CytoPy: an autonomous cytometry analysis framework

Ross J. Burton[1], Raya Ahmed[1], Simone M. Cuff[1], Andreas Artemiou[2], Matthias Eberl[1,3]

[1]Division of Infection and Immunity, School of Medicine, Cardiff University, Cardiff, United Kingdom; [2] School of Mathematics, Cardiff University, Cardiff, Wales, UK; [3]Systems Immunity Research Institute, Cardiff University, Cardiff, United Kingdom

Correspondence to: Ross Burton, Division of Infection and Immunity, Henry Wellcome Building, School of Medicine, Cardiff University, Heath Park, Cardiff CF14 4XN, Wales, United Kingdom; E-mail: burtonrj@cardiff.ac.uk

Key words

Cytometry bioinformatics, high-dimensional clustering, supervised machine learning, immunophenotyping, Python, MongoDB

Abstract

Cytometry analysis has seen a considerable expansion in in recent years with the expansion in the maximum number of parameters that can be acquired in a single experiment. In response to this technological advance, there has been an increased effort to develop computational methodologies for handling high-dimensional data acquired by flow or mass cytometry. Despite the success of numerous algorithms and published packages to replicate and outperform traditional manual analysis, widespread adoption of these techniques has yet to be realised in the field of cytometry. Here we present CytoPy, a Python framework for automated analysis of high dimensional cytometry data that integrates a document-based database for a data-centric and iterative analytical environment. The capability of supervised classification algorithms in CytoPy to identify cell subsets was successfully confirmed by using the FlowCAP-I competition data. The applicability of the complete analytical pipeline to real world datasets was validated by immunophenotyping the local inflammatory infiltrate in individuals with and without acute bacterial infection. CytoPy is open-source and licensed under the MIT license. Source code is available online at the https://github.com/burtonrj/CytoPy, and software documentation can be found at https://cytopy.readthedocs.io/.

1. Introduction

Cytometry data analysis has undergone a paradigm shift in response to the growing number of parameters that can be observed in any one experiment. As the field evolves, the traditional method of manual gating by sub-setting single cell data into populations and encircling data points in hand-drawn polygons in two-dimensional space has proven laborious, subjective, and difficult to standardise. In response to these shortcomings, a cross-disciplinary effort has given birth to a new approach often termed ‘cytometry bioinformatics’, to leverage complex computer algorithms and machine learning to automate analysis and improve the investigator’s ability to extract meaning from high dimensional data.

Where cytometry is used for data acquisition, the typical objective is to discern differences between groups of subjects or experimental conditions, or to identify a phenotype that correlates with an experimental or clinical endpoint. To this end, a computational approach to analysis of cytometry data can take one of two strategies: to separate single cell data into groups or classifications, which then form the variables (often descriptive statistics of the obtained groups) the investigator uses to test their hypothesis, or directly model the acquired distribution of single cell data with respect to a chosen endpoint. Classification strategies can be further subdivided: autonomous gating replicates traditional gating through the use of algorithms to cluster data in two-dimensional space (flowDensity (1), OpenCyto (2)); high-dimensional clustering groups cells according to their individual phenotypes (FlowSOM (3), PhenoGraph (4), Xshift (5), SPADE (6)); and supervised classification where training on an example of manually gated data produces a classifier capable of distinguishing cell populations (FlowLearn (7) and DeepCyTof (8)). Modelling strategies have been successfully adopted in applications such as ACCENSE (9), CellCNN (10), and CytoDX (11) although this approach requires pooling of sample data and is therefore sensitive to batch effects.

In addition, various pieces of software have been developed for data handling, transformation, normalisation and cleaning (*e.g.* flowCore, flowIO, flowUtils, flowTrans, reFlow, flowAI), visualisation (*e.g.* ggCyto, t-SNE, UMAP, PHATE), and pipelines for specific applications (*e.g.* Citrus, MetaCyto, flowType/RchyOptimyx). To date, there are over 30 different contributions to automated analysis (12; 13; 14; 15). However, there is no widespread adoption of these methods as yet, nor is there a consensus on how to adopt such techniques, with much of the analysis pipeline left to the individual investigator to establish. This inconsistency results in projects amassing collections of custom scripts and data management that are not standardised or centralised, which not only makes reproducing results difficult but also makes for a daunting landscape for newcomers to the field.

We here introduce 'CytoPy', a novel analysis framework that aims to mend these issues whilst granting access to state-of-the-art machine learning algorithms and techniques widely adopted in cytometry bioinformatics. CytoPy is developed and maintained in the Python programming language, which prides itself on readability and is becoming the language of choice amongst the open source data science community (16). CytoPy introduces a central data source for all single cell data, clinical/experimental metadata and analysis results, and provides a 'low code' interface that is both powerful and beginner friendly.

We demonstrate the capability of supervised classification techniques housed within CytoPy on the Flow Cytometry: Critical Assessment of Population Identification Methods (FlowCAP) data (17), which has been created for comparing the performance of automated analytical techniques for flow cytometry data. As the FlowCAP data underwent extensive pre-processing prior to their publication and hence do not reflect the challenges encountered with primary data generated by individual users, an in-house dataset of local immune cells in samples collected from patients undergoing peritoneal dialysis and who presented with and without acute bacterial infection was generated to demonstrate the applicability of CytoPy as a complete analytical pipeline for complex and unprocessed data. These data reflect the challenges presented by an observational study of complex clinical specimens collected over extended periods of time, often years. We believe that CytoPy provides a powerful and user-friendly framework to interrogate high dimensional data originating from investigations that utilise flow cytometry or mass cytometry for data acquisition, and has the potential to facilitate automated data analysis in a multitude of experimental and clinical contexts.

2. Design and Implementation

2.1. Building a framework that is data-centric

Reliable data management is a cornerstone of successful analysis, by improving reproducibility and collaboration. A typical cytometry project consists of many Flow Cytometry Standard (FCS) files, clinical or experimental metadata, and additional information generated throughout the analysis (*e.g.* gating, clustering results, cell classification, sample specific metadata). A further complication is that any analysis is not static but an iterative process. We therefore deemed it necessary to anchor a robust database at the centre of our software. In CytoPy, projects are instantiated and housed within this database, which serves as a single dynamic data repository that is then accessed continuously throughout the subsequent analysis. For the architecture of this database we chose a document-orientated database, MongoDB (18), where data are stored in JSON-like documents in a tree structure. Document-based databases carry many advantages, including simplified design, dynamic structure (*i.e.* database fields are not ’fixed’ and therefore resistant to unforeseen future requirements) and easy to scale horizontally, thereby improving integration into web applications and collaboration. In this respect, CytoPy depends upon MongoDB being deployed either locally or via a cloud service, and MongoEngine, a Document-Object Mapper based on the PyMongo driver (19).

2.2. Framework overview

An overview of the CytoPy framework is given in Figure 1 including a recommended pathway for analysis, although individual elements of CytoPy can be used independently. CytoPy follows an object-orientated design with a document-object mapper for both commitment to, and collection from, the underlying database. The user interacts with the database using an interface of several CytoPy classes, each designed for one or more tasks. CytoPy is algorithm agnostic, meaning new autonomous gating, supervised classification, clustering or dimensionality reduction algorithms can be introduced to this infrastructure and applied to cytometric data using one of the appropriate classes. CytoPy makes extensive use of the Scikit-learn (20) and SciPy (21) ecosystems and follows the naming conventions commonly used in this space. Throughout an analysis, whenever single cell data are retrieved from the database, they are stored in memory as Pandas DataFrames that are accessible for custom scripting at any stage.

Following the steps in Figure 1, a typical analysis in CytoPy would be performed as follows (functions are shown in italics and class names are shown in italics and title-case).

(1) Single cell data are generated and exported from the flow/mass cytometer in FCS 2.0 or 3.0 format. Experimental and clinical metadata are collected in tabular format either as Microsoft Excel document or CSV file, with the only requirement being that metadata be in ‘tidy’ format (22).

(2) A *Project* is defined and populated with the single cell data and accompanying metadata. Each subject (*e.g.* a patient, a cell line, or an animal) has a *Subject* document containing metadata that are dynamic and have no restriction on the data stored within, and that are associated to one or several *FileGroup* documents. Each *FileGroup* document is representative of one or more FCS files associated to a single biological sample collected from the subject. The single cell data from the multiple files in a *FileGroup* are saved to disk as a Hierarchical Data Format (HDF) file. The *FileGroup* is linked to this file on disk but the data can be migrated to any drive of the user’s choosing. The *FileGroup* and its associated HDF file contain all single cell data, gated populations, clusters and meta-information that attains to a single sample. This also includes any isotype or

Fluorescence-Minus-One (FMO) controls. Compensation is applied to single cell data at the point of entry using either an embedded spillover matrix or a provided CSV file. The *FileGroup* is associated to an *Experiment*, containing all samples collected under one particular set of staining conditions. There must always be a *Panel* document associated to an *Experiment*. For this, the investigator must provide a 'panel design' in the form of a simple Excel document (see CytoPy documentation https://cytopy.readthedocs.io/). CytoPy then uses regular expression to match FCS metadata such as channel names to the expected panel and offers error handling for when discrepancies arise.

(3) The common approach to cytometry data analysis is expert driven gating in sequential two-dimensional space. Attempts to automate gating in the past have focused on the application of a single methodology (1; 23; 24) or an assortment of selected algorithms (2). CytoPy attempts to expand on the later approach with an open design, allowing the use of any unsupervised learning algorithm supported in Scikit-Learn or libraries that follow the Scikit-Learn template. Algorithms that generate a label for each data-point can be handled using the *PolygonGate* object, converting the output into polygon gates by calculating the convex envelope of each cluster. Probabilistic models such as Gaussian Mixture Models are handled by the *EllipseGate* object, where elliptical gates are generated by drawing a confidence interval around the models components. Finally, we have designed a density based algorithm implemented in the *ThresholdGate* object, separating data in one or two-dimensional space based on properties of the estimated probability density function. In a traditional analysis, gates are applied in sequence to derive the cell population of interest. CytoPy replicates this with autonomous gates using the *GatingStrategy* class. Using this class, autonomous gates are defined once and then applied to subsequent samples. In each instance, algorithms will 'fit' the new data they encounter (generating a data-driven strategy to gating) and then locate the expected populations, annotating accordingly.

(4) The variation that separates biological specimens can originate from genuine biological differences but can also arise from technical variation introduced by pipetting errors, changes in

instruments, variation in experimental practice, or other uncontrollable experimental conditions. CytoPy offers the 'variance' module, containing utilities for visualising the univariant and multivariant deviations between biological specimens. If the data and study design allow, the user can consider pooling data and modelling the distribution of single cell data directly. In some circumstances the user will have to consider the contribution of batch effects, and in such a case the investigator can use the *SimilarityMatrix* class in CytoPy to group subjects according to their statistical distance from one another. Representative data from each group can be used as training data for a supervised classification algorithm used to annotate the entire dataset without the need for manual gating or complex autonomous gating strategies.

(5) Multiple strategies can be employed to classify cells based on a common phenotype. Strategies such as autonomous gating and supervised classification are biased by the training data provided (and the gating strategy used to label those data) whereas high-dimensional clustering is an unsupervised method that groups cell populations according to their phenotype but can be difficult to critique. CytoPy offers both supervised classification through the *CellClassifier* class and high dimensional clustering through the *Clustering* class, so that variables can be generated from either or both strategies. Importantly, the results of either strategy can be committed to the database and then visually interpreted using a class called *Explorer*. The *Explorer* class also facilitates exploratory data analysis with interactive plots of embedded space using multiple dimensionality reduction techniques.

(6) Once cells have been classified, the user can test their hypothesis. The single cell data are summarised into a 'feature space', summary statistics that describe the cell populations. This generates a large number of variables, many of which will be either uninformative or redundant. Filter and wrapper methods are applied to perform feature selection, finding only those variables important for predicting a clinical/experimental endpoint. In addition, there are multiple methods available for visualising extracted features, thereby allowing the investigator to quickly determine whether certain patterns exist in the dataset.

3. Results

3.1. Autonomous gating can standardise the cleaning of single cell data for rapid analysis

To validate CytoPy we demonstrate its use on the characterisation of immune cells in peritoneal drain fluid and whole blood of peritoneal dialysis (PD) patients with and without acute bacterial infection. We chose this dataset based on a wealth of previous experience in the field (25; 26; 27), the clinical relevance of acute peritonitis in those patients (28), and because of the technical challenges presented by the sample type. Samples were collected between 2017 and 2019 and stained with a comprehensive panel of monoclonal antibodies to identify T lymphocytes, monocytes, dendritic cells, eosinophils and neutrophils as the major constituents of peritoneal immune cells, together with activation and differentiation markers on those populations (Supplementary Tables S2 and S3).

Cytometry data are highly variable and surface marker expression must often be identified amongst a backdrop of cellular debris and staining artefacts. This is particularly relevant when studying complex samples such as local specimen taken from the site of acute infection. In the case of individuals receiving PD, bacterial infection leads to the influx of billions of inflammatory cells, predominantly neutrophils, into the peritoneal cavity within a few hours (27). Traditionally the variability this introduces into analysis is handled by laborious and subjective manual gating. We generated a computational framework that grants access to numerous algorithms for the purpose of data-driven autonomous gating. The user can design a sequence of autonomous gates, save this sequence to a *GatingStrategy*, and then apply the sequence to subsequent data. This is exemplified in Figure 2, showing the identification of T lymphocytes from local immune cells in the peritoneal effluent of PD patients. This example utilises density-driven threshold gates, where a threshold is determined based on properties of the Probability Density Function as estimated using Gaussian Kernel Density Estimation, and mixture models shown as elliptical gates. CytoPy hides the daunting complexity of this process behind a low-code interface (Supplementary Figure S1), in an attempt to make cytometry bioinformatics more approachable for newcomers to the field. Many popular algorithms are accessible through this interface, including Birch, mini-batch K means and mixture models (Supplementary Figure S2).

Autonomous gates are capable of replicating manual gates but are often dependent on the choice of hyperparameters and their optimal values may differ between data. Despite this, we have found their application is suitable for identification of large populations requiring simple gating strategies. Figure 3 shows the performance of autonomous gates for identifying common cell populations in peritoneal effluent by comparing their correlation with the same population identified manually by an expert analyst. It is evident that autonomous gates fail when a population is rare and/or displays poor separation from cells with similar phenotypes. The same observations were made when gating for T cell subsets, with good performance for CD4+ and CD8+ T cells, but poor performance for minor populations such as mucosal-associated invariant T (MAIT) cells and γδ T cells (data not shown).

3.2. CytoPy provides accurate cell classification using supervised machine learning algorithms

Unsupervised methods employed by autonomous gates may fail to generalise and struggle to reliably annotate rare cell populations, populations that significantly deviate between biological specimens, or populations that may be almost absent in some individuals yet abundant in others. Further to this, autonomous gates are only exposed to one or two dimensions of the n-dimensional feature space (that is, the vector of intensity values for all measured fluorochromes) rather than exploiting all available variables. The nature of cytometry data lends itself well to supervised classification, given that a typical biological sample yields hundreds of thousands of events but current technologies are limited to measuring up to a maximum of approximately 40 variables for each cell, resulting in an abundance of observations. We therefore hypothesised that a supervised classifier, trained on one or more annotated examples and then exposed to all available variables in the data, would result in increased performance. After consulting the literature we found that others had employed such techniques (8; 7) but no robust framework for their application exists.

