Supplementary Material for “DGEclust: differential expression analysis of clustered count data”

Dimitrios V Vavoulis*1 and Margherita Francescatto2 and Peter Heutink2 and Julian Gough*1

1Department of Computer Science, University of Bristol, Bristol, UK
2Genome Biology of Neurodegenerative Diseases, Deutsches Zentrum für Neurodegenerative Erkrankungen, Tübingen, Germany.

Email: Dimitris.Vavoulis@bristol.ac.uk; Margherita.Francescatto@dzne.de; Peter.Heutink@dzne.de; Julian.Gough@bristol.ac.uk;

*Corresponding author

Introduction

In this Supplementary Material, we provide the mathematical details of the algorithm we used to make posterior inferences on the generative model described by Eqs. 5. Currently, there are two major classes of methods for posterior inference in Hierarchical Dirichlet Process Mixture Models. The first class includes algorithms based on the Chinese Restaurant metaphor for representing the Dirichlet Process, as for example in [1, 2]. The second class includes variational inference approaches in either a batch or an online data processing context, as for example in [3]. The algorithm we present here is an extension of the blocked Gibbs sampler in [4] (which was developed for non–hierarchical Dirichlet Process mixture models) in the case of non–conjugate Hierarchical Dirichlet Process mixtures. Unlike variational methodologies, it approximates exact posteriors and, unlike approaches based on the Chinese Restaurant metaphor, it follows a blocked sampling strategy, which permits, in principle, faster convergence, and parallel implementation on multicore processors, which opens the possibility for applications on large gene expression datasets.

The first step in developing this algorithm is to cast the implicit generative model summarised in Eqs. 5 in an equivalent, explicit form, which makes use of Sethuraman’s constructive representation of a draw from
the Dirichlet Process [3, 5]:

\( \beta_k' \sim \text{Normal}(\mu_\beta, \sigma^2_\beta) \)

\( w_k \sim \text{Stick}(1, \delta), \quad k = 1, 2, \ldots \)

\( u_{rl} \sim \text{Stick}(1, \gamma_l), \quad r = 1, 2, \ldots \)

\( c_{rl} \sim \text{Categorical}(w) \)

\( d_{il} \sim \text{Categorical}(u) \)

\[ z_{il} \equiv c_{d_{il}} \]

\[ \beta_{il} \equiv \beta'_{z_{il}} \]

\[ \log(\phi_i) \sim \text{Normal}(\mu_\phi, \sigma^2_\phi) \]

\[ y_{ij}|\phi_i, \mu_i, \beta_{il} \sim \frac{\Gamma(y_{ij} + \alpha_i)}{\Gamma(\alpha_i)\Gamma(y_{ij} + 1)} \left( \frac{\alpha_i}{\alpha_i + m_{ij}} \right)^{\alpha_i} \left( \frac{m_{ij}}{\alpha_i + m_{ij}} \right)^{y_{ij}} \] (1)

where \( l \equiv \lambda(j), \alpha_i \equiv \phi_i^{-1} \) and \( \log(m_{ij}) = \log(c_j) + \log(\mu_i) + \beta_{il} \). This representation introduces matrices of group– and gene–specific indicator variables, \( c = \{c_{rl}\} \) and \( d = \{d_{il}\} \), with elements sampled from the categorical distribution with weights \( w = \{w_k\} \) (a vector) and \( u = \{u_{rl}\} \) (a matrix), respectively. What makes the above model special is how we generate the weights \( w \) and \( u \), which resembles a stick–breaking process; for \( w \), we set \( w_1 = V_1 \) and \( w_k = V_k \prod_{k=1}^{k-1} (1 - V_{k'}) \) for \( k \geq 2 \), where \( V_1, \ldots, V_k \) are random variables following a Beta distribution, \( V_k \sim \text{Beta}(1, \delta) \). For the sake of brevity, we write \( w_k \sim \text{Stick}(1, \delta) \). Similarly, \( u_{rl} \sim \text{Stick}(1, \gamma_l) \), for Beta–distributed variables sampled as \( V_{rl} \sim \text{Beta}(1, \gamma_l) \).

