ORIGINAL ARTICLE

Cytometry PART A
Journal of the International Society for Advancement of Cytometry

MIFlowCyt

SWIFT—Scalable Clustering for Automated Identification of Rare Cell Populations in Large, High-Dimensional Flow Cytometry Datasets, Part 1: Algorithm Design

Iftekhar Naim,[1] Suprakash Datta,[2] Jonathan Rebhahn,[3] James S. Cavenaugh,[3] Tim R. Mosmann,[3] Gaurav Sharma[4,5]*

[1]Department of Computer Science, University of Rochester, Rochester, New York

[2]Department of Computer Science and Engineering, York University, Ontario, Canada

[3]David H. Smith Center for Vaccine Biology and Immunology, University of Rochester Medical Center, Rochester, New York

[4]Department of Electrical and Computer Engineering, University of Rochester, Rochester, New York

[5]Department of Biostatistics and Computational Biology, University of Rochester, Rochester, New York

Received 30 May 2013; Revised 8 November 2013; Accepted 2 January 2013

Grant sponsor: National Institute for Allergy and Infectious Diseases through the Rochester Human Immunology Center; Grant number: R24AI054953; Grant sponsor: National Institute for Allergy and Infectious Diseases New York Influenza Center of Excellence; Grant number: HHSN266200700008C; Grant sponsor: National Center for Research Resources and the National Center for Advancing Translational Sciences through the University of Rochester CTSA, Grant number: UL1 RR024160

Additional Supporting Information may be found in the online version of this article.

Abstract

We present a model-based clustering method, SWIFT (Scalable Weighted Iterative Flow-clustering Technique), for digesting high-dimensional large-sized datasets obtained via modern flow cytometry into more compact representations that are well-suited for further automated or manual analysis. Key attributes of the method include the following: (a) the analysis is conducted in the multidimensional space retaining the semantics of the data, (b) an iterative weighted sampling procedure is utilized to maintain modest computational complexity and to retain discrimination of extremely small subpopulations (hundreds of cells from datasets containing tens of millions), and (c) a splitting and merging procedure is incorporated in the algorithm to preserve distinguishability between biologically distinct populations, while still providing a significant compaction relative to the original data. This article presents a detailed algorithmic description of SWIFT, outlining the application-driven motivations for the different design choices, a discussion of computational complexity of the different steps, and results obtained with SWIFT for synthetic data and relatively simple experimental data that allow validation of the desirable attributes. A companion paper (Part 2) highlights the use of SWIFT, in combination with additional computational tools, for more challenging biological problems.

Key terms

automated multivariate clustering; rare subpopulation detection; Gaussian mixture models; weighted sampling; ground truth data

Introduction

Flow cytometry (FC) has become an essential technique for interrogating individual cell attributes with a wide range of clinical and biological applications (1–4). The goals of FC analysis are to identify groups of cells that express similar physical and functional properties and to make biological inferences by comparing cell populations across multiple datasets. The massive size and dimensionality of modern FC data pose significant challenges for data analysis ($\sim 10^6$ cells, >35 dimensions in some instruments). FC data have traditionally been analyzed manually by visualizing the data in bivariate projections. This manual analysis is subjective, time consuming, can be inaccurate in case of overlapping populations, and scales poorly with increasing number of dimensions. Moreover, many discriminating features present in the high-dimensional data may not be distinguishable in 2D projections. As a result, automated multivariate clustering has become highly desirable for objective and reproducible assessment of high dimensional FC data. Recently several methods have been proposed, which can be broadly classified into two categories: (a) nonpro-

Correspondence to: Gaurav Sharma, Department of Electrical and Computer Engineering, University of Rochester, Rochester, New York, USA. E-mail: gaurav.sharma@rochester.edu

Published online 14 February 2014 in Wiley Online Library (wileyonlinelibrary.com)

DOI: 10.1002/cyto.a.22446

babilistic hard clustering (5–8) and (b) probabilistic soft clustering (9–14). Hard clustering, which assigns each cell to one of the possible clusters, is likely more familiar to users of manual gating and is also essential for cell sorting. Soft probabilistic clustering on the other hand determines, for each cell, a probability assignment distribution over the full set of clusters, thereby allowing for overlapping clusters.

Analysis of FC data seeks to identify biologically meaningful cell subpopulations[1] from per cell measurements of antigen expression correlates measured via a set of flurophore tags. Typical datasets exhibit a high dynamic range for the number of events in each subpopulation, i.e., within a dataset, there are subpopulations with a large percentage (10% or higher) of the total events and subpopulations with a small percentage of the total events (0.1% or lower). The small subpopulations are often biologically significant and therefore important to resolve. Distinguishing these small subpopulations is challenging because, in the measurement space, they often consist of observations that form skewed, non-Gaussian distributions that appear merged as "shoulders" of larger subpopulations with which they overlap.

To meet these challenges, we propose a soft mixture-model based framework "SWIFT" (Scalable Weighted Iterative Flow-clustering Technique), which scales to large FC datasets while preserving the capability of identifying small clusters representing rare subpopulations. *SWIFT differs algorithmically from prior methods in four main aspects:* (a) the mixture modeling is performed in a scalable framework enabled by weighted sampling and incremental fitting, allowing SWIFT to handle significantly larger datasets than alternative mixture model implementations; (b) the weighted sampling is explicitly designed to allow resolution of small potentially overlapping subpopulations in the presence of a high dynamic range of cluster sizes; (c) the algorithm includes a splitting and merging procedure that yields a final mixture model where each component is unimodal but not necessarily Gaussian; and (d) the determination of the number of clusters K is performed as an integral part of the algorithm via the intuitively appealing heuristic of unimodality. *Parts of the SWIFT framework have been previously presented in their preliminary form in (15). Recently, the detection of rare cell subpopulations has also been independently addressed in Ref. (14) using a hierarchical Dirichlet process model to solve the dual problems of finding rare events potentially masked by nearby large populations and to provide alignment of cell subsets over multiple data samples. Compared with (14) SWIFT achieves better resolution of rare populations (data presented in companion manuscript (16)). Also, the weighted iterative sampling and incremental fitting algorithmic approach in SWIFT strategy scales better to large datasets allowing the algorithm to operate on conventional workstations instead of requiring specialized GPU hardware. SWIFT is available for download at http://www.ece.rochester.edu/projects/siplab/Software/SWIFT.html.*

Problem Formulation

To describe our methodology in precise terms, we consider the following mathematical formulation for our problem: N independent events, each belonging to one of several classes that are unknown *a priori*, generate a corresponding set of N, d-dimensional observations $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N \in \mathbb{R}^d$. We will assume column vectors as our default notational convention so that each $\mathbf{x}_i$ is a $d \times 1$ vector. Given the $d \times N$ input dataset $\mathcal{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, we wish to estimate the number of distinct classes and the class for each of the N events. We refer to the estimated classes as *clusters* and denote by K the total number of clusters.

In the FC context, the events correspond to distinct triggering of FC measurements, usually caused by individual cells,[2] and the classes correspond to biologically meaningful cell subpopulations. For FC measurements, it is common for a given region of the d-dimensional observation space to contain a significant number of observations from different subpopulations. With some abuse of terminology, in such cases, we say that the corresponding subpopulations, or classes, overlap. Because of the overlaps between classes, it is appropriate to assign soft memberships, i.e., allow an event to belong to each of the K clusters with associated probabilities (or from an alternative perspective, to allow fractional memberships in each of the K clusters). Thus, our goal is to determine a membership probability matrix $\mathbf{\Omega}=\{\omega_{ij}\}$, where ω_{ij} represents the probability that event i belongs to cluster j, for $1 \le i \le N$ and $1 \le j \le K$, and $\sum_j \omega_{ij}=1$ for all $1 \le i \le N$.

A natural way to model the data in this setting is as a K-component mixture model. Specifically, we assume the given dataset $\mathcal{X}$ represents N independent observations of a d-dimensional random variable $\mathbf{X}$, that follows a K-component finite mixture model, whose probability density is given by:

[1]A subpopulation represents a set of events that is apparently homogeneous at the resolution of the FC experiment under consideration.

[2]Occasionally, the events may represent doublets composed of amalgamations of two cells each or debris from dead cells.

