# Artificial Intelligence-Driven Automation of Flow Cytometry Gating

Gabriella Rivera
Applied Data Science
Master's Program
Shiley Marcos School of
Engineering / University of
San Diego
gabriellarivera@sandiego.edu

John Vincent Deniega
Applied Data Science
Master's Program
Shiley Marcos School of
Engineering / University of
San Diego
jdeniega@sandiego.edu

## ABSTRACT

Flow cytometry is a biochemical process that measures the physical and chemical characteristic of cells in a liquid suspension. This method enables the identification and classification of various cellular populations, such as lymphocytes, monocytes, and granulocytes – from Peripheral Blood Mononuclear Cell (PBMC) samples collected from medical subjects. This process, however, requires a human analyst to subjectively interpret the results visually, which introduces human error and inconsistencies across different analysts. Clustering algorithms aim to solve these shortfalls by objectively grouping each cell by their respective cluster which may map to a given population's cellular type for further immunological and clinical diagnostic purposes. Three model types or algorithms were applied to the data: Gaussian mixture models (GMM), K-means clustering, and density-based spatial clustering of applications with Noise (DBSCAN). Each were further tested on three preprocessed datasets: 5% downsampled, principal component analysis (PCA), and PCA with t-distributed stochastic neighbor embedding. DBSCAN with PCA achieved the best balance of cluster compactness and separation as well as the fastest computation time (0.26 minutes) when compared to GMM (5.51 minutes) and K-means clustering (5.10 minutes). This study demonstrates that automated flow cytometry gating can be achieved through clustering algorithms with similar performance of a traditional human analyst even with low-cost hardware and software. Such a cost-effective solution potentially extends the operational life of existing laboratory environments, which would result in increased analytical throughput and a subsequent acceleration of medical and pharmacological discoveries.

## 1 Introduction

According to Brestoff and Frater (2022), flow cytometry is a cost-prohibitive cellular identification and pharmacological discovery process that both requires exceptional capital investment in infrastructure and is inherently difficult in adopting and implementing the latest techniques and technologies in the field. Open-source and low-cost options may serve as a viable bridge technology between capital-intensive investment cycles for laboratories to be able to continue evolving and increasing their analytical throughput without relying on the next hardware upgrade. By using existing mathematically driven principles inherent to artificial intelligence, biochemists may be able to reduce the analytical inputs required to perform routine cellular classification and clustering of PBMCs. This labor-intensive process ultimately serves to evaluate the efficacy of experimental groups

related to clinical trials required in the development of new life-saving medicines.

## 2 Background

Flow cytometry can be a capital-intensive process that requires significant investments in laboratory-grade biomedical equipment, dedicated graphics processing and tensor processing units, expansive random-access memory, and proprietary analytical software licenses. With Flow Cytometry Standard (FCS) files readily available on public repositories and by leveraging open-source and permissive license packages such as Scikit-Learn and Matplotlib to perform computational transformations to FCS data, the team aim to discover cost-effective alternatives to expensive enterprise software licenses that perform flow cytometry analysis, which may result in significant reduction in the barriers to entry in biochemical flow cytometry.

Because cellular populations related to these FCS files number in the millions of records across multiple laboratory readings, this project will place heavy emphasis on dimensionality reduction to meet the constraints of being both cost-effective and hardware-resource efficient. Accomplishing such a feat would result in independent biochemical scientists performing analyses without relying on exceptionally powerful computing hardware resources or costly proprietary enterprise-level software licenses.

### 2.1 Problem Identification and Motivation

As of this publication, flow cytometry gating is a manual process requiring a highly trained biochemist to process and analyze the results of optical scans of cellular assays that may be further augmented by fluorescent substrates. Because of the complex and highly dimensional nature of the data, these scientists rely on a best-practices approach based on their own respective processes and frameworks. Because of the potential variability of these processes and frameworks, the resulting findings from interpreting scan results is dependent on both the breadth and depth of methods of a given supervising scientist, resulting in both an increase in cost of analysis due to human error and omission and a reduction in the consistency of results.

### 2.2 Definition of Objectives

The research team aims to use open-source and publicly available resources from recognized algorithms known in data science to include principal component analysis, t-distributed stochastic neighbor embedding, unsupervised clustering machine learning methods, and FCS data hosted by FlowRepository (2020). Once data are cleaned for noise from scan data, the team aims to train models or machine-learning applications that have potential for value-added analysis relative to that of a typical human biochemist. Upon evaluation, success is generally defined when automated analysis reaches parity with a human analyst of at least 90% classification accuracy of PBMCs toward their respective dendritic cellular type on an unseen test set containing FCS scan data. If this evaluation criterion is not met, further justification would have to be provided. This justification would determine whether the measured degree of accuracy is acceptable relative to the speed of analyses in terms of the ability to identify clusters by silhouette scores and the computing time required to perform clustering using different models and methods.

## 3 Literature Review

Since 2016, a number of academic threads have been studied involving the advancement in flow cytometry, the iteration of methodologies when incorporating machine learning applications on FCS data, and different strategies in how to potentially automate the classification of cellular groups. By 2024, Ng et al. (2024) demonstrated

maturity over an 8-year period that transitions the focus of academic research from the *what* normally seen in earlier works into the *how* with respect to interdisciplinary guidelines as well as quality control and assurance of future deployment of artificial intelligence in flow cytometry.

### 3.1 FlowAI: Automatic and Interactive Anomaly-Discerning Tools For Flow Cytometry Data

FlowAI is a software package for the statistical computing language R, which Monaco et al. (2016) developed as a means to both clean FCS files from anomalies and assess the resulting quality of the cleaned data normalized by the flow rate of a given reading. When flow rate abruptly changes during a scan, the readings may exhibit data inconsistencies. These data inconsistencies are considered anomalous and are discarded from the data set. Using time-series analysis, the resulting data set is broken into trend and cyclical components before being normalized by penalization function measuring absolute deviation of a data point from the median. Monaco et al. placed an emphasis on data quality and anomaly handling, which are crucial considerations to flow cytometry; however, they do not address the next step in automatic gating of cellular types, which is the focus of our research.

### 3.2 An Open-Source Solution for Advanced Imaging Flow Cytometry Data Analysis Using Machine Learning

Hennig et al. (2017) identified the challenges associated with the manual and subjective nature of flow cytometry, resulting in an inconsistent analysis. The given solution has been to use open-source software (i.e., CellProfiler) to use raw image files to identify cell types from a flow cytometer image. Our research shares the open-source idea of being able to leverage existing machine learning algorithms to automatically classify these cell types. Contrasting the team of Hennig et al., the classification differs greatly in their use of visual image data as the basis for classification rather the numerical scan data from fluorescent biological marker excitation that is central to our approach.