CytoPy offers the *CellClassifier* class as a blueprint for supervised classification in a cytometry framework. Through this class CytoPy exposes the popular machine learning libraries Scikit-Learn (20), XGBoost (29) and Keras (30) to the task of annotating cytometry data. The *CellClassifier* class follows the conventions of Scikit-Learn by providing a familiar application programming interface (API) and the apparatus for any classification algorithm to be integrated into the CytoPy framework. In this study, we have chosen to demonstrate the following: XGBoost, a Feed-Forward Neural Network, Linear Discriminant Analysis, Support Vector Machines and K-Nearest Neighbours. The choice of algorithms to include in this analysis was based on prior experience with classification tasks (31), examples in the literature of supervised classifiers in this domain (8; 17; 23), and the relevance of including classifiers from multiple families (32). In order to validate the *CellClassifier* class and test the performance of each algorithm, we utilised the FlowCAP-I classification challenge (see Supplementary Methods). These data were chosen because of prior publication and their use for validation of algorithms applied to cytometry analysis (17; 8). As shown in Table 1, when judged by weighted F1 score we found acceptable performance for each algorithm applied (defined as an F1 score greater than 0.9) while XGBoost gave the best performance, and was therefore deemed the method of choice for the remainder of this study.

3.3. CytoPy provides visual and quantitative tools for evaluating inter-sample variation, assisting the choice of suitable training data

The FlowCAP-I data are suitable for validation of a method but are heavily pre-processed and simplified compared to data encountered in a complex observational study such as the peritonitis data introduced earlier. Studies designed to collect clinical specimens over several months or even years introduce unavoidable complications, for instance including but not limited to: deviations in experimental conditions, changes in instrument setup, and variation introduced by batch changes of staining monoclonal antibodies. The technical variation combined with the biological variation observed for each specimen can make it difficult to determine their comparability and choose representative training data for a supervised learning approach to cytometry data annotation.

CytoPy provides the *variance* module to visualise and quantify this variation, to assist the user in the choice of adequate training material. As an anchoring point, the user should choose a suitable "reference sample" to be used when comparing the observable variation amongst all samples in an experiment. A reference sample can be identified using the *calculate_ref_sample* function. Following the method presented by Li et al (8), CytoPy performs a pairwise computation of the Euclidean norm of each sample's covariance matrix, and selects the sample with the smallest average distance as reference. This reference sample can then be used for univariate comparison of each channel using the *marker_variance* function (Figure 4A) or multivariate comparison using a dimensionality reduction technique such as Principle Component Analysis (PCA), achieved with the *dim_reduction_grid* function (Figure 4B).

In Figure 4, the reference sample is shown in blue and compared to randomly selected samples shown in red; ten such samples are depicted to ease visual interpretation but there is no limit to the number of comparisons that can be made in a single plot. While Figure 4A shows the degree of inter-sample variance for individual fluorochromes and highlights abnormalities in a single channel, Figure 4B shows the same ten randomly selected samples, individually plotted to overlay the reference sample, thus illustrating the multivariate drift of a sample compared to the chosen reference. This allows for identification of samples that are explicit outliers and gives a general sense of the inter-sample variance in the complete immunological landscape measured.

The approach illustrated in Figure 4 defines methods that are helpful for visually critiquing the quality of the dataset and that can identify anomalies that should be addressed by changing technical procedures in data acquisition. To proceed with classifying cells into known phenotypical subsets we must take into account this variation. This is achieved in traditional manual gating by laboriously adjusting gates on a per-sample basis, with considerable variation depending on the investigator. For automated classification by supervised methods, we instead choose our training data in such a way that inter-sample variation is accounted for. CytoPy provides the *SimilarityMatrix* class and the output is shown for each sample type in Figure 5. Unlike the visualisation techniques depicted in Figure 4, the *SimilarityMatrix* quantifies the inter-sample variation by computing a pairwise statistical distance for each possible combination of samples. In brief, the joint probability density function (PDF) of the n-dimensional feature space is estimated using a kernel density estimation (KDE; multiple implementations are available but CytoPy defaults to a fast convolution based technique (33)). Bandwidth for KDE can be given either as a floating point number used for each KDE computation, estimated for each sample using normal approximation *(e.g.* Silverman's method), or the optimal bandwidth for each sample can be estimated by cross-validation. The latter is the preferred and default method, whereby the optimal bandwidth is chosen by grid-search hyperparameter tuning using cross-validaiton; the optimal bandwidth being the one which maximises the total log probability density under the model. If the the number of cells observed are few, the number of dimensions many, or the user lacks computational resources, they can opt to perform dimension reduction prior to the KDE. The statistical distance for each pair of PDFs is calculated to generate a matrix of distances. The statistical distance shown in Figure 5 is the square root of the Jenson-Shannon divergence (the default choice for this function), given by:

$$Jenson\ Shannon\ distance = \sqrt{\frac{KL(p\,||m) + KL(q\,||m)}{2}}$$

Where *m* is the pointwise mean of the left probability vector *p* (PDF of the first sample) and the right probability vector *q* (PDF of the second sample). *KL* is the Kullback-Leibler divergence. The Jenson-Shannon distance returns a value between 0 and 1, where 0 indicates that the distributions *p* and q are equivalent, and 1 that they are highly dissimilar (34; 35). Any statistical distance (a function taking two probability vectors and outputting a metric distance) can be used, but by default the Jenson-Shannon distance is applied, chosen for its properties of symmetry and finite output (35; 36). The *SimilarityMatrix* outputs a heatmap where the colour of each cell corresponds to the Jenson-Shannon distance of the x, y axis pair that overlaps on the given cell. The axes of the heatmap are clustered using single linkage clustering. Clustering on the pairwise Jenson-Shannon distance reveals groups of samples that are similar in the distribution of their single cell subsets in high dimensional space. Classification of cell populations in these groups can be performed independently per group but with the same objective of identifying phenotypically distinct cell populations. For each group, the investigator chooses training data (a uniform sample of cells from each member of the group) using the *calculate_ref_sample* function, or can choose to sample multiple members of a group with the *create_ref_sample* function to generate a new *FileGroup* containing cells from many specimens. Suitable training data are then annotated for the cell phenotypes of interest (*e.g.* for T lymphocytes this might be CD4$^+$ and CD8$^+$ T cell subsets) using the gating infrastructure discussed in section 3.1. Once annotated a classifier is trained using the labelled reference and subsequently predicts the cell populations for the remaining members of the group. This approach accounts for the inter-sample variation, and therefore improves the classifiers' ability to generalise.

3.4. Supervised classification algorithms can reliably identify cell subsets in complex sample types whilst providing tools to inspect and diagnose anomalies

In Figure 5, biological samples were clustered on pairwise Jenson-Shannon distances to reveal groups of samples of relatively high similarity; clustering results are shown as a dendrogram on the axis of each two-dimensional heatmap matrix. Groups are derived by cutting the dendrogram at a level that was heuristically chosen through visual inspection of the dendrogram. This process was repeated for each sample type and set of staining conditions to generate the groups shown in Figure 6A where each group was treated independently during supervised classification.

Figure 6A shows the performance of XGBoost classification of all leukocyte subsets in peritoneal drain fluid and more detailed subsets of the T cell compartment in peritoneal drain fluid and in PBMCs from whole blood. Performance is given as the weighted F1 score, a metric that captures the harmonic mean between precision and sensitivity, and is weighted by class support (the number of true instances for each label), which provides a value between 0 and 1, where 1 is the best possible score. This metric was captured by monitoring the performance of XGBoost on five randomly chosen validation samples from each classification group of each experimental condition and/or sample type. The validation samples were labelled by manual gating. Performance was best for PBMCs from whole blood where the weighted F1 score on average was above 0.95. Performance was worst for identifying leukocyte subsets in peritoneal effluent, which reflects the complex nature of the sample type and the diversity of cell subsets we intend to describe. The situation for T cell subsets classified from drain fluid was more complicated. For groups 2 and 3 performance was optimal (average weighted F1 score ≥ 0.95) yet for group 1 there was one significant outlier; one validation sample gave a weighted F1 score of 0.6, outside the interquartile range for this group. Of note, CytoPy provides functionality to easily visualise and explore the results of *CellClassifier* objects. For the particular outlier mentioned, Figures 6B and 6C show detailed results of the classification of T cell subsets. Figure 6B is a heatmap representation of a confusion matrix, provided if the user provides a value of *True* to the argument *confusion_matrix*, in the *validation* method of *CellClassifier*. The confusion matrix in Figure 6B shows 'predicted labels' versus the 'true label'; the ground truth being the results of manual gates. The values shown in the confusion matrix were normalised across each row (true label) meaning the values on the diagonal were equivalent to the accuracy for each class. The confusion matrix revealed that although this sample scored poorly in terms of Weighted F1 score, the classification accuracy was greater than 95% for all but two classes: γδ T cells and unclassified cells, *i.e.* those that would not fall into any 'gate'. 52% of cells that had been classed as γδ T cells by the manual gate in this particular sample were instead left unclassified and a large majority of unclassified cells from manual gating were classified into other categories by the XGBoost algorithm. The inclusion of unclassified cells into one or more other subsets was least concerning as it likely reflected the subjective nature of manual gating; the close fit of a gate to its chosen population being one common subjective property of manual gates. The classification of γδ T cells was of greater concern, as this is a T cell subset that is relatively rare in many individuals and hence challenging to assess, yet of significant importance especially in Gram negative infections (37).

The *CellClassifier* of CytoPy converts its classification results to population data and is associated back to the *FileGroup*. This makes comparison of supervised classification to the results of manual gating, semi-autonomous gating or clustering analysis straight-forward. For example, the *back_gating* method of *GatingStrategy* allows the investigator to plot the results of multiple methods on familiar bi-axial plots for comparison. As illustration, Figure 6C shows the interrogation of data likely to represent an outlier in the analysis. Overlaid is the result of the XGBoost classification for Vδ2$^+$ γδ T lymphocytes (red points) and the manual gate for the same subset (yellow line). Vδ2$^+$ γδ T cells were unusually sparse in this particular patient sample, which explains the poor classification performance in this instance. Of note, upon visual inspection the XGBoost algorithm was equally suited at identifying rare cell types compared to manual gating; and classification of γδ T cells was performed correctly by the XGBoost algorithm in all other samples (additional examples shown in Supplementary Figure S3 and S4).

3.5. Unbiased cell classification by high dimensional clustering

Although supervised classification provides us with one methodology for identifying cell subsets, it is biased by the gating strategy used in labelling training data. In recent years, numerous clustering algorithms have been proposed for high-dimensional clustering of single cell data. Two popular solutions are PhenoGraph (4) and FlowSOM (3; 38), both of which are available in CytoPy through the *Clustering* class. As with the *CellClassifier* class, *Clustering* is agnostic to the clustering algorithm of choice. Semi-automated gating, XGBoost classification, and PhenoGraph clustering are comparable in their identification of major cells subsets (Supplementary Figure S5) but using unison of methods (i.e. XGBoost classification and PhenoGraph clusters) provides many benefits and is encouraged in the CytoPy framework; high dimensional clustering offers the opportunity for exploratory data analysis, and obtained clusters can be contrasted with populations identified from supervised classification to improve the confidence of reported results.

Exploratory data analysis in CytoPy is facilitated by the *Explorer* class, which encapsulates the single cell data of one or multiple patients after clustering and supervised classification has been performed, and houses the data within a Pandas DataFrame. Operations can be performed on the DataFrame independently allowing custom scripting, but the *Explorer* class carries many utility functions that are designed for exploratory data analysis. Examples include methods for associating metadata to clusters (*e.g.* the patient phenotype), dimensionality reduction techniques, and interactive plotting tools.

Clustering is performed on a per-sample basis but to explore the immune landscape of the entire cohort, a consensus must be found such that similar clusters between patients can be grouped. This consensus gives rise to comparisons in cell abundance and phenotype between clinical phenotypes. To achieve this, CytoPy uses meta-clustering. In brief, each subject is independently min-max normalised, and the centroid of each cluster calculated. The centroids of clusters for each subject are then merged to form a dataframe that describes the clustering results of all subjects. Finally, a clustering algorithm of choice is applied to this dataframe (see Supplementary Methods). As example for the successful utilisation of PhenoGraph, Figure 7A shows the results of meta-clustering for total leukocytes in the peritoneal drain fluid of individuals receiving PD. The Uniform Manifold Approximation and Projection (UMAP) (39) plot shows all clusters (solid filled circles) from all patients displayed in two-dimensional space. The colour of a cluster corresponds to the associated meta-cluster while the size cluster represents the proportion of cells within the cluster (relative to the total CD45$^+$ single immune cells in each individual patient). The nature of the UMAP plot is such that clusters of similar phenotype are arranged closer to one another. However, CytoPy allows to utilise any dimensionality reduction technique (e.g. PCA, Isomap, PHATE (40) etc), depending on the preference of the investigator and the specific question to be addressed. Meta-clusters are manually labelled according to their phenotype, as displayed in the heatmap of Figure 7A. Clusters can be colour-coded using any desired metadata. For instance, given an instance of *Explorer* named *explorer*, one could associate the clinical phenotype of a patient to their clusters using the following single line of code:

```
explorer.load_meta(variable='peritonitis')
```

For each patient in this example, the database is queried for the variable named 'peritonitis' (as in "does this patient have acute peritonitis?") and populates the Pandas DataFrame stored in the *explorer* object. The UMAP plot is then repeated by colour-coding according to the metadata, as shown in Figure 7B. The distribution of clusters of different clinical phenotypes in the UMAP plot reveals changes in the immunological response. Subsets of cell compartments (e.g. 'Monocytes_0', 'Monocytes_1' etc.) can be consolidated and the proportion of cells within these consolidated groups (as percentage of all CD45$^+$ immune cells) is shown in the boxplots of Figure 7B.

Applying this cluster analysis to a cohort of PD patients, CytoPy found that acute bacterial peritonitis resulted in a dramatic shift in the composition of local immune cells, with a significant increase in the proportion of neutrophils and a parallel drop in the relative proportion of monocytes/macrophages, dendritic cells (DCs), B cells and T cells . These findings corresponded well with previous studies showing a significant influx of inflammatory cells into the peritoneal cavity on the first day of presenting with acute symptoms, compared to stable individuals in the absence of peritoneal inflammation (25; 26; 27; 41)

Figure 8 shows the same set of analytical techniques applied to the local T cell populations in individuals receiving PD. Figure 7A shows a UMAP plot of clusters, coloured according to their associated meta-cluster and revealing clean separation not only of CD4$^+$ and CD8$^+$ T cells as the major T cell populations but also of unconventional T cell populations such as Vα7.2$^+$ CD161$^+$ MAIT cells and Vδ2$^+$ γδ T cells. Figure 8B shows the same clusters as in Figure 8A but now colour-coded by the metadata regarding the presence or absence of bacterial infection. The differences in T cell subsets between stable controls and those with acute peritonitis were subtle and, due to the small size of this cohort, not statistically significant. Of note, CytoPy allows to explore the composition of the T cell compartment in even more detail, as illustrated for the CD8$^+$ T cell subset (Figure 8C). Here, PhenoGraph was capable of discerning distinct memory and effector subsets based on the expression of the surface markers CD45RA, CD27 and CCR7 (Figure 8C) further validating CytoPy as a reliable method for exploring changes in immune response in large flow cytometry data.