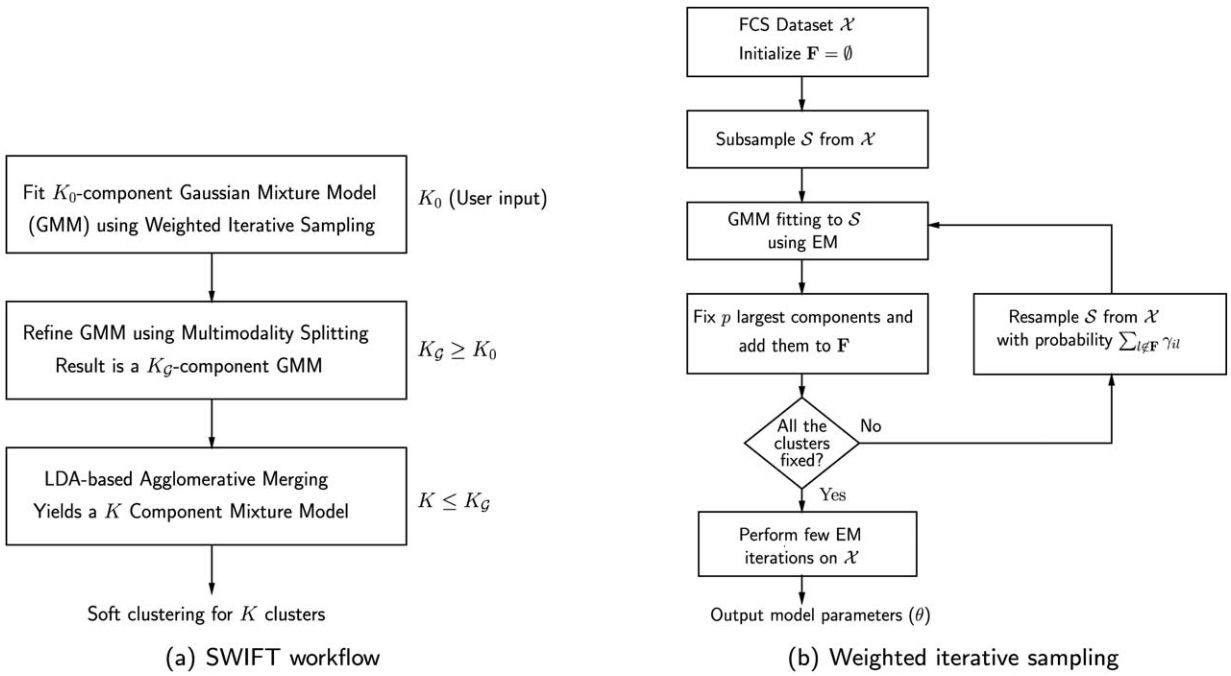

Figure 1. The SWIFT algorithm: (**a**) Overall workflow and (**b**) Weighted iterative sampling.

$$p(\mathbf{x}|\Theta)=\sum_{j=1}^{K}\pi_j f_j(\mathbf{x}|\Theta_j), \tag{1}$$

where $f_j(\mathbf{x}|\Theta_j)$ is the probability density function of the j-th mixture component having parameters Θ_j and mixing coefficient π_j ($\pi_j > 0$ and $\sum_{j=1}^{K}\pi_j=1$). Our goal is to estimate the parameter vector $\Theta=[\pi_j, \Theta_j]_{j=1}^{K}$ such that Θ maximizes the likelihood of the given data $\mathcal{X}$ and also the density function $f_j(\mathbf{x}|\Theta_j)$ in some parametric form. Once the mixture model parameter vector Θ is estimated, soft clustering can be performed by estimating the posterior membership probabilities using Bayes' rule, viz.,

$$\omega_{ij}=\frac{\pi_j f_j(\mathbf{x}_i|\Theta_j)}{\sum_{l=1}^{K}\pi_l f_l(\mathbf{x}_i|\Theta_l)}. \tag{2}$$

The finite mixture model therefore provides a framework for performing soft clustering in a principled manner, as has been done for a variety of problems (17, 18).

SWIFT Algorithm

Pragmatic considerations of complexity for the massive datasets encountered in FC motivated our choice of functional form for $f_j(\mathbf{x}|\Theta_j)$. Parameter estimation can be performed much more efficiently for Gaussian mixture models (GMMs) than for alternative models such as mixtures of skewed Gaussians or skewed t-distributions that allow a greater flexibility for modeling naturally occurring (e.g., FC) distributions, for a given number of components K. However, the value of K is, in truth, arbitrary and cannot be determined apart from external heuristic considerations. Because a wide class of distributions can be closely approximated by using sums of Gaussians (19, 20), we address non-Gaussianity of common FC data by using a larger number of Gaussians ($K_{\mathcal{G}} > K$) and allowing multiple Gaussians to represent a single non-Gaussian cluster.

In SWIFT, the probability density of $\mathbf{X}$ is approximated by fitting a $K_{\mathcal{G}}$ component ($K_{\mathcal{G}} \geq K$) GMM, and each density component f_j in Eq. (1) corresponds to a combination of one or more of these Gaussian components. Formally, the probability density $p(\mathbf{x}|\Theta)$ is approximated as:

$$p(\mathbf{x}|\Theta) = \sum_{j=1}^{K}\pi_j f_j(\mathbf{x}|\Theta_j) = \sum_{l=1}^{K_{\mathcal{G}}}\alpha_l g_l(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l), \tag{3}$$

where

$$g_l(\mathbf{x}|\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)=\frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}_l|^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_l)^T\boldsymbol{\Sigma}_l^{-1}(\mathbf{x}-\boldsymbol{\mu}_l)\right)$$

is the multivariate Gaussian distribution with mean μ_l, covariance matrix Σ_l, and mixing coefficient α_l. We seek to estimate the parameter vector of the GMM, $\Theta_{\mathcal{G}}=[\alpha_l, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l]_{l=1}^{K_{\mathcal{G}}}$. After obtaining $\Theta_{\mathcal{G}}$, we combine Gaussian mixture components (g_l) to represent the mixture components f_j of the general mixture model. Specifically, if the j-th mixture component f_j is a combination of the l_j Gaussians with indices $\{I_1^{(j)}, \ldots, I_{l_j}^{(j)}\}$, we obtain the parameters

$\Theta=[\pi_j, \Theta_j]_{j=1}^{K}$, such that $\pi_j=\sum_{m=1}^{l_j}\alpha_{l_m^{(j)}}$, and $\Theta_j=\{\boldsymbol{\mu}_{l_m^{(j)}}, \boldsymbol{\Sigma}_{l_m^{(j)}}\}_{m=1}^{l_j}$. Observe that the model in Eq. (3) represents a finite mixture model (17), where each individual mixture component is a combination of several Gaussian components.

The number $K_{\mathcal{G}}$ of Gaussians in Eq. (3) should be determined so as to provide an adequate approximation to the observed distributions. Specifically, it should provide enough resolution to identify rare subpopulations commonly of interest in FC data analysis, where it is often desirable to resolve subpopulations including 0.1% or fewer of the total events in a "background" of other larger subpopulations accounting for 10% or more of the total events. Intuitively, we expect that multimodal distributions do not correspond to a single subpopulation.

All these considerations motivated the *SWIFT* algorithm, which *consists of three main phases* shown schematically in Figure 1a: an *initial GMM fitting* using K_0 components; a *modality based splitting* stage that splits multimodal clusters and results in $K_{\mathcal{G}} \geq K_0$ Gaussian components in Eq. (3); and the final *modality-preserving merging* stage resulting in the $K \leq K_{\mathcal{G}}$ component general (not necessarily Gaussian) mixture model of Eq. (1), allowing representation of subpopulations with skewed but unimodal distributions as individual clusters. The individual phases are described in detail in the following subsections.

Scalable GMM Fitting Using Expectation Maximization

Traditionally, parameter estimation for GMMs is done using the Expectation Maximization (EM) algorithm (21), but the EM algorithm is computationally expensive for large FC datasets (e.g. $N \sim 10^6$ events, $K_{\mathcal{G}} \sim 10^2$ Gaussian components, and $d > 20$ dimensions). Each EM iteration requires $\mathcal{O}(NK_{\mathcal{G}}d^2)$ operations, and is therefore prohibitively slow. Moreover, FC datasets tend to show high dynamic ranges in subpopulation sizes. The EM algorithm often fails to isolate such small overlapping subpopulations, because of slow convergence rate. SWIFT's weighted iterative sampling addresses these twin challenges by scaling the EM algorithm to large datasets, while allowing better detection of small subpopulations. The parameter estimates are refined by performing a few iterations of the Incremental EM (IEM) (22) algorithm on the entire dataset $\mathcal{X}$. An optional scalable ensemble clustering step improves the robustness of clustering in a scalable manner. To make the description self-contained, we present a brief overview of the EM and the IEM algorithms in the context of GMM fitting in the Supporting Information (Section A).

Weighted iterative sampling based EM. Algorithm 1 and Figure 1b summarize the weighted iterative sampling based EM procedure used in SWIFT. Motivation and key steps are highlighted next. An intuitive way to reduce computational complexity for large datasets is to work on a smaller subsample $\mathcal{S}$ drawn from the dataset $\mathcal{X}$. When the mixing coefficients (α_j) exhibit a high dynamic range, a uniform random sample drawn from the dataset usually represents the large subpopulations with reasonable fidelity but is inadequate for resolving rare populations, for which parameter estimation is markedly poor when operating on a uniform subsample.