We wish to compute the joint posterior of the various random variables that appear in the model, \( p(\phi, \mu, \beta', c, d, w, u, \delta, \gamma, \mu_\beta, \sigma^2_\beta, \mu_\phi, \sigma^2_\phi|y) \), where \( y \) is a matrix of count data; \( \phi \) and \( \mu \) are vectors of gene–specific dispersion and mean expression parameters, respectively; \( \beta' \) is a vector of log–fold changes; \( \delta \) and \( \gamma \) are concentration parameters (a scalar and a vector, respectively); and \( \mu_\beta, \mu_\phi, \sigma^2_\beta \) and \( \sigma^2_\phi \) are scalar hyper–parameters. We approximate the above distribution through numerical (Monte Carlo) methods, i.e. by sampling a large number of \( \{\phi, \mu, \beta', c, d, w, u, \delta, \gamma, \mu_\beta, \sigma^2_\beta, \mu_\phi, \sigma^2_\phi\} \)–tuples from it. One way to achieve this is by constructing a Markov chain, which admits \( p(\phi, \mu, \beta', c, d, w, u, \delta, \gamma, \mu_\beta, \sigma^2_\beta, \mu_\phi, \sigma^2_\phi|y) \), as its stationary distribution. Such a Markov chain can be constructed by using Gibbs sampling, which consists of sampling repeatedly from the conditional posteriors for each of these random variables, given the most recent values of all other variables that appear in each conditional. Below, we explain how to sample from each of these conditionals. Notice that, with a slight abuse of notation, we use the symbol \( p \) to indicate either a probability density or a probability mass function, depending on the context. Similarly, conditionals are often indicated.
as $p(x|y, \cdot)$, where the variables which $x$ is conditioned on (other than the data) are summarised by a dot ($\cdot$) for simplicity. Finally, for a matrix $x$ with elements $x_{ij}$, we indicate the $i$-th row and the $j$-th column as $x_i$ and $x_j$, respectively.

**Updates for the vector of dispersion parameters, $\phi$**

Each element $\phi_i$ of the vector $\phi$ can be sampled independently from the others. If we let $y_i$ and $\beta_i$ be the vector of count data for gene $i$ across all samples and the vector of group–specific log–fold changes for gene $i$, respectively, we can write:

$$p(\phi_i|y_i, \mu_i, \beta_i) \propto p(y_i|\phi_i, \mu_i, \beta_i)p(\phi_i|\mu_\phi, \sigma_\phi^2)$$

$$\propto \prod_{j=1}^M p(y_{ij}|\phi_i, \mu_i, \beta_{\lambda(j)})p(\phi_i|\mu_\phi, \sigma_\phi^2)$$  \hspace{1cm} (2)

where the likelihood and the prior are a negative binomial and a log–normal distribution, respectively, as indicated in Eqs. 1. Since the posterior of $\phi_i$ cannot be sampled from directly, we employ a Metropolis step with proposals $\phi_i^*$ sampled from the prior of $\phi_i$ and acceptance probability equal to:

$$p_{acc} = \min \left( 1, \frac{\prod_{j=1}^M p(y_{ij}|\phi_i^*, \mu_i, \beta_{\lambda(j)})}{\prod_{j=1}^M p(y_{ij}|\phi_i, \mu_i, \beta_{\lambda(j)})} \right)$$  \hspace{1cm} (3)

Notice that, since we use the prior for sampling new proposals, the acceptance probability merely involves the ratio of two likelihoods.

