),3)
```

```
0.961
```

The average ARI over genes is close to 1, which indicates that the results from **EBSeq.v1** and **EBSeq.v2** are very similar. We increase the number of simulated genes to 10000 and compare the running time of two algorithms under $K = 2, 3$. Even at small K, **EBSeq.v2** is a lot faster than **EBSeq.v1** (Table 4)

EBSeq version	K	Average Run Time (minutes)
v1	2	3.3
v2	2	0.05
v1	3	45
v2	3	0.05

Table 4: Average run time comparison, 20 samples per group, 10000 genes. EM iteration for **EBSeq.v1** is set to 5

Synthetic data, larger K

We use the Chinese restaurant process (CRP) over both genes and samples to simulate synthetic expression data with plausible variation and clustering characteristics. We report numerical experiments with 20000 genes and 200 samples per group for various numbers of groups K; details are in Appendix D.

We evaluate the performance of **EBSeq.v2** using four metrics: coverage, extra patterns, scores and compute time. Let A be the set of patterns selected by **EBSeq.v2**, and B be the true underlying patterns. `Coverage` is $|A \cap B|/|B|$, `extra patterns` is $|A \setminus B|$. `Coverage` measures how well we capture the underlying patterns. `Extra patterns` measures the efficiency, namely how many extra patterns we are not able to filter out. For each gene, there is an underlying pattern π_g and an estimated pattern π'_g maximizing the posterior (MAP pattern). For the accuracy evaluation, we use the adjusted Rand index (ARI) between π_g and π'_g. `Time` is the CPU time of running **EBSeq.v2** using a single core 2.4GHz Intel Xeon E5645 with 126 Gb of RAM.

Figure 2 shows operating characteristics for $K = 15$ and $K = 20$ groups (1.38 billion and 51.7 trillion possible patterns). The pruning algorithm covers almost every true, data-generating pattern (Figure 2a). The ARI scores are close to 1 with small standard deviation (Figure 2c,2d), which shows the proposed method can accurately identify generative patterns. The number of extra patterns and compute time are acceptable considering that these cases are beyond the capacity of **EBSeq.v1** (Figure 2b,2e).

Empirical study

We apply **EBSeq.v2** to three unique-molecule-index (UMI)-based scRNA-seq data sets. These calculations demonstrate the scale of data sets that may be approached with the this new package as well as the sort of inference summaries that become enabled.

The first data set is from a study of mouse visual cortex in which 47,209 cells were classified into main cell types and subtypes through extensive analysis (Hrvatin *et al.* 2018). We preprocessed the data and applied **EBSeq.v2** to a subset of $n = 2164$ cells and $K = 12$ cell

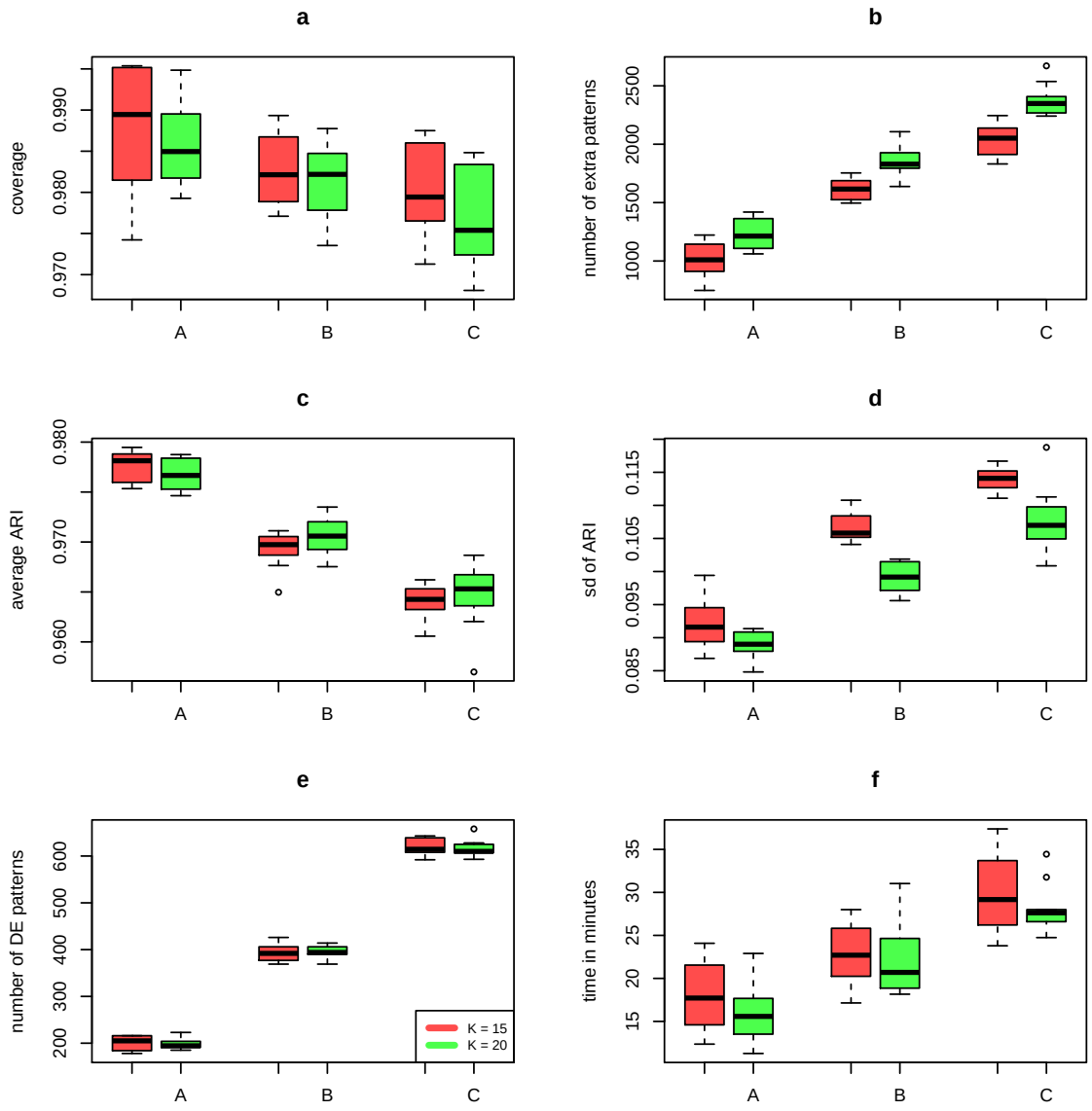

Figure 2: Operating characteristics, synthetic data sets with $K = 15$ (red) or $K = 20$ (green) groups and three settings (horizontal) of a clustering parameter α_0 which affects the expected number of distinct patterns that guide gene-level data. Blocks of genes are generated from Chinese restaurant process, genes within the same block will have same DE patterns. (Approximately 200, 400 and 600 patterns correspond to A, B, and C, respectively.) Ten synthetic data sets are simulated in each setting, and operating characteristics are computed on each data set. Panel *a* presents the coverage percentage; Panel *b* presents the extra patterns we selected but does not belong to the set of true underlying DE patterns; Panel *c* presents the average ARI (adjusted rand index) between the MAP pattern and true pattern; Panel *d* presents the standard deviation of ARI; Panel *e* presents the number of underlying DE patterns across the genome; Panel *f* presents the computation time (minutes)

types. Figure 3 shows expression data from a subset of genes, those whose most probable mean expression pattern matches one particular four-block partition. This maximum a posteriori (MAP) pattern π is identified by maximizing $p_{\text{local},\pi}$ at each gene. As expected

by the selection, mean expression (averaging log expression among samples in each group) is similar within block and quite different between blocks of the MAP partition. Sorting genes by their MAP pattern can be a useful filter in applied settings.