3.6. Feature extraction and feature selection reveal variables that differentiate the immune response during acute peritonitis compared to stable controls

Following cell classification by both biased and unbiased methodology, the immunological landscape of the observed subjects can be summarised in CytoPy into a 'feature matrix'. This includes the relative abundance of populations as identified by supervised classification and clusters produced by techniques such as PhenoGraph. There will be significant overlap here, and therefore the user may choose to specify to generate a consensus between the results of supervised classification and clustering by way of an average of the two methods. Supervised classification is more robust towards underlying batch effects but biased by the gating strategy imposed upon the training data, whereas clustering is unbiased but not stable to batch effects. By combining both methods the investigator can overcome the limitations that they present individually.

The methods described are implemented in the *feature_extraction* module of CytoPy. Once a feature matrix has been generated dimensionality reduction techniques can be employed to reveal immediately if subjects separate in accordance to the experimental or clinical endpoint of interest. Figure 9A shows a PCA plot where peritonitis patients and stable controls clearly separated across two components, as expected from earlier studies by us (25; 27) and from the analysis shown in Figure 7.

Filtering techniques can be employed within CytoPy to remove variables of low variance or identify high multi-colinearity (Supplementary Figure 2). This is often necessary to remove redundant variables. The immunological pattern that differentiates a clinical state or experimental end-point can then be visualised in a radial plot as shown in Figure 8B. In this example, cell populations are marked on the axis and the internal value is the proportion of cells relative to their respected parent, after consolidating the results of both PhenoGraph clustering and XGBoost classification. Figure 9B confirms the observations made in the exploratory data analysis of clustering results (Figures 7 and 8): although subtle differences exist in the T cell compartments, it is the stark differences in the proportion of myeloid cells that differentiates those with peritonitis compared to stable controls. Where further feature selection is necessary, CytoPy offers embedded methods in the form of L1-regularised linear models, where variables can be selected according to whether their coefficient remains non-zero as the regularisation parameter decreases. (Supplementary Figure S6).

4. Availability and Future Directions

CytoPy represents a framework for the analysis of cytometry data that facilitates automated analysis whilst introducing robust data management and an iterative analytical environment. The present study shows the ability of CytoPy to characterise the FlowCAP-I dataset with high precision and identified XGBoost as optimal classification algorithm for gating with supervised methods. To demonstrate the capabilities of CytoPy on real-world data, we chose to analyse samples from patients with and without acute peritonitis, taking advantage of our extensive experience with this type of samples over more than a decade. Initially acquiring such samples on a four colour BD FACSCalibur flow cytometer with two lasers and simple FSC/SSC settings (42), we later utilised an eight colour BD FACSCanto with three lasers and FSC/SSC area/height channels (24; 34; 35), and now in the present study took advantage of a 16 colour BD LSR Fortessa with four lasers and FSC/SSC area, height, width, and time, thus illustrating the technological advance in the field but also the increasing complexity of the data acquired. The exquisite and elegant performance of CytoPy confirmed a striking increase in total neutrophils at the site of infection and a parallel decrease in the proportion of monocytes/macrophages, dendritic cells and T cells, in agreement with previous findings (26; 27), thereby validating the utility of CytoPy.

We know of no other framework to date that offers the flexibility provided by CytoPy whilst also providing a low-code API for easy application. Existing platforms include the likes of CytoBank (43) which – whilst boasting ease of use and a diverse offering of existing cytometry clustering algorithms – is a proprietary product that would not allow for collaboration and expansion of methodologies through the open-source bioinformatics community. In contrast, CytoPy will remain open-source to encourage bioinformaticians and developers to expand on existing technologies. Alternative open-source frameworks include OpenCyto (2) and Cytofkit (44). OpenCyto is focused on the application of autonomous gates, whereas Cytofkit exposes popular high-dimensional clustering algorithms in an easy interface. Despite their successful application in the literature, neither framework provides the diversity offered in CytoPy, where autonomous gates, supervised classification, and high dimensional clustering can not only be performed in one analytical pipeline, but their outputs visualised and contrasted with one another. This results in increased confidence in our findings as we can assess the conformity of multiple methodologies applied to the same data. Additionally, these existing frameworks do not provide the robust data management that CytoPy offers, with each analytical process being captured in a central database.

We have chosen to develop and maintain CytoPy in Python, a programming language with growing popularity in the bioscience domain. The application of the popular Python deep learning frameworks such as Tensorflow (45) and Keras (30) offer potential for the autonomous analysis of cytometry data (8; 10; 38; 51). It is our intention to incorporate these methodologies in a future release. The agnostic object orientated design of CytoPy facilitates such additional implementations in a straight-forward manner. It is this agnostic design and the introduction of a document-based database as central repository for cytometry analysis that sets CytoPy apart from alternative solutions.

In addition to providing a new data-centric framework for applying existing methods of single cell classification and clustering, CytoPy offers novel tools to aid the analytical pipeline. In this study, we highlight the difficulties presented in complex cytometry data and demonstrate autonomous methods that improve the efficiency of pre-processing. We show how CytoPy can visualise and quantify the inter-sample variation, helping to account for batch-effects. Prior attempts to mitigate or remove batch-effects have either been tied to the application of gates in two-dimensional space (46; 47), involve manipulation of the input space in such a way that biological signals could be lost or distorted (48; 49), or requires some technical intervention during data acquisition (50). Here, we introduce an alternative strategy, instead of removing batch-effect by transforming or aligning the data, we propose a statistical measure be used to group data and supervised classification performed on each group individually. However, we appreciate the impact that a reliable method for mitigating or removing batch effect prior to analysis might have and are open to the integration of data normalisation or transformation methodologies that would achieve this and would see that it fits the data-centric design of CytoPy.

As high-dimensional cytometry analysis continues to grow in popularity there will be increasing demand for an analytical framework that is friendly for those who are new to programming, provides a database that directly relates metadata to single cell data, and scales in a fashion that encourages collaboration and expansion. CytoPy meets all these criteria whilst remaining open-source and freely available on GitHub (https://github.com/burtonrj/CytoPy). In future releases we wish to open CytoPy up to a wider audience through the integration of a graphical user interface, and we hope to expand the capabilities of CytoPy by continuing to support new methodologies in the cytometry domain. Those wishing to collaborate with us or extend our software capabilities should consult the documentation (https://cytopy.readthedocs.io/) and make a pull request on our GitHub repository.

ACKNOWLEDGMENTS

We are grateful to all peritoneal dialysis patients for participating in this study, and to the clinicians and nurses for their cooperation. We also thank Sarah Baker, Chantal Colmont, Donald Fraser, Alexander Greenshields-Watson, Ann Kift-Morgan, Kristin Ladell, Oliwia Michalak and John Pulford for their help and advice. This research received support from the Wales Kidney Research Unit (WKRU), UK Clinical Research Network (UKCRN) Study Portfolio, Medical Research Council (MRC) grant MR/N023145/1, the Welsh European Funding Office's Accelerate programme, and a School of Medicine PhD Studentship (to R.J.B.). The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

5. Supplementary Methods

5.1. FlowCAP

To assess the ability of CytoPy to classify cells we used the datasets provided in the Flow Cytometry: Critical Assessment of Population Identification Methods (FlowCAP) challenge [21], where the challenge is to accurately separate cells into subsets based on single cell phenotype. The FlowCAP-I data consist of four human studies (graft-versus-host disease, diffuse large B-cell lymphoma, symptomatic West Nile virus infection, and normal donors) and one mouse study (hematopoietic stem cell transplant). Data were labelled and pre-processing performed (removal of debris, dead material, and with fluorescence compensation applied) at source by the laboratory responsible for acquiring the original data. Here, classifiers were trained on 25% of data and classification performance tested on the remaining 75%. Performance was reported as the average of weighted F1 scores across all five datasets, where the F1 score for data with $|C|$ set of possible classes is given as:

$$weightedF1score = \frac{2}{|C|}\sum_{c\in C}\frac{precision_c \times recall_c}{precision_c + recall_c}$$

Run time was determined as the number of seconds elapsed for training and classification, as an average across every sample classified. Five supervised machine learning algorithms, housed within CytoPy, were compared without hyperparameter tuning:

1. Feed-forward neural network with three layers of size 12, 6 and 3 nodes, L2 penalty of 1×10^{-4}, ReLU activation function on the hidden layers, softmax activation function on the outer most layer, and categorical cross-entropy as the loss function; implemented in Keras v2.3.
2. XGBoost withdefault hyperparameters; implemented in XGBoost v0.9.

3. Linear Discriminant Analysis with singular value decomposition with no shrinkage and number of components equal to *min*(*n classes* – *n features*); implemented in Scikit-Learn v0.22.
4. K-Nearest Neighbours with number of neighbours used in constructing tree equalling 5 and 'ball tree' algorithm to compute nearest neighbour for classification; implemented in Scikit-Learn v0.22.
5. Support Vector Machine with radial basis function kernel; implemented in Scikit-Learn v0.22.

In each instance, data were standardised by removing the mean and scaling to unit variance; standard scores for each sample is given as $z = \left(\frac{x-u}{s}\right)$ where u is the mean and s the standard deviation.

5.2. Patients

The study cohort comprised 37 adult individuals receiving peritoneal dialysis (PD) who were admitted between October 2016 and October 2018 to the University Hospital of Wales, Cardiff, on day 1 of acute peritonitis, before commencing antibiotic treatment (34.6% female; median age 68 years, range 22-91 years). 20 age and gender-matched individuals receiving PD and with no previous infections for at least 3 months served as stable, non-infected controls (35.0% female; median age 69.5 years, range 28-93 years). Subjects known to be positive for HIV or hepatitis C virus were excluded. Clinical diagnosis of acute peritonitis was based on the presence of abdominal pain and cloudy peritoneal effluent with >100 white blood cells/mm^3. According to the microbiological analysis of the effluent by the routine Microbiology Laboratory, Public Health Wales, episodes of peritonitis were defined as infections caused by Gram-positive or Gram-negative organisms. Cases of fungal infection and negative or unclear culture results were excluded from this analysis. Basic patient demographics can be found in the Supplementary Methods and a summary of the bacterial culture results for patients with peritonitis are shown in Supplementary

Table S1. All methods were carried out in accordance with relevant guidelines and regulations, and written informed consent was obtained from all subjects. Recruitment of PD patients was approved by the South East Wales Local Ethics Committee under reference number 04WSE04/27, and conducted according to the principles expressed in the Declaration of Helsinki. The study was registered on the UK Clinical Research Network Study Portfolio under reference numbers #11838 "Patient immune responses to infection in Peritoneal Dialysis" (PERIT-PD).

5.3 Flow cytometry

Peritoneal leukocytes were harvested from overnight dwell effluents and processed as described previously (27; 41); samples were treated with DNase (Sigma; 1:2,500 dilution) when excessive debris was visually apparent. Leukocyte populations in total effluent were stained using monoclonal antibodies against CD1c, CD3, CD14, CD15, CD16, CD19, CD45, CD116, HLA-DR and Siglec-8 (Supplementary Table S2) and identified as CD45^{+} immune cells, CD3^{+} T cells, CD19^{+} B cells, CD15^{-}CD14^{+} monocytes/macrophages, CD15^{+} neutrophils, CD15^{-}CD14$^{+/-}$CD1c^{+} dendritic cells, and CD15^{-}Siglec-8^{+} eosinophils. T cell subsets in peripheral blood mononuclear cells (PBMCs) and in peritoneal effluent were stained after Ficoll-Paque (Fisher Scientific) separation of blood and peritoneal leukocytes, respectively, using monoclonal antibodies against CD3, CD4, CD8, CD161, TCR-Vα7.2, TCR-Vδ2, TCR-pan-$\gamma\delta$, CD45RA, CCR7 and CD27 (Supplementary Table S3). Cell acquisition by flow cytometry was performed using a 16 colour BD LSR Fortessa cell analyser (BD Biosciences). Live single cells were gated based on side and forward scatter area/height and live/dead staining (fixable Aqua; Invitrogen).

5.4 Meta-clustering

Meta-clustering was performed to find a consensus amongst the individual clustering results of many individual samples. Each sample was independently normalized; that is, each feature was scaled:

$$x_{norm} = \frac{x - min(x)}{max(x) - min(x)}$$

Where x is the original value for a given feature and x_{norm} is its values scaled between zero and one. Once each sample was individually normalized, the clusters from each sample were extracted and their centroid calculated; by default this was given as the median of their feature vector but other definitions of center can be used (*e.g.* mean, geometric mean *etc.*). Cluster centroids were annotated as to which sample they originated from and their original cluster ID and then concatenated into a single dataframe. This dataframe was then used as the input to a clustering algorithm of the user's choosing.

