PLOS COMPUTATIONAL BIOLOGY

RESEARCH ARTICLE

# CytoPy: An autonomous cytometry analysis framework

**Ross J. Burton[1]*, Raya Ahmed[1], Simone M. Cuff[1], Sarah Baker[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, United Kingdom, **3** Systems Immunity Research Institute, Cardiff University, Cardiff, United Kingdom

* burtonrj@cardiff.ac.uk





**Citation:** Burton RJ, Ahmed R, Cuff SM, Baker S, Artemiou A, Eberl M (2021) CytoPy: An autonomous cytometry analysis framework. PLoS Comput Biol 17(6): e1009071. https://doi.org/10.1371/journal.pcbi.1009071

**Editor:** Manja Marz, bioinformatics, GERMANY

**Received:** April 12, 2020

**Accepted:** May 12, 2021

**Published:** June 8, 2021



**Data Availability Statement:** Details on how to download the original dataset can be found in the software documentation: https://cytopy.readthedocs.io/en/latest/.

**Funding:** This research received support from the UK Clinical Research Network (UKCRN) Study Portfolio grant number: 11838 (to M.E.), the Welsh European Funding Office's Accelerate programme grant number:PR-0013 (to M.E.), Medical Research Council (MRC) grant number: MR/N023145/1(to M.E.), Wales Kidney Research Unit (WKRU) (to M.E.), and a School of Medicine PhD Studentship (to R.J.B.). The funders had no role in

## Abstract

Cytometry analysis has seen a considerable expansion in recent years 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 new computational methodologies for handling high-dimensional single cell 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 immunology. Here we present CytoPy, a Python framework for automated analysis of cytometry data that integrates a document-based database for a data-centric and iterative analytical environment. In addition, our algorithm-agnostic design provides a platform for open-source cytometry bioinformatics in the Python ecosystem. We demonstrate the ability of CytoPy to phenotype T cell subsets in whole blood samples even in the presence of significant batch effects due to technical and user variation. The complete analytical pipeline was then used to immunophenotype the local inflammatory infiltrate in individuals with and without acute bacterial infection. CytoPy is open-source and licensed under the MIT license. CytoPy is available at https://github.com/burtonrj/CytoPy, with notebooks accompanying this manuscript (https://github.com/burtonrj/CytoPyManuscript) and software documentation at https://cytopy.readthedocs.io/.

## Author summary

Cytometry is a popular technology used to quantify biological material. In recent years, the capabilities of cytometry have expanded, resulting in ever larger datasets. In order to analyse these data, new approaches are required, giving rise to the field of cytometry bioinformatics. Despite the success of numerous algorithms and tools in this domain, widespread adoption by the scientific community has yet to be realised. Here we introduce CytoPy, a comprehensive cytometry data analysis framework deployed in Python, a beginner-friendly programming language. We validate CytoPy's ability to handle batch effects and identify immune cell populations in human blood. Subsequently, we apply CytoPy to the analysis of drain fluid from patients undergoing peritoneal dialysis and compare the

study design, data collection and analysis, decision to publish, or preparation of the manuscript.

**Competing interests:** The authors have declared that no competing interests exist.

local immune response of stable patients to those presenting with acute peritonitis. CytoPy is open-source and available online: https://cytopy.readthedocs.io/en/latest/.

This is a *PLOS Computational Biology* Software paper.

## 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 is proving 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 group events based on similarity (*e.g.* cell populations), which then form the variables (often descriptive statistics of the obtained groups) the investigator uses to test their hypothesis, or directly model the acquired multi-dimensional distribution with respect to a chosen endpoint. Classification strategies can be further subdivided: autonomous gating replicates traditional gating by applying algorithms to data in one or two dimensions (flowDensity [1], OpenCyto [2]); clustering in high-dimensional space to group events according to their individual characteristics (FlowSOM [3], Phenograph [4], Xshift [5], SPADE [6]); and supervised or semi-supervised classification where manual annotations are used to train a model capable of identifying cell populations within unlabelled data (FlowLearn [7], ACDC [8], DeepCyTof [9]). Direct modelling strategies have been successfully adopted in applications such as ACCENSE [10], CellCNN [11], CytoDX [12] and in the work described by Hu *et al.* [13]. This approach has the benefit of removing any subjectivity and can be considered as truly automated but requires the pooling of sample data and is therefore sensitive to batch effects.

In addition, various pieces of software and pipelines 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 specific applications (*e.g.* Citrus, MetaCyto, flowType/RchyOptimyx) [14]. However, widespread adoption of cytometry bioinformatics has yet to be realised and a lack of consensus remains on how to implement such technologies across the scientific community, with much of the analysis pipeline left to the individual investigator to establish. This inconsistency continues to result 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 was developed in the Python programming

language, which prides itself on readability and a beginner-friendly syntax. CytoPy introduces a central data source for all single cell data, experimental metadata and analysis results, and provides a 'low code' interface that is both powerful and easy to maintain. CytoPy incorporates popular data science and machine learning libraries such as Pandas [15], Scikit-Learn [16] and Tensorflow [17], with an application programming interface (API) designed to help expand cytometry bioinformatics in the Python ecosystem. In addition, CytoPy provides convenient access to popular algorithms and techniques popular in cytometry data analysis such as PHATE [18], UMAP [19], Phenograph [4] and FlowSOM [3].

A burgeoning challenge as cytometry data grows in size is batch effect. Nowhere is this issue more pressing than in translational research where lengthy study designs and complex specimens make technical variation unavoidable. CytoPy provides tools for visualising and quantifying batch effect as well as methods to subvert and eliminate it. To validate CytoPy we used in-house data obtained from patients undergoing peritoneal dialysis and who presented with and without acute bacterial infection. Data had been collected over several years, therefore providing ample opportunity to demonstrate the ability of CytoPy to navigate issues such as batch effects and inconsistent meta-data. We first introduce the capabilities of CytoPy by characterising T cells in blood and comparing findings to manual gates. Finally, we employ the entire pipeline to describe a known phenotype of immune cells in the peritoneal effluent of dialysis patients that differentiates individuals with acute peritonitis from stable controls. Accompanying Jupyter Notebooks demonstrating all necessary code are available at https://github.com/burtonrj/CytoPyManuscript. We believe that CytoPy provides a powerful and user-friendly framework to interrogate high-dimensional data originating from investigations using flow cytometry or mass cytometry as readout, and has the potential to facilitate automated data analysis in a multitude of experimental and clinical contexts.

## Design and implementation

### Building a framework that is algorithm-agnostic and data-centric

Reliable data management is a cornerstone of successful analysis, improving reproducibility and fostering 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 [20], where data are stored in JavaScript Object Notation (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 [21], a Document-Object Mapper based on the PyMongo driver.

### Framework overview

An overview of the CytoPy framework is given in Fig 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

