A history of mass cytometry data analysis, and where the field is going

Tyler J Burns, PhD

AG Mei, DRFZ

Outline

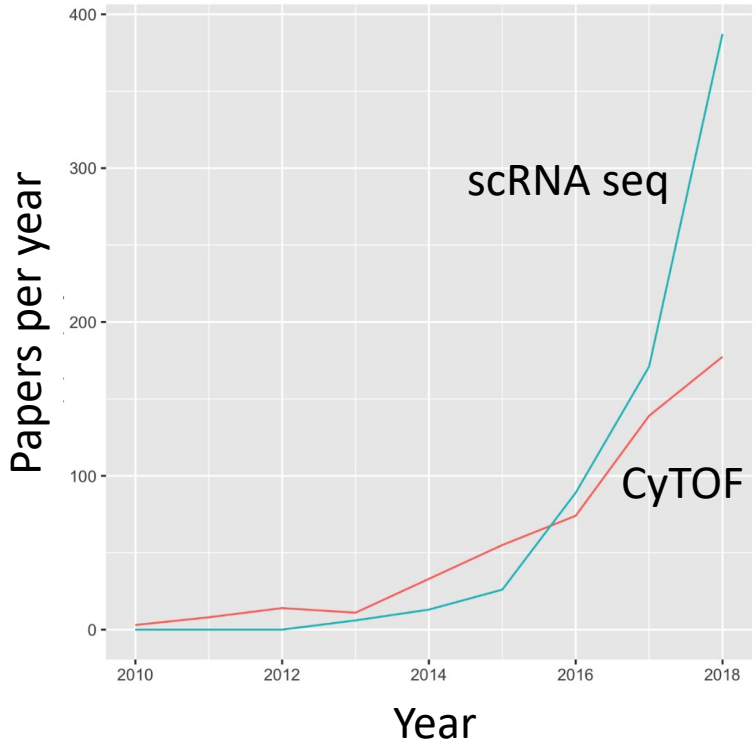
- Part 1: History of CyTOF analysis
- Part 2: How to develop a robust analytical pipeline

Outline

- Part 1: History of CyTOF analysis
- Part 2: How to develop a robust analytical pipeline

Single cell analysis is complex

There are lots of papers



Burns, 2018

There are lots of algorithms

Table 1 Overview of the trajectory inference methods included in this study, and several characteristics thereof. This table will be continuously updated online.

Method	Date	Most complex trajectory type	Fixes topology	Prior required	Prior optional	Evaluated	Reference
Monocle ICA	01/04/2014	Tree	Parameter	# branches	None	Yes	[13]
Wanderlust	24/04/2014	Linear	Fixed	Start cell(s)	None	Yes	[14]
SCUBA	30/12/2014	Tree	Free	None	Time course, Marker genes	Yes	[15]
Sincell	27/01/2015	Tree	Free	None	None	Yes	[16]
NBOR	08/06/2015	Linear	TBD	TBD	TBD	No[e]	[6]
Waterfall	03/09/2015	Linear	Fixed	None	None	Yes	[17]
ggnetboltime	15/09/2015	Linear	TBD	TBD	TBD	No[f]	[18]
Embeddr	18/09/2015	Linear	Fixed	None	None	Yes	[19]
ECLAIR	12/01/2016	Tree	TBD	TBD	TBD	No[f]	[20]
DPT	08/02/2016	Bifurcation	Fixed	None	Marker genes	Yes	[21]
Pseudogp	05/04/2016	Linear	Fixed	None	None	Yes	[22]
SLICER	09/04/2016	Graph	Free	Start cell(s)	End cell(s), Marker genes	Yes	[23]
SCell	19/04/2016	Linear	TBD	TBD	TBD	No[g]	[24]
Wishbone	02/05/2016	Bifurcation	Parameter	Start cell(s), # end states	Marker genes	Yes	[25]
TSCAN	13/05/2016	Tree	Free	None	None	Yes	[26]
SCOUP	08/06/2016	Multifurcation	Parameter	Start cell(s), Cell grouping, # end states	None	Yes	[27]
DeLorean	17/06/2016	Linear	TBD	TBD	TBD	No[g]	[28]
StemID	21/06/2016	Tree	Free	None	None	Yes	[29]
Ouija	23/06/2016	Linear	Fixed	Marker genes	None	Yes	[30]
Mpath	30/06/2016	Tree	Free	Cell grouping	None	Yes	[31]
cellTree	13/08/2016	Tree	Free	None	Cell grouping	Yes	[32]
WaveCrest	17/08/2016	Linear	TBD	Time course	None	No[f]	[33]
SCIMITAR	04/10/2016	Linear	Fixed	None	None	Yes	[34]
SCORPIUS	07/10/2016	Linear	Fixed	None	None	Yes	[35]
SCENT	30/10/2016	Linear	TBD	TBD	TBD	No[f]	[36]
k-branches	15/12/2016	Tree	TBD	TBD	TBD	No[h]	[37]
SLICE	19/12/2016	Tree	Free	None	Cell grouping, Marker genes	Yes	[38]
Topslam	13/02/2017	Linear	Fixed	Start cell(s)	None	Yes	[39]
Monocle DDRTree	21/02/2017	Tree	Free	None	# end states	Yes	[40]
Granatum	22/02/2017	Tree	TBD	TBD	TBD	No[g]	[41]
GPfates	03/03/2017	Multifurcation	Parameter	# end states	None	Yes	[42]
MFA	15/03/2017	Multifurcation	Parameter	# end states	None	Yes	[43]
PHATE	24/03/2017	Tree	TBD	TBD	TBD	No[h]	[44]
TASIC	04/04/2017	Tree	TBD	TBD	TBD	No[g,e]	[45]
SCMSC	05/04/2017	Tree	TBD	TBD	TBD	No[g]	[46]
Slingshot	19/04/2017	Tree	Free	None	Start cell(s), End cell(s)	Yes	[47]
scTDA	01/05/2017	Linear	TBD	TBD	TBD	No[f]	[48]
UNCURL	31/05/2017	Linear	TBD	TBD	TBD	No[f]	[49]
reCAT	19/06/2017	Cycle	Fixed	None	None	Yes	[50]
FORKS	20/06/2017	Tree	TBD	Start cell(s)	None	No[f]	[51]
MATCHER	24/06/2017	Linear	TBD	TBD	TBD	No[f]	[52]
PhenoPath	06/07/2017	Linear	Fixed	None	None	Yes	[53]
HopLand	12/07/2017	Linear	TBD	TBD	TBD	No[h]	[54]
SoptSC	26/07/2017	Linear	TBD	Start cell(s)	None	No[h]	[55]
PBA	30/07/2017	Multifurcation	TBD	TBD	TBD	No[f]	[56]
BGP	01/08/2017	Bifurcation	TBD	TBD	TBD	No[f]	[57]
scanpy	09/08/2017	Bifurcation	TBD	TBD	TBD	No[f]	[58]
B-RGPs	01/09/2017	Acyclic graph	TBD	TBD	TBD	No[f]	[59]
WADDINGTON-OT	27/09/2017	Graph	TBD	TBD	TBD	No[g,i]	[60]
AGA	27/10/2017	Disconnected graph	TBD	TBD	TBD	No[f]	[61]
GPseudoRank	30/10/2017	Linear	TBD	TBD	TBD	No[h]	[62]
p-Creode	15/11/2017	Tree	TBD	TBD	TBD	No[f]	[63]
iCpSc	30/11/2017	Linear	TBD	TBD	TBD	No[g]	[64]
GrandPrix	03/12/2017	Multifurcation	TBD	Time course	None	No[f]	[65]
Topographer	21/01/2018	Tree	TBD	None	Start cell(s)	No[f]	[66]
CALISTA	31/01/2018	Graph	TBD	None	None	No[f]	[67]
scEpath	05/02/2018	Tree	TBD	TBD	TBD	No[g]	[68]
MERLoT	08/02/2018	Tree	TBD	TBD	TBD	No[f]	[69]
ElPiGraph.R	04/03/2018	Graph	TBD	TBD	TBD	No[f]	

