Computational Techniques for Flow Cytometry

The Application for Automated Analysis of Innate Immune Response Flow Cytometry Data

by

Parisa Shooshtari

B.Sc., Amirkabir University of Technology, 2004
M.Sc., Sharif University of Technology, 2006

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

in

The Faculty of Graduate Studies

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

April 2012

© Parisa Shooshtari 2012
Abstract

Flow cytometry (FCM) is a technique for measuring physical, chemical and biological characteristics of individual cells. Recent advances in FCM have provided researchers with the facility to improve their understanding of the tremendously complex immune system. However, the technology is hampered by current manual analysis methodologies. In this thesis, I developed computational methods for the automated analysis of immune response FCM data to address this bottleneck. I hypothesized that highly accurate results could be obtained through learning from the patterns that a biology expert applies when doing the analysis manually.

In FCM data analysis, it is often desirable to identify homogeneous sub-sets of cells within a sample. Traditionally, this is done through manual gating, a procedure that can be subjective and time-consuming. I developed SamSPECTRAL, an automated spectral-based clustering algorithm to identify FCM cell populations of any shape, size and distribution while addressing the drawbacks of manual gating. A particularly significant achievement of SamSPECTRAL was its successful performance in finding rare cell populations. Similarly, in most FCM applications, it is required to match similar cell populations between different FCM samples. I developed a novel learning-based cluster matching method that incorporates domain expert knowledge to find the best matches of target populations among all clusters generated by a clustering algorithm.

Immunophenotyping of immune cells and measuring cytokine responses are two main components of immune response FCM data analysis. I combined the SamSPECTRAL algorithm and cluster matching to perform automated immunophenotyping. I also devised a method to measure cytokine responses automatically. After developing computational methods for each of the above analysis components separately, I organized them into a semi-automated pipeline, so they all work together as a unified package. My experiments on 216 FCM samples confirmed that my semi-automated pipeline can reproduce manual analysis results highly accurately both for immunophenotyping and measuring cytokine responses.

My other main contributions were correlation analysis of intracellular
and secreted cytokines, and developing a formula called GiMFI to improve measuring functional response of cytokine-producing cells using flow cytometry assay.
Preface

This thesis represents a culmination of work and learning that has taken place over a period of four years in the Terry Fox Laboratory, BC Cancer Research Center (2008 - 2011). Dr. Brinkman’s Laboratory provided the unique opportunity to establish collaborations with researchers from a variety of disciplines ranging from computational biology and mathematics to biology and immunology. The present research would not have been possible without the exquisite contributions from my supervisors Dr. Ryan Brinkman and Dr. Arvind Gupta, as well as my colleagues at Brinkman’s Lab and Kollmann’s Lab.

SamSPECTRAL clustering (Chapter 2) is based on collaborative work between myself, Habil Zare, Dr. Arvind Gupta, and Dr. Ryan Brinkman. A version of this chapter was published in BMC Bioinformatics as an original article entitled: “Data reduction for spectral clustering to analyze high throughput flow cytometry data”, BMC Bioinformatics 2010,11:403 [179]. As mentioned in this paper, the authors wished to appreciate the equal contribution of the first two authors, Habil Zare and myself, and that they should be regarded as joint first authors. Habil Zare (PhD student at Brinkman’s Lab) proposed the idea of using spectral clustering for identification of flow cytometry cell populations, and I proposed the idea of using nonuniform sampling for spectral clustering as an efficient data reduction scheme to solve the computational issue of spectral clustering when applied to huge high-throughput datasets. I developed the main idea on how flow cytometry data should be sampled non-uniformly in the data space such that no biologically important information is lost after sampling, and Habil Zare designed and implemented the faithful sampling algorithm based on this idea. Habil Zare developed the method for computing similarity between communities. Habil Zare and I jointly worked and equally contributed to the method for estimating the number of clusters (50-50%), and post-processing steps (50-50%). Habil Zare and I performed the experiments (75% of the experiments were performed by Habil Zare, and 25% by myself). To be more specific, Habil Zare ran SamSPECTRAL on the GvHD, Telomere, and Viability data examples, and I performed the experiments for
finding rare cell populations in the Stem Cell dataset. I also experimentally verified the stability of the SamSPECTRAL algorithm (100%). Habil Zare (55%) and I (45%) prepared and wrote the manuscript together. Dr. Ryan Brinkman provided the data and computing facilities. Dr. Arvind Gupta studied the convergence of faithful sampling. Dr. Arvind Gupta and Dr. Ryan Brinkman supervised the project. All authors reviewed, edited and approved the final manuscript.

The results of FlowCAP competition in Chapter 2 were submitted to a journal for consideration for publication. My contribution as an author in this manuscript was to run the SamSPECTRAL clustering algorithm on five datasets that were made available for this competition. SamSPECTRAL results on these datasets were jointly generated by Pillip Mah (Undergraduate student at UBC), Habil Zare, and myself (contributed almost equally). Nima Aghaeepour (PhD student at Brinkman’s Lab) computed F-measure and compared the methods. Other authors of this manuscript either supervised the project or participated in this competition by their own previously developed clustering algorithms.

I designed and implemented a learning-based cluster matching as a successful method to accurately identify and match target cell populations across different flow cytometry samples (100%). A version of my learning-based cluster matching method and my semi-automated analysis pipeline (Chapter 3) is in preparation for submission to a scientific journal as an original article entitled: “Semi-automated pipeline for analysis of innate immune response flow cytometry data: a proper substitute for manual analysis”. This work is to be submitted as a collaborative work for peer reviewed publication between myself, Dr. Arvind Gupta, Dr. Tobias Kollmann, and Dr. Ryan Brinkman. I am the first author on this paper, and my contributions to it can be summarized as (1) developing a learning-based cluster matching method, (2) designing a full semi-automated pipeline for analysis of innate immune response flow cytometry data, (3) applying the automated methods on the full dataset, (4) comparing the automated results against manual results, and (5) preparing and writing all parts of the manuscript. The biological datasets and manual identification of cell populations and cytokine-positive cells on this dataset was performed by Dr. Tobias Kollmann’s lab members, including Darren Blinkie, Kevin Ho, and Edgardo Fortuno III, and had already been published in [45, 102]. Dr. Arvind Gupta, Dr. Tobias Kollmann, and Dr. Ryan Brinkman supervised the project. Dr. Tobias Kollmann provided the FCM datasets and manual gating results. All authors read and edited the manuscript.

The correlation analysis of intracellular and secreted cytokines that I
present in Chapter 4 was published in Cytometry A as an original article entitled: “Correlation Analysis of Intracellular and Secreted Cytokines via the Generalized Integrated Mean Fluorescence Intensity”, Cytometry A, 2010, 77A: 873-880. This was a collaborative work between myself, Edgardo S. Fortuno III, Darren Blimkie, Miao Yu, Arvind Gupta, Tobias R. Kollmann, and Ryan R. Brinkman. As mentioned in this paper, Edgardo and I contributed equally to this work. Tobias R. Kollmann and Ryan R. Brinkman also made equal contributions. I was the lead author on this paper, and my contributions to it are as follows: I fully analyzed (100%) the correlation between the intracellular cytokines, and the secreted cytokines using two approaches: (a) for the whole blood cells of PBMC samples all together, and (b) for different antigen-presenting cell populations separately. I also developed the GiMFI formula (100%), a quantitatively more correliative mathematical approach for calculating the functional response of cytokine-producing cells. Edgardo Fortuno III (15%), Darren Blimkie (5%) and I (80%) jointly wrote the manuscript. Edgardo Fortuno III and Darren Blimkie worked on manual gating of antigen-presenting cell populations, and manual identification of cytokine-positive cells (100%), as well as providing Luminex results (100%). Miao Yu extracted the manual gates information from the FlowJo software and prepared it for further experiments on correlation analysis that was performed by myself. Dr. Tobias Kollmann provided the datasets. Dr. Arvind Gupta, Dr. Tobias Kollmann, and Dr. Ryan Brinkman supervised this project, and edited the manuscript. All authors reviewed, edited, and approved the final manuscript.
# Table of Contents

Abstract ......................................................... ii

Preface ......................................................... iv

Table of Contents ............................................. vii

List of Tables ................................................ xii

List of Figures ................................................ xiii

List of Abbreviations ..........................................xiv

Acknowledgements ............................................. xvi

Dedication ....................................................... xvii

1 Introduction ................................................. 1
   1.1 Flow Cytometry: Technology and Application .......... 1
   1.2 Analysis of Flow Cytometry Data ....................... 2
      1.2.1 Manual Analysis .................................. 2
      1.2.2 Automated Analysis ................................ 3
   1.3 Automated Gating for Identification of Cell Populations ... 6
      1.3.1 Model-Based Clustering Algorithms ............... 7
      1.3.2 K-means for Clustering Flow Cytometry Data .... 8
      1.3.3 Density-Based Clustering Algorithms ........... 9
      1.3.4 Other Automated Gating Methods ............... 11
      1.3.5 Flow Cytometry: Critical Assessment of Population Identification Methods (FlowCAP) Project ........ 11
   1.4 Assigning Labels to Cell Populations .................. 12
      1.4.1 Metaclustering ................................... 13
   1.5 Analysis of Innate Immune Response Flow Cytometry Data 13
      1.5.1 Introduction to Innate Immunity .................. 13
Table of Contents

1.5.2 The Study of Neonate Innate Immunity Using Flow Cytometry ........................................ 15
1.5.3 Measuring Functional Response of Cytokine Producing Cell Populations .......................... 17
1.6 My Contributions to the Automated Analysis of FCM Data .............................................. 17
   1.6.1 SamSPECTRAL Clustering Algorithm for Automated Gating of Flow Cytometry Data .... 18
   1.6.2 Learning-Based Cluster Matching Method ............................................................... 18
   1.6.3 Semi-Automated Analysis Pipeline .............................................................................. 19
   1.6.4 Correlation Analysis of Intracellular and Secreted Cytokines Using GiMFI Formula .... 19

2 Data Reduction for Spectral Clustering to Analyze High Throughput Flow Cytometry Data .......... 20
   2.1 Background .................................................................................................................. 20
      2.1.1 Problem Statement ............................................................................................... 21
      2.1.2 Our Approach ...................................................................................................... 21
   2.2 Methods ....................................................................................................................... 21
      2.2.1 Spectral Clustering ............................................................................................... 22
      2.2.2 Data Reduction Scheme ....................................................................................... 22
      2.2.3 Similarity Matrix ................................................................................................... 25
      2.2.4 Number of Clusters ............................................................................................. 27
      2.2.5 Combining Clusters ............................................................................................. 27
      2.2.6 Overview of SamSPECTRAL Algorithm ............................................................... 28
      2.2.7 Modified Markov Clustering Algorithm (MCL) .................................................. 30
      2.2.8 FLAME and flowMerge: State of the Art Model-Based Clustering Algorithms for FCM Data ................................................................. 31
      2.2.9 Adjusting Parameters ........................................................................................... 33
      2.2.10 FlowCAP-I Competition ..................................................................................... 34
      2.2.11 Datasets .............................................................................................................. 35
   2.3 Results ............................................................................................................................ 37
      2.3.1 Overlapping Populations ....................................................................................... 38
      2.3.2 Subpopulations of a Population ............................................................................. 38
      2.3.3 Non-elliptical Shaped Populations ................................................................ ...... 38
      2.3.4 Low Density Populations Close to Dense Populations ....................................... 38
      2.3.5 Rare Populations ................................................................................................... 42
      2.3.6 SamSPECTRAL with MCL .................................................................................. 45
      2.3.7 FlowCAP-I Competition ...................................................................................... 45
   2.4 Discussion ....................................................................................................................... 51
   2.5 Conclusions .................................................................................................................... 53
Table of Contents

3 Semi-Automated Pipeline for Analysis of Innate Immune Response Flow Cytometry Data .......................... 55
  3.1 Background ......................................................... 55
  3.2 Data and Wet Lab Description ................................. 57
  3.3 Manual vs. Semi-Automated Analysis .......................... 58
  3.4 Data Preparation and Preprocessing ............................ 58
    3.4.1 Removing Dead Cells and Debris .......................... 58
    3.4.2 Compensation and Transformation .......................... 60
    3.4.3 Normalization .............................................. 60
  3.5 Automated Gating Using SamSPECTRAL Clustering Algorithm .................................................. 60
  3.6 Learning-Based Cluster Matching for Labeling APC Populations .............................................. 62
    3.6.1 Definitions .................................................. 62
    3.6.2 Problem Statement ......................................... 63
    3.6.3 My Approach ................................................. 64
    3.6.4 Incorporating Biology Expert Knowledge ................. 64
    3.6.5 Learning From Training Samples ........................... 66
    3.6.6 Likelihood Ratio Measurement .............................. 67
    3.6.7 Scoring the Clusters to Find the Best Match of a Target Population ................................... 68
  3.7 Validating Locations of Semi-Automatically Identified Cell Populations ........................................ 69
  3.8 Measuring Functional Response of Cytokine Positive Cells ................................................... 70
  3.9 Comparing Manual and Semi-Automated Analysis .......... 71
    3.9.1 Comparison of the Manual and Semi-Automated Identifications of APC Populations .................... 71
    3.9.2 Comparison of Manual and Automated Functional Response Measurements ................................ 75
  3.10 Impact of Training Sample Size on Cluster Matching Results .................................................. 76
  3.11 Discussion ....................................................... 82
  3.12 Conclusion and Future Work .................................... 86

4 Correlation Analysis of Intracellular and Secreted Cytokines via the Generalized Integrated Mean Fluorescence Intensity ......................................................... 88
  4.1 Background ....................................................... 88
  4.2 Methods .......................................................... 89
    4.2.1 Data and Wet Lab Description .............................. 89
    4.2.2 Correlation Analysis of iMFI and Culture Supernatant Response ........................................... 90
Table of Contents

4.2.3 Automated Identification of All Cytokine-Positive Cells of PBMC ........................................... 91
4.2.4 Manual Identification of Antigen-Presenting Cell Populations and Detection of Cytokine-Positive Cells for Different APC Populations ........................................... 91
4.2.5 Generalized iMFI (GiMFI) ........................................... 92

4.3 Results ........................................... 93
4.3.1 Correlation Analysis of Whole PBMC Live Cells ........................................... 93
4.3.2 Correlation Analysis of Antigen-Presenting Cell Populations ........................................... 94
4.3.3 Parameter Estimation for the Generalized iMFI (GiMFI) Formula ........................................... 97

4.4 Discussion ........................................... 101

5 Summary, Conclusions, and Future Work ........................................... 105
5.1 SamSPECTRAL Clustering for Identification of Cell Populations ........................................... 109
5.2 Learning-Based Cluster Matching for Labeling Cell Populations ........................................... 110
5.3 Semi-Automated Pipeline for Analysis of Flow Cytometry Data ........................................... 114
5.4 Correlation Analysis of Intracellular and Secreted Cytokines ........................................... 115

Bibliography ........................................... 116

Appendices

Appendix A: Testing Stability of SamSPECTRAL Results ........................................... 134
Appendix B: Performance of SamSPECTRAL on Synthetic Data ........................................... 135
Appendix C: Comparing Uniform Sampling Against Faithful Sampling ........................................... 137
Appendix D: FlowCAP-I Competition ........................................... 139
Appendix E: Flow Cytometry Datasets Used in This Study ........................................... 141
Table of Contents

Appendix F: Finding Groups of Samples With Similar Monocytes Features ........................................... 143

Appendix G: Time Complexity of Learning-Based Cluster Matching ......................................................... 148

Appendix H: SamSPECTRAL Package in R BioConductor ................................................................. 150

Appendix I: Additional Information on SamSPECTRAL ................................................................. 160
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Summarizing the comparison between performance of</td>
<td>46</td>
</tr>
<tr>
<td>2.2</td>
<td>Results of the FlowCAP-I challenges</td>
<td>49</td>
</tr>
<tr>
<td>3.1</td>
<td>Target populations features that are used in my cluster</td>
<td>65</td>
</tr>
<tr>
<td>3.2</td>
<td>Percentile ($\beta$) of the empirical cumulative distribution</td>
<td>70</td>
</tr>
<tr>
<td>3.3</td>
<td>Pearson correlation coefficients between the manually and</td>
<td>72</td>
</tr>
<tr>
<td>3.4</td>
<td>The upper bound of the differences ($d$) between manually and</td>
<td>73</td>
</tr>
<tr>
<td>3.5</td>
<td>Pearson correlation coefficients between percentage/MFI of</td>
<td>75</td>
</tr>
<tr>
<td>3.6</td>
<td>Difference ($d$) between manually and automatically</td>
<td>76</td>
</tr>
<tr>
<td>3.7</td>
<td>Summarizing sample size results for Algorithm A</td>
<td>78</td>
</tr>
<tr>
<td>3.8</td>
<td>Summarizing sample size results for Algorithm B</td>
<td>79</td>
</tr>
<tr>
<td>4.1</td>
<td>Correlation between iMFI values and culture supernatant</td>
<td>96</td>
</tr>
<tr>
<td>4.2</td>
<td>$P$ values obtained by applying the Wilcoxon test with the null</td>
<td>98</td>
</tr>
<tr>
<td>A.1</td>
<td>Summary of the description of the FlowCAP-I datasets.</td>
<td>140</td>
</tr>
<tr>
<td>A.2</td>
<td>Flow cytometry datasets</td>
<td>142</td>
</tr>
<tr>
<td>A.3</td>
<td>Identification of rare cell population in stem cell dataset using</td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>different algorithms</td>
<td></td>
</tr>
</tbody>
</table>
# List of Figures

2.1 Running spectral clustering is impractical on data that . . . . 23  
2.2 Faithful sampling . . . . . . . . . . . . . . . . . . . . . . . . . . 24  
2.3 Defining the similarity between two communities and . . . . . 26  
2.4 Comparative clustering of the telomere dataset . . . . . . . . 39  
2.5 Comparative clustering of dead cells (PI positive) and . . . . . 40  
2.6 Comparative clustering of the GvHD dataset . . . . . . . . . . . 41  
2.7 Comparative identification of a low density population . . . . . 43  
2.8 Rare population in the stem cell data set . . . . . . . . . . . . 44

3.1 Manual identification of antigen-presenting cell . . . . . . . . 59  
3.2 Boxplots of (A) sensitivity and (B) specificity of four APC . . 74  
3.3 F-Measure vs. sample size for groups obtained by Algorithm A 80  
3.4 F-Measure vs. sample size for groups obtained by Algorithm B 81

4.1 Scatter plots of the amount of secreted IL-12p40 measured . . 93  
4.2 Boxplots of correlation coefficients between culture . . . . . 94  
4.3 (A-C) Boxplots of Pearson correlation coefficients for APC  
populations . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 95  
4.4 Example calculation of an optimum $\alpha$ which maximizes . . 95  
4.5 Distribution of $\rho_{\text{MFI}}$ (correlation coefficients of culture . . 100

A.1 Performance of SamSPECTRAL on synthetic data . . . . . . 135  
A.2 Comparing uniform sampling with faithful sampling . . . . . 138

A.3 The average density plots and the kernel probability density . 144  
A.4 Boxplots of frequency of manually gated monocytes for different 145  
A.5 The average density plots and the kernel probability density . 147
## List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>APC</td>
<td>Antigen Presenting Cells</td>
</tr>
<tr>
<td>BD</td>
<td>Becton-Dickinson</td>
</tr>
<tr>
<td>BFA</td>
<td>Brefeldin A</td>
</tr>
<tr>
<td>BIC</td>
<td>Bayes’ Information Criterion</td>
</tr>
<tr>
<td>cDC</td>
<td>Conventional Dendritic Cells</td>
</tr>
<tr>
<td>CF</td>
<td>Cytometric Fingerprinting</td>
</tr>
<tr>
<td>CFC</td>
<td>Cytokine Flow Cytometry</td>
</tr>
<tr>
<td>CMI</td>
<td>Cell-Mediated Immunity</td>
</tr>
<tr>
<td>CoP</td>
<td>Correlates of Protection</td>
</tr>
<tr>
<td>CPAS</td>
<td>Computational Proteomics Analysis System</td>
</tr>
<tr>
<td>DC</td>
<td>Dendritic Cells</td>
</tr>
<tr>
<td>DLBCL</td>
<td>Diffuse Large B-cell Lymphoma</td>
</tr>
<tr>
<td>DNA</td>
<td>Deoxyribonucleic Acid</td>
</tr>
<tr>
<td>ECDF</td>
<td>Empirical Cumulative Distribution Function</td>
</tr>
<tr>
<td>ELISA</td>
<td>Enzyme-Linked Immunosorbent Assay</td>
</tr>
<tr>
<td>FACS</td>
<td>Fluorescent Activated Cell Sorting</td>
</tr>
<tr>
<td>FCM</td>
<td>Flow Cytometry</td>
</tr>
<tr>
<td>FlowCAP</td>
<td>Critical Assessment of Population Identification Methods</td>
</tr>
<tr>
<td>FSC</td>
<td>Forward Scatter</td>
</tr>
<tr>
<td>GiMFI</td>
<td>Generalized Integrated Mean Fluorescence Intensity</td>
</tr>
<tr>
<td>GPU</td>
<td>Graphics Processing Units</td>
</tr>
<tr>
<td>GvHD</td>
<td>Graft Versus Host Disease</td>
</tr>
<tr>
<td>HSCT</td>
<td>Hematopoietic Stem Cell Transplant</td>
</tr>
<tr>
<td>ICS</td>
<td>Intracellular Cytokine Staining</td>
</tr>
<tr>
<td>IFN-α</td>
<td>Interferon α</td>
</tr>
<tr>
<td>IL-12</td>
<td>Interleukin 12</td>
</tr>
<tr>
<td>IL-6</td>
<td>Interleukin 6</td>
</tr>
<tr>
<td>iMFI</td>
<td>Integrated Mean Fluorescence Intensity</td>
</tr>
<tr>
<td>MC</td>
<td>Monocytes</td>
</tr>
<tr>
<td>MCL</td>
<td>Markov Clustering</td>
</tr>
<tr>
<td>MFI</td>
<td>Mean Fluorescent Intensity</td>
</tr>
<tr>
<td>MHC</td>
<td>Major Histocompatibility Complex</td>
</tr>
</tbody>
</table>
### List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ND</td>
<td>Normal Donors</td>
</tr>
<tr>
<td>NIH</td>
<td>National Institute of Health</td>
</tr>
<tr>
<td>PAM</td>
<td>Partitioning Around Medoids</td>
</tr>
<tr>
<td>PBMC</td>
<td>Peripheral Blood Mononuclear Cells</td>
</tr>
<tr>
<td>pDC</td>
<td>Plasmacytoid Dendritic Cells</td>
</tr>
<tr>
<td>PI</td>
<td>Propidium Iodide</td>
</tr>
<tr>
<td>pMHC</td>
<td>Peptide Major Histocompatibility Complex</td>
</tr>
<tr>
<td>QA</td>
<td>Quality Assessment</td>
</tr>
<tr>
<td>SSC</td>
<td>Side Scatter</td>
</tr>
<tr>
<td>TLR</td>
<td>Toll-Like Receptors</td>
</tr>
<tr>
<td>TNF-α</td>
<td>Tumor Necrosis Factor α</td>
</tr>
<tr>
<td>WNV</td>
<td>Symptomatic West Nile Virus</td>
</tr>
</tbody>
</table>
Acknowledgements

I would like to thank my supervisor, Dr. Arvind Gupta, for his invaluable scientific advices and financial support during my Ph.D. program. I would also like to thank my co-supervisor, Dr. Ryan Brinkman, for providing a research environment in which I could conveniently conduct my studies. I am also thankful for his scientific advices, and his invaluable comments and feedbacks on my papers, presentation and the current dissertation. I would like to thank Dr. Tobias Kollmann for his close collaborations, providing the data, and sharing interesting biology and bioinformatics questions with us. In addition, I would like to acknowledge my parents who have always been my greatest supporters during all years of my studies and research.

On behalf of the authors of the paper published in [179], I would like to thank Adrian Cortes, Connie Eaves, Peter Lansdorp and Keith Humphries for providing data, Bari Zahedi and Irma Vulto for their biological insight and manual data analysis, Josef Spidlen for assistance in running FLAME, Aaron Barsky and Nima Aghaeepour for their editorial comments, and Mani Ranjbar for programming guidance. This work was supported by NIH grants 1R01EB008400 and 1R01EB005034, the Michael Smith Foundation for Health Research, the National Science and Engineering Research Council and the MITACS Network of Centres of Excellence. On behalf of the authors of the paper published in [151], I would like to thank Habil Zare and Aaron Barsky for their editorial comments. In regard to the Chapter 3 of this dissertation, I would like to acknowledge Dr. Tobias Kollman for providing the data, Darren Blimkie and Edgardo S. Fortuno for manual analysis of the data, Kevin Ho for his help on data preparation, Aleksandra Legidowicz, Nathan Corbett and Radina Droumeva for their comments and feedbacks. This work was supported by NIH/NIBIB grant EB008400. Dr. Ryan Brinkman was supported in part by a Michael Smith Foundation for Health Research New Investigator Award. At the end, I would like to thank all my colleagues in Dr. Brinkman lab, Kieran O’Neill, Josef Spidlen, Ali Bashashati, Nima Aghaeepour, Habil Zare, Melanie Courtot, Nathan Corbett, Radina Droumeva, Ehsan Chitsaz, and all my friends who have been supportive to my research.
Dedication

I would like to dedicate this Doctoral dissertation to my parents and my brother. Their love, patience and encouragements have always been my greatest supports for following my dreams in the beautiful world of the science.
Chapter 1

Introduction

1.1 Flow Cytometry: Technology and Application

Flow cytometry (FCM) is the technology of measuring physical, chemical, and biological characteristics of cells as they pass through a tube one by one, and are interrogated with laser beams [148]. In FCM, intact cells are analysed individually by a flow cytometer after being tagged with fluorescently conjugated antibodies and/or with fluorescent reagents [19, 149]. In the instrument, cells are aligned by hydrodynamic forces and the laser beam excites the fluorescent molecules in/on cells when they pass through a tube at speeds higher than tens of thousands of cells per second [34]. The light scattered by each cell passing through the beam provides an indication of cell shape and size. Also fluorescent lights emitted by fluorescent chemicals found in the cell or attached to the cell provide information regarding chemical and biological characteristics of each individual cell. Typically, each data file generated by a flow cytometer contains measurements of cell properties (including phenotype and cytokine expression) in up to 17 distinct fluorescences and two scattered light parameters on each cell for up to millions of individual cells [42].

FCM has found many applications in pure basic medical research as well as clinical applications for diagnosis and disease monitoring. The areas of FCM applications include but not limited to DNA analysis and measurements [36, 53, 98, 128], drug discovery [65, 155], stem cell research [51, 109, 134], cancer research [17, 51, 109, 181], analysis of minimal residual disease (MRD) [7, 38, 67, 122], organ transplantation [161], diagnosis of immunodeficiency diseases [91], and clinical microbiology [10]. One crucial application of FCM technology is identifying functionally homogeneous populations of cells within the immune system [22]. Such identification, called immunophenotyping, is important for understanding disease pathogenesis [47, 56, 78, 94, 105]. Identification of immune cell subsets is usually followed by experimental tests to see whether cell subsets from healthy controls vs.
patients are different or respond differently when challenged with external stimuli. Cytokine flow cytometry (CFC) assays also are used for analysing a wide range of immunological parameters for many different diseases, and to measure intracellular cytokine responses of immune cells [159].

FCM has improved significantly over the last decade, advancing the field of cell research by allowing the analysis of multiple surface markers and intracellular cytokines at the level of single cells. The advances of FCM technology have been in the fields of hardware instrumentation, software analysis tools and reagent developments [41, 117]. In 2001, the Herzenberg laboratory in Stanford developed the first 11-colour, 13 parameter flow cytometer capable of identifying human naive T cells by phenotype, function, and T-cell receptor diversity [55]. In 2006, the Vaccine Research Centre at the NIH extended this technology to 17 colours by utilizing quantum dots conjugated to monoclonal antibodies [42]. Recently, Stanford University and the University of Toronto developed single-cell “mass cytometry” that combines FCM and mass spectrometry technologies allowing the measurement of 34 cellular parameters [25]. Mass cytometry technology is expected to be extended to 100 parameters per cell, affording the opportunity to significantly increase our understanding of the enormously complex immune system.

1.2 Analysis of Flow Cytometry Data

Although the development of new flow cytometric assays with high dimensionality can potentially improve our understanding of the immune system, this goal will not be accomplished without the aid of efficient data analysis techniques. The increased dimensionality and complexity of FCM data has increased the demand for developing new techniques for efficient analysis of high throughput FCM data. It is widely recognized that FCM data analysis is one of the most challenging aspects of FCM experiments [18].

1.2.1 Manual Analysis

Traditionally, cell biology researchers have mostly relied on intuition for analysis of FCM data, and the manually-managed analysis systems are the most developed ones [18, 41]. Manual analysis includes but is not limited to FCM data quality control [107], compensation [138], and manual cell population identification termed as manual gating [37, 82, 87]. Manual analysis can be subjective, and may produce variations in clinical tests as the analysis outcome can be influenced by the researcher’s judgment and
1.2. Analysis of Flow Cytometry Data

choice of method. Also, manual analyses are known to be time-consuming and labour-intensive [18]. The number of possible cell populations increases exponentially with the number of FCM parameters (e.g., for a 17-colour FCM data, there could be 131,072 possible cell populations). Since manual investigation of all these populations is infeasible, the choice of parameters to be examined tends to be hypothesis-driven, and limited in numbers. This can in turn result in ignoring relationships among markers that were not previously envisioned. Also, this constrains the discovery of unknown associations between cell populations and the underlying disease or specific immune responses [41].

1.2.2 Automated Analysis

The drawbacks of manual analysis and the increased volume and complexity of high throughput FCM data have increased the demand for developing sophisticated computational techniques and statistical tools for the purpose of exploration of FCM data in research and clinical applications [30, 41, 48, 112]. The development of automated tools and computational techniques can potentially transfer the utility of FCM data analysis from hypothesis-driven stage to hypothesis-generating and research discovery mode [143].

Recently, new computational methods and software have been developed to automate the procedures of FCM data analysis. The research areas that focus on automation of FCM data analysis include, but are not limited to data quality assessment, transformation, normalization, visualization, fingerprinting, data classification and automated gating [18, 81].

Analysis of FCM Data Using R/BioConductor- BioConductor is an open source open development software that provides tools for analysis and comprehension of high-throughput biology data [77]. BioConductor is a common research platform which enables bioinformaticians, statisticians and biologists to share their computational methods and statistical tools, and provides FCM bioinformaticians with wide range of tools available to analyze high-throughput FCM data in order to answer their biology questions. BioConductor packages are developed by using the R programming language [162].

Data Structures for Flow Cytometry- In [85], Hahne et al. provided flowCore, a freely available R/BioConductor package that enables researchers to handle FCM data using a suitable data structure that has proven to be highly efficient in capturing and organizing the analytical work.
1.2. Analysis of Flow Cytometry Data

flowCore also facilitates the automated analysis of complex FCM data by providing functions for applying basic analytical components including transformation, compensation and gating using its well-developed data structures. In [156], Strain et al. developed an R package called plateCore that uses flowCore to produce data structures and methods for analysis of plate-based FCM datasets. In addition, plateCore is capable of performing automated thresholding based on negative controls.

**Quality Assessment** - Quality assessment (QA) is an important step in the data analysis pipeline that helps researchers to identify the differences between samples originating from changes in conditions that are probably not biologically motivated. These samples are given special consideration or even excluded from further analysis. In [79], Gopalakrishnan et al. developed an R package called flowQ for performing quality assessment on FCM datasets.

**Transformation** - Values of FCM parameters can vary over several orders of magnitude. The choice of transformation for these parameters influences the data visualization and downstream analysis of the data, including cell population identification. The flowCore package [85] provides the ability to apply different transformations, including linear, logarithmic, hyperbolic arcsine, and biexponential (logicle). The Flow software developed by Frelinger et al. [73] provides a wide range of transformations including general scale, normal scale, quadratic clip and hyperlog [14] in addition to all other transformations implemented in flowCore package. Finak et al. investigated the influence of the commonly used transformations applied to FCM data in the context of automated gating results [70]. They concluded that the choice of transformation is data specific, however, the preferred transformation for fluorescence channels in terms of improving data visualization and decreasing mis-gating of events is the parameter-optimised biexponential, according to current best practices. Their algorithms are implemented in a BioConductor package called flowTrans.

**Normalization** - Non-biologically relevant variations between FCM samples which are usually imposed by the changes of the instrument settings result in significant challenges for data analysis, especially when the goal is to match biologically relevant cell populations across samples. Therefore, it is crucial to normalize FCM samples as a pre-processing step of data analysis. Hahne et al. developed two normalization methods that remove technical
between-sample variation by aligning prominent features (landmarks) in the raw data on a per-channel basis [84]. They implemented their algorithms in a BioConductor package called flowStats.