The second data set contains 1000 cells from each of $K = 10$ purified populations derived from peripheral blood[2]. Using **seurat**, (Stuart, Butler, Hoffman, Hafemeister, Papalexi, III, Hao, Stoeckius, Smibert, and Satija 2019), we carried out uniform manifold approximation projection (UMAP) visualizations of the cells considering whole genome (Panel A) versus genes with two particular MAP patterns (Panels B, C) and display results in Figure 4. A single amorphous cluster (Panel B) forms when using genes whose MAP pattern is equivalent expression across all cell types (i.e., the single-block π is the MAP pattern for these genes). The UMAP highlights other clusters when we use genes whose MAP pattern matches a partition of interest, such as in Panel C where specific lymphocytes are distinguished. In practice, the use of posterior probabilities for such gene selections can be more sensitive than simpler filters.

The third data set contained $n = 27,499$ mouse retinal bipolar cells, we used cluster annotation from $K = 14$ cell types from the author (Shekhar *et al.* 2016). We selected 2 genes "Epha7" and "BC030499" favorable for two partitions and show the cumulative distribution functions (cdfs) across the blocks of these partitions (Figure 5). We observe distributional differences between cells presenting differential means as expected by the selection.

To further check the overall performance of **EBSeq.v2**, we compare pairwise fold change statistics between cellular groups with marginal posterior probabilities of differential expression (DE) between these groups, in the third example with $K = 12$ cellular conditions. Pattern posterior probabilities produced by **EBSeq.v2** are marginalized to get pairwise DE probabilities: $P(\mu_i \neq \mu_j | X, y) = \sum_{\pi \in \zeta_{i,j}} p_{\text{local},\pi}$ where $\zeta_{i,j}$ is the collection of