[e] Not free
[h] Unavailable

Salens, 2018

There are lots of data types

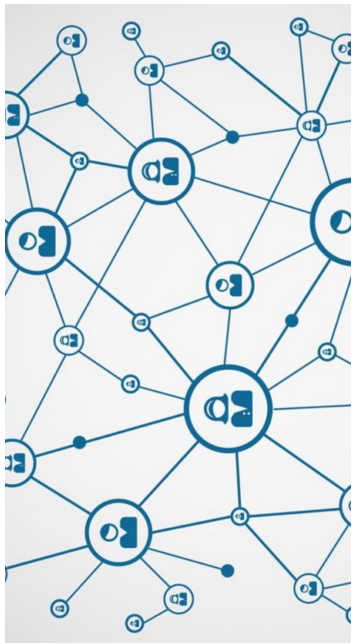
Table 1 | **Current experimental methods for unimodal and multimodal single-cell measurements**

Data types	Method name	Feature throughput	Cell throughput	Refs
Unimodal				
mRNA	Drop-seq	Whole transcriptome	1,000–10,000	4
	InDrop	Whole transcriptome	1,000–10,000	5
	10X Genomics	Whole transcriptome	1,000–10,000	6
	Smart-seq2	Whole transcriptome	100–300	38
	MARS-seq	Whole transcriptome	100–300	3
	CEL-seq	Whole transcriptome	100–300	1
	SPLiT-seq	Whole transcriptome	≥ 50,000	8
	sci-RNA-seq	Whole transcriptome	≥ 50,000	7
Genome sequence	SNS	Whole genome	10–100	9
	SCI-seq	Whole genome	10,000–20,000	10
Chromatin accessibility	scATAC-seq	Whole genome	1,000–2,000	11
	sciATAC-seq	Whole genome	10,000–20,000	14
	scTHS-seq	Whole genome	10,000–20,000	15
DNA methylation	scBS-seq	Whole genome	5–20	17
	snmC-seq	Whole genome	1,000–5,000	16
	sci-MET	Whole genome	1,000–5,000	19
	scRRBS	Reduced representation genome	1–10	18
Histone modifications	scChIP–seq	Whole genome + single modification	1,000–10,000	24
Chromosome conformation	scHi-C-seq	Whole genome	1–10	26
Multimodal				
Histone modifications + spatial	NA	Single locus + single modification	10–100	23
mRNA + lineage	scGESTALT	Whole transcriptome	1,000–10,000	32
	ScarTrace	Whole transcriptome	1,000–10,000	33
	LINNAEUS	Whole transcriptome	1,000–10,000	34
Lineage + spatial	MEMOIR	NA	10–100	27
mRNA + spatial	osmFISH	10–50 RNAs	1,000–5,000	35
	STARmap	20–1,000 RNAs	100–30,000	31
	MERFISH	100–1,000 RNAs	100–40,000	108
	seqFish	125–250 RNAs	100–20,000	29
mRNA + cell surface protein	CITE-seq	Whole transcriptome + proteins	1,000–10,000	20
	REAP-seq	Whole transcriptome + proteins	1,000–10,000	21
mRNA + chromatin accessibility	sci-CAR	Whole transcriptome + whole genome	1,000–20,000	48
mRNA + DNA methylation	scM&T-seq	Whole genome	50–100	46
mRNA + genomic DNA	G&T-seq	Whole genome + whole transcriptome	50–200	44
mRNA + intracellular protein	NA	96 mRNAs + 38 proteins	50–100	50
		82 mRNAs + 75 proteins	50–200	49
DNA methylation + chromatin accessibility	scNOMe-seq	Whole genome	10–20	13

CEL-seq, cell expression by linear amplification and sequencing; CITE-seq, cellular indexing of transcriptomes and epitopes by sequencing; G&T-seq, genome

