A method for cell-type deconvolution based on a sequential Monte Carlo framework

Kyle Coleman

9 May 2021

Reproducibility

Code chunk (1) can be run to estimate the proportions of brain and heart cells in heterogeneous samples generated according to the proportions in M^*, defined in the results section of this paper.

(1) M^*, no parallelization:

```
library(smc.deconvolution)
X = smc.deconvolution::brain_heart_spec_ge
X = smc.deconvolution::hvg_selection(X)
M = smc.deconvolution::proportions
Y = smc.deconvolution::create_mixture_samples(X,M)
set.seed(10)
est_props = smc.deconvolution::smc_deconvolution(Y=Y,K=2,L=10)
M_est = smc.deconvolution::combine_est_props(est_props[[1]],M)
smc.deconvolution::simulation_summary(est_props[[2]]) #provides simulation run time
smc.deconvolution::jsd_mse(M,M_est) #calculates JSD and MSE
```

Code chunk (2) performs the same task as (1), but parallelizes over 8 cores and plots Figure 1.

(2) M^*, parallelization:

```
Y = smc.deconvolution::create_mixture_samples(X,M)
set.seed(10)
est_props = smc.deconvolution::smc_deconvolution(Y=Y,K=2,L=10,parallel = T,num_cores = 8)
M_est = smc.deconvolution::combine_est_props(est_props[[1]],M)
est_props[[2]] #provides simulation run time
smc.deconvolution::scatter_plot(M,M_est) #plot shown in Figure 1
```

Code chunks (3-5) perform SMC deconvolution on a set of 10, 15, and 20 heterogeneous samples, respectively, generated according to proportions drawn from a Dirichlet distribution.

(3) Dirichlet, 10 samples

```
M = prop_rdirichlet(10,10,2,colnames(X))
Y = smc.deconvolution::create_mixture_samples(X,M)
set.seed(10)
est_props = smc.deconvolution::smc_deconvolution(Y=Y,K=2,L=10,parallel = T,num_cores = 8)
M_est10 = smc.deconvolution::combine_est_props(est_props[[1]],M)
```

```
run_time10 = est_props[[2]] #provides simulation run time
mad10 = smc.deconvolution::mad(M,M_est10)
```

(4) Dirichlet, 15 samples

```
M = prop_rdirichlet(seed=10,n =15,K=2,types=colnames(X))
Y = smc.deconvolution::create_mixture_samples(X,M)
set.seed(10)
est_props = smc.deconvolution::smc_deconvolution(Y=Y,K=2,L=10,parallel = T,num_cores = 8)
M_est15 = smc.deconvolution::combine_est_props(est_props[[1]],M)
run_time15 = est_props[[2]] #provides simulation run time
mad15 = smc.deconvolution::mad(M,M_est15)
```

(5) Dirichlet, 20 samples

```
M = prop_rdirichlet(seed=10,n=20,K=2,types=colnames(X))
Y = smc.deconvolution::create_mixture_samples(X,M)
set.seed(10)
est_props = smc.deconvolution::smc_deconvolution(Y=Y,K=2,L=10,parallel = T,num_cores = 8)
M_est20 = smc.deconvolution::combine_est_props(est_props[[1]],M)
run_time20 = est_props[[2]]
mad20 = smc.deconvolution::mad(M,M_est20)
```

(7) MAD plot for samples generated from Dirichlet distributions, Figure 3

```
mad_plot(c(mad10,mad15,mad20), c(10,15,20))
```

We have uploaded our SMC-Deconvolution R package to GitHub, and it passes the travis-ci check:

https://github.com/kpcoleman/SMC-Deconvolution

Introduction

With technological advances in the field of genetics has come a vast increase in the amount of available gene expression (GE) data. These data have many applications, including in the study of the genetc basis of diseases. Though there is an abundance of GE data, downstream analysis may be inhibited by the lack of single-cell resolution in many of these datasets. Specifically, samples in GE datasets without single-cell resolution can contain multiple cells, potentially of different cell types. A lack of knowledge of the cell-type compositions of the samples in such datasets can cause limitations for analysis and application of these data. As a result, many cell-type deconvolution algorithmns have been developed to estimate the proportions of cells of given types across samples in heterogeneous GE datasets that lack single-cell resolution. One such method, developed by Ogundijo and Wang (2017), utilizes a sequential Monte Carlo (SMC) approach to estimate cell-type specific gene expression levels and cell-type proportions in heterogeneous GE datasets. The intention of this project was to replicate some of the simulations performed by these authors, as well as extend their investigation by considering multiple datasets with samples composed of different proportions of cell types and including more evaluation metrics. We also incorporated a highly variable gene selection step to increase the computational efficiency of the algorithm.

Methods

Highly Variable Gene Selection

The SMC deconvolution algorithm is used to estimate the proportions of cell types across heterogeneous samples in a GE matrix. This method requires as input a GE matrix of heterogeneous samples, Y, as well as the total number of cell types, K, in the samples of Y. The algorithm then provides the estimated proportion of the K cell types in each of the samples of Y. In some cases, a matrix of cell-type specific gene expression profiles, X, may be available. For such situations, we included a function in our package to, prior to deconvolution, reduce the dimensionality of Y by only selecting counts for the top $(100 * p)\%$ highly variable genes (HVGs) in X, where p is set to 0.05 by default. HVGs are genes that show the most variation in expression levels across different cell types, and are therefore likely more informative for analyses investigating cell-type composition compared to genes that show less variability across cell types (Yip, Sham, and Wang 2019). HVG selection is commonly used for the clustering of cells in single-cell RNA-sequencing (scRNA-seq) data (Andrews and Hemberg 2018), and has been incorporated as a step prior to performing cell-type deconvolution (Elosua-Bayes et al. 2021). Selection of HVGs both greatly reduces computation time and has been shown to lead to accurate results when compared with including all genes for a given method.

Distributions of Model Parameters

After HVG selection is performed, we run the SMC deconvolution algorithm. This algorithm treats the proportions of each cell type in the samples in Y, $M = [\underline{m_1} \cdots \underline{x_J}]$ as unknown parameters to be estimated by the model. Here, $\underline{m_j} = (m_{1j} \cdots x_{Kj})^T$ are the proportions of cell-types $k = 1, ..., K$ that compose sample j in Y. The counts in the cell-type specific gene expression matrix $X = [\underline{x_1} \cdots \underline{x_K}]$, are also treated as unknown parameters, where $\underline{x_k} = (x_{1k} \cdots x_{Ik})^T$ contains the cell-type specific counts for genes $i = 1, ..., I$ for cell type k. There is also an unknown precision parameter, $\lambda = \text{Var}^{-1}(y_{ij})$, resulting in a total of $(J * K + I * K + 1)$ model parameters. Let θ denote the set of all model parameters. Given θ, the distribution of the count of gene i in sample j of Y is:

$$p(y_{ij}|\theta) = \phi\Big(y_{ij}\Big|\sum_{k=1}^{K} x_{ik}m_{kj}, \lambda^{-1}\Big),$$

where $\phi(\cdot|\mu, \sigma^2)$ is the pdf of a $N(\mu, \sigma^2)$ distribution. Thus, the joint distribution of all gene counts in Y is:

$$\begin{aligned} p(Y|\theta) &= \prod_{i=1}^{I}\prod_{j=1}^{J} \phi\Big(y_{ij}\Big|\sum_{k=1}^{K} x_{ik}m_{kj}, \lambda^{-1}\Big) \\ &\propto \lambda^{IJ/2} \exp\Big\{-\frac{\lambda}{2}\sum_{i=1}^{I}\sum_{j=1}^{J}\Big(y_{ij} - \sum_{k=1}^{K} x_{ik}m_{kj}\Big)^2\Big\} \end{aligned}$$