We start with a uniform random sample $\mathcal{S}$ containing n $n \ll N$ observations drawn from $\mathcal{X}$. First, a K_0 component GMM is fitted to $\mathcal{S}$. Next, we fix the parameters of the p (a user defined parameter) most populous Gaussians and reselect a sample of n observations from $\mathcal{X}$, drawn according to a weighted distribution, where the probability of selecting a data point equals the probability that the data point does not belong to the already fixed clusters. Specifically, let $\mathbf{F}$ be the set of Gaussian components whose parameters have already been fixed and γ_{ij}[3] be the posterior probability that $\mathbf{x}_i$ belongs to the jth Gaussian component. Then, in the next iteration, we resample according to a weighted distribution where the probability of selecting each point $\mathbf{x}_i$ is $1-\sum_{l\in\mathbf{F}}\gamma_{il}$. The EM algorithm is applied on the new sample with random reinitialization of the Gaussian components that are not fixed yet (the means are set to randomly chosen observations from the new sample). In each E-step, we estimate posterior probabilities (γ_{ij}) for all K_0 Gaussian components. In the M-step we re-estimate parameters of the remaining components excluding the already fixed ones. After each M-step, the mixing coefficients $\alpha_j, j \in \mathbf{F}$ are normalized such that they add up to $(1-\sum_{l\in\mathbf{F}}\alpha_l)$. As the algorithm proceeds, larger clusters get fixed and the weighted resampling favors selection of observations from smaller clusters, thereby improving the chances of discovering smaller subpopulations. The resampling and model-fitting steps alternate until all the cluster parameters are fixed. A visual demonstration of the weighted sampling method is shown in Figure 2. It can be seen (see Supporting Information, Section B) that *under idealized conditions when observed data are indeed drawn from a GMM and the parameters and posteriors for the fixed clusters are correctly estimated, the weighted iterative sampling algorithm proposed here exhibits the correct behavior.* The samples obtained with the weighted resampling are equivalent to samples that would be drawn from a mixture model consisting of only the clusters that are not fixed (so far), where the mixing coefficients remain proportional to their values in the original mixture but are re-normalized to meet the unit sum constraint. Furthermore, in the presence of the large dynamic range for the mixing coefficients, the *weighted iterative sampling mitigates problems with convergence in the vicinity of the true parameters* (Supporting Information, Section B). The weighted iterative sampling significantly reduces the computational complexity of each iteration of EM from $\mathcal{O}(NK_0d^2)$ to $\mathcal{O}(nK_0d^2)$, where n is the sample size ($n \ll N$).

[3]At each iteration step, the posterior probability γ_{ij} is obtained for the current GMM by a computation directly analogous to the computation of ω_{ij} in Eq. (2).

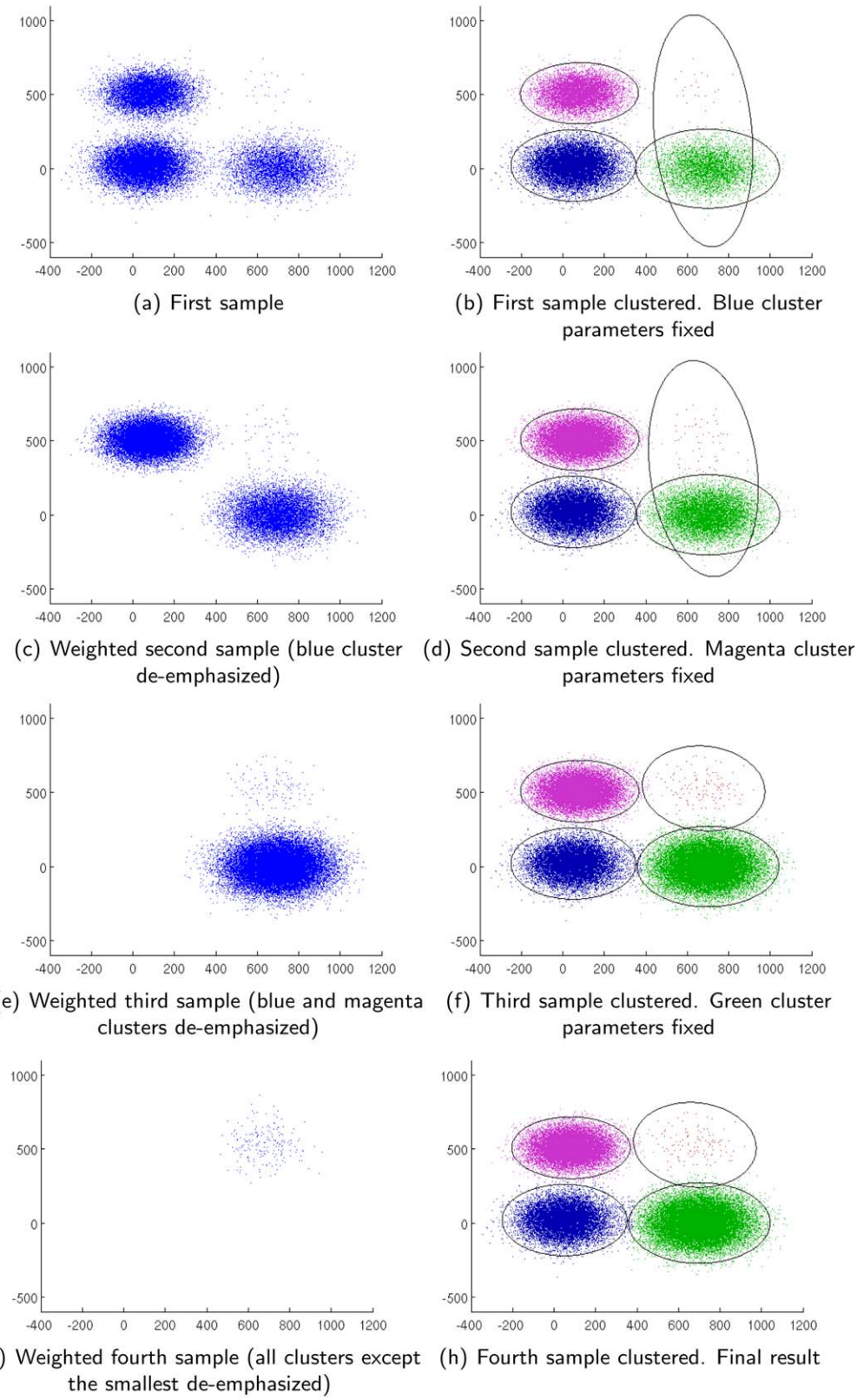

Figure 2. Weighted iterative sampling based Gaussian mixture model (GMM) clustering for better estimation of smaller subpopulations. Intermediate results along different stages of the algorithm and the final result are shown highlighting how smaller subpopulations are emphasized in the weighted iterative sampling process. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

Input: $\mathcal{X}, K_0, n, p$

$\mathcal{X}$: sequence of N data vectors $\{\mathcal{X}^{(i)}\}_{i=1}^{N}$

K_0: Number of initial Gaussian mixture components

n: Sample size

p: Number of components to fix at a time

Output: Θ_{K_0}: Parameters of the initial Gaussian mixture model (GMM)

1. Obtain set $\mathcal{S}$ of n random samples drawn from $\mathcal{X}$.
2. Estimate GMM parameters $\Theta_{\mathcal{S}}$ using EM on $\mathcal{S}$.
3. Estimate posterior probabilities γ_{ij} via an E-step on $\mathcal{X}$ using parameters $\Theta_{\mathcal{S}}$.
4 Let $\mathbf{F}$ be the set of Gaussian components whose parameters have been fixed. Initialize $\mathbf{F} \leftarrow \varnothing$.
5. repeat
6. Determine $\mathbf{F}_1 = \{$The p most populous Gaussian components $\in \mathbf{F}\}$ for the current model $\Theta_{\mathcal{S}}$.
7. Fix the parameters of components $\in \mathbf{F}_1$. Set $\mathbf{F} \leftarrow \mathbf{F} \cup \mathbf{F}_1$.
8. Resample a set of n observations $\mathcal{S}$ from $\mathcal{X}$ with a weighted distribution where each observation is selected with probability $(1 - \sum_{l \in \mathbf{F}} \gamma_{il})$.
9. Apply modified EM algorithm on $\mathcal{S}$ that does not update the parameters of already fixed components. In the M step, update only components $\in \mathbf{F}$.
10. Normalize the mixing probabilities $\alpha_j, j \in \mathbf{F}$, computed in the M step to $(1 - \sum_{l \in \mathbf{F}} \alpha_l)$.
11. Perform a single E-step on $\mathcal{X}$ to recalculate the posteriors γ_{ij}.
12. Until *all the components are fixed.*
13. $\Theta_{K_0}{}' \leftarrow$ parameters of all the (K_0) Gaussian components $\in \mathbf{F}$.
14. Perform a few (incremental) EM iterations on $\mathcal{X}$ with $\Theta_{K_0}{}'$ as initial parameters.
15. $\Theta_{K_0} \leftarrow$ parameters estimated in the previous step.

Algorithm 1. Weighted iterative sampling based EM in SWIFT

Incremental EM iterations. Upon completion of the weighted iterative sampling based EM procedure for GMM fitting, SWIFT performs a few (typically 10) EM iterations on the entire dataset to improve the fit taking the entire data into account. However, even a few iterations on the entire dataset $\mathcal{X}$ can be computationally expensive, particularly in terms of memory requirements; the posterior probability distribution $\Gamma = \{\gamma_{ij}\}, 1 \le i \le N, 1 \le j \le K_0$ requires $O(N \times K_0)$ storage, which can be prohibitive for large datasets. Therefore, we use memory-efficient IEM (22) (Supporting Information, Section A) for the iterations performed over the entire dataset. The IEM algorithm divides data into multiple blocks and performs a partial E-step, one block at a time. For each block, the partial E-step estimates the sufficient statistics for the associated block, which are used in the subsequent M-step for updating parameters. IEM is memory-efficient, because it processes only one block of data at a time. Moreover, IEM can exploit information from each data block earlier (without waiting for the entire data scan), and thus can improve the speed of convergence for large datasets (23) when each block is sufficiently large.