Data Visualization- Visualizing FCM data can be used as an efficient technique for checking the quality of the FCM data as well as the results of automated analysis in order to quickly identify the failures. To this purpose, Sarkar et al. developed a BioConductor package called flowViz [145] that provides useful visualizations for aiding the automation of FCM data analysis. Another use case of visualization techniques is using them for data exploratory purposes, where the goal is to find structures in a high-dimensional data space. Frelinger et al. developed an open source software package called Flow that facilitates this aim [73]. Also, Roederer et al. explored the use of colour to encode up to 5 independent dimensions of data in a single display [139]. This visualization technique works as a powerful display type that enables rapid data exploration.

Finger Printing- In [143], Rogers et al. introduced a new method of analysis of FCM data called Cytometric Fingerprinting (CF) that facilitates quantitative comparisons of samples. They implemented their method in a BioConductor package called FlowFP [142]. This method generates a description of the multivariate probability distribution function of FCM data in form of a “fingerprint”. CF was shown to be able to discover rare events with frequency of 0.01%, without the introduction of prior knowledge, and its approach was an extension of the probability binning method developed by Roederer et al. in [140, 141].

Automated Gating- The process of identifying homogeneous cells with similar properties and functions (“gating”) is a crucial step in FCM data analysis. Recently, new automated gating methods based on clustering algorithms have been developed for identification of cell populations [4, 33, 40, 90, 113, 126, 135, 146, 158, 172, 179].

Automated Analysis Pipeline- In [19] and [18], Bashashati et al. introduced the concept of integrating the automated tools discussed above into an automated pipeline for analysis of FCM data. Their proposed pipeline contained 7 main parts as follows: (1) quality assessment; (2) normalization; (3) outlier removal; (4) automated gating; (5) cluster labelling; (6) feature extraction; and (7) interpretation. Depending on the application, some of
the blocks of this automated analysis pipeline can be removed, and new analytical blocks can be added if required.

**Data Classification and Supervised Learning** Recently different classification methods have been applied to FCM data with the aim to classify subjects automatically [31, 116], and to investigate the associations between FCM data and clinical outcomes—including underlying disease or specific type of immune response—for biomarker discovery purposes [66, 116, 177]. Supervised learning has been also used for immunophenotyping purposes in many applications [23, 46, 72, 108, 130].

### 1.3 Automated Gating for Identification of Cell Populations

In applications of FCM data analysis, it is often desirable to first identify homogeneous subsets of cells, and then investigate the association between specific cell subtypes and the underlying disease or immune response. Traditionally, the identification of cell populations is done through a procedure called manual gating [37, 82, 87]. For manual gating, the biological expert has to use a graphical interface to sequentially visualize the scatter or density plots of all cells using two data parameters each time, and select cells with similar parameters values by drawing a 2 dimensional “gate” around them. It is well recognized that manual gating can be subjective, time-consuming, and labour-intensive [18]. Also, manual analysis is constrained by sequential 2D gating, which ignores the high dimensionality of the data, and may propagate errors sequentially. In addition, the choice of two parameters in each step of the 2D sequential gating is hypothesis-driven and can be a source of variation in clinical tests, as different researchers may choose different strategies for their sequential gating.

Recently, new automated gating methods using clustering algorithms have been developed to substitute for manual gating. The recently developed clustering algorithms for identifying cell populations in FCM data include, but are not limited to, model-based clustering algorithms [33, 40, 69, 90, 113, 135], K-means clustering algorithms for FCM [4, 118, 180], and density-based algorithms [126, 146, 158, 172]. In this study, I developed a novel spectral-based clustering algorithm for FCM data called SamSPECTRAL [179].
1.3. Automated Gating for Identification of Cell Populations

1.3.1 Model-Based Clustering Algorithms

In model-based clustering algorithms, the assumption is that the data consists of mixtures of components with \textit{a priori} assumed probability distribution, and the goal is to find the model parameters, such that the mixture of components with the optimised parameters best represent the data.

Boedigheimer \textit{et al.} developed the idea of applying Gaussian mixture models to FCM data for automated identification of cell subtypes \cite{33}. In their approach, they did not identify number of clusters computationally, but rather chose the number of mixture components empirically based on six samples from the experimental data set. In \cite{40}, Chan \textit{et al.} used the Gaussian mixture model clustering, but at the same time, they addressed this issue by using Bayes’ Information Criterion (BIC) as a method for identification of the number of cell populations \cite{147}.

In order to allow identification of non-symmetric clusters, and to reduce the effect of outliers in clustering FCM data, Lo \textit{et al.} \cite{113} generalized Gaussian mixture models using a flexible model-based clustering approach based on t-mixture models with a Box-Cox transformation \cite{62}. In their approach the modified Expectation-Maximization algorithm handles parameter estimation and transformation selection simultaneously \cite{58}. They implemented their clustering algorithm for automated identification of FCM cell populations in a BioConductor package called flowClust \cite{114}.

Using transformations like Box-Cox can diminish skew but may lead to less accurate models. To address this issue, Pyne \textit{et al.} developed a direct multivariate finite mixture modeling approach, using skew and heavy-tailed distributions (skew-t mixture model), to find the populations in high-dimensional cytometric data without the need for transformation \cite{135}. They referred to their method as flow analysis with automated multivariate estimation (FLAME), and made it freely available in a software package with the same name. The running time of the FLAME algorithm increases with the fourth degree of the number of dimensions. In practice this tends to make the algorithm impractical for more than five dimensions.

Since no single component of the mixture model can properly fit to an asymmetric and non-convex cell population, in \cite{69}, Finak \textit{et al.} developed the idea of merging multiple components of the mixture model, such that a combination of multiple components represent an individual cell population with asymmetric and non-convex shape. In their method, the flowClust algorithm in which the best model is selected by BIC provides the initial mixture components. Then, their merging algorithm merges pairs of overlapping clusters in order to minimize the entropy of the model. They implemented
1.3. Automated Gating for Identification of Cell Populations

Their algorithm in a BioConductor package called flowMerge. Baudry et al. used a similar entropy-minimizing approach for merging the clusters obtained from Gaussian mixture model clustering, and used an automatic way of selecting the actual number of cell populations via a piecewise linear regression fit to the rescaled entropy plot [20].

Mixture model clustering algorithms tend to be computationally challenging and time-intensive when the size and dimensionality of the data is large. On the other hand, finding small clusters (rare populations) is challenging in large size FCM data with high dimensionality. Naim et al. addressed these two challenges through developing a 2-tiered scalable multivariate parametric clustering algorithm called SWIFT [90]. In the first stage of this method, they used a Gaussian mixture model in combination with an iterative weighted sampling technique to estimate the mixture components successively in order of decreasing size. In the second stage, they used a graph-based hierarchical merging to combine the Gaussian components with significant overlaps to obtain the final cell populations. Their result indicated that this scalable mixture model clustering was capable of detecting small populations in large size FCM data [90].

Another attempt at addressing the computational challenges of model-based clustering algorithms for massive datasets is the work of Frelinger et al. [74], where the mixture-model algorithms are optimised and implemented for massively parallel yet highly affordable graphics processing units (GPU) [157].

Overall, the major drawback of these parametric methods is the requirement for assumptions on either the size of the clusters or the cluster distributions and shapes [40], which could result in incorrect identification of biologically interesting populations.

1.3.2 K-means for Clustering Flow Cytometry Data

The K-means clustering algorithm can be considered as a special case of mixture model clustering in which the clusters are assumed to have spherical shapes. K-means is similar to mixture model clustering in the sense that both algorithms try to find the centres of the clusters in an iterative fashion. Despite the implicit restriction of the K-means algorithm to spherical cell populations, some researchers have shown interest in applying K-means clustering to FCM data, because of the time-efficiency and fast convergence of K-means clustering [15, 118, 125]. However, there are serious limitations in applying K-means to FCM data. (1) K-means require a priori identification of the number of clusters or cell populations. (2) The clustering results
1.3. Automated Gating for Identification of Cell Populations

are sensitive to the cluster centers initialization. (3) K-means implicitly is restricted to spherical shaped clusters.

In [180], Zeng et al. designed a K-means based algorithm that was driven by histogram features; that is, the relevant single parameter histogram features were used to guide multidimensional K-means clustering without an a priori estimate of the cluster number. However, this extension of K-means was not tested extensively on real FCM datasets. Dabdoub et al. used K-means in their analytical software called FIND to apply the automated gating to FCM data [50]. FIND starts with a high number of clusters, and then the resulting clusters from K-means are iteratively merged together using statistical shape comparison [15], to discover those most similar, until the user-specified number of clusters is achieved. The main problem with this algorithm is that it still needs the user to specify the number of clusters. Aghaeepour et al. developed a K-means based algorithm called flowMeans [4], and showed that flowMeans was successful in addressing the issues commonly exist in applying K-means to FCM data. flowMeans can identify non-spherical populations by combining multiple clusters to model a single population. It also determines the number of FCM cell populations through a change point detection algorithm. Using the spherical model of the K-means algorithm, flowMeans has a significantly lower runtime compared to more flexible but computationally expensive statistical models (e.g., a skew/t-mixture model). Because flowMeans is not designed to be robust to outliers, Luta suggested to replace the K-means clustering with more robust and still fast enough versions of K-means, such as trimmed K-means clustering [76], in order to provide the needed protection against the influence of outliers present in FCM data [118].

1.3.3 Density-Based Clustering Algorithms

Model-based clustering algorithms require assumptions about the cluster distributions that limit their general application [158]. On the other hand, a priori identification of the number of clusters pose more limitations, and can negatively impact the quality of the final clustering results. By comparison, model-independent methods (like density-based clustering algorithms) usually do not impose any a priori assumptions on the clusters shapes, distributions and sizes.

In [172], Walther et al. developed a density-based method for clustering FCM data. Their approach follows the rationale behind manual gating that implies that the clusters of the data can be identified as the contours of high density regions. Following this paradigm, they modelled each high-density
1.3. Automated Gating for Identification of Cell Populations

region by a set of grid points. To this end, they established links between
certain neighbouring grid points based on statistical decisions regarding the
gradient of the density estimate. Following the chains of links that connect
certain grid points to their ends, each chain was determined to be either
cluster or the background. As the last step of this algorithm, the similar
clusters were merged together to identify the cell populations. This nonpara-
metric method is shown to be able to reproduce non-convex populations that
are difficult to identify by model-based clustering algorithms before merging
clusters. However, one drawback of this method is that although it can po-
tentially be extended to higher dimensions, so far it is implemented for 2D
automated gating only.

In [146], Qian et al. developed a grid-based density clustering called
FLOCK (FLOw Clustering without K) for identification of FCM cell popu-
lations. FLOCK consists of four main stages. First, the data space is partitioned into “hyper-regions”. Then each hyper-region is assessed to determine
the number of events it includes, and to identify the dense hyper-regions for
which the number of events exceeded a predefined threshold. In the third
stage, dense hyper-regions adjacent to each other in n-dimensional space
are grouped together, and the centroid representing all events in each dense
hyper-region group is determined. The centroids of hyper-region groups are
used to assign each event to the closest centroid. Then the centroids are
updated based on the member events of the dense regions, and the cluster
memberships are updated using the new centroids. FLOCK was shown to be
able to identify 17 distinct subpopulations of B cells objectively, while elimi-
inating operator-dependent variability resulted from manual gating [146].

Another density-based clustering algorithm for FCM data is the curvHDR
method developed by Naumann et al. [126]. This method uses two statistical
concepts regarding to the density of samples: (1) significant high negative
curvature that corresponds to modal regions, and (2) highest density regions
(HDR). In the first phase of curvHDR method, the regions containing pos-
sibly interesting cell populations are identified considering the high negative
curvature, while the second phase tries to improve upon the high negative
curvature regions obtained in the first phase by modifying them to include
the local high density regions. This non-parametric method clusters the
data without restricting the clusters to be ellipsoidal or some other regular
shapes as imposed by model-based methods.

Misty Mountain developed by Sugar et al. is a density contour clustering
algorithm that can be understood as a progressive top-down removal of
clouds covering a data histogram relief map to identify clusters based on
the appearance of statistically distinct peaks and ridges [158]. It was shown
that Misty Mountain is fast, robust to noise, and unbiased for the cluster shapes. It also does not need to estimate the number of clusters. However, it has the limitation of considering two closely located populations as one when the data histogram presents only one peak for them.

### 1.3.4 Other Automated Gating Methods

In addition to the three main categories of clustering algorithms for FCM data (model-based algorithms, K-means extensions, and density-based clustering approaches), there are several other automated methods for clustering FCM data, including the template-based for batch clustering developed by Suni et al. [119–121, 159], predefined histogram-based gating [166], real-time adaptive clustering [75], Kohonen self-organizing maps [32], and fuzzy clustering for automated gating of FCM data [96].

### 1.3.5 Flow Cytometry: Critical Assessment of Population Identification Methods (FlowCAP) Project

As mentioned before, several clustering methods have been recently developed in an effort for automating the procedure of cell populations identification in FCM data. However, it was unclear how each of these algorithms perform in comparison to each other and to the manual gating, as these methods were tested on different datasets, and their evaluation methods were different as well. To address this shortcoming, Aghaeepour et al. initiated the Flow Cytometry: Critical Assessment of Population Identification Methods (FlowCAP) project [2]. The goal of FlowCAP was to advance the development of clustering algorithms for automated identification of FCM cell populations through providing the means to test and compare these automated methods in an objective way by using standardized common datasets. In FlowCAP-I, five typical FCM datasets, and 4 different challenges were presented for competition (Appendix D). The following algorithms participated in FlowCAP-I competition: ADICyt (unpublished), flowMeans [4], FLOCK [146], FLAME [135], Misty Mountain [158], FlowVB (unpublished), flowClust/Merge [69, 113], L2kmeans (unpublished), CDP [157], SWIFT [90], curvHDR [126], TCLUST [76], flowKoh (unpublished), Radial SVM (unpublished), randomForests [35]. The winners of these four challenges were ADICyt (Challenge 1), ADICyt (Challenge 2), ADICyt and SamSPECTRAL (Challenge 3), and Radial SVM (Challenge 4).
1.4 Assigning Labels to Cell Populations

Clustering algorithms for FCM data basically generate homogeneous subsets of cells, but the labels assigned to the clusters are just arbitrary numbers. In most of the applications of FCM data analysis, it is crucial to assign biologically meaningful labels to the cells subsets and match clusters across different samples, in order to compare corresponding populations of different samples, and to perform downstream analysis after clustering. Matching similar cell populations across samples after clustering is called cluster matching or cluster labelling [18].

Despite the crucial need for matching the desired cell populations across FCM samples, to my knowledge, there is only one study addressing the automated labelling of the FCM cell populations after clustering [135], and in the rest of the cases, either the clustering is followed by manual labelling of the clusters [15, 33, 40, 73, 113, 125], or the cluster labelling is embedded in the clustering and initialised by the operator [159, 166]. An example of cases for which assigning labels to the clusters is embedded in the automated gating procedure is the method that was developed by Suni et al. [159], and was used in several studies [119–121, 159] to measure cytokine responses of T-cell subsets in a consistent way. In this method, templates for automated gating were built, and then samples were processed by these templates using batch analysis.

In addition to the methods in which assigning labels to the cell subsets is embedded in automated unsupervised gating, there are some other methods that make use of supervised learning for automated identifications of cell populations [23, 46, 72, 108, 130]. In supervised methods for FCM data, the cells are classified into one of the predefined cell populations introduced to the algorithm in the training stage, and the corresponding labels are assigned to them. A choice of unknown class would exclude the cells that do not belong to the classes of interest [18]. Here again, assigning labels to the cell populations is embedded in the supervised automated gating. However, one point to note is that label assignment embedded either in supervised or unsupervised automated gating is not in the scope of my thesis, and by automated cluster labelling (or cluster matching), I mean assigning labels to the populations of interest after applying an unsupervised clustering method to the FCM data.
1.4.1 Metaclustering

In [135], Pyne et al. developed a 2-tiered metaclustering method for automated matching of the populations across FCM samples. In their approach, first the modes of all clusters of all FCM samples were pooled and clustered by PAM (Partitioning Around Medioids) [99] to form the metaclusters. Then, the assignments of every sample’s populations to the metaclusters of the template were refined with a bipartite matching algorithm on population modes and metacluster locations.

There are some limitations that constrain the usability of the metaclustering approach, and special care should be taken when using this method. For instance, if there are significant technical variations in the data, the metaclustering step will assign similar cell populations with dissimilar absolute positions to different metaclusters. This results in different labelling for these biologically similar cell populations [84]. Hahne et al. suggested to perform a normalization step before clustering, to reduce the data variability and to improve cluster matching using metaclustering [84]. However, special care should be taken for data normalization, because incorrect normalization may result in losing biologically important information, and this in turn can result in producing errors in FCM data analysis. Another metaclustering related issue arises when the variations between the centres of similar populations of different samples are small, but the number of clusters pooled into the metaclustering and/or the number of samples are very high, resulting in distributing the cluster centres in the data space almost everywhere. This makes it difficult to decide on the boundaries of the metaclusters by the automated metaclustering approach. On the other hand, in general, identifying the number of metaclusters is a tricky part of the metaclustering approach, and different model selection criteria (e.g., BIC [147] and IIR [135]) can result in different clustering outcomes.

1.5 Analysis of Innate Immune Response Flow Cytometry Data

1.5.1 Introduction to Innate Immunity

The immune system consists of all cells, molecules and different organs that protect the body against dangerous microbes and viruses. This protection is achieved through the coordinated actions of the innate and adaptive immune systems. Although the innate and adaptive immune systems both act to protect the body against dangerous microbes and viruses, they differ in
1.5. Analysis of Innate Immune Response Flow Cytometry Data

...a number of ways. (1) The adaptive immune system takes time to react against infections, whereas the innate immune system is ready to react to infection. (2) Adaptive immunity is antigen-specific and reacts only to invading organisms that induce response, whereas the innate immune system is not antigen-specific, and reacts equally well to a wide variety of organisms. (3) The adaptive immune system has “memory”, and “remembers” its challenge with an infection, and acts fast when it encounters the same infection again. In comparison, the innate immune system has no immunological memory. On encountering a pathogen, the innate immune system must sense the pathogen and control its early spread and replication. If innate defense mechanisms alone are not successful, the adaptive arm of immunity is required.

Antigen presenting cells (APC) are a subtype of immune cells that bridge innate and adaptive immunity. The major human APCs are monocytes/macrophages, dendritic cells (DCs), both conventional (cDC) and plasmacytoid (pDC), and B cells. The important roles of APCs in immunity include: (1) detecting pathogens through their pattern recognition receptors; (2) presenting pathogens to T lymphocytes through the peptide-MHC complex (pMHC); and (3) releasing cytokines that lead to further activation of T cells. APCs’ actions in the immune system influence the quality and magnitude of the adaptive immune response and outcome of infection [16, 93, 110].

The innate immune system has an inherent ability to discriminate between microbes and self [123, 124]. Discrimination is accomplished through invariant pattern recognition receptors. A receptor is a molecule of protein, embedded in a membrane, to which a mobile signaling (or “signal”) molecule may attach. Toll-like receptors (TLRs) are a family of pattern recognition receptors which have a key role in detection of microbial structures or molecular “danger signals”. Thirteen different types of TLRs have been recognized to date, 10 of which are expressed in humans [5, 93, 160, 164]. Each type of TLR has distinct subcellular localization, though they usually are found on the surface of APCs [95].

A molecule which binds to a receptor is called a “ligand”, and when such binding occurs, the receptor ordinarily initiates a cellular response. The ligand may be a peptide, a hormone, a pharmaceutical drug, or a toxin [8]. When an antigen binds to a TLR, it is drawn into the cell by endocytosis. Subsequently, MHC-II molecules are synthesized in the endoplasm, and then form a pMHC-II complex. This complex is transported to the cell surface for presentation to T cells. An additional co-stimulatory signal is then produced by APCs leading to activation of the T cell. Cytokines are a category of signaling proteins secreted by immune cells when encountering pathogens.
Secretion of cytokines by APCs results in activation of T cells in order to increase protection against pathogens.

### 1.5.2 The Study of Neonate Innate Immunity Using Flow Cytometry

It is estimated that in 2008, 8.8 million children under five died worldwide, while 20% of the deaths was vaccine preventable (1.7 million) [1]. Newborns and young infants are at higher risk of morbidity and mortality compared to older children and adults [132, 153]. In fact, neonatal mortality forms around 40% of the deaths under the age of five [97]. Mortality and morbidity due to infection are at the highest rate in the first few weeks after birth, and decrease over the course of the next several years. It is estimated that 4 million of the 130 million infants born each year die in the first 4 weeks of life [97]. The cause of death in 36% of cases is infection [106].

The innate immune system is known to be central to all immunity, because it recognizes the microbial patterns and presents them to other immune cells. This in turn provides appropriate immediate protection and determines the quality of subsequent adaptive immune response [102]. Knowing this fact, a more complete understanding of the mechanism of innate immunity in the first few years of the life is necessary, as this could lead to improving the health of neonates by providing appropriate vaccinations in early ages, including around birth [45].

Previous studies for innate immune analysis were limited in the sense that they were analysing either only innate immune cell numbers [171], or cytokines secreted into culture supernatants [57], and consequently, they did not provide the single cell-specific information needed for accurate study of the innate immune response to TLR ligand stimulation. In [89], Ida et al. developed a 4-colour flow cytometry assay to assess the dendritic cells response when stimulated by the TLR ligands, however, the capacity of this assay was limited in the sense that it used only 2 colours for the functional read out. Recent advances in FCM allowed the inherent complexity of the innate immune response to be captured more fully at the single cell level. In [95], Jansen et al. designed an automated FCM-based assay to provide a platform for accurate analysis of the innate immune response. This platform enabled the rapid interrogation and large scale screening of human blood APC responses to TLR ligand stimulation. They divided each sample of whole blood (WB) and peripheral blood mononuclear cells (PBMC) from healthy adults into 52 aliquots, and stimulated 50 of them with 10 different TLR ligands at 5 different concentrations, leaving two aliquots unstimulated.
as negative controls. The full list of TLR ligands and their concentrations is available in [95]. They then used a polychromatic flow cytometric high-throughput assay to analyze the innate immune response to TLR stimulations. In this study, the cell surface markers were used to identify various innate immune cell subpopulations: monocytes, conventional DCs (cDCs) and plasmacytoid DCs (pDCs), while the intracellular cytokine markers were used to measure functional response of different APC subpopulations.

The fact that neonates and infants are more susceptible to infections compared to adults reflects differences between infants and adults both in innate and adaptive immunity. However, the extent to which adults and neonates differ in terms of innate immune response to microbial stimuli needs more investigation to be fully characterized. In [102], Kollmann et al. used the same platform as developed in [95] to study the differences between innate immune response of neonates and adults when challenged with microbiologically derived danger signals. In their study, they used three concentrations of each TLR ligand known to yield low, medium, and high responses based on their previous study [95], to assess possible concentration-dependent differences between adults’ and neonates’ innate immune responses. The results of this study suggested that the neonates’ innate immune responses to TLR stimulation compared to the adults’ innate immune responses were not so much deficient in quantity, but differed in quality [102]. In fact, their findings indicated that innate immune cells of neonates had a significantly reduced capacity to produce some cytokines, including IL-12, IFN-α and IFN-γ, a more modest reduced capacity to produce TNF-α, and greater capacity to produce IL-10. However, in general, neonatal innate immune cells were less polyfunctional compared to adults’ innate immune cells, that is, they less often produced two or more cytokines simultaneously.

In another study, Corbett and Kollman et al. extended their work to comprehensively compare the differences between the innate immune responses of neonates and infants (n=35) at three different ages (at birth, one year, and two years) and those of adults (n=25) when the blood samples were stimulated by TLR ligands [45]. In that study, they analysed both the global (through cytokine quantification in the culture supernatant) and the single-cell (via intracellular cytokine staining (ICS)) response of APC subpopulations to a broad range of well-defined TLR stimuli. Similar to their previous works [95, 102], high-throughput FCM was used to identify APC populations, and to study cytokine responses to TLR ligand stimulation at the single-cell level. Their findings contradicted a linear progression patterns from immature neonatal to mature adult innate immune reactivity, but instead indicated that there existed “age-specific” changes in innate immune
1.6. My Contributions to the Automated Analysis of FCM Data

reactivity to TLR ligand stimulations [45].

Although this work was the most comprehensive analysis focused on the innate immune response following TLR stimulation over the first two years of life in the largest such longitudinal cohort studied to that date (35 subjects), their goal is to expand the study. Follow-up studies are projected to soon include 10,000 neonates-infants at three ages (at birth, one year, and two years). Since for each subject, there will be four TLR ligand stimulations and 2 unstimulated control samples, the total number of FCM samples would reach a total of 180,000. Obviously, the analysis of such a huge number of samples is not feasible manually, and this in turn highlights the importance of developing automated tools for the analysis of this data.

1.5.3 Measuring Functional Response of Cytokine Producing Cell Populations

The immune response in humans is usually assessed using immunogenicity assays measuring the expression of biomarkers which are correlates of protection (CoP). FCM is the assay of choice to measure intracellular cytokine staining (ICS) of cell-mediated immunity (CMI) biomarkers. For CMI analysis, the integrated mean fluorescence intensity (iMFI) was introduced as a metric to represent the total functional CMI response as a CoP [52]. iMFI is computed by multiplying the relative frequency (percent positive) of cells expressing a particular cytokine with the mean fluorescent intensity (MFI) of that population, and correlates better with protection in challenge models than either the percentage or the MFI of the cytokine-positive population. While determination of the iMFI as a CoP can readily be accomplished in animal models that allow challenge/protection experiments, this is not feasible in humans for ethical reasons.

1.6 My Contributions to the Automated Analysis of FCM Data

As mentioned before, it is well-recognized that there are emerging needs for developing computational methods and statistical tools that facilitate the automated analysis of FCM data [18, 41]. In this thesis, I contributed to this area of research by (1) developing a method for automated gating of FCM data, (2) developing a learning-based approach for cluster matching across different samples, (3) designing and implementing a semi-automated analytical pipeline for full analysis of immune response FCM datasets, and
analysing correlation between intracellular and secreted cytokines, and deriving a mathematical formula for measuring the functional response of cytokine producing cells.

1.6.1 SamSPECTRAL Clustering Algorithm for Automated Gating of Flow Cytometry Data

Spectral clustering is a non-parametric clustering method that avoids the problems of estimating probability distribution functions by using a heuristic based on graphs [169]. It has proved useful in many areas of pattern recognition [11, 12, 127, 131]. However, it cannot be directly applied to large multidimensional FCM datasets due to time and memory limitations. To address this issue, Habil Zare and I jointly modified spectral clustering by adding an information preserving sampling procedure and applying a post-processing stage [179]. Please refer to the preface of this thesis for the details of our contributions. We called the resulting algorithm SamSPECTRAL, and tested its performance on different FCM datasets (See Appendix E for the list of datasets). Chapter 2 is devoted to the explanation of the SamSPECTRAL clustering algorithm, and to studying its performance when applied to the FCM datasets.

1.6.2 Learning-Based Cluster Matching Method

I developed a novel learning-based cluster matching method that searches for the best matches of the desired populations among all clusters generated by a clustering algorithm. My cluster matching is a two-tiered semi-automated method that incorporates biological expert knowledge to find the cluster that best matches a target cell population. In this method, first all FCM samples are clustered by running a clustering algorithm multiple times to form a cluster set. Then, in the first step of the cluster matching method, for each group of samples, the desired populations of at most six randomly selected training samples are identified manually by a biological expert, and the features of the target populations in these samples are measured. Then, for other FCM samples, the same features of all clusters of their cluster set are measured, and the likelihood that each cluster matches to the population of interest is computed considering the similarity of its features to the target population features in the training samples. The cluster with the highest likelihood ratio is selected as the best match of the desired population. This novel cluster matching method is compatible with all clustering algorithms, and can be used in combination with any unsupervised automated gating.
1.6. My Contributions to the Automated Analysis of FCM Data

method to identify cell populations of interest across different FCM samples. Section [3.6] of my thesis is dedicated to explaining this novel learning-based cluster matching method in more detail.

1.6.3 Semi-Automated Analysis Pipeline

I also designed a semi-automated pipeline for the full analysis of innate immune response FCM data. My pipeline includes the following steps; (1) data preparation and preprocessing, (2) automated gating using the SamSPECTRAL clustering algorithm, (3) learning-based cluster matching for finding the best matches of target cell populations and labelling antigen-presenting cell (APC) populations, and (4) measuring the functional responses of cytokine producing cells. Automated gating using the SamSPECTRAL algorithm in combination with the learning-based cluster matching is used to identify APC subpopulations in a semi-automated way. My pipeline is also capable of measuring cytokine responses of APC populations to challenge by TLR ligands in an automated fashion. More details regarding this semi-automated pipeline are provided in Chapter [3].

1.6.4 Correlation Analysis of Intracellular and Secreted Cytokines Using GiMFI Formula

As a first step towards extending the iMFI concept to humans, I investigated the correlation between the iMFI derived from a human innate immune response ICS assay and functional cytokine release into the culture supernatant, as innate cytokines need to be released to have a functional impact. Next, I developed a quantitatively more correlative mathematical approach for calculating the functional response of cytokine-producing cells by incorporating the assignment of different weights to the magnitude (frequency of cytokine-positive cells) and the quality (the MFI) of the observed innate immune response. I refer to this model as Generalized iMFI (GiMFI). Chapter [4] is devoted to explaining this in more detail. This was collaborative work with Dr. Tobias Kollmann’s lab. Please refer to the preface of this thesis for more information on my contribution to this research.
Chapter 2

Data Reduction for Spectral Clustering to Analyze High Throughput Flow Cytometry Data

2.1 Background

High throughput data analysis is a crucial step in research endeavours involving gene expression, protein classification, and FCM. A classical approach for analysing biological data is to first group individual data points based on some similarity criterion, a process known as clustering, and then compare the outcome of clustering with the biological hypotheses. Automated identification of FCM cell populations is complicated by overlapping and adjacent populations, especially when low and high density populations are close to each other. Analysing such data requires clustering methods that can separate these populations.

Spectral clustering is a non-parametric clustering method that avoids the problems of estimating probability distribution functions by using a heuristic based on graphs [169]. It has proved useful in many pattern recognition areas [11, 12, 127, 131]. Not only does it not require a priori assumptions on the size, shape or distribution of clusters, but it has features that make it particularly well-suited to clustering biological data:

- it is not sensitive to outliers, noise or shape of clusters;
- it is adjustable so that biological knowledge can be utilized to adapt it for a specific problem or dataset;
- its proper performance is studied [170].
2.1.1 Problem Statement

Two main challenges in applying spectral clustering algorithm on large data sets are the computationally expensive steps of constructing the normalized matrix and computing its eigenspace. For instance, for high throughput biological data containing one million data points (i.e., vertices), it requires computing eigenspace of a million by million matrix, which is infeasible in terms of memory and time. Although there are some approximation methods for speeding up this computation [49, 163], these could produce undesired errors in the final results. The problem of applying this algorithm on large datasets has been studied in [71] using Nyström’s method. They suggest a strategy of sampling data uniformly, clustering the sampled points and extrapolating this solution to the full set of points. However, sampling data uniformly can miss low-density populations entirely when the density of adjacent populations varies considerably, a situation that often arises for biologically interesting populations in FCM data. Appendix C includes an experiment to explain the effect of uniform sampling in such cases.

Data reduction schemes have been developed to reduce the complexity of the FCM data while preserving the information [39, 44]. These methods reduce the dimensionality but not the size of the dataset, the latter being the more important bottleneck for spectral clustering.

2.1.2 Our Approach

In our collaborative work, Habil Zare and I hypothesized that spectral clustering could significantly improve high throughput biological data analysis. However, serious empirical barriers are encountered when applying this method to large data sets. Specifically, for \( n \) data points, the running time is \( O(n^3) \), requiring \( O(n^2) \) units of memory. For instance, it would take 2 years and 5 terabytes of memory to analyze a typical FCM sample with 300,000 events using a single CPU machine. I proposed a novel solution for this problem through non-uniform information preserving sampling. This heuristic approach is specific to cytometry applications and made it possible for the first time, to apply spectral clustering method on FCM data.

2.2 Methods

In this chapter, I distinguish between the terms biological populations, clusters and components as follows. A population is a set of cells with similar functionality or molecular content. By a cluster, I mean a set of data points.
that are grouped together by a given spectral clustering algorithm. I incorporate a post-processing stage on spectral clusters to find the connected components intended to estimate the biological populations.

### 2.2.1 Spectral Clustering

The first step is to build a graph. The vertices represent the $n$ data points (e.g., cells in FCM data), and the edges between the vertices are weighted based on some similarity measure. The adjacency matrix of the graph is then normalized using the following formula:

$$\tilde{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}}, \quad (2.1)$$

where $A$ is the adjacency matrix of the original graph and $D$ is a diagonal matrix where the $(i,i)$ entry is equal to the sum of the weights on the edges that are adjacent to vertex $i$.

The next step is to compute the eigenspace of the normalized matrix. That is, all vectors $\mathbf{V}_i$ and values $\lambda_i$ satisfying the following equation are computed:

$$\tilde{A} \mathbf{V}_i = \lambda_i \mathbf{V}_i. \quad (2.2)$$

In order to find $k$ clusters, an $n$ by $k$ matrix is built using the $k$ eigenvectors with highest eigenvalues. The rows of this matrix are normalized and finally k-means is used to cluster the rows.