The prior distribution for the cell-type k specific count of gene i, x_{ik} is given by:

$$\pi(x_{ik}) = \phi(x_{ik}|\mu_{ik}, \nu_{ik}^{-1}),$$

where μ_{ik} and ν_{ik}^{-1} are hyperparameters that are assumed known. By default, ν_{ik}^{-1} is set to 1000 and μ_{ik} is sampled from a Unif$(0, 100)$ distribution, $i = 1, ..., I,\ k = 1, ..., K$. The prior distribution for the proportion of cell type k in sample j, m_{kj}, is:

$$\pi(m_{kj}) = \phi(m_{kj}|\mu_{kj}, \nu_{kj}^{-1}),$$

where μ_{kj} and ν_{kj}^{-1} are assumed known and set to 0 and 0.01, respectively. The prior distribution for λ is:

$$\pi(\lambda) \sim \text{Gamma}(\alpha, \beta),$$

where α and β are assumed known and set to 2 and 10000, respectively.

The SMC algorithm defines, at time t, the intermediate posterior distribution for the model parameters as:

$$p_t(\theta|Y) = \pi(\theta)p(Y|\theta)^{\epsilon_t},$$

where $N = 40$ samples are taken by default and

$$\{\epsilon_t\}_{t=1}^{T} = \{0, \epsilon_2, ..., \epsilon_{T-1}, 1\}.$$

For the algorithm we implemented,

$$\epsilon_t = \epsilon_{t-1} + 10^{-4},\ t = 2, ..., T \implies T = 1001.$$

.

As opposed to other Monte Carlo-based methods, defining the posterior distribution in this way gradually increases the influence of the distribution $p(Y|\theta)$. Then, for time t, a joint distribution for the model parameters at times $b = 1, ..., t$, $\theta_{1:t} = \{\theta_1, ..., \theta_t\}$, is defined by:

$$\tilde{p}_t(\theta_{1:t}) \propto \pi(\theta_t)p(Y|\theta_t)^{\epsilon_t} \prod_{b=1}^{t-1} \mathcal{L}_b(\theta_{b+1}, \theta_b),$$

where $\mathcal{L}_b(\theta_{b+1}, \theta_b)$ is a backwards Markov kernel. The goal of the SMC algorithm is to obtain N samples from $\tilde{p}_t(\theta_{1:t})$ for each $t = 1, ..., T$. However, the difficulty associated with sampling from this distribution leads to the construction of an importance distribution that is more suitable. At time t, this distribution is defined by:

$$q_t(\theta_{1:t}) = q_1(\theta_1) \prod_{f=2}^{t} \mathcal{K}_f(\theta_{f-1}, \theta_f),$$

where $\mathcal{K}_f(\theta_{f-1}, \theta_f)$ is a forwards Markov kernel.

A set of importance weights are calculated at each time t to account for sampling the parameters from the importance distribution q_t as opposed to the posterior distribution p_t. For sample n at time $t - 1$, the unnormalized importance weight, $\tilde{w}_{t-1}^{n}$ is calculated as:

$$\tilde{w}_{t-1}^n \propto \frac{\tilde{p}_t(\theta_{1:t}^n)}{q_t(\theta_{1:t}^n)} = \frac{p_{t-1}(\theta_{t-1}^n|Y)\prod_{b=1}^{t-2}\mathcal{L}_b(\theta_{b+1}^n,\theta_b^n)}{q_1(\theta_1^n)\prod_{f=2}^{t-1}\mathcal{K}_f(\theta_{f-1}^n,\theta_f^n)}$$

These weights are then normalized to obtain the importance weights at time $(t-1)$:

$$w_{t-1}^n = \frac{\tilde{w}_{t-1}^n}{\sum_{n'=1}^N \tilde{w}_{t-1}^{n'}}.$$

The relationship between the unnormalized weights for sample n at times $t-1$ and t can be shown to be:

$$\tilde{w}_t^n \propto \tilde{w}_{t-1}^n \frac{p_t(\theta_t^n|Y)\mathcal{L}_{t-1}(\theta_t^n,\theta_{t-1}^n)}{p_{t-1}(\theta_{t-1}^n|Y)\mathcal{K}_t(\theta_{t-1}^n,\theta_t^n)}.$$