partitions in which groups i and j belong to different blocks. Figure 6 compares DE probabilities (below diagonal) to empirical fold changes (above diagonal), here standardized and averaged over the genome, and reveals consistency between these two measures of group difference. Fold changes are often reported, but posterior probabilities have the capacity for self calibration in terms of false discovery rate.

[2]Available `https://support.10xgenomics.com/single-cell-gene-expression/datasets`

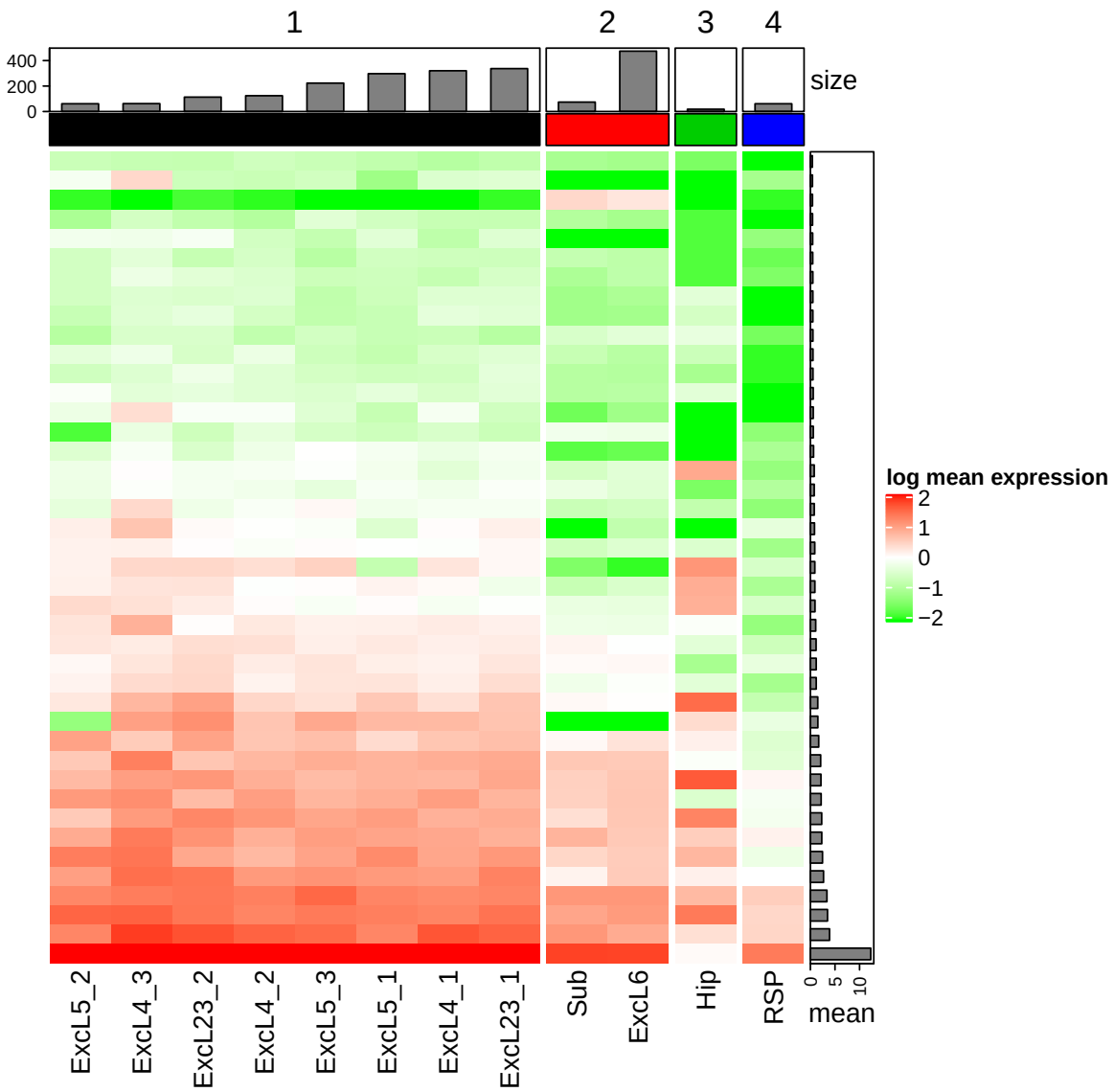

Figure 3: Heatmap of the mean expression on log scale across $K = 12$ groups at those genes with one four-block MAP pattern; genes are filtered by mean expression bigger than 0.4. The top barplot shows the number of cells each group ordered within each block, the right barplot shows the ordered marginal mean across all cells. Data are mouse cortex cells from (Hrvatin *et al.* 2018)

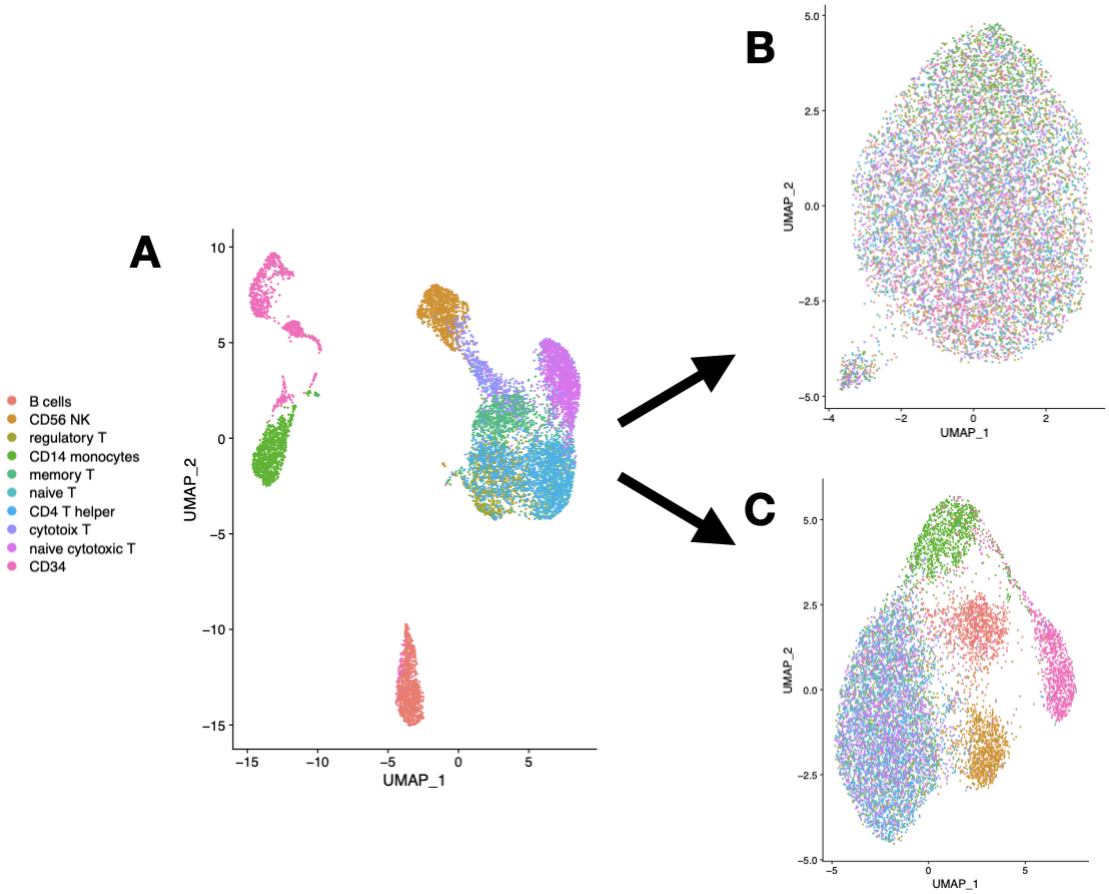

Figure 4: UMAP of PBMC data, left considering whole genome. Panel B only uses genes with null partition as the MAP. Panel C uses genes with MAP patterns distinguishing B cells, CD14, CD34, CD56.

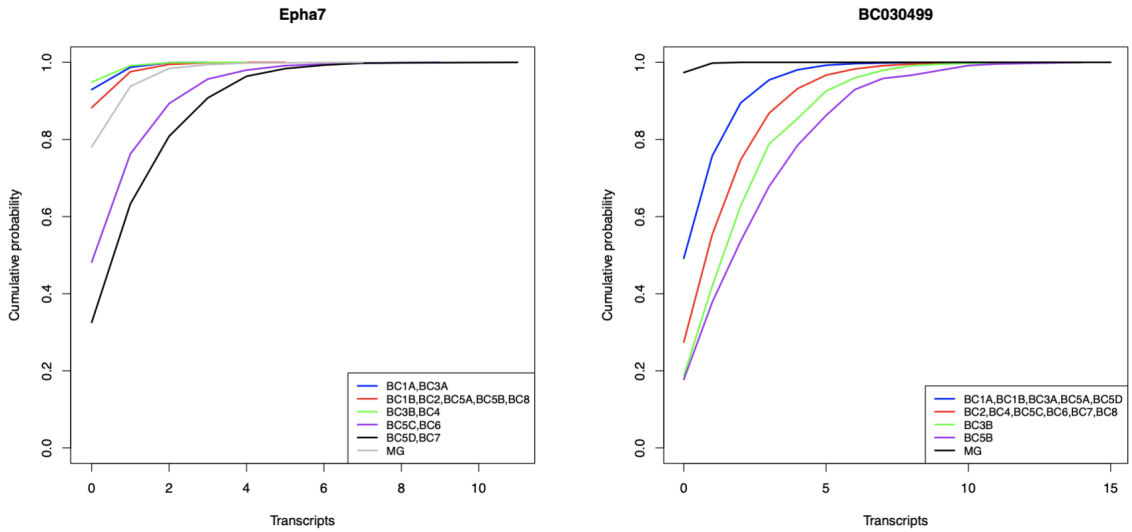

Figure 5: Cumulative distribution of expression at two genes "Epha7" and "BC030499" distinguished by cell condition. Colors represent blocks of conditions sharing the same mean expression in the MAP pattern for these genes. Data are from murine bipolar cells from RETINA (Shekhar *et al.* 2016).

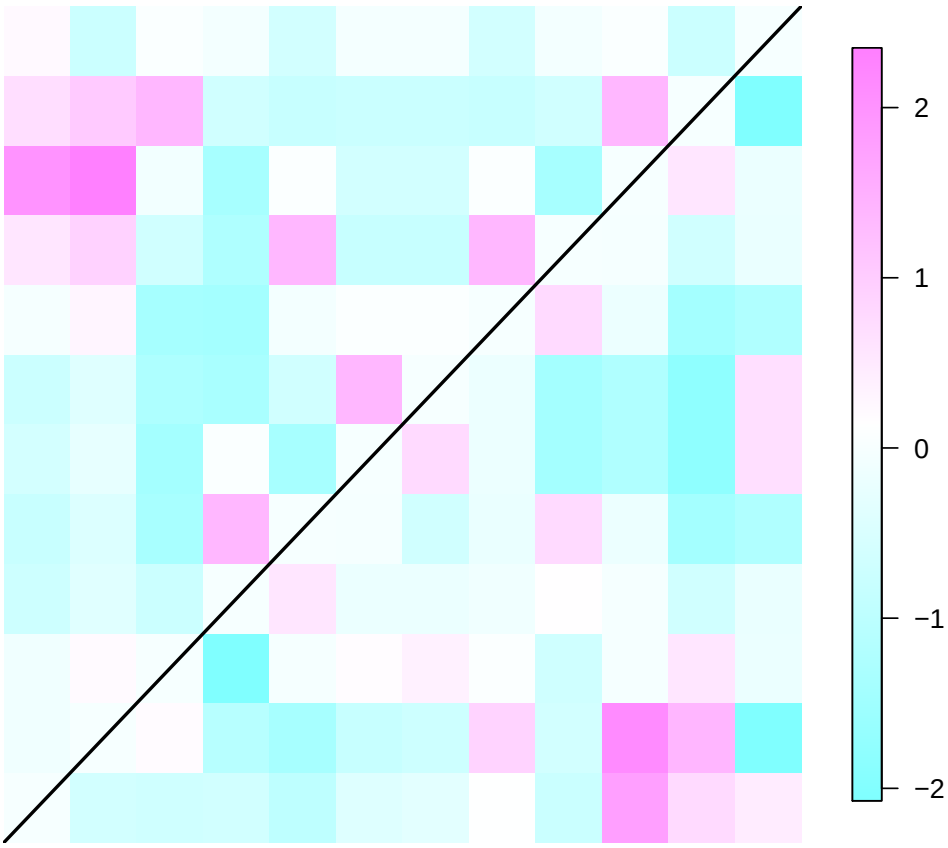

Figure 6: Heatmap shows concordance between standardized fold change (above diagonal) and marginal posterior DE probabilities (below diagonal) between all pairs of $K = 12$ cellular groups using data (Hrvatin *et al.* 2018). Both measures are standardized by centering to zero, scaling to unit standard deviation, and averaging over the genome.