Multimodality Splitting

The initial GMM fitting may produce clusters that have several density maxima in the d-dimensional observation space. FC experts usually interpret each mode as a distinct subpopulation. Therefore, *SWIFT splits such multimodal clusters into unimodal subclusters.* Algorithm 2 summarizes this multimodality splitting procedure. Let $\mathcal{V}_i$ be the set of observations associated with the ith Gaussian cluster. SWIFT estimates one-dimensional kernel density functions for each of the d observation dimensions and d principal components of $\mathcal{V}_i$, where the optimal smoothing parameter for the kernel density estimation procedure is determined in a data-dependent manner using the normal optimal smoothing method (24). A cluster is identified as multimodal if any of the kernel density functions has more than one local maximum. If the i-th initial cluster is identified as multimodal, SWIFT fits a K_i component GMM to $\mathcal{V}_i$, where K_i is the smallest number of components such that each fitted subcomponent corresponds to a unimodal set of observations. To estimate K_i, SWIFT initiates GMM fitting with a value of $K_i = 2$, and increases $K_i \leftarrow K_i + 1$ until each of the fitted subcomponents is unimodal. After performing splitting for all the initial multimodal clusters, we get a $K_{\mathcal{G}}$ component GMM with refined parameters $\Theta_{K_{\mathcal{G}}}$, where $K_{\mathcal{G}} = \sum_{i=1}^{K_0} K_i$.

For small clusters, many small spurious modes often result because of the fact that there are not enough observations to allow for reliable density estimation. Therefore, modes that are t_{small} times smaller than the largest mode, for a chosen threshold t_{small} are ignored in estimating modality. Furthermore, each multimodal cluster is split into no more than $K_{\max}$ components. The upper bound $K_{\max}$ is useful for the background clusters that are too diverse and sparse and require a large number of components in order to render each component unimodal.[4]

In the GMM fitting procedure in SWIFT, we also identify some clusters as "background clusters" through an automatic background detection technique that extends the method described in Ref. (9). Background clusters are identified by their low density and high volume, where the volume of a cluster is approximated by the determinant of its covariance matrix, and its density is estimated as the ratio of its population size to its volume (9). SWIFT identifies a cluster as "background" if its density is less than the overall data density, and the cluster volume is larger than mean cluster volume.[5] The sparse background clusters are typically multimodal in many dimensions. Depending on the biological study, a user may or may not want to split these background clusters. Biologists interested in major populations do not need to analyze

[4]Based on empirical experiments on our datasets, we typically set $K_{\max} = 40$ and $t_{\text{small}} = 20$.

[5]Volume for a cluster (or entire dataset) is estimated as the determinant of the covariance of points in the cluster (entire data set).

background clusters. However, in some biological studies (e.g., stem cells, peptide stimulation, etc.), it is crucial to identify biologically significant small subpopulations (less than 100 observations, out of a total in the millions) that are assigned to background cluster(s). In such situations, these rare populations can be resolved by splitting the background cluster(s)—an option that can be enabled in SWIFT via a user-defined input parameter. Often background clusters do not have enough population sizes for reliable GMM fitting. To solve this problem, SWIFT performs an oversampling by replicating the observations in the background cluster with a small random perturbation and then performs splitting. This oversampling and background splitting operation is effective for finding rare subpopulations in large FC datasets.

The multimodality splitting stage is the most computationally expensive step in the current SWIFT implementation. Let $N^i_{\max}$ be the number of data points in the most populous multimodal cluster, $K_{\max}$ be the upper bound on the number of resulting split clusters from a single multimodal cluster, K_{m} be the number of such multimodal clusters, d be the number of dimensions, and $T_{\max}$ be the maximum number of EM iterations allowed. Then the worst case computational complexity of the modality splitting stage is $O(K_{\mathrm{m}} N^i_{\max} K^2_{\max} d^2 T_{\max})$.

Input: $\mathcal{X}, \Theta_{K_0}, K_{\max}$
$\mathcal{X}$: Input dataset
Θ_{K_0}: parameters of the initial K_0 component Gaussian mixture model
$K_{\max}$: upper bound on maximum number of Gaussians fit to an initial cluster
Output: $\Theta_{\mathcal{G}}, K_{\mathcal{G}}$
$\Theta_{\mathcal{G}}$: parameters of the refined Gaussian mixture model
$K_{\mathcal{G}}$: refined number of Gaussians
1. $K_{\mathcal{G}} \leftarrow 0$
2. for $i = 1$ ***to*** K_0 **do**
3. $\mathcal{V}_i \leftarrow$ set of observations in $\mathcal{X}$ associated with the ith initial Gaussian cluster.
4. $K_i \leftarrow 1$
5. if *isMultiModal*$(\mathcal{V}_i)$ **then**
6 repeat
7. $K_i \leftarrow K_i + 1$
8. $\Theta_i^{'} \leftarrow EM(\mathcal{V}_i, K_i)$
9. until $K_i \geq K_{\max}$ *or all the subclusters of* $\mathcal{V}_i$ *are unimodal*
10. end
11. $K_{\mathcal{G}} \leftarrow K_{\mathcal{G}} + K_i$;
12. Update the parameters $\Theta_{\mathcal{G}}$ according to $\Theta_i^{'}$;
13. end
14. Return final parameters $\Theta_{\mathcal{G}}$ and final number of clusters $K_{\mathcal{G}}$.

Algorithm 2. Multimodality splitting in SWIFT

LDA-Based Agglomerative Merging

The final step of SWIFT merges together Gaussian mixture components obtained from the GMM fitting and multimodality splitting stages, allowing representation of subpopulations with skewed but unimodal distributions. Merging mixture components to represent skewed subpopulations is well-established in the clustering literature (9, 11, 12, 20, 25, 26). We propose a novel agglomerative merging algorithm based on Fisher linear discriminant analysis (LDA) (27) that outperforms previously proposed entropy-based merging method (26), in terms of both speed and accuracy (Supporting Information Fig. S7 and Table S1). The algorithm is explicitly motivated by the need to maintain distinct unimodal clusters in the observed datasets as distinct subpopulations. For a pair of clusters associated with two GMM components, LDA allows us to compute the one-dimensional projection of the d-dimensional data for which the separation between the clusters is maximized. Clusters for which the LDA projection is unimodal are also unimodal in the d-dimensional space and can therefore be merged without compromising unimodality. This intuition is the basis of the method that we adopt for merging, which is described next.

The GMM estimation procedure combined with the modality based splitting process yields a set of $K_{\mathcal{G}}$ Gaussian mixture components. For $i = 1, 2, \ldots, K_{\mathcal{G}}$, denoting the ith Gaussian (mixture component) by g_i, we associate with it a corresponding cluster $\mathcal{X}_i$, comprising the subset of the observed data $\mathcal{X}$ that the mixture model identifies as belonging to g_i. Our LDA merging algorithm successively merges pairs of Gaussians until no further merging is possible while maintaining unimodality of associated cluster data points. For each pair of Gaussians (g_i, g_j), the symmetric KL divergence defined as

$$\mathcal{D}_s(g_i, g_j) = \frac{1}{2} Tr\left[\boldsymbol{\Sigma}_i^{-1}\boldsymbol{\Sigma}_j + \boldsymbol{\Sigma}_j^{-1}\boldsymbol{\Sigma}_i\right] + \frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^T\left[\boldsymbol{\Sigma}_i^{-1} + \boldsymbol{\Sigma}_j^{-1}\right](\boldsymbol{\mu}_i - \boldsymbol{\mu}_j) - d \quad (4)$$

is computed and the pairs are considered for merging in ascending order of the pairwise symmetric KL divergence. For a pair of Gaussians (g_i, g_j) under consideration, by using LDA on the corresponding pair of clusters $(\mathcal{X}_i, \mathcal{X}_j)$, we determine a unit norm $d \times 1$ vector w^* for which separation between the clusters is maximized (on average) in the one-dimensional linear projections $\mathbf{w}^{*T}\mathbf{x}_i$ and $\mathbf{w}^{*T}\mathbf{x}_j$ of d-dimensional observations $\mathbf{x}_i$ in $\mathcal{X}_i$ and $\mathbf{x}_j$ in $\mathcal{X}_j$. Specifically, $\mathbf{w}^*$ maximizes the ratio of the squared-difference of projected means to the sum of individual cluster variances (27). For each element $\mathbf{x}_{ij}$ in the combined set $\mathcal{X}_{ij} = \mathcal{X}_i \cup \mathcal{X}_j$ of observations from the two clusters, a corresponding LDA projection $y_{ij} = \mathbf{w}^{*T}\mathbf{x}_{ij}$ is then obtained. Modes (local maxima) in the 1D kernel density estimate for sample projected data $\mathcal{Y}_{ij} = \{y_{ij}\}$ are then determined to test for unimodality of the LDA projections for the combined cluster. The combined cluster $\mathcal{X}_{ij}$ is also tested for unimodality along all its given dimensions and principal components. The class-wise dispersions σ_i and σ_j of the projected data $\mathcal{Y}_i$ and $\mathcal{Y}_j$ for the individual clusters are also evaluated and their ratio $r_\sigma = \max(\sigma_i/\sigma_j, \sigma_j/\sigma_i)$ is computed. The pair of Gaussians (g_i, g_j) is merged if the three following conditions are met: (a) the LDA projection $\mathcal{Y}_{ij}$ is unimodal, (b) $\mathcal{X}_{ij}$ is

unimodal along original data axes and principal component directions, and (c) r_σ is less than a certain threshold τ_σ (we set $\tau_\sigma=3$). The screening based on dispersion ratio helps us to avoid merging a dense foreground cluster with a sparse background cluster. If a merge occurs, we proceed to the next iteration of agglomerative merging after computing the symmetric KL divergence of the merged cluster to other Gaussians in the GMM.[6] If on the other hand, a merge does not occur because at least one of the three test conditions is violated, we move on to the next pair in the ascending symmetric KL divergence order. The merging algorithm continues until no such pairs can be found.