Each time the normalized weights are computed, we calculate the effective sample size (ESS) by:

$$ESS = \frac{1}{\sum_{n=1}^N (w_t^n)^2}.$$

A small ESS implies that some of the samples have a relatively large weight compared to others. To remedy this, a resampling is performed at any time t such that $ESS < \frac{N}{10}$. Specifically, N samples are drawn with replacement according to the weights $w_t^1, ..., w_t^N$ and the weights are set to $1/N$. In order sample the unknown parameters from $q_t(\theta_{1:t})$, the posterior distribution for the parameters at time t, $p_t(\theta|Y)$ is required. Samples from $p_t(\theta|Y)$ are obtained using a Gibbs sampler, i.e. by sampling from the conditional posterior distributions of the model parameters (Gelman et al. 2013). Specifically, at time t, the posterior distribution of m_{kj} conditioned on the other parameters, is given by:

$$p_t(m_{kj}|\cdot) = \phi\Big\{m_{kj}\Big|\frac{\mu_{kj}\nu_{kj} + \epsilon_t\lambda\sum_{i=1}^I\left(y_{ij}x_{ik} - (\sum_{k'\neq k}x_{ik'}m_{k'j})x_{ik}\right)}{\nu_{kj} + \epsilon_t\lambda\sum_{i=1}^I x_{ik}^2}, \frac{1}{\nu_{kj} + \epsilon_t\lambda\sum_{i=1}^I x_{ik}^2}\Big\},$$

where "$\cdot$" indicates Y and the most recent updates of the other parameters. The conditional posterior distribution of x_{ik} at time t is given by:

$$p_t(x_{ik}|\cdot) = \phi\Big\{x_{ik}\Big|\frac{\mu_{ik}\nu_{ik} + \epsilon_t\lambda\sum_{j=1}^J\left(y_{ij}m_{kj} - (\sum_{k'\neq k}x_{ik'}m_{k'j})m_{kj}\right)}{\nu_{ik} + \epsilon_t\lambda\sum_{j=1}^J m_{kj}^2}, \frac{1}{\nu_{ik} + \epsilon_t\lambda\sum_{j=1}^J m_{kj}^2}\Big\},$$

and λ is sampled according the conditional posterior distribution:

$$p_t(\lambda|\cdot) \sim \text{Gamma}\Big\{\alpha + \frac{\epsilon_t IJ}{2}, \beta + \frac{\epsilon_t}{2}\sum_{i=1}^I\sum_{j=1}^J\Big(y_{ij} - \sum_{k=1}^K x_{ik}m_{kj}\Big)^2\Big\}.$$

In addition to the posterior distribution, the Markov kernels $\mathcal{K}_t$ and $\mathcal{L}_{t-1}$ need to be specified. Following Nguyen *et al.* (2016), the kernel $\mathcal{K}_t$ is chosen to be:

$$\mathcal{K}_t(\theta_{t-1}, \theta_t) = p_t(\theta_t),$$

and the kernel $\mathcal{L}_{t-1}$ is:

$$\mathcal{L}_{t-1}(\theta_t, \theta_{t-1}) = \frac{p_t(\theta_{t-1})\mathcal{K}_t(\theta_t, \theta_{t-1})}{p_t(\theta_t)}.$$

Defining the forward and backward Markov kernels in this way results in the following form of the unnormalized weight at time t for sample n:

$$\tilde{w}_t^n \propto \tilde{w}_{t-1}^n p(Y|\theta_{t-1}^n)^{\epsilon_t - \epsilon_{t-1}}$$

Steps of SMC algorithm

For time $t = 1$ of the SMC algorithm, N samples for the unknown parameters λ, $\{m_{kj}\}$, and $\{x_{ik}\}$ from their respective prior distributions defined above, and the weights $w_1^1, ..., w_1^N$ are set to $\frac{1}{N}$. Then, for times $t = 2, ..., T$, the following steps are peformed:

(1) The log-likelihood $\log(p(Y|\theta_{t-1}^n)$ is calculated and exponentiated to obtain the likelihood $p(Y|\theta_{t-1}^n$.

(2) The unnormalized weights $\tilde{w}_t^n$ are calculated as:

$$\tilde{w}_t^n = w_{t-1}^n p(Y|\theta_{t-1}^n)^{\epsilon_t - \epsilon_{t-1}},$$

and then normalized to sum to 1.

