Adventures in computational immunology: a novel approach to analyse high dimensional flow cytometry data

Ross J Burton[1], Simone Cuff[1], Peter Ghazal[2], Matthew Morgan[1,3], Andreas Artemiou[4], and Matthias Eberl[1,2]

1. Division of Infection and Immunity, School of Medicine, Cardiff University Heath Park, Cardiff, CF14 4XN
2. Systems Immunity Research Institute, School of Medicine, Cardiff University Heath Park, Cardiff, CF14 4XN
3. Cardiff & Vale University Health Board, Heath Park, Cardiff, CF14, 4XN
4. School of Mathematics, Cardiff University, Cardiff, CF24 4AG

Introduction

Flow cytometry is fundamental for the investigation of immunological states in disease, permitting the generation of vast quantities of single cell data. It is often the case that investigators will compare an array of immunological markers and contrast clinical or experimental endpoints. Statistically significant correlations in this setting point towards biomarkers that might eventually have clinical application. The number of biomarkers we investigate in any single study is limited by the technical capabilities of the cytometry instruments and the analytical abilities of the investigator. The former is rapidly changing to allow larger staining panels with spectral and mass cytometry promising up to 30/40 markers per experiment. If the ambition of high dimensional single cell analysis by cytometry is to be realised then the way in which this data is managed and analysed must change. Traditional manual analysis of flow cytometry output is laborious, subjective, and often not reproducible. Although in the past decade there has been many efforts to address this issue, few have resulted in practical application; many of the methods proposed require extensive programming knowledge, do not provide the rigorous data management needed for large clinical trials, and are not accessible to the wider immunology community. Here we describe a pipeline in development to address the issues of flow cytometry data analysis.

Methods: developing Immunova

Immunova is an analytical pipeline and programming library developed in the Python programming language (version 3.7). Although historically flow cytometry bioinformatics has been largely confined to the R programming ecosystem, we chose to develop our solution in Python to help increase accessibility; Python is designed for ‘readable and beginner friendly code’. Figure 1 shows the analytical steps in Immunova. Prior to analysis a spillover matrix for compensation must be prepared externally. At the heart of Immunova is a document-based database; data is stored in JSON format as opposed to tabular. This design choice provides flexibility in our analysis, as the investigator can easily include additional data post-hoc (e.g. RNAseq, ELISA, meta).

There has been a divergence in flow cytometry analysis in recent years with some investigators opting for high-dimensional clustering as opposed to traditional ‘gating’. We recognise the advantage of this approach but appreciate that traditional gating is easier to interpret for immunologists. To circumvent the laborious and subjective nature of ‘gating’ we provide autonomous gating, performed prior to high-dimensional analysis. Gating and clustering help engineer variables that can then be compared to identify biomarkers that correlate with a clinical or experiment end-point. This is achieved by feature selection in a supervised framework; variables of low variance are removed and what remains is ranked based on their contribution to a predictive model.

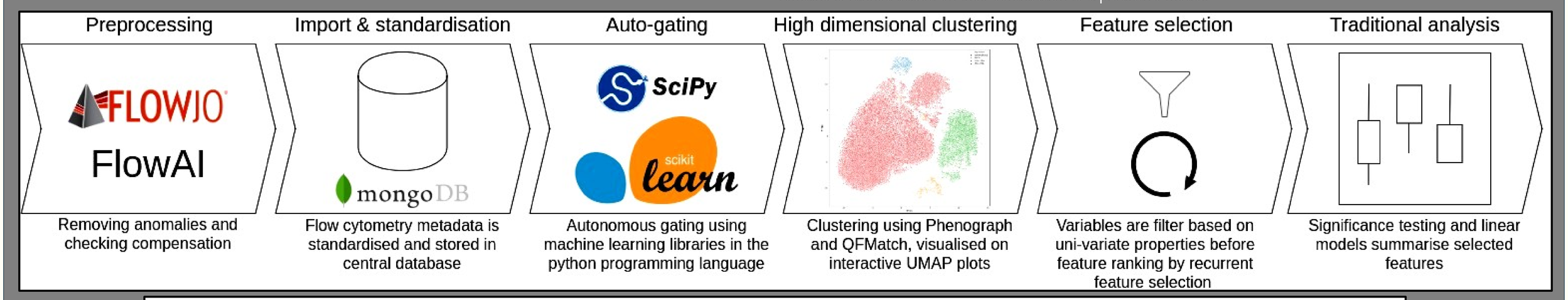

Figure 1. Overview of the Immunova analytical pipeline. All stages following ‘preprocessing’ are housed within the Immunova programming library

Machine learning replicates manual gating and is ‘data-driven’