### 3.3 Comprehensive Phenotyping of Human Dendritic Cells and Monocytes

Mair and Liechti (2020) identified the potential benefits in using biological markers to identify the phenotypes specific to dendritic cells and monocytes for cellular classification. This research focuses on a potentially more significant subset of biological markers and lineages that aim to identify different cellular categories as a result of their fluorescence excitation scan data more precisely. This work serves as the source data of our project, which uses Python-based machine learning packages for automatic gating. A similar methodology was employed by Hennig et al. (2017), who instead synthesized with visual imagery data using the open-source software, CellProfiler.

### 3.4 Application of Machine Learning for Cytometry Data

Hu et al. (2022) acknowledged the complex challenge of high-dimensional flow cytometry data and the potential for existing machine learning software packages to perform analyses on this type of data. Hu et al. first focuses on dimensionality reduction by principal component analysis and stochastic methods and unsupervised and supervised machine learning methods to predict resulting clinical outcomes such as healthy populations versus diseased populations. Our project aims to build on this research with greater training and tuning toward existing biological knowledge cross-validated across different FCS file scan results.

### 3.5 Recommendations for Using Artificial Intelligence in Clinical Flow Cytometry

Most recently, Ng et al. (2024) focused on a more interdisciplinary approach to using artificial intelligence in flow cytometry with unique considerations for clinical risk management, quality control and assurance, and computational efficiency. This required extensive consideration as to the narrative annotations required for clinical implementation. Though the article is comprehensive across multiple sectors related to flow cytometry and the technical and regulatory nuances required when applying artificial intelligence, Ng et al. only provided general recommendations and guidance for future scientists who wish to leverage this new technology. Relative to our existing work, our research team aims to apply these general recommendations and implement them in an open-source and demonstrable product for flow cytometry automatic gating.

## 4 Methodology

Our platform approach is organized into several key subtopics, starting with data extraction. The FCS files were read using FlowCal. Then, they were transformed into NumPy and Pandas objects to facilitate compatibility with interactive development environments (i.e., Jupyter Notebook and Google Collab) for our purposes. Exploratory data analysis (EDA) was then performed to generate data visualizations and detect outliers, which assisted the data preprocessing. Dimensionality reduction measures were used to simplify the multichannel complexity of the original data before feeding the sets into the various models and machine learning methods. Products for the final launch of the completed flow analysis can be found at the following GitHub repository link at https://github.com/vanguardfox/ADS599.

### 4.1 Data Extraction and Data Structure Conversion

The flow cytometry data set was acquired from FlowRepository, a public database for flow cytometry peer-reviewed experiments. It contains a staining panel from Mair and Leichti's (2020) article that aimed to refine traditional and recently described markers for phenotyping dendritic cells and monocytes—cells that play critical roles in the immune system and are thus indicators of immune response, disease status, and other markers of pharmacological efficacy. The panel data are composed of 23 fluorochrome markers, the time of collection, and forward scatter and side scatter measurements. There were 28 fluorescence channels in total; five of the channel wavelengths were unlabeled due to continuous data acquisition. About 2 million cells were collected per sample, which were reflected in the file sizes ranging between 267 to 405 megabytes for one PBMC FCS file. Additional compensation FCS files were also included in the data set.

FCS data were parsed and ingested as a FlowCal.io.FCSData object, which is derived from a NumPy array. Data were found to be of float big-endian format, which was converted to little-endian format in a NumPy array to facilitate downstream visualization plots and other data transformations for analysis.

The available attributes from the FCS metadata were further parsed to retrieve the channel marker labels using the channel_labels() method. The first three features for forward scatter area (FSC-A) and side scatter area measurements (SSC-A) were corrected and renamed to "FSC-A," "FSC-H," and "SSC-A." The "Time" label was reiterated in the resulting NumPy array. This array was then converted into a Pandas DataFrame for better compatibility with further downstream visualizations. Finally, the formatted DataFrame was saved as a comma-separated values file for

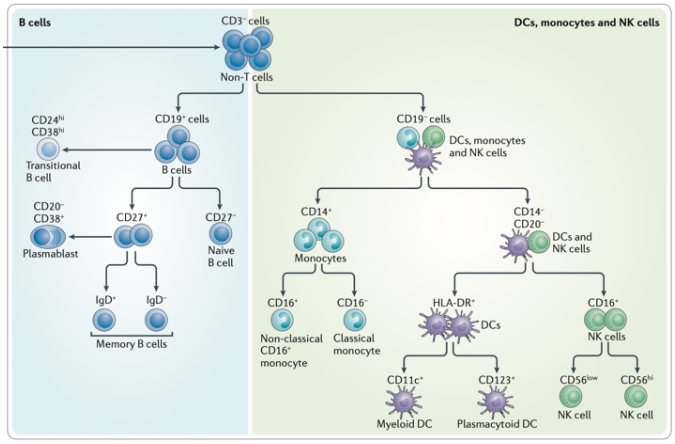
computational compatibility purposes to be used for data preprocessing.

### 4.2 Data Feature Selection

Features were selected based on their relevancy to their ability to provide marker information on dendritic and monocyte cells, which were our target cellular populations. Although 23 fluorescence markers were used to identify specific cell surface proteins, a total of 28 channels were recorded when Mair and Leichti (2020) conducted the original data collection. The five unused channels with missing marker labels were discarded because they were blank, while the remaining 23 markers have known response ranges. To focus on the cellular pathways relevant for dendritic cell phenotyping (see Figure 4.2.1), only those markers and their corresponding lineages relevant to dendritic cells were selected, with the remaining markers discarded as they have no value for our target cell population. Specifically, markers following the lineage through CD45RA, CD3, CD19, CD14, CD20, HLA-DR, CD123, CD11c, and Live Dead UV Blue were retained, along with Time and scattering measurements. This reduced the feature set to 13 with the rest of the lineages and subsequent markers pruned.

**Figure 4.2.1**

*Maecker et al. (2012) Dendritic Cell Lineage*




### 4.3 Flow Fluorescence Compensation

Flow compensation is a crucial process in flow cytometry, especially in high dimensional multichannel experiments. In this case, spectral overlap occurs when the emission spectra of one fluorochrome spill into the detection channels of other markers. Spectral overlap distorts the measurements that consequently lead to inaccurate data interpretation. To address this problem, flow compensation uses a compensation matrix that quantifies the degree of spectral overlap between fluorochrome responses.