**Updates for the vector of mean expression levels, $\mu$**

As in the case of $\phi_i$, the gene–specific expression levels $\mu_i$ can be sampled independently to each other. We observe that instead of working directly with $\mu_i$, it is easier to work with the proxy probability variable $p_i = \alpha_i/(\alpha_i + \mu_i)$, where $\alpha_i \equiv \phi_i^{-1}$. Assuming a non–informative Beta prior for $p_i$, $p(p_i) = \text{Beta}(0.5, 0.5)$, we can write:

$$p(p_i|\tilde{y}_i, \phi_i) \propto p(\tilde{y}_i|\phi_i, p_i)p(p_i)$$

$$\propto \prod_{j=1}^M p(\tilde{y}_{ij}|\phi_i, p_i)p(p_i)$$  \hspace{1cm} (4)

where $\tilde{y}_{ij} = y_{ij}/(c_j\beta_{\lambda(j)})$ and $p(\tilde{y}_{ij}|p_i, \phi_i) \propto p_i^{\alpha_i}(1 - p_i)^{\beta_{\lambda(j)}}$. It follows that the posterior of $p_i$ is a Beta distribution:

$$p(p_i|\tilde{y}_i, \phi_i) = \text{Beta}(0.5 + c_1, 0.5 + c_2)$$  \hspace{1cm} (5)
where \( c_1 = \sum_{j=1}^{M} \tilde{y}_{ij} \), \( c_2 = M/\phi_i \) and \( M \) is the number of libraries. A sample from the posterior of \( \mu_i \) is then obtained by a simple transformation of the newly sampled \( p_i \):

\[
\mu_i = \frac{1 - p_i}{\phi_ip_i}
\]  

(6)

**Updates for the vector of weights, \( w \)**

In Eq. 1, the indices \( k \) for the sequence of weights \( w = \{w_k\} \) go from 1 to \( \infty \). In practice, we truncate this infinite series by rejecting all terms with index larger than \( K \) and setting \( w_K = 1 - \sum_{k=1}^{K-1} w_k \), which is equivalent to setting \( V_K = 1 \). It is known that for sufficiently large values of \( K \), the above truncation is virtually indistinguishable from the full (infinite) sequence [4].

The first step in updating the truncated sequence of weights \( w \) is generating \( K-1 \) Beta–distributed random numbers, as follows:

\[
V_k \sim \text{Beta} \left( 1 + N_k, \delta + NL - \sum_{k'=1}^{k} N_{k'} \right), \quad k = 1, \ldots, K-1
\]  

(7)

where \( N \) is the number of genes, \( L \) is the number of library groups and \( N_k \) is the number of elements of \( z \) that equal \( k \). Notice that \( z \) is a matrix with elements \( z_{il} = c_{d_{il}} \). Also, \( V_K = 1 \) in order to ensure that the weights add up to 1. These are then simply generated by setting \( w_1 = V_1 \) and \( w_k = V_k \prod_{k'=1}^{k-1} (1 - V_{k'}) \) for \( k \geq 2 \), as mentioned in a previous section.

**Updates for the matrix of weights, \( u \)**

Similarly to the case of \( w \), the matrix of gene– and group–specific weights \( u = \{u_{rl}\} \) is truncated by rejecting all elements for which \( r \) is larger than \( R \). For each group of libraries \( l \), the vector of weights \( u_{.l} \) can be updated as described above for \( w \). We begin by sampling \( K-1 \) Beta–distributed random numbers:

\[
V_{rl} \sim \text{Beta} \left( 1 + N_{rl}, \gamma_l + N - \sum_{r'=1}^{r} N_{r'l} \right), \quad r = 1, \ldots, R-1
\]  

(8)

where \( N_{rl} \) is the number of elements of the vector of indicators \( d_{.l} \) equal to \( r \), \( N \) is the number of genes and \( V_R = 1 \). The weights \( u_{.l} \) are then generated by a stick–breaking process, as described above for \( w \).