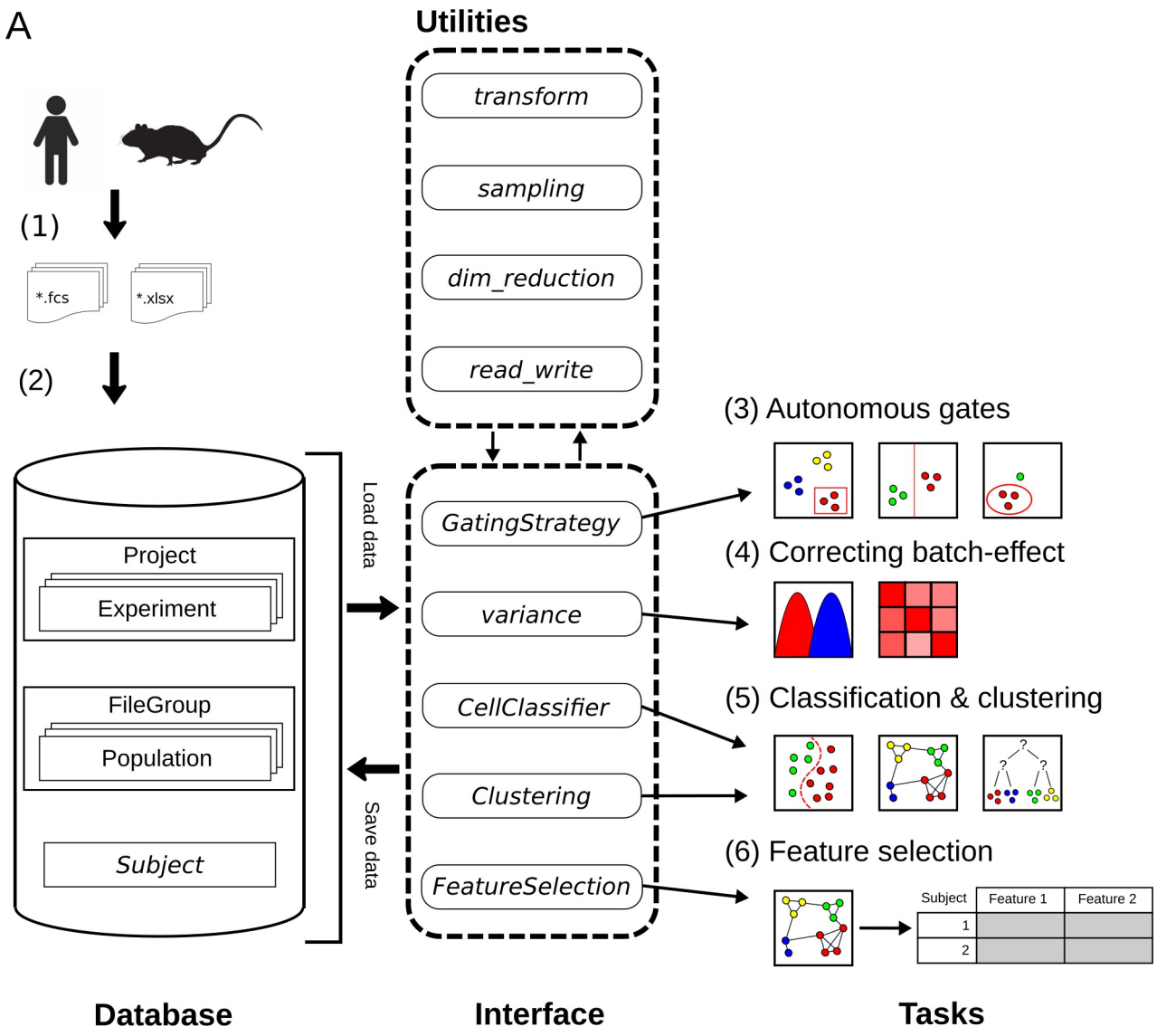


**Fig 1.** Overview of the CytoPy framework. Single cell data and experiment/clinical metadata (1) are used to populate a project within the CytoPy database (2). The CytoPy database models analytical data in MonogDB documents (cylinder), and an interface of CytoPy classes retrieves and commits data to this database (dotted rounded rectangle). Utility modules perform regular tasks such as data transformations and sampling throughout the framework. The components of this interface can be used independently, but the recommended workflow is as follows: (3) autonomous gates identify a 'clean' population of interest from where to start analysis, (4) batch effect is visualised, quantified and corrected using the Harmony algorithm, (5) supervised and unsupervised algorithms classify cells into groups of similar phenotype, and finally (6) a feature space of cell population descriptive statistics is generated and feature extraction/selection methods deployed to identify a predictive signature that characterises an endpoint of interest.

https://doi.org/10.1371/journal.pcbi.1009071.g001

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 and SciPy [22] ecosystems. 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 Fig 1, a typical analysis in CytoPy would be performed as follows (functions show in italics and class names are shown in italics and title-case):

1. Single cell data are generated and exported from the flow cytometer; CytoPy supports FCS (Flow Cytometry Standard) files version 2.0, 3.0 and 3.1, but additionally supports the introduction of data using a Pandas DataFrame object, therefore supporting wider formats, although this requires that the end user generates this object with suitable formatting. Experimental and clinical metadata are collected in tabular format either as Microsoft Excel document or Comma Separated Values (CSV) files, with the only requirement being that metadata be in 'tidy' format.

2. A *Project* is defined and populated with the single cell data and accompanying metadata. A *Project* contains one or more *Experiment* documents, each defining a set of staining conditions. 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 *FCSGroup* documents. Each *FCSGroup* document contains one or more FCS files (or *DataFrames*) associated to a single biological sample collected from the subject. This document contains all single cell data, 'gated' populations, clusters and meta-information that attains to a single 'sample', which also includes any isotype or Fluorescence-Minus-One (FMO) staining controls. Compensation is applied to single cell data at the point of entry using either an embedded spillover matrix or a provided CSV file. It should be noted that data are stored on a linear scale with a variety of transformations available during subsequent analysis; this provides flexibility in analysis as the user can compare the effects of different transformations, including the commonly used biexponential and hyperbolic arcsine transformations (compensation and transformations are implemented with the FlowUtils package [22]).

3. Any cytometry analysis will require that single cell data be cleaned of debris and artefacts. We recommend FlowAI [23] be used independently of CytoPy prior to analysis to improve the quality of data. Within CytoPy, manual or autonomous gates can be employed to identify cell populations in two-dimensional space, replicating traditional manual analysis conducted with tools such as FlowJo. We recommend autonomous gates be used for eliminating doublets, dead cells and debris, and to select a starting population for analysis; for instance, in a mixture of immune cells this could be the T cell population ($CD3^+$ live single lymphocytes). Autonomous gates are applied with the *GatingStrategy* module and cell populations are then stored within the database as *Population* documents embedded within a *FileGroup*. These *Population* documents record the index of events belonging to a population, detail how they were identified, and the conditions in which they were identified such as transformations applied to linear space, *e.g.* biexponential transformation of axis.

4. Batch effects are common and must always be addressed prior to analysis. If the batch effect is minimal the investigator can consider pooling data and modelling the distribution of single cell data directly. If batch effects are considerable, the investigator should include methods to alleviate this prior to further analysis. The *Variance* module of CytoPy provides methods to visualise and quantify batch effect. Autonomous gates provide methods to address batch effect through landmark registration, but technical variation can be addressed at a global level using the Harmony algorithm [24], implemented in CytoPy using the harmonypy package [25].

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. 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. These classes provide objects that are algorithm-agnostic, allowing for the introduction of any function with specific signatures, whilst also providing much convenient functionality for visualising and critiquing results; this includes but is not limited to, cross-validation, learning curves, heatmaps, plotting with dimension reduction and common metrics. Importantly, the results of either strategy generate common *Population* documents that are committed to the database and can then be used as input to any additional analysis or visualisations.

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 available through the *feature_selection* module finding only those variables that are important for predicting a biological or experimental endpoint. This module deploys methods from the discipline of interpretable machine learning, from simple L1-regularised linear models and decision trees, to complex modelling and interpretation through permutation feature importance and SHapley Additive exPlanations (SHAP) [26]

## Results

### Identifying significant batch effect in blood T cell subsets

To validate and exhibit the individual elements of CytoPy we decided to use the framework to identify T cells subsets in PBMCs isolated from whole blood. 14 individuals were chosen (based on availability of data) from a local study of patients undergoing peritoneal dialysis, 4 of whom presented with symptoms of acute peritonitis whereas the remainder were stable and asymptomatic (see supplementary methods). The objective was to identify T cells (single live $CD3^+$ cells) in the first instance and then subsequently identify $CD4^+$ T helper cells, $CD8^+$ cytotoxic T cells, $V\alpha7.2^+$ $CD161^+$ mucosal-associated invariant T (MAIT) and $V\delta2^+$ γδ T cell subsets. These populations were chosen to test a range of functionality: the ability to identify large and easy to distinguish cell populations ($CD4^+$ and $CD8^+$ T cells), and more complex cell types that can be rare in some patients and difficult to identify reliably in two-dimensional space ($V\delta2^+$ γδ T cells and MAIT cells). Performance was compared to manual gates decided by user expertise.

The chosen study obtained patient material over 24 months resulting in significant batch effect. S1 Fig shows the variation between individual fluorochromes, exhibiting 'drift' in the fluorescent intensity of multiple channels. Fig 2 shows UMAP plots of data from individual patients compared to a reference patient (blue); to choose a reference the pairwise Euclidean distance of a set of covariance matrices for each sample was computed and the sample with the smallest average distance to every other sample was chosen [9]. The UMAP plots revealed common structures shared between patients but a lack of alignment, suggesting the infiltration of noise from technical variation.

### Autonomous gates reliably identify T cell subsets despite batch effect

CytoPy replicates traditional manual gating using autonomous gates building on previous examples in the literature [1, 2]. The *Gate* object is used to implement a single algorithm for