The compensation matrix is generated using single-color control samples, where each fluorochrome is measured individually. The control files, labeled "comp_filename," contain spillover data, which shows how much one fluorochrome contaminates the detection channel of another. To create the matrix, we loaded the compensation files and extracted the spillover values for each marker. Then, we applied the compensation matrix to the PBMC data set by adjusting the fluorescence values for each marker based on the spillover data. This correction compensates for spectral interference, ensuring the fluorescence of each marker is accurately

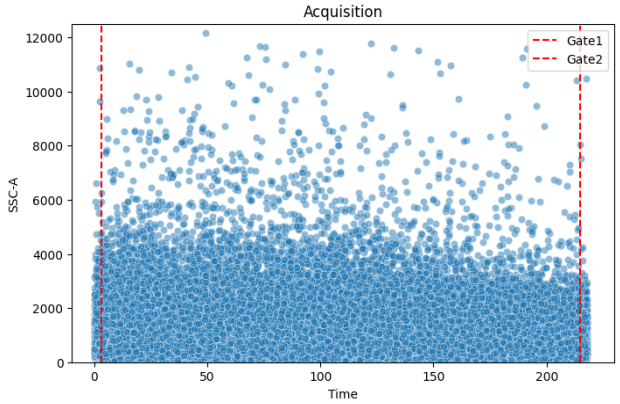
represented without distortion. The final compensated PBMC data set is then fed into EDA for visualization and gating out data noise.

### 4.4 Exploratory Data Analysis

Two-dimensional visualizations were plotted to identify general areas and priorities for cleaning the data set. An SSC versus Time scatter plot (see Figure 4.4.1) was created to identify inconsistencies during data acquisition as cells pass through the inflection point against the detection probe. This plot ensures only cells collected during the stable portion of the sample run are included in the analysis. Figure 4.4.1 further illustrates the gating boundaries, which capture consistent readings across time and help exclude artifacts or outliers caused by fluctuations in the data acquisition process.

**Figure 4.4.1**

*Acquisition Plot*




A histogram of the FSC-A is shown in Figure 4.4.2. FSC-A is used to measure cell size in a given sample mixture. In this case, the resulting plot reveals three distinct peaks, suggesting at least three distinct cellular populations corresponding to the expected cell types in PBMC as lymphocytes, monocytes, and granulocytes (left to right). The red gate is applied to exclude cellular debris, which typically appears as smaller events at the lower end of the FSC-A distribution.

**Figure 4.4.2**

*Cellular Debris Plot*

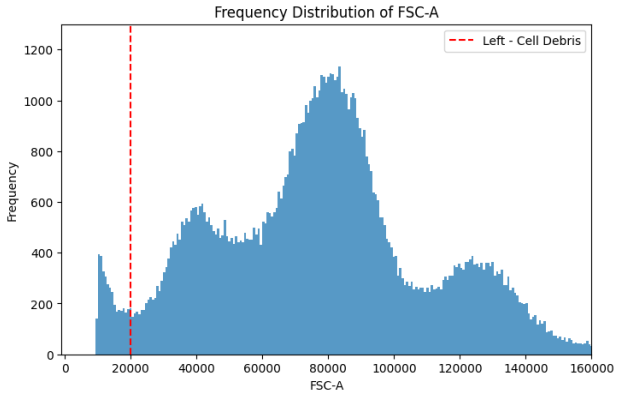
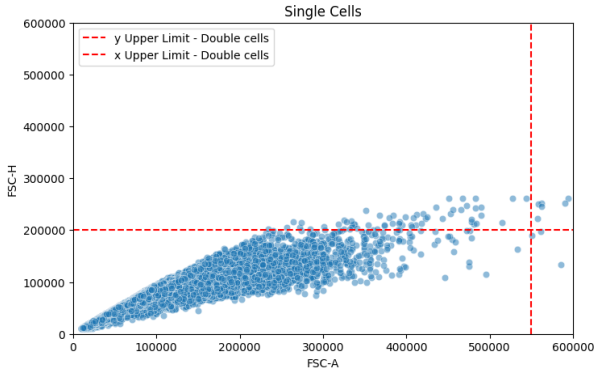



Figure 4.4.3 illustrates the removal of doublets or cell aggregates, which are typically identified by an inconsistent ratio between FSC-A and FSC-H, from the data set. Doublets tend to exhibit a higher FSC-H relative to FSC-A as they are larger due to the presence of two cells but still emit a “tall” scatter signal. In this case, the plot reveals a relatively small population of cell aggregates, identified by the gate on the y-axis. The x-axis limit is set further out to avoid truncating the monocyte population in the SSC-A versus FSC-A plot, ensuring all monocytes are still recalled while removing as many doublets as possible.

**Figure 4.4.3**

*Single Cell Plot*




Finally, the Live/Dead™ UV Blue marker was used to exclude dead cells from the data set. The dye binds to free amines present on the surface and the interior of dead cells because of a broken cellular membrane, resulting in a higher fluorescent signal response than that of a live cell. In contrast, live cells emit a much weaker signal. As shown in Figures 4.4.4 and 4.4.6, live cells are gated to the left of the Live/Dead marker at 1,000,000 relative fluorescence units in both the original and log-transformed Live/Dead channels. This gating strategy cleaned/reduced the number of cell records to 0.8 million viable cells.

**Figure 4.4.4**

*Live Dead Cells: Linear Scale*

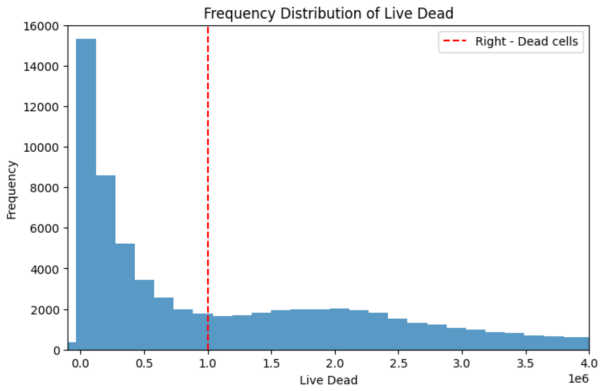
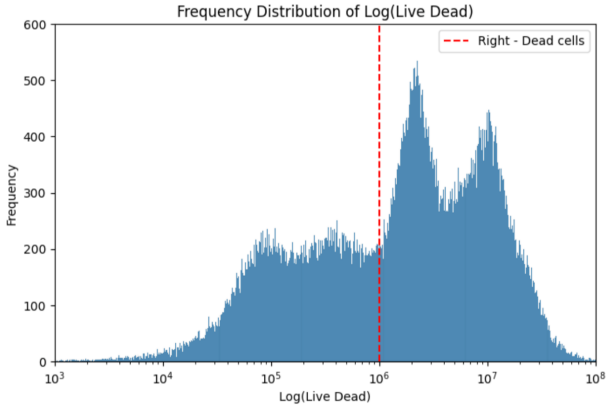



**Figure 4.4.5**

*Live Dead Cells: Logarithmic Scale*




## 4.5 Data Preprocessing

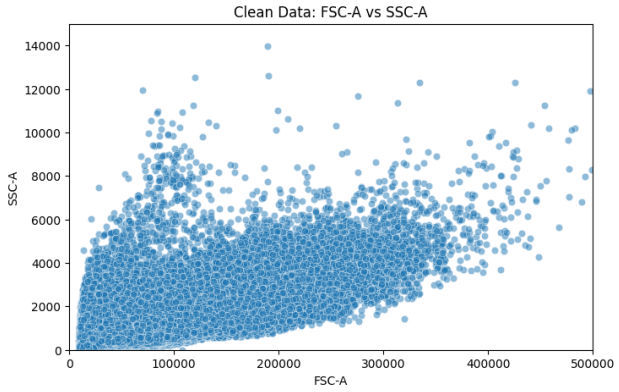
According to FlowRepository (2020), the data obtained for this project were collected from four compensated donors. Because no further information is provided as to the selection criteria of these four donors, and because the sample size of the millions of cells from the files was still constrained to these four donors, there may be bias in the data of unknown magnitude with respect to its representation of the overall global population. Additionally, upon inspection of the FCS file data, there is no personally identifiable information present in the data set that may be used to identify a particular patient or donor in violation of Health Insurance Portability and Accountability Act statutes. Flow cytometry only analyzes clinical data irrespective of the individual who provided the donation upon original data acquisition.