A sparse cluster may get subsumed by the tail of a dense cluster and may not appear as a separate mode even if the underlying distribution is multimodal. We avoid this pitfall by performing the LDA-based modality check not only for the actual observations of the two Gaussian clusters g_i and g_j, but also for synthetic data points randomly sampled from the Gaussians. By sampling an equal number of points from both components, issues related to imbalanced cluster densities are avoided.

A naive implementation of the proposed LDA merging procedure requires $O(K_\mathcal{G}^3)$ LDA estimations in the worst case, resulting in $O((N_{\max}^i d^2+d^3)K_\mathcal{G}{}^3)$ complexity, where $N_{\max}^i$ is the population size for the most populous cluster. We reduce the number of LDA estimations very significantly by filtering out Gaussian component pairs that have almost no overlap, because pairs of Gaussian components whose means differ by a large amount in relation to their standard deviation (in the d-dimensional space) will be multimodal in their LDA projection and need not be considered as prospects for merging. Specifically, we approximate a Gaussian component g_j by a multidimensional ellipsoid with center μ_j and dispersion $4\mathbf{\Sigma}_j$, and estimate (multidimensional) rectangular bounding boxes for the ellipsoids. If the bounding boxes for two Gaussians do not intersect, then their associated ellipsoids cannot intersect and the corresponding pairs of Gaussians are considered non-overlapping. Determining whether 2 rectangular boxes in d-dimensions intersect requires only $O(d)$ operations and is significantly faster than directly determining whether two d-dimensional ellipsoids intersect. A large number of candidate Gaussian pairs are eliminated from consideration by this efficient bounding box based filtering, and LDA estimation is required only for the remaining pairs. Moreover, at each merging step the LDA-based modality criterion needs to be recomputed only for the merged cluster produced in the previous merging step. Values for the other cluster pairs computed previously are reused, saving computation. Algorithm 3 summarizes the LDA based merging step used in SWIFT and Figure 3 presents a visualization of the operations in the algorithm using a sample 2-D dataset.

[6]KL divergences involving a merged cluster are approximated by using Eq. (4) with the mean and variance for the merged cluster, i.e., by using a Gaussian approximation for the merged cluster.

Input: $\mathcal{X}, \Theta_\mathcal{G}$
$\mathcal{X}$: Input dataset
$\Theta_\mathcal{G}$: parameters of the $K_\mathcal{G}$ component Gaussian mixture model
Output: Θ, K
Θ: parameters of the combined mixture model
K: final number of clusters
1. Initialize: $K' \leftarrow K_\mathcal{G}, \Theta' \leftarrow \Theta_\mathcal{G}$
2. repeat
3. **for** $i = 1$ **to** K' **do**
4. $E_i \leftarrow$ the ellipsoid with center $\boldsymbol{\mu}_i$, and dispersion $4\mathbf{\Sigma}_i$
5. **end**
6. $Q \leftarrow \varnothing$
7. **for each** *(i,j) such that* $1 \le i, j \le K'$ **do**
8. $B_i \leftarrow$ the smallest bounding box covering E_i
9. $B_j \leftarrow$ the smallest bounding box covering E_j.
10. **if** *intersect*(B_i, B_j) **then**
11. $Q \leftarrow Q \cup (i, j)$
12. **end**
13. **end**
14. Estimate the pairwise symmetric KL divergence $D_{KL}=\{d_{ij}\}, 1 \le i, j \le K'$ among the K' Gaussian components in the current model Θ'.
// See text for full details of unimodality test. Following version is abbreviated due to space constraints.
15. **for each** $(i, j) \in Q$ *ordered by ascending value of* d_{ij} **do**
16. $\mathcal{X}_i \leftarrow$ set of observations sampled from g_i
17. $\mathcal{X}_j \leftarrow$ set of observations sampled from g_j
18. $(\mathcal{Y}_i, \mathcal{Y}_j) \leftarrow$ LDA $(\mathcal{X}_i, \mathcal{X}_j)$
19. $\sigma_i \leftarrow$ standard deviation of $\mathcal{Y}_i, \sigma_j \leftarrow$ standard deviation of $\mathcal{Y}_j$
20. $r_\sigma = \max(\sigma_i/\sigma_j, \sigma_j/\sigma_i)$
21. **if** *isUnimodal*$(\mathcal{X}_i \cup \mathcal{X}_j)$ *and isUnimodal*$(\mathcal{Y}_i \cup \mathcal{Y}_j)$ *and* $r_\sigma < \tau_\sigma$ **then**
22. Merge (g_i, g_j)
23. $\Theta' \leftarrow$ the updated model after merging (g_i, g_j)
24. $K' \leftarrow K'-1$
25. break;
26. **end**
27. **end**
28. until *no more merging is possible*
29. $\Theta \leftarrow \Theta', K \leftarrow K'$
30. Return final parameters Θ and final number of clusters K.

Algorithm 3. LDA-based agglomerative merging in SWIFT

Results

For proper evaluation and validation of any clustering algorithm, one needs reliable ground truth data. To address this challenge, one can use either simulated data, or electronically mixed data. In this article, we report on experiments for evaluating SWIFT using both approaches. Detailed evaluation of SWIFT for a biologically relevant analysis is presented in the companion article (16).

Results on Simulated Data

In this section, using simulated mixtures of Gaussians, we evaluate SWIFT's scalability and capability for detecting

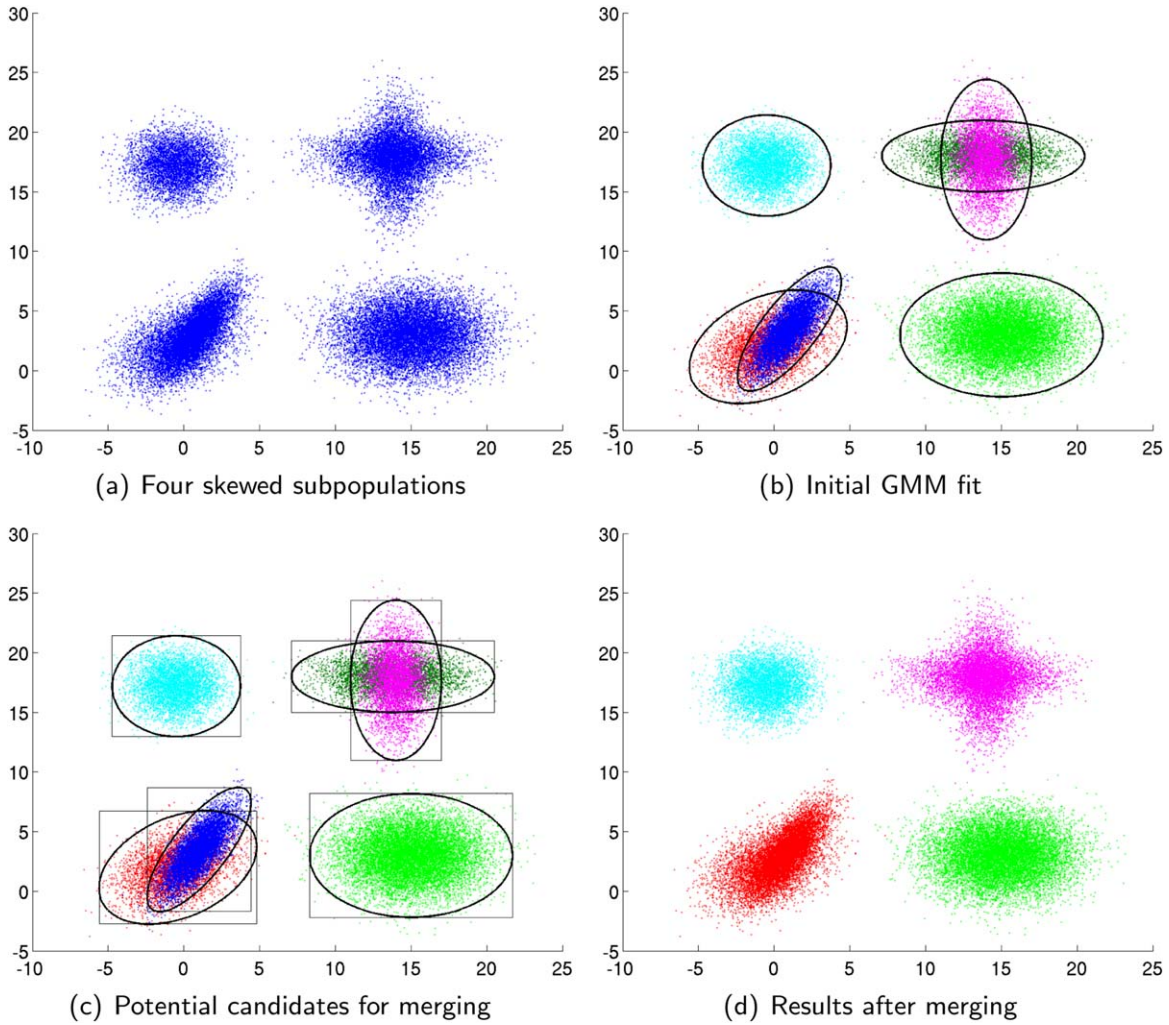

Figure 3. Cluster merging in SWIFT illustrated via a 2D example: (**a**) Four original skewed subpopulations, (**b**) Initial GMM fit, (**c**) Potential pairs considered for merging, the bounding box filtering introduced for computational efficiency eliminates all pairs except (1, 2) and (5, 6), and (**d**) Resulting clusters after merging. Note that in the final result, the original skewed and non-Gaussian subpopulations are well-represented via the merged clusters formed from combining initially fit Gaussians. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

rare populations, and compare these against the traditional EM algorithm. The main reasons for using simulated data are two-fold. First, we know full ground truth for simulated data for each of the clusters. Second, the traditional EM algorithm is prohibitively slow for actual large, high dimensional FC datasets, making the direct comparison on actual FC data prohibitively time consuming (or impossible to complete using the computational hardware we use for SWIFT).