However, the above method cannot be directly applied to FCM data, due to large number of data points (cells) per sample. My solution for this problem is a data reduction scheme developed specifically for this purpose. This reduces the number of vertices significantly, but in a way such that biological information can be preserved by updating the weights on the edges.

### 2.2.2 Data Reduction Scheme

While data size can be reduced by known sampling methods [174], an appropriate method should be used to preserve biologically important information. From a high-level perspective, the data reduction scheme that Habil Zare and I developed together (Figure 2.1) consists of two major steps; first the data is sampled in a representative manner to reduce the number of vertices of the graph (Figure 2.1b ). Sample points cover the whole data space uniformly (Figures 2.2b), a property that aids in the identification of both low density and rare populations. In the second step as described below, a
2.2. Methods

Figure 2.1: (a) Running spectral clustering is impractical on data that contains thousands of points. (b) Faithful sampling picks up a reasonable subset of points such that running spectral clustering is possible on them. However, all information about the local density is lost by considering only these sample points. (c) We assign weights to the edges of the graph; the edges between the nodes in denser regions are weighted considerably higher. The information about the local density is retrieved in this way.

similarity matrix that assigns weights to the edges between the sampled data points is defined. Higher weights are assigned to the edges between nodes in dense regions so that information about the density is preserved (Figure 2.1c). I developed the first stage of this data reduction scheme and Habil Zare developed the second stage. Please refer to the preface of this thesis for more information about our contributions to this collaborative work.

Faithful Sampling Algorithm

Assume the data contains \( n \) points in a \( d \) dimensional space of volume \( V \), and the parameter \( m \) defines the max number of points after sampling. Set the neighbourhood threshold \( h \) as 
\[
h = \frac{1}{2} \sqrt{\frac{V}{m}}.
\]
Label all data points as unregistered.

Repeat

- Pick a random unregistered point \( p \) (the representative of a new community)
- Label all unregistered data points within distance \( h \) from \( p \) as registering
- Put registering points in a set called community \( p \)
- Relabel registering points as registered

Until All points are registered.

Return All Communities (i.e. representatives of all communities and the
After faithful sampling is completed, the set of all representatives can be regarded as a sample from the data. Reducing the value of parameter $h$ will increase the number of sample points, resulting in increased computation time and required memory. Conversely, increasing $h$ will result in fewer sample points that may lead to too low a resolution. In such a case, the computed spectral clusters may fail to estimate the real cell populations appropriately. In our implementation, we use an iterative procedure (explained in the overview of SamSPECTRAL algorithm) to adjust $h$ automatically such that the number of representatives will be within the range 1500-3000. As a result of this adjustment, the following two objectives are achieved. First, computing the eigenspace of a graph with a number of points in this range is feasible, as it takes less than one minute on a 2.7 GHz processor. Second, $h$ is much lower than the range of the data dimensions (Figure 2.2) and the resulting resolution is high enough such that no biologically interesting information is lost.

In the sampling stage, there is no preference in picking up the next data point, therefore, the final distribution of the sampled points will be uniform in the “effective” space. That is, the representatives are distributed
2.2. Methods

almost uniformly in the space where data points were present (Figure 2.2). As a consequence, by repeating the sampling procedure, the final results of clustering will not change significantly. I confirmed this observation quantitatively as given in Appendix A. By considering just the representatives, density information is effectively ignored, so working directly with these representatives results in improper outcome. On the other hand, some biological information from the original data is preserved by the above algorithm that can be retrieved to guide the clustering algorithm. More precisely, for each sample point, the list of all points in its neighbourhood (i.e., the members of the corresponding community) is known. In the next stage, this information is used to define the similarity between two sample points to modify the behaviour of spectral clustering. In this sense, our sampling scheme is faithful, meaning that the valuable biological information from the original data points is preserved even after sampling. We call the overall procedure, which consists of faithful sampling, computing modified similarity matrix and spectral clustering, SamSPECTRAL clustering.

2.2.3 Similarity Matrix

The following heat kernel formula \[103\] was used to compute the similarity between two vertices \(i\) and \(j\):

\[
s_{i,j} = e^{-\frac{D^2(p_i,p_j)}{2\sigma^2}},
\]

where \(D(p_i,p_j)\) is the Euclidean distance between \(p_i\) and \(p_j\). \(\sigma\) is a scaling parameter that controls how rapidly similarity between \(p_i\) and \(p_j\) falls off with increasing distance. Developed by Habil Zare, the similarity between two communities \(c\) and \(c'\) is defined as the sum of all pairwise similarities between all members of the first community and all members of the second community. That is,

\[
S_{c,c'} = \sum_{i \in c} \sum_{j \in c'} s_{i,j},
\]

where \(i\) and \(j\) are members of \(c\) and \(c'\) respectively.

We do not normalize the similarity by dividing the above sum by the size of communities, because we would lose valuable biological information that is supposed to be preserved. In short, the size of the communities determines the local density of the data points, which is biologically of great importance, as the dense parts of the data normally correspond to the cell populations.

25
2.2. Methods

Figure 2.3: Defining the similarity between two communities and identifying the number of clusters. (a) We define the similarity between two communities $C$ and $C'$ as the sum of pairwise similarities between the members of $C$ and the members of $C'$. (b) This figure shows the largest eigenvalues of a sample from the stem cell dataset. The number of clusters is estimated according to the knee point of eigenvalues curve. This point is defined as the intersection of the above regression line and the line $y = 1$. The horizontal coordinate of the knee point estimates the number of spectral clusters.
2.2. Methods

The above definition is motivated by the following intuition from potential theory that explains how biological information is preserved after faithful sampling by assigning similarities in this way. The eigenvectors of a graph are interpreted as potential functions on the electric network modeled by the graph [28]. Assuming the radius of each community is small enough, the potential values of the community members are almost the same. On the other hand, in potential theory, the equivalent conductance between a group of nodes \( \{v_i\} \) with equal potential values and another group of nodes \( \{w_j\} \) that also have equal potential values is computed by the summation of pairwise conductance between nodes \( v_i \) and \( w_j \) for all \( i \) and \( j \). Since in this model, the similarity between two vertices is equivalent to the conductance between the corresponding electrical nodes, it is reasonable to sum up pairwise similarities to estimate the equivalent similarity between communities (Figure 2.3a).

2.2.4 Number of Clusters

The number of clusters must be determined before running the spectral clustering algorithm [173]. To find this number automatically and in an efficient manner, we propose a method that is motivated by the following observation from spectral graph theory:

**Theorem** [26]: The number of connected partitions of a graph is equal to the number of eigenvectors with eigenvalue 1.

We observed that typically for FCM data, if \( \sigma \) is adjusted properly, as explained in the SamSPECTRAL package vignette (Appendix H), the first few eigenvalues are close to one, and at a point we call *knee point* they start to decrease almost linearly. We compute the knee point by applying linear regression to the eigenvalues curve (Figure 2.3b) and use the horizontal coordinate of this point as a rough estimate for the number of spectral clusters (See Appendix H). Habil Zare and I contributed equally to this part of our method.

2.2.5 Combining Clusters

Applying spectral clustering on sampled data results in graph partitioning, which is almost optimum in the sense of having minimum normalized cut [43, 150]. However, in some cases, a biologically interesting population might be split into two or more smaller clusters by SamSPECTRAL. We addressed this issue by adding a post-processing stage wherein the partitions of a population are combined based on known properties of FCM cell populations.
Typically, biologically meaningful cell populations in FCM data have their highest density at the centre, and their density decreases towards the border of the population. Since higher density regions indicate communities with relatively more members, the conductance between them is expected to be relatively higher (Equation [2.4]). Thus, similarity between communities is higher in regions with higher densities and the highest similarity is expected to be at the centre of the biological population. This observation forms the basis for our criterion for combining clusters. Specifically, similarity between communities determines the weight on graph edges and we define the maximum weight of the edges of a spectral cluster as within similarity of that cluster. Also, the maximum weight of the edges between two different spectral clusters is defined as between similarity. If the ratio of between similarity to within similarity is greater than a predefined threshold (separation factor), we conclude that these clusters are partitions of a single population, and should combine them to form a component. We repeat this stage until no two components can be combined. The final components computed in this way are called connected components of the data, and estimate the real biological populations.

With smaller separation factors, spectral clusters tend to combine more often. In order to set the separation factor threshold, the user needs to test few values (normally in the range \([0, 1]\)) on a few samples, and then apply it to the rest of the samples of the dataset. Habil Zare and I together developed the main idea of the method for combining clusters to form the connected components. As mentioned in the thesis preface, we equally contributed to this part of our research.

**2.2.6 Overview of SamSPECTRAL Algorithm**

In summary, the stages of our algorithm are as follows, assuming the data contains \(n\) points in a \(d\) dimensional space of volume \(V\), and the parameters \(m\) (max number of communities), \(\sigma\) (scaling parameter), and separation factor are set properly.

1. Sampling:
   
   (a) Let \( h = \frac{1}{2} \sqrt[\frac{d}{m}] \frac{V}{m} \).

   (b) **Repeat:**

      - Run faithful (biological information preserving) sampling algorithm. Suppose \(m'\) communities are built.
2.2. Methods

- Update: \( h = h\left(\sqrt[\frac{d}{m^{'}}}m\right) \).

Until \( \frac{m}{2} \leq m' \leq m \).

2. Compute the similarities between communities by adding pairwise similarities \( s_{i,j} \) defined by \[ 2.3 \]

\[
S_{c,c'} = \sum_{i \in c} \sum_{j \in c'} s_{i,j}.
\] (2.5)

3. Build a graph wherein each community is a vertex. Put edges between all pairs of vertices and weight them by similarity between corresponding communities.

4. Analyze the spectrum of the above graph to find the clusters;
   (a) Normalize the adjacency matrix of the graph according to Equation \[ 2.1 \]
   (b) Compute the eigenspace of the graph and set \( k \), number of clusters, according to the knee point of eigenvalues curve.
   (c) Run classical spectral clustering algorithm to find \( k \) clusters.

5. Combine the clusters to find connected components:
   (a) Initiate the list of components equal to the list of spectral clusters.
   (b) \textbf{Repeat}:
     - For any pairs of components \( C_i, C_j \), set:
       \[
       \text{separation\_ratio}(C_i, C_j) := \frac{\text{between\_similarity}(C_i, C_j)}{\text{within\_similarity}(C_i)}
       \] (2.6)
     - For each component \( C_i \), compute:
       \[
       M(i) := \max_{j \neq i} (\text{separation\_ratio}(C_i, C_j))
       \]
     - If for all \( i \), \( M(i) \leq \text{separation\_factor} \), \textbf{break}.
     - Pick an \( i \) such that \( M(i) > \text{separation\_factor} \) and let:
       \[
       j = \arg \max_{j \neq i} (\text{separation\_ratio}(C_i, C_j))
       \]
     - Combine \( C_i \) and \( C_j \), then update list of components.
2.2. Methods

Until number of components > 1.

In the sampling stage, we start with the initial value $h = \frac{1}{2} \sqrt[2d]{V/m}$ for the neighbourhood. $m$ is a parameter that controls $m'$, the final number of sample points such that $m/2 \leq m' \leq m$. In our implementation, we used Manhattan metric to measure the distance between the points, therefore, each community that was built around a sampled point could be considered as a $d$-dimensional hyper-square with edge length $2h$. Assuming that the whole data space with volume $V$ was covered by $m$ non-overlapping communities (hyper-squares with volume $v' = (2h)^d$), then $m \times v' = V$, or in other words, $h = \frac{1}{2} \sqrt[2d]{V/m}$. Therefore if the data points were distributed uniformly in the space, we would get $m$ sample points in the first run. However, in practice, we need to repeat the procedure after updating the neighbourhood value. Assume that in one iteration of data sampling, we get $m'$ sample points, and after updating $h$ with $h_2$, we get $m$ sample points in the next iteration. Assuming $h = \frac{1}{2} \sqrt[2d]{V/m'}$ and $h_2 = \frac{1}{2} \sqrt[2d]{V/m}$, we obtain $h_2 = \sqrt[2d]{m'/m}$. This is the rational behind updating $h$ using $h_2 = \left(\frac{m'}{m}\right)^{\frac{1}{2d}}$ formula in our method. According to our experiments, a few iterations (usually < 5) are enough to fulfill the terminating condition $m/2 \leq m' \leq m$. As the running time of this part of SamSPECTRAL is $O(nm)$, which is negligible compared to eigenspace computation time, we did not attempt to optimise the sampling loop.

2.2.7 Modified Markov Clustering Algorithm (MCL)

Step 4 in the above algorithm is the classic spectral clustering method. This step potentially could be substituted by any clustering algorithm for weighted graphs. To verify that our approach is extensible in this sense, we substituted classic spectral clustering with Markov Clustering (MCL) [59] keeping the rest of our algorithm, sampling and post-processing steps, unchanged.

MCL finds the partitions of a graph by simulating flow on the nodes. Simulation is done by iteratively multiplying two type of matrices that correspond to expansion and inflation operations [59]. Because flow and eigenspace of a graph are strongly related, the outcome of this approach tends to be similar to spectral clustering through computing eigenspace. The Cheeger inequality is an example of such a relation [43].
2.2. Methods

2.2.8 FLAME and flowMerge: State of the Art
Model-Based Clustering Algorithms for FCM Data

As mentioned in Chapter 1, recently several different model-based clustering algorithms have been developed to identify cell populations in FCM datasets [33, 40, 69, 90, 113, 135]. Although model-based clustering algorithms are quite successful in many applications of FCM data clustering, they require assumptions about clusters shapes, distributions and sizes that limit their general applicabilities. This specially impose limitations when automated identification of (1) non-elliptical shape populations, (2) low density populations surrounded by dense ones, and (3) rare populations are desired.

In this thesis, I aimed to compare the performance of SamSPECTRAL algorithm to that of model-based clustering methods in order to investigate if my non-parametric clustering algorithm, SamSPECTRAL, was successful in addressing drawbacks of model-based clustering algorithms. To this purpose, I selected two model-based clustering algorithms, FLAME and flowMerge, that were shown to outperform other model-based clustering algorithms [69, 135], and I repeatedly compared performance of SamSPECTRAL to their performances throughout this chapter of my thesis. Here, I bring brief explanations of these two algorithms’ methods.

flowClust and flowMerge: flowMerge is an extension of flowClust model-based clustering approach which is based on finite $t$-distribution mixture model [69, 113]. In this model, each cluster can be regarded as a separate $t$ density with mean $\mu$, covariance matrix $\nu(\nu - 2)^{-1}\Sigma$ ($\nu > 2$) and $\nu$ degree of freedom as below.

$$
\varphi(y; \mu, \Sigma, \nu) = \frac{\Gamma(\frac{\nu+p}{2})|\Sigma|^{-1/2}}{(\pi \nu)^{p/2}\Gamma(\frac{\nu}{2})} \left\{ 1 + \frac{(y - \mu)^T \Sigma^{-1} (y - \mu)}{\nu} \right\}^{\frac{\nu+p}{2}}
$$

(2.7)

In the multivariate $t$ mixture model, it is assumed that the $p$-dimensional data points, $y_1, y_2, ..., y_n$, have come from a mixture of $G$ components with unknown weights $w_1, w_2, ..., w_G$, such that $\sum_{g=1}^{G} w_g = 1$. Therefore, given $p$-dimensional observations, $y_1, y_2, ..., y_n$, the likelihood for a mixture of $G$ components with $t$-distribution can be estimated by

$$
L(\mu_1, ..., \mu_G; \Sigma_1, ..., \Sigma_G; \nu_1, ..., \nu_G; w_1, ..., w_G; y) = \prod_{i=1}^{n} \sum_{g=1}^{G} w_g \varphi_g(y_i; \mu_g, \Sigma_g, \nu_g)
$$

(2.8)
2.2. Methods

Using expectation-maximization (EM) [58], the unknown mixing proportions, \( w_1, w_2, \ldots, w_G \), and unknown parameters \( \mu_1, \ldots, \mu_G \), \( \Sigma_1, \ldots, \Sigma_G \) and \( \nu_1, \ldots, \nu_G \) are updated iteratively until the above likelihood converges to its maximum value.

Since asymmetrically distributed cell populations are frequently found in FCM data, flowClust algorithms apply Box-Cox transformation [62] to the data in order to bring skewed data back to symmetry. Recalling symmetry is a property of \( t \)-distribution, Box-Cox transformation improves the performance of \( t \)-mixture model clustering, helping to find asymmetric cell populations. This transformation is defined by \( y^{(\lambda)} = \frac{\text{sgn}(y)|y|^\lambda - 1}{\lambda} \) for \( \lambda \neq 0 \). flowClust algorithm uses Bayesian Information Criterion (BIC) to identify number of mixture components (or clusters) in the data [147].

Since even after using Box-Cox transformation, the asymmetric and non-convex cell population could not be modeled accurately by a single component of the \( t \)-distribution mixture model, in [69], Finak et al. developed the idea of merging multiple components of the mixture model, such that a combination of multiple components represent an individual cell population with asymmetric and non-convex shape. flowMerge algorithm starts with the optimal mixture model obtained by applying flowClust algorithm to FCM data. Then in each step two clusters are selected to merge, such that the entropy of the clustering under the new cluster assignment is minimized. This approach is based on the fact that existence of two highly overlapping clusters results in a large data entropy, while if clusters do not overlap, the entropy of clustering is low. The procedure of merging two clusters is repeated several times until only one cluster remains. Looking at the curve of entropy versus number of clusters, one can find a knee point in the curve after which the data entropy does not decrease significantly with further merging of the clusters. At this knee point, the remaining clusters have low overlaps, and they are all well separated. This is referred as the flowMerge optimal solution.

**FLAME:** Using transformations like Box-Cox can diminish skew but may lead to less accurate models. To address this issue, FLAME algorithm applies a direct multivariate finite mixture modeling approach, using skew and heavy-tailed distributions (skew-\( t \) mixture model) to FCM data, to find the populations in high-dimensional cytometric data without the need for transformation [135]. In this approach, each cluster is modeled by a separate
2.2. Methods

The skew-$t$ distribution with its density function as

$$
\phi(y; \xi, \Sigma, \delta, \nu) = 2t_{p, \nu}(y; \xi, \Omega) T_{\nu+p}(\frac{\mu}{\sigma \sqrt{\nu + \eta}})
$$

(2.9)

where, \( y \in \mathbb{R}^p \), \( \Omega = \Sigma + \delta \delta^T \), \( \mu = \delta^T \Omega^{-1}(y - \xi) \), \( \sigma^2 = (1 - \delta^T \Omega^{-1} \delta) \), \( \eta = (y - \xi)^T \Omega^{-1}(y - \xi) \). Here, \( t_{p, \nu}(y; \xi, \Omega) \) is the density function of a \( p \)-dimensional \( t \)-distribution with \( \nu > 2 \) degrees of freedom, mean \( \xi \), and scale matrix \( \Omega \). \( T_{\nu+p}(.) \) stands for the distribution function of a univariate \( t \) random variable with \( \nu + p \) degrees of freedom. Here, \( \delta \) is the skew parameter.

Similar to multivariate \( t \) mixture model, it is assumed that the \( p \)-dimensional data points \( y_1, y_2, ..., y_n \) have come from a mixture of \( G \) components with unknown weights \( w_1, w_2, ..., w_G \), and therefore, the likelihood for a mixture of \( G \) components with skew \( t \)-distribution can be estimated by

$$
L(\xi_1, ..., \xi_G; \Sigma_1, ..., \Sigma_G; \delta_1, ..., \delta_G; \nu_1, ..., \nu_G; w_1, ..., w_G | y) = \prod_{i=1}^{n} \sum_{g=1}^{G} w_g \phi(y_i; \xi_g, \Sigma_g, \delta_g, \nu_g) \nonumber
$$

(2.10)

Here again the expectation-maximization (EM) algorithm [58] is used to iteratively update unknown mixing proportions \( w_1, w_2, ..., w_G \), and unknown parameters \( \xi_1, ..., \xi_G, \Sigma_1, ..., \Sigma_G, \delta_1, ..., \delta_G, \) and \( \nu_1, ..., \nu_G \) until the above likelihood converges to its maximum value.

In practice, FLAME is run for a wide range of number of mixture components, and then the Scale-free Weighted Ratio (SWR) criterion (by default) is used to select the model that best fits the data [135]. SWR is a weighted ratio of average intracenter distance to average intercluster distance, and therefore, the resulting model with the highest SWR optimises both intercluster heterogeneity and intracenter homogeneity.

2.2.9 Adjusting Parameters

I used the GenePattern implementation of FLAME to run it on the FCM datasets. In this implementation, five different criteria were available to select number of clusters. These five criteria were Scale-free Weighted Ration (SWR), Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), distance ratio, and intercluster distance. To run FLAME optimally, I selected a relatively wide range for number of clusters (2 \( \leq k \leq 30 \)), and used all above criteria to select the optimum number of clusters in this range. Each criteria resulted in one clustering outcome. I visually
2.2. Methods

inspected the results, and selected the one that identified cell populations more accurately. Similarly, to run flowClust and flowMerge optimally, I used two model selection criteria, Bayesian Information Criterion (BIC) and Integrated Completed Likelihood (ICL), that were available in flowClust package (R BioConductor). Then for each FCM sample, I visually inspected the clustering results that were obtained by using each criterion, and selected the one that identified cell populations more accurately.

For SamSPECTRAL algorithm, I set $m = 3000$ to keep the running time below 1 minute by a 2.7 GHz processor and the obtained results remained satisfactory for all samples I analysed. The separation factor and scaling parameter ($\sigma$) are two main parameters that needed to be adjusted. Decreasing $\sigma$ and increasing the separation factor will result in identifying more populations. In particular, if $\sigma$ is decreased, then according to the heat kernel formula, the weights on the edges of the graph will decrease exponentially. Therefore, the graph will be sparser and tends to obtain more partitions. In consequence, the algorithm identifies more spectral clusters. This phenomenon can be useful in identifying rare populations. On the other hand, if separation factor is too high, a single population may be split into parts. In my experiments, I applied SamSPECTRAL on one or two random data samples of a data set and tried different values. Then, the selected parameters were fixed and used to apply SamSPECTRAL on the rest of data samples. The parameters values for the data sets presented in this thesis are provided in Appendix I. The reader is referred to the SamSPECTRAL BioConductor package vignette for more explanation on how to adjust parameters for a given data set (Appendix H).

2.2.10 FlowCAP-I Competition

I participated in the FlowCAP-I competition using the SamSPECTRAL algorithm in order to achieve two main objectives: (1) to compare the performance of SamSPECTRAL clustering algorithm against a wide range of clustering algorithms specialized for automated identification of FCM data; and (2) to run SamSPECTRAL on the standardized datasets that were provided in the FlowCAP-I competition, in order to make comparisons between SamSPECTRAL and other algorithms stronger and more trustful through using the benched FCM datasets. In FlowCAP-1, algorithms competed in the following four challenges:

1. Completely Automated: Algorithms participated in this challenge were asked to either not have tuning parameters or use fixed values for
their tuning parameters by setting them initially and using the same set of fixed parameters across all datasets. Successful algorithms were recognized to be useful for exploratory analysis.

2. Manually Tuned: In this challenge, algorithms were allowed to have manually adjusted parameters that were tuned separately for individual datasets. In this way, the semi-automated gating algorithms were compared.

3. Assignment of Cells to Populations with Pre-defined Number of Populations: In this challenge, number of expected populations was provided to the algorithms in order to identify their abilities in correct labeling of the cells when the number of clusters was known in advance.

4. Supervised Approaches Trained using Human-Provided Gates: In this challenge, 25% of FCM samples with manual gates were given to the participants for tuning their algorithms. The complete dataset was used to evaluate the results.

In this competition, the results of all clustering algorithms were compared against manual gating results performed by biology experts. Detailed explanations of the methods for validating performance of clustering algorithms participated in FlowCAP-I are given in Appendix D. Results from SamSPECTRAL were submitted to the first three challenges of this competition for all five datasets that were available.

2.2.11 Datasets

In this study, SamSPECTRAL algorithm was tested on four different FCM datasets (Stem Cells, Telomere, GvHD and Viability) as explained briefly here, in addition to five standardized datasets (GvHD, DLBCL, ND, WNV and HSCT) provided in FlowCAP-I competition. A brief description of datasets’ characteristics is given in Table A.2. The GvHD dataset is available in flowCore package through BioConductor. Stem Cells, Telomere, GvHD and Viability datasets are also available upon request. FlowCAP-I datasets are publicly available through the FlowCAP website (flowcap.flowsite.org), the Immunology Database and Analysis Portal (ImmPort; www.immport.org), and CytoBank (www.cytobank.org; experiment name: FlowCAP-I).

Habil Zare ran SamSPECTRAL on Telomere, GvHD and Viability data samples, and I ran SamSPECTRAL on Stem Cell dataset in order to identify
rare cell populations. Habil Zare, Phillip Mah and I worked together to apply SamSPECTRAL on FlowCAP-I datasets (equally contributed).

**Stem Cells**

To investigate heterogeneity in the differentiation behaviour of hematopoietic stem cells, a subpopulation of adult mouse bone marrow was isolated and then each single stem cell was transplanted into one of 352 recipients \[63\]. 16 blood samples were taken from the recipients in biweekly intervals and were studied in a cytometer. The investigation contained hundreds of data files that needed to be analysed to count the frequency of each subtype of white cells they contained.

**Telomere**

In all vertebrates, telomeres consist of tandem DNA repeats of the sequence d(TTAGGG) and associated proteins. Telomere length is known to be crucial elements in ageing and various diseases including cancer and it can be estimated by FCM \[13\]. Since telomere length is different for various cell populations, these need to be distinguished before calculating telomere length.

**GvHD**

Acute graft versus host disease (GvHD) is a common outcome after bone marrow transplantation. It is difficult to diagnose in its early stages in order to provide timely treatment. To investigate how FCM can help predict the development of GvHD, and to study its advantages over microarrays, peripheral blood samples from 31 patients undergoing allogeneic blood and marrow transplant were analysed \[37\]. The samples were taken at progressive time points post-transplant and were stained with four appropriate lymphocyte phenotypic and activation markers defining 121 different populations using six markers.

**Viability**

Propidium iodide (PI) is a widely used marker for determining viability of mammalian cells \[175\] because it has the capability of passing through only damaged cell membranes. However, depending on the complexity of the data, identifying dead cells automatically might still be difficult even if this
2.3. Results

marker is used. The capability of our algorithm in identifying dead cells using PI marker was tested on a dataset from the Terry Fox Laboratory.

FlowCAP-I Datasets

In FlowCAP-I, five datasets were used for analysis: Graft versus Host Disease (GvHD); Diffuse Large B-cell Lymphoma (DLBCL); Hematopoietic Stem Cell Transplant (HSCT); Symptomatic West Nile Virus (WNV); Normal Donors (ND). Supplementary Table A.1 summarizes descriptions of these datasets (Appendix D). Detailed explanations of the pre-processing steps that were applied to FlowCAP-I datasets are also given in Appendix D.

2.3 Results

Habil Zare and I implemented our algorithm with R, and applied it on four different datasets. We were able to identify some types of biologically interesting populations that were previously known to be hard to distinguish, including:

1. Overlapping populations (Figure 2.4a-c).
2. Subpopulations of a major population (Figure 2.4d-f).
3. Non-elliptical shaped populations (Figures 2.5 and Figure 2.6a-c).
4. Low density populations close to dense ones (Figures 2.6d-f and Figure 2.7).
5. Rare populations comprising less than 2% of all data points (Figure 2.8).

Here, I demonstrate the capabilities of SamSPECTRAL in identifying biological populations in these cases and compare our results with two state of the art methods for clustering FCM data, flowMerge (version 0.4.1) and FLAME (version 3), respectively obtained through BioConductor and GenePattern. In addition, SamSPECTRAL participated in FlowCAP-I. I also include the results of attending in this competition (Table 2.2).
2.3. Results

2.3.1 Overlapping Populations

Traditionally, identifying cell populations in FCM data is accomplished by visualizing the multidimensional data as a series of bivariate plots, and separating interesting sections manually, in a process termed gating. Gating becomes challenging for high dimensional data since when the data is mapped to two dimensions, some clusters may overlap, resulting in the mixing of different populations. Consequently, even a trained operator cannot identify overlapping populations properly in all cases. However, SamSPECTRAL algorithm prevents this undesired error by considering all data dimensions together (Figure 2.4a-c). Model-based multidimensional techniques also perform generally well in this regard.

2.3.2 Subpopulations of a Population

Figure 2.4d-f shows a major blood population (granulocytes) formed from two distinct subpopulations as verified by expert manual analysis. SamSPECTRAL could clearly distinguish between two subpopulations. flowMerge merged these two populations into one, while FLAME split both subpopulations.

2.3.3 Non-elliptical Shaped Populations

While most model based techniques have a priori assumptions on the shape of populations that resulted in mixing or splitting populations, SamSPECTRAL worked relatively well on the samples with arbitrary shape populations. In Figure 2.5, the PI positive population (blue diagonal one) was clearly identified despite its non-elliptical shape. flowMerge could also distinguish this population, but it incorrectly split the PI negative population into two parts. FLAME did not correctly distinguish the two populations.

Figure 2.6a-c shows the output of the three algorithms on a four dimensional sample from GvHD dataset. While the red population has a complex shape, it could be identified with high accuracy by SamSPECTRAL. While FLAME produced a satisfactory result, flowMerge mixed this population with the one below it.

2.3.4 Low Density Populations Close to Dense Populations

Figure 2.6d-f shows a sample from GvHD dataset containing a relatively low density and a high density population close together. SamSPECTRAL clearly distinguished the red population in the centre of the plot from the
2.3. Results

Figure 2.4: Comparative clustering of the telomere dataset. (a-c) Proper identification of overlapping populations. Although two populations shown by red and blue contours are overlapping in all bi-variant plots of this 3-dimensional sample, SamSPECTRAL can properly distinguish them by considering multiple parameters simultaneously. (d) SamSPECTRAL can also identify two major subpopulations of granulocytes correctly, as verified by expert analysis. (e) flowMerge does not distinguish between two populations of interest, and (f) FLAME improperly splits the same sample into several clusters.
2.3. Results

Figure 2.5: Comparative clustering of dead cells (PI positive) and live cells (PI negative) in the viability data. (a) SamSPECTRAL could distinguish between dead cells (blue) and live cells (red) properly. (b) flowMerge identified dead cells correctly, but split live cells into two clusters. (c) FLAME did not distinguish between these two population.
2.3. Results

Figure 2.6: Comparative clustering of the GvHD dataset. (Left) Identification of non-elliptical shaped populations. (a) SamSPECTRAL could properly identify the red, non-elliptical population, while (b) flowMerge mixed this population with the one below it. (c) FLAME produced satisfactory results in identifying this population. (Right) Identification of low density populations close to dense populations. (d) SamSPECTRAL and (e) flowMerge could identify the low density population shown in red at the centre of the figure correctly, while (f) FLAME merged this population with the other ones surrounding it.
yellow dense population to its left. Moreover, it did not mix the red population with the other low density population to its right. FlowMerge also clustered this sample relatively well, requiring five times more processing time. The performance of FLAME was not satisfactory for this sample due to mixing the desired population with the other low density ones.

Figure 2.7 depicts a sample from the stem cell dataset containing a relatively low density population shown in blue. In each row, three 2-dimensional plots of the 3-dimensional data sample are presented. SamSPECTRAL could distinguish the blue population although it was surrounded by three relatively denser populations (the yellow, green and red ones). FlowMerge mixed this population with the yellow one, while FLAME mixed it with the red one.

### 2.3.5 Rare Populations

Identifying rare populations has many significant applications in FCM experiments including distinguishing cancer stem cells, hematopoietic stem cell transplantation, detection of fetal cells in maternal blood, detection of leukocytes in leukocyte-depleted platelet products, detection of injected cells for biotherapy and malaria diagnosis [60].

Figure 2.8 shows a typical sample from the stem cell data set that contains a rare population in red. This population is positive for all the three markers and in each sample, it comprises between 0.1% to 2% of total cells. I performed an experiment on 34 samples from the stem cell data set and compared the performance of SamSPECTRAL, flowMerge and FLAME. This rare population was distinguished manually and the result of manual gating was considered as the basis for my comparison. FLAME and flowMerge could identify this population only in 11 (32%) and 9 (26%) of samples, respectively. SamSPECTRAL could distinguish this population in 27 (79%) samples including all the ones that were identified by FLAME and flowMerge. In the 7 (21%) samples that SamSPECTRAL failed, the rare population of interest contained less than 0.15% of all data points.