As alluded to during EDA, data cleaning involved several steps to ensure the quality of the data set before passing it through the model classification development. These steps included checking for missing values using the Klib package—with no missing values detected—removing irrelevant cell populations, identifying cellular debris, excluding doublets, and filtering out non-viable cells. Instances where the Time variable was outside the

desired range were excluded by gating on values greater than 3 and less than 215. Additionally, cell populations with abnormal FSC and SSC characteristics were removed as those are indications a cell reading is either debris or doublets. For FSC, cells were selected by gating for readings between 20,000 and 550,000 for FSC-A and for readings less than 200,000 for FSC-H. For SSC, a similar gating strategy was applied, where cells were retained only if SSC-A values were between 110 and 20,000. Finally, dead cells were excluded by applying a threshold on the Live/Dead UV Blue marker which keeps only those with values below $10^6$ RFU. After these steps, the data set was refined to include quality viable cells suitable for model development and clustering. The cleaned data set profile is shown in Figure 4.5.1.

**Figure 4.5.1**

*Clean Data Set Light Scatter Plot*




## 4.6 Dimension Reduction

To efficiently prepare the large datasets for modeling, we applied three different types of data reduction techniques. First, downsampling is used to reduce the data set size by selecting a stratified representative subset using the top ten clusters resulting from KMeans clustering and stratified sampling, which ensured proportional cluster representation. Stratified sampling was picked over random sampling, as it is important to preserve the cells' relative densities along their respective axes, whereas random sampling does not guarantee cell density preservation. The resulting data set is scaled down to be both manageable for training and practical from a cost-effective computing perspective while preserving target clusters.

The next technique was principal component analysis (PCA), which transformed the data into a set of three uncorrelated principal components to capture maximum variance while highlighting the most important features and minimizing noise. This technique also further simplifies the data to three components for computational efficiency purposes.

Finally, a PCA-based t-distributed neighbor embedding (t-SNE) data frame was generated to observe the effect of processing the data further to see if more focus on the local structure of the data could highlight structures across the clusters.

Downsampling, PCA, and t-SNE simplify the data, enhance computing and training efficiency, and optimize for the most informative aspects of the original PBMC data that would otherwise be too computationally intensive to use on its own.

### *4.6.1 Downsampling (Stratified Sampling)*

Downsampling is applied to reduce the data set size while preserving the structure and distribution of the data. In this case, K-means clustering is first used to group the data into 10 clusters. Each cluster is formed by identifying patterns and similarities in the feature space, and the data points in each cluster are assigned a cluster label. Once the data are clustered, stratified sampling is performed to ensure the downsampled data maintain the relative density of each cluster. Specifically, 5% of the samples are randomly selected from each cluster, with larger clusters

being represented proportionally. This process ensures no cluster is overrepresented or underrepresented in the downsampled data. The result is a smaller, more manageable data set that retains the original data's structural properties, which we used to train the models effectively without losing key patterns in the data.

### *4.6.2 Principal Component Analysis*

After selecting features specific to dendritic cell markers and reducing the data set to 12 columns, computationally expensive pairwise comparisons pose a challenge if data are used directly. To address this hardware limitation, PCA was applied to reduce the data's dimensionality. Using the elbow method to determine the optimal number of components, the cumulative explained variance plot (see Figure 4.6.2.1) shows PCA1 captures less than 95% of the variance, whereas the inclusion of PCA2 accounts for 97%. PCA3 is also included to enable 3D visualization of the 13 selected features (see Figure 4.6.2.2), which provides an additional perspective on the data's structure and relationships, helping distinguish patterns that may not be as apparent in lower dimensional representations. By transforming the 13 columns into three principal components, variance is maximized while simplifying the data, improving the efficiency of downstream clustering and modeling steps.

**Figure 4.6.2.1**

*PCA Cumulative Explained Variance Plot*

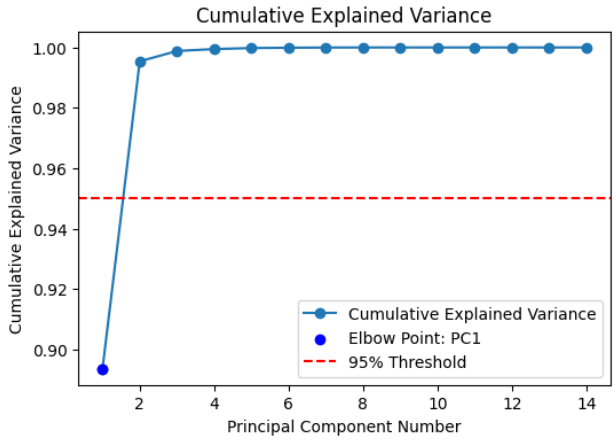
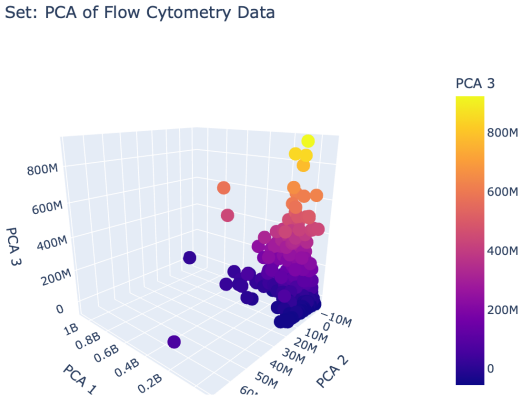