6. References

1. *flowDensity: reproducing manual gating of flow cytometry data by automated density-based cell population identification.* **Malek, Mehrnoush, et al., et al.** 10 2014, Bioinformatics, Vol. 31, pp. 606-607. ISSN: 1367-4803.
2. *OpenCyto: An Open Source Infrastructure for Scalable, Robust, Reproducible, and Automated, End-to-End Flow Cytometry Data Analysis.* **Finak, Greg, et al., et al.** s.l. : Public Library of Science, 8 2014, PLOS Computational Biology, Vol. 10, pp. 1-12.
3. *FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data.* **Van Gassen, Sofie, et al., et al.** 2015, Cytometry Part A, Vol. 87, pp. 636-645.
4. *Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis.* **Levine, Jacob H., et al., et al.** 2015, Cell, Vol. 162, pp. 184-197. ISSN: 0092-8674.
5. *Automated mapping of phenotype space with single-cell data.* **Samusik, Nikolay, et al., et al.** 2016, Nature Methods, Vol. 13, pp. 493-496. ISSN: 1548-7105.
6. *Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE.* **Qiu, Peng, et al., et al.** 2011, Nature Biotechnology, Vol. 29, pp. 886-891. ISSN: 1546-1696.
7. *flowLearn: fast and precise identification and quality checking of cell populations in flow cytometry.* **Lux, Markus, et al., et al.** 2 2018, Bioinformatics, Vol. 34, pp. 2245-2253. ISSN: 1367-4803.
8. *Gating mass cytometry data by deep learning.* **Li, Huamin, et al., et al.** 7 2017, Bioinformatics, Vol. 33, pp. 3423-3430. ISSN: 1367-4803.
9. *Automatic Classification of Cellular Expression by Nonlinear Stochastic Embedding (ACCENSE).* **Shekhar, Karthik, et al., et al.** s.l. : National Academy of Sciences, 2014, Proceedings of the National Academy of Sciences, Vol. 111, pp. 202–207. ISSN: 0027-8424.
10. *Sensitive detection of rare disease-associated cell subsets via representation learning.* **Arvaniti, Eirini and Claassen, Manfred.** 2017, Nature Communications, Vol. 8, p. 14825. ISSN: 2041-1723.
11. *Robust prediction of clinical outcomes using cytometry data.* **Hu, Zicheng, Glicksberg, Benjamin S. and Butte, Atul J.** 8 2018, Bioinformatics, Vol. 35, pp. 1197-1203. ISSN: 1367-4803.
12. *Automated analysis of flow cytometry data comes of age.* **Brinkman, Ryan R., et al., et al.** 2016, Cytometry Part A, Vol. 89, pp. 13-15.
13. *Comparison of clustering methods for high-dimensional single-cell flow and mass cytometry data.* **Weber, Lukas M. and Robinson, Mark D.** 2016, Cytometry Part A, Vol. 89, pp. 1084-1096.
14. *Computational flow cytometry: helping to make sense of high-dimensional immunology data.* **Saeys, Yvan, Van Gassen, Sofie and Lambrecht, Bart N.** 2016, Nature Reviews Immunology, Vol. 16, pp. 449-462. ISSN: 1474-1741.
15. *The end of gating? An introduction to automated analysis of high dimensional cytometry data.* **Mair, Florian, et al., et al.** 2016, European Journal of Immunology, Vol. 46, pp. 34-43.
16. **StackOverflow.** Stack Overflow Developer Survey. *Stack Overflow.* [Online] 2019. https://insights.stackoverflow.com/survey/2019.
17. *Critical assessment of automated flow cytometry data analysis techniques.* **Aghaeepour, Nima, et al., et al.** 2013, Nature Methods, Vol. 10, pp. 228-238. ISSN: 1548-7105.
18. **Inc, MongoDB.** *MongoDB.* [Online] MongoDB Inc. [Cited: March 28, 2020.] https://www.mongodb.com/.
19. **Mongoengine organisation.** github/monogengine. *github.* [Online] [Cited: April 3, 2020.] https://github.com/mongoengine.
20. *Scikit-learn: Machine Learning in Python.* **Pedregosa, F., et al., et al.** 2011, Journal of Machine Learning Research, Vol. 12, pp. 2825–2830.
21. *SciPy 1.0: fundamental algorithms for scientific computing in Python.* **Virtanen, Pauli, et al., et al.** 2020, Nature Methods, Vol. 17, pp. 261-272. ISSN: 1548-7105.

22. *Tidy data.* **Wickham, Hadley.** 10, 2014, The Journal of Statistical Software, Vol. 59.
23. *Hierarchical Modeling for Rare Event Detection and Cell Subset Alignment across Flow.* **Cron A, Gouttefangeas C, Frelinger J, Lin L, Singh SK, et al.** 7, s.l. : PLoS Comput Biol, 2013, Vol. 9. doi:10.1371/journal.pcbi.1003130.
24. *flowClust: a Bioconductor package for automated gating of flow.* **Kenneth Lo, Florian Hahne , Ryan R Brinkman and Raphael Gottardo.** 145, s.l. : BMC Bioinformatics, 2009, Vol. 10. 10.1186/1471-2105-10-145.
25. *Machine-learning algorithms define pathogen-specific local immune fingerprints in peritoneal dialysis patients with bacterial infections.* **Jingjing Zhang, et al.** 1, s.l. : Kidney International, 2017, Vol. 92. 10.1016/j.kint.2017.01.017.
26. *Peritoneal macrophage heterogeneity is associated with different peritoneal dialysis outcomes.* **Liao, Chia-Te, et al., et al.** s.l. : Elsevier, 5 01, 2017, Kidney International, Vol. 91, pp. 1088-1103. ISSN: 0085-2538.
27. *Pathogen-Specific Local Immune Fingerprints Diagnose Bacterial Infection in Peritoneal Dialysis Patients.* **Lin, Chan-Yu, et al., et al.** s.l. : American Society of Nephrology, 2013, Journal of the American Society of Nephrology, Vol. 24, pp. 2002–2009. ISSN: 1046-6673.
28. *Measurement of innate immune response biomarkers in peritoneal dialysis effluent using a rapid diagnostic point-of-care device as a diagnostic indicator of peritonitis.* **Catriona Goodlad, Sophiamma George, Shella Sandoval, Stephen Mepham, Gita Parekh, Matthias Eberl, Nicholas Topley, Andrew Davenport, et al.** s.l. : Kidney International, 2020. 10.1016/j.kint.2020.01.044.
29. *XGBoost: A Scalable Tree Boosting System.* **Chen, Tianqi and Guestrin, Carlos.** s.l. : Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016. 10.1145/2939672.2939785.
30. **Chollet, François and others.** Keras. *Keras.* 2015.
31. *Using artificial intelligence to reduce diagnostic workload without compromising detection of urinary tract infections.* **Ross J. Burton, Mahableshwar Albur, Matthias Eberl, Simone M. Cuff.** s.l. : BMC Medical Informatics and Decision Making, 2019, Vol. 19: 171. 10.1186/s12911-019-0878-9.
32. *Do we Need Hundreds of Classifiers to Solve Real World Classification Problems?* **Manuel Fern´andez-Delgado, Eva Cernadas, Sen´en Barro, Dinani Amorim.** s.l. : Journal of Machine Learning Research, 2014, Vols. 15: 3133-3181. 10.5555/2627435.2697065.
33. *Density Estimation in Python (Version v0.9.10).* **Odland, Tommy.** s.l. : Zenodo, 2018. http://doi.org/10.5281/zenodo.2392268.
34. **SciPy.** scipy.spatial.distance.jensenshannon. *scipy.github.io.* [Online] [Cited: March 28, 2020.] https://scipy.github.io/devdocs/generated/scipy.spatial.distance.jensenshannon.html.
35. *LAVENDER: latent axes discovery from multiple cytometry samples with non-parametric divergence estimation and multidimensional scaling reconstruction.* **Naotoshi Nakamura, Daigo Okada, Kazuya Setoh, Takahisa Kawaguchi, Koichiro, Yasuharu Tabara, Fumihiko Matsuda, Ryo Yamada.** s.l. : BioRxiv, 2019, Vol. 673434. https://doi.org/10.1101/673434.
36. *Jensen-Shannon Divergence and Hilbert space embedding.* **Bent Fuglede, Flemming Topsøe.** s.l. : International Symposium on Information Theory, 2004. 10.1109/ISIT.2004.1365067.
37. *Pathogen-Specific Immune Fingerprints during Acute Infection: The Diagnostic Potential of Human γδ T-Cells.* **Matthias Eberl, Ida M. Friberg, Anna Rita Liuzzi, Matt P. Morgan, Nicholas Topley.** s.l. : Frontiers in Immunology, 2014, Vol. 5: 572. 10.3389/fimmu.2014.00572.
38. **Shen, Sangyu.** FlowSOM. *GitHub.* [Online] May 13, 2019. [Cited: March 28, 2020.] https://github.com/Hatchin/FlowSOM.
39. *Dimensionality reduction for visualizing single-cell data using UMAP.* **Becht, E., McInnes, L., Healy, J. et al.** s.l. : Nature Biotechnology, 2019, Vols. 37: 38-44. https://doi.org/10.1038/nbt.4314.
40. *Visualizing structure and transitions in high-dimensional biological data.* **Moon, K.R., van Dijk, D., Wang, Z. et al.** s.l. : Nature Biotechnology, 2019, Vols. 37: 1482-1492. https://doi.org/10.1038/s41587-019-0336-3.

41. *Control of neutrophil influx during peritonitis by transcriptional cross-regulation of chemokine CXCL1 by IL-17 and IFN-γ.* **Catar, R.A., Chen, L., Cuff, S.M., Kift-Morgan, A., Eberl, M., Kettritz, R., Kamhieh-Milz, J., Moll, G., Li, Q., Zhao, H., Kawka, E., Zickler, D., Parekh, G., Davis, P., Fraser, D.J., Dragun, D., Eckardt, K.-U., Jörres, A. and Witowski, J.** s.l. : Journal of Pathology, 2020. https://doi.org/10.1002/path.5438.
42. *A Rapid Crosstalk of Human γδ T Cells and Monocytes Drives the Acute Inflammation in Bacterial Infections.* **Matthias Eberl, Gareth W. Roberts,Simone Meuter,John D. Williams,Nicholas Topley,Bernhard Moser.** s.l. : PLOS Pathogens, 2009. 10.1371/journal.ppat.1000308.
43. **Cytobank Inc.** Cytobank. [Online] [Cited: October 26, 2020.] https://www.cytobank.org/.
44. *Cytofkit: A Bioconductor Package for an Integrated Mass Cytometry Data Analysis Pipeline.* **Hao Chen, Mai Chan Lau,Michael Thomas Wong,Evan W. Newell,Michael Poidinger,Jinmiao Chen.** 9, s.l. : PLOS Computaitonal Biology, 2016, Vol. 12. https://doi.org/10.1371/journal.pcbi.1005112.
45. *TensorFlow: A system for large-scale machine learning.* **Abadi, Martin, et al., et al.** 2016. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16). pp. 265–283.
46. *Hahne F, Khodabakhshi AH, Bashashati A, et al. Per-channel basis normalization methods for flow cytometry data. Cytometry A. 2010;77(2):121–131. doi:10.1002/cyto.a.20823.* **Hahne, F., Khodabakhshi, A. H., Bashashati, A., Wong, C. J., Gascoyne, R. D., Weng, A. P., Seyfert-Margolis, V., Bourcier, K., Asare, A., Lumley, T., Gentleman, R, Brinkman, R. R.** s.l. : Cytometry A, 2010, Vols. 77(2): 121-131. 10.1002/cyto.a.20823.
47. *High-throughput flow cytometry data normalization for clinical trials.* **Finak G, Jiang W, Krouse K, Wei C, Sanz I, Phippard D, Asare A, De Rosa SC, Self S, Gottardo R.** s.l. : Cytometry A, 2014, Vols. 85(3): 277-286. 10.1002/cyto.a.22433.
48. *Robust prediction of clinical outcomes using cytometry data.* **Zicheng Hu, Benjamin S Glicksberg, Atul J Butte.** s.l. : Bioinformatics, 2019, Vols. 35(7): 1197-1203. 10.1093/bioinformatics/bty768.
49. *Removal of Batch Effects using Distribution-Matching Residual Networks.* **Shaham U, Stanton KP, Zhao J, Li H, Raddassi K, Montgomery R, Kluger Y.** s.l. : Bioinformatics, 2017, Vols. 33(16): 2539-2546. 10.1093/bioinformatics/btx196.
50. *CytoNorm: A Normalization Algorithm for Cytometry Data.* **Sofie Van Gassen, Brice Gaudilliere, Martin S. Angst, Yvan Saeys, Nima Aghaeepour.** s.l. : Cytometry Part A, 2019, Vols. 97 (3): 268-279. doi.org/10.1002/cyto.a.23904.
51. *A robust and interpretable, end-to-end deep learning model for cytometry data.* **Hu, Zicheng, et al., et al.** s.l. : Cold Spring Harbor Laboratory, 2020, bioRxiv.

7. Tables

Model	Weighted F1 score (mean [95% CI])	Runtime (seconds)
Feed-Forward Neural Network (4 layers)	0.966 [0.956 – 0.975]	9.62
K-Nearest Neighbours	0.917 [0.884 – 0.948]	0.93
Linear Discriminant Analysis	0.918 [0.892 – 0.943]	0.61
Support Vector Machine (Radial Kernel)	0.964 [0.955 – 0.972]	8.68
XGBoost	0.980 [0.976 – 0.984]	18.6

Table 1. Performance of five different supervised classifiers on FlowCAP-I data.

8. Figures

Figure 1: Overview of the CytoPy framework and list of primary dependencies. Single cell data and experiments/clinical metadata (1) are used to populate a project within the CytoPy database (2). The CytoPy database models analytical data in MongoDB documents (cylinder) and an interface of CytoPy classes retrieves and commits data to this database (dotted rounded rectangle). The components of this interface are used to complete the following tasks. (3) Autonomous gating identifies a clean 'root' population for analysis. (4) Inter-sample variation is visualised, quantified, and samples are grouped according to their similarity in high-dimensional space. (5) Cells are classified by supervised and unsupervised methodologies and visualised for exploratory data analysis. (6) Finally, single cell data can be summarised and feature selection techniques employed to find variables of interest.

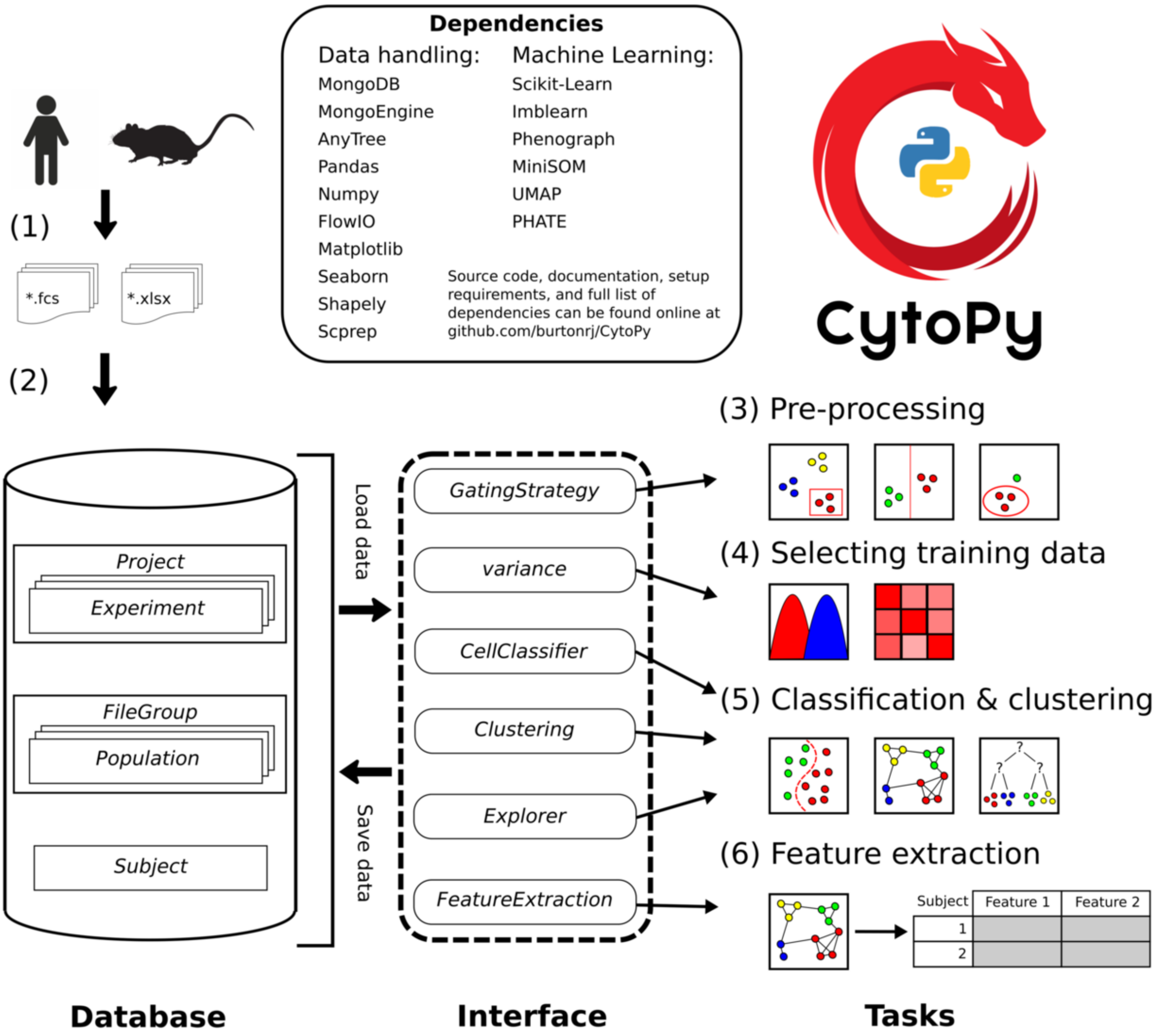


Figure 2: Examples of using autonomous gates for identification of immune cells in a biological sample, as exemplified by the identification of T lymphocytes in peritoneal drain fluid from a patient with acute peritonitis. Algorithm-driven gates are applied on each two-dimensional plot in accession according to a user defined gating template and population hierarchy. The first gate (A) in the sequence filters out the majority of debris using a static rectangular boundary applied to forward scattered light area (FSC-A) and sideward scattered light area (SSC-A). Those events positive for the pan-T cell marker CD3 are identified with a density-dependent autonomous gate (B) that finds a threshold at the point of minimal density using properties of a probability density function. (C) Density-dependent autonomous gating then identifies live cells within the CD3$^+$ cell population; those below the threshold found for live/dead stain. Live single CD3$^+$ cells are further discriminated from other events by applying Gaussian mixture models to create an elliptical gate (D) using FSC-A and forward scattered height (FSC-H), and a density-dependent gate (E) using sideward scattered light width (SSC-W). Finally, the T cell population is identified using FSC-A and SSC-A and encapsulated by an elliptical gate generated by a Gaussian mixture model.