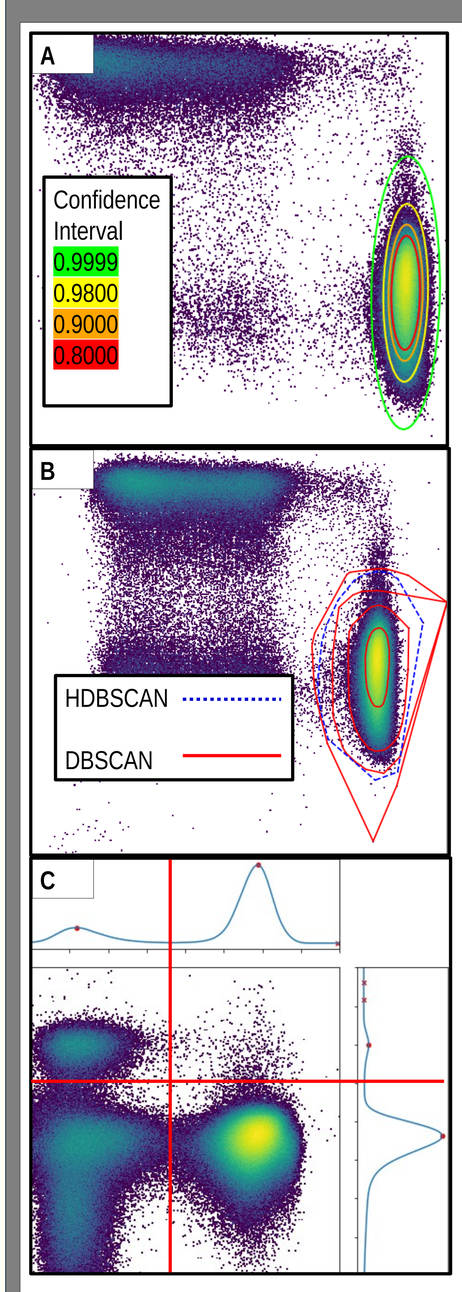

Figure 2. Overview of the four algorithms provided by Immunova for automated gating

Autonomous gating is provided through the application of four algorithms shown in Figure 2. The user designs a ‘gating template’ using a reference sample, choosing the most appropriate algorithm for the population of interest. This template can then be applied to further samples. In each case, the algorithm of choice is “learning” the landscape of the data in an unsupervised manner. Resulting gates therefore fluctuate according to the data without the need for human intervention.

Figure 2A shows the Gaussian Mixture Model algorithm. Suited to well defined populations, this is a probabilistic approach that assumes the underlying data is derived from a finite number of gaussian distributions. This algorithm requires three hyper-parameters when defining the template: the number of expected populations in the biaxial plot, the estimated centroid for the target population, and the confidence interval (CI). Amongst the populations detected, that closest to the estimated centroid will be chosen. The CI helps specify the “tightness” of the resulting gate. This is facilitated by the probabilistic nature of the underlying method and varying CI is shown in Figure 2A.

The DBSCAN and HDBSCAN algorithms identify areas of high density amongst regions of low density. They are advantageous as they do not assume convex clusters and are capable of identifying ‘noise’ (low density regions). The DBSCAN algorithm requires that the user provides the estimated centroid of the target population(s), the minimum number of events that pertain to a population, and the estimated distance between a core member of the population and it’s neighbour. These hyper-parameters, especially the neighbouring distance, can be difficult to estimate and DBSCAN is sensitive to subtle changes as shown in Figure 2B; both DBSCAN and HDBSCAN are provided with a minimum population size of 1000 but DBSCAN, in red, has varying neighbouring distance (a difference of 0.02 between each red line). Due to the limitations of DBSCAN, we also provide access to HDBSCAN, a variation of this algorithm that does not require neighbouring distance to be user defined.

The final algorithm (Figure 2C) identifies thresholds of separation that segment populations using properties of the kernel density estimate in 1-dimensional space. In Figure 2C the KDE of both axis is shown and the red points on each density plot show peaks identified using the SciPy peak finding algorithm. Peaks that are considerably smaller than the maxima are ignored (red crosses). The point of minimum density between the two highest peaks (local minima) is then taken as the threshold for that axis. The performance of the four discussed algorithms are shown in Figure 3.