Stuart, Satija, 2018

There are social complexities



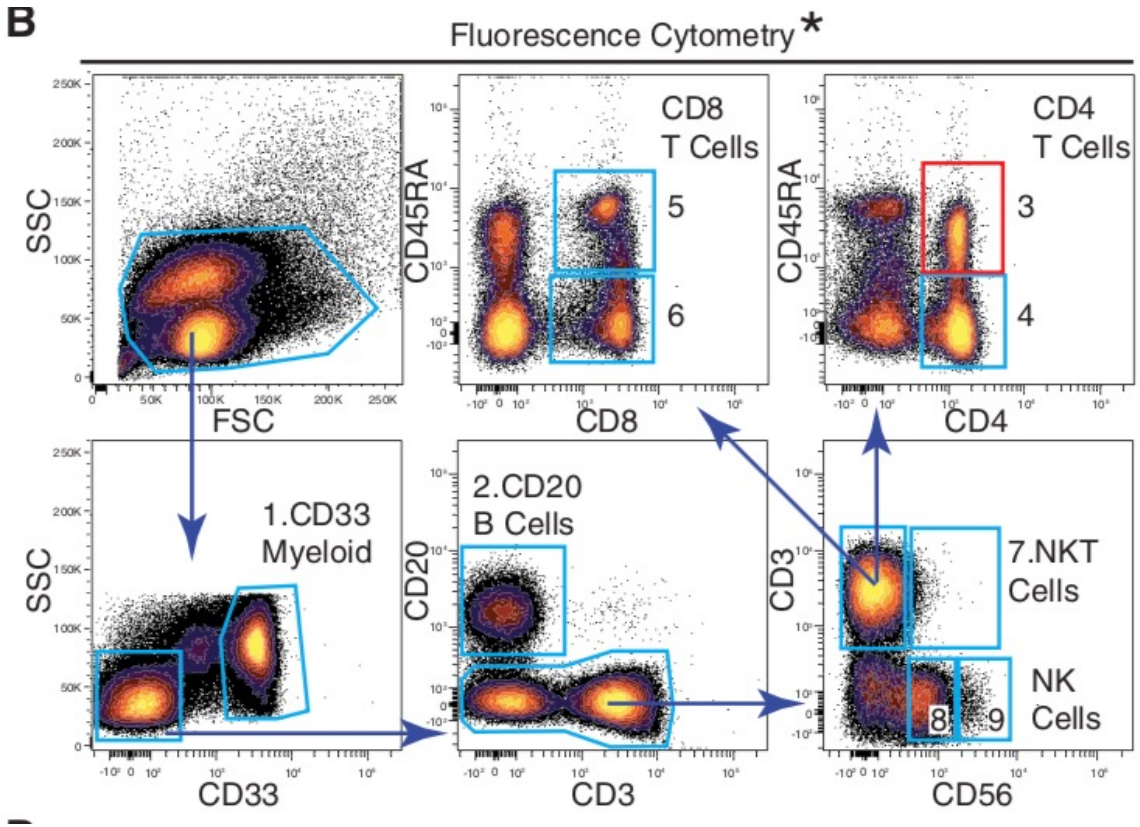
The debut of mass cytometry

Science
AAAS

Single-Cell Mass Cytometry of Differential Immune and Drug Responses Across a Human Hematopoietic Continuum

Sean C. Bendall,[1*] Erin F. Simonds,[1*] Peng Qiu,[2] El-ad D. Amir,[3] Peter O. Krutzik,[1] Rachel Finck,[1] Robert V. Bruggner,[1,7] Rachel Melamed,[3] Angelica Trejo,[1] Olga I. Ornatsky,[4,5] Robert S. Balderas,[6] Sylvia K. Plevritis,[2] Karen Sachs,[1] Dana Pe'er,[3] Scott D. Tanner,[4,5] Garry P. Nolan[1†]

CyTOF pre-processing: make it as similar to flow cytometry as possible



mass cytometry

What we see as computational biologists

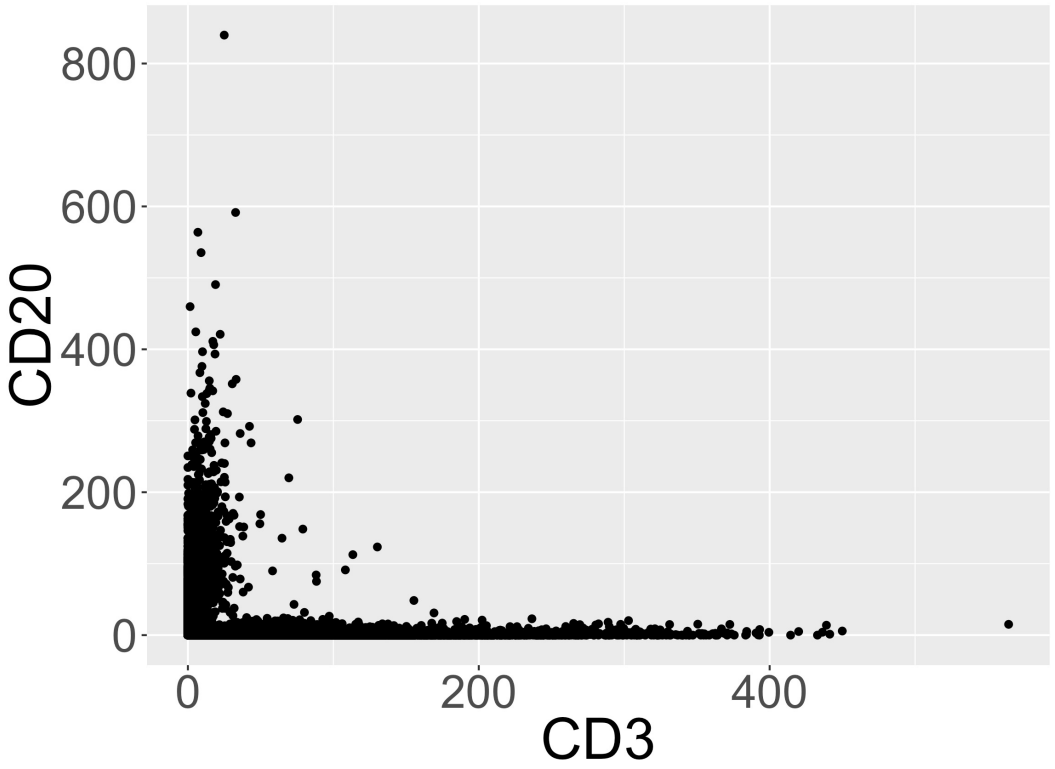
Data structure: tibble
Package: tibble (found in the larger package “tidyverse” by Hadley Wickham)

```
ff <- flowCore::read.FCS(list.files(pattern = "fcs"))
cells <- exprs(ff) %>% as_tibble()
colnames(cells) <- ff@parameters@data$desc
```