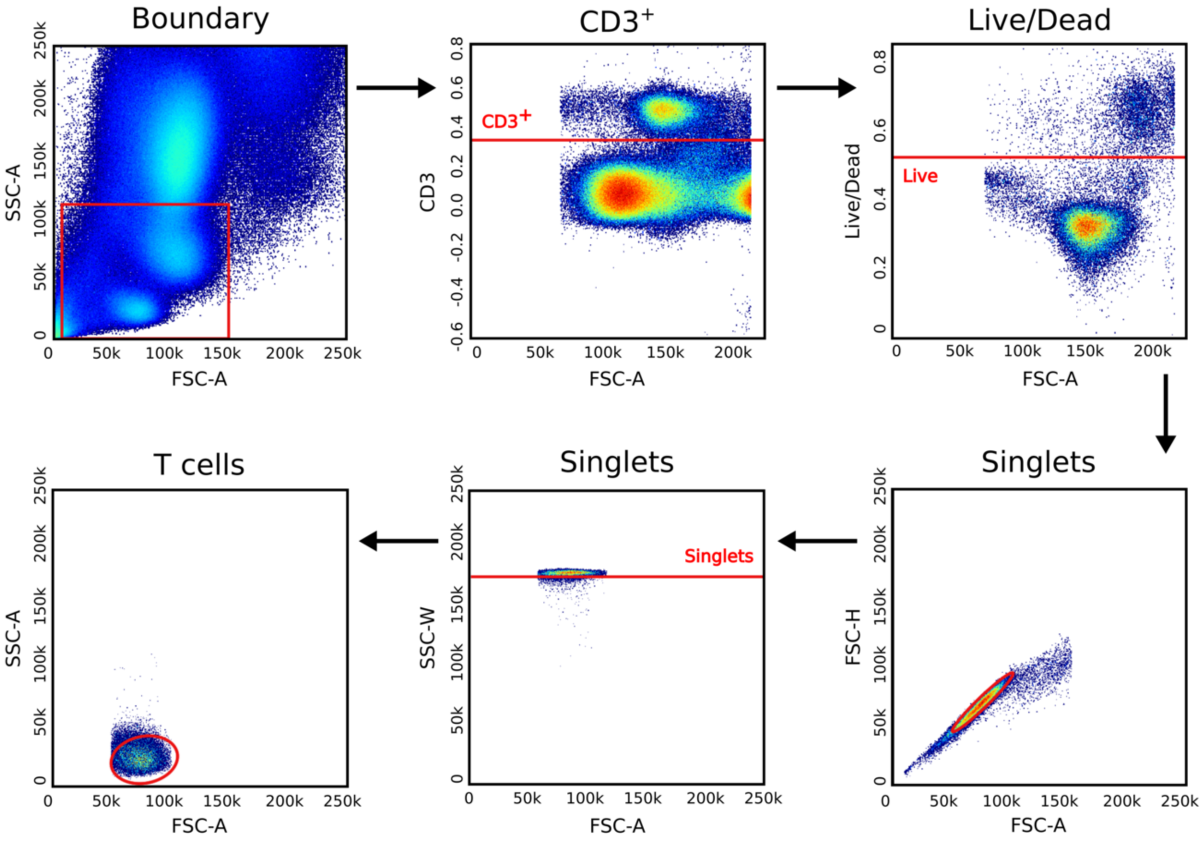


Figure 3: Comparison of autonomous gates and expert manual gating when identifying major immune cell subsets in peritoneal effluent. From left to right, the bar chart shows the most populous cell subsets such as neutrophils (CD19$^-$ CD3$^-$ CD14$^-$ Siglec8$^-$ CD15$^+$) down to small populations such as B cells (CD19$^+$ CD3$^-$) and Eosinophils (CD19$^-$ CD3$^-$ CD14$^-$ Siglec8$^+$ CD15$^{+/-}$). The accompanying line plot shows the R-squared value when comparing the correlation between the population size detected by expert manual gating vs. autonomous gates.

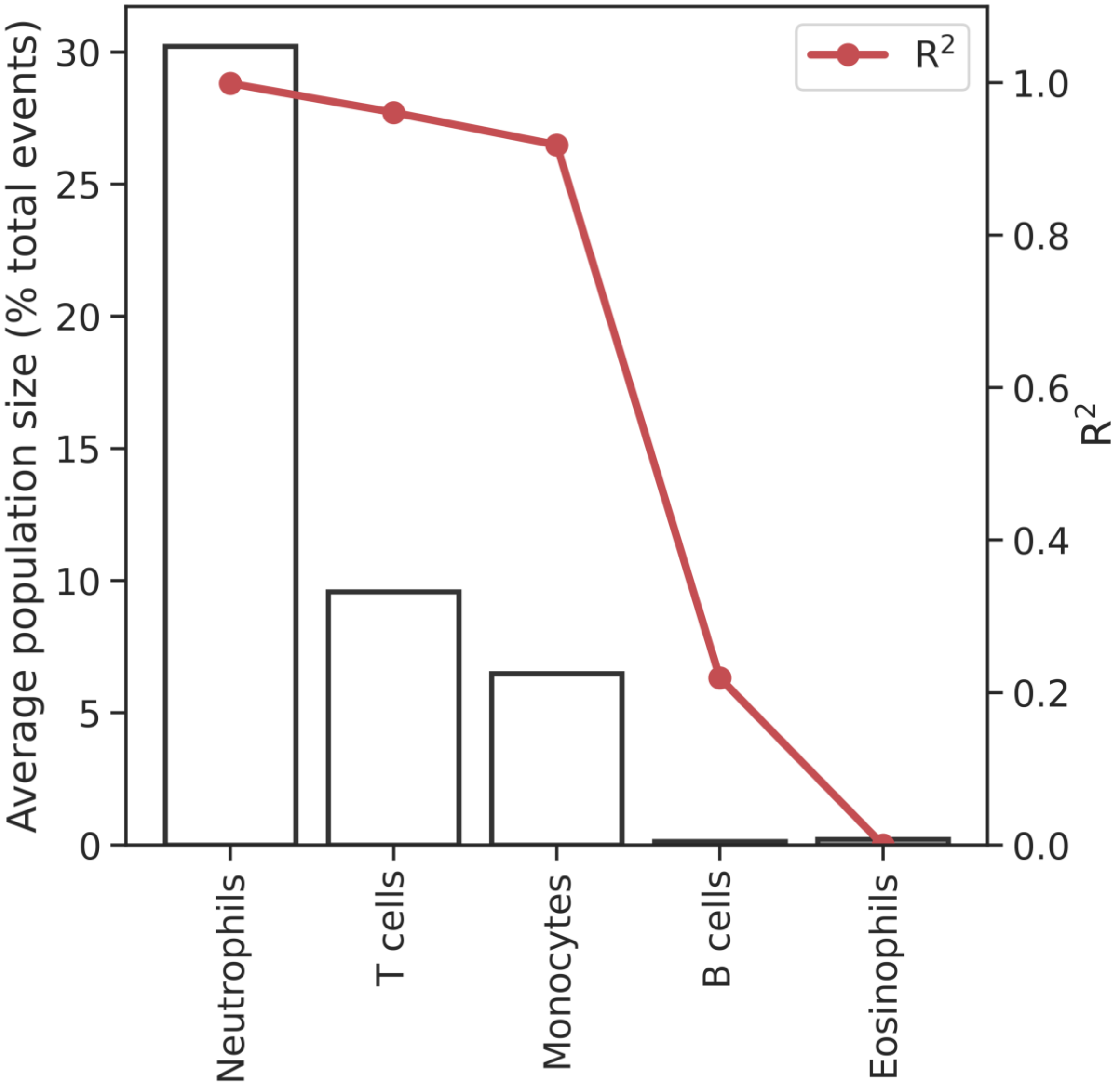


Figure 4: Variance in cell marker abundance as measured by flow cytometry for T cells in peritoneal drain fluid (CD3$^+$ lymphocytes). A reference subject (325-01) is shown in blue and 9 other randomly selected subjects are overlaid for comparison in red. (A) Variation in individual parameters can be shown by kernel density estimation as shown here for 6 common parameters of interest in T cell biology, identifying all T cells (CD3) or the helper T cells (CD4) and cytotoxic T cells (CD8) populations, as well as surface markers associated with specific effector and memory subsets within these populations (CD45RA, CD27, CCR7). (B) Multi-variant drift can be visualised using dimensionality reduction techniques such as PCA. The same reference subject 325-01 as shown in (A) is given in blue and in each plot a different subject in red is overlaid.

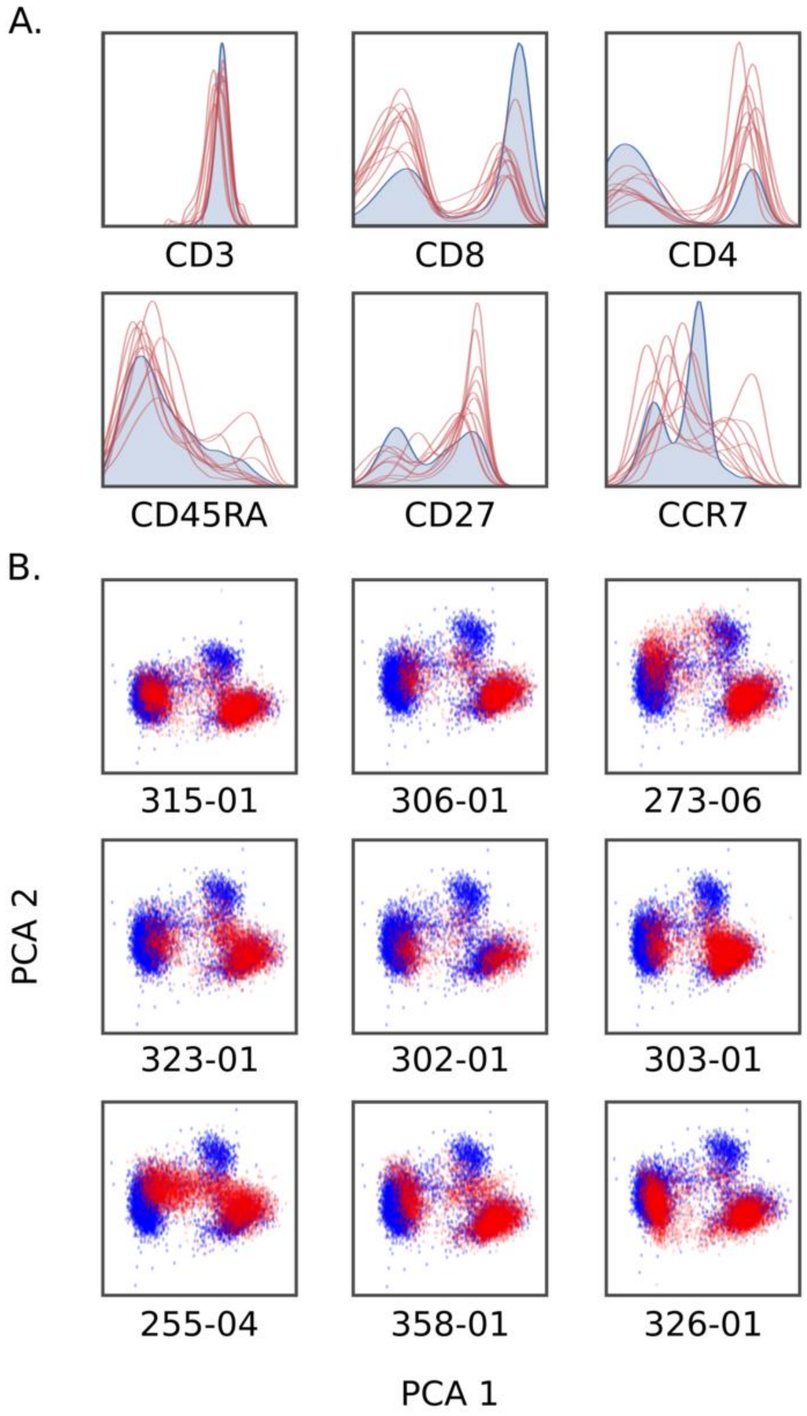


Figure 5: Heatmap display of pairwise Jenson-Shannon Distances for all leukocyte subsets present in peritoneal drain fluid and subsets within the T cell compartment present in peritoneal drain fluid and whole blood. Jenson-Shannon distance is given as √JSD(p,q) where p and q are the PDFs of each given pair as estimated using a Gaussian kernel and JSD is a function for Jenson-Shannon divergence. Single linkage clustering is applied to each matrix to reveal groups of broad similarity.