A synthetic mixture of two-dimensional Gaussians with 6-components (shown in Fig. 4) was generated, where the mixing coefficients of the Gaussian components were chosen as $1\times10^6, 7.5\times10^5, 1.9\times10^5, 5\times10^4, 1\times10^4$, and 2×10^3 to be *representative of situations with large dynamic range that are of primary interest to us*. For this dataset, GMM parameters were estimated by using both the traditional EM algorithm and SWIFT's weighted iterative sampling based EM algorithm with the number of Gaussians K_0 set to 6 in both cases. The sample size for the weighted sampling was chosen as $n = 20{,}000$.

For quantitative evaluation of clustering accuracy, we estimate the error by computing the symmetric Kullback–Leibler (KL) divergence between each estimated Gaussian parameter and the associated true Gaussian parameter, where correspondence between the estimated and true Gaussians is first determined by a weighted bipartite graph matching (28) (also using symmetric KL divergence as matching cost). For each cluster, the error in estimated parameters is computed as the symmetric KL divergence between the estimated parameters and the true parameters for the matching Gaussian determined by the bi-partite matching. An overall error is also computed as the sum of the errors over all six clusters.

Since the EM algorithm only assures convergence to a local optimum, we performed 10 repeated runs of EM with random initializations, and chose the run with the maximum log-likelihood. To ensure the estimations are statistically significant, we performed the same experiment (EM fitting with 10

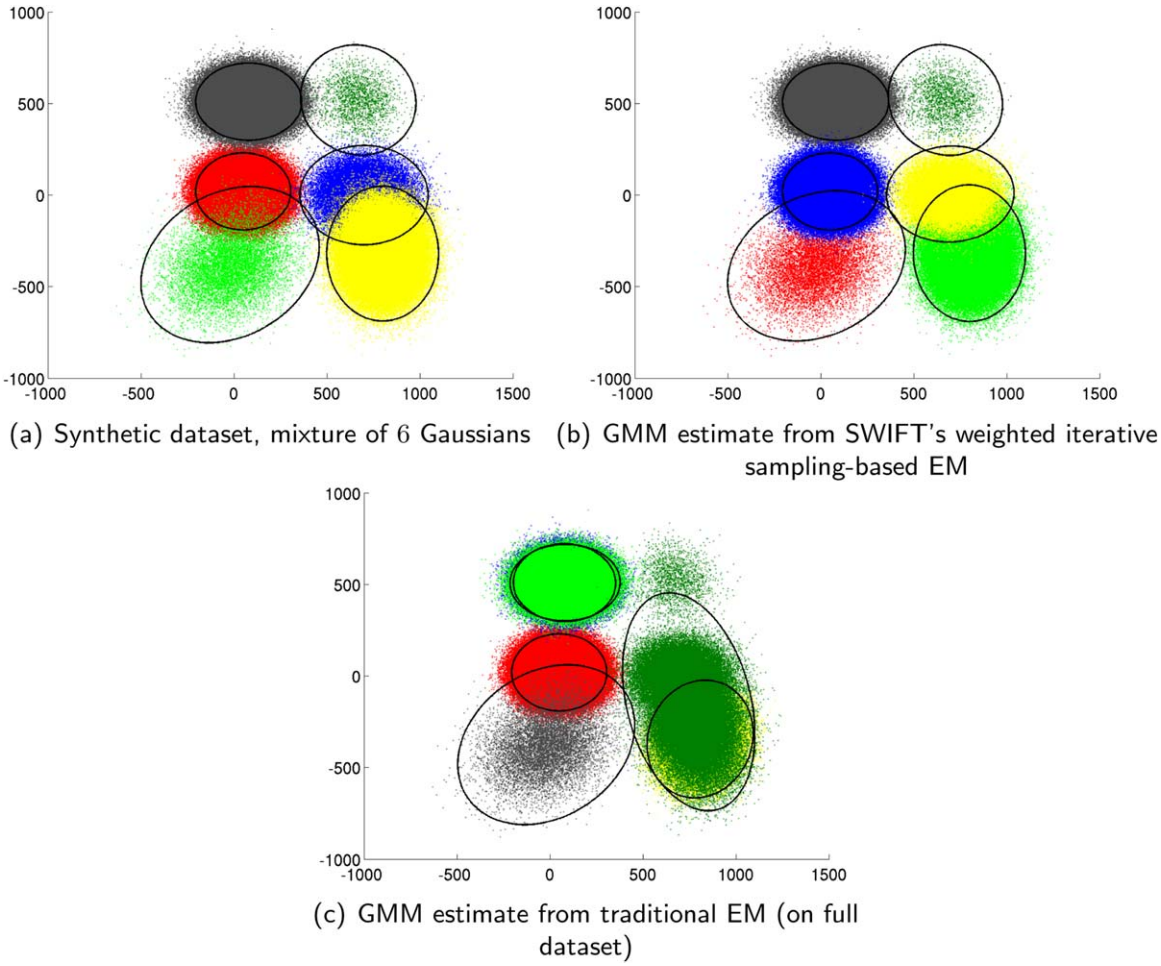

Figure 4. Comparison of weighted sampling based EM and the traditional EM algorithm on a synthetic mixture of 6 Gaussians: (**a**) Original dataset, (**b**) GMM estimate from the weighted sampling based EM used in SWIFT, and (**c**) GMM estimate from traditional EM algorithm. Note that smallest subpopulation is missed by the traditional EM algorithm but is represented with good accuracy by the weighted sampling based EM used in SWIFT. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

repetitions) 10 times and then finally estimated the average runtime, total error, and the error associated with the smallest cluster.

The results are presented in Table 1 and are shown in Figure 4 for a typical EM run. The weighted iterative sampling based EM is nearly 18 times faster and estimates the parameters of the smallest cluster with significantly greater accuracy than the traditional EM algorithm, which performs rather poorly. The poor performance of the traditional EM is due to: (a) the slow convergence of EM in the presence of overlapping and small clusters (see Supporting Information, Section B), and (b) convergence of EM to poor local optima depending on random initialization. The results clearly illustrate the advantages of the weighted iterative sampling for large datasets with high dynamic range in mixing coefficients. The weighted iterative sampling also provides a significant computational benefit. For a typical $d = 17$ dimensional FC dataset with $N = 1.5$ million events, a pure IEM approach for the initial mixture modeling phase, without the weighted iterative sampling in SWIFT and with an IEM block size of 50,000, increases the computational time by a factor of 10.53 and memory requirement by a factor of 1.8 (reported data are on an 8-core 2.4 GhZ Mac workstation) while providing results comparable with the traditional EM where the smaller clusters are frequently overwhelmed by larger clusters, though this can often be remedied by the subsequent splitting and merging stages of SWIFT.

Although the above example explored a large dynamic range, typical dynamic ranges for FC data are even larger. In the above example, the smallest cluster had 2000 points out of a total of 2 million, whereas actual FC datasets often have biologically significant subpopulations with fewer than a hundred cells in a sample of 2 million cells. We therefore also evaluated the performance of the weighted iterative sampling based EM as the size of the smallest cluster is further reduced; specifically, we generated 5 mixtures, where the smallest cluster sizes are set to 1500, 1000, 500, 200, and 100, respectively and the remaining clusters were left unchanged from the previous example. The results obtained are summarized in Table 2 and indicate that SWIFT's weighted iterative sampling works well until the

Table 1. Comparison of the weighted iterative sampling based EM against the traditional EM for a synthetic two-dimensional Gaussian mixture with mixing coefficients 1×10^6, 7.5×10^5, 1.9×10^5, 5×10^4, 1×10^4, and 2×10^3 chosen to be representative of the high dynamic range encountered for rare population detection

	WEIGHTED ITERATIVE SAMPLING	TRADITIONAL EM
Avg runtime (s)	134.1	2414.1
Avg cumulative error	0.0157	37.687
Avg errors for the smallest cluster	0.0012	34.3397

Listed error values correspond to symmetric KL divergences averaged over 10 independent runs. See text for details.

point where the smallest cluster has 200 points out of a total of 2 million. Results incorporating the additional stages (split and merge) in SWIFT also included within the table show that these additional steps further improve SWIFT's capability to detect small clusters.

Results on Flow Cytometry Data

A key challenge in validation on actual FC data is the scarcity of datasets with ground truth. Visual identification of populations via manual gating is hardly a gold standard, because of several limitations. First, gating is usually focused, rather than exhaustive, and not suitable for validation of all clusters. Second, the gating procedure cannot exploit high dimensional features and is also less accurate in the presence of cluster overlap. Third, the subjectivity of gating is well-known to contribute to the variability of FC analysis results (29). Therefore, an objective validation is desirable.