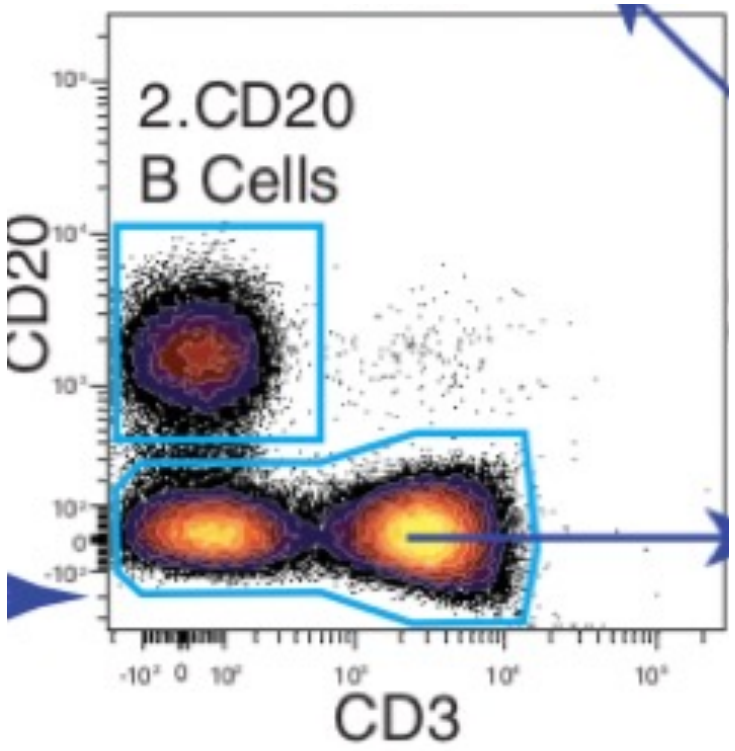
```
# A tibble: 19,696 x 62
   `CD25_1 (v)` CD25_2 Ba138Di `CD45 (v)` Cs133Di `CD28 (v)` `CD23_aAPC (v)` `CrTH2 (v)` `CCR10 (v)`
          <dbl>  <dbl>   <dbl>      <dbl>   <dbl>      <dbl>           <dbl>       <dbl>       <dbl>
 1        2.64   0.935    4.22      134.        0    114.             23.4         1.67        1.15
 2        2.01   0       24.5        64.0       0      9.84            1.94        0           0
 3      106.    36.3      2.28       84.3       0     69.3             0           0           0
 4        0      3.62     5.62       78.0       0      0.901           0.691       0           0
 5        0.934  0       39.5        51.0       0      0               0           2.81        2.12
 6        7.45   2.20    17.7        43.0       0     96.7             0           0           0
 7        0      2.48     9.72       12.6       0     12.3             2.12       41.2         0
 8        9.40  16.5     16.3        87.1       0     75.7             0.0114      0           0
 9       45.1   33.6      4.33      107.        0      0               8.45        0           0
10        2.86   2.36    21.9        41.3       0      2.98            1.44        0           0
# … with 19,686 more rows, and 53 more variables: `CD36 (v)` <dbl>, `CD38 (v)` <dbl>, `CD73
#   (v)` <dbl>, `iNKT_aCy5 (v)` <dbl>, `TCRgd (v)` <dbl>, `IgE (v)` <dbl>, Event_length <dbl>, `CD27
#   (v)` <dbl>, `CXCR3 (v)` <dbl>, `CD16 (v)` <dbl>, `CCR4 (v)` <dbl>, `CD14 (v)` <dbl>, `CD127
#   (v)` <dbl>, `CD20 (v)` <dbl>, `CD3 (v)` <dbl>, DNA1 <dbl>, DNA2 <dbl>, `CD57 (v)` <dbl>,
#   `CD21_aFITC (v)` <dbl>, `CD19 (v)` <dbl>, `CD123 (v)` <dbl>, `CD4 (v)` <dbl>, `CD11c (v)` <dbl>,
#   `CD7 (v)` <dbl>, `IgA (v)` <dbl>, `TCRa72 (v)` <dbl>, Pd102Di <dbl>, BC1 <dbl>, Pd105Di <dbl>,
#   BC2 <dbl>, BC3 <dbl>, BC4 <dbl>, `CCR6 (v)` <dbl>, Pt194Di <dbl>, `CD5 (v)` <dbl>, `CD8
#   (v)` <dbl>, BC5 <dbl>, LD <dbl>, `IgD (v)` <dbl>, `CD39 (v)` <dbl>, `CD95 (v)` <dbl>, `CD1c
```

What we see as computational biologists: the raw data

ggplot2::qplot()

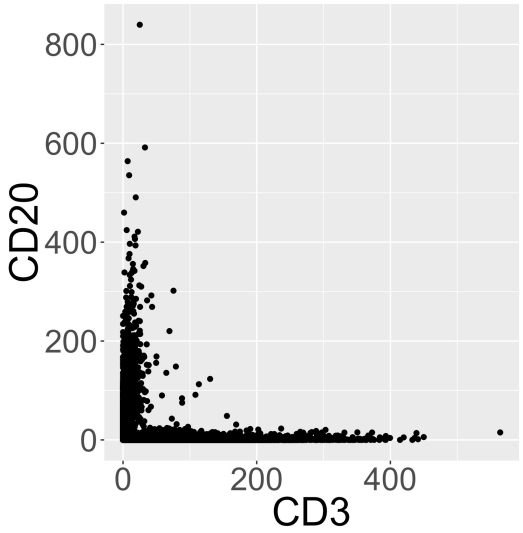


?