To measure the accuracy of SamSPECTRAL, I define sensitivity and specificity as follows. For each sample, I call a cell positive if it belongs to the rare population of interest, and it is negative otherwise. Sensitivity is defined to be the number of truly identified rare cells divided by the total number of rare cells. Accordingly, specificity is the number of cells identified as negative divided by the total number of truly negative cell. The 27 (79%) cases where SamSPECTRAL correctly identified the rare population, had a 0.83 mean sensitivity with a 0.26 standard deviation. The median sensitivity was .99.
2.3. Results

Figure 2.7: Comparative identification of a low density population surrounded by much denser populations in the stem cell data set. (ac) SamSPECTRAL correctly identified the blue, low density population, while (d-f) flowMerge merged it to the yellow, high density population. (g-i) FLAME merged it to the red population. (j-l) The outcome of our modified MCL was similar to that obtained by SamSPECTRAL using classic spectral clustering. This shows that SamSPECTRAL is extensible by substituting classic spectral clustering with other clustering algorithms for weighted graph.
2.3. Results

Figure 2.8: Rare population in the stem cell data set. (a-c) This is a typical sample from the stem cell data set that contains a rare population. In these three dimensional plots, the red dots represent the cells that are positive for all three markers. Only 23/9721 (0.24%) events belong to this population in this sample. SamSPECTRAL could properly identify the rare population in 27/34 (79.4%) samples from the stem cell data set.
Specificity was 1 except for one sample. If I consider the samples with a rare population bigger than 0.2% of the total data, I obtained median=1, mean=0.93 and standard deviation of 0.15 for sensitivity. A detailed report of the results of this experiment is provided as a table in Appendix I. Table 2.1 summarizes the results from Sections 2.3.1–2.3.5.

2.3.6 SamSPECTRAL with MCL

Figure 2.7j-l depicts the output of MCL on a sample from stem cells dataset. We ran MCL on the sampled data obtained by our faithful sampling algorithm and then the post-processing step was applied to the resulting clusters. This experiment showed there was no significant difference for SamSPECTRAL in clustering either through computing eigenvectors (Figure 2.7a-c) or by MCL (Figure 2.7j-l). The CD45+ cells that are considered as outliers by MCL are not plotted in Figure 2.7j-l.

2.3.7 FlowCAP-I Competition

Table 2.2 summarizes the results of FlowCAP-I competition. As highlighted in Table 2.2, SamSPECTRAL was in the 5th rank in Challenge 1 based on the Rank Score criteria. As explained in this chapter of my thesis, SamSPECTRAL algorithm is based on the spectral clustering algorithm. A drawback of spectral clustering in general is that it can result in unsatisfactory outcomes if the similarity graph (controlled by $\sigma$) is not chosen with care. In other words, spectral clustering cannot serve as a “black box” which produce good results for any given dataset using the same $\sigma$ value, but instead its scaling factor parameter $\sigma$ needs to be adjusted for each dataset separately [169]. In Appendix H, we proposed a method to adjust $\sigma$ for any given dataset. Given this constraint of the spectral clustering and the fact that in Challenge 1 of FlowCAP-I competition, the parameters of algorithms were not allowed to be adjusted for each dataset separately, SamSPECTRAL algorithm could not produce satisfactory results in that challenge, and it ranked the 5th based on the Rank Score criteria (highlighted in Table 2.2). Note that the parameter setting requirement might or might not be an issue for other clustering algorithms participated in Challenge 1 of FlowCAP-I competition, depending on the nature of the algorithms and how their parameter settings were designed.

In Challenge 2 where the parameters could be tuned for each dataset separately, SamSPECTRAL results improved significantly, and it was scored as the second highest ranked clustering algorithm. Only ADICyt algorithm
Table 2.1: Summarizing the comparison between performance of SamSPECTRAL and those of flowMerge and FLAME as presented in sections 2.3.1–2.3.5

<table>
<thead>
<tr>
<th></th>
<th>Performance</th>
<th>Ref</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Stem Cells</td>
<td>Telomere</td>
</tr>
<tr>
<td>Challenging Problem 1:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SamSPECTRAL</td>
<td>–</td>
<td>✓</td>
</tr>
<tr>
<td>flowMerge</td>
<td>–</td>
<td>✓</td>
</tr>
<tr>
<td>FLAME</td>
<td>–</td>
<td>✓</td>
</tr>
<tr>
<td>Challenging Problem 2:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SamSPECTRAL</td>
<td>–</td>
<td>✓</td>
</tr>
<tr>
<td>flowMerge</td>
<td>–</td>
<td>×</td>
</tr>
<tr>
<td>FLAME</td>
<td>–</td>
<td>×</td>
</tr>
<tr>
<td>Challenging Problem 3:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SamSPECTRAL</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>flowMerge</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>FLAME</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Challenging Problem 4:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SamSPECTRAL</td>
<td>✓</td>
<td>–</td>
</tr>
<tr>
<td>flowMerge</td>
<td>×</td>
<td>–</td>
</tr>
<tr>
<td>FLAME</td>
<td>×</td>
<td>–</td>
</tr>
<tr>
<td>Challenging Problem 5:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SamSPECTRAL</td>
<td>79%</td>
<td>–</td>
</tr>
<tr>
<td>flowMerge</td>
<td>26%</td>
<td>–</td>
</tr>
<tr>
<td>FLAME</td>
<td>32%</td>
<td>–</td>
</tr>
</tbody>
</table>

Challenging Problem 1: Overlapping Populations
Challenging Problem 2: Subpopulations of a Population
Challenging Problem 3: Non-Elliptical Shaped Populations
Challenging Problem 4: Low Density Populations Close to Dense Populations
Challenging Problem 5: Rare Populations
✓: Satisfactory Results
✗: Unsatisfactory Results
2.3. Results

slightly outperformed SamSPECTRAL based on the Rank Score (Rank Score of 34 for ADICyt compared to 31 for SamSPECTRAL), however, the running time of ADICyt was substantially higher than that of SamSPECTRAL (04:50:37 for ADICyt compared to 00:06:47 for SamSPECTRAL). In Challenge 3, SamSPECTRAL and ADICyt shared the highest Rank Score (26.2), while again, SamSPECTRAL produced the results in less time (00:10:49 for ADICyt compared to 00:02:30 for SamSPECTRAL). The running time of ADICyt in Challenge 3 (00:10:49) was considerably lower than that of Challenge 2 (04:50:37). In order to investigate the reason behind this substantial reduction of the running time of ADICyt algorithm in Challenge 3 compared to Challenge 2, it worths highlighting the fact that two datasets WNV and ND were included in Challenge 2, but excluded from Challenge 3 of FlowCAP-I competition. WNV contained around 10 fold higher number of cells per sample (100,000 cells) and relatively higher number of dimensions (6 dimensions) compared to those of GvHD, DLBCL and HSCT datasets (Table A.1). ND was also a high-dimensional dataset that had the highest number of dimensions (10 dimensions) among all datasets tested in FlowCAP-I. Therefore, I hypothesized that the considerable reduction of the average running time of ADICyt in Challenge 3 compared to Challenge 2 was due to excluding these two high-dimensional datasets (WNV and ND). This in turn shows that ADICyt is slow for FCM samples with high number of dimensions. However, this needs to be confirmed through studying the time complexity of ADICyt in regard with data dimensionality. Unfortunately, this commercial clustering algorithm is unpublished yet. This prevents us from studying it in more detail. In comparison to ADICyt, the average running time of SamSPECTRAL was relatively low for both Challenge 2 (00:06:47) and Challenge 3 (00:02:30), confirming that its speed remained high with increasing the data dimensionality. In summary, the SamSPECTRAL outperformed other published approaches that were submitted in the FlowCAP-I competition in Challenges 2 and 3 based on the Rank Score. Keeping in mind that the running time of SamSPECTRAL is low (order of minutes), SamSPECTRAL can be considered as an efficient clustering algorithm that optimises speed and quality of the clustering, and has improved the field in this regard.

Table 2.2 demonstrates that the F-Measure was slightly lower, slightly higher, and substantially higher in Challenge 3 compared to that of Challenge 2 for GvHD, DLBCL and HSCT datasets, respectively. This indicates that providing the number of clusters to SamSPECTRAL can improve the results in some cases, but not all of them. Therefore, my overall recommendation would be to use the approach we developed in Section 2.2.4 to
2.3. Results

estimate the number of clusters automatically, instead of being dependent to the user-defined number of clusters.
## Table 2.2: Results of the FlowCAP-I challenges

<table>
<thead>
<tr>
<th>Method</th>
<th>Challenge 1</th>
<th>Challenge 2</th>
<th>Challenge 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>crema</td>
<td>0.81 (0.72, 0.88)</td>
<td>0.81 (0.71, 0.81)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>DHCIL</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.91 (0.91, 0.92)</td>
</tr>
<tr>
<td>HSCIL</td>
<td>0.89 (0.88, 0.90)</td>
<td>0.89 (0.89, 0.90)</td>
<td>0.90 (0.90, 0.92)</td>
</tr>
<tr>
<td>Overall</td>
<td>0.89 (0.89, 0.90)</td>
<td>0.89 (0.89, 0.90)</td>
<td>0.90 (0.90, 0.92)</td>
</tr>
<tr>
<td>Tukey's test</td>
<td>04:50:37</td>
<td>05:14:37</td>
<td>02:44:37</td>
</tr>
<tr>
<td>Cohen's d</td>
<td>52</td>
<td>21</td>
<td>44</td>
</tr>
<tr>
<td>Overall</td>
<td>Student's t</td>
<td>Student's t</td>
<td>Student's t</td>
</tr>
</tbody>
</table>

### Challenge 1: Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Challenge 1</th>
<th>Challenge 2</th>
<th>Challenge 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>crema</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>DHCIL</td>
<td>0.90 (0.90, 0.91)</td>
<td>0.90 (0.90, 0.91)</td>
<td>0.90 (0.90, 0.91)</td>
</tr>
<tr>
<td>HSCIL</td>
<td>0.91 (0.91, 0.92)</td>
<td>0.91 (0.91, 0.92)</td>
<td>0.91 (0.91, 0.92)</td>
</tr>
<tr>
<td>Overall</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>Tukey's test</td>
<td>04:50:37</td>
<td>05:14:37</td>
<td>02:44:37</td>
</tr>
<tr>
<td>Cohen's d</td>
<td>52</td>
<td>21</td>
<td>44</td>
</tr>
<tr>
<td>Overall</td>
<td>Student's t</td>
<td>Student's t</td>
<td>Student's t</td>
</tr>
</tbody>
</table>

### Challenge 2: Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Challenge 1</th>
<th>Challenge 2</th>
<th>Challenge 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>crema</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>DHCIL</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>HSCIL</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>Overall</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>Tukey's test</td>
<td>04:50:37</td>
<td>05:14:37</td>
<td>02:44:37</td>
</tr>
<tr>
<td>Cohen's d</td>
<td>52</td>
<td>21</td>
<td>44</td>
</tr>
<tr>
<td>Overall</td>
<td>Student's t</td>
<td>Student's t</td>
<td>Student's t</td>
</tr>
</tbody>
</table>

### Challenge 3: Results

<table>
<thead>
<tr>
<th>Method</th>
<th>Challenge 1</th>
<th>Challenge 2</th>
<th>Challenge 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>crema</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>DHCIL</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>HSCIL</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>Overall</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
<td>0.93 (0.93, 0.95)</td>
</tr>
<tr>
<td>Tukey's test</td>
<td>04:50:37</td>
<td>05:14:37</td>
<td>02:44:37</td>
</tr>
<tr>
<td>Cohen's d</td>
<td>52</td>
<td>21</td>
<td>44</td>
</tr>
<tr>
<td>Overall</td>
<td>Student's t</td>
<td>Student's t</td>
<td>Student's t</td>
</tr>
</tbody>
</table>

# 2.3. Results
2.3. Results

\(^a\) Mean and 95 percent confidence intervals for the F-Measures, and Rank Scores were calculated as described in Appendix D.

\(^b\) Runtime is calculated as time per CPU per sample.

\(^c\) In each dataset/challenge, the top algorithm (highest mean F-measure) and the algorithms with overlapping confidence intervals with the top algorithm are bolded.

\(^d\) Algorithms are sorted by rank score within each challenge.

\(^e\) These algorithms are unpublished.
2.4 Discussion

Although spectral clustering algorithm is a powerful technique, it cannot be directly applied to large datasets as it is computationally expensive both in time and memory. In this study, we developed a sampling method and combined it with spectral clustering by modifying the similarity matrix based on potential theory. As a result, for the first time, analysing FCM data using spectral methods becomes possible and practical. We applied SamSPECTRAL to four different FCM datasets to demonstrate its applicability on a broad spectrum of FCM data, and compared its performance to two state of the art model-based clustering methods optimised for FCM data.

Detecting rare populations is a challenging problem and in spite of its significant applications in medical and biological research, little progress has been achieved in automatic identification of such populations. Our data reduction scheme is delicate enough not to miss rare populations comprising between 0.2% to 2% of the total data. SamSPECTRAL can identify populations of relative size in this range with acceptable accuracy.

Since SamSPECTRAL, is a multidimensional clustering approach, it can identify overlapping populations that are generally hard to identify by manual gating that uses sequential two dimensional visualizations of the data. SamSPECTRAL is the first method that has demonstrated the ability to correctly identify subpopulations of major FCM cell populations.

An important challenge in analysing FCM data is in clustering data files that contain populations that significantly differ in density. Model-based techniques can produce errors in identifying a low density population close to denser populations because they typically make assumptions on the density of clusters [40]. Our experiments demonstrated that SamSPECTRAL can properly tackle this problem.

Besides the practical observations, this capability is justified by the following observation. Spectral methodology clusters the graph such that the normal cut is “almost” optimum [43]. Now, assume that it can distinguish between two clusters when their densities are comparable. Then, if the size of the smaller cluster is reduced without change in its shape or distribution, the normal cut between them remains similar because the number of vertices and edges reduces almost proportionally to each other. Therefore, the clusters remain distinguishable. This explains why the overall performance of SamSPECTRAL is independent of cluster densities as long as their shapes are preserved.

Since parametric methods such as FLAME and flowMerge make a priori assumptions on the distribution or shape of the clusters [40, 69, 135],
they may fail in identifying populations with arbitrary shapes. Although flowMerge attempts to solve this issue by finding more clusters than needed and then merging them together, it does not produce satisfactory results when the shape of the cluster is complex. SamSPECTRAL has the capability of identifying arbitrary shape clusters since it is a non-parametric approach that makes no assumptions on the shape and distribution of clusters, and clusters data based only on similarity between data points.

In order to apply SamSPECTRAL to FCM data samples, users need to manually adjust the parameters only once by running SamSPECTRAL on one or two random samples from a FCM data set. Then, the algorithm can be run on the rest of data set without changing the parameters. Adjusting parameters including the scaling factor and the separation factor, as well as the method for estimating the number of clusters impact the final results. This may impose difficulties for users of SamSPECTRAL in setting the parameters manually. On the other hand, in the applications of most clustering algorithms including SamSPECTRAL, applying automated gating using a single set of parameters identified initially may result in a set of clusters that model some populations well, but cannot model other desired populations correctly. Therefore, in this study, I tried to reduce the sensitivity of the final results to the parameter setting step of the clustering algorithm as follows. I hypothesized that if the clustering algorithm was run multiple times with different initial values for the parameters (e.g. number of clusters), forming a set of clusters, then one could find the desired populations among this set reliably without being too concerned with finding the optimal initial set of parameters. This is because this approach increases the probability that all possible clusters are found in the cluster set, without needing to spend a significant amount of time adjusting the parameters for the clustering algorithm. I tested this hypothesis by developing a learning-based cluster matching method that searched for desired populations among all clusters generated by multiple SamSPECTRAL runs with different sets of initial parameters. This approach is explained in detail in Chapter 3 of my thesis. Using this method, the user does not have to find the optimal values for the parameters in order to identify target populations accurately. In other words, this approach reduces the sensitivity of the final results to the initial values of the parameters.

Not only does our sampling scheme increase the speed of spectral clustering without losing important biological information, but the resulting algorithm is faster than other methods considered in this study. More precisely, the running time of SamSPECTRAL is $O(dmn) + O(m^3)$ where $O(dmn)$ is the running time for building $m$ communities from $n$ points in $d$ dimension.
2.5. Conclusions

and $O(m^3)$ is the running time for computing the eigenspace. After this step, the k-means clustering runs very fast in time $O(k m t)$ to find $k$ clusters using eigenvectors in $t$ iterations. In practice, we set an upper bound for $t$ (e.g. 200) to limit the number of iterations. In comparison, the time complexity of the original MCL method is $O(nr^2)$ with no guarantee on upper bound for number of iterations $r$, other than $n$. Practically, for our model of FCM data where all pairs of data points are connected, we could not run MCL before applying our modification to it. Moreover, SamSPECTRAL running time is generally less than model-based techniques. The running time of flowMerge is $O(d^2k^2nt)$ and FLAME runs in time $O(d^4klnt)$ where $l$ is the number of times it runs to find the optimal number of clusters. In practice, we can keep $m$ as small as 1500-3000 without loosing important biological information, and consequently SamSPECTRAL ran at least 5-20 times faster than flowMerge on the studied datasets (Table 2.2). Furthermore, the time efficiency of my algorithm is more noticeable for higher dimensional data such as the one explained in Appendix I. This sample contains 100,000 events in 23 dimensions and SamSPECTRAL can analyze it in less than 25 minutes by a 2.7GHz processor.

2.5 Conclusions

Faithful sampling is based on potential theory. It reduces the size of input for spectral clustering algorithms and consequently they can now be efficiently applied on FCM data in spite of its large size. Practically, our approach demonstrated significant advantages in proper identification of populations with non-elliptical shapes, low density populations close to dense ones, minor subpopulations of a major population, rare populations, and overlapping populations. No state-of-the-art method can solve all the challenges in identifying populations with the above properties simultaneously. Moreover, applying SamSPECTRAL to other biological data such as microarrays and protein databases may result in significant improvements in gene expression and protein classification.

Our faithful sampling algorithm also has interesting applications by itself. For instance, it can be used appropriately to reduce the size of input for other clustering algorithms that are based on spectral graph theory such as Markov Clustering Algorithm (MCL), electrical circuit based clustering, and agent based graph clustering [9]. We have shown the extendibility of our approach in this sense by substituting classic spectral clustering with MCL, a method that has many applications in bioinformatics.
2.5. Conclusions

Other directions for future work include applying other schemes for estimating similarities between communities and combining clusters based on other combinatorial algorithms or biological criteria.
Chapter 3

Semi-Automated Pipeline for Analysis of Innate Immune Response Flow Cytometry Data

3.1 Background

Analysis of FCM data includes: (1) identification of homogeneous cells with similar properties and functions; and (2) finding the association between the features of the identified populations and the clinical conditions. While homogeneous cell populations within a sample can be identified by a growing number of automated clustering algorithms [4, 18, 90, 113, 126, 135, 146, 158, 172, 179], in most FCM applications there is a need for matching similar cell populations between different FCM samples. This key difference permits reliable comparison of phenotypically identical cell populations between different samples in order to derive the biological significance of experimental or clinical differences.

Here, I developed a novel learning-based cluster matching method that searches for the best matches of the desired populations among all clusters generated by a clustering algorithm. My cluster matching is a 2-tiered semi-automated method that incorporates domain expert knowledge to identify target cell populations as follows: (1) All FCM samples are clustered by running a clustering algorithm multiple times to form a cluster set; (2) Desired populations of at most 6 randomly selected training samples are identified manually by the domain expert, and the features of the target populations in these samples are measured; (3) The cluster with the highest likelihood ratio is then returned as the best match of the desired population using feature comparison with the domain expert defined set. Since my method matches only one cluster to the target population for each FCM sample, it overcomes a cluster matching problem commonly observed in metach clerusting approach.
3.1. Background

[135], where adjacent cell populations are not resolved independently. Since the set of possible clusters for each sample are obtained by applying the clustering algorithm multiple times with different initial values, the optimal cluster is selected from a comparatively large number of possibilities per sample. Accordingly, my approach increases the probability that the selected cluster is in fact the best representative of the target population, while being flexible in selecting the unique descriptive features for every target population in a complex dataset. The functional consequence of this is that the user can decide on a wide variety of features—including population shape, size, and location—for each target population independently from the features of other cell populations.

In this study, I have incorporated my proposed learning-based cluster matching method into a semi-automated analysis pipeline in order to fully analyse FCM data. I present here a use-case analysing innate immune response FCM data. My pipeline includes: (1) data preparation and preprocessing, (2) automated gating using the SamSPECTRAL clustering algorithm, (3) learning-based cluster matching to target antigen-presenting cell (APC) populations, and (4) quantifying the functional—cytokine—responses of defined cell populations. Pre-defined and well established target cell populations identified in this use-case include three main APC subpopulations: monocytes (MC), conventional dendritic cells (cDC), and plasmacytoid dendritic cells (pDC). My pipeline is also capable of automatic quantification of the cytokine responses of said APC populations to stimulation by toll-like receptor ligands.

Here, I have fully analysed innate immune response FCM data semi-automatically. Given the remarkable pattern-recognition capability of the human neocortex, humans regularly—and with remarkable reliability for a non-mechanical system—identify and describe cell populations in multiparametric space. While the ultimate goal of automated flow-cytometric analysis of cells is a perfectly reproducible and fully automatic definition of cell populations and cellular processes, we can not avoid the fact that current fully-*in silico* approaches fail to cope with the variability and complexity inherent in today’s FCM datasets. Therefore, in order to verify the performance of my semi-automated pipeline, I have compared my semi-automated identification of APC populations to domain-expert manual gating (the current state of the art). In addition, the functional responses of cytokine producing cells measured automatically are compared against those measured manually. My results show that my semi-automated pipeline can reproduce the results obtained from manual analysis with high accuracy both in identification of APC populations and measuring the functional response of
cytokine producing cells. Based on analysis presented here, I propose that my semi-automated analytic pipeline can replace manual analysis entirely. This will mean a tremendous time-saving for FACS users, with no discernible quality loss.

3.2 Data and Wet Lab Description

Dr. Tobias Kollmann’s lab from BC Children Hospital provided the FCM data that were obtained from the analysis of peripheral blood mononuclear cells (PBMC) from neonates at birth, 1 year old, and 2 years old, as previously described [45, 95]. Briefly, PBMC from 12 subjects were purified using Ficoll-Hypaque density centrifugation. 500,000 cells in 200 ul of RPMI supplemented with 10% human AB serum and 1% penicillin-streptomycin were plated on premade 96-well plates containing 10 ul of TLR ligands (final concentrations of the ligands as follow: PAM3CSK4 at 1 µg/ml; LPS at 100 ng/ml; 3M-003 at 10 µM, and; CpG-A (ODN-2216) at 25 µg/ml). For each subject, two wells were left unstimulated as controls. The cells plated for the intracellular staining (ICS) were cultured for 6 hours in the presence of the Golgi transport inhibitor Brefeldin A (BFA) at 10 µg/ml with the exception of the wells containing the CpG-A ligand. The cells stimulated with the CpG-A ligand were cultured for 3 hours without BFA followed by an additional 3 hours in the presence of BFA. At the 6-hr mark, cells were treated with EDTA (final concentration of 2 mM), spun down, and the pellet resuspended in 100 µl of Becton-Dickinson (BD) FACS Lysing Solution and then frozen in -80°C until antibody staining and cytometric analysis using a BD LSRII instrument [95]. For each FCM data sample, the forward scatter (FSC), side scatter (SSC), 4 surface proteins markers: MHCII, CD14, CD123, CD11c, and 4 cytokine markers: IL-12, IL-6, TNF-α, IFN-α were measured. Data from the LSRII instrument were analysed in two ways: (1) Manually by FlowJo (TreeStar, Inc.) and CPAS (LabKey Solutions) [152]; and (2) Using my novel semi-automated method that I present here.

Please note that the members of Dr. Tobias Kollmann’s lab including Darren Blimkie and Edgardo Fortuno performed the manual analysis of the data. I developed computational methods for semi-automated analysis of FCM data. Also the semi-automated analysis was completely performed by myself, except those parts which required human interactions, including manual gating of training samples, removing dead cells and debris, and compensating FCM data.
3.3 Manual vs. Semi-Automated Analysis

In the manual analysis, FlowJo was used to manually identify antigen presenting cell populations including conventional dendritic cells (cDC), plasmacytoid dendritic cells (pDC), and monocytes and also the positive cells for each population. First, dead cells and cell debris were excluded by gating them out from an FSC vs. SSC plot (Figure 3.1). From this “live” gate, the cells on MHC-II vs. CD14 were plotted. CD14\(^{\text{high}}\), and MHC-II\(^{+}\) cells were identified as monocytes. By plotting the CD14\(^{\text{low/neg}}\), MHC-II\(^{+}\) cells on CD123 vs. CD11c axes, cDCs were identified as the CD11c\(^{+}\) population while the pDCs were identified as CD123\(^{+}\). Next, the monocytes, cDC and pDC were graphed on scatter plots with the fluorescence intensity for each of the cytokines as the axes. For each subject, the quadrants inside these scatter plots which separated the cytokine-producing cells from cytokine-negative ones were manually determined based on the cells that were not stimulated with any TLR ligand.

Unlike the manual analysis described above, in the semi-automated method that I developed here, most of the analysis—including identification of antigen presenting cell populations and cytokine-positive cells—were done automatically. In the discussion of this chapter, I explain in detail which steps of the analysis were completely automated, and which steps were semi-automated with human interactions. My semi-automated analysis pipeline is best illustrated via the explanation of the following steps: (1) data preparation and preprocessing; (2) automated gating using SamSPECTRAL clustering algorithm; (3) learning-based cluster matching for finding the best matches of target cell populations and labeling APC populations; (4) measuring functional response of cytokine-positive cells.

3.4 Data Preparation and Preprocessing

This part of the semi-automated analysis pipeline included the following steps: (1) removing dead cells and debris; (2) compensation and transformation; and (3) normalization.

3.4.1 Removing Dead Cells and Debris

In this study, FlowJo software was used to manually identify dead cells and debris [151]. Here, dead cells and cell debris were excluded by gating them out from forward scatter (FSC) vs. side scatter (SSC) plot (Figure 3.1). Then the live gate coordinates were extracted from the FlowJo, and
3.4. Data Preparation and Preprocessing

Figure 3.1: Manual identification of antigen-presenting cell populations (monocytes, other APCs, cDC and pDC), and the identification of the cytokine-producing cells within cDC, pDC and monocytes. Each axis presents the intensity of the measured parameter.
applied to the FCM files using R BioConductor to identify all live cells to be computationally analysed in R.

3.4.2 Compensation and Transformation

The next step in preprocessing the FCM data was to compensate the FCM files. FCM data samples were compensated manually in FlowJo using matched isotype control samples. I extracted the resulting compensation matrices from FlowJo, and used them to compensate FCM files in R using the FlowCore package [85]. After compensation, all FCM data dimensions were transformed except forward scatter (FSC) and side scatter (SSC) using logicle transformation. FSC and SSC were left untransformed.

3.4.3 Normalization

All dimensions of the data that corresponded to cell surface proteins underwent sequential normalizations, using the formula $y = \frac{x - \bar{x}}{\sigma}$, where the desired cell surface protein intensity is shown by $x$. $\bar{x}$ and $\sigma$ are mean and standard deviation of cell surface protein intensities respectively, and $y$ is the normalized cell surface protein intensity.

In the FCM samples, the intensities of cell surface proteins were used to identify cell populations and assign biological labels to them. For identification of cell populations, the actual values of the expression of cell surface proteins as measured by flow cytometers are not important, but instead what matter the most are the relative locations of populations. Therefore, the normalization helped to improve cluster matching step of the automated analysis where I tried to match the clusters across different samples. Unlike cell surface proteins, normalization of the expression of intracellular cytokines using $y = \frac{x - \bar{x}}{\sigma}$ formula results in missing the information regarding the level of functional response of cytokine producing cells. Therefore, the intracellular cytokine expression was left unchanged, with no normalization.

3.5 Automated Gating Using SamSPECTRAL Clustering Algorithm

One important stage of the semi-automated analysis of the immune response FCM data was identification of antigen presenting cell (APC) populations based on the level of intracellular or surface proteins expression of the cells. Three main APC populations, monocytes, conventional dendritic cells (cDC) and plasmacytoid dendritic cells (pDC) formed the target populations of the
3.5. Automated Gating Using SamSPECTRAL Clustering Algorithm

In the current study, I faced two main challenges. The first were cases in which the density of the target population was very low compared to that of adjacent populations. For instance, 3M-002 stimulation usually resulted in decreasing the density and frequency of monocytes, such that monocyte density and frequency become very low compared to those of its adjacent MHCII$^+$CD14$^-$ population. This made separating monocytes from MHCII$^+$CD14$^-$ population a challenging task. The second challenge was identification of pDCs. pDCs are a rare cell type that constitutes a very small fraction (around 0.2 to 0.4%) of PBMC\cite{111,165}. Both manual and automated identification of rare cell populations are recognized as very challenging tasks by the FCM and flow informatics communities\cite{115,135,179}.

In Chapter 2, I presented a method called SamSPECTRAL that addresses the above challenges, and showed its superiority over model-based clustering algorithms in identification of rare cell populations and low density populations surrounded by high density ones. Because of these advantages of SamSPECTRAL clustering algorithm, I used it in the current study to find the cell populations of the FCM samples.

SamSPECTRAL could have been run using all 4 dimensions that correspond to the cell surface proteins, but in order to be consistent with the manual analysis, I decided to apply the algorithm sequentially, 2 parameters at a time. First I applied SamSPECTRAL to each FCM sample using MHCII and CD14, to identify two populations, (1) MHCII$^+$CD14$^+$ population, which corresponds to monocytes, and (2) MHCII$^+$CD14$^-$ population of Other APCs. SamSPECTRAL parameters were $\sigma = \frac{1}{500}$, sep.factor = 0.7, and $15 \leq k \leq 30$, where, $\sigma$, sep.factor, and $k$ stand for scaling parameter, separation factor, and number of clusters, respectively. These parameters were adjusted by piloting them on a few samples, before the result was used for all FCM samples\cite{178,179}. Running SamSPECTRAL multiple times with different numbers of clusters ($k = 15, 16, ..., 30$) resulted in an aggregate set of potential clusters. Experimentally testing on few training samples ($< 6$), I found that the selected range for the number of clusters was wide enough to produce a comprehensively large number of possible clusters per sample. I used my novel learning-based cluster matching method to find the best match of monocytes and Other APCs among all clusters of this set.

I then applied the SamSPECTRAL algorithm to the MHCII$^+$CD14$^-$ matched clusters, in order to identify two main sub-populations therein, (1) MHCII$^+$CD14$^-$CD123$^-$CD11c$^+$ population that corresponds to cDCs, and (2) MHCII$^+$CD14$^-$CD123$^+$CD11c$^-$ population that corresponds to pDCs. Here, the SamSPECTRAL parameters were the same as before, with the
3.6 Learning-Based Cluster Matching for Labeling APC Populations

2 dimensions used for clustering being CD123 and CD11c. Like previous stage, I used my learning-based cluster matching method to find the best match of cDCs and pDCs among all clusters of the set that resulted from running SamSPECTRAL multiple times with different numbers of clusters \((k = 15, 16, ..., 30)\).

3.6 Learning-Based Cluster Matching for Labeling APC Populations

Applying clustering algorithms to an FCM sample results in generating a set of clusters. Now, one important question is how to match the clusters across different samples, and how to assign biologically meaningful labels to the clusters. Also sometimes it is desired to find some well-established target populations (e.g., monocytes) in all FCM samples. Here, I developed a novel method to find the best matches of target populations among all clusters generated by clustering algorithms, and assign biologically meaningful labels to them.

3.6.1 Definitions

Here, I first formally define some terms that I frequently use in this part of my thesis.

**FCM Data Sample:** An FCM data sample \(D\) can be defined as a set of \(n\) \(d\)-dimensional data points, \(D = \{p_i | p_i \in \mathbb{R}^d, 1 \leq i \leq n\}\). Here, each data point \(p_i\) corresponds to one cell, and each dimension corresponds to one parameter of the cell.

**Target Population:** A target population \(P\) is a subset of FCM data sample \((P \subseteq D)\), such that, all of its members have distinct biological functions and characteristics that make them distinguishable from the rest of data points (or cells). Target populations (e.g., monocytes, cDC, and pDC) are identified by biology experts through manual gating.

**Cluster:** A cluster \(C\) is a set of \(d\)-dimensional data points, such that \(C \subseteq D\), and its members are similar to each other based on a predefined metric. The clusters are generated by applying a clustering algorithm (e.g., SamSPECTRAL) to \(D\).
3.6. Learning-Based Cluster Matching for Labeling APC Populations

Note that although a *cluster* and a *population* are both subsets of FCM data samples, I distinguish between these two terms. In fact in my definitions, a *population* has a biological meaning (all of its members have distinct biological functions and characteristics), while a *cluster* is simply the output of an automated method called clustering.

**Similarity of a Cluster and a Population:** I call a cluster \( C \) is similar to a population \( P \) if their features (e.g., centers or frequencies) are similar under a distance metric. Here, the cluster \( C \) and the population \( P \) are not necessarily subsets of the same FCM sample, that is, it is possible that \( C \subseteq D_1, P \subseteq D_2 \), and \( D_1 \neq D_2 \).

**Cluster Set:** A cluster set \( E \) is a set that its members are clusters of an FCM data sample, that is, \( E = \{ C_i | C_i \subseteq D \} \).