(3) The ESS is calculated and resampling occurs if necessary.

(4) Sample m_{kj} N times from $p_t(m_{kj}|\cdot)$, $k = 1, ..., K$, $j = 1, ..., J$.

(5) Sample x_{ik} N times from $p_t(x_{ik}|\cdot)$, $i = 1, ..., I$, $k = 1, ..., K$.

After T iterations, the estimated cell-type proportions across the samples in Y are calculated as:

$$\hat{m}_{kj} = \sum_{n=1}^{N} w_T^n (m_{kj}^n)_T, \qquad k = 1, ..., K, j = 1, ..., J.$$

Results

For this project, we analyzed the Affymetrix oligonucleotide microarray dataset containing the GE levels of samples consisting of brain and heart cells. We used the samples in this dataset to construct a matrix of cell-type specific GE profiles. We then identified the top 5% HVG's in this matrix to construct X^*. The matrix of cell-type proportions specified by the authors contains 3 of each of the following compositions of brain and heart cells:

Brain	0.05	0.10	0.25	0.50	0.75	0.90	0.95
Heart	0.95	0.90	0.75	0.50	0.25	0.10	0.05

We then performed matrix multiplication between X^* and M^* to construct a GE matrix consisting of 27 heterogeneous samples, Y^*.

We ran the SMC deconvolution algorithm using Y^* as input for 10 iterations. The SMC algorithm is unsupervised, and so does not rely on an annotated reference dataset. As a result, this method does not provide the name of the cell type associated with a given proportion estimate. Denote the cell types in the matrix of proportion estimates as $k = 1, 2$. To evaluate model performance, we associate type k with the cell type that that results in the minimum mean absolute deviation (MAD) between the estimated and true proportions across all samples. We then calculated the mean of the cell-type proportion estimates across the 10 iterations for each sample. The correlation between the true and estimated proportions is approximately 1, and a scatter plot of the true and estimated proportions for each cell type in each sample is shown in Figure 1. The MAD between the true and estimated proportions is 0.035. We ran the 10 iterations of SMC deconvolution algorithm twice, once using only 1 core, and once parallelizing over 8 cores. When not running the iterations in parallel, the mean run time across the iterations was 20.0908452 minutes (sd = 0.3825698 minutes), resulting in a total of approximately 201 minutes. When parallelizing over 8 cores, the total run time of the 10 iterations was 102.2025 minutes, resulting in a 49% decrease in run time.

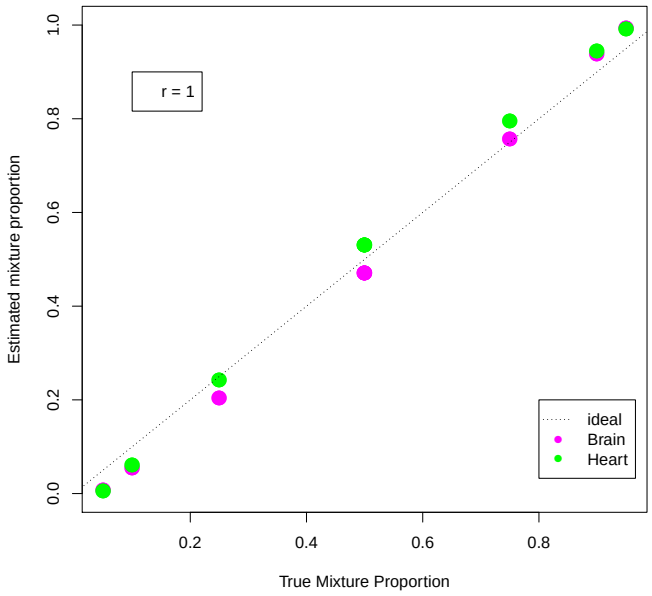

Figure 1: Scatter plot of true versus estimated cell-type proportions (M*)

We then constructed a new matrix of proportions of brain and heart cells across 10 samples, where the proportions in each sample were generated using a Dirichlet distribution. The GE matrix of heterogeneous samples, Y, was constructed by multiplying X^* with this matrix of cell-type proportions. We ran the SMC-deconvolution for 10 iterations to obtain the proportion estimates of each cell-type across the 10 samples in Y. The correlation between the true and estimated proportions is approximately 0.52, much lower than that calculated when using M^* as the matrix

of cell-type proportions. The MAD between the true and estimated proportions is 0.21. When using the proportions generated from a Dirichlet distribution, all proportion estimates are close to 0.5. We ran these iterations in parallel across 8 cores, and the run time was 79.9 minutes.