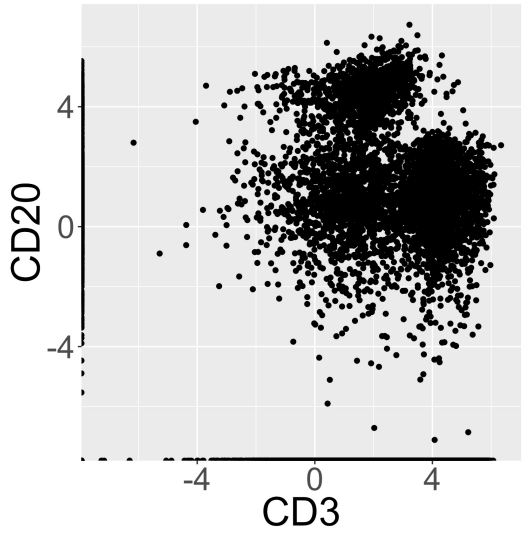


Testing log normality of the data

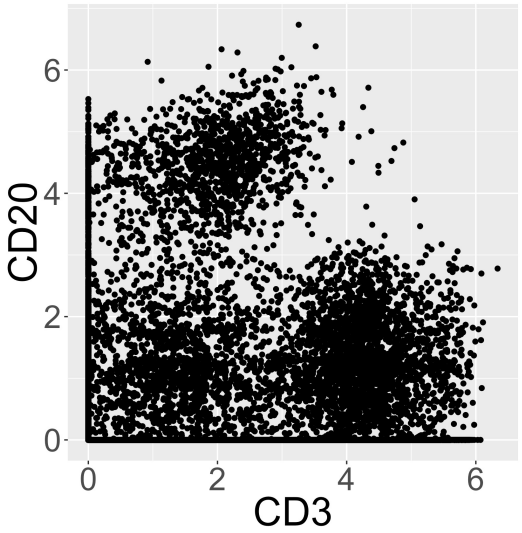
Raw

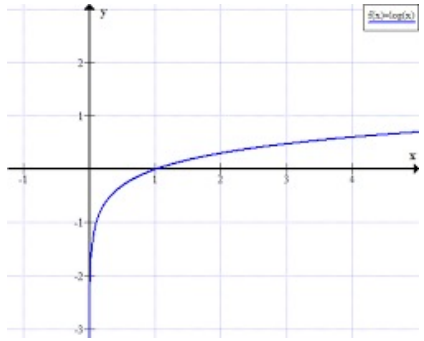


Log(x)

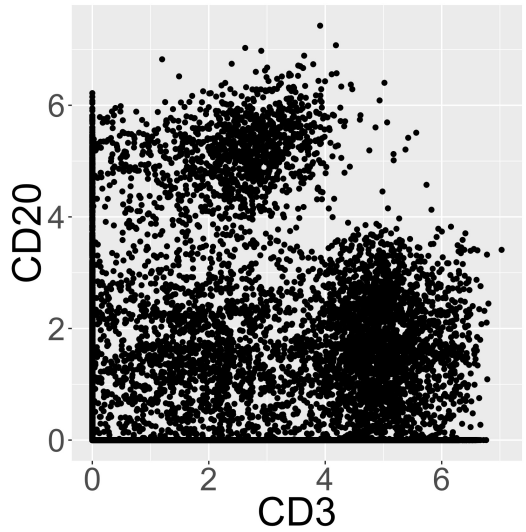


Log(x + 1)

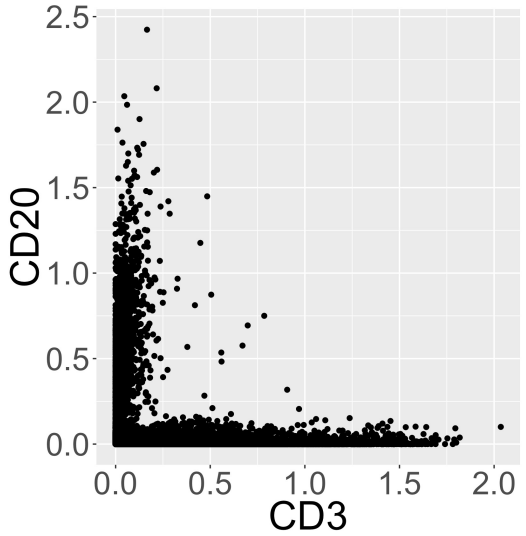




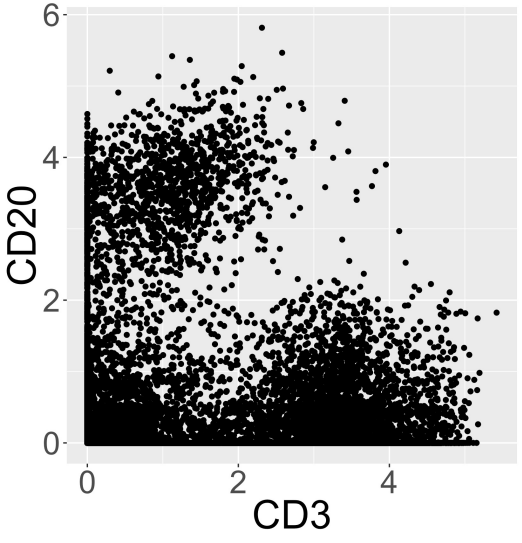
asinh(x)

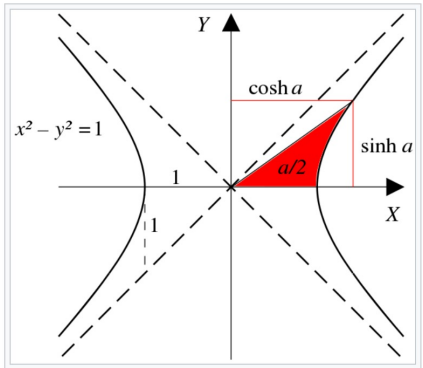


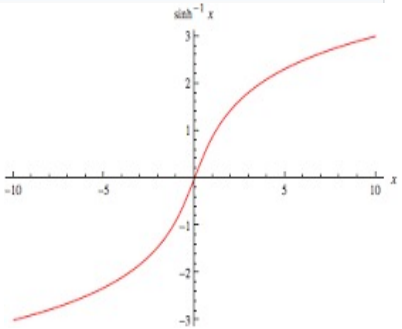
asinh(x/150)



asinh(x/5)