The Rochester Human Immunology Center generated a pair of datasets for which ground truth labels can be applied: one consisted of human peripheral blood cells, and the other consisted of mouse splenocytes. Both human and mouse cells were stained with the same set of fluorescently-labeled antibodies (directed against homologous proteins in both species) such that half of the antibodies were human-specific, and the rest were mouse-specific. Human antigens in a human cell bind only to the antihuman antibodies and express high signal for a subset of human antibodies and low signal for all the mouse antibodies. The mouse cells exhibit the opposite behavior. FC data was acquired for both samples using an LSR II cytometer (BD Immunocytometry Systems). The datasets are made available on the FlowRepository server (30) for use by other researchers for testing FC data analysis algorithms.

We electronically mixed these two datasets (total 544,000 observations and 21 dimensions), and created a series of hybrid datasets containing both human and mouse cells, where the label for each cell (either human or mouse) is known because of the electronic mixing. SWIFT was used for clustering each electronic mixture without using the human/mouse label in the clustering process. An ideal clustering solution should resolve the distinction between human and mouse groups and produce clusters that contain either only human cells, or only mouse cells, but not both. We note here that the dataset and the evaluation task are explicitly designed to allow validation against known ground truth, which makes them atypical of common FC analysis tasks. A companion article (16) uses datasets and tasks that are typical of a substantial field of immune response evaluation and provides information on the validation of SWIFT's ability to find rare clusters, and also to find clusters that are biologically significant.

The initial Gaussian mixture model fitting was done with $K_0 = 80$ Gaussian components. After the initial clustering, SWIFT's multimodality splitting resulted in 148 Gaussians, and its LDA-based agglomerative merging resulted in 122 final clusters. Each of these 122 clusters was classified as either human or mouse by a majority decision rule. Figure 5a shows the actual number of human and mouse cells per cluster. Figure 5b shows the fractional proportion. Almost all the clusters are well-resolved as either only human or only mouse.

To evaluate SWIFT's rare population detection using sensitivity analysis, we electronically mixed varying proportions of human and mouse cells and observed how its performance varied with decreasing proportion of human cells: 50%, 25%, 10%, 1%, and 0.1%. By definition, $precision = \frac{TP}{TP+FP}$ and $recall = \frac{TP}{TP+FN}$. In this experiment, we benchmarked detection of the human clusters as the proportion of human cells decreases. Therefore, the precision and recall can be equivalently redefined as:

$$Precision = \frac{Human_{detected} \cap Human_{true}}{Human_{detected}} \quad (5)$$

$$Recall = \frac{Human_{detected} \cap Human_{true}}{Human_{true}} \quad (6)$$

The results (Table 3) show that SWIFT can resolve up to 1% human cells with high precision and recall. For the case of 0.1%, SWIFT correctly identified 2 human clusters

Table 2. Performance of the weighted iterative sampling based EM and the overall SWIFT (weighted sampling + split + merge) for small cluster detection in a total population size of 2 million events.

	WEIGHTED SAMPLING		WEIGHTED SAMPLING + SPLIT+MERGE	
SMALLEST CLUSTER SIZE	AVG TOTAL ERROR	SMALLEST CLUSTER ERROR	AVG TOTAL ERROR	SMALLEST CLUSTER ERROR
1500	0.0159	0.0019	0.1020	0.0003
1000	0.0128	0.0128	0.0198	0.0046
500	0.0220	0.0220	0.0751	0.0044
200	23.3622	23.3622	1.7141	1.4561
100	27.4113	27.0221	7.1430	6.7043

Listed error values correspond to symmetric KL divergences averaged over 10 independent runs. See text for details.

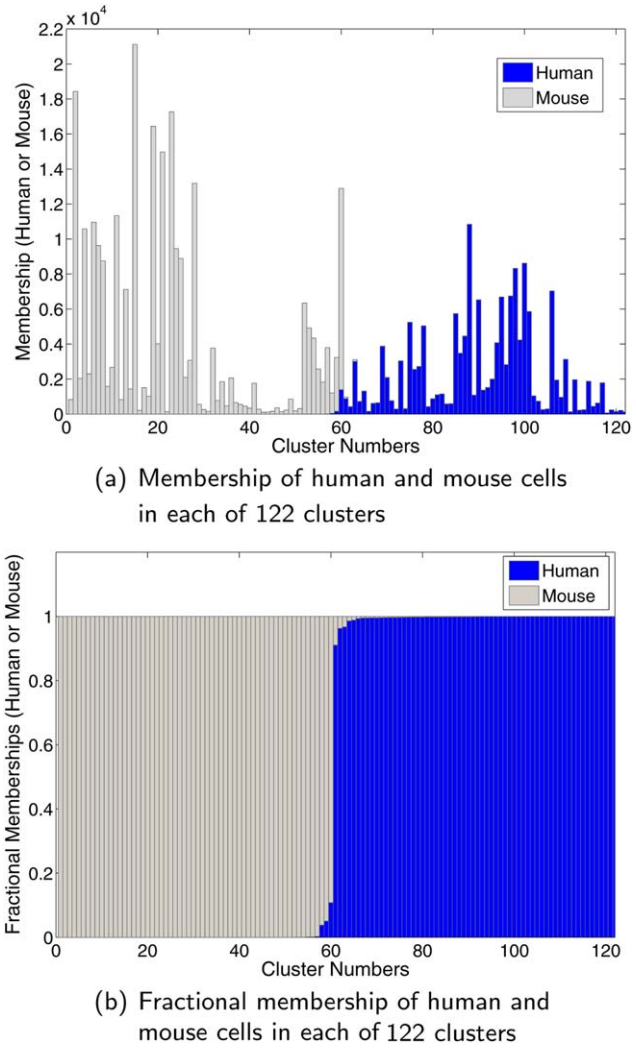

Figure 5. Results from SWIFT clustering of the known-ground-truth, electronically mixed, human-mouse dataset. SWIFT yields 122 clusters that clearly separate the human vs. mouse cells: most clusters are comprised of entirely human or entirely mouse cells. See text and caption for Supporting Information Fig. S.12 for details of the dataset. [Color figure can be viewed in the online issue, which is available at wileyonlinelibrary.com.]

with high recall, but the precision is relatively low (68.40%) because these human clusters also included quite a few mouse cells. For this dataset, we also compared SWIFT against FLOCK (5). FLOCK also resolves this simple dataset but with greater overlap (results shown in Supporting Information, Fig. S.12).

Discussion

SWIFT incorporates several novel components to address the challenges arising in FC. All the three stages of SWIFT are motivated by two major requirements: scalability to large datasets and identification of rare populations. All major components of SWIFT (weighted iterative sampling, the incremental EM iterations, and efficient LDA-based merging) are designed to be efficiently scalable to big datasets, providing a significant improvement over the existing soft clustering methods (9–12, 14). SWIFT identifies rare populations using weighted iterative sampling and multimodality splitting. The multimodality splitting stage serves a critical role for rare subpopulation identification. SWIFT can also represent skewed clusters by LDA-based agglomerative merging, which reduces the number of clusters while preserving the distinct unimodal populations. The interplay between multimodality splitting and merging results in a reasonable number of clusters, uses a sensible heuristic (modality of clusters), and is more intuitive as compared to the knee point in BIC or entropy plots previously used (10, 11). Finally, the soft clustering used in SWIFT is useful for comprehending overlapping clusters (Supporting Information, Section H) as compared with alternative hard clustering methods such as *k*-means (31) or spectral clustering (6). SWIFT is partly similar to flowPeaks (13) in that they both rely on the unimodality criterion. However, flowPeaks aims for major peaks only (no modality splitting stage), and tends to miss small overlapping clusters. The significance of modal regions in identifying interesting subpopulations has also motivated curvHDR (32), where high curvature regions are used to identify the modal regions, which are then exploited for (partly) automating gating.

A recent article (14) describes an alternative approach to rare population detection and provides a point of reference for comparing SWIFT against the current state of the art in FC data analysis methods designed specifically for rare population identification. In (14), FC data are modeled as hierarchical Dirichlet process Gaussian mixture model (HDPGMM) to solve the dual problems of finding rare events potentially masked by nearby large populations and to provide alignment of cell subsets over multiple data samples. The HDPGMM is shown to identify biologically relevant subpopulations occurring at frequencies in the 0.01–0.1% of the entire dataset and the method is shown to be superior at finding rare populations as compared with manual gating (using a panel of 10 people), FLAME (12), FLOCK (33) (albeit indirectly), and flowClust (34). These comparisons were done with 3 color (five-dimensional) FCS 2.0 (FACSCalibur) dataset, having around 50,000 events. In our companion manuscript (16), we demonstrate that SWIFT handles much larger datasets (having tens of millions of events with 17 independent dimensions) and identifies cell subpopulations at a

Table 3. Performance of SWIFT with varying proportion of human and mouse cells

PERCENTAGE OF HUMAN CELLS (%)	PRECISION (%)	RECALL (%)	HUMAN CLUSTERS
50	99.59	99.93	49
25	99.62	99.83	33
10	99.43	95.90	21
1	91.82	99.34	11
0.1	68.40	99.48	2

frequency as low as 10^{-6} in 17-dimensional FC datasets of up to 25 million events, which is significantly more sensitive than the existing current state of the art. A direct comparison of SWIFT against other existing FC data analysis methods is stymied by the fact that most existing methods do not scale to the extremely large datasets we are exploring, nor are these designed to detect rare populations at the level of sensitivity targeted by SWIFT. These claims are supported by benchmarking results on smaller datasets that we report in the supporting information accompanying our companion manuscript (16).