**Figure 4.6.2.2**

*3D PCA Plot of Three PBMC Components*



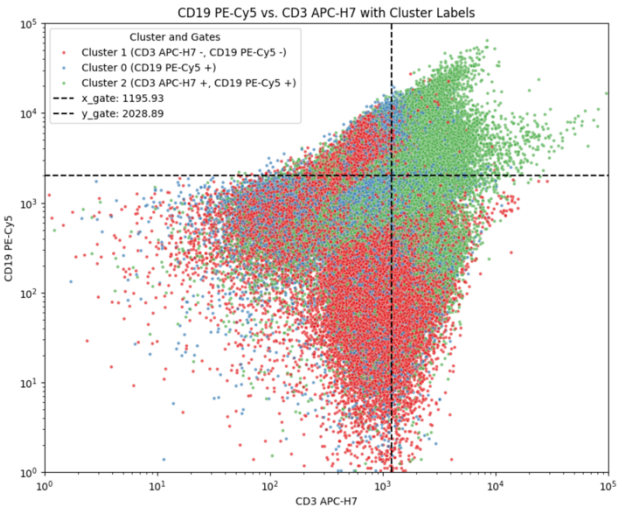

### *4.6.3 T-Distributed Neighbor Embedding*

Using t-SNE helps analysts capture local structures in high-dimensional data. Using three resulting components from PCA, t-SNE aids in being able to visualize different clusters while still being memory-efficient due to this algorithm storing the distances of k-nearest neighbors only rather than of all points in the data. Further, according to Policar (2023), this implementation of t-SNE reduction results in a computational calculation of $O(N)$ and a look-up read time of $O(N \log N)$, which is far more efficient than the pair-wise calculations and storage required from

the original, costing $O(N^2)$. This is a key consideration for our cost-effective solution, as the original constraints imply avoiding the use of high-capital computing resources to achieve a competitive level of analysis. Because about 99% of the variance is still captured with only three components, the loss of 1% of the data for significant memory efficiency directly addresses expected hardware limitations with a computationally intensive algorithm such as t-SNE. Further, t-SNE directly addresses the subjectivity issue that lends to analysts potentially being inconsistent across multiple scatter plots. As such, this method provides clearer and more objective population boundaries for the purposes of gating where different clusters may be isolated for further downstream analysis as shown in Figure 4.6.3.1.

**Figure 4.6.3.1**

*Working t-SNE Gating of CD19 Versus CD3 Markers*




## 4.7 Modeling

Three model types and methods were applied to the data: GMM, DBSCAN, and k-means clustering. These algorithms were chosen for their potential fitness to identify different types of cell population clusters in flow cytometry data. GMM is effective at detecting overlapping or elliptically shaped clusters using a Gaussian kernel, which are expected in biological data sets. K-means is best suited for well-separated, spherical clusters, which are expected in distinct cell populations. DBSCAN can find clusters of arbitrary shape and is robust to outliers. Additionally, DBSCAN does not require the number of clusters to be predefined during modeling.

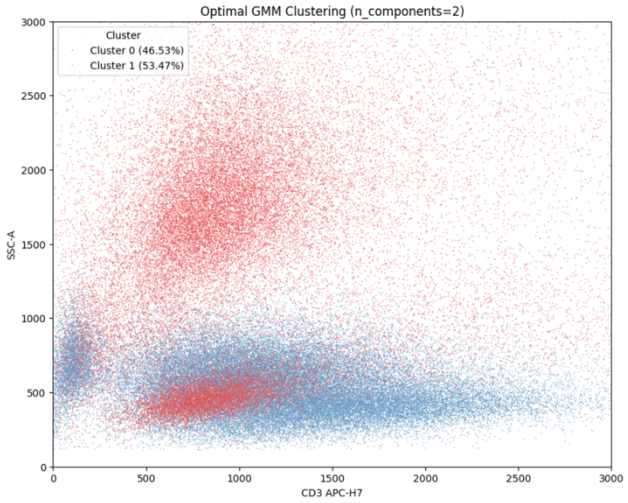
Each model was tested on each of the three preprocessed data sets: Downsampled, PCA, and PCA with t-SNE. All these methods intended to address the cost-effective computing problem such that flow cytometry analysis can be performed without the need for high-capital computing resources and software licenses. As such, computation time and cluster identification were compared to find the most cost-effective solution that provides the next best alternative to more expensive industry options.

### *4.7.1 GMM*

GMM requires the number of components or clusters to be predefined during the model training process. Using domain-knowledge of expected flow cytometry scans, the GMM model was first cross-validated to test models using between two and five clusters with a silhouette score being used to determine the best cluster number. In Figure 4.7.1.1, two clusters were identified as optimal for this PBMC data with a silhouette score of 0.1863.

**Figure 4.7.1.1**

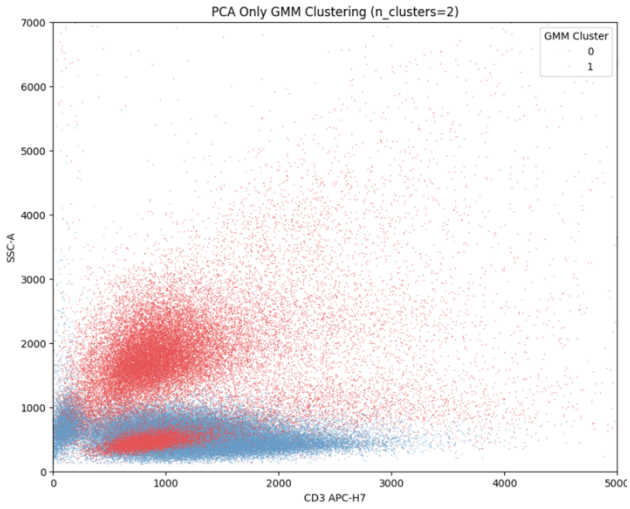
*Downsampled GMM SSC-A Versus CD3 APC-H7*




The PCA transformed data in Figure 4.7.1.2 resulted in a silhouette score of 0.2104, which is an improvement over the downsampled model.

**Figure 4.7.1.2**

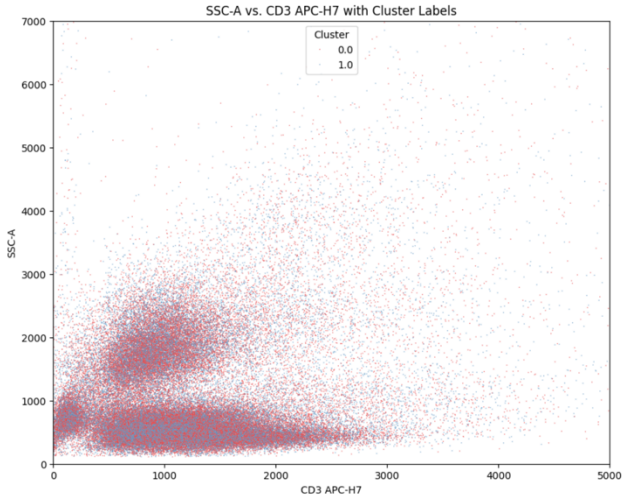
*PCA GMM SSC-A Versus CD3 APC-H7*




The final GMM model plot using PCA and t-SNE in Figure 4.7.1.3 resulted in a silhouette score of 0.3456. Though this score is higher, there does not appear to be much cluster separation from the human eye. Further investigation is warranted to verify if this higher score also validates practical application.