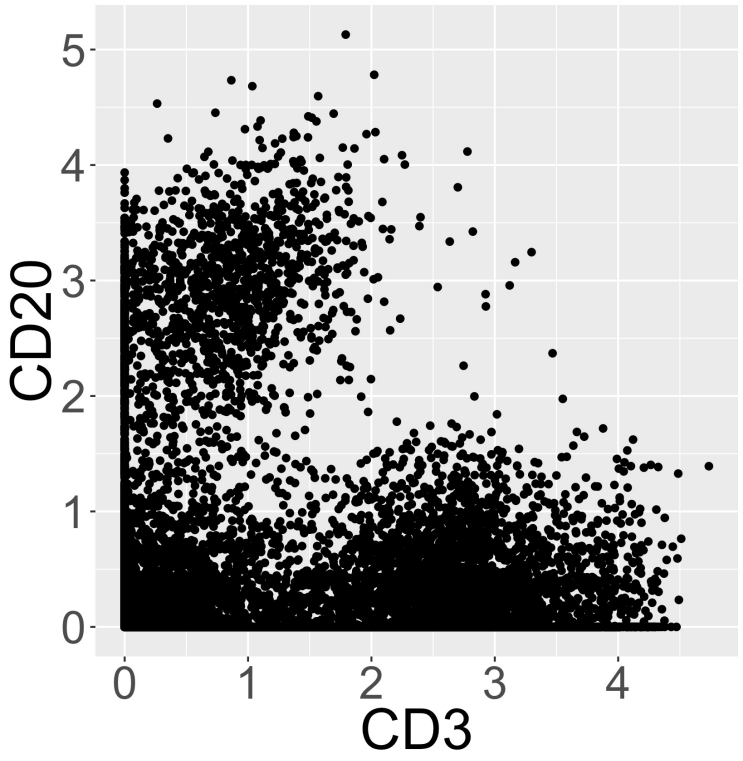




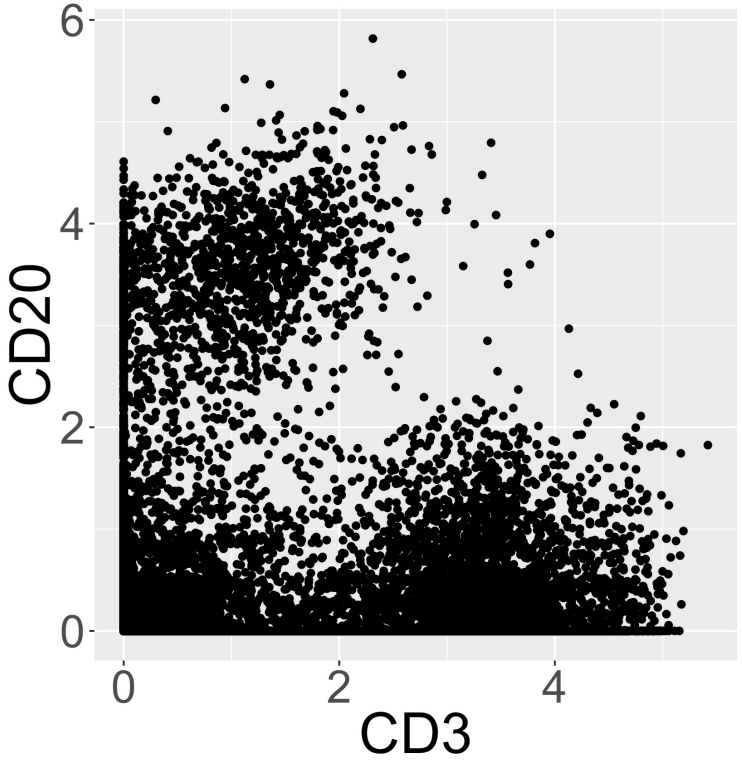
Similar scale arguments can be worked into the logarithmic functions

Try evaluating multi-modality rigorously (eg. Hartigan's dip test)

log(x/5 + 1)

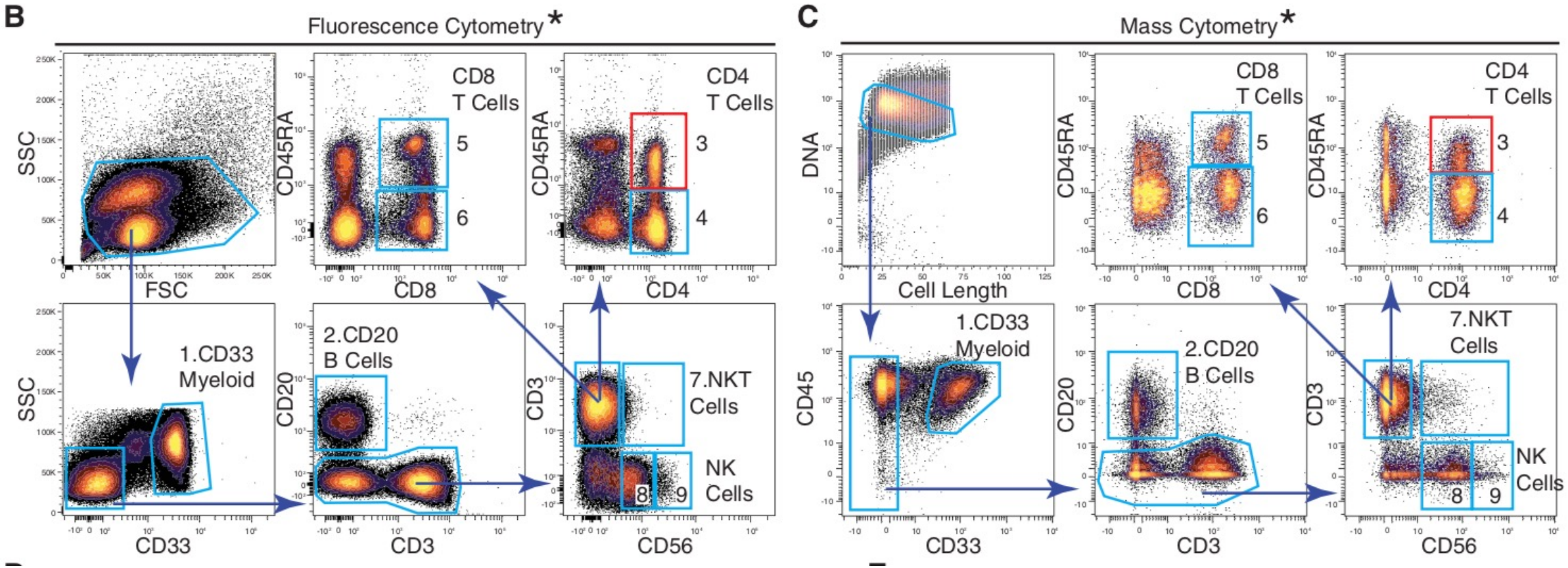


asinh(x/5)