The weighted iterative sampling is one of the key contributions of SWIFT. Most of the existing scalable EM variants (35, 36) do not specifically address the challenge of rare population detection. Moreover, some assumptions of these methods are quite restrictive. For example, the scalable EM (SEM) (35) algorithm requires the covariance matrix to be diagonal, and the multistage EM (36) assumes all the clusters to share the same covariance matrix. These assumptions are too restrictive for FC data. SWIFT provides sufficient flexibility by allowing full covariance matrices for each individual Gaussian and performs well in the presence of rare populations. Although we implemented the weighted iterative sampling for mixture of Gaussians only, the method is general enough and can be extended to other soft clustering methods (e.g., mixture of *t* distributions, mixture of skewed *t* distributions, fuzzy *c*-means, etc.).

The LDA-based agglomerative merging combined with a pruning process allows efficient and robust merging of Gaussian mixture components. The efficiency of the LDA-based agglomerative merging carries over to other applications where the number of observations and the number of clusters are much larger than the number of dimensions. Unlike the entropy-based merging, our LDA criterion is insensitive to relative cluster population sizes (see Supporting Information, Section E and Fig. S.7), and is guided by the modality criterion.

Conclusion

This article presents the algorithm design for SWIFT (Scalable Weighted Iterative Flow-clustering Technique). SWIFT uses a three stage workflow consisting of iterative weighted sampling, multimodality splitting, and unimodality-preserving merging, to scale model-based clustering analysis to the large high-dimensional datasets common in modern FC, while retaining resolution of subpopulations with rather small relative sizes—populations that are often biologically significant. Evaluations over synthetic datasets demonstrate that SWIFT offers improvements over conventional model-based approaches in scaling to large datasets and in resolving small populations. In the companion manuscript (16), SWIFT is applied to a task typical in immune response evaluation and both scaling to very large FC datasets (having tens of millions of events) and capability to identify extremely rare populations (1 in 10^6 of the total events) are demonstrated. SWIFT is available for download at http://www.ece.rochester.edu/projects/siplab/Software/SWIFT.html.

Acknowledgments

The authors thank Jyh-Chiang (Ernest) Wang for collecting the Human-Mouse dataset used in the second reported experiment and Sally Quataert for helpful discussions.

Literature Cited

1. Shapiro H. Practical Flow Cytometry, 4th edn. New York, NY: Wiley; 2003.
2. McLaughlin BE, Baumgarth N, Bigos M, Roederer M, De Rosa SC, Altman JD, Nixon DF, Ottinger J, Oxford C, Evans TG, et al. Nine-color flow cytometry for accurate measurement of T cell subsets and cytokine responses. Part I: Panel design by an empiric approach. Cytom Part A 2008;73A:400–410.
3. Nolan JP, Yang L. The flow of cytometry into systems biology. Brief Funct Genomics Proteomics 2007;6:81–90.
4. Perfetto SP, Chattopadhyay PK, Roederer M. Seventeen-colour flow cytometry: Unravelling the immune system. Nat Rev Immunol 2004;4:648–655.
5. Qian Y, Wei C, Eun-Hyung Lee F, Campbell J, Halliley J, Lee JA, Cai J, Kong YM, Sadat E, Thomson E, et al. Elucidation of seventeen human peripheral blood B-cell subsets and quantification of the tetanus response using a density-based method for the automated identification of cell populations in multidimensional flow cytometry data. Cytom Part B: Clin Cytom 2010;78B:69–82.
6. Zare H, Shooshtari P, Gupta A, Brinkman R. Data reduction for spectral clustering to analyze high throughput flow cytometry data. BMC Bioinform 2010;11:403, doi: 10.1186/1471–2105-11–403.
7. Aghaeepour N, Nikolic R, Hoos H, Brinkman R. Rapid cell population identification in flow cytometry data. Cytom Part A 2011;79A:6–13.
8. Qiu P, Simonds EF, Bendall SC, Gibbs KD Jr, Bruggner RV, Linderman MD, Sachs K, Nolan GP, Plevritis SK. Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE. Nat Biotechnol 2011;29:886–891.
9. Chan C, Feng F, Ottinger J, Foster D, West M, Kepler T. Statistical mixture modeling for cell subtype identification in flow cytometry. Cytom Part A 2008;73A:693–701.
10. Lo K, Brinkman R, Gottardo R. Automated gating of flow cytometry data via robust model-based clustering. Cytom Part A 2008;73A:321–332.
11. Finak G, Bashashati A, Gottardo R, Brinkman R. Merging mixture components for cell population identification in flow cytometry. Adv Bioinform 2009; vol. 2009, Article ID 247646, doi:10.1155/2009/247646.
12. Pyne S, Hu X, Wang K, Rossin E, Lin TI, Maier LM, Baecher-Allan C, McLachlan GJ, Tamayo P, Hafler DA, et al. Automated high-dimensional flow cytometric data analysis. Proc Natl Acad Sci U S A 2009;106:8519-8524.
13. Ge Y, Sealfon SC. flowPeaks: A fast unsupervised clustering for flow cytometry data via *k*-means and density peak finding. Bioinformatics 2012;28:2052–2058.
14. Cron A, Frelinger J, Lin L, Gouttefangeas C, Singh SK, Britten CM, Welters MJ, van der Burg SH, West M, Chan C. Hierarchical modeling for rare event detection and cell subset alignment across flow cytometry samples. PLoS Comput Biol 2013;9: e1003 130, doi:10.1371/journal.pcbi.1003130.
15. Naim I, Datta S, Sharma G, Cavenaugh J, Mosmann T. SWIFT: Scalable weighted iterative sampling for flow cytometry clustering. Proceedings of IEEE International Conference Acoustics Speech and Signal Processing, Dallas, Texas, USA, 2010;509–512.
16. Mosmann TR, Naim I, Rebhahn J, Datta S, Cavenaugh JS, Weaver JM, Sharma G. SWIFT—scalable clustering for automated identification of rare cell populations in large, high-dimensional flow cytometry datasets. Part 2: Biological evaluation. Cytometry Part A 2014; doi:10.1002/cyto.a.22445.
17. McLachlan G, Peel D. Finite Mixture Models. New York, NY: Wiley InterScience; 2000.
18. Figueiredo M, Jain A. Unsupervised learning of finite mixture models. IEEE Trans Pattern Anal Mach Intel 2002;24:381–396.
19. Fraley C, Raftery A. Model-based clustering, discriminant analysis, and density estimation. J Am Stat Assoc 2002;97:611–631.
20. Hennig C. Methods for merging Gaussian mixture components. Adv Data Anal Classification 2010;4:3–34.
21. Dempster A, Laird N, Rubin D. Maximum likelihood from incomplete data via the EM algorithm. J R Stat Soc Ser B (Methodological) 1977;39:1–38.
22. Neal RM, Hinton GE. A view of the EM algorithm that justifies incremental, sparse, and other variants. In: Jordan MI, editor. Learning in Graphical Models, NATO ASI Series, vol. 89. Dordrecht, Netherlands: Kluwer Academic; 1998. pp 355–368, doi: 10.1007/978-94-011–5014-9_ 12.
23. Thiesson B, Meek C, Heckerman D. Accelerating EM for large databases. Mach Learn 2001;45:279–299.
24. Bowman A, Azzalini A. Applied Smoothing Techniques for Data Analysis: The kernel Approach with S-Plus Illustrations. New York, NY: Oxford University Press; 1997.
25. Tantrum J, Murua A, Stuetzle W. Assessment and pruning of hierarchical model based clustering. Proc. Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM: August 24–27, 2003, Washington, DC, USA, 2003; p 205.
26. Baudry J, Raftery A, Celeux G, Lo K, Gottardo R. Combining mixture components for clustering. J Comput Graph Stat 2010;19:332–353.

27. McLachlan G. Discriminant Analysis and Statistical Pattern Recognition. New York: Wiley InterScience; 1992.
28. Kuhn H. The Hungarian method for the assignment problem. Nav Res Logist Q 1955; 2:83–97.
29. Maecker H, Rinfret A, D'Souza P, Darden J, Roig E, Landry C, Hayes P, Birungi J, Anzala O, Garcia M, et al. Standardization of cytokine flow cytometry assays. BMC Immunol 2005;6:13.
30. Human mouse dataset for ground truthing flow cytometry clustering methods (originally generated for SWIFT) Nov 2013. URL http://flowrepository.org/id/FR-FCM-ZZ8F.
31. Murphy R. Automated identification of subpopulations in flow cytometric list mode data using cluster analysis. Cytometry 1985;6:302–309.
32. Naumann U, Luta G, Wand M. The curvHDR method for gating flow cytometry samples. BMC Bioinformatics 2010;11:44.
33. Scheuermann R, Qian Y, Wei C, Sanz I. ImmPort FLOCK: Automated cell population identification in high dimensional flow cytometry data. J Immunol 2009; 182(Meeting Abstracts 1):42–17.
34. Lo K, Hahne F, Brinkman R, Gottardo R. flowClust: A Bioconductor package for automated gating of flow cytometry data. BMC Bioinformatics 2009;10:145.
35. Bradley P, Fayyad U, Reina C. Scaling EM (expectation-maximization) clustering to large databases. Microsoft Research Report, MSR-TR-98-35 1998.
36. Maitra R. Clustering massive datasets with application in software metrics and tomography. Technometrics 2001;43:336–346.