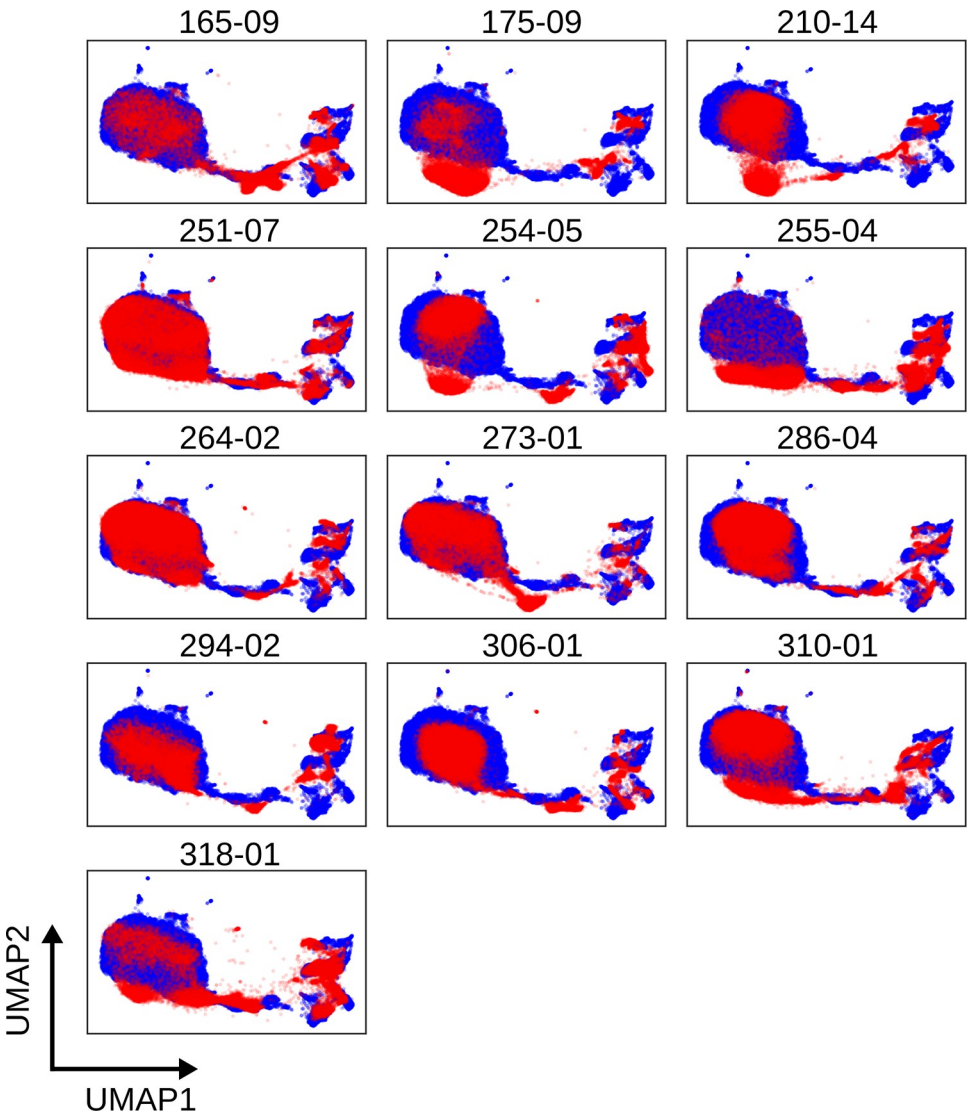


**Fig 2. UMAP plots revealing batch effect in T cell staining of whole blood.** A reference sample (blue) is chosen as the 'average' sample in Euclidean space. A low dimension embedding of this sample is made using UMAP (other algorithms are available in CytoPy, *e.g.* PCA, PHATE, t-SNE) and samples for comparison are projected into this same space (red), demonstrating 'drift' in cell populations between patient samples. Each plot depicts results obtained with cells from an individual patient; numbers shown are unique patient sample identifiers.

https://doi.org/10.1371/journal.pcbi.1009071.g002

the identification of one or more cell populations in one or two dimensions. *Gate* objects can then be 'stacked' within a *GatingStrategy*, saved to the database and applied in sequence to subsequent data. Each *Gate* is defined using some example data and an algorithm chosen that best encapsulates the population of interest. The example data that a *Gate* is defined on act as a reference to the expected populations in subsequent data. On exposure to new data, the algorithm is reapplied and the resulting populations are matched to the expected populations from the example data. Multiple algorithms are available for autonomous gates and are discussed in detail in the supplementary methods.

A challenge when defining autonomous gates is the choice of hyperparameters that will generalise beyond the chosen example data; this is further exacerbated by batch effects. CytoPy employs two techniques to overcome this issue: hyperparameter search and landmark

registration. Hyperparameter search allows the user to specify a range of hyperparameters to use when a *Gate* is applied to new data. An exhaustive search is performed across all permutations of chosen hyperparameters resulting in a set of populations. For populations defined by a polygon gate, the convex hull of each population is computed and the population with the minimum Hausdorff distance to the population in the example data is chosen. For populations defined by a positive/negative threshold in one or two dimensions, the median fluorescent intensity of each population is computed and the population with the minimum Euclidean distance to the original example population is chosen. This is repeated for each population captured by a *Gate*.

Additionally, a user can apply landmark registration at the point of application of a *Gate*. First described in the context of cytometry data by Hahne *et al.* [27], landmark registration is used to align data to a reference by finding a warping function that aligns landmarks in their estimated probability density functions (S2 Fig). Landmarks are identified as points of maximum density and grouped by a K means algorithm [27]. In CytoPy we follow the method described by Finak *et al.* [28] and perform local normalisation when a *Gate* is applied.

Whilst accounting for batch effect with hyperparameter search and landmark registration, we applied autonomous gates to identifying T cell subsets in PBMCs (S3 Fig). Fig 3 shows a comparison of the number of events identified by autonomous gates (x-axis) compared to the same population identified by manual gates (y-axis), where each data point is an individual patient. Autonomous gates showed good conformity with manual gates, even for small and difficult to distinguish populations of Vδ2$^+$ γδ T cells and MAIT cells.

## Batch effect can be addressed 'globally' using the Harmony algorithm

Despite the success of autonomous gates for the identification of T cell subsets in the wake of significant batch effect, they are heavily biased by the choice of example data when defining

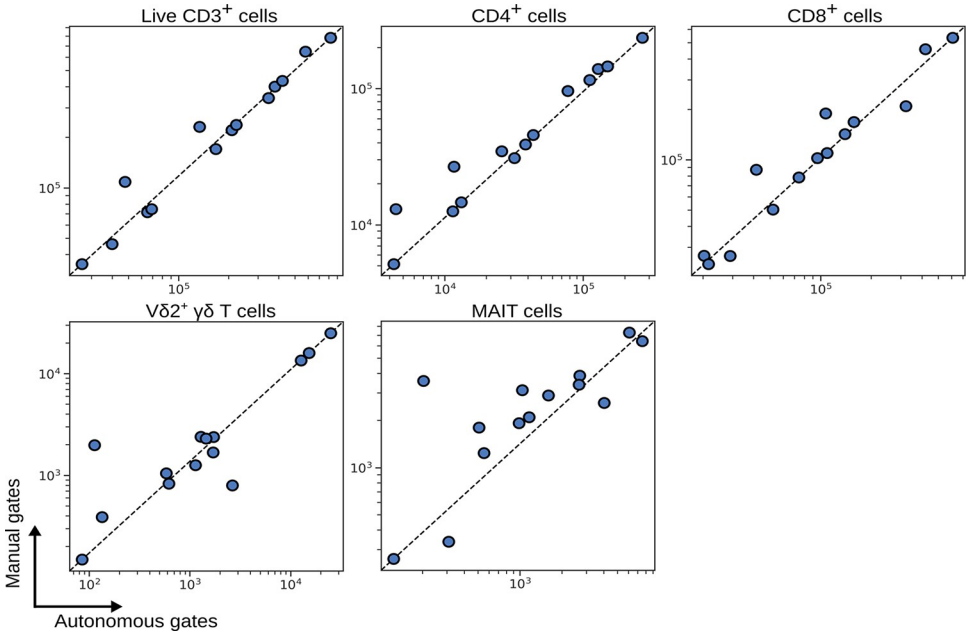


**Fig 3. Number of events captured by autonomous gates for blood T cell subsets compared to the same subsets as defined by manual expert gates.** Each symbol depicts results obtained with cells from an individual patient.

https://doi.org/10.1371/journal.pcbi.1009071.g003

*Gate* objects and by the choice of reference for landmark registration. An alternative approach to addressing batch effect is to try to align cell populations between individual subjects in high dimensional space prior to analysis. There have been several methods proposed with this objective [29], most prominently applied to single cell RNA sequencing data, although some examples such as SAUCIE [30] demonstrate application to cytometry data.

We decided to implement the Harmony algorithm [24] given its ability to scale to large data and its transparent hyperparameters. Harmony was originally described as being applied to low-dimensional embeddings. This is necessary for RNA sequence data where the number of available features can be in the thousands or tens of thousands but is not necessary for cytometry data with only a dozen or more parameters. Therefore, we exposed the original data to the Harmony algorithm, after removal of debris, doublets and dead cells. Biexponential transformation followed by scaling of each parameter to unit variance (by subtracting the mean and dividing by standard deviation) was performed prior to batch effect correction.

The performance of Harmony when applied to our T cell population (as identified by autonomous gates) from PBMCs is shown in Fig 4. Harmony has a range of hyperparameters that influence its behaviour. We found that default values for most of these parameters provide good performance but $\sigma$ should be varied to improve performance on cytometry data; this hyperparameter influences the entropy regularisation term of the soft-clustering step of the algorithm and as it approaches zero, clustering is more alike to hard K means clustering. We chose an optimal value of 0.2 for $\sigma$ whilst limiting the number of iterations to 5. The quality of batch correction is assessed by observing the distribution of the local inverse Simpson's Index