**Training and Non-Training Data Samples:** In my study, each FCM data sample was either used for the purpose of training, or used for testing the performance of my algorithm. All target populations of each training sample were identified manually by a biology expert. In comparison, the assumption was that the target populations were not identified manually for the non-training samples, and they were instead aimed to be approximated by using an automated method.

### 3.6.2 Problem Statement

I initially defined the problem as follows.

**Problem 1:**

*Given*

- a non-training data sample \( D \), and
- a target population \( P \) identified manually \( (P \subseteq D) \);

*Find*

- a cluster \( C_1 \) automatically, such that \( C_1 \subseteq D \), and the cluster \( C_1 \) is highly similar to the target population \( P \).

As mentioned before, the target populations were assumed to be not identified manually for non-training data samples, and in fact I aimed to identify them automatically. Therefore, I revised the above problem as follows.
3.6. Learning-Based Cluster Matching for Labeling APC Populations

Problem 2:

Given
- a non-training data sample $D$,
- training data samples $D_1', D_2', ..., D_s'$, and
- the corresponding target populations $P_1', P_2', ..., P_s'$ which are identified manually for all training data samples ($P_i' \subseteq D_i'$ for $1 \leq i \leq s$);

Find
- a cluster $C_2$ automatically, such that $C_2 \subseteq D$, and $C_2$ is highly similar to $P_1', P_2', ..., P_s'$.

Assuming that the corresponding target populations of different FCM samples show some levels of similarity, one can conclude that the target population $P$ of FCM sample $D$ (if identified manually) would be similar to populations $P_1', P_2', ..., P_s'$ of the samples $D_1', D_2', ..., D_s'$. This leads to the conclusion that the cluster $C_2$ that is found in Problem 2 approximates the cluster $C_1$ that would be found in Problem 1, if $P$ was identified manually.

3.6.3 My Approach

For each FCM sample, SamSPECTRAL clustering algorithm was run multiple times with different number of clusters (e.g., $k = 15, 16, ..., 30$) to obtain an ensemble of clusterings, $M^{(1)}, M^{(2)}, ..., M^{(R)}$. Here, each $M^{(i)}$ was a cluster set with $k_i$ members, $M^{(i)} = \{c_1^{(i)}, c_2^{(i)}, ..., c_{k_i}^{(i)}\}$. The members of $M^{(i)}$ were the clusters that were generated by the $i^{th}$ run of SamSPECTRAL algorithm, where number of clusters was $k_i$. A new cluster set was built by putting all members of the cluster sets $M^{(1)}, M^{(2)}, ..., M^{(R)}$ into one set.

$$Cluster\ Set = \{c_1^{(1)}, c_2^{(1)}, ..., c_{k_1}^{(1)}, ..., c_1^{(R)}, c_2^{(R)}, ..., c_{k_R}^{(R)}\} \tag{3.1}$$

Then for each target population (e.g., cDC), my method searched the $Cluster\ Set$, and looked for the cluster that best matched to the target population of interest.

3.6.4 Incorporating Biology Expert Knowledge

In order to compare every cluster of the $Cluster\ Set \tag{3.1}$ against the target population to find the cluster that best matches to it, first it was needed to get some sense of how the target population looked like, or in other words, what characteristics and features it had. To this aim, a biology expert was asked to manually find all target populations (monocytes, Other APCs, cDCs and pDCs) on at most six randomly selected FCM samples.
3.6. Learning-Based Cluster Matching for Labeling APC Populations

These samples formed my training set from which the features of the target populations were extracted.

Table 3.1: Target populations features that are used in my cluster matching method.

<table>
<thead>
<tr>
<th>Target Population</th>
<th>Features</th>
</tr>
</thead>
</table>
| Monocyte          | Population frequency  
|                   | Position: MHCII (lq), CD14 (med, diff)  
|                   | Average of local densities |
| Other APCs        | Population frequency  
|                   | Position: MHCII (lq), CD14 (med, diff)  
|                   | Average of local densities |
| cDC               | Population frequency  
|                   | Position: CD123 (uq), CD11c (lq)  
|                   | Average of local densities |
| pDC               | Population frequency  
|                   | Position: CD123 (med, diff), CD11c (med, diff)  
|                   | Average of local densities |

lq: lower quartile of the marker fluorescence intensity  
med: median of the marker fluorescence intensity  
uq: upper quartile of the marker fluorescence intensity  
diff: difference between upper quartile and lower quartile of the marker fluorescence intensity  
All values of the markers fluorescence intensities are given after transformation and normalization.

In my method, the target populations’ features can include but are not limited to four main categories: (1) position of the population (e.g., the centre of population measured as mean fluorescence intensities); (2) size or frequency of the population; (3) shape of the population, and (4) the average of local density of the population. In this method, the features extracted for different populations do not have to be the same, and can vary depending on the population characteristics. Table 3.1 summarizes the features that were used for the target populations in this study. As shown in this table, population position, frequency and density features were used for all target populations. The population position features were individually adjusted for each population, considering the information obtained from the training set regarding the expected location of the population in the data space.
One important point in selecting the features of the target populations is that the values of the selected features should be similar across different samples. If the value of a population feature varies highly across samples, it should not be selected as the target population feature. However, there might be cases for which one can find groups of samples, such that the feature of interest is similar for samples within a group, but different for samples from different groups. In such cases, one can still use that feature, but first group the samples, and randomly select the training samples for each group of samples separately. One example of this is the monocyte population for which the population local density and frequency change by TLR-ligand stimulation. In Appendix F, I explain how the samples were grouped automatically, such that samples with similar monocyte features stayed in the same group.

### 3.6.5 Learning From Training Samples

For each training sample, the Cluster Set \((3.1)\) was built using the Sam-SPECTRAL algorithm. Then for each manually identified target population, \(p\), every cluster \(c \in \text{Cluster Set}\) was considered, one by one, and the overlap ratio was calculated as

\[
\text{overlap\_ratio}(p, c) = \frac{|p \cap c|}{|p \cup c|}
\]

(3.2)

where, \(p \cap c\) and \(p \cup c\) are intersection and union of \(p\) and \(c\), respectively, and \(|.|\) stands for the set cardinality. If \(\text{overlap\_ratio}(p, c) \geq 0.5\), \(c\) was put in the set \(S^{(p)}\). Otherwise, \(c\) was put in the set \(T^{(p)}\). For every target population \(p\), this was applied for all FCM samples of the training set to build two sets of the clusters, \(S^{(p)}\) and \(T^{(p)}\). Therefore, \(S^{(p)}\) included all training samples' clusters that were similar to the target population, and \(T^{(p)}\) included all training samples' clusters that were dissimilar to the target population.

After dividing all clusters of all training samples into two groups based on their similarities to the desired target population, the features of all clusters of \(S^{(p)}\) and \(T^{(p)}\) were measured. The measured features for \(p \in \{\text{monocyte, Other APC, cDC, pDC}\}\) are given in Table 3.1. The feature vector of each cluster of the set \(S^{(p)}\) was put in one row of a matrix called \(F_{S^{(p)}}\). In this way, a \(|S^{(p)}| \times m\) feature matrix was built, where, \(|S^{(p)}|\) was the number of clusters in the set \(S^{(p)}\), and \(m\) was the number of features measured for population \(p\). Similarly, a \(|T^{(p)}| \times m\) feature matrix (called \(F_{T^{(p)}}\)) was built from the cluster set \(T^{(p)}\).
3.6. Learning-Based Cluster Matching for Labeling APC Populations

Comparing the feature vector $v$ of a new cluster of a non-training FCM sample against the feature matrices $F_{S(p)}$ and $F_{T(p)}$, one could find out the level of similarity of the cluster feature to $F_{S(p)}$ or $F_{T(p)}$, or in other words, the level of similarity and dissimilarity of the new cluster to the target population $p$. One way to compare the feature vector $v$ of a cluster of a non-training FCM sample against the feature matrix $F_{S(p)}$ (or $F_{T(p)}$) was to simply find the average of distances between $v$ and rows of $F_{S(p)}$ (or $F_{T(p)}$) using a distance metric (e.g., Euclidean distance). However, my preliminary experiments showed that like [6], this simple measure of similarity was not successful enough in capturing the variabilities between the population’s features of different FCM samples. Therefore, I concluded that a more sophisticated probability-based method was required to provide more sensible and more accurate measure of similarities.

In [6], Aksoy et al. presented a likelihood-based similarity measure, and used it for an image-retrieval application. In their application, the objective was to measure similarity of two images (one being the query image and the other one being a database image), and to compute the likelihood of two images being similar or dissimilar. Their experiments showed that their likelihood-based measure of similarity significantly outperformed other commonly used geometric measures with the $L_p$ metric both in terms of precision and recall. Motivated by their findings, I developed a similar probability-based method to measure the similarity of a cluster to the target population $p$ using a likelihood ratio measurement. It is worth mentioning that in my approach, first the columns of $F_{S(p)}$ and $F_{T(p)}$ were linearly normalized, such that they always took values in the range $(0, 1)$. In this way, all features became comparable, and had similar weights.

### 3.6.6 Likelihood Ratio Measurement

Pair set $A^{(p)}$ was built, such that $\forall a, b \in S^{(p)}$, and $a \neq b$, I had $(a, b) \in A^{(p)}$. Similarly, the pair set $B^{(p)}$ was built such that, $\forall a \in S^{(p)}$, and $\forall b \in T^{(p)}$, we had $(a, b) \in B^{(p)}$. In this way, pair set $A^{(p)}$ contained pairs of training samples’ clusters that were similar to the target population $p$. In comparison, pair set $B^{(p)}$ contained training samples’ cluster pairs for which one cluster was similar to the target population $p$, while the other one was dissimilar to $p$.

Next, $\forall(a, b) \in A^{(p)}$, the distance vector $d$ was defined by $d = v_b - v_a$, where, $v_a$ and $v_b$ denoted the feature vectors of the clusters $a$ and $b$. Here, $d$ was a distance vector of length $m$, where, $m$ was the number of features measured for population $p$. In other words, $d$ presented the distance
between the feature vectors of two training samples clusters \( a \) and \( b \) that were both similar to the target population \( p \). Computing the distance vector for all pairs of the set \( A(p) \), a set of distance vectors was obtained, and the probability density function (pdf) was estimated for each element of these vectors separately. Similarly, the distance vectors for all pairs of the set \( B(p) \) were computed, and the kernel densities of all elements of the distance vectors of this set were estimated.

Assume that \( f^{(i)}(d) \) and \( g^{(i)}(d) \) represent the kernel density estimation of the \( i \)th element of the distance vector for class \( A \) and \( B \), respectively. The likelihood ratio was computed as follows:

\[
likelihood\_ratio(d) = \frac{\prod_{i=1}^{m} f^{(i)}(d)}{\prod_{i=1}^{m} g^{(i)}(d)} \tag{3.3}
\]

Taking the logarithm of the likelihood ratio, an alternative measurement called log likelihood ratio was obtained as follows:

\[
log\_likelihood\_ratio(d) = \log\left(\frac{\prod_{i=1}^{m} f^{(i)}(d)}{\prod_{i=1}^{m} g^{(i)}(d)}\right) = \sum_{i=1}^{m} \log(f^{(i)}(d)) - \sum_{i=1}^{m} \log(g^{(i)}(d)) \tag{3.4}
\]

The above two measurements, likelihood ratio and log likelihood ratio, estimate the likelihood that two clusters \( a \) and \( b \) belong to the same group (e.g., \( (a,b) \in A(p) \)) compared to the likelihood that they do not (e.g., \( (a,b) \in B(p) \)), given that the feature distance between them is \( d \).

### 3.6.7 Scoring the Clusters to Find the Best Match of a Target Population

Assume that Cluster Set \((3.1)\) contained all clusters generated by applying SamSPECTRAL clustering algorithm to a non-training FCM sample. In order to find the cluster \( C_{opt}^p \in Cluster Set \), with the highest similarity to the target population \( p \), I devised a method for measuring similarity scores of the clusters of the Cluster Set using a model for estimating the likelihood that two clusters belong to the same group through log likelihood ratio measurement \((3.4)\).

Since \( S(p) \) contained all training samples’ clusters that were similar to the target population \( p \), each cluster \( c \in Cluster Set \) was compared to all clusters \( s \in S(p) \) to measure the similarity between the cluster \( c \) and the
3.7 Validating Locations of Semi-Automatically Identified Cell Populations

Clusters of $S^{(p)}$. This corresponds to the similarity between the cluster $c$ and the desired population $p$. To do this, first the feature vectors $v(c)$ and $v(s)$ were computed for $\forall c \in \text{Cluster Set}$, and $\forall s \in S^{(p)}$, and the distance vectors were measured as $d(c, s) = v(c) - v(s)$. Then the $\text{log likelihood ratio}(d(c, s))$ was computed using (3.4), and the score of similarity between the cluster $c$ and the population $p$ was measured using the following formula:

$$Score^p(c) = \frac{1}{|S^{(p)}|} \sum_{s \in S^{(p)}} \text{log likelihood ratio}(d(c, s)). \quad (3.5)$$

The cluster $C^p_{\text{opt}} \in \text{Cluster Set}$, with the highest similarity score was selected as the best match of the target population $p$, that is,

$$C^p_{\text{opt}} = \arg\max_{c \in \text{Cluster Set}} (Score^p(c)) \quad (3.6)$$

A discussion on the time complexity of my learning-based cluster matching algorithm is given in Appendix G.

3.7 Validating Locations of Semi-Automatically Identified Cell Populations

In this study, I verified the locations of all semi-automatically identified populations $p$ for all FCM samples, in order to exclude cases with inaccurately identified cell population from further downstream analysis. To do this, for all FCM samples, the locations of the centres of the populations (measured as the MFI of surface protein markers after transformation and normalization) were computed. Then for each population $p$, the average value of MFI of all FCM samples belonging to the same group was computed. If the MFI of a sample population deviated more than 20% from the average value, it was flagged as an outstanding case, and it was excluded from downstream analysis. The interpretation of this would be that no biologically relevant variation of a population MFI deviates more than 20% of the MFI average value computed for each population. This 20% cutoff is data dependent and can be adjusted accordingly if higher or lower levels of variations are observed in the datasets of interest.

In order to find a satisfactory cutoff value, I used flowQ package [79] and modified it such that it flagged the outstanding samples for a specific target population, while depicting scatter plots of the semi-automatically identified target cell population across different samples. Changing the cutoff value resulted in flagging different sets of samples as outstanding cases.
3.8. Measuring Functional Response of Cytokine Positive Cells

Therefore, in this study, I tested few cutoff values ($x = 5, 10, 15, 20, 25, 30$) to empirically estimate a cutoff that was low enough to be able to flag the outstanding cases, and high enough to not flag correctly identified cases that deviated from the average value because of the between sample variations.

3.8 Measuring Functional Response of Cytokine Positive Cells

Table 3.2: Percentile ($\beta$) of the empirical cumulative distribution function (ECDF) of cytokine intensity which was used to automatically calculate the negative control thresholds.

<table>
<thead>
<tr>
<th>Target Population</th>
<th>IL-12</th>
<th>IL-6</th>
<th>TNF-α</th>
<th>IFN-α</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monocyte</td>
<td>99.8</td>
<td>99</td>
<td>99</td>
<td>99.9</td>
</tr>
<tr>
<td>cDC</td>
<td>99.8</td>
<td>99</td>
<td>99</td>
<td>99.9</td>
</tr>
<tr>
<td>pDC</td>
<td>99.8</td>
<td>99.8</td>
<td>99</td>
<td>99</td>
</tr>
</tbody>
</table>

After identifying APC populations, I measured the functional response of the cytokine producing cells. The first step in measuring functional response of APCs was to find the negative control defined threshold that distinguished between cells that produced cytokines (positive cells) and those that did not (negative cells). In my method, for each subject, the unstimulated control sample was used to determine the cytokine intensity threshold for stimulus-induced cytokine production. The negative control threshold was calculated as the $\beta^{th}$ percentile of the empirical cumulative distribution function (ECDF) of cytokine intensity. The value of $\beta$ was set as 99th by default, however, my preliminary experiments indicated that using a fixed constant for $\beta$ was not the best approach to take. In fact the results would improve if the values of $\beta$ were adjusted for each combination of target cell population and cytokine of interest separately. To this aim, the values of $\beta$ were adjusted by looking at the manually identified thresholds and comparing them against automatically identified ones in the training samples. Then the value of $\beta$ was slightly changed around the default value (99th percentile) until automatically and manually identified thresholds of the training samples were such that they agreed on the percentage and MFI of cytokine-positive cells in the training samples. After adjusting the values of $\beta$ for every combination of target cell population and cytokines of interest by using training samples, they were used to measure the negative control...
3.9. Comparing Manual and Semi-Automated Analysis

In order to verify if my semi-automated analysis pipeline was a proper substitute for the manual analysis, I needed to prove that it could replicate the results of manual analysis with sufficient accuracy. Therefore, I compared the results of semi-automated analysis against those of manual analysis. This comparison included two main parts. First I verified the accuracy of my semi-automated method in identifying APC cell populations by comparing the populations identified semi-automatically against those obtained manually. Then I compared the functional response of APC populations measured automatically against those obtained by manual analysis.

3.9.1 Comparison of the Manual and Semi-Automated Identifications of APC Populations

I compared the antigen-presenting cell populations (monocytes, other APCs, cDCs and pDCs) obtained manually against those obtained by applying SamSPECTRAL clustering algorithm in combination with my learning-based cluster matching method. The comparison was done in two ways. First, I compared the general features and characteristics of the populations including their sizes and mean fluorescent intensity (MFI). Second, I performed a cell by cell comparison between corresponding populations identified manually and semi-automatically. Note that for both types of comparisons, first the outstanding populations of the samples that failed the validation of the cell population location were excluded (11% of all samples).

Comparison of the Basic Features of the Populations

I compared some basic features of the corresponding populations, including the size or frequency of the APC populations, and MFI of four surface protein markers of the populations. Note that MFI values are given after applying logicle transformation and normalization as described in the data.
3.9. Comparing Manual and Semi-Automated Analysis

Table 3.3: Pearson correlation coefficients between the manually and automatically measured values. Percentage of outliers is given in parentheses.

<table>
<thead>
<tr>
<th>Target Population</th>
<th>Percentage</th>
<th>MHCII (MFI)</th>
<th>CD14 (MFI)</th>
<th>CD123 (MFI)</th>
<th>CD11c (MFI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monocyte</td>
<td>0.97</td>
<td>0.99</td>
<td>0.89</td>
<td>0.99</td>
<td>0.99</td>
</tr>
<tr>
<td></td>
<td>(3.63%)</td>
<td>(3.66%)</td>
<td>(3.14%)</td>
<td>(1.57%)</td>
<td>(0.52%)</td>
</tr>
<tr>
<td>Other APCs</td>
<td>0.98</td>
<td>0.99</td>
<td>0.91</td>
<td>0.98</td>
<td>0.95</td>
</tr>
<tr>
<td></td>
<td>(5.85%)</td>
<td>(4.88%)</td>
<td>(4.39%)</td>
<td>(4.88%)</td>
<td>(0.49%)</td>
</tr>
<tr>
<td>cDC</td>
<td>0.82</td>
<td>0.95</td>
<td>0.68</td>
<td>0.91</td>
<td>0.98</td>
</tr>
<tr>
<td></td>
<td>(4.49%)</td>
<td>(6.18%)</td>
<td>(5.62%)</td>
<td>(7.3%)</td>
<td>(5.06%)</td>
</tr>
<tr>
<td>pDC</td>
<td>0.90</td>
<td>0.99</td>
<td>0.87</td>
<td>0.99</td>
<td>0.80</td>
</tr>
<tr>
<td></td>
<td>(3.83%)</td>
<td>(4.92%)</td>
<td>(6.01%)</td>
<td>(3.83%)</td>
<td>(3.28%)</td>
</tr>
</tbody>
</table>

The cases for which the difference between automated and manual values deviated beyond the 95% prediction interval of the normal distribution (Z-Score > 1.64) were classified as outliers, and were excluded from analysis. Table 3.3 shows the Pearson correlation coefficients between the automatically measured percentage and MFI (after applying transformation and normalization) of the APC populations and the manually measured ones. In this table the percentage of outliers (Z-Score > 1.64) is given in parentheses. Although the Pearson correlation coefficient measures the strength of the linear relationship between two variables (automated vs. manual measurements), it does not indicate how similar their actual values are. In order to compare the automatically and manually measured values, a t-test with the null hypothesis that $|x - y| > d$ was applied, where $|x - y|$ represents the absolute difference between the automatically and manually identified measurements. Table 3.4 shows the value of $d$ for different combinations of populations and measurements (populations’ percentages and markers’ intensities). In this table, the $p$-values obtained from the t-test are given in parentheses. Before applying the t-test, the normality of the distributions was checked by applying a Pearson Chi-Square test for normality for different combinations of populations and measurements ($p$ value = $1e-5$).

**Single-Cell-Based Comparison**

In addition to comparing the general characteristics of manually and automatically identified APC cell populations, I did single-cell-based comparisons between the cell populations identified manually and automatically.
3.9. Comparing Manual and Semi-Automated Analysis

Table 3.4: The upper bound of the differences \(d\) between manually and automatically identified measurements (percentage of the target cell populations and MFI of the markers –MHCII, CD14, CD123 and CD11c– on the surfaces of the target cell populations). These values were obtained by applying a t-test with the null hypothesis that \(|x - y| > d\). \(p\) values from the t-test are given in parentheses.

<table>
<thead>
<tr>
<th>Target Population</th>
<th>Percentage</th>
<th>MHCII (MFI)</th>
<th>CD14 (MFI)</th>
<th>CD123 (MFI)</th>
<th>CD11c (MFI)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monocyte</td>
<td>1%</td>
<td>0.05</td>
<td>0.1</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>((p=1.5e-5))</td>
<td>((p=1.67e-12))</td>
<td>((p=7.63e-25))</td>
<td>((p=3.76e-14))</td>
<td>((p=1.2e-34))</td>
</tr>
<tr>
<td>Other APCs</td>
<td>1.5%</td>
<td>0.05</td>
<td>0.1</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>((p=9.1e-21))</td>
<td>((p=7.02e-43))</td>
<td>((p=1.82e-25))</td>
<td>((p=1.04e-14))</td>
<td>((p=1.29e-31))</td>
</tr>
<tr>
<td>cDC</td>
<td>0.5%</td>
<td>0.15</td>
<td>0.2</td>
<td>0.2</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>((p=2.04e-4))</td>
<td>((p=1.49e-20))</td>
<td>((p=9.43e-5))</td>
<td>((p=1.00e-5))</td>
<td>((p=4.68e-21))</td>
</tr>
<tr>
<td>pDC</td>
<td>0.1%</td>
<td>0.05</td>
<td>0.15</td>
<td>0.05</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>((p=8.60e-47))</td>
<td>((p=1.30e-11))</td>
<td>((p=1.15e-18))</td>
<td>((p=1.80e-4))</td>
<td>((p=3.97e-7))</td>
</tr>
</tbody>
</table>

MFI values were logarithmically transformed (base=10).

Assume that the set \(\text{Manual}(p)\) contains all cells of manually identified population \(p\). Similarly the set \(\text{Auto}(p)\) contains all cells of semi-automatically identified population \(p\). Here, the sets \(\text{Manual}(p)\) and \(\text{Auto}(p)\) were compared for \(\forall p \in \{\text{Monocyte, other APCs, cDC, pDC}\}\), and the sensitivity and specificity were measured as follows:

\[
\text{Sensitivity} = \frac{|\text{Manual}(p) \cap \text{Auto}(p)|}{|\text{Manual}(p)|}, \quad (3.7)
\]

\[
\text{Specificity} = 1 - \frac{|\text{Auto}(p)| - |\text{Manual}(p) \cap \text{Auto}(p)|}{N - |\text{Manual}(p)|}, \quad (3.8)
\]

where \(N\) was the total number of cells in the sample. Sensitivity and specificity always take values in the range \((0, 1)\). Higher sensitivity means that a larger portion of cells labeled as the target population by manual gating was labeled the same by automated clustering. Higher specificity means that a lower portion of cells that were not labeled as the target population \(p\) by the manual gating was labeled as the target population \(p\) by automated clustering. Figure 3.2 depicts the boxplots of sensitivity and specificity of four APC populations, (Monocyte, Other APCs, cDCs and pDCs), measured for all 216 FCM samples. As shown in this figure, for \(p \in \{\text{Monocyte, Other APCs}\}\) both of sensitivity and specificity were high, indicating that if a cell belonged to \(\text{Auto}(p)\), then with high certainty it belonged to \(\text{Manual}(p)\) too (high specificity), and vice versa (high sensitivity).
For $p \in \{cDC, pDC\}$, specificity was always high, but sensitivity varied between medium to high. This suggested that if a cell belonged to $Auto^{(p)}$, then with high certainty it belonged to $Manual^{(p)}$ too (high specificity), but the opposite statement was not necessarily true, that is, there might be cells that belonged to $Manual^{(p)}$, but did not belong to $Auto^{(p)}$ (medium-high sensitivity). In other words, for $p \in \{\text{Monocyte, other APCs}\}$, $Manual^{(p)}$ and $Auto^{(p)}$ included almost the same sets of cells, whereas for $p \in \{cDC, pDC\}$, $Auto^{(p)}$ could be considered as a subset of $Manual^{(p)}$.

![Boxplots of sensitivity and specificity of four APC populations (monocytes, other APCs, cDCs, and pDCs) measured for all 216 FCM samples.](image)

In addition to the sensitivity and specificity measurements, the F-Measure [3] was computed for all 216 FCM samples. The F-Measure also takes values in the range $(0, 1)$, and as shown in [3], the higher the F-Measure – the closer the automated clustering result is to the manual gating result. The mean and standard deviation of the F-Measure for my dataset were 0.976 and 0.029, respectively. This suggested that in overall, the results of cell population identification using SamSPECTRAL in combination with my learning-based cluster matching were highly close to the results of manual gating.
3.9. Comparing Manual and Semi-Automated Analysis

3.9.2 Comparison of Manual and Automated Functional Response Measurements

Functional response of cytokine producing cells can be presented by two measurements: (1) quantity of cytokine production estimated by percentage of cytokine-positive cells; and (2) quality of cytokine production measured by mean florescence intensity (MFI) of cytokine-positive cells. Here, I compared percentage and MFI of cytokine-positive cells measured automatically against those measured manually in order to identify levels of agreement between manual and automated measurements. In my comparison, the cases for which the absolute difference between manual and automated values exceeded the 99.5% prediction interval of the normal distribution ($Z$-Score > 2.57) were classified as outliers, and were excluded. Table 3.5 shows the Pearson correlation coefficients between percentages of cytokine-positive cells identified manually and those measured automatically. This table also presents the Pearson correlation coefficients between logarithmically transformed (base=10) MFI of cytokine-positive cells measured manually and those computed automatically. In Table 3.5, percentages of outliers ($Z$-Score > 2.57) are given in parentheses. In addition to measuring linear association between corresponding variables (automated vs. manual), the absolute differences between them ($|x - y|$) was computed, and a t-test ($p$-value=0.01) with null-hypothesis that $|x - y| > d$ was applied. Table 3.6 shows the value of $d$ for different combinations of the target populations and the measurements (percentages and MFI of cytokine-positive cells). Before applying the t-test, the normality of the distributions was confirmed by using Pearson Chi-Square test for normality ($p$ value = 0.005).

Table 3.5: Pearson correlation coefficients between percentage/MFI of cytokine-positive cells measured manually and automatically. The percentage of outliers is given in parentheses.

<table>
<thead>
<tr>
<th>Population</th>
<th>Measurement</th>
<th>IL-12</th>
<th>IL-6</th>
<th>TNF-α</th>
<th>IFN-α</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monocyte</td>
<td>Percentage</td>
<td>0.94 (3.7%)</td>
<td>0.98 (1.9%)</td>
<td>0.98 (2.8%)</td>
<td>0.63 (1.9%)</td>
</tr>
<tr>
<td></td>
<td>MFI</td>
<td>0.93 (2.8%)</td>
<td>0.90 (2.8%)</td>
<td>0.97 (4.8%)</td>
<td>0.80 (4.2%)</td>
</tr>
<tr>
<td>cDC</td>
<td>Percentage</td>
<td>0.95 (4.4%)</td>
<td>0.95 (3.5%)</td>
<td>0.96 (3.5%)</td>
<td>0.50 (4.4%)</td>
</tr>
<tr>
<td></td>
<td>MFI</td>
<td>0.92 (2.7%)</td>
<td>0.92 (3.6%)</td>
<td>0.94 (5.4%)</td>
<td>0.53 (4.3%)</td>
</tr>
<tr>
<td>pDC</td>
<td>Percentage</td>
<td>-0.04 (2.2%)</td>
<td>0.38 (2.2%)</td>
<td>0.95 (4.4%)</td>
<td>0.97 (3.3%)</td>
</tr>
<tr>
<td></td>
<td>MFI</td>
<td>0.42 (3.2%)</td>
<td>0.59 (4.9%)</td>
<td>0.92 (2.8%)</td>
<td>0.95 (1.6%)</td>
</tr>
</tbody>
</table>

Table 3.5 shows that correlation coefficients between manually and automatically identified measurements (percentage and MFI of cytokine-positive cells) were significant for those cytokines that were produced moderately or
3.10. Impact of Training Sample Size on Cluster Matching Results

Table 3.6: Difference ($d$) between manually and automatically identified measurements (percentage and MFI of cytokine-positive cells of different target populations). These values were obtained by applying a t-test with null hypothesis that $|x - y| > d$ ($p$ value = 0.01).

<table>
<thead>
<tr>
<th>Target Population</th>
<th>Measurement</th>
<th>IL-12</th>
<th>IL-6</th>
<th>TNF-α</th>
<th>IFN-α</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monocyte</td>
<td>Percentage</td>
<td>1.3%</td>
<td>5.1%</td>
<td>5.4%</td>
<td>0.7%</td>
</tr>
<tr>
<td></td>
<td>MFI</td>
<td>0.1</td>
<td>0.07</td>
<td>0.08</td>
<td>0.22</td>
</tr>
<tr>
<td>cDC</td>
<td>Percentage</td>
<td>3.7%</td>
<td>5.9%</td>
<td>6.5%</td>
<td>0.9%</td>
</tr>
<tr>
<td></td>
<td>MFI</td>
<td>0.12</td>
<td>0.07</td>
<td>0.12</td>
<td>0.22</td>
</tr>
<tr>
<td>pDC</td>
<td>Percentage</td>
<td>1.6%</td>
<td>2.4%</td>
<td>6.9%</td>
<td>6.7%</td>
</tr>
<tr>
<td></td>
<td>MFI</td>
<td>0.5</td>
<td>0.14</td>
<td>0.14</td>
<td>0.09</td>
</tr>
</tbody>
</table>

MFI values were logarithmically transformed (base=10).

highly by target populations. That is, the correlation coefficient was significant (> 0.90) for IL-12, IL-6 and TNF-α produced by monocytes and cDCs, and also IFN-α and TNF-α produced by pDCs. Table 3.6 shows that for these cases not only the association between two variables (manual vs. automated) was high, but also the actual difference between variables was low ($p$ value = 0.01), indicating that manual and automated approaches produced similar values. For the other cases that the cytokine of interests were not produced by target populations, the correlation coefficients between manual and automated measurements were either weak or moderate. These cases included IFN-α for monocytes and cDCs, and also IL-12 and IL-6 for pDCs. I believe that the reason for observing low correlations between manual and automated variables in these cases is that the target populations either did not produce cytokines of interest or produced very low levels of them [45]. Both manual and automated measurements were highly sensitive to the noise and therefore, none of them could give the exact values of the measurements. This resulted in weak-moderate correlations in these exceptional cases. Table 3.6 confirms this as for these cases the differences between percentage of cytokine-positive cells measured manually and automatically were quite low (0.7%, 0.9%, 1.6% and 2.4%).

3.10 Impact of Training Sample Size on Cluster Matching Results

A 2-fold cross validation was performed in order to investigate the effect of training sample size on cluster matching results. Samples were randomly
3.10. Impact of Training Sample Size on Cluster Matching Results

divided into two equal size sets \( d_0 \) and \( d_1 \), each of which was used for training and test in turn. Let’s assume \( d_0 \) was used for training, and \( d_1 \) was the test set. The samples of \( d_0 \) set were randomly ordered, and the following steps were repeated for all \( s \) in range \([1, |d_0|]\).

- A set of samples was built by selecting the first \( s \) samples of the ordered \( d_0 \) set. This set was called \( d_{tr}(s) \).
- \( d_{tr}(s) \) was used as the training set, and the learning-based cluster matching method was applied to all samples of \( d_1 \) test set in order to identify target cell populations in all of test samples.
- For each data sample in \( d_1 \), F-Measure between manual and semi-automated gating results was computed, in order to verify performance of the cluster matching algorithm when training sample size was equal to \( s \).

The above analysis was repeated five times with five different random ordering of \( d_0 \) set. Also the same process was performed when \( d_1 \) was selected for training and \( d_0 \) was used as the test set. In each run of the experiments, each sample was used exactly \( |d_0| \) times for test (once for each \( s \) in range \([1, |d_0|]\)). Note that \( |d_0| = |d_1| \), as the original samples were randomly divided into two equal size sets.