DISCUSSION

Technologies to measure high-dimensional transcript profiles are still developing, and we expect more large-scale data sets from experiments on immune response, cancer progression, tissue development, and other biological systems. Investigating patterns of gene expression is an essential component of data analysis, e.g., for identifying genes with distributional changes across multiple biological conditions. The mixture of negative binomials model encoded in **EBSeq** approximates the variation of gene expression data in many settings, and the proposed version update extends the domain of application of this important inference tool.

Using an empirical Bayesian approach, **EBSeq** evaluates genes according to likely patterns of differential expression they display over two or more biological conditions. For all genes in a system and for all sorts of patterns of differential expression among sample groups, **EBSeq** scores patterns with local posterior probabilities, and these may be used to measure the false discovery rate of any list of identified genes or to filter genes according to their MAP pattern. In the proposed version 2.0 of **EBSeq**, `pruning` reduces the number of components of the mixture model by virtue of a theoretically guided pairwise Bayes factor calculation, which is key to scaling model-based computations as the number of cellular conditions grows. Also, `equalHandle` deals with a corner case and keeps the model sensitive to the difference between groups. Synthetic and empirical examples demonstrate the efficiency and accuracy of this new version. Future efforts could enable further extensions, such as to allow flexibility in distributional forms while retaining the partition/pattern structure for inference summarization.

Computational details

The results in this paper were obtained using R 3.5.1 with the packages **Rcpp** 0.12.11, **RcppEigen** 0.3.2.9.0, **BH** 1.69.0-1. R itself and all packages used are available from the Comprehensive R Archive Network (CRAN) at https://CRAN.R-project.org/. **EBSeq.v2** is maintained under the original package name at http://github.com/wiscstatman/EBSeq.

Acknowledgments

This research was supported in part by NIH grants P50 DE026787, P30CA14520-45, R01 GM102756, NSF grant 1740707, and a Data Science Initiative grant from the University of Wisconsin Office of the Vice Chancellor for Research and Graduate Education. This manuscript was prepared using `overleaf`.

References

Aldous DJ, Ibragimov IA, Jacod J (2006). *Ecole d'Ete de Probabilites de Saint-Flour XIII, 1983*, volume 1117. Springer. ISBN 3540393161.

Baños-Lara MDR, Zabaleta J, Garai J, Baddoo M, Guerrero-Plata A (2018). "Comparative analysis of miRNA profile in human dendritic cells infected with respiratory syncytial virus and human metapneumovirus." *BMC research notes*, **11**(1), 432.

Boost (2015). "Boost C++ Libraries." http://www.boost.org/. Last accessed 2015-06-30.

Dahl DB (2009). "Modal clustering in a class of product partition models." *Bayesian Anal.*, **4**(2), 243–264. doi:10.1214/09-BA409. URL https://doi.org/10.1214/09-BA409.

Dempster AP, Laird NM, Rubin DB (1977). "Maximum likelihood from incomplete data via the EM algorithm." *Journal of the Royal Statistical Society: Series B (Methodological)*, **39**(1), 1–22.

Deng Q, Ramsköld D, Reinius B, Sandberg R (2014). "Single-Cell RNA-Seq Reveals Dynamic, Random Monoallelic Gene Expression in Mammalian Cells." *Science*, **343**(6167), 193–196. ISSN 0036-8075. doi:10.1126/science.1245316. https://science.sciencemag.org/content/343/6167/193.full.pdf, URL https://science.sciencemag.org/content/343/6167/193.

Engel I, Seumois G, Chavez L, Samaniego-Castruita D, White B, Chawla A, Mock D, Vijayanand P, Kronenberg M (2016). "Innate-like functions of natural killer T cell subsets result from highly divergent gene programs." *Nature immunology*, **17**(6), 728–739.

Gardner M (1978). "Mathematical Games." *Scientific American*, **238**(5), 24–30. doi:10.1038/scientificamerican0578-24.

Guennebaud G, Jacob B, *et al.* (2010). "Eigen v3." http://eigen.tuxfamily.org.

Heller KA, Ghahramani Z (2005). "Bayesian hierarchical clustering." In *Proceedings of the 22nd international conference on Machine learning*, pp. 297–304.

Hrvatin S, Hochbaum DR, Nagy MA, Cicconet M, Robertson K, Cheadle L, Zilionis R, Ratner A, Borges-Monroy R, Klein AM, *et al.* (2018). "Single-cell analysis of experience-dependent transcriptomic states in the mouse visual cortex." *Nature neuroscience*, **21**(1), 120–129.

Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, Bravo HC, Davis S, Gatto L, Girke T, Gottardo R, Hahne F, Hansen KD, Irizarry RA, Lawrence M, Love MI, MacDonald J, Obenchain V, Ole's AK, Pag'es H, Reyes A, Shannon P, Smyth GK, Tenenbaum D, Waldron L, Morgan M (2015). "Orchestrating high-throughput genomic analysis with Bioconductor." *Nature Methods*, **12**(2), 115–121. URL http://www.nature.com/nmeth/journal/v12/n2/full/nmeth.3252.html.

Kendziorski C, Newton M, Lan H, Gould M (2003). "On parametric empirical Bayes methods for comparing multiple groups using replicated gene expression profiles." *Statistics in medicine*, **22**(24), 3899–3914.

Lee CJ, Ahn H, Jeong D, Pak M, Moon JH, Kim S (2020). "Impact of mutations in DNA methylation modification genes on genome-wide methylation landscapes and downstream gene activations in pan-cancer." *BMC Medical Genomics*, **13**(3), 1–14.