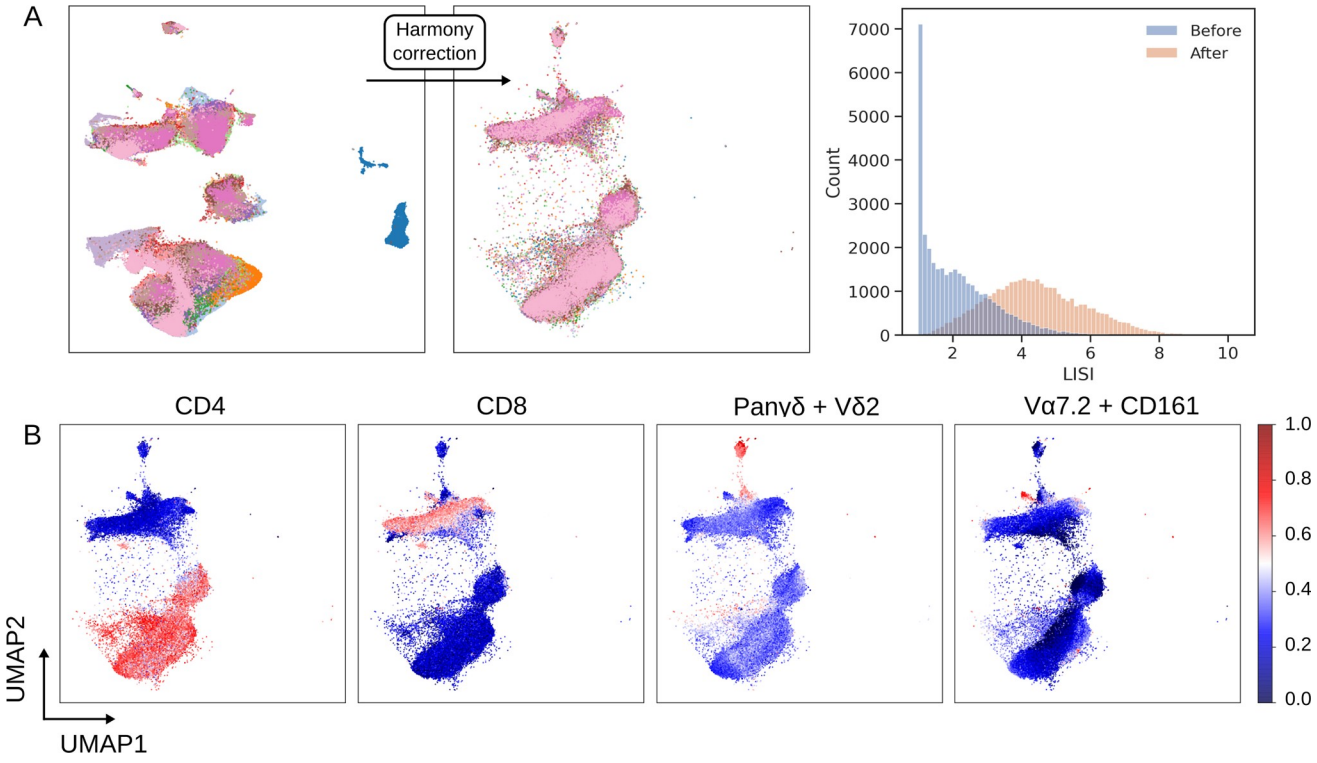


**Fig 4. Batch correction using the Harmony algorithm.** (A) Single cell UMAP plots are coloured by cell origin, where each colour represents a unique patient. Shift in batch membership in the local neighbourhood of cells is shown by the change in the UMAP plot after Harmony is applied and by the shift in LISI distribution. (B) Cell population structure is conserved after correction as shown by the shape of latent variables UMAP1 and UMAP2, and the distribution of the cell surface markers CD4, CD8, the linear combination of p Pan-γδ and Vδ2 (to identify $V\delta2^+$ γδ T cells), and the linear combination of CD161 and V7.2 (to identify MAIT cells).

https://doi.org/10.1371/journal.pcbi.1009071.g004

(LISI) and a UMAP embedding of single cell data before and after running Harmony (Fig 4A); LISI is effectively the number of batches in a cell's local neighbourhood [24]. The objective here was to redistribute LISI such that the local neighbourhood around a cell contains a greater representation of different batches, without over-correcting and distilling biological variation that differentiates groups of subjects.

The UMAP plots in Fig 4A show that prior to applying Harmony large communities of cells consist of single batches whereas after application these communities are diffused yet maintain a topology of separate cell populations. The concern with batch correction is over-correction that disrupts the biological meaning of the single cell data, but Fig 4B and 4C demonstrates that biological meaning is conserved, with distinct cell populations identified by their structure and lineage markers.

## Supervised methods can replicate the performance of autonomous gates

Considering hundreds of thousands of data points can be obtained for each subject, cytometry data lends itself well to a supervised classification approach for identifying cell populations. Supervised classification of cell populations is exposed in CytoPy through the *SklearnCellClassifier* and *KerasCellClassifier* classes, which inherit from the *CellClassifier* class. Objects of this class can accept any classifier that conforms to/supports the Scikit-Learn API (such as XGBoost) or a Keras [17] model. Many convenient methods are pre-built into these objects and predictions can be saved as *Population* objects, providing compatibility with all other tools in the CytoPy framework.

To benchmark this supervised approach, we compared four native classifiers from Scikit-Learn (logistic regression, linear discriminate analysis, support vector machine with radial kernel, and K-nearest neighbours), XGBoost [31], and a deep feed-forward neural network built with Keras to classifiers reported from the Flow Cytometry: Critical Assessment of Population Identification Methods competition (FlowCAP) [32]. Algorithms were chosen from a range of classifier families based on their popularity in the literature. S1 Table reports the weighted F1 score for each classifier across the five example datasets from FlowCAP. A deep neural network, with the architecture described by Huamin *et al.* [9], showed good performance as previously reported. XGBoost additionally showed exceptional performance, which again highlights the ability of this classifier to generalise to a wide range of use-cases.

The FlowCAP competition provides example data that have been heavily pre-processed and is not representative of data encountered in large clinical studies. We therefore decided to test the utility of XGBoost on the classification of T cell subsets. Since batch effect has been accounted for using Harmony, we pooled data from all available samples to generate training data that were manually labelled using the gating infrastructure within CytoPy. Tools for assessing the performance of a classifier such as cross-validation, learning curves, and confusion matrices are provided in CytoPy as convenient methods in *CellClassifier* (S4 Fig). As illustrated in Fig 5, XGBoost is capable of identifying T cell subsets and is comparable to manual gating. Since batch effect correction with Harmony involves a down-sampling step, comparisons are shown as the percentage of T cells as observed by manual gates vs populations identified by XGBoost.

## FlowSOM and Phenograph clustering for identifying T cell subsets after removal of batch effects

Autonomous gates and supervised classification are capable of identifying known populations of interest but are biased by the investigator's understanding and expectations of the immune

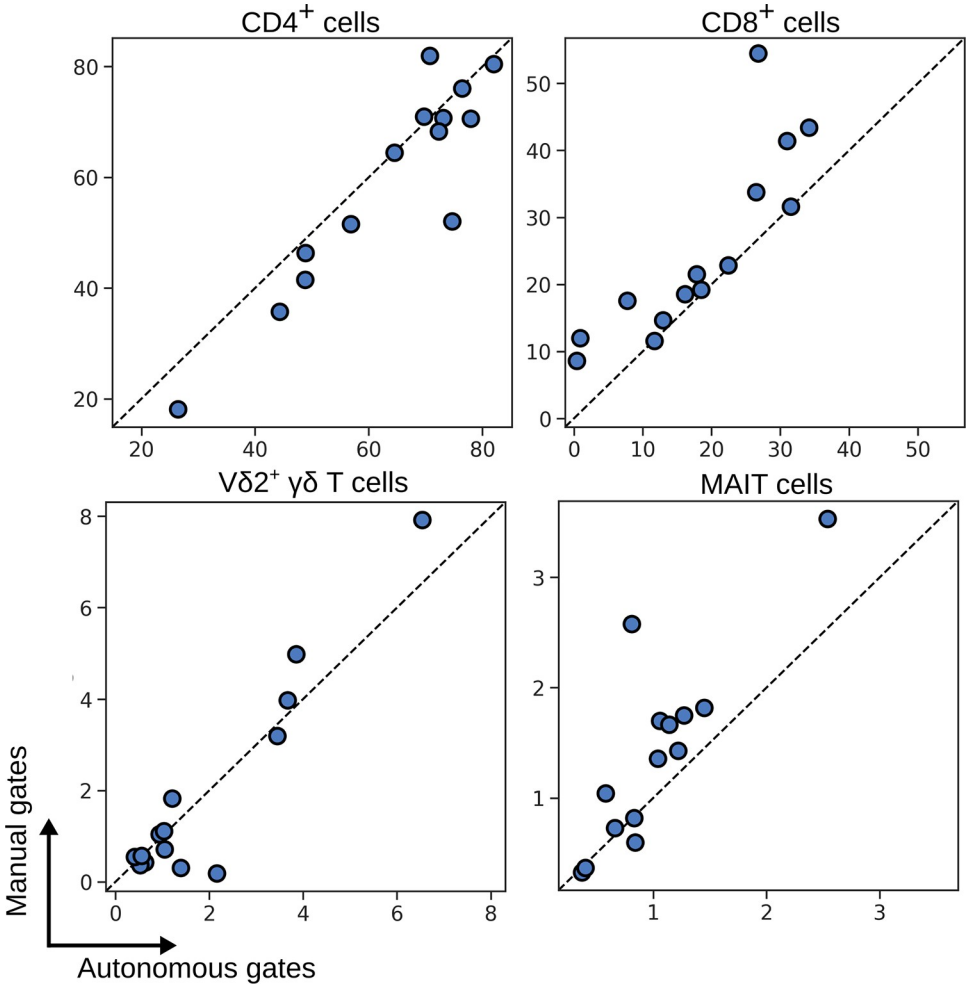


**Fig 5. Percentage of blood T cell subsets as identified by XGBoost compared to the same subsets as identified by expert manual gates.** Each symbol depicts results obtained with cells from an individual patient.

https://doi.org/10.1371/journal.pcbi.1009071.g005

landscape. To diminish this bias, CytoPy encourages the use of unsupervised techniques alongside directed analysis.

Unsupervised clustering is a popular approach to identifying structures in single cell data, with techniques such as FlowSOM and Phenograph growing in popularity in the field of cytometry data analysis. Both algorithms are available in CytoPy, along with clustering algorithms from the Scikit-Learn ecosystem and consensus clustering. Any clustering algorithm can be applied within the framework to generate *Population* objects using the *Clustering* class and a function with a common signature and expected output. This design was chosen to future-proof CytoPy against further developments in the field so that new techniques will be easy to integrate using a simple wrapper function.