**Figure 4.7.1.3**

*PCA t-SNE GMM SSC-A Versus CD3 APC-H7*



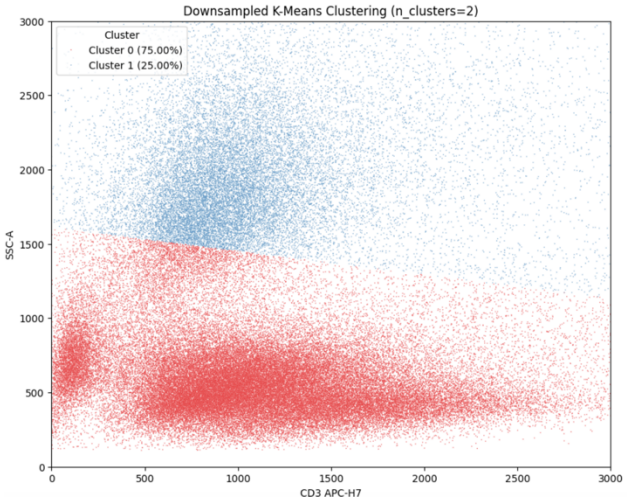

*4.7.2 K-Means Clustering*

Similarly to GMM, the k-means clustering method requires the number of clusters to be predefined. Cross-validation was performed for cluster numbering from two through five with maximum silhouette score as the selection criterion. Cross-validation in this range showed two clusters yielded the highest score, which was used throughout the k-means process.

The downsampled k-means plot shown in Figure 4.7.2.1 resulted in a silhouette score of 0.5456, which is a marked improvement over the GMM models.

**Figure 4.7.2.1**

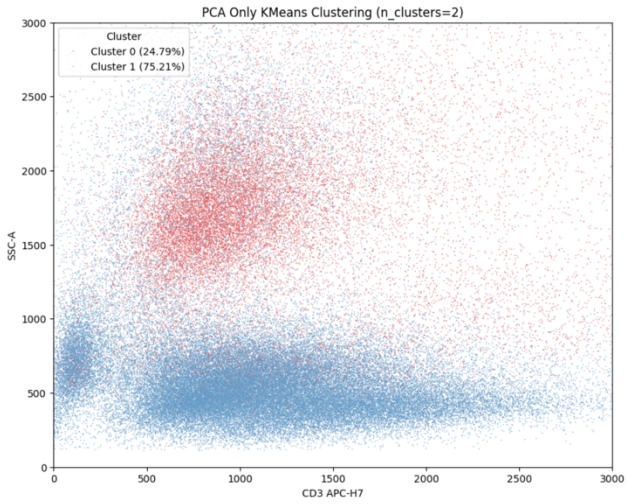
*Downsampled K-Means SSC-A Versus CD3 APC-H7*




The PCA K-means plot shown in Figure 4.7.2.2 resulted in a silhouette score of 0.5502, which is a slight improvement over its sampled counterpart. Notably, the clusters in this plot seem to show less of the hard linear boundary found in the sampled version with Cluster 0 being identified in a more circular shape.

**Figure 4.7.2.2**

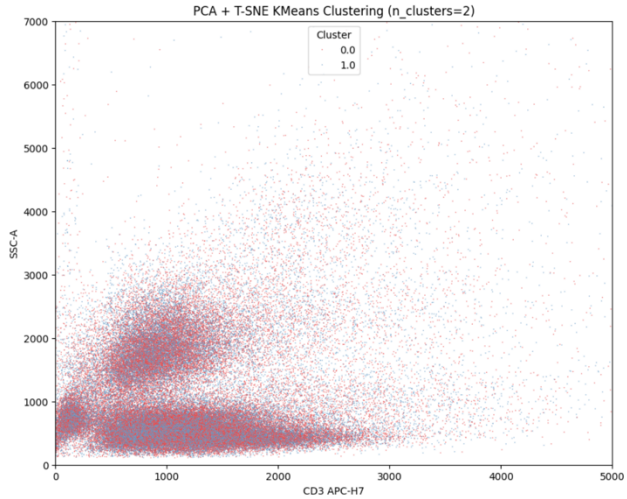
*PCA K-Means SSC-A versus CD3 APC-H7*




The PCA and t-SNE k-means plot shown in Figure 4.7.2.3 resulted in a silhouette score of 0.3455, which is remarkably similar to the GMM model. Cluster separation visually looks similarly poor.

**Figure 4.7.2.3**

*PCA t-SNE K-means SSC-A Versus CD3 APC-H7*



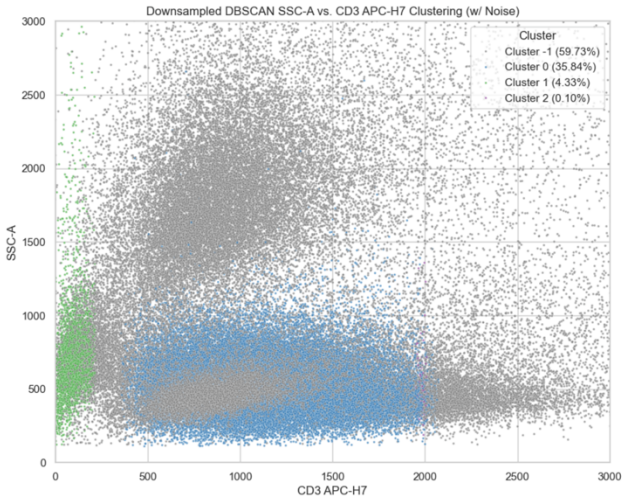

*4.7.3 DBSCAN*

DBSCAN does not have the same requirement to predefine the number of clusters expected as they do for both the GMM and k-means clustering method. Epsilon was set at 30 with minimum samples required for a cluster set at 110. Cross-validation on these two hyperparameters was not feasible from a computationally efficient standpoint, as the cartesian products resulting from performing a grid search resulted in impractical resource use.

The resulting downsampled DBSCAN resulted in a silhouette score of 0.2500, which is markedly lower than most of the models already tested. However, the clusters appear to be well-separated (see Figure 4.7.3.1).

**Figure 4.7.3.1**

*Downsampled DBSCAN SSC-A Versus CD3 APC-H7*




The PCA preprocessed version resulted in the highest silhouette score of 0.5610 for this plot seen in Figure 4.7.3.2. It does appear to accurately isolate one cluster, though it appears a smaller cluster has been grouped with a larger blue one, visually.