Leng N, Dawson JA, Thomson James Aand Ruotti V, Rissman AI, Smits BMG, Haag JD, Gould MN, Stewart RM, Kendziorski C (2013). "EBSeq: an empirical Bayes hierarchical model for inference in RNA-seq experiments." *Bioinformatics*, **29**(8), 1035–1043. ISSN 1367-4803. doi:10.1093/bioinformatics/btt087. https://academic.oup.com/bioinformatics/article-pdf/29/8/1035/17106263/btt087.pdf, URL https://doi.org/10.1093/bioinformatics/btt087.

Leng N, Kendziorski C (2019). *EBSeq: An R package for gene and isoform differential expression analysis of RNA-seq data*. R package version 1.24.0.

Lopez R, Regier J, Cole MB, Jordan MI, Yosef N (2018). "Deep generative modeling for single-cell transcriptomics." *Nature Methods*, **15**(12), 1053–1058. doi:10.1038/s41592-018-0229-2. URL https://doi.org/10.1038/s41592-018-0229-2.

Louro B, Marques JP, Manchado M, Power DM, Campinho MA (2020). "Sole head transcriptomics reveals a coordinated developmental program during metamorphosis." *Genomics*, **112**(1), 592–602.

Love MI, Huber W, Anders S (2014). "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2." *Genome biology*, **15**(12), 550.

Ma BX, Korthauer K, Kendziorski C, Newton MA (2019). "A Compositional Model to Assess Expression Changes from Single-Cell Rna-Seq Data." *bioRxiv*. doi:10.1101/655795. https://www.biorxiv.org/content/early/2019/05/31/655795.full.pdf, URL https://www.biorxiv.org/content/early/2019/05/31/655795.

MacEachern SN (1998). "Computational methods for mixture of Dirichlet process models." In *Practical nonparametric and semiparametric Bayesian statistics*, pp. 23–43. Springer.

Newhouse DJ, Hofmeister EK, Balakrishnan CN (2017). "Transcriptional response to West Nile virus infection in the zebra finch (Taeniopygia guttata)." *Royal Society open science*, **4**(6), 170296.

O'Grady T, Baddoo M, Flemington EK (2017). "Analysis of EBV Transcription Using High-Throughput RNA Sequencing." In *Epstein Barr Virus*, pp. 105–121. Springer.

Patel AP, Tirosh I, Trombetta JJ, Shalek AK, Gillespie SM, Wakimoto H, Cahill DP, Nahed BV, Curry WT, Martuza RL, *et al.* (2014). "Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma." *Science*, **344**(6190), 1396–1401.

Quintana FA, Iglesias PL (2003). "Bayesian clustering and product partition models." *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **65**(2), 557–574.

Sabbagh MF, Heng JS, Luo C, Castanon RG, Nery JR, Rattner A, Goff LA, Ecker JR, Nathans J (2018). "Transcriptional and epigenomic landscapes of CNS and non-CNS vascular endothelial cells." *Elife*, **7**, e36187.

Sanders SM, Cartwright P (2015). "Patterns of Wnt signaling in the life cycle of Podocoryna carnea and its implications for medusae evolution in Hydrozoa (Cnidaria)." *Evolution & development*, **17**(6), 325–336.

Shekhar K, Lapan SW, Whitney IE, Tran NM, Macosko EZ, Kowalczyk M, Adiconis X, Levin JZ, Nemesh J, Goldman M, *et al.* (2016). "Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics." *Cell*, **166**(5), 1308–1323.

Son JC, Jeong HO, Park D, No SG, Lee EK, Lee J, Chung HY (2017). "miR-10a and miR-204 as a potential prognostic indicator in low-grade gliomas." *Cancer informatics*, **16**, 1176935117702878.

Song X, Tang T, Li C, Liu X, Zhou L (2018). "CBX8 and CD96 are important prognostic biomarkers of colorectal cancer." *Medical science monitor: international medical journal of experimental and clinical research*, **24**, 7820.

Stuart T, Butler A, Hoffman P, Hafemeister C, Papalexi E, III WMM, Hao Y, Stoeckius M, Smibert P, Satija R (2019). "Comprehensive Integration of Single-Cell Data." *Cell*, **177**, 1888–1902. doi:10.1016/j.cell.2019.05.031. URL https://doi.org/10.1016/j.cell.2019.05.031.

Yang F, Lv SX, Lv L, Liu YH, Dong SY, Yao ZH, Dai Xx, Zhang XH, Wang OC (2016). "Identification of lncRNA FAM83H-AS1 as a novel prognostic marker in luminal subtype breast cancer." *OncoTargets and therapy*, **9**, 7039.

Yoon KJ, Ringeling FR, Vissers C, Jacob F, Pokrass M, Jimenez-Cyrus D, Su Y, Kim NS, Zhu Y, Zheng L, *et al.* (2017). "Temporal control of mammalian cortical neurogenesis by m6A methylation." *Cell*, **171**(4), 877–889.

Yuan M, Newton M, Sarkar D, Kendziorski C (2019). *EBarrays: Unified Approach for Simultaneous Gene Clustering and Differential Expression Identification*. R package version 2.50.0.

Zhang Q, Zeng LP, Zhou P, Irving AT, Li S, Shi ZL, Wang LF (2017). "IFNAR2-dependent gene expression profile induced by IFN-α in Pteropus alecto bat cells and impact of IFNAR2 knockout on virus infection." *PloS one*, **12**(8).

A. EBSeq margin

A core computation is to evaluate the joint probability mass f of input data recordings $X = (X_1, \cdots, X_m)$ that have a common, unknown, mean, whose value is integrated out (i.e., continuous mixing). In **EBSeq**, we integrate a Negative Binomial mass function, $\mathrm{NB}(q, \gamma)$, which has mean $q\gamma/(1-q)$, against a Beta prior for q.

$$f(x) \quad = \quad \int_0^1 \prod_{i=1}^{m} [p(x_i|\gamma, q)] \; p(q|\alpha, \beta) \, dq$$

where $x|\gamma, q$ is a Negative Binomial(γ, q) distribution and $q|\alpha, \beta$ is a Beta distribution with shape parameters α and β. The result is reported in (0.0.2), where recall $\mathrm{Beta}(a, b) = \Gamma(a)\Gamma(b)/\Gamma(a+b)$, and for $x > 0$, $\Gamma(x) = \int_0^\infty u^{x-1} \exp\{-u\} \, du$. In this representation, the expression data X are expected to be normalized and the shape parameter γ is common across samples. Thus the conditional mean μ is fully determined by q.

In **EBSeq** hyperparameters are unit specific, except for α which is shared across all units. Different units may have different β values, but **EBSeq** allows some blocking of units if

they are all isoforms of the same gene, in which case β is estimated for the block. Hyperparameters γ_g are set by the method of moments. The other hyperparameters parameters and the mixing rates $p_{\text{global},\pi}$ are estimated via an EM algorithm using data on all samples and all units (Dempster, Laird, and Rubin 1977).