**Updates for the matrix of indicator variables, \( d \)**

Each gene– and group–specific element \( d_{il} \) in the matrix of indicator variables \( d \) takes values between 1 and \( R \) and it can be updated independently to the others. The posterior probability that \( d_{il} \) is equal to \( r \) for
\( r = 1, \ldots, R \) is:

\[
p(d_{il} = r | y^l_i, \phi, \mu_i, \beta', c, \mu, \beta'_{c, r}) \propto \prod_{j=1}^{M_l} p(y^l_{ij} | \phi, \mu_i, \beta'_{c, r}) u_{rl}
\]

where \( y^l_i \) is the set of expression measurements for gene \( i \) in the group of libraries \( l \), \( u_{rl} \) is the prior probability that \( d_{il} = r \) and \( M_l \) is the number of libraries in group \( l \). A straightforward way to sample from the posterior of \( d_{il} \) is to compute the posterior probabilities \( \pi_{ilr} = p(d_{il} = r | y^l_i, \cdot, \cdot) \) for all possible values of \( r \) and then sample \( d_{il} \) from a Categorical distribution with weights \( \tilde{\pi}_{il1}, \ldots, \tilde{\pi}_{ilR} \), where \( \tilde{\pi}_{ilr} = \pi_{ilr} / \sum_{r'=1}^{R} \pi_{ilr} \).

**Updates for the matrix of indicator variables, \( c \)**

The elements \( c_{rl} \) of matrix \( c \) take values from 1 to \( K \) and they are sampled independently to each other, similarly to \( u \). We start by writing the posterior probability that \( c_{rl} = k \):

\[
p(c_{rl} = k | y^l, \phi, \mu, \beta', \cdot, \cdot, \cdot) \propto N \prod_{i=1}^{N} \prod_{j=1}^{M} p(y_{ij} | \phi, \mu_i, \beta'_{k}) | z_{i\lambda(j)} = k | w_k
\]

where \( y^l \) is the \( N \times M_l \) matrix of counts corresponding to group of libraries \( l \), \( w_k \) is the prior probability that \( c_{rl} = k \) and \( [\cdot] \) is equal to 1 if the expression inside the square brackets is true and 0 otherwise. A sample from the posterior of \( c_{rl} \) is obtained by computing the posterior probabilities for all possible values of \( k \), normalising and sampling from a Categorical distribution, similarly to the case of \( d_{il} \) above.

**Updates for the vector of log–fold changes, \( \beta' \)**

The posterior density of each element \( \beta'_k \) of \( \beta' \) is as follows:

\[
p(\beta'_k | y, \phi, \mu, \cdot, \cdot) \propto \prod_{i=1}^{N} \prod_{j=1}^{M} p(y_{ij} | \phi, \mu_i, \beta'_k) | z_{i\lambda(j)} = k | \beta'_k | \mu_{\beta}, \sigma^2_{\beta}
\]

where the prior \( p(\beta'_k | \cdot) \) is normal and \( z_{il} = c_{d_{il}} \) as indicated in Eqs. 1. In order to sample from the above posterior, we employ a Metropolis step with proposals \( \beta'_k^* \) sampled from the prior and acceptance probability:

\[
p_{acc} = \min \left( 1, \frac{\prod_{i=1}^{N} \prod_{j=1}^{M} p(y_{ij} | \phi, \mu_i, \beta'_k^*) | z_{i\lambda(j)} = k | \beta'_k | \mu_{\beta}, \sigma^2_{\beta}}{\prod_{i=1}^{N} \prod_{j=1}^{M} p(y_{ij} | \phi, \mu_i, \beta'_k) | z_{i\lambda(j)} = k | \beta'_k | \mu_{\beta}, \sigma^2_{\beta}} \right)
\]

Notice that for those \( k \) for which the condition \( z_{i\lambda(j)} = k \) is never true, the corresponding \( \beta'_k \) is simply sampled from the prior of \( \beta' \).

**Updates for the concentration parameters \( \delta \) and \( \gamma \)**

The concentration parameter \( \delta \) and the vector of concentration parameters \( \gamma \) influence the expected number of clusters of the corresponding Dirichlet processes. Sampling from the posterior of the concentration parameter
of the Dirichlet process has been covered previously in [6] and in [4], which is the approach we follow here.