**Figure 4.7.3.2**

*PCA DBSCAN SSC-A Versus CD3 APC-H7*

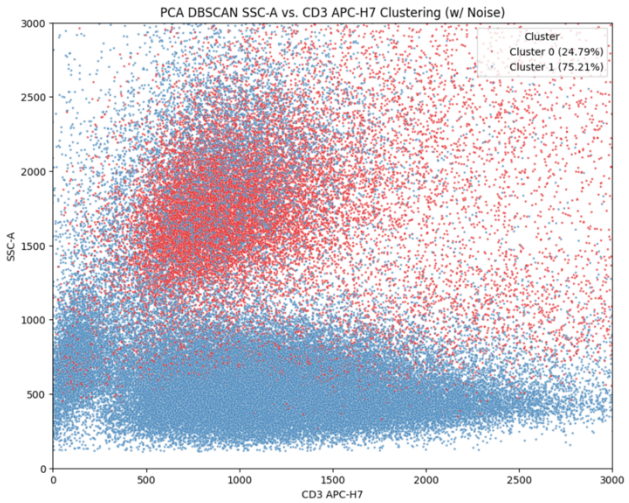
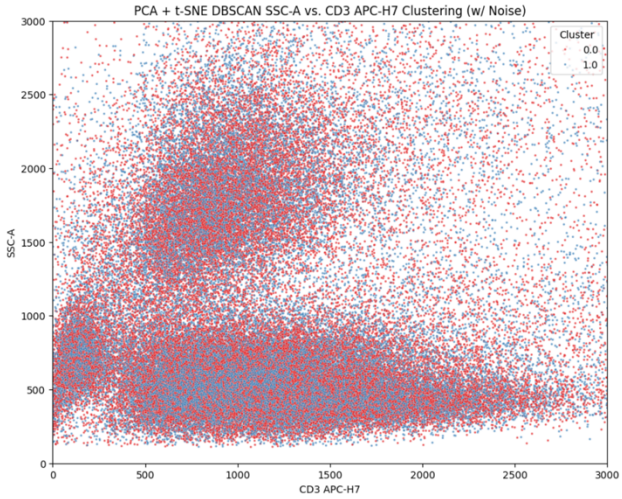



Figure 4.7.3.3 behaved similarly as the previous t-SNE models with a silhouette score of 0.3455. Clusters remain poorly separated on the SSC-A versus CD3 APC-H7 plot throughout these tests.

**Figure 4.7.3.3**

*PCA t-SNE DBSCAN SSC-A Versus CD3 APC-H7*




## 5 Discussion and Evaluation

Upon further inspection of the data, it became apparent our original hypothesis—90% of samples could be accurately classified—was infeasible, given the nature of the data set and how the data were constructed. Because labels were not present from the originally sourced data, and because of the rather subjective nature required to infer a label based on cluster location, cluster axes, and cluster characteristics, the evaluation criteria defaulted to the time required to identify clusters and how well the clusters could be identified by these algorithms.

### 5.1 Silhouette Scores

Silhouette score as a measure of how cohesive and compact a cluster is relative to how well separated they are from other clusters was chosen as an objective and deterministic method to assess cluster validity. The human analogue to this would

be a scientist visually and subjectively identifying dense clusters and drawing lines to separate clusters for further identification.

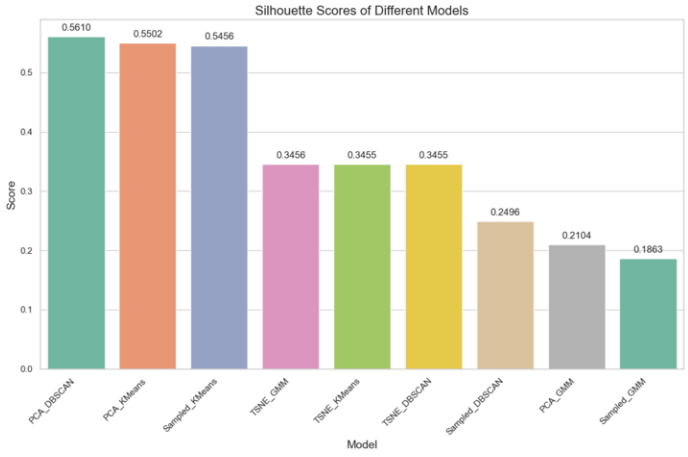
Figure 5.1.1 shows PCA-based DBSCAN performed best in terms of being able to find compact and separated clusters. PCA-based k-means and downsampled k-means performed similarly well and should be considered for their ability to cluster across different axis types.

The t-SNE-based models all performed similarly, which suggests PCA is sufficient to reduce the data for computational-efficiency means and further t-SNE transformation resulted in lower quality clustering.

Downsampling the data set proportionally to 5% of its original size while maintaining relative density did not perform as consistently as expected. Clusters from downsampling performed highly with k-means, though this was to be expected, as the downsampling already preprocessed the original data using k-means clusters as the original basis for reduction, potentially leading to positive bias in the k-means cluster silhouette scores.

**Figure 5.1.1**

*Silhouette Scores by Preprocessing and Model*




## 5.2 Computing Time

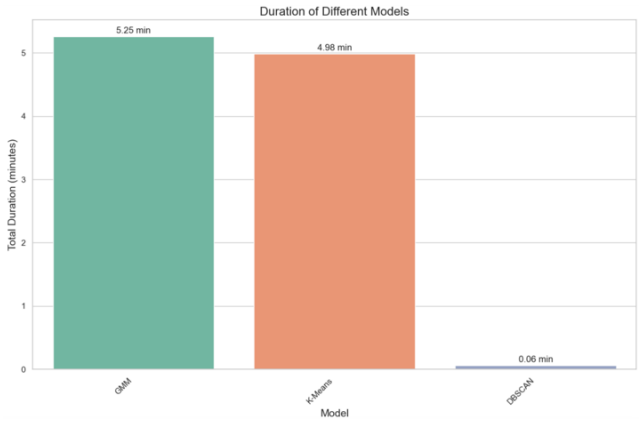
Cluster performance alone cannot determine the success criteria of this experiment. Computing time adds context as to how efficiently the clusters were achieved.

As expected, the GMM and k-means clustering method resulted in significantly more time to compute (see Figure 5.2.1). Because scientists cannot know apriori how many clusters are expected in a given scan data's plot, cross-validation is crucially dynamic and necessary to determine the best fit number of clusters the algorithms suggest, respectively, on their own. This means no one-size-fits-all model can be deployed, as cross-validation to optimize for cluster count must be performed each time a new PBMC file is provided, adding computational requirements in addition to modeling and inferring.

GMM took a total of 5.25 minutes and k-means clustering scored a slightly faster 4.98 minutes. DBSCAN performed the fastest for a total of 0.06 minutes to perform all clustering operations. As a result, DBSCAN performed both the best in terms of silhouette score when using PCA data and the best in terms of time to compute and identify clusters.

**Figure 5.2.1**

*Cluster Time Comparison by Model*



### 5.3 Results Comparison

The trial run by Hennig et al. (2017), as previously described, used raw image data to perform cluster identification. In contrast, our method used flow cytometry sensor readings from fluorescence excitation, which were stored numerically in a PBMC formatted file. Because our original method stems from incorporating the original readings of dozen various cellular markers in our modeling, we are able to perform more robust ad hoc EDA and iterate through different marker lineages and combinations of markers to more accurately identify cluster separation by the axes observed on a two-dimensional plot. For example, SSC-A versus the CD3 marker can be plotted, or the CD3 marker versus the CD19 marker, or any other combination with the uniform computational cost. We can then apply the PCA DBSCAN algorithm to a given combination to assess empirically how well-separated the two markers or scans are clustered.