Clustering of the batch effect corrected T cells was performed using both FlowSOM and Phenograph by directing the *Experiment* towards a *Clustering* object and supplying the relevant functions. Each sample within an *Experiment* was clustered independently and then inter-sample comparisons were made through meta-clustering, as originally described by Levine *et al.* [4]. The results of meta-clustering are displayed in Fig 6 and demonstrate the ability of these algorithms to discern individual cell populations. The UMAP plots show each

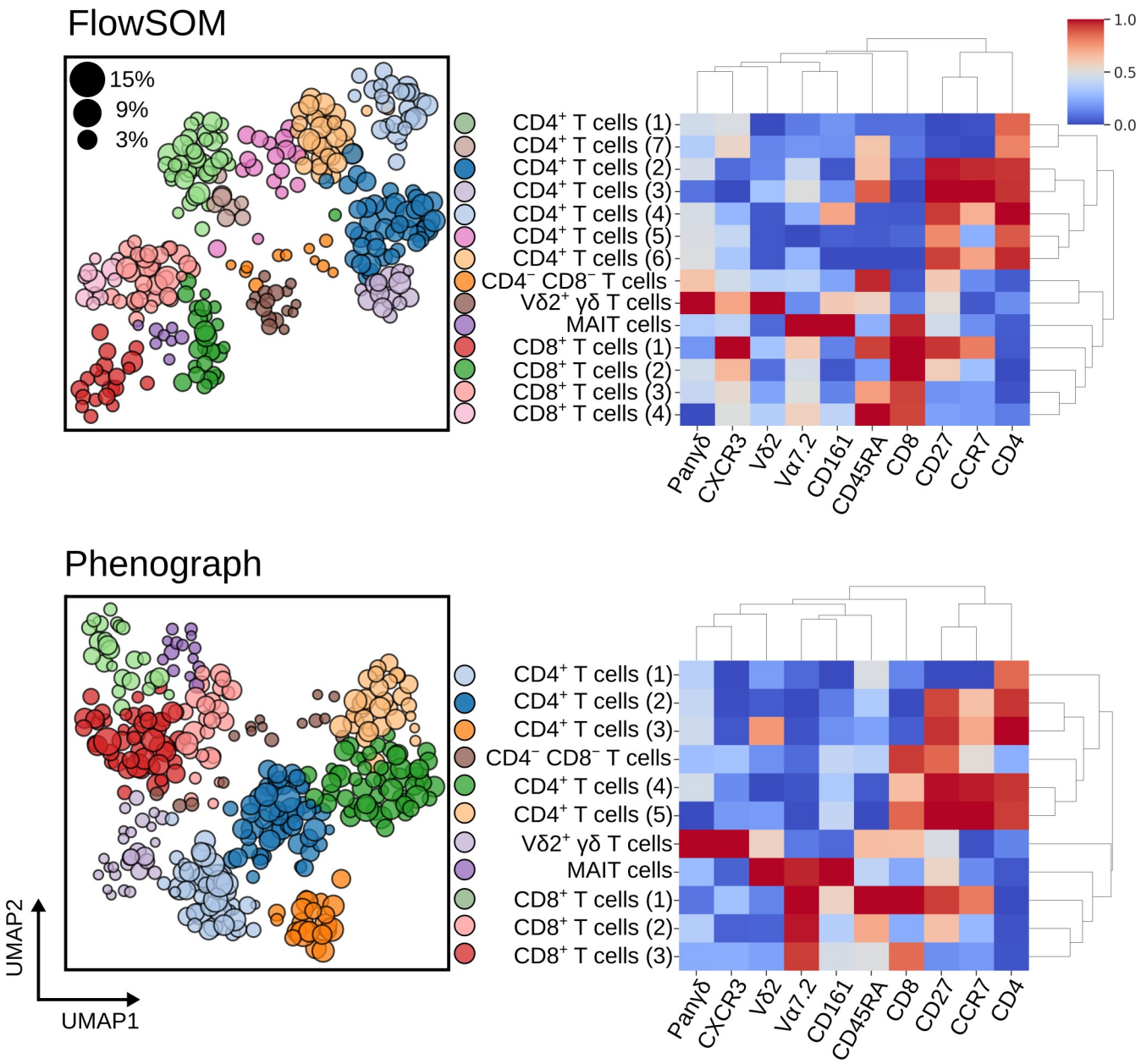


**Fig 6.** Meta-clustering results for FlowSOM (top) and Phenograph (bottom) when applied to blood T cells after batch effect correction with Harmony. Heatmaps show the normalised expression of cell surface markers for meta-clusters (clustered centroids of individually clustered patient samples). In the neighbouring UMAP plots, clusters from all patients are shown in the same embedded space and coloured by their meta-cluster membership. The size of each data point corresponds to the percentage of T cells this cluster represents in the patient it was derived from.

https://doi.org/10.1371/journal.pcbi.1009071.g006

individual cluster as obtained from individual subjects but plotted in the same two-dimensional space and coloured by meta-cluster membership; the size of the data point corresponds to the proportion of events as a percentage of T cells in each individual. Most clusters are represented by a mixture of all subjects in an *Experiment* (S5 Fig) yet rare cell populations are under-represented in FlowSOM clustering; for instance, five patients had MAIT cells absent in clustering results despite being identified by manual gates (Fig 7). A comparison of the proportion of cells obtained by FlowSOM and Phenograph to the same cell type identified by manual

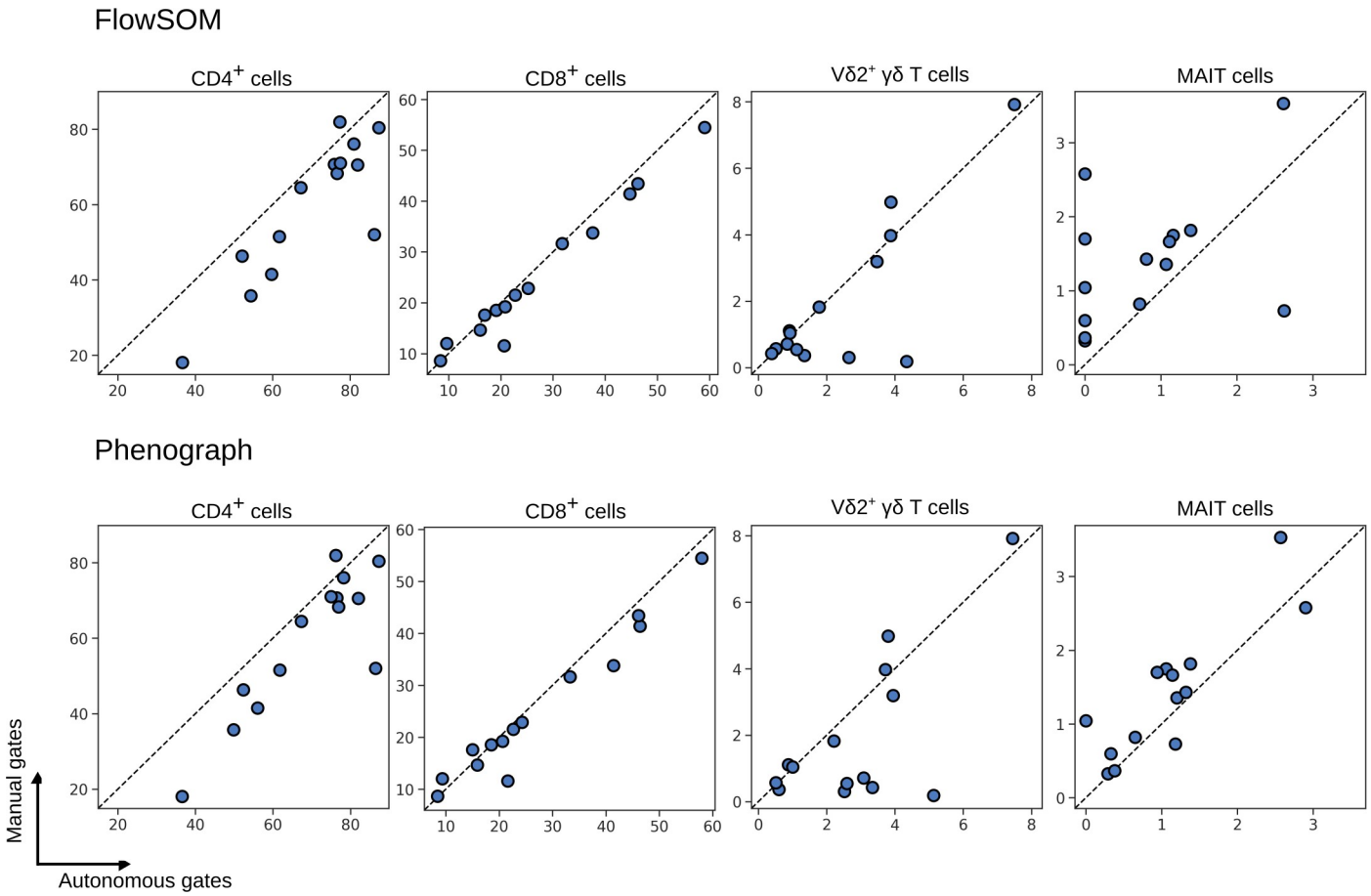


**Fig 7.** Percentage of T cell subsets as identified by FlowSOM (top) and Phenograph clustering (bottom), compared to the same subsets as identified by expert manual gates. Each symbol depicts results obtained with cells from an individual patient.

https://doi.org/10.1371/journal.pcbi.1009071.g007

gates showed that Phenograph gives preferable performance over FlowSOM. Despite this, Phenograph overestimated the proportion of Vδ2$^+$ γδ T cells in a number of patients. This highlights the importance of using multiple techniques of both supervised and unsupervised classification when investigating cytometry data, and CytoPy simplifies this process.