As mentioned earlier, before applying cluster matching algorithm to the FCM data samples, the samples were grouped by Algorithm A of Appendix F, such that samples with similar monocyte features stayed in the same group (Figure A.3). Figure 3.3 depicts F-Measure vs. training sample size for different groups of samples (A-E). In Figure 3.3, mean of F-Measure is shown by black circles, and the error bars are shown for 99% confidence intervals. Although for each group of samples, the results were generated for all sample sizes in range \([1, |d_0|]\), I showed the result for sample sizes up to 25 just for the sake of figure clarity. As shown in this figure, for all groups of samples, at the beginning of the curve, the F-Measure was increasing noticeably by adding new training samples (slope of the curve was positive). However, after a certain training sample size (\( \leq 6 \)), F-Measure curve remained almost horizontal, meaning that the impact of adding new training samples on the performance of cluster matching was negligible after this sample size \( (s_c) \). Having relatively low values for \( s_c \) after which F-Measure remained almost unchanged, confirmed that cluster matching algorithm reached to its full performance by using just few samples \( (\leq 6) \) as training samples. In Figure 3.3, two red horizontal lines show the margins
3.10. Impact of Training Sample Size on Cluster Matching Results

of confidence intervals of F-Measure for \( s_c \leq s \leq |d_0| \). Table 3.7 summarizes number of samples, number of required training samples \( (s_c) \), and the margins of F-Measure confidence intervals for the groups of samples (A-E) that were obtained by Algorithm A of Appendix F. Only group F is not included in Table 3.7 and Figure 3.3 because it had too low number of samples, and they were all flagged as outstanding samples.

Table 3.7: Summarizing training sample size results for Algorithm A

<table>
<thead>
<tr>
<th>Group</th>
<th># Samples</th>
<th># Training Samples</th>
<th>Confidence Interval of F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>63</td>
<td>5</td>
<td>(0.935, 0.965)</td>
</tr>
<tr>
<td>B</td>
<td>85</td>
<td>6</td>
<td>(0.958, 0.973)</td>
</tr>
<tr>
<td>C</td>
<td>23</td>
<td>5</td>
<td>(0.947, 0.978)</td>
</tr>
<tr>
<td>D</td>
<td>21</td>
<td>4</td>
<td>(0.910, 0.955)</td>
</tr>
<tr>
<td>E</td>
<td>19</td>
<td>5</td>
<td>(0.955, 0.980)</td>
</tr>
</tbody>
</table>

In this study, I was interested to see if each group contained samples with lower similarities, then still a low number of training samples \( (s_c \leq 6) \) was adequate to obtain satisfactory cluster matching results. In order to test this, I used Algorithm B of Appendix F to generate three groups of samples with higher levels of variabilities in monocyte features within each group (Figure A.5). Figure 3.4 depicts F-Measure vs. training sample size for different sample groups (A-C) defined by Algorithm B. In this figure, mean of F-Measure and the error bars are shown for 99% confidence intervals. Figures 3.3 and 3.4 both indicated that F-Measure curves of all groups of samples followed the same patterns, that is, F-Measure curves went up at the beginning, and then became flat after relatively low training sample sizes \( (s_c \leq 6) \). There was no significant difference between grouping by Algorithms A and B in this sense. Table 3.8 summarizes number of samples, number of required training samples \( (s_c) \), and the margins of F-Measure confidence intervals for groups of samples (A-C) that were obtained by Algorithm B of Appendix F. Only group D is not included in Table 3.8 and Figure 3.4 because it had too low number of samples, and they were all flagged as outstanding samples.
### 3.11. Discussion

Table 3.8: Summarizing training sample size results for Algorithm B

<table>
<thead>
<tr>
<th>Group</th>
<th># Samples</th>
<th># Training Samples</th>
<th>Confidence Interval of F-Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>100</td>
<td>6</td>
<td>(0.954, 0.970)</td>
</tr>
<tr>
<td>B</td>
<td>76</td>
<td>6</td>
<td>(0.930, 0.955)</td>
</tr>
<tr>
<td>C</td>
<td>35</td>
<td>5</td>
<td>(0.960, 0.980)</td>
</tr>
</tbody>
</table>
Figure 3.3: F-Measure vs. size of training set for sample groups A-E obtained by Algorithm A. In all cases, the F-Measure curve became flat after a relatively low training sample size ($s_c \leq 6$).
Figure 3.4: F-Measure vs. size of training set for sample groups A-C obtained by Algorithm B. In all cases, the F-Measure curve became flat after a relatively low training sample size ($s_c \leq 6$).
3.11 Discussion

Although manual analysis of FCM data had been the only feasible way of analysing this data for around two decades, it has serious disadvantages including time inefficiency and subjective bias due to interobserver variability in analysis. In addition, in many applications the number of FCM samples is so large that manual analysis is infeasible. In this study, I developed a semi-automated method that can substitute the manual analysis to replace time-consuming tasks, while replicating the manual analysis results with high accuracy. A significant strength of my semi-automated analysis method was that the variability in target sample population surface marker expression secondary to stimulation with relevant ligands or intersubject variabilities was addressed by automated grouping samples with similar patterns as a preprocessing step of my algorithm. I applied my semi-automated pipeline to an innate immune response FCM dataset that included 216 FCM samples from 12 subjects at 3 time points to demonstrate its applicability to FCM data. I also compared its performance to the manual analysis for FCM data to show its capability in replicating the manual analysis performed by an expert biologist. Although manual analysis of the current dataset of 216 samples was feasible, the follow-up studies of my innate immune response FCM data are projected to soon reach up 180,000 FCM samples (10,000 subjects at 3 time points with 6 aliquots per subject per time point). It is obvious that analysing this huge dataset will not be feasible in terms of time and effort without the aid of automated computational techniques.

Comparing the basic features including frequency and MFI of cell surface markers of the target cell populations (identified manually vs. semi-automatically), I showed that these measurements highly agreed in terms of both linear correlation (Table 3.3) and the actual values (Table 3.4). Cell-wise comparison of manually and automatically identified APC cell subsets showed that specificity was high for all populations of interest, while sensitivity was high for monocytes and Other APCs, and medium-high for cDCs and pDCs (Figure 3.2). This observation indicated that for monocytes and Other APCs, manually and automatically identified cell populations consisted of almost the same subsets of cells, while for cDCs and pDCs, manual gates were almost supersets of automatically identified cell populations. This result is quite satisfactory, as in many applications like this study what matters the most is that the automatically identified cell populations are highly pure, and do not include the cells from other populations (high specificity). However, depending on the application if it is desirable to identify cell populations more completely and as close as possible to the manually identified
3.11. Discussion

cell populations, then the features of the cluster matching method need to be tuned accordingly to achieve this goal.

In this study, I made two main contributions to the automated analysis of FCM data. (1) I developed a novel learning-based cluster labeling method that incorporates the biology expert knowledge to find the cluster that best matches the desired target population; (2) I put different stages of the data analysis including data preprocessing, automated clustering, learning-based cluster matching, and measuring functional response of cytokine producing cells into a semi-automated pipeline to fully analyse the FCM data. To my knowledge, this is the first complete semi-automated data analysis pipeline for immune response FCM data that reproduces manual analysis results with high accuracy.

My first main contribution to the automated analysis of FCM data was designing a learning-based method for finding the desired target populations among all clusters obtained by applying a clustering algorithm, and assigning biologically meaningful labels to them. My novel method incorporated biology expert knowledge to define the target populations on a few training samples, and extracted the features of the target populations in order to use them for finding the best matches of the target populations in other FCM samples automatically. Although recently several automated clustering methods have been developed to identify cell types automatically, current algorithms usually fail to accurately identify all populations of interest simultaneously. Even if a clustering algorithm can generate all clusters that correspond to target populations in a single run, there still is the need for a labeling method to match the populations across samples. My proposed cluster matching method reached the above two goals at the same time by finding all target populations, and assigning labels to them.

The populations that were studied here had a wide range of characteristics, some of which made clustering and cluster matching very challenging. First, some populations (e.g., pDCs, and cDCs) were rare cell populations, which are, as previously mentioned, challenging to locate both manually and using automated approaches. Second, the features of some populations change with the stimulation type. An example of this is the frequency of the monocytes that significantly decreased when the sample was treated with 3M-003 stimuli, which introduced the challenge of tracking the same population with variable characteristics across different samples. Finally, in some cases the low density populations were adjacent to high density populations. In such cases, there was the chance that the cluster matching may erroneously select a cluster that included both the low density population and its adjacent high density population as the best match of either popula-
3.11. Discussion

My experiments showed that my cluster matching method was general enough to deal with all these different characteristics, and successfully identify all types of cell populations.

My second main contribution was designing a semi-automated pipeline consisting of four main computational blocks: (1) Preprocessing and data preparation, (2) clustering by SamSPECTRAL, (3) learning-based cluster matching, and (4) automated measuring of functional response of cytokine producing cells. Depending on the application, one can use each computational block of my pipeline independently, or use all components together. It is also possible to replace each block of the pipeline with a similar method, and keep the rest of the blocks unchanged. An example of this is to use another clustering algorithm instead of SamSPECTRAL in combination with the other computational blocks of my semi-automated pipeline.

I used the term of semi-automated for my analysis pipeline because some parts of the analysis were done in a fully automated fashion, while others were done with human interactions. In my semi-automated pipeline, all labour-intensive and time-consuming tasks of the manual analysis were replaced with the automated computational techniques. Only a few fast and easy tasks are still done manually. Among all tasks of the first block of my pipeline, only FCM data compensation and removing dead cells and debris were done manually. Recently a method has been developed to automate the compensation of FCM data samples [167], but in this study, I preferred to use the same compensation matrices as were used in the manual analysis. The reason was that I aimed to compare my results against manual analysis results, and therefore, I wanted to have the same markers values after compensation to make this comparison more fundamentally accurate. In the future, this manual compensation can be replaced with the automated one for the follow-up studies that include 180,000 FCM samples. Removing dead cells and debris is a tricky task that is not easily done automatically if the dead cell marker is not available. Even though using a static gate can be a solution for making removal of dead cells and debris automated, this is not a proper solution in our case because of the high between-samples variations. In the future, by adding a marker for dead cells, this manual part of my pipeline will be replaced by an automated dead cell removal method.

The second block of my pipeline, SamSPECTRAL clustering, was done in a fully automated fashion, and only the initial setting of the clustering parameters was done manually. In comparison, the third block of my semi-automated method, the cluster matching, was a learning-based method and needed 4-6 randomly selected samples from each group to be used as the training samples (Tables 3.7 and 3.8). As Figures 3.3 and 3.4 suggest my
3.11. Discussion

learning-based cluster matching reached to its full performance in identification of target cell populations by using just few samples ($\leq 6$) for training. In addition, Tables [3.7] and [3.8] show that F-Measures were quite high (always between 0.91 and 0.98) for different groups of samples, confirming that outcomes of the cluster matching were very close to manual gating results. My learning-based cluster matching method needed the manual identification of cell populations for the training samples, but no human-interaction was required for labeling the populations of the rest of samples. This was quite desirable in terms of manual time and effort, as this limited the manual tasks to just a few samples ($\leq 6$), while the analysis for the rest of samples was done fully automatically.

Similarly, for the fourth block of my pipeline which measured the functional response of cytokine producing cells, an ECDF percentile value for the threshold was selected through testing on a few training samples ($\leq 6$), and then it was used to measure the threshold for the rest of samples in a fully automated way. In order to measure the functional response of cytokine producing cells, a threshold based on unstimulated control samples was used to distinguish between the cells that produce cytokines, and those that did not. In my approach, the empirical cumulative distribution function (ECDF) was used to identify this threshold. The percentile of ECDF where the threshold was set was the 99th percentile by default; however, it was adjusted for different combinations of cytokines and target populations independently through testing on a few training samples (Table [3.2]). This adds more flexibility to my approach as the threshold can be adjusted based on the operator specifications, which results in more accurate outcomes.

It should be noted that removing the dead cells and debris (the first block), the initial setting of the non-sensitive values for the parameters of the clustering algorithm (the second block), selecting the features for the cluster matching (the third block), and adjusting the ECDF percentile value for the threshold when measuring the functional response of cytokine producing cells (the fourth block) contribute to the “semi”-automated nature of my method. Therefore, the users need to consider them as the steps that need to be done manually when using my semi-automated analysis pipeline.

Since a single run of a clustering algorithm may not generate all populations of interest at the same time, and for instance split a population into two, or merge a population with its adjacent cluster, I hypothesized that running the clustering algorithm several times with different initial values for the parameters (e.g., number of clusters) increases the probability that all clusters that correspond to all of the target populations can be found in the Cluster Set. This is the main reason for running SamSPECTRAL
3.12 Conclusion and Future Work

I developed a complete semi-automated pipeline for analysing FCM data. One of my major contributions in automating the analysis of FCM data was designing a learning-based cluster matching method that incorporates biology expert knowledge to identify the target cell populations, and integrating this method into my semi-automated pipeline. In this study I verified the performance of my complete pipeline by comparing the results of semi-automated cell population identification and automated measurement of functional response of cytokine producing cells against those computed manually. My experiments confirmed that my semi-automated analysis pipeline can replicate the manual analysis with high accuracy, and therefore, can replace the time-consuming manual analysis.

Although my semi-automated pipeline was quite successful in identifying the cell populations in 89% of the cases, I believe that there is still room for improvement. Two main directions for improvement are (1) improving the performance of the cluster matching by using an efficient automated feature selection method; and (2) generating a more complete Cluster Set using different clustering algorithms with different initialization. Improvement in the first direction removes the need for manual identification of the features, multiple times. This idea can be extended to generating a set of clusters by using different clustering algorithms with different initial values, and then using the resulting cluster set to search for the target populations. 

One important point here is that the performance of my pipeline in identifying the target cell populations depends on two factors: (1) the ability of the clustering algorithm to generate all clusters that correspond to the target populations; and (2) the ability of the learning-based cluster matching method to find the best match of the target populations among all clusters generated by the clustering method. If either of these fail for any reason, then the target population cannot be identified correctly. In my results, I showed that in 89% of the cases my pipeline was successful in identifying the cell populations correctly. For the remaining 11%, my quality checking method could successfully flag them as outstanding cases by verifying the locations of the identified populations. The outlier cases were removed from downstream automated analysis. For such cases, manual gating can be used to identify cell populations. In future, I am going to substitute the manual setting of cluster matching features (Table 3.1) with an appropriate automated feature selection approach, in order to reduce the percentage of failing cases.
3.12. Conclusion and Future Work

and increases the rate of successfully identified populations by facilitating the selection of the optimum features for every target population.
Chapter 4

Correlation Analysis of Intracellular and Secreted Cytokines via the Generalized Integrated Mean Fluorescence Intensity

4.1 Background

While direct measurement of protection from infection after a defined challenge provides the most meaningful information in vaccine trials, in human studies intermediate biomarkers (e.g., antibody titers or various measurements of cell-mediated immunity (CMI)) are used as correlates or surrogates of protection [133]. The CMI response is often determined by measuring cytokines within the cell or secreted in serum or in culture supernatant. Given that cytokines exert their function mostly after being secreted, both approaches potentially measure different aspects of CMI, yet are often used interchangeably. To our knowledge, a direct correlative comparison of these two approaches has not been conducted. While quantification of secreted cytokines can be conducted using ELISA or multiplex bead arrays, these methods do not identify the specific cell source of these secreted cytokines [168]. Alternatively, flow cytometric analysis of intracellular cytokine staining (ICS) is able to identify the specific cells producing a given cytokine, but it does not allow their absolute quantification. ICS results are determined as either percent positive cells or as mean fluorescent intensity (MFI) of a population of cytokine producing cells, with both measurements compared to a control population of untreated or unstimulated cells. The integrated MFI (iMFI) [52] was devised to increase the quantitative informational content of data obtained from ICS by combining the relative amount of a cytokine produced per cell or population (the MFI) with the relative number of cells.
4.2 Methods

4.2.1 Data and Wet Lab Description

Kollmann’s lab members including Darren Blimkie and Edgardo Fortuno obtained flow cytometry and Luminex data from the analysis of peripheral blood mononuclear cells (PBMC) from healthy human adult volunteers as previously described [102]. The study was approved by the Clinical Research Ethics Board of the University of British Columbia. Briefly, PBMC from 20 adult subjects (8 males and 12 females) were purified using Ficoll-Hypaque density centrifugation. Half a million cells in 200 μl of RPMI supplemented with 10% human AB serum and 1% penicillin-streptomycin were plated on premade 96-well plates containing 10 μl of TLR ligands in various concentrations (final concentrations of the ligands were: PAM3CSK4 at 0.01, 0.1 and 1 μg/ml; R-FSL at 0.1, 1 and 10 μg/ml; poly I:C at 12.5, 25 and 50 μg/ml; LPS at 1, 10 and 100 ng/ml; 3M-002, 3M-003, 3M-013 at 0.1, 1 and 10 μM each, and; CpG-A, CpG-B and CpG-C at 6.25, 12.5 and 25 μg/ml each).

The cells plated for the ICS were cultured for 6 hours in the presence of the Golgi transport inhibitor Brefeldin A (BFA) at 10 μg/ml with the exception of the wells containing the CpG ligands. The cells stimulated with the CpG ligands were cultured for 3 hours without BFA followed by an additional 3 hours in the presence of BFA. BFA addition was delayed, because it hinders
the response to CpG stimulation if added immediately [95]. At the 6-hr mark, these cells were treated with EDTA (final concentration of 2 mM), spun down, and the pellet resuspended in 100 µl of Becton-Dickinson (BD) FACS Lysing Solution and then frozen in -80°C until antibody staining and cytometric analysis using a BD LSRII instrument [95]. Identical set of plates except without BFA were setup and cultured for 18 hours, spun down, and 100 µl of the supernatants harvested and stored in -80°C prior to Luminex assay using kits from Millipore.

Data from the LSRII instrument were analysed by Kollmann’s lab using FlowJo (TreeStar, Inc.) and CPAS (LabKey Solutions) [152]. They also further analysed Luminex data from the Bio-plex 200 instrument and the Bio-plex Manager software (Bio-rad) in an in-house database, in Excel (Microsoft) and in Prism (GraphPad).

4.2.2 Correlation Analysis of iMFI and Culture Supernatant Response

In order to test the hypothesis that culture supernatant concentration of a particular cytokine correlated to its corresponding intracellular production cytokine as estimated by the iMFI, I computed Pearson correlation coefficients between iMFI values and culture supernatant responses of all samples (including all listed TLR ligands and concentrations) obtained from every single subject, and determined strength of linear correlations between these two values. The levels of correlations were interpreted as no correlation, weakly, moderately, strongly, and perfectly positive correlation when Pearson correlation coefficients were approximately 0, 0.2, 0.5, 0.8 and 1, respectively [182]. Two different approaches were examined to compute the levels of correlation. In the first approach, iMFI values were computed for all the cytokine-positive cells in the PBMC together, while in the second approach, the iMFI values were computed for the cytokine-positive cells from each specific antigen-presenting cell (APC) population (i.e., monocytes, conventional dendritic cells (cDC) and plasmacytoid dendritic cells (pDC)) separately. In both of these two approaches, the first step was to identify cytokine-expressing (positive) cells, and to measure their percentage and mean fluorescence intensity (MFI) values to compute iMFI by the following formula:

\[ i\text{MFI} = MFI \times P \]  

(4.1)

where \( P \) is the relative percentage of the cells expressing a specific cytokine.
4.2. Methods

4.2.3 Automated Identification of All Cytokine-Positive Cells of PBMC

As the bulk amount of cytokines present in the culture supernatant was measured by the Luminex assay regardless of cell source, I performed my initial correlation analysis of the results from this assay with the iMFI values derived from all the cytokine-positive cells in the 6-hr ICS of the corresponding PBMC. I developed an automated method to identify the cytokine-expressing (i.e., positive) cells in our ICS assay by setting a threshold based on the assumption that 2% of unstimulated PBMC cells (i.e., in a control sample) are cytokine-positive. In other words, after identifying live cells using a FSC vs. SSC plot in FlowJo, a threshold was computed based on the 98% cutoff of the empirical cumulative distribution function (ECDF) of the cytokine intensity for the unstimulated sample (see supplementary materials), and this threshold was used to identify positive cells of all other samples from the same subject. Identifying positive cells and knowing their percentages and MFI values, iMFI value of each cytokine was computed using equation 4.1.

4.2.4 Manual Identification of Antigen-Presenting Cell Populations and Detection of Cytokine-Positive Cells for Different APC Populations

I hypothesized that Luminex assay results may correlate better with iMFI values from specific antigen-presenting cell populations instead of the iMFI values from the cytokine-positive cells in the whole PBMC, and thus computed the correlation coefficients between iMFI values of different APC populations and culture supernatant responses. Here, unlike the automated identification of the cytokine-positive cells in PBMC described above, FlowJo was used to manually identify the cDC, pDC and monocytes antigen-presenting cell populations and also the positive cells for each population. First, dead cells and cell debris were excluded by gating them out from an FSC vs. SSC plot (Figure 3.1). From this “live” gate, the cells on MHC II vs. CD14 were plotted. CD14\textsuperscript{high}, and MHC\textsuperscript{+} cells were identified as monocytes. By plotting the CD14\textsuperscript{low/neg}, MHC\textsuperscript{+} cells on CD123 vs. CD11c axes, the conventional dendritic cells were identified as the CD11c\textsuperscript{+} population while the plasmacytoid dendritic cells were identified as CD123\textsuperscript{+}. Next, the monocytes, cDC and pDC were graphed on scatter plots with the fluorescence intensity for each of the cytokines as the axes. For each subject, the quadrants inside these scatter plots which separated the cytokine-producing cells...
from cytokine-negative ones were manually determined based on the cells that were not stimulated with any TLR ligand.

4.2.5 Generalized iMFI (GiMFI)

The iMFI assumes the impact of MFI and \( P \) are the same when estimating a functional response of cytokine producing cells. To better model the functional response of cytokine producing cells, this requirement was relaxed, and my Generalized iMFI was defined as:

\[
GiMFI(\alpha) = MFI \times P^\alpha
\]

(4.2)

where MFI and \( P \) are mean of fluorescent intensity and percentage of cytokine positive cells, respectively, and \( \alpha \) is the relative impact weight assigned to the percentage of positive cells. If \( 0 \leq \alpha < 1 \), then the impact weight of \( P \) in GiMFI is less significant compared to the iMFI formula. If \( \alpha > 1 \) a higher relative impact weight is assigned to \( P \) in GiMFI. GiMFI(\( \alpha \)) and iMFI would have the same values for \( \alpha = 1 \).

In order to find \( \alpha \), such that GiMFI(\( \alpha \)) modeled the functional response of cytokine-producing cells, GiMFI(\( \alpha \)) was computed for discrete \( \alpha > 0 \) values and the Pearson correlation coefficients between GiMFI(\( \alpha \)) and culture supernatant cytokine response were computed for different \( \alpha \) values. Then \( \alpha \) was chosen such that the mean of correlation coefficients (\( R \)) of all subjects was maximized. Note that there was no restriction on selecting mean, and using the median could result in similar outcomes. Although the Pearson correlation coefficients were computed for a relatively wide range of \( \alpha \), no values of \( \alpha > 4 \) were included in the calculations for finding the maximum \( R \), due to the observation that the mean of correlation coefficients between GiMFI(\( \alpha \)) and culture supernatant cytokine response monotonically decreased after \( \alpha = 4 \) in all cases studied here. To keep the notation simple, I will use \( \alpha_{\text{max}} \) to present the value of \( \alpha \) when the correlation coefficient between GiMFI(\( \alpha \)) and culture supernatant cytokine response was at its maximum value. I also use the notation \( \rho(\alpha) \) for correlation coefficients between culture supernatant cytokine response and GiMFI(\( \alpha \)). \( \rho_{iMFI} \) is the correlation coefficient between culture supernatant cytokine response and iMFI. While \( \rho(\alpha) \) and \( \rho_{iMFI} \) can be computed for any combination of cytokine and population, we were not interested in studying these values for those cytokines which are known not to be at least moderately produced by the population of interest. For example, the correlation analysis on monocytes and cDCs was performed for IL-12p40, IL-6 and TNF-\( \alpha \). Similarly, the correlation analysis for IFN-\( \alpha \) was done only on pDCs, because cDCs
and monocytes do not generate IFN-α under the culture conditions chosen [102].

![Figure 4.1: Scatter plots of the amount of secreted IL-12p40 measured by Luminex versus calculated iMFI values from the intracellular cytokine staining for IL-12p40 of cells from one subject treated with different TLR ligands at different concentrations ($\rho_{iMFI} = 0.94$). Each (*) corresponds to a stimulations of a blood sample of the same subject. The solid line shows the best line fitted to the data based on simple regression model.]

4.3 Results

4.3.1 Correlation Analysis of Whole PBMC Live Cells

Because our Luminex assay measured the cytokine concentration in bulk without regard to cell source, I wanted to first examine if these concentrations correlate with the iMFI computed from all the cytokine-positive cells in the corresponding whole PBMC population. I computed the correlation
4.3. Results

Figure 4.2: Boxplots of correlation coefficients between culture supernatant cytokine response and iMFI values computed for IL-12, IL-6, TNF-α, and IFN-α) for 20 subjects. All PBMC live cells are considered together.

coefficients between the iMFI values and the culture supernatant cytokine concentration for the cytokines TNF-α, IFN-α, IL-6 and IL-12p40. Figure 4.1 shows an example of a scatter plot of the amount of secreted IL-12p40 as measured by Luminex versus the calculated iMFI values from the intracellular cytokine staining for IL-12p40 of cells from one subject treated with different TLR ligands at different concentrations ($\rho_{iMFI} = 0.94$). The solid line shows the best line fitted to the data based on simple regression model. Linear model parameters were estimated using the least square method for linear regression. The boxplots in Figure 4.2 represent the correlation coefficients computed for all four cytokines from the TLR ligand-stimulated PBMC from 20 individuals. Correlations between iMFI values and Luminex responses were strongly positive for TNF-α, while they were only moderately positive for IL-6 and IL-12p40. Almost no correlation was observed between iMFI values and Luminex responses for IFN-α.

4.3.2 Correlation Analysis of Antigen-Presenting Cell Populations

Motivated by the lack of correlation between iMFI and Luminex for IFN-α, I next examined if the cytokine concentrations measured from the supernatants of our 18-hr culture would exhibit higher correlation with the com-
4.3. Results

Figure 4.3: (A-C) Boxplots of Pearson correlation coefficients between culture supernatant cytokine response and iMFI values from the intracellular cytokine staining for four cytokines (IL-12, IL-6, TNF-α, and IFN-α) produced by (A) monocytes, (B) cDCs, and (C) pDCs. (D-F) Boxplots of Spearman correlation coefficients between culture supernatant cytokine response and iMFI values from the intracellular cytokine staining for four cytokines (IL-12, IL-6, TNF-α, and IFN-α) produced by (A) monocytes, (B) cDCs, and (C) pDCs.
4.3. Results

Table 4.1: Correlation between iMFI values and culture supernatant cytokine responses of IL-12, IL-6, TNF-\(\alpha\), and IFN-\(\alpha\) produced by monocytes, cDC, and pDC based on the median of correlation coefficients.

<table>
<thead>
<tr>
<th>Antigen Presenting Cell Populations</th>
<th>Cytokines</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IL-12</td>
</tr>
<tr>
<td>Monocyte</td>
<td>Moderate</td>
</tr>
<tr>
<td>cDC</td>
<td>Strong</td>
</tr>
<tr>
<td>pDC</td>
<td>No Correlation</td>
</tr>
</tbody>
</table>

Computed iMFI values when these were derived from specific subpopulations in the PBMC. Figure 4.3 depicts boxplots of correlation coefficients between iMFI values and supernatant cytokine levels for three antigen-presenting cell populations (monocytes, cDCs, and pDCs) for 20 subjects. Figure 4.3C shows that for pDCs, the level of correlation between iMFI and culture supernatant cytokine response is strong for IFN-\(\alpha\) and weak for TNF-\(\alpha\). No correlation was observed for IL-12p40 and IL-6. This indicated that considering only pDCs instead of whole PBMC improves the correlation between iMFI and culture supernatant cytokine response for IFN-\(\alpha\).

Figure 4.3A shows that correlation coefficients between iMFI and culture supernatant cytokine response was strong for TNF-\(\alpha\), weak/moderate for IL-6 and IFN-\(\alpha\), and moderate for IL-12p40 in monocytes. Similarly, Figure 4.3B shows that for cDCs, correlation coefficients between iMFI and culture supernatant cytokine response was strong for IL-12p40 and TNF-\(\alpha\), while it was moderate for IL-6. No correlation was observed for IFN-\(\alpha\) in cDCs. Table 4.1 summarizes the level of correlation between iMFI values and culture supernatant cytokine response. Overall, for any specific APC population, the correlation between iMFI values and culture supernatant cytokine response varies from cytokine to cytokine, though generally the more cytokine produced by the APC population, the higher the correlation. How-
ever, observing a high correlation does not always mean that the population under study was a major producer of the cytokine of interest.

Here, I also examined the correlation between the culture supernatant cytokine response and iMFI values using Spearman correlation (Figure 4.3D-F). Spearman correlation coefficient is a non-parametric correlation which is well-suited for detecting the monotonic trend between two variables. In other words, it shows how well the association between two variables is fitted to a monotonic non-linear function. Comparing the boxplots shown in Figure 4.3A-C against those of Figure 4.3D-F, one can conclude that generally the correlation patterns obtained by Spearman method were very similar to those obtained by Pearson; however, the strength of Spearman correlation coefficient was similar for those cytokines that were produced at least moderately by the target population of interest, no matter what the level of cytokine production was.

4.3.3 Parameter Estimation for the Generalized iMFI (GiMFI) Formula

For each APC population and each cytokine, the optimum $\alpha$ was determined such that the correlation coefficients between GiMFI($\alpha$) and culture supernatant cytokine response was maximized. Figure 4.4 illustrates my strategy for deriving $\alpha_{\text{max}}$. My experiments showed that for $\alpha > 4$, $\rho(\alpha)$ (correlation coefficients between culture supernatant cytokine response and GiMFI($\alpha$)) decreases monotonically, so $\rho(\alpha)$ was not plotted for higher values, as I was interested only in the maximum of $\rho(\alpha)$. My computation showed that range of variation of $\alpha_{\text{max}}$ was narrow for different APC populations and various cytokines. More precisely, the lower quartile, mean, upper quartile and max were 0.29, 0.39, 0.47 and 0.60 respectively. Note that for $\alpha = 1$, GiMFI($\alpha$) and iMFI result in the same values, that is, $\rho(1) = \rho_{\text{iMFI}}$. The fact that $\alpha_{\text{max}}$ was found to always be less than 1 suggests that assigning a lower impact weight to the percentage of cells positive as compared to the MFI in the GiMFI formula would strengthen the correlation.

Although $\alpha_{\text{max}}$ was chosen in a way to maximize the average of Pearson correlation coefficients between GiMFI($\alpha$) and culture supernatant response, it was not clear if the correlation coefficients were in fact higher for GiMFI($\alpha_{\text{max}}$) compared to iMFI for individual subjects. Figure 4.5 presents a comparison between distributions of $\rho_{\text{iMFI}}$ (correlation coefficients of culture supernatant response and iMFI) and $\rho(\alpha_{\text{max}})$ (correlation coefficients of GiMFI($\alpha_{\text{max}}$) and culture supernatant response) for the cytokines produced moderately or highly by populations of interest (cDCs, pDCs and mono-
4.3. Results

Table 4.2: \( P \) values obtained by applying the Wilcoxon test with the null hypothesis \( \rho(\alpha_{max}) < \rho_{MFI} \) to the correlation coefficients measured for four cytokines (IL-12, IL-6, TNF-\( \alpha \), and IFN-\( \alpha \)) produced by monocytes, cDCs, and pDCs of PBMC samples from 20 healthy human adult volunteers. The exact values of \( \alpha_{max} \) are given in parenthesis for every cytokine and cell population combination.