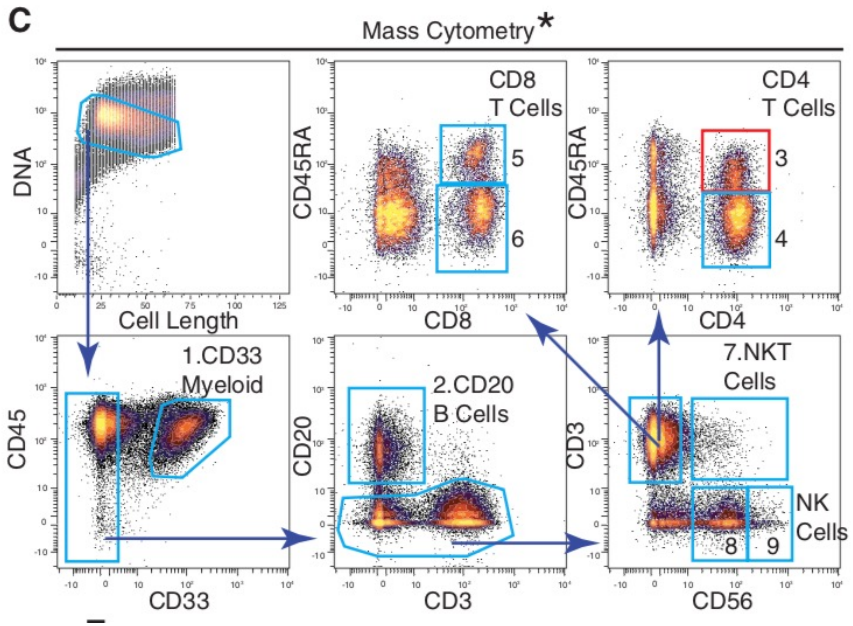


CyTOF pre-processing: make it as similar to flow cytometry as possible

asinh(x/5)

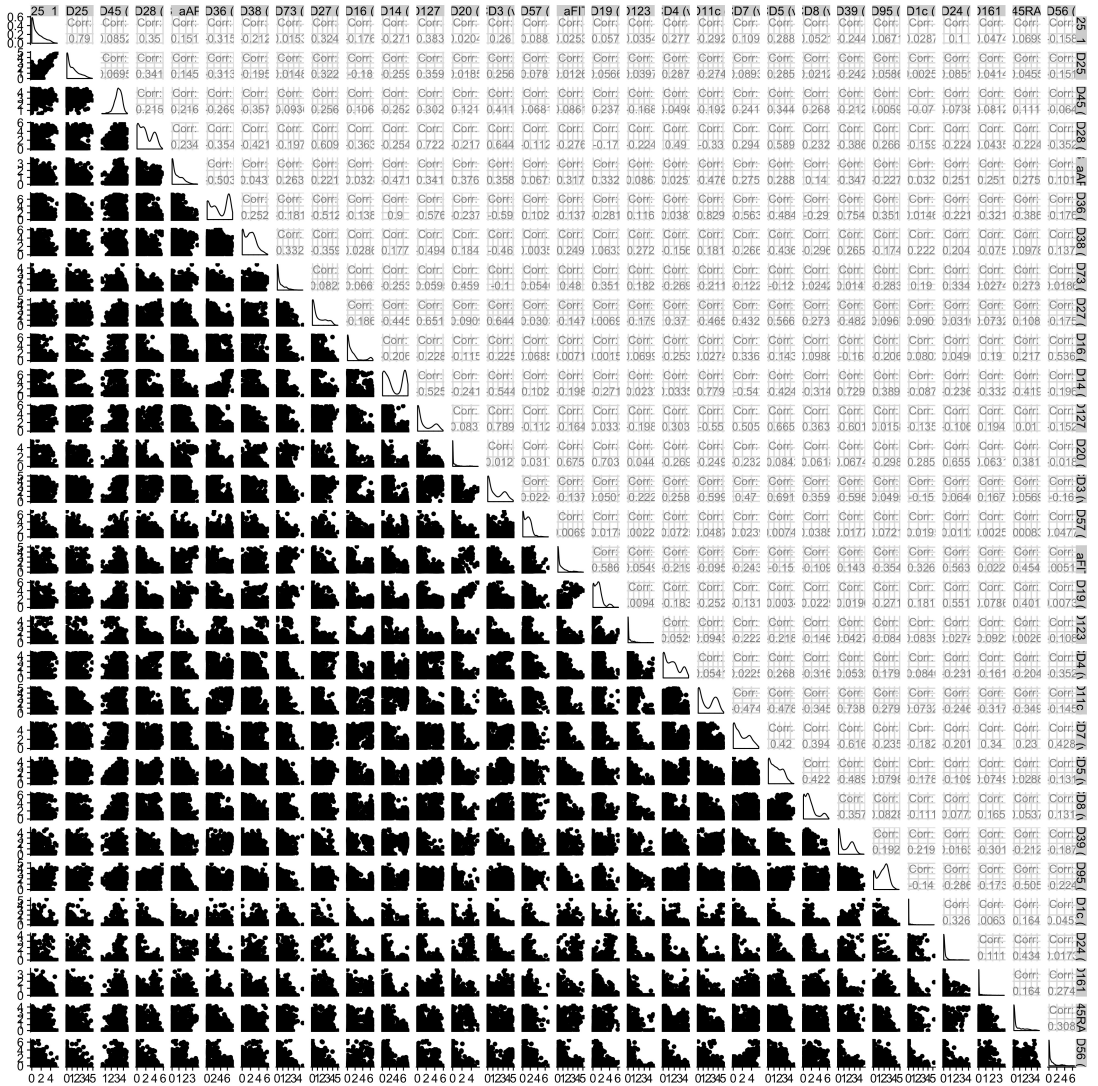


Exhaustive manual gating for mass cytometry, for novel subset discovery



GGally::ggpairs()