## Implementing the CytoPy framework to identify an immune signature that differentiates patients with acute peritonitis from stable controls

To demonstrate the application of the entire CytoPy framework to an immunophenotyping project, we investigated the peritoneal effluent of patients undergoing peritoneal dialysis, some of whom presented with symptoms of acute peritonitis, with the objective to distinguish patients with acute peritonitis from stable controls based on their peritoneal immune signatures. This was chosen based on our long-standing expertise and published findings demonstrating the significance of the local immune response in recognising pathogen-specific patterns of infection [33] and the correlation between changes in myeloid populations and treatment failure [34].

T cells from PBMCs and the CD45$^+$ fraction of cells from total effluent were obtained by autonomous gates prior to batch correction with Harmony (S6 Fig). XGBoost classification identified cell subsets using manual gates as training data displaying significant differences in

the proportion of neutrophils and monocytes between patients with acute peritonitis compared to controls (Fig 8). This was clarified in Phenograph and FlowSOM clustering (Fig 8). The proportion of T cell subsets was not significantly different between stable controls and those presenting with acute peritonitis (S7 Fig).

The ratio of cell populations observed by XGBoost classification and Phenograph, and FlowSOM clustering, conformed with one another (Fig 8). The average live $CD45^+$ fraction (for T cells, B cells, monocytes, neutrophils, eosinophils) and average live T cell fraction (for $CD4^+$, $CD8^+$, $V\delta2^+$ γδ T cells, MAIT cells) across the three classification methods were pooled

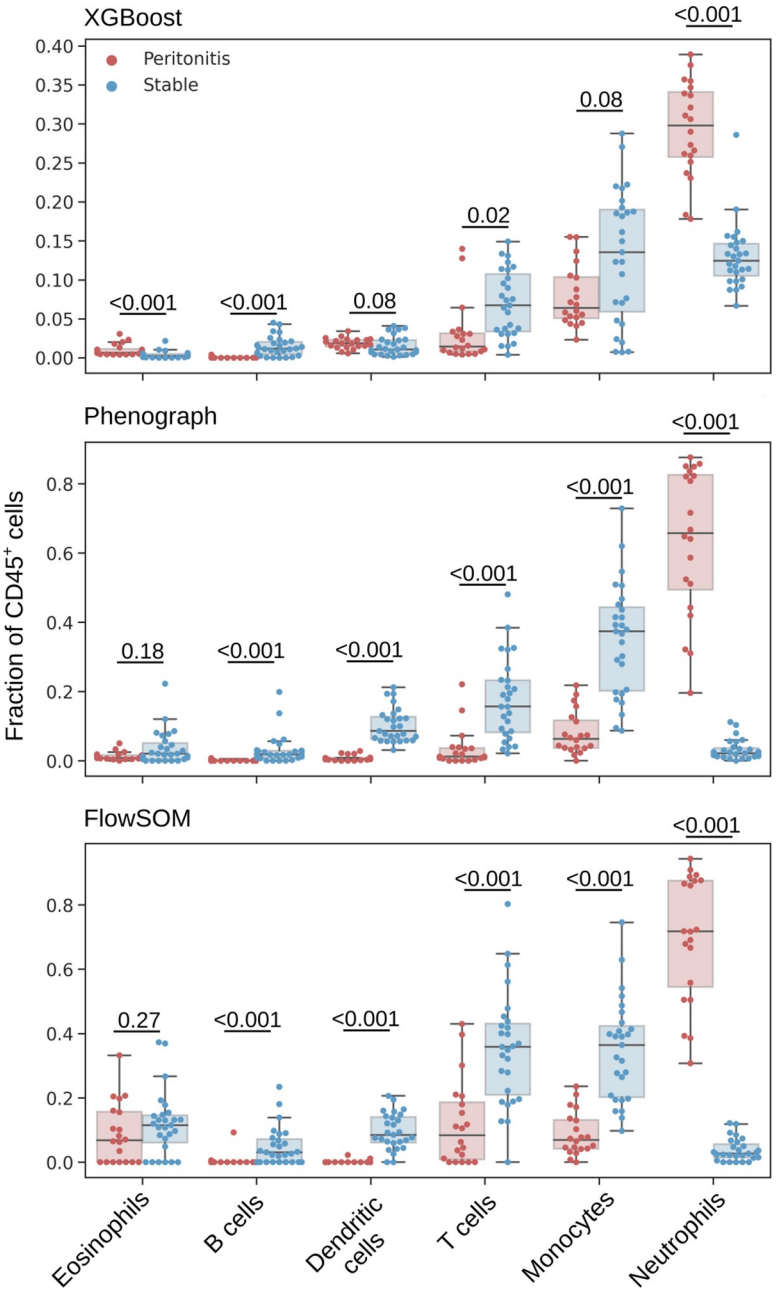


**Fig 8.** Leukocyte subsets as a fraction of $CD45^+$ cells as identified by an XGBoost classifier (top), Phenograph clustering (centre) and FlowSOM clustering (bottom). Mann-Whitney U test were applied for comparisons between patients with acute peritonitis and stable controls, and p-values are reported after correction for multiple comparisons using Holm's method (significance level was set as 0.05).

https://doi.org/10.1371/journal.pcbi.1009071.g008

using the *feature_selection* module to generate a feature space representative of the local immune profile of the peritoneum. Age and gender were included in this feature space as potential confounding variables. High collinearity was observed between the fraction of $CD4^+$ and $CD8^+$ T cells, monocytes and DCs, and T cells and B cells (Fig 9A). $CD8^+$ T cells, DCs and

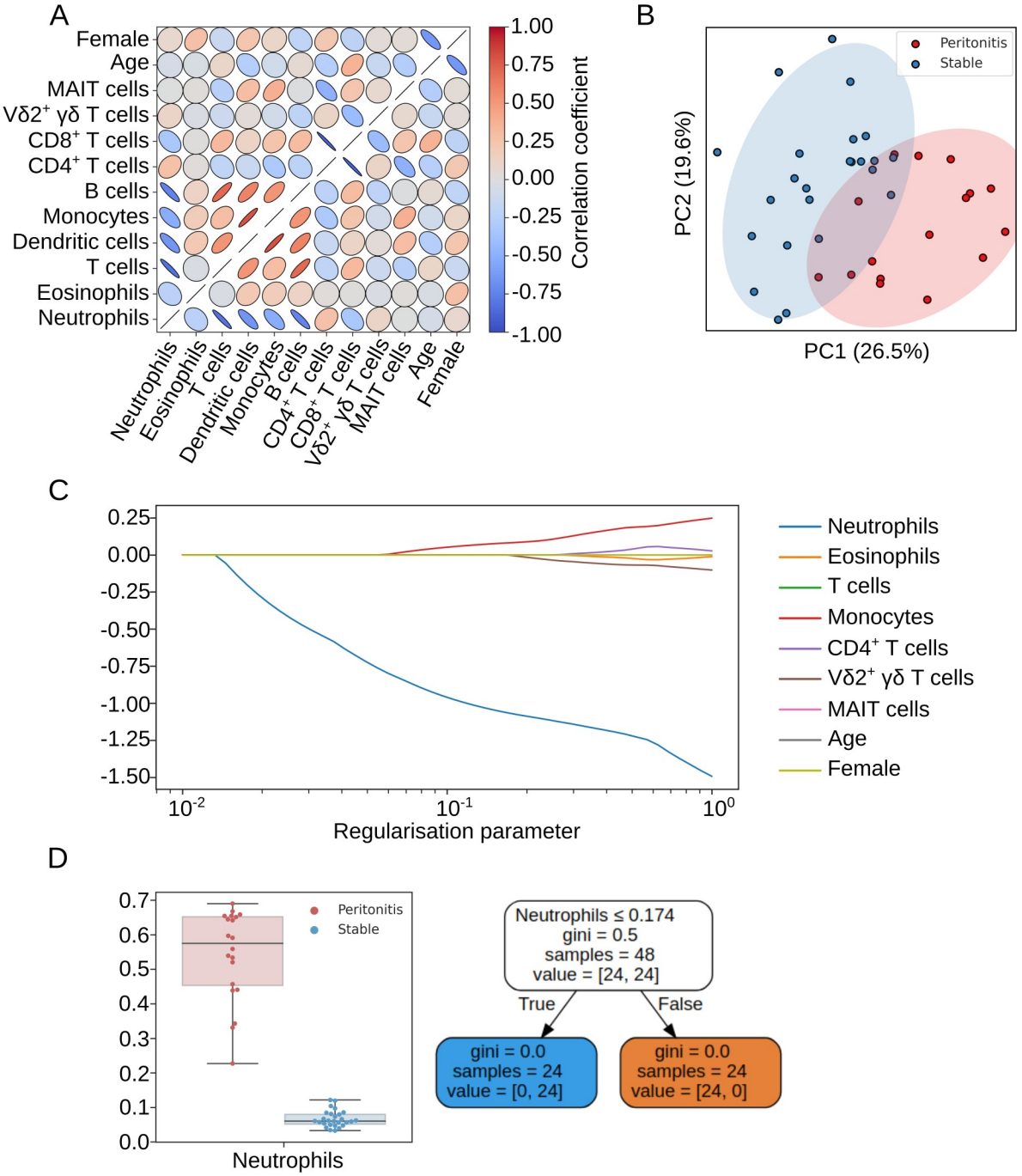


**Fig 9. Feature selection process to reduce variables for predicting acute peritonitis.** (A) Multicollinearity was addressed before generating linear models with redundant features removed prior to further analysis. (B) Principal component analysis shows that patients with acute peritonitis are discernible from stable controls. (C) L1 restricted modelling with a linear support vector machine reveals that neutrophils are the most predictive feature. (D) A simple cutoff applied to neutrophils is predictive of acute peritonitis in this cohort and is demonstrated by a shallow decision tree, where gini index is the chosen criterion for measuring the quality of split.