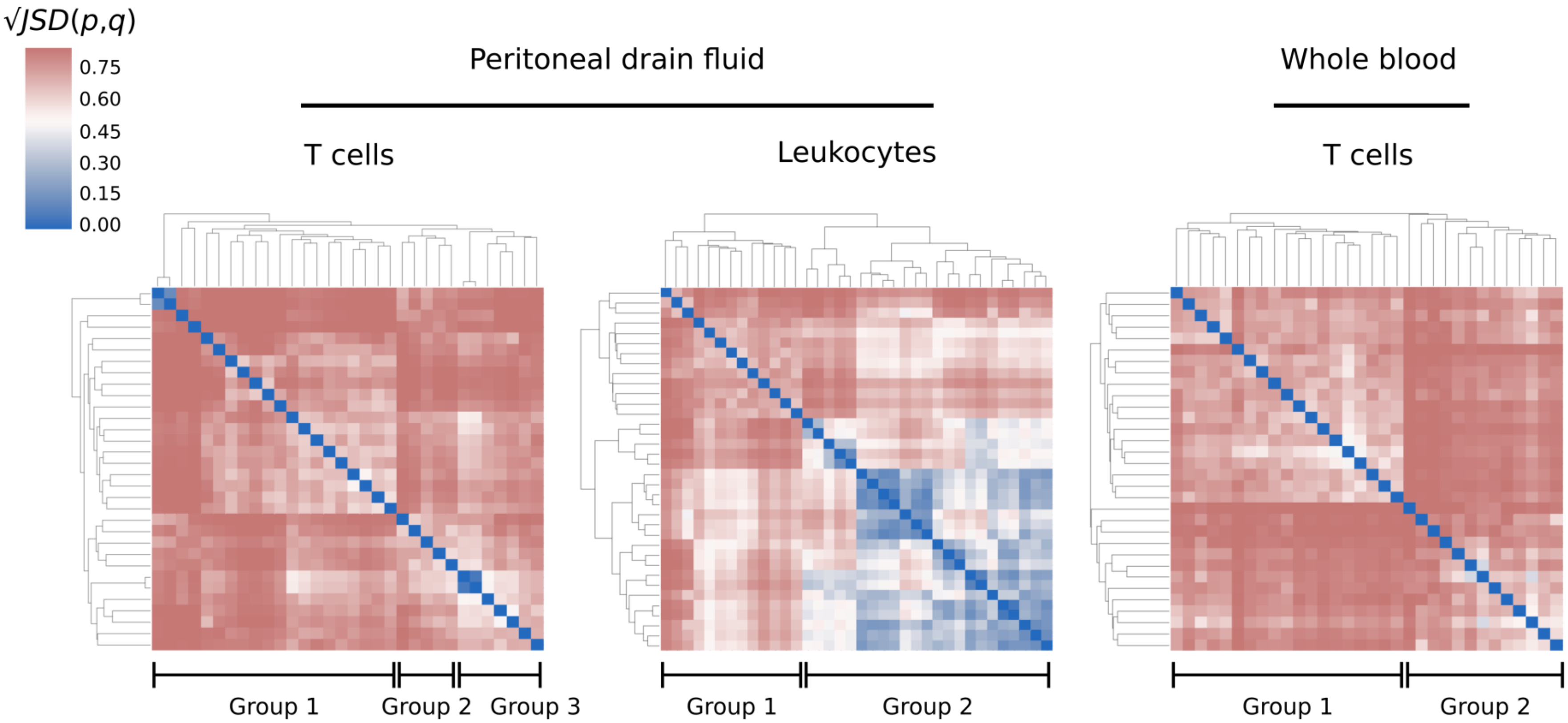


Figure 6: Performance of XGBoost for cell classification of CD45$^+$ leukocytes from peritoneal drain fluid and T cells from peritoneal drain fluid and whole blood. Groups are generated from all patients (infected and non-infected) as described in Figure 4. XGBoost performance was assessed by weighted F1 score on 5 randomly chosen validation samples within each independent group (A); where groups represent samples clustered on pairwise JSD and an independent classifier is trained for each group. (B) Classification performance of individual classes for an obvious outlier in T lymphocytes from drain fluid, group 1 (weighted F1 score equal to 0.6) is shown visually as a confusion matrix. The values in each row are normalised according to class support (the number of events in a given class). The diagonal of the confusion matrix is equivalent to the accuracy of classification for a particular class. (C) Back-gating functionality allows for close inspection of supervised classification results and comparison to manual gates, semi-autonomous gates, or clustering results. The classification of γδ T cells in this example is compared to a manual gate.

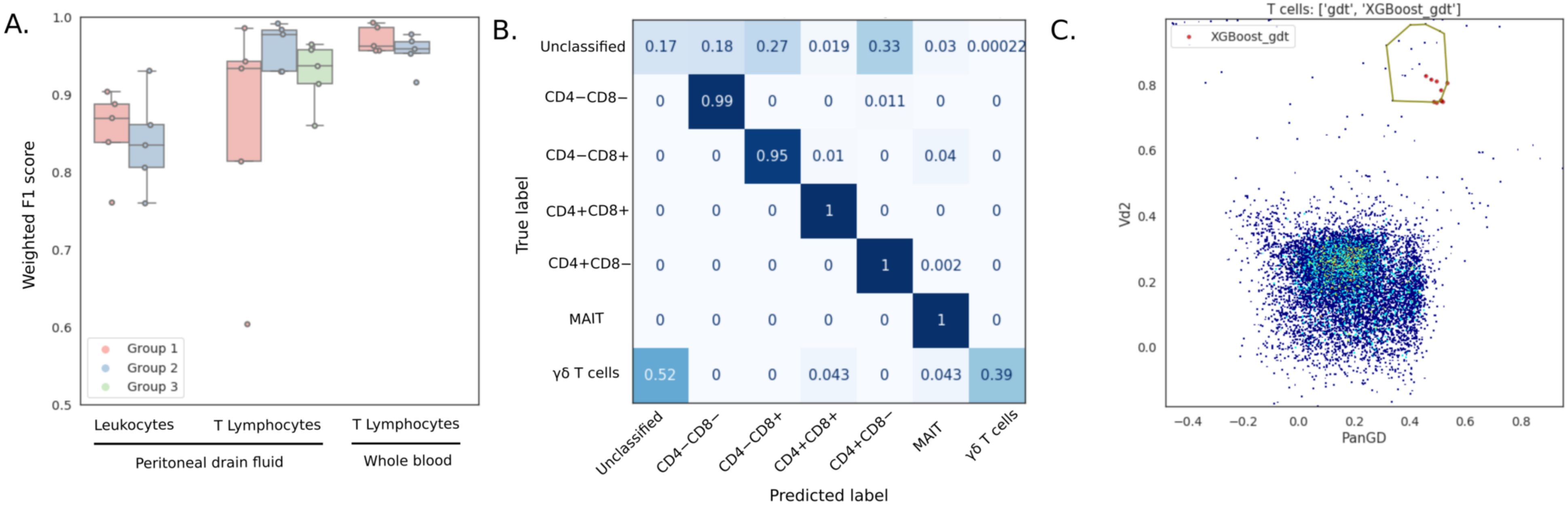


Figure 7: PhenoGraph meta-clustering results for CD45$^+$ leukocytes present in peritoneal drain fluid from all available patient samples. (A) The heatmap shows the phenotype of meta-clusters. Individual clusters from all patients are shown in a UMAP plot where each colour filled circle is a unique cluster from an individual subject. Its colour corresponds to its meta-cluster enrolment and its size the proportion of cells relative to the number of CD45$^+$ leukocytes. (B) Patient phenotype (stable control or acute peritonitis) is categorised by colour in a UMAP plot, showing individual clusters from all patients, and box plots show the difference in the proportion of cells as a percentage of CD45$^+$ leukocytes; the difference in distribution of population proportions was tested by Mann-Witney U test; **** $p \leq 0.001$

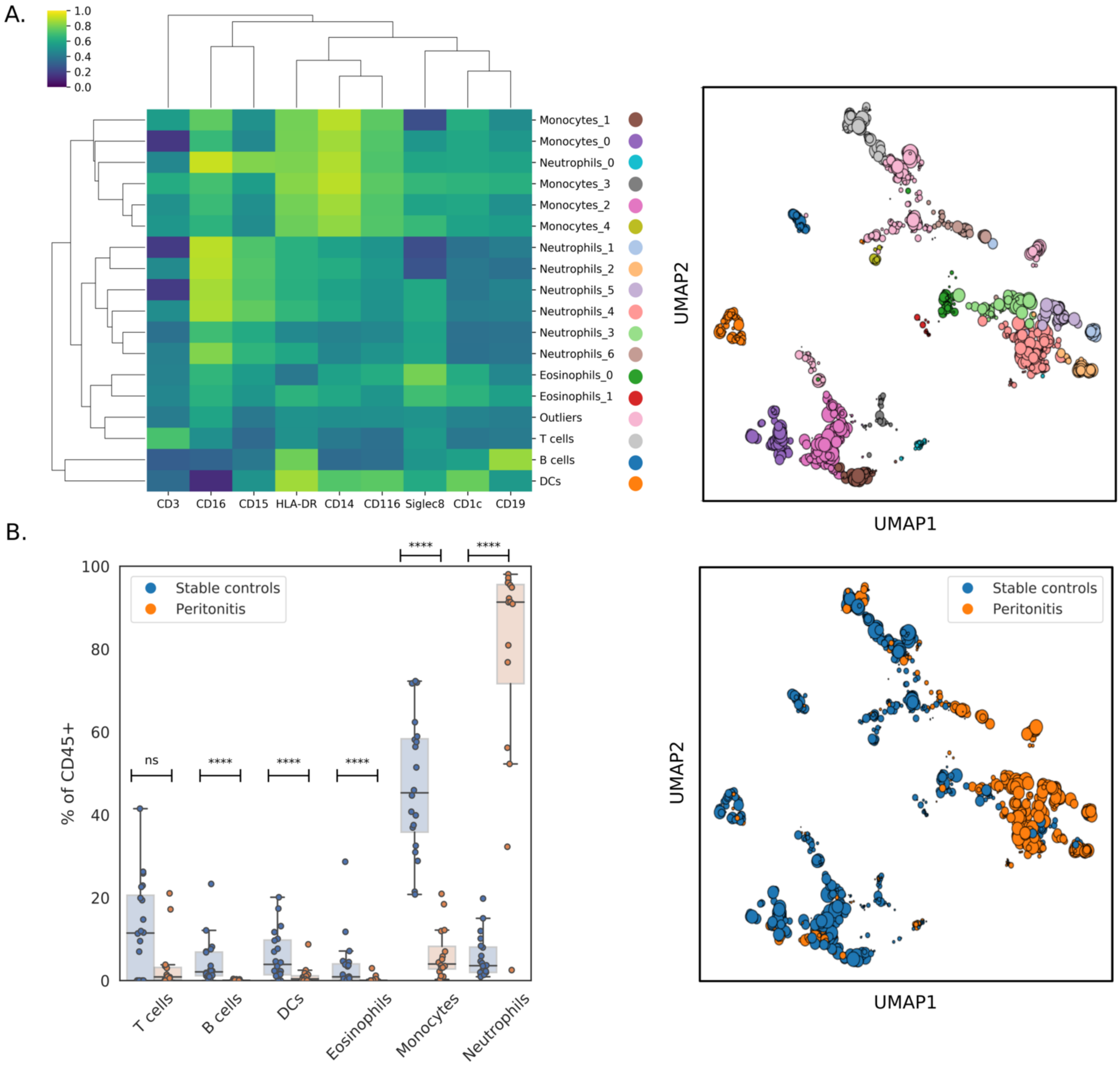


Figure 8: PhenoGraph meta-clustering results for T cells in peritoneal drain fluid from all available patient samples. (A) The heatmap shows the phenotype of meta-clusters. Individual clusters from all subjects are shown in a UMAP plot where each colour filled circle is a unique cluster from an individual subject. Its colour corresponds to its meta-cluster enrolment, and its size the proportion of cells relative to the number of T cells. (B) Meta-cluster results can be coloured by patient phenotype to reveal regions that distinguish clinical endpoints. Patient phenotype is contrasted by colour in clusters on a UMAP plot and in box plots of major T cell subsets as a percentage of total T cells. (C) Close inspection of CD8^{+} T cells shows that functionally distinct effector/memory subsets can be identified by PhenoGraph clustering.