Assuming a Gamma prior with shape \( a \) and rate \( b \), the posterior of \( \delta \) depends only on the vector of weights \( w \):

\[
p(\delta|w) = \text{Gamma}(a + K - 1, b - \log(w_K))
\]  

(13)

Similarly, for each element \( \gamma_l \) of \( \gamma \), the posterior becomes:

\[
p(\gamma_l|u_l) = \text{Gamma}(a + R - 1, b - \log(u_{R_l}))
\]  

(14)

Both \( \delta \) and the elements of \( \gamma \) are sampled independently to each other.

**Updates for the hyper–parameters \( \mu_\beta, \sigma^2_\beta, \mu_\phi \) and \( \sigma^2_\phi \)**

Assuming non–informative priors, the marginal posteriors of \( \mu_\phi \) and \( \sigma^2_\phi \) are normal and inverse gamma, respectively [7]:

\[
p(\mu_\phi|\sigma^2_\phi) = \text{Normal}\left(\bar{m}_\phi, \frac{\sigma^2_\phi}{N}\right)
\]  

\[
p(\sigma^2_\phi) = \text{InvGamma}\left(\frac{N}{2}, \frac{s^2_\phi}{2}\right)
\]  

(15)

where \( \bar{m}_\phi = \sum_{i=1}^{N} \log(\phi_i)/N \), \( s^2_\phi = \sum_{i=1}^{N} (\log(\phi_i) - \bar{m}_\phi)^2 \) and \( N \) is the number of genes.

Similarly, the marginal posteriors for \( \mu_\beta \) and \( \sigma^2_\beta \) are:

\[
p(\mu_\beta|\beta', \sigma^2_\beta) = \text{Normal}\left(\bar{m}_\beta, \frac{\sigma^2_\beta}{K'}\right)
\]  

\[
p(\sigma^2_\beta|\beta') = \text{InvGamma}\left(\frac{K'}{2}, \frac{s^2_\beta}{2}\right)
\]  

(16)

where \( \bar{m}_\beta = \sum_{k=1}^{K'} \beta'_k/K' \) and \( s^2_\beta = \sum_{k=1}^{K'} (\beta'_k - \bar{m}_\beta)^2 \). Notice that only the \( \beta'_k \) for which the condition \( z_{i\lambda(j)} = k \) is true for at least one pair of \( i \) and \( j \) participate in the above computation. \( K' \) is the number of these \( \beta'_k \).

**Summary**

Having determined the methodology for sampling from the various conditional posterior distributions, we can now summarise the algorithm for sampling from the full joint posterior \( p(\phi, \mu, \beta', c, d, w, u, \delta, \gamma, \mu_\beta, \sigma^2_\beta, \mu_\phi, \sigma^2_\phi|y) \). Starting with a set of initial values for all random variables in the model, we repeat the following steps:
1. Sample $\phi_i$ for all $i$ independently given the most recent values of all other variables

2. Sample $\mu_i$ for all $i$ independently given the most recent values of all other variables

3. Sample $\beta'_k$ for all $k$ independently given the most recent values of all other variables

4. Sample $c_{rl}$ for all $r$ and $l$ independently given the most recent values of all other variables

5. Sample $d_{il}$ for all $i$ and $l$ independently given the most recent values of all other variables

6. Sample $\delta$ and $\gamma_l$ for all $l$ independently given the most recent values of all other variables

7. Sample $\sigma^2_\phi$ and $\sigma^2_\beta$ independently given the most recent values of all other variables

8. Sample $\mu_\phi$ and $\mu_\beta$ independently given the most recent values of all other variables

The initial values of the various variables that appear in the model do not seem to affect the final equilibrium state, but they might influence the speed of convergence. Truncation levels were $K = R = 100$. 

References