Hu et al. (2022) incorporated their use of a clinical sample on similar reduction methods such as PCA, t-SNE, and uniform manifold approximation and projection. However, they do not state their hardware capabilities to include any computing, graphics, or tensor processing units as well as memory sizes used for their trial. Our experiment extended this approach by exploring the effectiveness of these similar approaches when limited to either Apple M1 processing units with eight gigabytes of memory or Google Colaboratory's free-tier usage of their version 2.8 TPU with 12.7 gigabytes of memory. PCA and t-SNE were viable using our team's hardware constraints; however, they were not viable (i.e., able to complete computation time within a 20-minute threshold or did not run out of random-access memory) to calculate a silhouette score without downsampling. Uniform manifold approximation and projection was found to be entirely unviable within our hardware constraints. Without the silhouette scores, cross-validation would not have the required metric to compare clustering algorithms. Thus, our work extends on Hu et al. (2022) on what is both viable and of practical use with respect to finding clusters with large flow cytometry data when limited to low-cost or no-cost alternatives.

## 6 Model Conclusions

The team found low-cost or no-cost hardware and software can perform automated flow cytometry clustering within a comparable period and performance threshold to that of a human analyst without necessitating the purchase of high-capital computing equipment or high-cost enterprise software licenses.

DBSCAN and PCA provided the optimal balance of optimizing for cluster compactness and separation while holding to efficient computing constraints that may be found in a capital-restrictive environment.

We also found cross-validation is crucial, given any flow cytometry data can have a wide number of clusters that might only be visible across certain marker dimensions and scan readings. Pretrained models without cross-validation as a preparatory step may risk improperly fitting the data for the structure they already present; therefore, we deem unsupervised methods may yield the greatest performance in this sector of analysis.

Further, it is crucial to acknowledge there is no one-size-fits-all solution available when interpreting flow cytometry data. In a similar complication that human analysts rely on best practices and visual identification of clusters, different hyperparameters will need to be set to train models appropriately for a given set of PBMC data. Though our approach provides a best starting point with publicly available methods and data, our approach only serves as a likely

generalization of clusters across the possible range of scan data in flow cytometry.

## 7 Recommendations

Downsampling and dimensionality reduction appear to be a necessity at the time of this article's print when constrained to no-cost and low-cost hardware and software solutions when performing automated flow cytometry analysis. Unsupervised machine learning methods are advised for future experimentation and exploration due to the inherent problem of not knowing the structure of the cellular data beforehand. If automation is required for other algorithms, cross-validation on different cluster sizes is highly recommended to reflect the structure of the data.

Next steps include refining downsampling and PCA processes that optimize the balance of preserving the local structures of data for clustering purposes and the global structures of data for interpretability when mapping results to markers, which are required for determining the type of cell being observed.

## References


Beckman Coulter. (2022). *Automatic gating*. https://www.beckman.com/flow-cytometry/software/cytobank-premium/learning-center/automatic-gating

Brestoff, J. R., & Frater, J. L. (2022). Contemporary challenges in clinical flow cytometry: Small samples, big data, little time. *Journal of Applied Laboratory Medicine*, *7*(22), 931–944. https://doi.org/10.1093/jalm/jfab176

Cossarizza, A., Chang, H. D., Radbruch, A., Acs, A., Adam, D., Adam-Klages, S., Agace, W. W., Aghaeepour, N., Akdis, M., Allez, M., Almeida, L. N., Alvisi, G., Anderson, G., Andrä, I., Annunziato, F., Anselmo, A., Bacher, P., Baldari, C. T., Bari, S., Barnaba, V., … Zychlinsky, A. (2019). Guidelines for the use of flow cytometry and cell sorting in immunological studies (2nd ed.). *European Journal of Immunology*, *49*(10), 1457–1973. https://doi.org/10.1002/eji.201970107

FlowRepository. (2020). *FlowRepository ID FR-FCM-Z32U.* http://flowrepository.org/experiments/3166/download_ziped_files

Hennig, H., Rees, P., Blasi, T., Kamentsky, L., Hung, J., Dao, D., Carpenter, A. E., & Filby, A. (2016). An open-source solution for advanced imaging flow cytometry data analysis using machine learning. *Methods*, *112*(2017), 201–210. https://doi.org/10.1016/j.ymeth.2016.08.018

Hu, Z., Bhattacharya, S., & Butte, A. J. (2022). Application of machine learning for cytometry data. *Frontiers in Immunology*, *12*, Article 787574. https://doi.org/10.3389/fimmu.2021.787574

Lee, J. A., Spidlen, J., Boyce, K., Cai, J., Crosbie, N., Dalphin, M., Furlong, J., Gasparetto, M., Goldberg, M., Goralczyk, E. M., Hyun, B., Jansen, K., Kollmann, T., Kong, M., Leif, R., McWeeney, S., Moloshok, T. D., Moore, W., Nolan, G., Nolan, J., … Brinkman, R. R. (2008). MIFlowCyt: The minimum information about a flow cytometry experiment. *Cytometry. Part A: The Journal of the International Society for Analytical Cytology*, *73*(10), 926–930. https://doi.org/10.1002/cyto.a.20623

Maecker, H. T., McCoy, J. P., & Nussenblatt, R. (2012). Standardizing immunophenotyping for the Human Immunology Project. *Nature Reviews Immunology*, *12*(3), 191–200. https://doi.org/10.1038/nri3158

Mair, F., & Leichti, T. (2020). Comprehensive phenotyping of human dendritic cells and

monocytes. *Journal of Quantitative Cell Science*, *99*(3), 231–242. https://doi.org/10.1002/cyto.a.24269

Monaco, G., Chen, H., Poidinger, M., Chen, J., Magalhaes, J., & Larbi, A. (2016). FlowAI: Automatic and interactive anomaly discerning tools for flow cytometry data. *Bioinformatics*, *32*(16), 2473–2480. https://doi.org/10.1093/bioinformatics/btw191

Ng, D. P., Simonson, P. D., Tarnok, A., Lucas, F., Kern, W., Rolf, N., Bogdanoski, G., Green, C., Brinkman, R. R., & Czechowska, K. (2024). Recommendations for using artificial intelligence in clinical flow cytometry. *Cytometry Part B: Clinical Cytometry*, *106*(4), 228–238. https://doi.org/10.1002/cyto.b.22166

Policar, P. (2023). *How t-SNE works*. https://opentsne.readthedocs.io/en/latest/tsne_algorithm.html

Spidlen, J., Breuer, K., Rosenberg, C., Kotecha, N., & Brinkman, R. R. (2012). FlowRepository - A resource of annotated flow cytometry datasets associated with peer-reviewed publications. *Cytometry Part A*, *81*(9), 727–731. https://doi.org/10.1002/cyto.a.22106