https://doi.org/10.1371/journal.pcbi.1009071.g009

B cells showed low variability and were therefore removed from analysis. With the remaining features principal component analysis (PCA) was performed showing that patients with acute peritonitis were highly discernible from stable controls along the axis of the first principal component (Fig 9B). The absolute value of the coefficients for this component showed that neutrophils contributed the most to the observed variation. To confirm these findings, we generated a linear support vector machine with an L1 regularisation term using the *L1Selection* class. The regularisation parameter, *C*, was varied and the coefficient of each feature plotted; as the value of *C* decreases a sparse model is encouraged, eliminating features that do not contribute to the prediction. Fig 9C demonstrates that the fraction of neutrophils is the only feature to persist in a constrained model. CytoPy's *feature_selection* module contains interpretable models for classification and regression problems, and its *DecisionTree* class can be used to demonstrate how the fraction of neutrophils alone can classify acute peritonitis (Fig 9D).

## 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 cell populations with high precision and the validation of the entire framework in identifying a known immune phenotype that distinguishes patients with acute peritonitis. This dataset was chosen based on our extensive experience with this sample type for 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 [35], we later utilised an eight colour BD FACSCanto with three lasers and FSC/SSC area/height channels [36], 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 [33], thus illustrating the technological advance in the field but also the increasing complexity of the data acquired.

CytoPy exposes multiple techniques for the classification of cell populations in cytometry data with a simplistic design and a low-code interface. Autonomous gates provide a familiar interface with cytometry data whilst reducing the labour cost of analysis, yet they are biased by the investigator's expectations of the data and computationally expensive. It is recommended that autonomous gates be employed for pre-processing and generating training data for supervised classifiers. Supervised classification offers a more efficient method for guided analysis, with training data provided in the form of a gated example. In contrast, unsupervised clustering is unbiased and offers exploratory analysis that can allude to the discovery of uncharacteristic cell populations or features that correlate with disease or experimental endpoints. In this study, we demonstrate that clustering algorithms such as FlowSOM and Phenograph were incapable of identifying rare cell populations for a small fraction of our cohort. This highlights the importance of not relying on a single method when engineering features from cytometry data. A cornerstone of CytoPy's design is to expose multiple methodologies with minimal friction and provide consistent data structures to pool results. This strategy was employed for immune phenotyping peritoneal effluent and 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 [33, 36], thereby validating the utility of CytoPy.

We have chosen to develop and maintain CytoPy in Python, a programming language with growing popularity in the bioscience domain. To date, Python has been lacking a framework for generalised cytometry data analysis offered by counterparts in R. CytoPy extends cytometry bioinformatics into the Python ecosystem by presenting an object-orientated infrastructure that is algorithm-agnostic and ready for deployment in the cloud. Compared to current

solutions in R [2, 37, 38], CytoPy boasts a low-code interface and a data-centric design that enables rapid prototyping and comparison of analytical techniques, with seamless integration of metadata. Another popular solution for cytometry data analysis is CytoBank, which whilst supporting many popular algorithms and an accessible graphical user interface, is a propriety product that could limit uptake. In contrast, CytoPy is open-source and whilst offering popular algorithms, is also designed for expansion by the open-source community; new algorithms can be introduced with very simple wrapper functions to match existing signatures and expected data types.

The current version of CytoPy offers the most popular aspects of automated analysis of cytometry data, with autonomous gating, high-dimensional clustering and supervised learning, whilst also implementing Harmony [24] for batch effect correction. Future versions of CytoPy will expand on this to include algorithms such as SAUCIE [30] and BBKNN [39]. Another paradigm of immune phenotyping for predictive modelling is multiple-instance learning, whereby the single cell data from multiple patients are exposed to a model as a single matrix, with instances labelled by the kind of patient they originate from. This was successfully demonstrated by Hu *et al.* [13] to identify a signature predictive of latent cytomegalovirus (CMV) infection and by CellCNN [11] to identify paracrine signalling, AIDS onset, and rare CMV infection-associated cell subsets. It is our ambition to extend the capabilities of CytoPy to support this design.

As high-dimensional cytometry analysis continues to grow in popularity there is increasing demand for an analytical framework that is friendly for those who are new to programming, provides a database that directly relates experimental 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). Those wishing to collaborate with us or extend our software capabilities are invited to consult the documentation (https://cytopy.readthedocs.io/) and make a pull request on our GitHub repository.

## Supporting information

**S1 Fig. Individual fluorochrome inter-sample variation amongst PBMCs from 14 patient samples. Reference patient is shown in blue with subsequent patients overlaid as individual red lines.**
(TIFF)

**S2 Fig. Example of landmark registration to align 'peaks' of high density in the probability density function (PDF) of CD4 expression on blood T cells for a target distribution and chosen reference.** Left, target (orange) PDF compared to the reference (blue) prior to alignment. Centre, warping function defined between landmarks by taking a monotone cubic interpolation. Right, registered curve with aligned peaks obtained using function composition.
(TIFF)

**S3 Fig. Autonomous gating strategy for identifying blood T cell subsets. Threshold (straight red line) gates are obtained using density-based algorithms as the ThresholdGate class, elliptical gates are obtained using the EllipseGate class and Gaussian mixture models with a varying number of components. Hyperparameter search and landmark registration was applied to all gates.**
(TIFF)

**S4 Fig.** Example of a learning curve (A) for training XGBoost for identifying T cells subsets, and confusion matrix (B) for the same algorithm when exposed to validation data.
(TIFF)

**S5 Fig. UMAP plots showing all clusters as obtained by FlowSOM (left) and Phenograph (right) and coloured according to patient origin.** The size of the data points correspond to the % of T cells the given cluster represents from the respective patient. Subject numbers shown are unique patient sample identifiers.
(TIFF)

**S6 Fig. Batch effect correction with Harmony for peritoneal effluent stained for T cell subsets (top) and leukocyte subsets (bottom).** Single cell UMAP plots are coloured to show the origin of cells where each colour is a unique patient. UMAP plots following correction and LISI distribution show the effectiveness of Harmony to correct for technical variation.
(TIFF)

**S7 Fig. T cell subsets as a fraction of T cells, identified by an XGBoost classifier (top), Phenograph clustering (centre) and FlowSOM clustering (bottom).** Mann-Whitney U tests were applied for comparisons between patients with acute peritonitis and stable controls and p-values are reported after correction for multiple comparisons using Holm's method (significance level was set as 0.05).
(TIFF)

**S1 Table. Performance of supervised classification algorithms for identifying cell populations from the FlowCAP competition data.** *Performance from the original competition as reported by Aghaeepour N *et al.* [1]; all other algorithms are implemented through CytoPy.
(DOCX)

**S2 Table. Summary of microbiological culture results for peritoneal dialysis patients with acute peritonitis.**
(DOCX)

**S3 Table. Staining panel for T cells.**
(DOCX)

**S4 Table. Staining panel for leukocytes.**
(DOCX)

**S1 Methods. Supplementary Methods.**
(DOCX)

## 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 Chantal Colmont, Donald Fraser, Alexander Greenshields-Watson, Ann Kift-Morgan, Kristin Ladell, Oliwia Michalak and John Pulford for their help and advice. We would also like to acknowledge the open-source community that has contributed to the expansion of cytometry data analysis in Python, notably Peter Brodin, Brian Teague, Scott White, and Kamil Slowikowski.

### Ethics statement

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).

## Author Contributions

**Conceptualization:** Ross J. Burton.

**Data curation:** Raya Ahmed, Simone M. Cuff, Sarah Baker.

**Formal analysis:** Ross J. Burton.

**Funding acquisition:** Matthias Eberl.

**Investigation:** Raya Ahmed, Simone M. Cuff, Sarah Baker.

**Methodology:** Ross J. Burton, Simone M. Cuff, Andreas Artemiou, Matthias Eberl.

**Project administration:** Matthias Eberl.

**Software:** Ross J. Burton.

**Supervision:** Andreas Artemiou, Matthias Eberl.

**Validation:** Ross J. Burton, Simone M. Cuff, Andreas Artemiou, Matthias Eberl.

**Visualization:** Ross J. Burton.

**Writing – original draft:** Ross J. Burton.

**Writing – review & editing:** Ross J. Burton, Raya Ahmed, Simone M. Cuff, Sarah Baker, Andreas Artemiou, Matthias Eberl.