<table>
<thead>
<tr>
<th>Antigen Presenting Cell Populations</th>
<th>Cytokines</th>
<th>IL-12</th>
<th>IL-6</th>
<th>TNF-( \alpha )</th>
<th>IFN-( \alpha )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monocyte</td>
<td></td>
<td>0.006</td>
<td>0.001</td>
<td>0.001</td>
<td>0.007</td>
</tr>
<tr>
<td></td>
<td>( \alpha_{max} = 0.3 )</td>
<td>( \alpha_{max} = 0.25 )</td>
<td>( \alpha_{max} = 0.45 )</td>
<td>( \alpha_{max} = 0.1 )</td>
<td></td>
</tr>
<tr>
<td>cDC</td>
<td>0.364</td>
<td>0.018</td>
<td>0.12</td>
<td>Not</td>
<td>Producing</td>
</tr>
<tr>
<td></td>
<td>( \alpha_{max} = 0.6 )</td>
<td>( \alpha_{max} = 0.45 )</td>
<td>( \alpha_{max} = 0.55 )</td>
<td></td>
<td></td>
</tr>
<tr>
<td>pDC</td>
<td>Not</td>
<td>Not</td>
<td>Not</td>
<td>0.005</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Produced</td>
<td>Produced</td>
<td>Produced</td>
<td>( \alpha_{max} = 0.45 )</td>
<td></td>
</tr>
</tbody>
</table>
4.3. Results

Figure 4.4: Example calculation of an optimum $\alpha$ which maximizes the mean value of $\rho(\alpha)$ (correlation coefficients between culture supernatant cytokine response and GiMFI($\alpha$)). Mean of $\rho(\alpha)$ for IL-6 response produced by monocytes is plotted versus $\alpha$ in range $0 \leq \alpha \leq 4$. $\alpha_{\text{max}}$ and $\rho(\alpha_{\text{max}})$ were determined by using simple grid search method.
4.3. Results

Figure 4.5: Distribution of $\rho_i MFI$ (correlation coefficients of culture supernatant cytokine response and iMFI) plotted against distribution of $\rho(\alpha_{max})$ (correlation coefficients of culture supernatant response and GiMFI($\alpha_{max}$)) for IL-12, IL-6, TNF-\(\alpha\), and IFN-\(\alpha\) produced by (A) monocytes, (B) cDCs, and (C) pDCs. These distributions were measured over individual subjects.
4.4 Discussion

This result suggests that statistically \( \rho(\alpha_{\text{max}}) \) is at least as high as \( \rho_{\text{iMFI}} \), if not higher. This was verified by applying the Wilcoxon test with the null hypothesis \( \rho(\alpha_{\text{max}}) < \rho_{\text{iMFI}} \) to perform a pairwise comparison between the values of \( \rho(\alpha_{\text{max}}) \) and \( \rho_{\text{iMFI}} \) for the individual subjects. Table 4.2 presents \( P \) values computed for this test. The null hypothesis was rejected at 2% significance level for all cases other than IL-12p40 and TNF-\( \alpha \) produced by cDCs, indicating \( \rho(\alpha_{\text{max}}) \geq \rho_{\text{iMFI}} \). For these two exceptional cases, iMFI and GiMFI can be used interchangeably with no significant difference in the final results.

4.4 Discussion

While the measurement of cytokines in culture supernatant or serum, and the identification of intracellular cytokines by ICS represent the two most commonly employed assays of CMI, the degree to which these two assays correlate with each other has not been investigated. To quantify the degree of correlation between intracellular cytokine production and their subsequent secretion, and to optimise the mathematical modeling of this correlation (i.e., to improve the calculation of the iMFI), I analysed the correlation of the iMFI and cytokines secreted into culture supernatants as determined by multiplex bead array (Luminex). Our experiment was designed to examine the innate immune responses in PBMC stimulated with known TLR ligands. PBMC receptors are expressed on antigen-presenting cells such as monocytes and dendritic cells that play a critical role in the process of sensing pathogens through pattern recognition [21]. TLRs are strong inducers of inflammatory cytokines such as IL-6, IL-12 and TNF-\( \alpha \); they also induce production of Type I interferon like IFN-\( \alpha \) [104]. Qualitatively, the results of my ICS-Luminex correlation testing were in line with what is currently known about the cell-type specific source of the TLR-induced cytokines I analysed. My analysis indicates that the iMFI and culture supernatant cytokine response to TLR stimulation correlate with each other (Figure 4.2), and this correlation was strong for those cytokines that are highly produced in PBMC. For example, TNF-\( \alpha \) was produced at high levels by most monocytes, which form a significant proportion of the whole PBMC (about 10% of leukocytes in human blood for monocytes) [29] [80], thus resulting in a high level of correlation between iMFI and the concentration of TNF-\( \alpha \) in the culture supernatant.

On the other hand, for cytokines such as IFN-\( \alpha \), the correlation was low given that IFN-\( \alpha \) is produced at high amounts only by pDC, which consti-
4.4. Discussion

tutes a very small fraction (around 0.2-0.4%) of PBMC [165] [111]. Changes in the pDC cytokine production in response to TLR stimulation were hard to detect when looking at all the PBMC, as they were obscured by the lack of response by the more prevalent cell populations (e.g., monocytes and lymphocytes). This could result in critical errors in analysing the functional response of cytokine-producing cells in PBMC, and I therefore considered each cell population separately. For each population of APC, higher correlations were obtained for those cytokines known to be produced by a given APC population (Figure 4.3 Table 4.1). For example, the level of correlation for IFN-α improved significantly when I considered pDCs alone (Figure 4.3C). Similarly for monocytes (the major producer of TNF-α, and a moderate producer of IL-6 and IL-12p40 [80]) the correlation was strong for TNF-α but only moderate for IL-6 and IL-12p40 (Figure 4.3A). As expected cDCs had a significant correlation as these cells are known to produce significant amounts of all three cytokines, IL-12p40, IL-6 and TNF-α [102] [29] (Figure 4.3B). However, it needs to be emphasized that although the correlation could be moderate or high for producers of cytokines, observing a moderate or high correlation does not necessarily mean that a given population under study is the producer of the cytokine of interest. That is, iMFI and culture supernatant responses could be correlated even though the level of production was low. An example would be IFN-α production by monocytes. As expected, the level of IFN-α produced by monocytes was very low, resulting in low iMFI value and a low Luminex response; yet the correlation between these two values was moderate (Figure 4.3A). Overall, My results confirm that a functional response as measured by intracellular cytokine detection modeled by iMFI does correlate with the level of secreted cytokine for those populations that moderately or highly generate cytokines of interest.

With the GiMFI, I improved the iMFI modeling approach by assigning different weights to the magnitude (frequency of cytokine-positive cells) and the quality (the MFI). I hypothesized that the GiMFI could provide the iMFI with a wider, more general applicability by assigning different weights to magnitude (percentage of positive cells) and quality (MFI) of the response. My hypothesis was based on the observation that functional response of cytokine producing cells and Luminex response do correlate for populations that are major producers of cytokine-of-interest, and therefore, I tried to find a formula that maximizes this correlation. GiMFI, my proposed mathematical formula for estimating functional response of cytokine producing cells, assigns different weights to the magnitude (percentage of positive cells) and quality (MFI) of the response. In this formula, $\alpha_{\text{max}}$ is the optimum parameter that maximizes the mean correlation across dif-
4.4. Discussion

ferent subjects. $\alpha_{max}$ was found to always be less than 1 (Table 4.2), suggesting that assigning a lower impact weight to the percentage of cells positive as compared to the MFI in the GiMFI formula would strengthen the correlation. The observation that $\alpha_{max}$ is always less than 1 suggests that the total cytokine production is mainly influenced by the potential of cytokine secretion per cell rather than by the quantity of cytokine-secreting cells within a given cell subset. This can be related to the fact that, after cell activation, cytokine production per cell can usually vary from several orders of magnitude over the background production, while the variation of cell frequency is more limited. In my study, I did not assign a fixed value to $\alpha_{max}$, but adjusted it depending on the target population and cytokine-of-interest. My results showed that the range that $\alpha_{max}$ can take is narrow, with values mostly across a limited range (lower quartile and upper quartiles were 0.29 and 0.47, respectively). Therefore, automatically assigning any constant value to $\alpha_{max}$ within this range can result in a more appropriate model, compared to iMFI where $\alpha$ is equal to 1. Using a Wilcoxon test with null hypothesis $\rho(\alpha_{max}) < \rho(iMFI)$, I showed that for individual subjects, $\rho(\alpha_{max})$ (correlation coefficients of GiMFI($\alpha_{max}$) and culture supernatant response) was greater than or equal to $\rho(iMFI)$ (correlation coefficients of culture supernatant response and iMFI) (Table 4.2), confirming that GiMFI was a better model for estimating functional response of APCs compared to iMFI for most cytokines measured. However, there were two exceptions, namely IL-12p40 and TNF-\(\alpha\) produced by cDCs. Table 4.2 shows that $P$ values for IL-12p40 and TNF-\(\alpha\) produced by cDCs were such that the null hypothesis $\rho(\alpha_{max}) < \rho(iMFI)$ could be neither accepted nor rejected. Figure 4.5B compares the distributions of $\rho(iMFI)$ and $\rho(\alpha_{max})$ for these two cases, showing that $\rho(iMFI)$ and $\rho(\alpha_{max})$ were equally distributed for IL-12p40 and TNF-\(\alpha\) produced by cDCs (i.e., it should be expected that iMFI and GiMFI similarly estimated the functional response of cDCs for IL-12p40 and TNF-\(\alpha\) produced by cDCs). For these two exceptional cases, the curvature of GiMFI($\alpha$) versus $\alpha$ (Figure 4.4) is relatively low, and therefore, the value of GiMFI($\alpha$) does not change significantly with respect to $\alpha$ in this range. Consequently, for these two exceptional cases, GiMFI($\alpha_{max}$) and iMFI=GiMFI($\alpha$=1) get the similar values, and as a result, iMFI and GiMFI could be used interchangeably with neither having an advantage.

In summary, my results show that compared to iMFI, GiMFI in overall tends to provide better estimates of the degree of correlation between intracellular cytokine detected by flow cytometry and the secretion of cytokines into culture supernatant. Therefore, since there were no cases where GiMFI provided qualitatively worse results, GiMFI is a superior approach for cal-
4.4. Discussion

culating the functional response of cytokine producing cells. I have shown here that the intracellular cytokine production of APC in response to TLR stimulation as estimated by iMFI correlated with the level of cytokines secreted into culture supernatant as measured by Luminex assay, and that this correlation was higher for those APC populations that represent the main producer of the cytokine-of-interest. To properly evaluate the potentially improved predictive value of the GiMFI over the iMFI as a correlates of protection (CoP), vaccine-specific GiMFI needs to be correlated with clinical protection in human clinical trials. My results set the stage to apply the GiMFI to adaptive immune responses in vitro, as well as measurements ex vivo in human trials. As this data accumulates, appropriate intermediate biomarkers, such as cytokine concentrations in serum or culture supernatant or intracellular cytokine determination via ICS, should allow more accurate identification of CoP. It would also be interesting to know the generalizability of the $\alpha_{\text{max}}$ values for each APC subpopulation computed from my studies to other analysis that correlates iMFI with some immune response outcome.
Chapter 5

Summary, Conclusions, and Future Work

Recent advances in flow cytometry have provided researchers with the facility to improve their understanding of the tremendously complex immune system. However, the technology is hampered by current manual analysis methodologies. In this thesis, I developed computational methods for the automated analysis of immune response FCM data to address this bottleneck. I hypothesized that highly accurate results could be obtained through learning from the patterns that a biology expert applies when performing the analysis manually.

In general, immune response FCM data analysis consists of three main components: (1) finding homogeneous subsets of immune cells with similar biological characteristics; (2) measuring functional response of cytokine producing cells when stimulated by danger signals; (3) discovering the association between the level of cytokine response of immune cell populations and the underlying pathological conditions. In this study, I aimed to develop computational techniques and use statistical tools in order to automate the first two components of immune response FCM data analysis. Developing computational methods for the third analysis component is left for the future work.

Briefly, my method gets an immune response FCM data sample as the input, and gives its labeled homogeneous immune cell subsets, and their measured functional responses as the outputs. To be more specific, I developed computational methods in order to (1) perform immunophenotyping of immune cells in a semi-automated way, and (2) measure functional response of cytokine producing cells automatically. Automated immunophenotyping itself consisted of (1) automated gating, and (2) semi-automated labeling of cell subsets. Here, I first developed computational methods for each of the above components separately, and then organized them into a semi-automated pipeline, so they all work together as a unified package.

It is worth mentioning that although my main objective was to develop computational methods for immune response FCM data analysis, my semi-
automated approach of immunophenotyping is not limited to immune response FCM datasets. In fact, my approach can be used for any type of FCM dataset where the objective is to find target cell populations and match them across different samples. I showed the high applicability of my clustering method in this sense through applying SamSPECTRAL to a wide range of FCM data samples including telomere and stem cell datasets (Table A.2). In comparison, automated measuring of functional response of FCM datasets is specific to immune response FCM datasets where intracellular markers are used to measure cytokine responses.

In previous work [45, 95, 102], the percentage and MFI of cytokine positive cells were commonly used as independent measurements to estimate functional response of cytokine producing cells using ICS assays. The work of Darrah et al. in [52] extended this approach by combining these two measurements into a formula called iMFI. Since the last component of my semi-automated analysis pipeline included automated measuring of functional response of cytokine producing cells (using percentage and MFI separately), I became interested to see if I could use a formula similar to iMFI [52] in order to combine these two measurements. I hypothesized this could help to improve the estimation of actual quantitative responses of cytokine producing cells (e.g., cytokine concentrations in culture supernatant). In Chapter 4, I demonstrated that both iMFI and its extension GiMFI (the formula that I developed in this study) were correlated to the cytokine responses measured in culture supernatant. My results indicate that as a general rule GiMFI provides better correlative estimations of actual quantitative responses of cytokine producing cells measured in the culture supernatant. Therefore, in the future I will modify the last component of my semi-automated pipeline by substituting percentage and MFI of cytokine positive cells with the GiMFI formula as a better estimator of functional response of cytokine producing cells.

In the summary, my contributions to improving the state of the art computational techniques for FCM data analysis were:

- I developed an automated clustering algorithm called SamSPECTRAL to find homogeneous subsets of cells in flow cytometry data. Comparing to the current state of the art clustering algorithms, SamSPECTRAL proved to be an efficient clustering algorithm that optimizes speed and quality of the clustering (Challenges 2 and 3 in Table 2.2). A particularly significant achievement in applying SamSPECTRAL to FCM data was its successful performance in finding rare cell populations – a problem known to be very challenging both in manual
and automated analysis [115, 135, 179]. SamSPECTRAL was shown to outperform two state of the art model-based clustering algorithms (flowMerge [69] and FLAME [135]) in this regard.

- I developed a learning-based cluster matching method in order to find the best matches of desired cell populations among all clusters generated by clustering algorithms. Compared to the state of the art metaclustering algorithm [135] (an unsupervised cluster matching algorithm), my method has the advantage of being trained based on user specifications, and therefore, it can follow the patterns that a biology expert applies. This characteristic adds flexibility to my method, and results in more accurate identification of cell populations while matching them across different samples.

- I combined SamSPECTRAL clustering algorithm and my learning-based cluster matching method to perform automated immunophenotyping. Previous studies were mostly focused on finding homogeneous cell subsets in a single sample [4, 69, 113], while my method has the advantage of matching the same phenotypes across different samples. The FLAME algorithm [135] also matches cell populations across different samples. Although metaclustering has the potential of being extended to be used in combination with other clustering algorithms, the matching algorithm developed in [135] is dependent on the model-based clustering. That is, the metacusters features are in fact the populations’ modes identified based on the mixture model. In comparison, my learning-based cluster matching is general, and can be used in combination with any clustering algorithm to do automated immunophenotyping of different FCM samples.

- I designed a semi-automated pipeline for the analysis of flow cytometry data. My pipeline includes the following components: (1) data preparation and preprocessing; (2) automated gating by SamSPECTRAL clustering; (3) learning-based cluster matching; and (4) automated measuring of functional response of cytokine producing cells. My semi-automated pipeline extends the analysis pipeline developed in [19]. It has three additional analysis components (1, 3, and 4) that were not included in [19]. Component 3 is specially important as it matches the same populations across different samples, and make them comparable in downstream analysis. Component 4 of my semi-automated pipeline is specific to immune response FCM data analysis, and was not studied in [19].
Chapter 5. Summary, Conclusions, and Future Work

- I applied my semi-automated pipeline to the data from the innate immune response FCM platform developed in [45, 95, 102], in order to fully analyse this data in a semi-automated way. This dataset was previously analyzed through manual analysis [45]. To my knowledge, this is the first complete pipeline specialized for analysis of immune response FCM datasets. On the other hand, compared to the automated analysis framework presented in [18], my pipeline has the advantage that its performance was comprehensively tested on real FCM datasets, instead of standing solely as a theoretical framework. I also demonstrated that my semi-automated pipeline replicates manual analysis with high accuracy, and therefore, it is an appropriate substitute for time-consuming labour-intensive manual tasks. To my knowledge, this is the most comprehensive study that compares the performance of a non-manual FCM analysis pipeline to that of manual analysis extensively (for 216 FCM samples). Here, the comparisons were made in terms of accuracy of (1) semi-automated cell population identification, and (2) automated functional response measurements.

- I conducted a correlation analysis of intracellular and secreted cytokines. While the measurement of cytokines in culture supernatant or serum, and the identification of intracellular cytokines by ICS represent the two most commonly used assays of CMI, the degree to which these two assays correlate with each other was never investigated before this study.

- I derived a formula called GiMFI for measuring functional responses of cytokine producing cells using flow cytometry assay. The GiMFI formula is an extension of the state of the art iMFI formula developed in [52]. My experiments indicated that compared to iMFI formula, the functional responses estimated by GiMFI are more highly correlated to the culture supernatant responses, and therefore, GiMFI can be considered to be more accurate formula for measuring functional response of cytokine producing cells when using ICS assay only.

In the following, I discuss more detailed conclusions and the ideas for the future direction of the research presented in this thesis.
5.1 SamSPECTRAL Clustering for Identification of Cell Populations

In this study, I made spectral clustering applicable to the huge datasets involved in flow cytometry by developing faithful sampling for data reduction and a novel method for measuring similarities between the down sampled data points. SamSPECTRAL does not have \textit{a priori} assumptions on the shape, size and distribution of the clusters, and therefore it can be used for finding the populations of any shape and any size, including rare cell populations.

As shown by the FlowCAP-I competition results, SamSPECTRAL could be applied to a wide range of flow cytometry datasets successfully (Table 2.2). Results of this competition confirmed that SamSPECTRAL was quite successful in identifying cell populations and reproducing manual gating results (overall F-Measure > 0.85). Participating in the FlowCAP-I competition, I was also able to compare the performance of SamSPECTRAL against other state of the art clustering algorithms recently developed in this field. FlowCAP-I results showed that in cases which SamSPECTRAL parameters were prompted to be adjusted separately for each dataset (like Challenges 2 and 3), SamSPECTRAL outperformed most of clustering algorithms in terms of quality of the clustering (measured by overall F-Measure and the rank score). To be more specific, its rank score was the second highest (placing after ADICyt algorithm, among 7 algorithms), and the highest (sharing the first rank with ADICyt, among 11 algorithms) in FlowCAP-I Challenges 2 and 3, respectively. Also considering the runtime of clustering reported as FlowCAP-I results (Table 2.2), SamSPECTRAL was quite efficient by taking only few minutes (< 7 min) to run on a sample using one CPU. Comparing the runtime of SamSPECTRAL to its competitor, ADICyt algorithm which ranked first, SamSPECTRAL was much faster (minutes for SamSPECTRAL compared to hours for ADICyt). Keeping that in mind, SamSPECTRAL can be considered as an efficient clustering algorithm which optimises speed and quality of the clustering, and has improved the field in this regard.

One important characteristic of SamSPECTRAL is that its parameters can be tuned depending on the application. An example of this is to decrease the value of the scaling parameter ($\sigma$) in order to find rare cell populations in the data samples. A possible direction for future work is to verify if it is possible to use adaptive $\sigma$ values that change over the data space, instead of using a single constant value for all parts of the data sample. This can
simply be done by dividing the FCM sample into two or more partitions (by using either a static gate or an automated clustering method), and adjusting $\sigma$ values for each part separately. An example of usability of this feature is the cases for which the rare cell populations are located in a specific part of the data space (requiring a relatively lower $\sigma$ value), and the other parts include only larger and higher density populations (requiring a relatively higher $\sigma$ value).

The R implementation of SamSPECTRAL clustering algorithm is freely available in BioConductor, and can be used by flow cytometry bioinformaticians and biostatisticians. End users can use the SamSPECTRAL module in GenePattern [137] to find cell populations in FCM data. GenePattern is a powerful platform specialized for analysis of gene expression, proteomics, SNP, flow cytometry and RNA-seq datasets, and provides access to many computational tools for analysis of these biological data. Having a web-based interface, GenePattern provides an easy access to such tools for the end users who are not familiar with or are not willing to do specialized programming for data analysis purposes, and therefore, it is a good target for such users. Considering this, I provided the computational techniques that I developed for FCM data analysis in GenePattern in order to make them easy to use for the researchers with lower programming skills.

5.2 Learning-Based Cluster Matching for Labeling Cell Populations

In this study, I developed a novel learning-based cluster matching method that searches for the best matches of the desired populations among all clusters generated by a clustering algorithm in each sample. Without cluster matching, arbitrary numbers are assigned to the clusters obtained by clustering algorithms, and it is not possible to compare corresponding cell populations in different samples. Despite the crucial need for matching the clusters across samples, there was only one previous attempt to address the automated matching of FCM data clusters generated by unsupervised clustering [135]. Automated cluster matching for FCM data is a challenging problem complicated by variations between the data samples. My method of cluster matching incorporates biological expert knowledge to label cell populations and to improve the identification of target cell populations in different samples. My cluster matching method is general and can be combined with any clustering algorithm to find the best matches of target populations in the cluster set generated by any clustering algorithm and to assign sensible
labels to them.

There are a number of factors in my semi-automated cluster matching method that add flexibility and result in reliable identification of target cell populations. One of these factors is that the cluster matching features used for each target cell population can be selected independently from those used for other populations. To my knowledge, this is the first cluster matching method for FCM data that has this characteristic. The state of the art metachlustering approach developed by Pyne et al. [135] uses the same set of features for all clusters in order to find metaclusters and to match populations across different samples. Another factor that adds flexibility to my method is that the features of my cluster matching can be selected from a wide range of cell population characteristics including size, shape, density, and centres of the populations. In addition, it is possible to apply different weights to different features depending on their importance, in order to find the desired cell populations more accurately.

In this study, I selected the features manually following the patterns that were used in manual analysis by a human expert. However, in the future, this manual identification of features can be replaced by an automated method for identifying the features. Here the main idea would be to select the discriminative features by looking at the training samples. In Section 3.6.5, I explained how one could divide all automatically identified clusters of all training samples into two groups, \( S(p) \) and \( T(p) \), based on their similarities to the desired target population. Here, \( S(p) \) included all training samples’ clusters that were similar to the target population \( p \), and \( T(p) \) included all training samples’ clusters that were dissimilar to the target population \( p \). As the first step of the automated feature identification method, a wide range of features including cluster size, shape, location, density, etc. would be measured for all clusters of \( S(p) \) and \( T(p) \). Then a feature selection method like [101] would be used to select the most discriminative features, such that the selected features are highly similar for all clusters of \( S(p) \), and highly different between two clusters, one from \( S(p) \) and another one from \( T(p) \). Finally, the selected features would be used to form \( F_{T(p)} \) and \( F_{S(p)} \) (see Section 3.6.5). The rest of the learning-based cluster matching algorithm would follow as described in Sections 3.6.6 and 3.6.7.

Feature selection methods are frequently used for three main objectives: (1) to improve performance of the model; (2) to make the model faster and more cost-effective; and (3) to explore the characteristics of processes that generated the data [54, 144]. In our case, the main objective is to improve accuracy of prediction (objective 1), and therefore, algorithms that are specialized for this objective are preferred ([101] is an example). Fea-
ture selection methods can be classified into three categories, filter methods [24, 86, 101, 176], wrapper methods [68, 92, 100, 154] and embedded methods [61, 83], among which the filter methods are the only ones that are independent of the learning algorithm used. This characteristic of filter methods makes them appropriate candidates for my study. I expect that using an efficient feature selection scheme with the above characteristics will lead to (1) considerable improvement in accuracy of cell population identification (e.g., increase of sensitivity specially for the rare cell populations like cDCs and pDCs from medium (≃ 0.7) to high (> 0.9)), and (2) reduction in the number of failures in finding the best matches of target cell populations (e.g., from 11% failures in this study to just 2-3% failures).

In FCM datasets, moderate to high variations are commonly observed between the features (e.g., frequency, position and local density) of the same populations of different samples. These variations can be caused by specific stimulations, or it can be simply due to differences between the subjects. One example of this is the monocyte population for which the population local density and frequency change with TLR-ligand stimulation. It is clear that if a population feature (e.g., frequency) is highly variable between different samples (e.g., standard deviation > 0.25 for a feature in range [0,1]), then not only using that feature has no value as it is not specific to the population of interest, but also it can mislead the cluster matching algorithm (like any other learning-based method). However, there might be cases for which one can find groups of samples, such that the feature of interest is similar for samples within a group, but different for samples from different groups (multi-class problem). In such situations, one can still use the feature of interest as a cluster matching feature, but after grouping the samples based on their similarities. In that case, the training samples should be selected randomly for each group separately. In [135], Pyne et al. mentioned that the samples of different classes should be used separately for metaclustering. However, they did not address how to group the samples based on the similarity of their patterns. A significant strength of my method was that the variability in target sample population surface marker expression secondary to stimulation with relevant ligands or intersubject variabilities was addressed by automated grouping of samples with similar patterns as a preprocessing step performed before applying cluster matching. In Appendix F, I explain how I grouped the samples, such that samples with similar monocyte features stayed in the same group.

As my experiments supported, my learning-based cluster matching algorithm was quite successful in identification of monocytes in the presence of variations in the frequency feature as high as $\sigma = 0.15$ (Appendix F).
5.2. Learning-Based Cluster Matching for Labeling Cell Populations

However, in order to find the maximum feature variability that my cluster matching can tolerate, a more comprehensive experiment is required, but left for the future work. The simplest way to do this is to group the samples several times with different levels of intersample variabilities in the features (e.g., different standard deviations for the frequency feature), and measure the success rate of the algorithm each time. My expectation is that by increasing the intersample variability, the success rate drops greatly at a certain point. This point gives an estimation of the maximum feature variability that my learning-based cluster matching algorithm can withstand.

My experiments showed that selecting only a few samples (≤ 6) from each sample group as the training set was sufficient to accurately identify target cell populations in the rest of the samples of the same group (Figures 3.3 and 3.4). Since in this way operators need to manually gate just a few samples (≤ 6) for each group, this highly reduces the human time required for analysis of the data. The amount of human time that can be saved in this way depends on the size of the dataset, and can vary from several hours (≃ 200 FCM samples) to months (≃ 2000 FCM samples). My learning-based cluster matching itself is also fast. In my experiments, it took less than 2 minutes to perform the learning phase of my algorithm on a 2.7GHz processor. Finding the best matches of target populations of a non-training sample took about 20 seconds per target population per sample. Please see Appendix G for more detail about the time complexity of my learning-based cluster matching.

In many applications of FCM data analysis, it is desirable to identify pre-defined and well-established target cell populations in a large number of FCM data samples, as the first step of the data analysis. These identified cell populations are later used to discover the immunological patterns in the data samples. Considering the accuracy and time efficiency of my cluster matching method, it can be efficiently applied to such large datasets to identify target cell population accurately, whereas this goal is not achievable by manual analysis, considering the time and human effort it requires. Automated metaclustering approaches [135] in combination with a clustering algorithm can also be used to label cell populations. However, metaclustering is an unsupervised cluster matching method which does not learn from the human experts' specifications (e.g., features of the target populations including their approximate locations and sizes). In comparison, my cluster matching approach is a tunable learning-based method that learns from the training samples in which target cell populations are manually identified by human experts. Having a reliable semi-automated FCM data analysis method which follows experts' patterns for identification of cell populations
facilitates the analysis of large FCM datasets. This can result in including a huge number of data samples in the study without being concerned about the time and human effort required for the data analysis, and at the same time, taking advantage of following the patterns that human experts apply for identification of target cell populations. Increasing sample size in turn can boost the statistical power of tests.

As mentioned before, my method of cluster matching is useful for cases in which a set of predefined and well-established target cell populations are to be identified automatically. However, if somebody is interested in matching all of the unknown cell subsets generated by a clustering algorithm across samples (e.g., for FCM data exploration, where cell populations are unknown), an unsupervised method such as the metachustering approach developed in [135] may be preferred. Since my cluster matching method incorporates biological expert knowledge, it can identify cell populations more accurately compared to an unsupervised approach for cluster matching, but on the condition that the target populations are already known and have been identified in a set of training samples.

My learning-based cluster matching is going to be added to BioConductor, so that flow cytometry bioinformaticians and biostatisticians can easily use it for their FCM data analysis. I also aim to make my algorithm available as an FCM analysis module in GenePattern [137], so that the end users with lower skills in programming can easily use the GenePattern web-based interface to apply my algorithm to their FCM datasets.

5.3 Semi-Automated Pipeline for Analysis of Flow Cytometry Data

In this study, I designed and implemented a semi-automated pipeline to fully analyse immune response flow cytometry data. A significant strength of the method is the low sensitivity of the final results to the initial parameters. This was achieved by repeating the clustering algorithm multiple times using different initial parameters, to facilitate more stable cell population identification results.

Since the experiments on 216 FCM innate immune response samples showed that my pipeline can reproduce manual results accurately, it seems to be reliable enough to be used with other large datasets that otherwise could not be analyzed. In the future, my pipeline will be used to fully analyze the innate immune response of a more comprehensive cohort of neonates and adults that may include up to 10,000 subjects (180,000 FCM samples).
5.4 Correlation Analysis of Intracellular and Secreted Cytokines

Considering the fact that my pipeline was able to find target cell populations accurately and measure the functional response of cytokine producing cells automatically, we aim to make this pipeline available to the end users through providing data analysis modules in GenePattern [137]. These analysis modules would facilitate complete analysis of immune response in huge FCM datasets. At the same time, using the clustering and cluster matching components of this pipeline, end users will be able to perform automated immunophenotyping efficiently and accurately. GenePattern specially is a good choice for the above purposes, as it allows to build a multi-step analysis pipeline that enables using the output of one computational module as the input of another one.

5.4 Correlation Analysis of Intracellular and Secreted Cytokines

While the measurement of cytokines in culture supernatant or serum, and the identification of intracellular cytokines by ICS represent the two most commonly used assays of CMI, the degree to which these two assays correlate with each other has not been investigated before this study. Investigating the correlation of the intracellular cytokine measurement derived from a human innate immune response ICS assay with functional cytokine release into the culture supernatant was the first step toward extending the iMFI concept to humans, as innate cytokines need to be released to have a functional impact.

To properly evaluate the potentially improved predictive value of the GiMFI over the iMFI as a correlates of protection (CoP), vaccine-specific GiMFI needs to be correlated with clinical protection in human clinical trials. My results set the stage to apply the GiMFI to adaptive immune responses in vitro, as well as measurements ex vivo in human trials.
Bibliography


Bibliography


Bibliography


Bibliography


Bibliography


[122] M. Malec, VH. Velden, E. Bjorklund, JM. Wijkhujs, S. Soderhall, J. Mazur, M. Bjorkholm, and A. Porwit-MacDonald. Analysis of min-


Appendix A: Testing Stability of SamSPECTRAL Results

In the results section, I explained that the resolution of the sample points (Figure 2.2) is high enough such that by repeating the randomised faithful sampling procedure, the outcome of SamSPECTRAL does not vary significantly. The following experiment is performed to confirm this observation quantitatively.

In this experiment I used F-measure, which is known to be appropriate for comparing clustering results of flow cytometry data [3]. F-measure varies in range 0-1 and reaches its best value at 1 when the two clustering results are identical. I ran SamSPECTRAL on a sample from the stem cell data set 20 times and compared the final results. The F-measure values obtained by pairwise comparison between the final results had mean=0.98, median=0.98 and standard deviation 0.0097.
Appendix B: Performance of SamSPECTRAL on Synthetic Data

Figure A.1: Performance of SamSPECTRAL on synthetic data. (a) This synthetic two dimensional data consists of a normal distribution with 30,000 points, four normal distribution each with 300 points and a uniform background noise with 4000 points. (b) Around 3000 sample points are picked up by faithful sampling. These are distributed almost uniformly in the space, therefore, almost all information about density will be lost if one considers only the samples points. (c) The final outcome of SamSPECTRAL confirms that the information about density could be retrieved by properly assigning weights to the edges of the graph. The high density cluster is shown in red and the surrounding sparser clusters are shown in yellow, light blue, green and black.

We performed the following experiment to show the effect of edge weights on performance of spectral clustering. As shown in Figure A.1, we produced synthetic data containing one normal distribution with relatively high density surrounded by four relatively small clusters with lower densities. The number of points in each small cluster is less than 0.01% of the whole data and noise is added to the data space uniformly (Figure A.1a). For the central dense distribution, we set $\sigma_{xx} = \sigma_{yy} = 2$, $\sigma_{xy} = \sigma_{yx} = 0$ and the
surrounding clusters are normal distributions with $\sigma_{xx}^1 = 0.08$, $\sigma_{yy}^1 = 0.30$, $\sigma_{xy}^1 = \sigma_{yx}^1 = 0$, $\sigma_{xx}^2 = 0.07$, $\sigma_{yy}^2 = 0.08$, $\sigma_{xy}^2 = \sigma_{yx}^2 = 0$, $\sigma_{xx}^3 = 0.50$, $\sigma_{yy}^3 = 0.10$, $\sigma_{xy}^3 = \sigma_{yx}^3 = 0$, $\sigma_{xx}^4 = 0.10$, $\sigma_{yy}^4 = 0.70$, $\sigma_{xy}^4 = \sigma_{yx}^4 = 0$. The R code to produce this synthetic data and run SamSPECTRAL on it is provided in additional file 7.