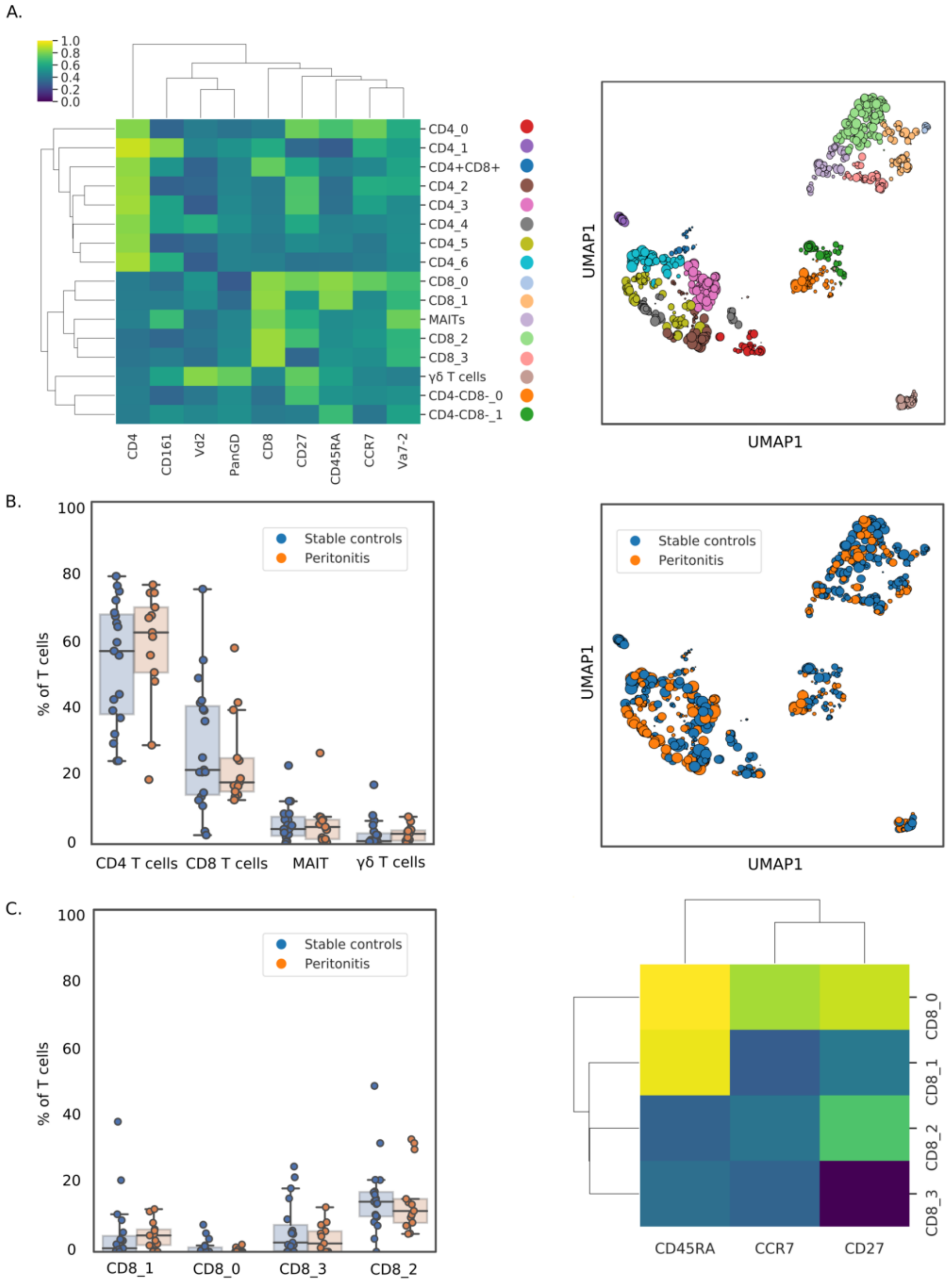
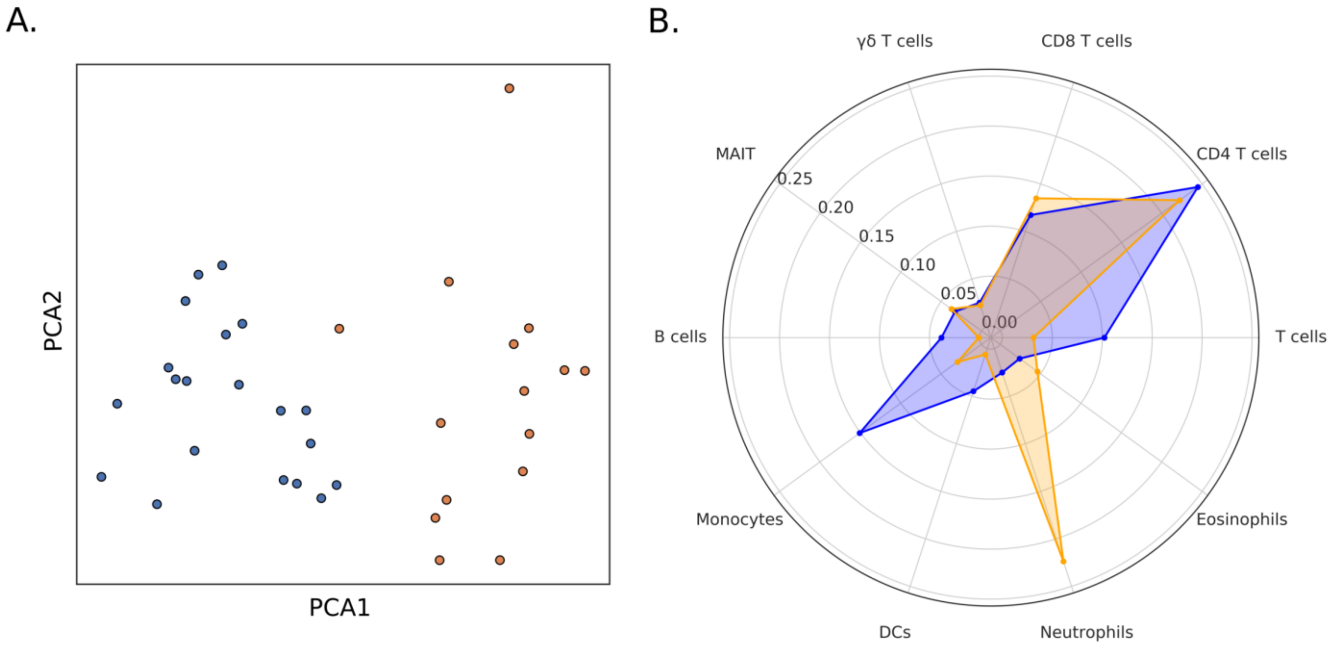


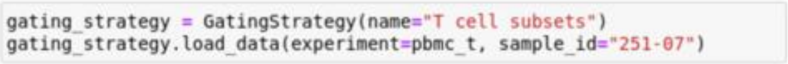
Figure 9: (A) Principle component analysis of all identified cell populations shows separation of patients with acute peritonitis from stable controls. (B) Radial plot of major cell subsets given as the proportion of their derived parent population (MAIT cells, γδ T cell, CD8 T cells, CD4 T cells: proportion of total T cells; all others: proportion of CD45^{+} immune cells). Values shown are the consensus of XGBoost classification and PhenoGraph clustering.



9. Supplementary data

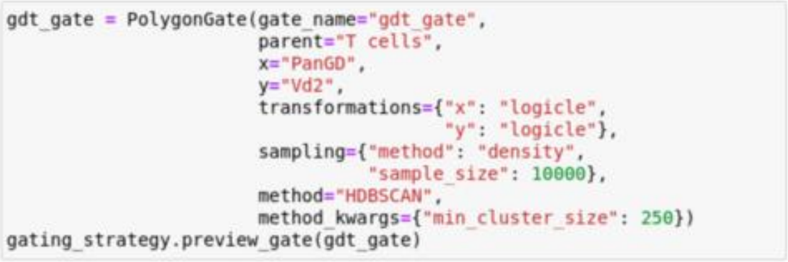
Supplementary Figure S1: Demonstration of low-code API for generation of an autonomous gate and subsequent application of gate to new data. Generation of a new gating strategy and import of existing cytometry data is simplified by the *GatingStrategy* class (A). Autonomous gates can then be defined and associated to a *GatingStrategy* object (B) using any existing algorithm in the Scikit-Learn library and libraries that follow the Scikit-Learn standard template architecture (e.g. HDBSCAN). A preview of the gate is generated that the use can then annotate with the *label_children* method (C). When applying this gate to new data (D), the chosen algorithm is ‘fitted’ to the newly encountered data and resulting clustered matched to the clusters generated when the gate was defined (as shown in the prior steps, B and C).

A.

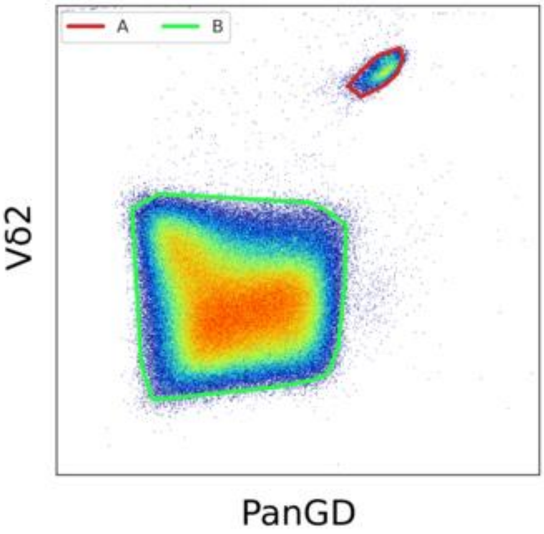


```
gating_strategy = GatingStrategy(name="T cell subsets")
gating_strategy.load_data(experiment=pbmc_t, sample_id="251-07")
```

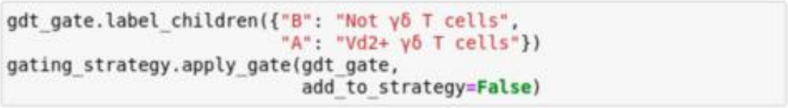
B.



```
gdt_gate = PolygonGate(gate_name="gdt_gate",
                       parent="T cells",
                       x="PanGD",
                       y="Vd2",
                       transformations={"x": "logicle",
                                        "y": "logicle"},
                       sampling={"method": "density",
                                 "sample_size": 10000},
                       method="HDBSCAN",
                       method_kwargs={"min_cluster_size": 250})
gating_strategy.preview_gate(gdt_gate)
```

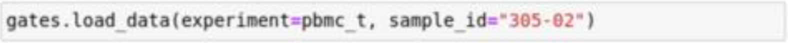


C.

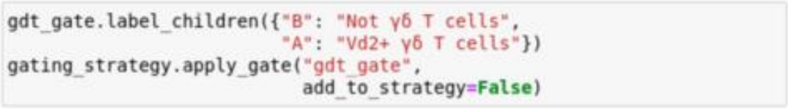


```
gdt_gate.label_children({"B": "Not γδ T cells",
                         "A": "Vd2+ γδ T cells"})
gating_strategy.apply_gate(gdt_gate,
                           add_to_strategy=False)
```

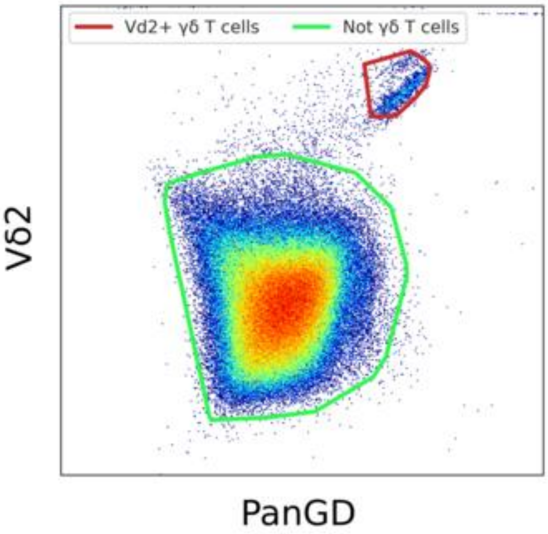
D.



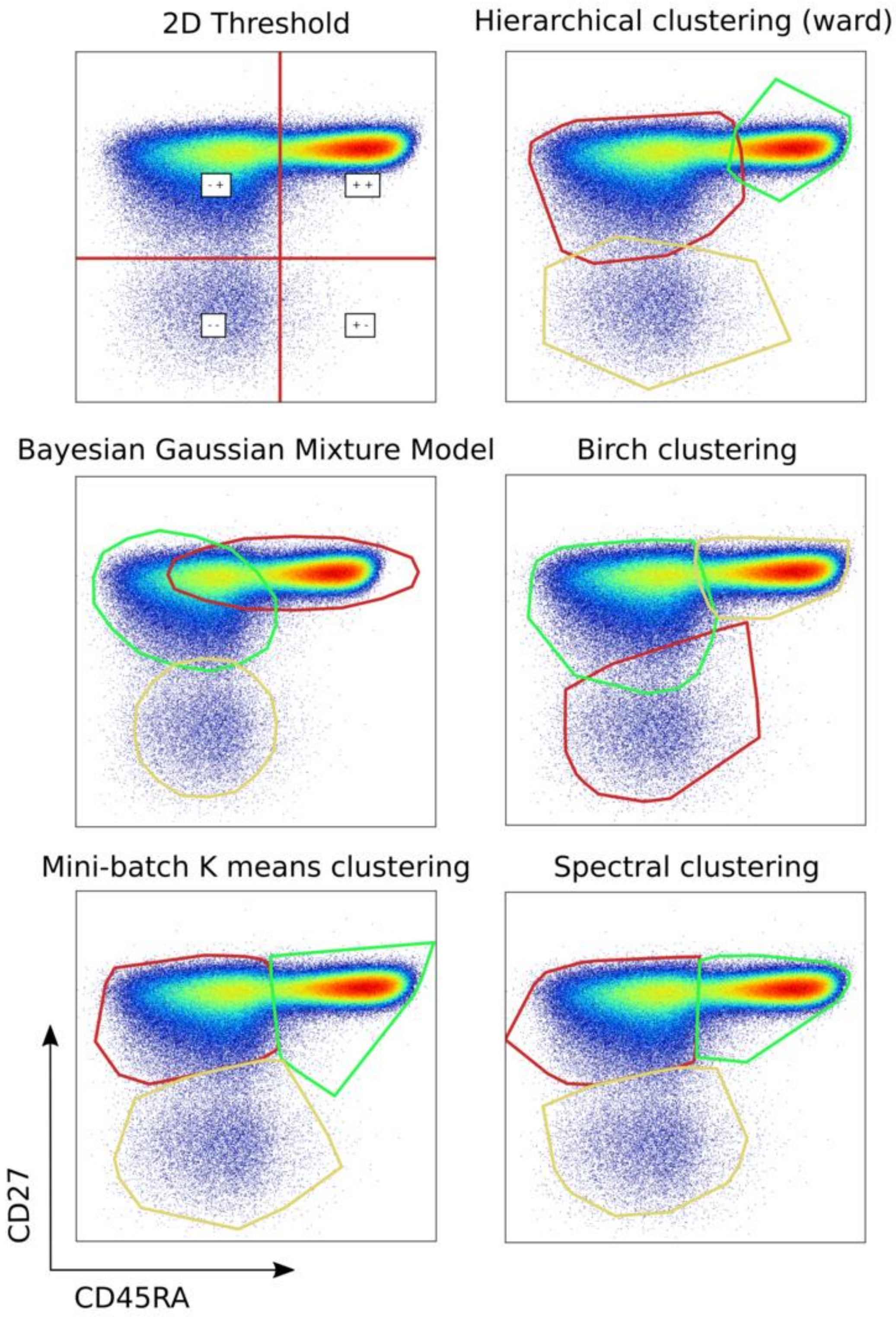
```
gates.load_data(experiment=pbmc_t, sample_id="305-02")
```



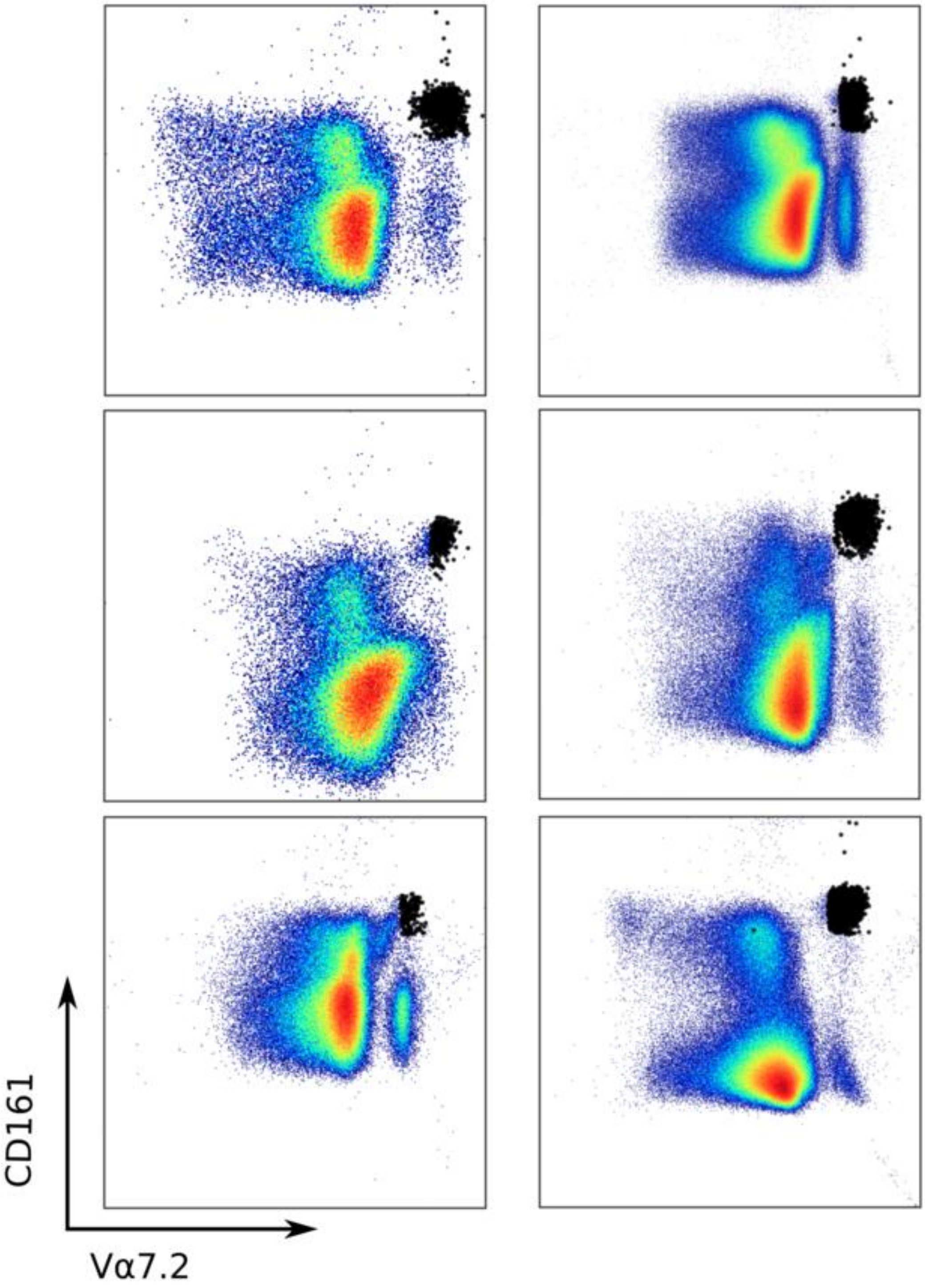
```
gdt_gate.label_children({"B": "Not γδ T cells",
                         "A": "Vd2+ γδ T cells"})
gating_strategy.apply_gate("gdt_gate",
                           add_to_strategy=False)
```