## References

1. Malek M, Taghiyar MJ, Chong L, Finak G, Gottardo R, Brinkman RR. flowDensity: reproducing manual gating of flow cytometry data by automated density-based cell population identification. Bioinformatics. 2015; 31(4):606–7. https://doi.org/10.1093/bioinformatics/btu677 PMID: 25378466
2. Finak G, Frelinger J, Jiang W, Newell EW, Ramey J, Davis MM, et al. OpenCyto: an open source infrastructure for scalable, robust, reproducible, and automated, end-to-end flow cytometry data analysis. PLoS Comput Biol. 2014; 10(8):e1003806. https://doi.org/10.1371/journal.pcbi.1003806 PMID: 25167361
3. Van Gassen S, Callebaut B, Van Helden MJ, Lambrecht BN, Demeester P, Dhaene T, et al. FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data. Cytometry A. 2015; 87 (7):636–45. https://doi.org/10.1002/cyto.a.22625 PMID: 25573116
4. Levine JH, Simonds EF, Bendall SC, Davis KL, Amir el-AD, Tadmor MD, et al. Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis. Cell. 2015; 162(1):184–97. https://doi.org/10.1016/j.cell.2015.05.047 PMID: 26095251
5. Samusik N, Good Z, Spitzer MH, Davis KL, Nolan GP. Automated mapping of phenotype space with single-cell data. Nat Methods. 2016; 13(6):493–6. https://doi.org/10.1038/nmeth.3863 PMID: 27183440
6. Qiu P, Simonds EF, Bendall SC, Gibbs KD, Bruggner RV, Linderman MD, et al. Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE. Nat Biotechnol. 2011; 29(10):886–91. https://doi.org/10.1038/nbt.1991 PMID: 21964415
7. Lux M, Brinkman RR, Chauve C, Laing A, Lorenc A, Abeler-Dörner L, et al. flowLearn: fast and precise identification and quality checking of cell populations in flow cytometry. Bioinformatics. 2018; 34 (13):2245–53. https://doi.org/10.1093/bioinformatics/bty082 PMID: 29462241
8. Lee HC, Kosoy R, Becker CE, Dudley JT, Kidd BA. Automated cell type discovery and classification through knowledge transfer. Bioinformatics. 2017; 33(11):1689–95. https://doi.org/10.1093/bioinformatics/btx054 PMID: 28158442
9. Li H, Shaham U, Stanton KP, Yao Y, Montgomery RR, Kluger Y. Gating mass cytometry data by deep learning. Bioinformatics. 2017; 33(21):3423–30. https://doi.org/10.1093/bioinformatics/btx448 PMID: 29036374

10. Shekhar K, Brodin P, Davis MM, Chakraborty AK. Automatic Classification of Cellular Expression by Nonlinear Stochastic Embedding (ACCENSE). Proc Natl Acad Sci U S A. 2014; 111(1):202–7. https://doi.org/10.1073/pnas.1321405111 PMID: 24344260

11. Arvaniti E, Claassen M. Sensitive detection of rare disease-associated cell subsets via representation learning. Nat Commun. 2017; 8:14825. https://doi.org/10.1038/ncomms14825 PMID: 28382969

12. Hu Z, Glicksberg BS, Butte AJ. Robust prediction of clinical outcomes using cytometry data. Bioinformatics. 2019; 35(7):1197–203. https://doi.org/10.1093/bioinformatics/bty768 PMID: 30169745

13. Hu Z, Tang A, Singh J, Bhattacharya S, Butte AJ. A robust and interpretable end-to-end deep learning model for cytometry data. Proc Natl Acad Sci U S A. 2020; 117(35):21373–80. https://doi.org/10.1073/pnas.2003026117 PMID: 32801215

14. Montante S, Brinkman RR. Flow cytometry data analysis: Recent tools and algorithms. Int J Lab Hematol. 2019; 41 Suppl 1:56–62. https://doi.org/10.1111/ijlh.13016 PMID: 31069980

15. McKinney W. Data Structures for Statistical Computing in Python. SciPy. 2010. p. 56–61.

16. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. J Mach Learn Res. 2011; 12:2825–30.

17. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, et al. TensorFlow: A system for large-scale machine learning. Proc 12th USENIX Symp Oper Syst Des Implementation, OSDI 2016. 2016;265–83.

18. Moon KR, van Dijk D, Wang Z, Gigante S, Burkhardt DB, Chen WS, et al. Visualizing structure and transitions in high-dimensional biological data. Nat Biotechnol [Internet]. 2019; 37(12):1482–92. Available from: https://doi.org/10.1038/s41587-019-0336-3 PMID: 31796933

19. Becht E, McInnes L, Healy J, Dutertre CA, Kwok IWH, Ng LG, et al. Dimensionality reduction for visualizing single-cell data using UMAP. Nat Biotechnol. 2019; 37(1):38–47.

20. MongoDB [Internet]. Available from: https://www.mongodb.com/

21. Harry M, Stefan W. Mongoengine. 2020; Available from: https://github.com/MongoEngine/mongoengine

22. Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, et al. SciPy 1.0: fundamental algorithms for scientific computing in Python. Nat Methods. 2020; 17(3):261–72. https://doi.org/10.1038/s41592-019-0686-2 PMID: 32015543

23. Monaco G, Chen H, Poidinger M, Chen J, De Magalhães JP, Larbi A. FlowAI: Automatic and interactive anomaly discerning tools for flow cytometry data. Bioinformatics. 2016; 32(16):2473–80. https://doi.org/10.1093/bioinformatics/btw191 PMID: 27153628

24. Korsunsky I, Millard N, Fan J, Slowikowski K, Zhang F, Wei K, et al. Fast, sensitive and accurate integration of single-cell data with Harmony. Nat Methods 2019; 16(12):1289–96 https://doi.org/10.1038/s41592-019-0619-0 PMID: 31740819

25. Slowikowski K. Harmonypy [Internet]. Zenodo; 2020. Available from: http://doi.org/10.5281/zenodo.4531401

26. Lundberg SM, Lee SI. A unified approach to interpreting model predictions. Adv Neural Inf Process Syst. 2017; 2017-Decem(Section 2):4766–75.

27. Hahne F, Khodabakhshi AH, Bashashati A, Wong CJ, Gascoyne RD, Weng AP, et al. Per-channel basis normalization methods for flow cytometry data. Cytom Part A. 2010; 77(2):121–31. https://doi.org/10.1002/cyto.a.20823 PMID: 19899135

28. Finak G, Jiang W, Krouse K, Wei C, Sanz I, Phippard D, et al. High-throughput flow cytometry data normalization for clinical trials. Cytom Part A. 2014; 85(3):277–86. https://doi.org/10.1002/cyto.a.22433 PMID: 24382714

29. Thi H, Tran N, Ang KS, Chevrier M, Zhang X, Yee N, et al. A benchmark of batch-effect correction methods for single-cell RNA sequencing data. Genome Biol. 2020;1–32.

30. Amodio M, van Dijk D, Srinivasan K, Chen WS, Mohsen H, Moon KR, et al. Exploring single-cell data with deep multitasking neural networks. Nat Methods [Internet]. 2019; 16(11):1139–45. Available from: https://doi.org/10.1038/s41592-019-0576-7 PMID: 31591579

31. Chen T, Guestrin C. XGBoost: A scalable tree boosting system. Proc ACM SIGKDD Int Conf Knowl Discov Data Min. 2016;13-17-Augu:785–94.

32. Aghaeepour N, Chattopadhyay P, Chikina M, Dhaene T, Van Gassen S, Kursa M, et al. A benchmark for evaluation of algorithms for identification of cellular correlates of clinical outcomes. Cytom Part A. 2016; 89(1):16–21. https://doi.org/10.1002/cyto.a.22732 PMID: 26447924

33. Zhang J, Friberg IM, Kift-Morgan A, Parekh G, Morgan MP, Liuzzi AR, et al. Machine-learning algorithms define pathogen-specific local immune fingerprints in peritoneal dialysis patients with bacterial infections. Kidney Int. 2017; 92(1):179–91. https://doi.org/10.1016/j.kint.2017.01.017 PMID: 28318629

34. Liao C Te, Andrews R, Wallace LE, Khan MWA, Kift-Morgan A, Topley N, et al. Peritoneal macrophage heterogeneity is associated with different peritoneal dialysis outcomes. Kidney Int. 2017; 91(5):1088–103. https://doi.org/10.1016/j.kint.2016.10.030 PMID: 28065517

35. Eberl M, Roberts GW, Meuter S, Williams JD, Topley N, Moser B. A rapid crosstalk of human γδ T cells and monocytes drives the acute inflammation in bacterial infections. PLoS Pathog. 2009; 5(2). https://doi.org/10.1371/journal.ppat.1000308 PMID: 19229322

36. Lin CY, Roberts GW, Kift-Morgan A, Donovan KL, Topley N, Eberl M. Pathogen-specific local immune fingerprints diagnose bacterial infection in peritoneal dialysis patients. J Am Soc Nephrol. 2013; 24 (12):2002–9. https://doi.org/10.1681/ASN.2013040332 PMID: 24179164

37. Chen H, Lau MC, Wong MT, Newell EW, Poidinger M, Chen J. Cytofkit: A Bioconductor Package for an Integrated Mass Cytometry Data Analysis Pipeline. PLoS Comput Biol. 2016; 12(9):1–17. https://doi.org/10.1371/journal.pcbi.1005112 PMID: 27662185

38. Lun ATL, Richard AC, Marioni JC. Testing for differential abundance in mass cytometry data. Nat Methods 2017; 14(7):707–9 https://doi.org/10.1038/nmeth.4295 PMID: 28504682

39. Polański K, Young MD, Miao Z, Meyer KB, Teichmann SA, Park JE. BBKNN: Fast batch alignment of single cell transcriptomes. Bioinformatics. 2020; 36(3):964–5. https://doi.org/10.1093/bioinformatics/btz625 PMID: 31400197