B. Proofs of Lemma 1 and Theorem 1

Proof of Lemma 1. We prove a 1-1 correspondence between compatible partitions and ways of assigning "=" or "$\neq$" in each slot in equation (0.0.4). The cardinality 2^{K-1} follows immediately. Without loss of generality, we label groups so that the empirical rank vector r is $r = (1, 2, \cdots, K)$ and the sample means are ordered, $\hat{\mu}_1 \leq \cdots \leq \hat{\mu}_K$. Such labeling induces $A_r(b) = b$. We fix partitions by resolving slots in equation (0.0.4), which becomes

$$\mu_1 _ \mu_2 _ \cdots _ \mu_K. \tag{B.0.1}$$

Denote the number of blocks in π as $N(\pi)$. For the trivial case $N(\pi) = 1$, π can be represented by assigning equality "=" to all slots in (B.0.1). If some partition π with $N(\pi) > 1$ is compatible with r, then $A_r(b_j)$, $A_r(b_k)$ are not overlapping for any two blocks $b_j, b_k \in \pi$, and so either $\max(b_j) < \min(b_k)$ or $\max(b_k) < \min(b_j)$. (These extrema are taken over group labels within the block.) For convenience, denote $v_j = \min(b_j)$ and $w_j = \max(b_j)$ and let blocks $b_1, \cdots, b_{N(\pi)}$ of this compatible partition be labeled so they are ordered by the largest element: $w_1 \leq w_2 \cdots \leq w_{N(\pi)}$. Since b_j and b_k are not overlapping, we have $1 = v_1 \leq w_1 < v_2 \leq w_2 \cdots < v_{N(\pi)} \leq w_{N(\pi)} = K$, and further $v_{j+1} = w_j + 1, j = 1, \cdots N(\pi) - 1$. Thus, the compatible π induces the assignment:

$$\mu_1 = \cdots = \mu_{w_1};\ \mu_{w_1} \neq \mu_{v_2};\ \mu_{v_2} = \cdots = \mu_{w_2};\ \cdots;\ \mu_{v_{N(\pi)}} = \cdots = \mu_K.$$

To prove the opposite direction, consider any assignment of "$\neq$" or "=" for the slots in equation (B.0.1), and define partition π by the rule that group indices are in the same block if and only if they have equal means by this assignment. If π is not compatible with r, then it must contain distinct, overlapping blocks b and $\tilde{b}$ with $v = \min(b) < \tilde{w} = \max(\tilde{b}) < w = \max(b)$. But such an arrangement would imply $\mu_v = \mu_{\tilde{w}} = \mu_w$, which contradicts the presumption that w and $\tilde{w}$ are in different blocks. Thus π is compatible with r.

□

Without loss of generality, we label adjacent groups by $j = 1, 2$ and recall the notation from Section 2.2 where $X_{\{j\}}$ denotes the vector of n_j sample measurements in group j at the unit on test. In preparing to prove Theorem 1, we decompose predictive functions in (0.0.2) which are specific mixtures of Negative Binomial masses:

$$\begin{aligned}
\log f\left(X_{\{1,2\}}\right) &= \phi[\alpha + (n_1 + n_2)\gamma] + \phi[\beta + n_1\hat{\mu}_1 + n_2\hat{\mu}_2] - \\
&\quad \phi[\alpha + \beta + (n_1 + n_2)\gamma + n_1\hat{\mu}_1 + n_2\hat{\mu}_2] + \phi(\alpha) + \phi(\beta) - \phi(\alpha + \beta) \\
\log\left[f\left(X_{\{1\}}\right) f\left(X_{\{2\}}\right)\right] &= \phi(\alpha + n_1\gamma) + \phi(\beta + n_1\hat{\mu}_1) - \phi(\alpha + \beta + n_1\gamma + n_1\hat{\mu}_1) + \\
&\quad \phi(\alpha + n_2\gamma) + \phi(\beta + n_2\hat{\mu}_2) - \phi(\alpha + \beta + n_2\gamma + n_2\hat{\mu}_2) + \\
&\quad 2(\phi(\alpha) + \phi(\beta) - \phi(\alpha + \beta))
\end{aligned}$$

where $\phi(x) = \log\Gamma(x)$, $\Gamma(x)$ is the usual Gamma function, hyperparameters α, β, and γ are fixed positive integers, and $\hat{\mu}_j = \sum_i X_i 1[y_i = j]/n_j$ with ordering $\hat{\mu}_1 \leq \hat{\mu}_2$. We use this decomposition to understand the two-group Bayes factor (0.0.3), which in this setting is:

$$g(n_1, n_2, \hat{\mu}_1, \hat{\mu}_2) := \log(D_{1,2}) = \log\left[f\left(X_{\{1\}}\right) f\left(X_{\{2\}}\right)\right] - \log f\left(X_{\{1,2\}}\right).$$

In the notation we suppress dependence of the two-group Bayes factor on the hyperparameters; these are fixed throughout and our analysis rests on changes in g with respect to changes in the other inputs. (We also note that in applications the hyperparameters may take positive, non-integer values; we restrict to integers in the theorem only because the proof method is more straightforward.)

Lemma 2. *If $\hat{\mu}_2$ is sufficiently larger than $\hat{\mu}_1$, then $g(n_1, n_2, \hat{\mu}_1, \hat{\mu}_2) > 0$.*

Proof of Lemma 2. For notation, let $S_j = n_j\hat{\mu}_j$, $R_j = n_j\gamma$, for $j = 1, 2$, and recall that for positive integers x, $\phi(x+1) = \log(x!)$. Manipulating the two-group Bayes factor we obtain:

$$\begin{aligned} g = &\left[\log\frac{(\alpha+\beta+R_1+R_2+S_1+S_2-1)!}{(\alpha+\beta+R_1+S_1-1)!\,(\alpha+\beta+R_2+S_2-1)!} - \phi(\alpha+\beta)\right] \\ &- \left[\log\frac{(\alpha+R_1+R_2-1)!}{(\alpha+R_1-1)!\,(\alpha+R_2-1)!} + \phi(\alpha)\right] - \left[\log\frac{(\beta+S_1+S_2-1)!}{(\beta+S_1-1)!\,(\beta+S_2-1)!} + \phi(\beta)\right] \\ &:= A - B - C \end{aligned}$$

Further we have

$$\begin{aligned} A &= \log\frac{\prod_{i=0}^{R_1+S_1-1}(\alpha+\beta+R_2+S_2+i)}{\prod_{i=0}^{R_1+S_1-1}(\alpha+\beta+i)} - 2\phi(\alpha+\beta) \\ &= \log\left[\prod_{i=0}^{R_1+S_1-1}\left(1+\frac{R_2+S_2}{\alpha+\beta+i}\right)\right] - 2\phi(\alpha+\beta) \\ &\geq (R_1+S_1)\log\left(1+\frac{R_2+S_2}{\alpha+\beta+R_1+S_1-1}\right) - 2\phi(\alpha+\beta), \\ B &= \log\frac{\prod_{i=0}^{R_1-1}(\alpha+R_2+i)}{\prod_{i=0}^{R_1-1}(\alpha+i)} \\ &= \log\left[\prod_{i=0}^{R_1-1}\left(1+\frac{R_2}{\alpha+i}\right)\right] \leq R_1\log\left(1+\frac{R_2}{\alpha}\right), \\ C &= \log\frac{\prod_{i=0}^{S_1-1}(\beta+S_2+i)}{\prod_{i=0}^{S_1-1}(\beta+i)} \leq S_1\log\left(1+\frac{S_2}{\beta}\right). \end{aligned}$$

Thus,

$$
\begin{aligned}
g &\geq (R_1+S_1)\log\left(1+\frac{R_2+S_2}{\alpha+\beta+R_1+S_1-1}\right) - R_1\log\left(1+\frac{R_2}{\alpha}\right) - S_1\log\left(1+\frac{S_2}{\beta}\right) - 2\phi(\alpha+\beta) \\
&\geq S_1\left[\log\left(1+\frac{1+S_2}{\alpha+\beta+R_1+S_1}\right) - \log(1+S_2)\right] + R_1\left[\log\left(1+\frac{1+S_2}{\alpha+\beta+R_1+S_1}\right) - \log(1+R_2)\right] \\
&- 2\phi(\alpha+\beta) \\
&\geq S_1\log\left(\frac{1}{\alpha+\beta+R_1+S_1}\right) + R_1\log\left(\frac{1+S_2}{(\alpha+\beta+R_1+S_1)(1+R_2)}\right) - 2\phi(\alpha+\beta)
\end{aligned}
$$

This becomes positive as $S_2 = n_2\hat{\mu}_2$ increases. □

Recall the digamma function $\psi(x) = \phi'(x)$, and the recursive property of ψ:

Lemma 3. *For positive real x and natural N, changes in the digamma function ψ satisfy:*

$$\psi(x+N) - \psi(x) = \sum_{k=0}^{N-1} \frac{1}{x+k}$$

Lemma 4. *If $\hat{\mu}_2$ is sufficiently larger than $\hat{\mu}_1 > 0$, then g is:*

a) monotone decreasing in $\hat{\mu}_1$

b) monotone increasing in $\hat{\mu}_2$

c) monotone increasing in n_1

d) monotone increasing in n_2

These monotonicities are in one variable at a time with the others fixed.