After faithful sampling is done (Figure A.1b), the sample points are distributed almost uniformly, and the information about the local density of original data is lost. However, faithful sampling provides us with more information than only the sample points. It will also return the members of each community and my data reduction scheme uses this information to assign weight to the edges. According to formulas 2.3 and 2.4, the more populated and closer two communities are, the higher the weight between them will be (Figure 2.1c). According to Figure A.1c, this strategy is successful in retrieving information about local density as all the five clusters are distinguished properly by SamSPECTRAL.
Appendix C: Comparing Uniform Sampling Against Faithful Sampling

I observed that some low density populations disappeared entirely when simple uniform sampling was employed. To investigate the effect of this phenomenon on the final clustering results, I performed an experiment on a sample of the stem cell dataset that contained 48,000 events in 3 dimensions. First, 3,000 data points were selected uniformly at random. Then, I assigned a label to each of these selected points by applying classical spectral clustering on them. Finally, for each original data point, the label of the closest selected point was considered as its cluster label. Figures A.2d and A.2e show the results of this approach and SamSPECTRAL, accordingly. The red population that was distinguished by SamSPECTRAL correctly in Figure A.2e consists of only 4% of the data. This population could not be distinguished properly by any setting of the parameters after uniform sampling (Figure A.2d).
Appendix C: Comparing Uniform Sampling Against Faithful Sampling

Figure A.2: Comparing uniform sampling with faithful sampling. Directly applying classical spectral clustering is not efficient on this sample of the stem cell dataset which contains 48000 cytometry events in 3 dimensions. (a) Although only 2115 data points were selected by faithful sampling, each population has a considerable number of representatives in the selected points. (b) 3000 points were selected by uniform sampling. The low density population in the middle of the plot consists of only 55 sample points resulting in mixing this population with a high density one incorrectly (d). (c) The result of SamSPECTRAL on the original data is satisfactory because the low density red population and other high density populations are identified properly.
Appendix D: FlowCAP-I Competition

Pre-Processing Steps Applied to FlowCAP-I datasets

Table A.1 summarizes the description of FlowCAP-I datasets. The following pre-processing steps were applied to all FlowCAP-I datasets before circulating to participants: (1) compensation (to account for the overlap of emission spectra from antibody fluorescent labels); (2) logicle transformation (to scale data appropriately for visualization) [129]; (3) gating for dead cell removal.

Performance Evaluation Methods in FlowCAP-I

F-measure – In order to evaluate the performance of each clustering algorithm, its results were compared against the results of manual gating performed by biology experts. Previously, it was shown that F-measure is a good candidate for comparing clustering solutions against manual gating results [3]. Combining measures of precision and recall, F-Measure maximizes the trade-off between these two measurements. It can be defined as $F = (2 \cdot P \cdot R) / (P + R)$, where $P$ (precision) is the number of cells assigned correctly to a cluster divided by the total number of cells in that cluster, and $R$ (recall) is the number of cells assigned correctly to a cluster divided by the number of cells that should have been assigned to that cluster. F-measure values are always in $[0, 1]$ range, with 1 indicating a perfect prediction.

Rank Score – In order to obtain an overall ranking of the algorithms, their rank scores were calculated as the sum of fractional rankings of each algorithm across different datasets. Fractional ranking is based on the Borda count strategy [64]: For $N$ algorithms, the top algorithm scored $N$ points, the second one scores $N-1$ points, and so on [27]. The last algorithm scores 1 point. In case of overlappings, the average of points is used. For $D$ datasets, rank score values are in the range $[D, N \times D]$. An algorithm that scored the best in all datasets would get a rank equal to $N \times D$. 

139
Table A.1: Summary of the description of the FlowCAP-I datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#Samples</th>
<th>#Events</th>
<th>Analyte</th>
<th>Detector</th>
<th>Reporter</th>
<th>Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>GvHD</td>
<td>12</td>
<td>14,000</td>
<td>CD4</td>
<td>Anti-CD4</td>
<td>FITC</td>
<td>BCCRC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8b</td>
<td>Anti-CD8b</td>
<td>PE</td>
<td>&amp;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD3</td>
<td>Anti-CD3</td>
<td>PerCP</td>
<td>TreeStar</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8</td>
<td>Anti-CD8</td>
<td>APC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD5</td>
<td>Anti-CD5</td>
<td>CY5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8b</td>
<td>Anti-CD8b</td>
<td>PE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD19</td>
<td>Anti-CD19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DLBCL</td>
<td>30</td>
<td>5,000</td>
<td>CD3</td>
<td>Anti-CD3</td>
<td>FITC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD5</td>
<td>Anti-CD5</td>
<td>PE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD19</td>
<td>Anti-CD19</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8</td>
<td>Anti-CD8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD3/CD8</td>
<td>Anti-CD3/CD8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ND</td>
<td>30</td>
<td>17,000</td>
<td>CD5</td>
<td>Proprietary</td>
<td>FITC</td>
<td>Amgen</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD6</td>
<td>Proprietary</td>
<td>PerCPy5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8</td>
<td>Proprietary</td>
<td>PacificBlue</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8</td>
<td>Proprietary</td>
<td>PacificOrange</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8</td>
<td>Proprietary</td>
<td>APC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD45</td>
<td>Proprietary</td>
<td>Alexa700</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD56</td>
<td>Proprietary</td>
<td>PE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD56</td>
<td>Proprietary</td>
<td>PECy5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD56</td>
<td>Proprietary</td>
<td>PECy7</td>
<td></td>
</tr>
<tr>
<td>WNV</td>
<td>13</td>
<td>100,000</td>
<td>IFNγ</td>
<td>Anti-IFNγ</td>
<td>PEA</td>
<td>McMaster</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD3</td>
<td>Anti-CD3</td>
<td>PECy7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD4</td>
<td>Anti-CD4</td>
<td>APC</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IL17</td>
<td>Anti-IL17</td>
<td>AlexaFluo700</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD8</td>
<td>Anti-CD8</td>
<td>CFSE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Free Amines</td>
<td>Anti-CD45.1</td>
<td>CFSE</td>
<td></td>
</tr>
<tr>
<td>HSCT</td>
<td>30</td>
<td>10,000</td>
<td>CD45.1</td>
<td>Anti-CD45.1</td>
<td>FITC</td>
<td>BCCRC</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Ly65/Mac1</td>
<td>Anti-Ly65/Mac1</td>
<td>PE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Dead Cells</td>
<td>NA</td>
<td>PI</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CD45.2</td>
<td>Anti-CD45.2</td>
<td>APC</td>
<td></td>
</tr>
</tbody>
</table>
Appendix E: Flow Cytometry Datasets Used in This Study

In this study, I have used a number of FCM datasets to validate the performance of the computational techniques and algorithms that I have developed for flow cytometry. In Chapter 2, I used four different types of FCM data including (a) stem cells, (b) telomere, (c) GvHD, and (d) viability, in order to verify the performance of SamSPECTRAL clustering in identification of homogeneous subsets of cells forming cell populations. In addition, I participated in FlowCAP-I competition, and applied SamSPECTRAL algorithm to five different datasets that were available for different challenges of this competition. In Chapter 3, I used the PBMC samples of neonates at birth, 1-year, and 2-years old. This FCM dataset is basically the one that was used in [45] study. The full semi-automated analysis pipeline that I have developed was applied to this dataset to identify APC cell populations, and to measure functional response of cytokine-producing cells. In Chapter 4, I used the PBMC samples of adults, and then considering both intracellular cytokine response obtained from flow cytometry and culture supernatant response measured by Luminex assay, I derived a formula for measuring functional response of cytokine producing cells using flow cytometry assay only. Table A.2 summarizes the flow cytometry datasets that I have used in different chapters of my thesis. This table also gives a summary of different tasks applied to these datasets, and whether these tasks were manual or automated.
## Appendix E: Flow Cytometry Datasets Used in This Study

### Table A.2: Flow cytometry datasets

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Dataset</th>
<th># Samples</th>
<th>Markers</th>
<th>Gating</th>
<th>Labeling</th>
<th>Immune Response</th>
<th>Provider</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Stem Cells</td>
<td>34</td>
<td>CD45.1 Ly1 CD45.2</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC</td>
</tr>
<tr>
<td>2</td>
<td>Telomere</td>
<td>—</td>
<td>FSC SSC LDS</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC</td>
</tr>
<tr>
<td>2</td>
<td>GvHD</td>
<td>—</td>
<td>CD8 CD3 CD4 CD134</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC</td>
</tr>
<tr>
<td>2</td>
<td>Viability</td>
<td>—</td>
<td>GFP PI</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC</td>
</tr>
<tr>
<td>2</td>
<td>FlowCAP-I GvHD</td>
<td>12</td>
<td>CD4 CD8b CD3 CD6</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC &amp; TreeStar</td>
</tr>
<tr>
<td>2</td>
<td>FlowCAP-I DLBCL</td>
<td>30</td>
<td>CD19 CD3 CD5</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC</td>
</tr>
<tr>
<td>2</td>
<td>FlowCAP-I HSCT</td>
<td>30</td>
<td>CD45.1 Ly65/Mac1 Dead Cells CD45.2</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>BCCRC</td>
</tr>
<tr>
<td>2</td>
<td>FlowCAP-I WNV</td>
<td>13</td>
<td>IFNγ CD3 CD4 IL17 CD8 Free Amines</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>MacMaster</td>
</tr>
<tr>
<td>2</td>
<td>FlowCAP-I ND</td>
<td>30</td>
<td>CD56 CD8 CD45 CD3/CD14 6 other unknown markers</td>
<td>AG</td>
<td>—</td>
<td>—</td>
<td>Amgen</td>
</tr>
<tr>
<td>3</td>
<td>Neonates PBMC at birth, 1 year, and 2 years old</td>
<td>216</td>
<td>MHC-II CD14 CD11c CD123 IL-6 IL-12 TNF-α IFN-α</td>
<td>MG &amp; AG ML &amp; AL M &amp; A</td>
<td>BC Children Hospital</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Adult PBMC</td>
<td>600</td>
<td>MHC-II CD14 CD11c CD123 IL-6 IL-12 TNF-α IFN-α</td>
<td>MG ML M &amp; A</td>
<td>BC Children Hospital</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

M: Manual
A: Automated
MG: Manual Gating
AG: Automated Gating by SamSPECTRAL
ML: Manual Labeling
AL: Automated Labeling by Learning-Based Cluster Matching
Appendix F: Finding Groups of Samples With Similar Monocytes Features

In the concept of selecting features for a target population, there might be cases for which one can find groups of samples, such that the feature of interest is similar for samples within a group, while it is different for samples from different groups. In such cases, it is required to first group the samples, and then randomly select the training samples for each group of samples separately. One example of this is monocyte population for which the population local density and frequency change by TLR-ligand stimulation. Even for the same TLR-ligand stimulation, different samples may have monocytes with different features. Here, I explain two algorithms that I developed to group the samples, such that samples with similar monocyte features (frequency and density) stayed in the same group. The algorithms for grouping samples can change depending on the characteristics of the population of interest and the type of variations exist in the population features.

Algorithm A:

1. Remove non-antigen presenting cells (MHCII− cells). This was done automatically by estimating the histogram of MHCII marker, and defining the threshold for separating MHCII+ cells from MHCII− cells as the local minimal of the histogram in a predefined range.

2. Use gaussNorm method [84] to normalize CD14 marker for all samples.

3. Compute the CD14 marker kernel probability density for each sample separately.

4. Define the sample feature vector as its CD14 marker kernel probability density.
Appendix F: Finding Groups of Samples With Similar Monocytes Features

5. Apply hierarchical clustering algorithm to cluster the samples based on the similarities between their feature vectors (values of CD14 marker kernel probability density).

Applying the above algorithm for grouping 216 FCS samples based on the similarities of monocytes frequency and density features resulted in generating 6 different groups of samples. Figure A.3 shows the average density plots and the kernel probability density (pdf) of the CD14 marker for each group obtained by Algorithm A.

Figure A.3: The average density plots and the kernel probability density of the CD14 marker for 6 groups of samples obtained by Algorithm A. Groups A-F included 63, 85, 23, 21, 19, and 5 FCM samples, respectively (Table 3.7). Here, each group of samples had a distinct pattern for probability density.

Figure A.4 depicts frequency of manually gated monocytes for different groups (A-F) which were obtained by Algorithm A. Here, the frequency was calculated as the number of cells in monocyte population divided by the
Appendix F: Finding Groups of Samples With Similar Monocytes Features

total number of MHCII$^+$ cells. Mean and standard deviation of monocytes’ frequencies were 0.17 ($\sigma = 0.07$), 0.29 ($\sigma = 0.08$), 0.35 ($\sigma = 0.15$), 0.08 ($\sigma = 0.04$), 0.61 ($\sigma = 0.1$), and 0.36 ($\sigma = 0.06$) for groups A to F, respectively.

![Boxplots of frequency of manually gated monocytes for different groups (A-F) which were obtained by Algorithm A.](image)

Figure A.4: Boxplots of frequency of manually gated monocytes for different groups (A-F) which were obtained by Algorithm A.
Appendix F: Finding Groups of Samples With Similar Monocytes Features

My other alternative algorithm for grouping samples based on similarities of their monocyte features (frequency and density) was as follows.

Algorithm B:

1. Remove non-antigen presenting cells (MHCII\(^{-}\) cells). This was done automatically by estimating the histogram of MHCII marker, and defining the threshold for separating MHCII\(^{+}\) cells from MHCII\(^{-}\) cells as the local minimal of the histogram in a predefined range.

2. Use gaussNorm method [84] to normalize CD14 marker for all samples.

3. Compute the CD14 marker kernel probability density for each sample separately.

4. Find two main peaks of CD14 marker kernel probability density. These two peaks correspond to the centers of Monocyte and Other APC populations.

5. Measure values of CD14 marker kernel probability density at two peaks found in the previous step, and use them as the features of the sample.

6. Apply K-means clustering algorithm to cluster the samples based on the similarities between their feature vectors (values of CD14 marker kernel probability density at two peaks).

Applying the above algorithm for grouping 216 FCS samples based on the similarities of monocytes frequency and density features resulted in generating 4 different groups of samples. Figure A.5 shows the average density plots and the kernel probability density (pdf) of the CD14 marker for each group obtained by Algorithm B. Here, each group of samples had a distinct pattern for probability density. To be more specific, samples of group A had similar local densities for Other APC (first peak) and Monocyte (second peak) populations (Figure A.5A). Samples of group B had relatively higher values of local densities for Other APC population compared to that of Monocytes (A.5B). Samples of group C had relatively lower values of local densities for Other APC population compared to that of Monocytes (A.5C). Samples of group D are mostly outstanding cases that were not similar to any other groups of samples (A.5D).
Appendix F: Finding Groups of Samples With Similar Monocytes Features

Figure A.5: The average density plots and the kernel probability density of the CD14 marker for 4 groups of samples obtained by Algorithm B. Groups A-D included 100, 76, 35, and 5 FCM samples, respectively (Table 3.8). Here, each group of samples had a distinct pattern for probability density.
Appendix G: Time Complexity of Learning-Based Cluster Matching

My learning-based cluster matching method consists of three main parts. Here, I discuss the time complexity of each part of the algorithm one by one.

Part 1: Learning From Training Samples

If the number of target populations is $P$, and $U$ is the total number of all training samples’ clusters, then the time complexity of this part of our learning-based cluster matching algorithm is $O(m \times P \times U)$. In fact, the required time for dividing clusters into two groups (similar or dissimilar to target populations) is $O(P \times U)$, and the time complexity of calculating $m$ features for all clusters that belong to $S_p$ or $T_p$ is $O(m \times P \times U)$. Here, $S_p$ represents for the set of all training samples’ clusters that are similar to the target population, and $T(p)$ includes all training samples’ clusters that are dissimilar to the target population. This is calculated by considering that $U = |S_p| + |T_p|$ for all $p$.

Part 2: Likelihood Ratio Measurement

For each population $p$, the time complexity of computing the distance vectors for all pairs of the sets $A^p$ and $B^p$ are $O(m \times |A^p|)$ and $O(m \times |B^p|)$, respectively. If the required time for computing one dimensional Gaussian kernel density estimation at $x$ evaluation points given $y$ sample points is shown by $t(x, y)$, then the time complexity of Gaussian kernel density estimation for $m$
dimensions of sets $A^p$ and $B^p$ are $O(m \times (t(x, |A^p|) + t(x, |B^p|)))$. Fast computations of kernel density estimation are linear with regard to the number of evaluation points and sample points, that is, $O(t(x, y)) = O(x + y)$ [136]. Therefore, $O(m \times (t(x, |A^p|) + t(x, |B^p|))) = O(m \times (x + |A^p| + |B^p|))$. If $x$ gets a constant value initialized manually at the beginning of our algorithm, then the time complexity of this part of our algorithm is $O(m \times (|A^p| + |B^p|))$, for each population $p$. Since $|A^p| = |S^p|^2 - |S^p|$, and $|B^p| = |S^p||T^p|$, and $U = |S^p| + |T^p|$, the total time complexity of this part of our algorithm is simplified to $O(m \times U \times |S^p|)$, for each populations $p$. Therefore, considering $S_{max} = max(|S^p|)$, the total time complexity of this part of our algorithm for $P$ populations is $O(m \times P \times U \times S_{max})$. In other words, the time complexity for the learning phase of my algorithm is linear with respect to the number of features ($m$), number of target populations ($P$), total number of clusters in all training samples ($U$), and the maximum number of clusters in all training samples that sufficiently overlap with the target populations ($S_{max}$).

Part3: Scoring Clusters to Find the Best Match of a Target Population

The time complexity of measuring $m$ features of the $i^{th}$ non-training sample is $O(m \times P \times |S_i|)$, where $S_i$ is the set of all clusters generated by applying a clustering algorithm to the $i^{th}$ non-training data sample. The required time for computing the distance vectors between all pairs of clusters $(c, s)$, $c \in S_i$ and $s \in S^p$, and the required time for computing the log likelihood ratio values is $O(m \times |S_i| \times |S^p|)$, for each population $p$. Therefore, for each non-training sample, the total time complexity of this part of our algorithm is $O(m \times P \times |S_i| \times S_{max})$, for $P$ populations. That is, the time complexity required for a new non-training sample is linear with respect to all of the parameters ($m$, $P$, $|S_i|$, and $S_{max}$).
Appendix H: SamSPECTRAL Package in R BioConductor

Introduction

Data analysis is a crucial step in most of recent biological research areas such as microarray techniques, gene expression and protein classification. A classical approach for analysing biological data is to first group individual data points based on some similarity criterion, a process known as clustering, and then compare the outcome of clustering with the desired biological hypotheses. Spectral clustering is a non-parametric clustering method which has proved useful in many pattern recognition areas. Not only it does not require a priori assumptions on the size, shape and distributions of clusters, but it has several features that make it an appropriate candidate for clustering biological data:

- It is not sensitive to outliers, noise or shape of clusters.
- It is adjustable so we can make use of biological knowledge to adapt it for a specific problem or dataset.
- There is mathematical evidence to guarantee its proper performance.

However, because of the machine limitations, one faces serious empirical barriers in applying this method for large data sets. SamSPECTRAL is a modification to spectral clustering such that it will be applicable on large size datasets.

How to Run SamSPECTRAL?

SamSPECTRAL is an R package source that can be downloaded from BioConductor. In Linux, it can be installed by the following command:
How to Run SamSPECTRAL?

R CMD INSTALL SamSPECTRAL_x.y.z.tar.gz

where x.y.z. determines the version.

The main function of this package is SamSPECTRAL() which is loaded by using the command library(SamSPECTRAL) in R. Before running this function on a data set, some parameters are required to be set including: normal.sigma and separation.factor. This can be best done by running the algorithm on some number of samples (Normally, 2 or 3 samples are sufficient). Then the function SamSPECTRAL() can be applied to all samples in that data set to identify cell populations in each sample data.

An Example

This example shows how SamSPECTRAL can be run on flow cytometry data. If \( f \) is a flow frame (which is normally read from an FCS file using flowCore), then the object "small" in the following example should be replaced by expr(f).

```r
> library(SamSPECTRAL)
> data(small_data)
> full <- small
> L <- SamSPECTRAL(full, dimension = c(1, 2, 3), normal.sigma = 200,
+   separation.factor = 0.39)
> plot(full, pch = ".", col = L)
```

SamSPECTRAL is done. The results are in \( L \), a vector that provides a numeric label for each event. All events with equal label are in one component and isolated outliers are labelled by \( NA \). The following piece of code is not a part of the analysis and it is included only for more clear presentation of the results. The code computes the frequency of events in each component and adds a legend to the figure.

```r
> plot(full, pch = ".", col = L)
> frequency <- c()
> minimum.frequency <- 0.01
> frequency.large <- c()
> labels <- as.character(unique(L))
> for (label in labels) {
+   if (!is.na(label)) {
+     frequency[label] <- length(which(L == label))/length(L)
+   }
+ }
```
Adjusting Parameters

For efficiency, one can set \( m = 3000 \) to keep the running time below 1 minute by a 2 GHz processor and normally the results remained satisfactory.
Adjusting Parameters

for flow cytometry data. The separation factor and scaling parameter ($\sigma$) are
two main parameters that needed to be adjusted. The general way is to run
SamSPECTRAL on one or two random data samples of a flow cytometry
data set and try different values for $\sigma$ and separation factor. Then, the
selected parameters were fixed and used to apply SamSPECTRAL on the
rest of data samples. An efficient strategy is explained by the following
example.

Example

First we load data and store the transformed coordinates in a matrix called
full:

```r
> data(small_data)
> full <- small
```

The objects needed for creating this vignette can be directly computed
or loaded from previously saved workspace to save time. The later increases
the speed of building this vignette.

```r
> run.live <- FALSE
```

The following parameters are rarely needed to be changed for flow cy-
tometry data:

```r
> m <- 3000
> community.weakness.threshold <- 1
> precision <- 6
> maximum.number.of.clusters <- 30
```

The following piece of code, scales the coordinates in range [0,1]:

```r
> for (i.column in 1:dim(full)[2]) {
+   ith.column <- full[, i.column]
+   full[, i.column] <- (ith.column - min(ith.column))/(max(ith.column) -
+                         min(ith.column))
+ }
> space.length <- 1
```

To perform faithful sampling, we run:

```r
> society <- Building_Communities(full, m, space.length,
> community.weakness.threshold)
> plot(full[society$representatives, ], pch = 20)
```
We intend to first find an appropriate value for $\sigma$ and then set separation factor. Note that normal.sigma=$\frac{1}{\sigma^2}$, therefore, decreasing normal.sigma is equivalent to increasing $\sigma$ and visa versa. We start with normal.sigma=10:

```r
> normal.sigma <- 10
> conductance <- Conductance_Calculation(full, normal.sigma, space.length, + society, precision)
> if (run.live) {
+   clust_result.10 <- Civilized_Spectral_Clustering(
+     full,
+     maximum.number.of.clusters,
+     society, conductance, stabilizer = 1)
+   eigen.values.10 <- clust_result.10@eigen.space$values
+ } else data("eigen.values.10")
> plot(eigen.values.10[1:50])
```
We observe that the eigen values curve does not have a "knee" shape. So we increase sigma:

```r
> normal.sigma <- 1000
> conductance <- Conductance_Calculation(full, normal.sigma, space.length,
+    society, precision)
> if (run.live) {
+    clust_result.1000 <- Civilized_Spectral_Clustering(full,
+        maximum.number.of.clusters, society, conductance, stabilizer = 1)
+    eigen.values.1000 <- clust_result.1000@eigen.space$values
+  } else data("eigen.values.1000")
> plot(eigen.values.1000[1:50])
```
We observe that in the eigen values plot, "too many" values are close to 1 but for this example we do not expect 20 populations. So we decrease sigma:

```r
> normal.sigma <- 250
> conductance <- Conductance_Calculation(full, normal.sigma, space.length, +    society, precision)
> clust_result.250 <- Civilized_Spectral_Clustering(
    +    full,
    +    maximum.number.of.clusters,
    +    society, conductance, stabilizer = 1)
> eigen.values.250 <- clust_result.250@eigen.space$values
> plot(eigen.values.250[1:50])
```
This is "a right" value for normal.sigma because the curve has now a knee shape. Even some variation to this parameter does not change the shape significantly (200 or 300 can be tried).

Now having sigma been adjusted, separation factor can be tuned:

```r
> labels.for_num.of.clusters <- clust_result.250@labels.for_num.of.clusters
> number.of.clusters <- clust_result.250@number.of.clusters
> L33 <- labels.for_num.of.clusters[[number.of.clusters]]
> separation.factor <- 0.1
> component.of <- Connecting(full, society, conductance, number.of.clusters,
+   labels.for_num.of.clusters, separation.factor)$label
> plot(full, pch = ".", col = component.of)
```
This value is too small for the separation factor and a population is combined by mistake. Therefore, we increase septation factor to separate the components more:

```r
> separation.factor <- 0.5
> component.of <- Connecting(full, society, conductance, number.of.clusters, + labels.for_num.of.clusters, separation.factor)$label
> plot(full, pch = ".", col = component.of)
```
This is the right value for separator factor as all population are now separated.

Now, we can fix these values for the parameters; normal.sigma=250 and separation.factor=0.5. One can run the SamSPECTRAL algorithm on the rest of the data set without changing them, hopefully, obtaining as appropriate results.
Appendix I: Additional Information on SamSPECTRAL

Report on identification of rare population.

Table [A.3] contains the full detailed report on my comparative experiment for identifying rare populations.

High dimensional flow cytometry data.

I prepared a data file containing a matrix with 100,000 rows and 23 columns that represents a flow cytometry sample with 100,000 events. It is freely available in [179] as an additional file (#2). This data file can be directly loaded in R and analysed by SamSPECTRAL. It takes less than 12 minutes to perform faithful sampling on this 23 dimensional data.

Parameters for GvHD dataset.

These values are appropriate for running SamSPECTRAL on GvHD dataset. See Appendix H for definition of each parameter.

dimensions = c(7,5,3,4)
space.length = 400
community.weakness.threshold = 1
normal.sigma = 200
seperation.factor = 0.35
determine.number.of.clusters = “Automatically”
maximum.number.of.clusters = 30
m = 3000
precision = 6
Parameters for stem cell dataset.

These values are appropriate for running SamSPECTRAL on stem cell dataset. See Appendix H for definition of each parameter.

\begin{verbatim}
iterations = 200

dimensions <- c(3,4,7)
space.length = 3
community.weakness.threshold = 1
normal.sigma = 200
seperation.factor = 0.39
determine.number.of.clusters = "Automatically"
maximum.number.of.clusters = 30
m = 3000
precision = 6
iterations = 200
\end{verbatim}

Parameters for telomere dataset.

These values are appropriate for running SamSPECTRAL on telomere dataset. See Appendix H for definition of each parameter.

\begin{verbatim}
iterations = 200

dimensions <- c(1,2,4)
space.length = 400
community.weakness.threshold = 1
normal.sigma = 80
seperation.factor = 0.39
determine.number.of.clusters = "Automatically"
maximum.number.of.clusters = 20
m = 3000
precision = 6
iterations = 200
\end{verbatim}

Parameters for viability dataset.

These values are appropriate for running SamSPECTRAL on viability dataset. See Appendix H for definition of each parameter.
Simulation with synthetic data.

The following R source code produces synthetic data with 5 clusters shown in Figure 2.5. The resulting data is passed to SamSPECTRAL to be clustered.

Preparing two dimensional gaussians:

```r
dimensions <- c(4,5,3)
space.length = 70
community.weakness.threshold = 1
normal.sigma = 50
seperation.factor = 0.5
determine.number.of.clusters = "Automatically"
maximum.number.of.clusters = 30
m = 3000
precision = 6
iterations = 200

gaussian.big <- mvrnorm(n=30000, mu=c(5,5), Sigma=matrix(c(2,0,0,2),2,2))
gaussian.small.1 <- mvrnorm(n=300, mu=c(9,10), Sigma=matrix(c(0.08,0,0,0.3),2,2))
gaussian.small.2 <- mvrnorm(n=300, mu=c(1,10), Sigma=matrix(c(0.07,0,0.08),2,2))
gaussian.small.3 <- mvrnorm(n=300, mu=c(2,0), Sigma=matrix(c(0.5,0,0,1),2,2))
gaussian.small.4 <- mvrnorm(n=300, mu=c(10,1), Sigma=matrix(c(0.1,0,0.7),2,2))
gaussian.all <- rbind(gaussian.big, gaussian.small.1, gaussian.small.2, gaussian.small.3, gaussian.small.4)
xlim=c(min(gaussian.all[,1]), max(gaussian.all[,1]))
ylim=c(min(gaussian.all[,2]), max(gaussian.all[,2]))

Uniform background noise:

uniform.num <- 4000
x.uniform <- (runif(uniform.num)*(xlim[2]-xlim[1]))+xlim[1]
y.uniform <- (runif(uniform.num)*(ylim[2]-ylim[1]))+ylim[1]
uniform.noise <- cbind(x.uniform, y.uniform)

Combining all gaussians and the background noise:

synthetic.data <- rbind(gaussian.all, uniform.noise)
```

162
Table A.3: Identification of rare cell population in stem cell dataset using different algorithms

<table>
<thead>
<tr>
<th>Sample Index</th>
<th>Number of Rare Cells</th>
<th>Percentage</th>
<th>SamSPECTRAL Sensitivity</th>
<th>SamSPECTRAL Specificity</th>
<th>flowMerge Sensitivity</th>
<th>flowMerge Specificity</th>
<th>FLAME Sensitivity</th>
<th>FLAME Specificity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>23</td>
<td>0.24</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>2</td>
<td>85</td>
<td>0.88</td>
<td>0.99</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>0.31</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>0.12</td>
<td>NF</td>
<td>NF</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>101</td>
<td>1.03</td>
<td>0.99</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>22</td>
<td>0.22</td>
<td>0.67</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>0.08</td>
<td>0.67</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>8</td>
<td>21</td>
<td>0.29</td>
<td>0.47</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>15</td>
<td>0.19</td>
<td>0.8</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>10</td>
<td>36</td>
<td>0.37</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>11</td>
<td>134</td>
<td>1.37</td>
<td>1</td>
<td>1</td>
<td>0.99</td>
<td>1</td>
<td>0.99</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>29</td>
<td>0.29</td>
<td>0.66</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>13</td>
<td>10</td>
<td>0.1</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>14</td>
<td>85</td>
<td>0.80</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>19</td>
<td>0.19</td>
<td>0.39</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>16</td>
<td>6</td>
<td>0.06</td>
<td>0.16</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>17</td>
<td>20</td>
<td>0.2</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>0.12</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>19</td>
<td>18</td>
<td>0.18</td>
<td>0.35</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>20</td>
<td>94</td>
<td>0.94</td>
<td>0.92</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.98</td>
<td>1</td>
</tr>
<tr>
<td>21</td>
<td>22</td>
<td>0.22</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>22</td>
<td>3</td>
<td>0.03</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>23</td>
<td>31</td>
<td>0.32</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>0.96</td>
<td>1</td>
</tr>
<tr>
<td>24</td>
<td>8</td>
<td>0.08</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>25</td>
<td>12</td>
<td>0.12</td>
<td>0.34</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>26</td>
<td>151</td>
<td>1.49</td>
<td>0.93</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.99</td>
<td>1</td>
</tr>
<tr>
<td>27</td>
<td>21</td>
<td>0.21</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>28</td>
<td>45</td>
<td>0.46</td>
<td>0.97</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>29</td>
<td>140</td>
<td>1.43</td>
<td>1</td>
<td>0.99</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>30</td>
<td>46</td>
<td>0.47</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>31</td>
<td>3</td>
<td>0.03</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>32</td>
<td>87</td>
<td>0.9</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>33</td>
<td>17</td>
<td>0.17</td>
<td>1</td>
<td>1</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
<tr>
<td>34</td>
<td>13</td>
<td>0.13</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
<td>NF</td>
</tr>
</tbody>
</table>
Simulation with synthetic data.

Specificity (Table A.3)

- *SamSPECTRAL*: Min=0.99, Max=1, Mean=1, Median=1, 1stQu=1, 3rdQu=1, std=0.
- *flowMerge*: Min=1, Max=1, Mean=1, Median=1, 1stQu=1, 3rdQu=1, std=0.
- *FLAME*: Min=1, Max=1, Mean=1, Median=1, 1stQu=1, 3rdQu=1, std=0.

Sensitivity (Table A.3)

- *SamSPECTRAL*: Min=0.16, Max=1, Mean=0.83, Median=0.99, 1stQu=0.67, 3rdQu=1, std=0.26.
- *flowMerge*: Min=0.99, Max=1, Mean=1, Median=1, 1stQu=1, 3rdQu=1, std=0.
- *FLAME*: Min=0.96, Max=1, Mean=0.99, Median=1, 1stQu=0.99, 3rdQu=1, std=0.01.