We constructed two more matrices of cell-type proportions using a Dirichlet distribution, one with 15 samples and one with 20 samples. For each of these matrices, we again constructed a GE matrix of heterogeneous samples and estimated the cell-type proportions by running the the SMC-deconvolution for 10 iterations. The MAD between the true and estimated proportions for the matrices with 15 and 20 samples are 0.21 and 0.24. A plot showing the MAD between the true and estimated proportions when constructing heterogeneous samples according to a Dirichlet distribution for different sample sizes is shown in Figure 3. When parallelizing over 8 cores, the run time for the set with 15 samples was 88.73 minutes, and that for 20 samples was 101.10 minutes.

For all simulations discussed, we calculated the mean squared error (MSE) and Jensen-Shannon divergence (JSD) between the true and estimated proportions for each cell type across samples, where JSD is a distance metric based on the Kullback-Leibler (KL) divergence that measures the difference between two probability distributions. MSE and JSD are commonly used to evaluate cell-type deconvolution performance, and lower values of these metrics indicate a greater similarity between the true and estimated proportions. These evaluation metrics for all simulations are shown below:

	M*, 27		**Dirichlet, 10**		**Dirichlet, 15**		**Dirichlet, 20**	
	JSD	MSE	JSD	MSE	JSD	MSE	JSD	MSE
Brain	0.061	0.0013	0.36	0.064	0.51	0.068	0.72	0.078
Heart	0.061	0.0013	0.21	0.064	0.43	0.068	0.70	0.078

Conclusion

Cell-type deconvolution is an important step when analyzing GE data consisting of samples of a mixture of different cell types. Here we presented a sequential Monte Carlo based method for cell-type deconvolution, proposed by Ogundijo and Wang. We incorporated a highly variable gene selection step, that can be used to reduced the dimensionality of the GE matrix of heterogeneous samples when a cell-type specific gene expression matrix is available. We evaluated the SMC deconvolution algorithm on multiple sets of mixture samples using brain and heart specific gene expression profiles and specified proportions of these cell types. The algorithm was shown to provide accurate results when a specific matrix of cell-type proportions was used, but the results were not accurate when the heterogeneous samples were genetated according to proportions sampled from a Dirichlet distribution. The computation time of the simulations decreased greatly when parallelizing over multiple cores.

Bibliography

Andrews, T S, and M Hemberg. 2018. "Identifying Cell Populations with scRNASeq." *Molecular Aspects of Medicine* 59: 114–22.

Elosua-Bayes, M, P Nieto, E Mereu, I Gut, and H Heyn. 2021. "SPOTlight: Seeded Nmf Regression to Deconvolute Spatial Transcriptomics Spots with Single-Cell Transcriptomes." *Nucleic Acids Research*.

Gelman, A, J Carlin, H Stern, D B Dunson, A Vehtari, and D B Rubin. 2013. *Bayesian Data Analysis, Third Edition*. London: Chapman & Hall.

Nguyen, T LT, F Septier, G W Peters, and Y Delignon. 2016. "Efficient Sequential Monte-Carlo Samplers for Bayesian Inference." *IEEE Transactions on Signal Processing* 64: 1305–19.

Ogundijo, O E, and X Wang. 2017. "A Sequential Monte Carlo Approach to Gene Expression Deconvolution." *PLoS ONE* 12: 10.

Yip, S H, P C Sham, and J Wang. 2019. "Evaluation of Tools for Highly Variable Gene Discovery from Single-Cell Rna-Seq Data." *Briefings in Bioinformatics* 20: 1583–9.