Proof of Lemma 4. We use notation for S_j and R_j as in the proof of Lemma 2, but treat g as a differentiable function of its arguments and study its partial derivatives. To prove a), we have the derivative

$$
\begin{aligned}
\frac{1}{n_1}\frac{\partial g}{\partial \hat{\mu}_1} &= [\psi(\alpha+\beta+S_1+S_2+R_1+R_2) - \psi(\beta+S_1+S_2)] - [\psi(\alpha+\beta+S_1+R_1) - \psi(\beta+S_1)] \\
&:= A - B. \qquad \text{(B.0.2)}
\end{aligned}
$$

Using the Lemma 3 separately on A and B leads to

$$A < \frac{\alpha+R_1+R_2}{\beta+S_1+S_2} \qquad \text{and} \qquad B > \frac{\alpha+R_1}{\alpha+\beta+S_1+R_1}.$$

The derivative (B.0.2) is negative if $A < B$, which is assured when $\hat{\mu}_2$ is sufficiently in excess of $\hat{\mu}_1$ because then $S_2 = n_2\hat{\mu}_2$ is large enough to push A below B.

To prove b), the derivative for $\hat{\mu}_2$ is

$$\begin{aligned}\frac{1}{n_2}\frac{\partial g}{\partial \hat{\mu}_2} &= [\psi(\alpha+\beta+S_1+S_2+R_1+R_2)-\psi(\beta+S_1+S_2)]-[\psi(\alpha+R_2+\beta+S_2)-\psi(\beta+S_2)] \\ &:= C-D. \end{aligned} \tag{B.0.3}$$

Using Lemma 3 again, we have

$$C > \frac{\alpha+R_1+R_2}{\alpha+\beta+S_1+S_2+R_1+R_2} \qquad \text{and} \qquad D < \frac{\alpha+R_2}{\beta+S_2},$$

which guarantees positive derivative B.0.3 if

$$\frac{\beta+S_2}{\alpha+\beta+S_1+S_2+R_1+R_2} > \frac{\alpha+R_2}{\alpha+R_1+R_2}.$$

This holds when $\hat{\mu}_2$ is sufficiently larger than $\hat{\mu}_1$ because the left hand side approaches 1 while right hand side remains bounded below 1. To prove c), the derivative for n_1 is

$$\begin{aligned}\frac{\partial g}{\partial n_1} &= \gamma\psi(\alpha+R_1)+\hat{\mu}_1\psi(\beta+S_1)-(\gamma+\hat{\mu}_1)\psi(\alpha+R_1+\beta+S_1)-\gamma\psi(\alpha+R_1+R_2) \\ &\quad -\hat{\mu}_1\psi(\beta+S_1+S_2)+(\gamma+\hat{\mu}_1)\psi(\alpha+R_1+R_2+\beta+S_1+S_2). \\ &= \gamma\left\{[\psi(\alpha+R_1+R_2+\beta+S_1+S_2)-\psi(\alpha+R_1+R_2)]-[\psi(\alpha+R_1+\beta+S_1)-\psi(\alpha+R_1)]\right\} \\ &\quad +\hat{\mu}_1\left\{[\psi(\alpha+R_1+R_2+\beta+S_1+S_2)-\psi(\beta+S_1+S_2)]-[\psi(\alpha+R_1+\beta+S_1)-\psi(\beta+S_1)]\right\} \\ &:= \gamma(E-F)+\hat{\mu}_1(G-H)\end{aligned}$$

Following the same approach using Lemma 3, we have

$$E = \Sigma_{i=0}^{\beta+S_1+S_2}\left(\frac{1}{\alpha+R_1+R_2+i}\right), \quad F < \frac{\beta+S_1}{\alpha+R_1}, \quad G > 0, \quad \text{and} \quad H < \frac{\alpha+R_1}{\beta+S_1}.$$

$E > F+H$ is assured for sufficiently large $\hat{\mu}_2 = S_2/n_2$ since E tracks a divergent sum.

Proof of d) is similar and details are omitted. □

Proof of Theorem 1. Without loss of generality we label the groups $1, 2, \cdots, K$, so that their empirical means are non-decreasing: $\hat{\mu}_1 \le \hat{\mu}_2 \le \cdots \le \hat{\mu}_K$. Let ζ be the set of all possible partitions of K elements (numbering B_K), and for some $j < K$ where $\hat{\mu}_j < \hat{\mu}_{j+1}$, and let ζ_e be the subset of these partitions assigning equivalent means $\mu_j = \mu_{j+1}$ to groups j and $j+1$. In other words, every partition in ζ_e is such that j and $j+1$ are elements of the same partition block. Note that sample sizes n_j and n_{j+1} are associated with these two groups of samples. Further, let $\zeta_d = \zeta \setminus \zeta_e$ be set of partitions assigning differential means to j and $j+1$. As for partitions in ζ_e, we can merge $X_{\{j\}}$ and $X_{\{j+1\}}$ into one group and view ζ_e as partitions on $K-1$ elements, so $|\zeta_e| = B_{K-1}$, while $|\zeta_d| = B_K - B_{K-1}$. Therefore, $|\zeta_e| < |\zeta_d|$. There is an injection mapping from ζ_e to ζ_d. Any block $b \in \pi \in \zeta_e$ that excludes j and $j+1$ is part of the partition $\pi' \in \zeta_d$ uniquely associated with π. The single block $\tilde{b} \in \pi \in \zeta_e$ that contains both j and $j+1$ is split into two blocks $\tilde{b}_l$ and $\tilde{b}_r$ in π'; this is the one and only split for which $\hat{\mu}_j$ is the largest sample mean in $\tilde{b}_l$ and $\hat{\mu}_{j+1}$ is the smallest sample mean in block $\tilde{b}_r$; let us say that these two blocks contain n_l and n_r samples, respectively.