Autonomous gating matches the performance of a human expert

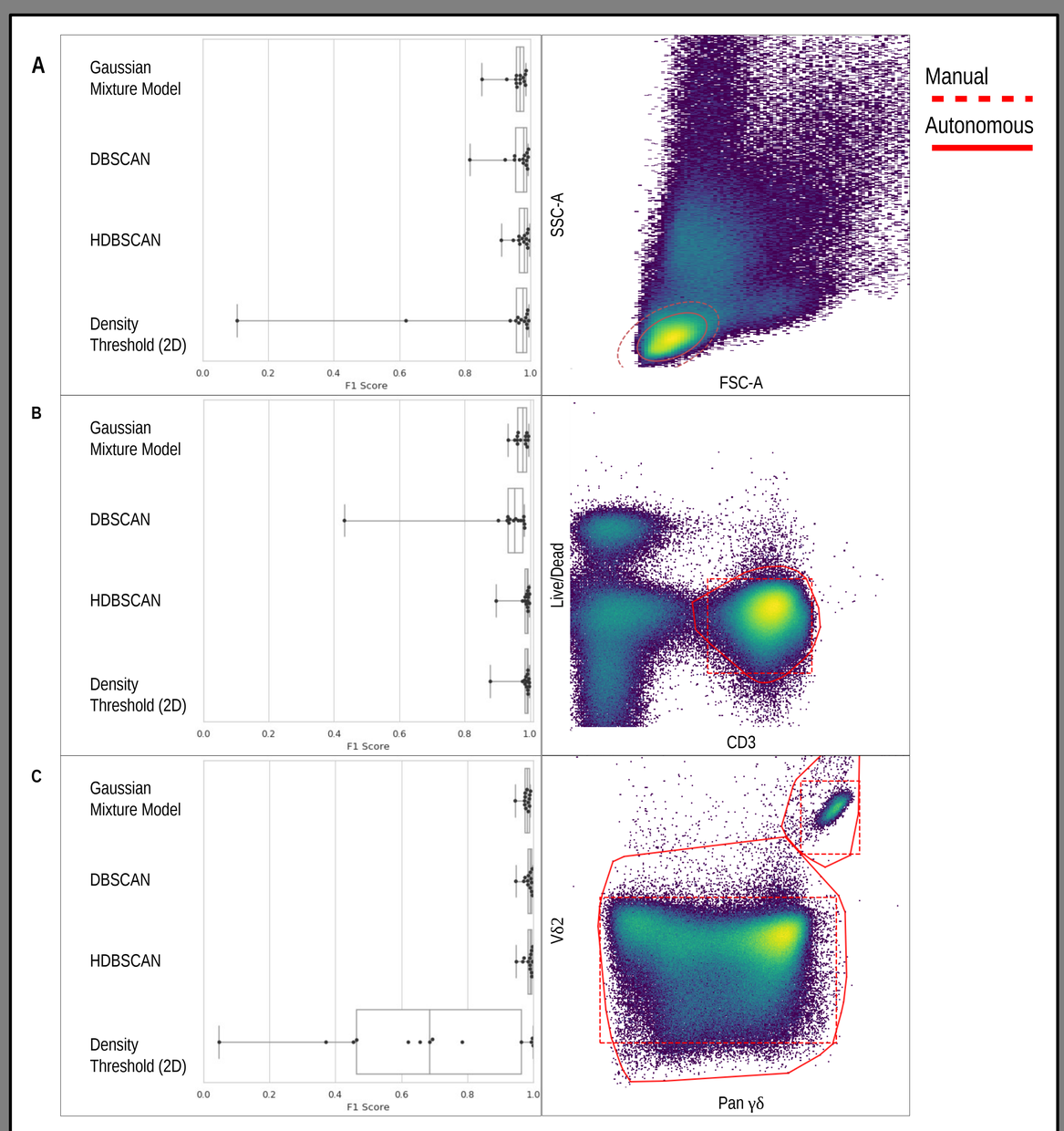

Figure 3. Comparison of autonomous gating algorithms to manual gating of lymphocytes (A), live CD3⁺ cells (B), and γδ T cells (C). Box plots show performance by F1 score across 14 separate flow cytometry experiments. Plots in the right-hand column show overlay of the best performing algorithm compared to manual gating in the same sample.

Conclusions & Future work

Immunova is currently nearing the end of the development stage and will shortly be applied to in-house datasets. One such dataset describes the immunological profile of peritoneal dialysis patients with acute peritonitis and has previously been manually analysed. We hypothesise that autonomous gating will reduce inter-sample variation when compared to manual gating. High dimensional clustering will then be used secondary to autonomous gating to identify populations that a traditional approach might fail to identify. These techniques will be contrasted in UMAP plots. The resulting identified variables will be subjected to feature selection and compared to previous findings for validation.

Acknowledgments

Poster template by ResearchPosters.co.za

Contact Information

Corresponding author’s Name

Address

Tel: +1 066 - 666666
Fax: +1 066 - 777777
Email: author@address.org
Web: www.yourwebsite.org