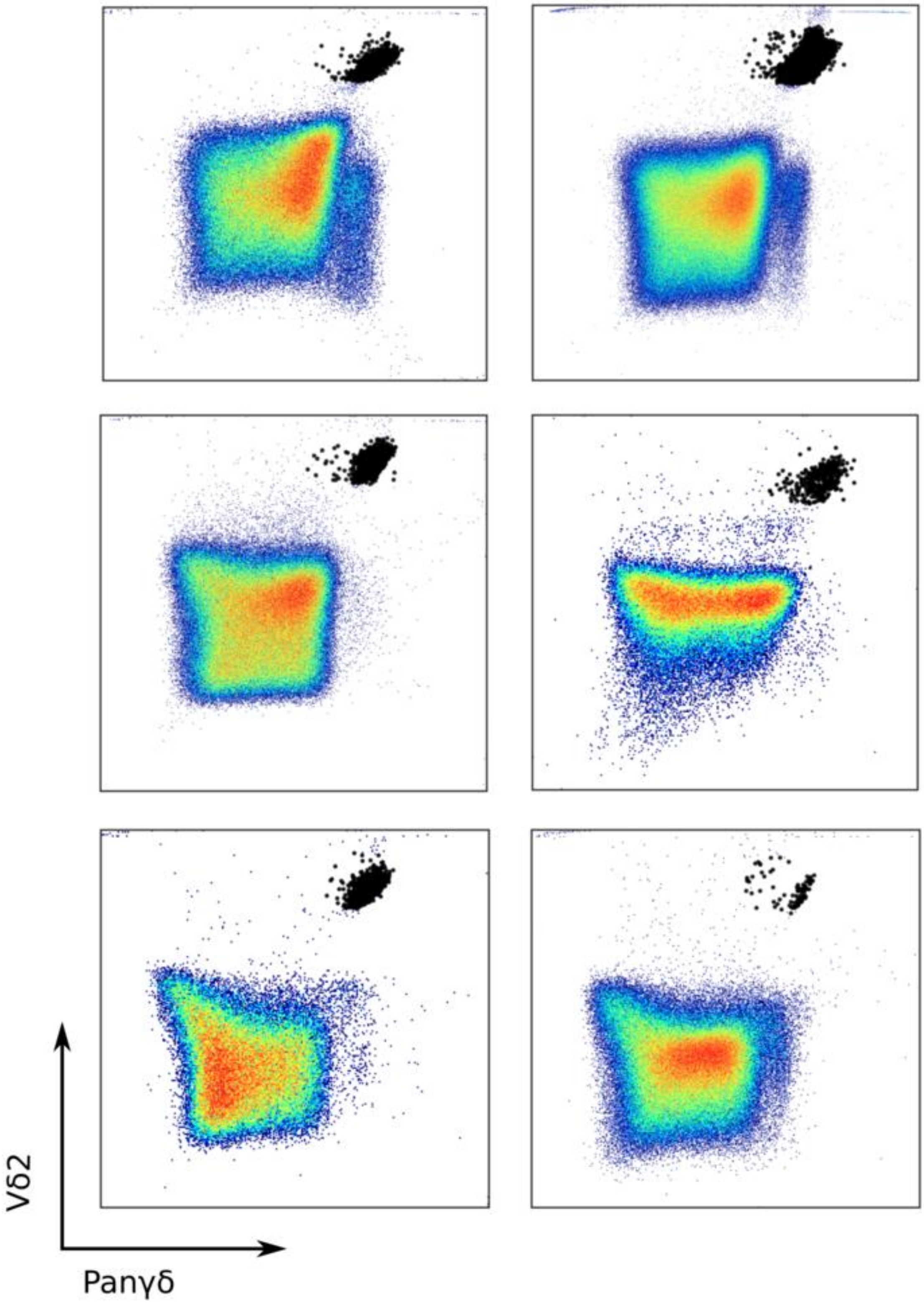
Supplementary Figure S2: Analysis of CD4$^+$ T lymphocytes from PBMC. Cells are separated based on their expression of the surface markers CD45RA and CD27, which delineate CD45RA$^+$ CD27$^+$ naïve, CD45RA$^-$ CD27$^+$ central memory and CD45RA$^-$ CD27$^-$ effector memory subsets. Examples of different algorithms available through the CytoPy's *Gate* API are shown.



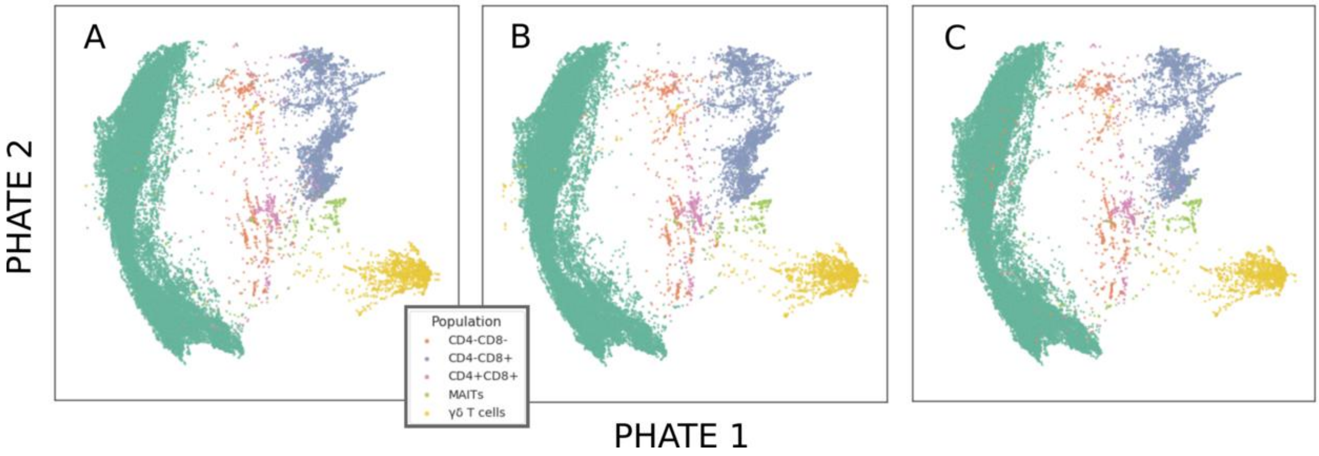
Supplementary Figure S3: Analysis of T cells from PBMC based on their expression of the surface markers CD161 and Vα7.2 for the identification of MAIT cells. Shown are data from six individual PD patients after annotation by XGBoost. Data points corresponding to those cells annotated as belonging to the CD161^{+} Vα7.2^{+} MAIT cell subset are visualised in black.



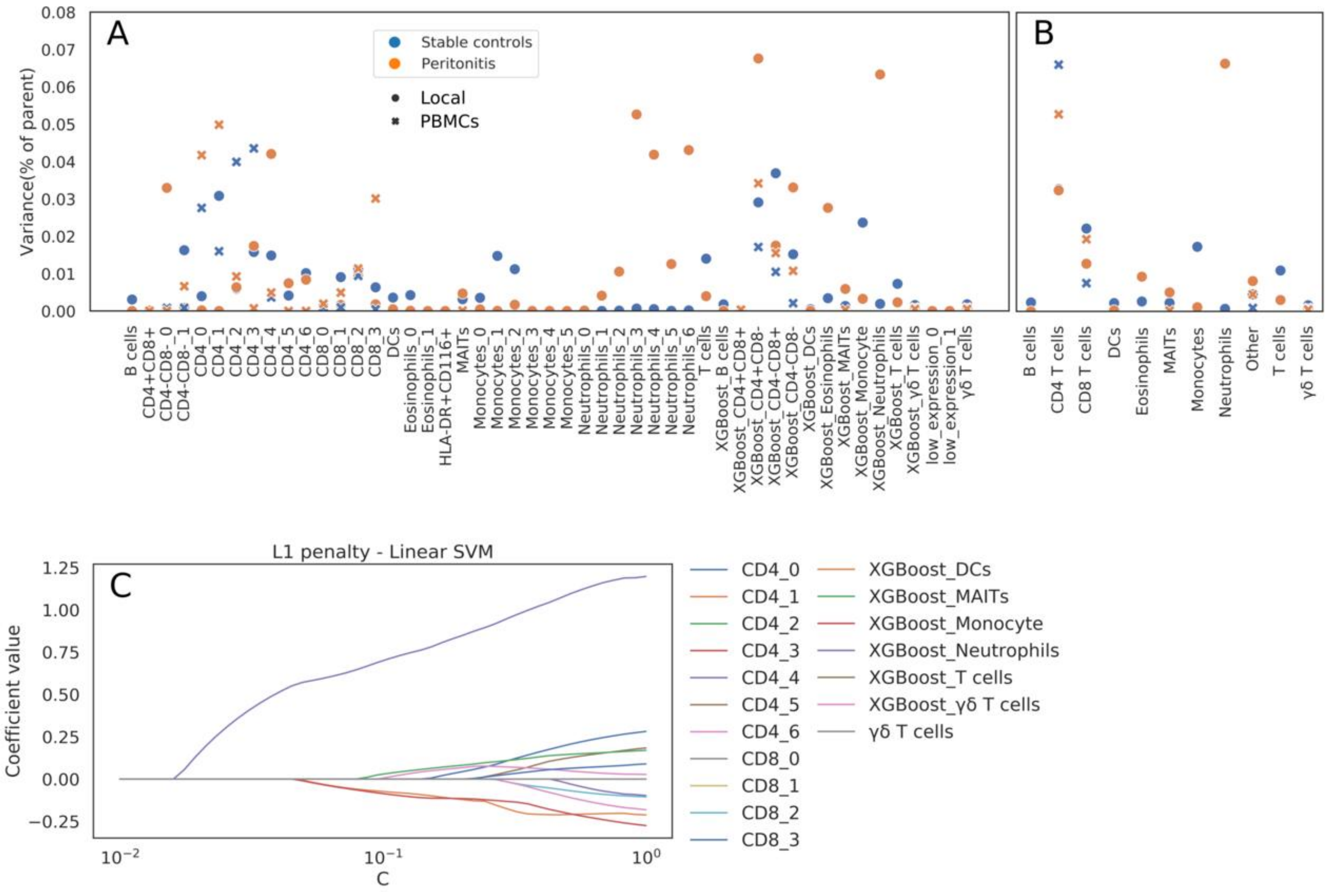
Supplementary Figure S4: Analysis of T cells from PBMC based on their expression of the surface markers pan- γδ and Vδ2 for the identification of Vδ2$^+$ γδ T cells. Shown are six separate subjects after annotation by and XGBoost classification model. Shown are data from six individual PD patients after annotation by XGBoost. Data points corresponding to those cells annotated as belonging to the Vδ2$^+$ γδ T cell subset are visualised in black.



Supplementary Figure S5: PHATE plots showing the classification of T lymphocyte subsets in whole blood by (A) manual gating, (B) XGBoost classification, and (C) PhenoGraph clustering.



Supplementary Figure S6: Visualisation of feature selection techniques. (A) Variance of population proportions for all classified populations and clusters for cells isolated from peritoneal drain fluid (local) and whole blood (PBMCs). (B) Cell populations and clusters are summarised into common compartments and variation in proportion relative to parent population shown. (C) Support Vector Machine with a linear kernel was used to classify patient phenotype. The coefficient (y-axis) associated with each variable included in the feature space of this classifier is shown as the L1 regularisation parameter (x-axis) decreases. Variables of increasing importance to accurate classification of patient phenotype will take longer to converge to 0 as the regularisation parameter decreases.



Culture result	***n***
Coagulase-negative *Staphylococcus*	4
Staphylococcus aureus	2
Streptococcus agalactiae	1
Streptococcus mitis	1
Alpha-haemolytic *Streptococcus*	1
Corynebacterium amycolatum	1
Escherichia coli	1
Pseudomonas aeruginosa	1
Yeast	1
No growth	4

Supplementary Table S1. Summary of microbiological culture results for peritoneal dialysis patients with acute peritonitis

Marker/Cytokine	Fluorochrome	Manufacturer (Clone)
CD45	Alexa Fluor 700	BioLegend (2D1)
CD14	FITC	BioLegend (63D3)
CD16	Per-CP Cy5.5	BioLegend (3G8)
CD3	APC/Fire	BioLegend (UCHT1)
Siglec-8	APC	BioLegend (7C9)
CD1c	Brilliant Violet 421	BioLegend (L161)
CD15	Brilliant Violet 605	BioLegend (SSEA-1)
HLA-DR	Brilliant Violet 711	BioLegend (L243)
CD116	PE	BioLegend (4H1)
CD19	PE-Cy7	BioLegend (HIB19)

Supplementary Table S2. Staining panel for leukocytes

Marker	Fluorochrome	Manufacturer (Clone)
CD3	APC/Fire	BioLegend (UCHT1)
CD4	PE-Cy5.5	BioLegend (OKT4)
CD8	Brilliant Violet 711	BioLegend (RPA-T8)
CD161	APC	Miltenyi Biotec (191B8)
Vα7.2	Brilliant Violet 605	Biolegend (3C10)
TCR-pan-γδ	PE-Cy5	Beckman Coulter (IM2662)
Vδ2	PE	BD Biosciences (B6 RUO)
CCR7	Brilliant Violet 421	BioLegend (G043H7)
CD27	PE-Cy7	BioLegend (M-T271)
CD45RA	PE Dazzle	BioLegend (HI100)

Supplementary Table S3. Staining panel for T lymphocytes