In searching for a partition to maximize the the joint predictive mass of all data X on the unit in focus, consider the ratio of predictive masses comparing partitions π and π' defined above:

$$\begin{aligned}
\frac{P(X|M_{\pi'}, y)}{P(X|M_{\pi}, y)} &= \frac{\prod_{b\in\pi'} f(X_b)}{\prod_{b\in\pi} f(X_b)} \\
&= \frac{f(X_{\tilde{b}_l})\, f(X_{\tilde{b}_r}) \prod_{b\in\pi';b\neq\tilde{b}_l;b\neq\tilde{b}_r} f(X_b)}{f(X_{\tilde{b}}) \prod_{b\in\pi;b\neq\tilde{b}} f(X_b)} \\
&= \frac{f(X_{\tilde{b}_l})\, f(X_{\tilde{b}_r})}{f(X_{\tilde{b}})} \\
&= \exp\left[g(n_l, n_r, \hat{\mu}_{\tilde{b}_l}, \hat{\mu}_{\tilde{b}_r})\right] \\
&\geq \exp\left[g(n_j, n_{j+1}, \hat{\mu}_j, \hat{\mu}_{j+1})\right] \qquad \text{if } \hat{\mu}_j, \hat{\mu}_{j+1} \text{ are sufficiently distinct, by Lemma 4} \\
&= \frac{f(X_{\{j\}}) f(X_{\{j+1\}})}{f(X_{\{j,j+1\}})} \\
&> 1 \qquad \text{further by Lemma 2}
\end{aligned}$$

The first inequality above uses the monotonicity in Lemma 4 together with the readily confirmed facts that: $n_l \geq n_j$, $n_r \geq n_{j+1}$, $\hat{\mu}_{\tilde{b}_l} \leq \hat{\mu}_j$, and $\hat{\mu}_{\tilde{b}_r} \geq \hat{\mu}_{j+1}$.

Thus, the partition with maximal $P(X|M_\pi, y)$ must assign differential means between groups j and $j+1$ when $\hat{\mu}_j$ and $\hat{\mu}_{j+1}$ are sufficiently distinct.

□

C. Further computational details

C.1. Number of groups K

For all the empirical examples used in Section 2, their group number K is determined via clustering analysis in **scDDboost** (Ma *et al.* 2019).

C.2. Hyperparameters of two-group Bayes factor

For different hyperparameters the selected partitions $\mathcal{S}$ may vary. To measure the similarity between two sets of partitions $\mathcal{S}_1$ and $\mathcal{S}_2$, we have the metric:

$$\mathcal{I} = \frac{|\mathcal{S}_1 \cap \mathcal{S}_2|}{|\mathcal{S}_1 \cup \mathcal{S}_2|}.$$

The default value of α is 0.4, and β is set to 2 for all units.

α \ β	1	2	3
0.3	0.97	0.93	0.91
0.4	0.9	1	0.94
0.5	084	0.91	0.97

Table 5: $\mathcal{I}$ for comparing $\mathcal{S}$ under different hyperparamters to the default setting ($\alpha = 0.4, \beta = 2$). Results are averaged over data sets GSE74596 and GSE57872

Table shows that the final selected partitions are robust to the choice of (α, β) used in two-group Bayes factor. The scale of α and β are determined by estimations from **EBSeq.v1** applied on benchmark data sets built into it and an empirical data GSE45719. Similar calculations (not shown) confirm that the two-group Bayes factor is quite insensitive to plausible values of hyperparameter γ.

C.3. Number of uncertain positions

We provide an option to limit the number of uncertain positions. Users can specify a upper bound U^* for $N_{\mathrm{UC},g}$, which is shared by all the units. Let $K_g^* = \min(N_{\mathrm{UC},g}, U^*)$. Then for unit g, we keep the smallest K_g^* uncertain positions in terms of the absolute value of two-group Bayes factor in the log scale.

D. Simulation details

In the simulation study we set 200 samples per group and 20000 units (genes). It is biologically plausible that a DE pattern can be shared by several units, so we assign blocks to units and units belong to the same block have same pattern. The blocks are generated from a Chinese restaurant process (CRP) with strength and discount parameter (α_0, α_1) (Aldous, Ibragimov, and Jacod 2006). Namely

$$\Pr(C_i = c \mid C_1, \cdots, C_{i-1}) = \begin{cases} \dfrac{\alpha_0 + B\alpha_1}{\theta + i - 1} & \text{if } c \in \text{new block}, \\[2ex] \dfrac{|b| - \alpha_1}{\alpha_0 + i - 1} & \text{if } c \in b; \end{cases} \tag{D.0.1}$$

where $C_1, \cdots, C_i$ are the cluster label for elements $1, \cdots, i$. B is the number of blocks, $|b|$ is the number of elements inside that block. We set $\alpha_1 = 1$ and use α_0 to control the number of blocks. After we obtain the blocking structure of genes, the next step is to generate partitions of groups and assign them to those blocks of genes. One challenge is that the size of possible partitions increases rapidly, for example when $K = 15$, there is 1.38 billion partitions, makes it difficult to sample directly. Again, we use CRP to generate partitions. The remaining problem is to assign partitions to blocks of genes. We observe that in the expression data, partitions π with big $p_{\mathrm{local},\pi}$ for most genes have small number of blocks and thus are coarse. Therefore, we use a monotone mapping, for example, big block of units will be associated with coarse partitions and small block of units will get fine partitions. We use entropy to measure the complexity(fineness) of a partition. That is, for a partition

π, we have the entropy: $-\Sigma_b \log(\frac{|b|}{\Sigma_b|b|})\frac{|b|}{\Sigma_b|b|}$, it reaches minimum when all samples belong to a single block and reaches maximum when every sample forms a distinct block. We assign the patterns with low entropy to big blocks. At each unit, counts are sampled from a Beta mixture of Negative Binomial (NB) distributions.

Affiliation:

Xiuyu Ma
Department of Statistics
University of Wisconsin Madison
1300 University Ave
53706 WI, USA
E-mail: ma79@wisc.edu

Christina Kendziorski
Department of Biostatistics and Medical Informatics
University of Wisconsin Madison
425g Henry Mall
53706 WI, USA
E-mail: kendzior@biostat.wisc.edu
URL: https://www.biostat.wisc.edu/~kendzior/

Michael A. Newton
Department of Biostatistics and Medical Informatics, and
Department of Statistics
University of Wisconsin Madison
1300 University Ave
53706 WI, USA
E-mail: newton@stat.wisc.edu
URL: http:www.stat.wisc.edu/~newton/