30 markers = 435 biaxial plots

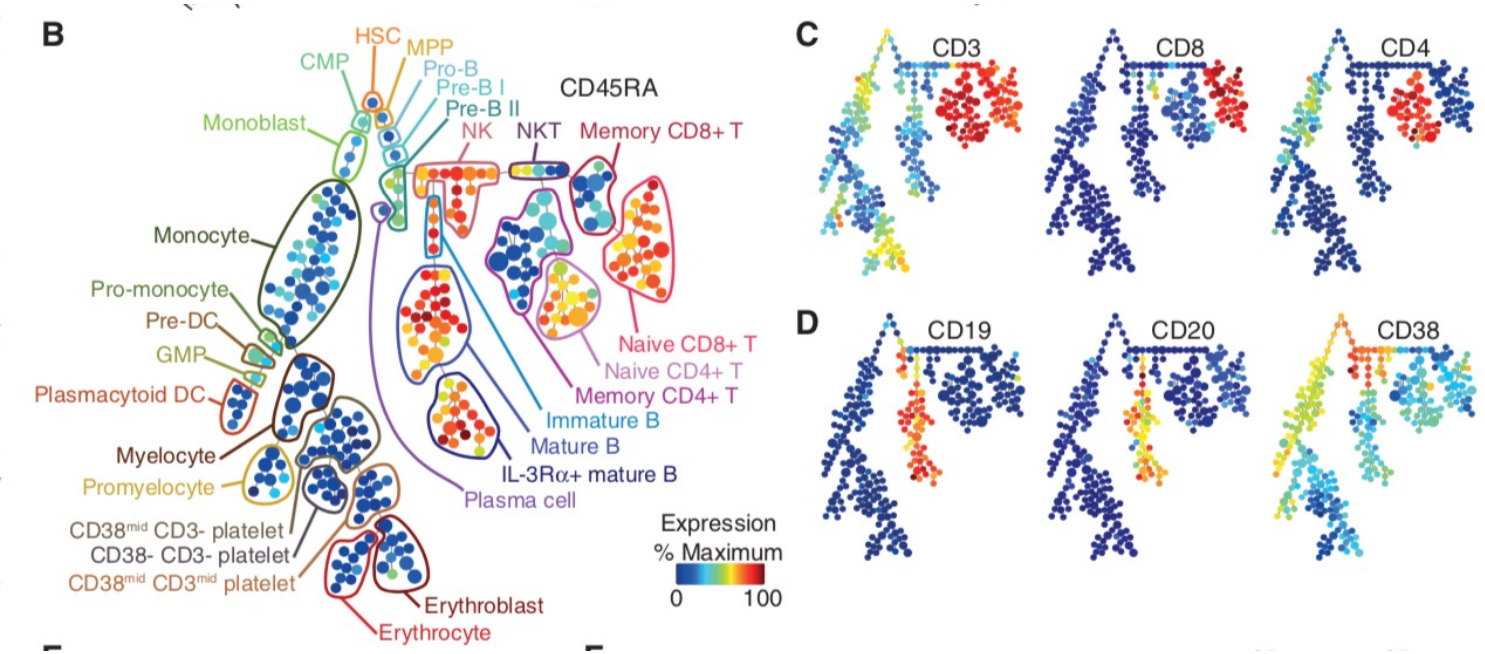


Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE

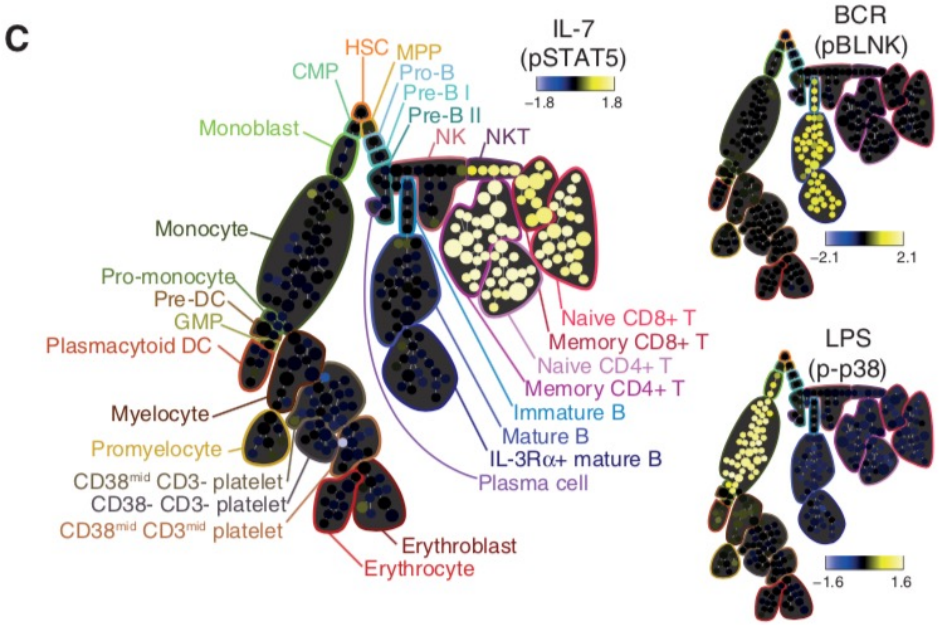
VOLUME 29 NUMBER 10 OCTOBER 2011 NATURE BIOTECHNOLOGY

Peng Qiu[1,2], Erin F Simonds[3], Sean C Bendall[3], Kenneth D Gibbs Jr[3], Robert V Bruggner[3], Michael D Linderman[4], Karen Sachs[3], Garry P Nolan[3] & Sylvia K Plevritis[1]

Marker expression

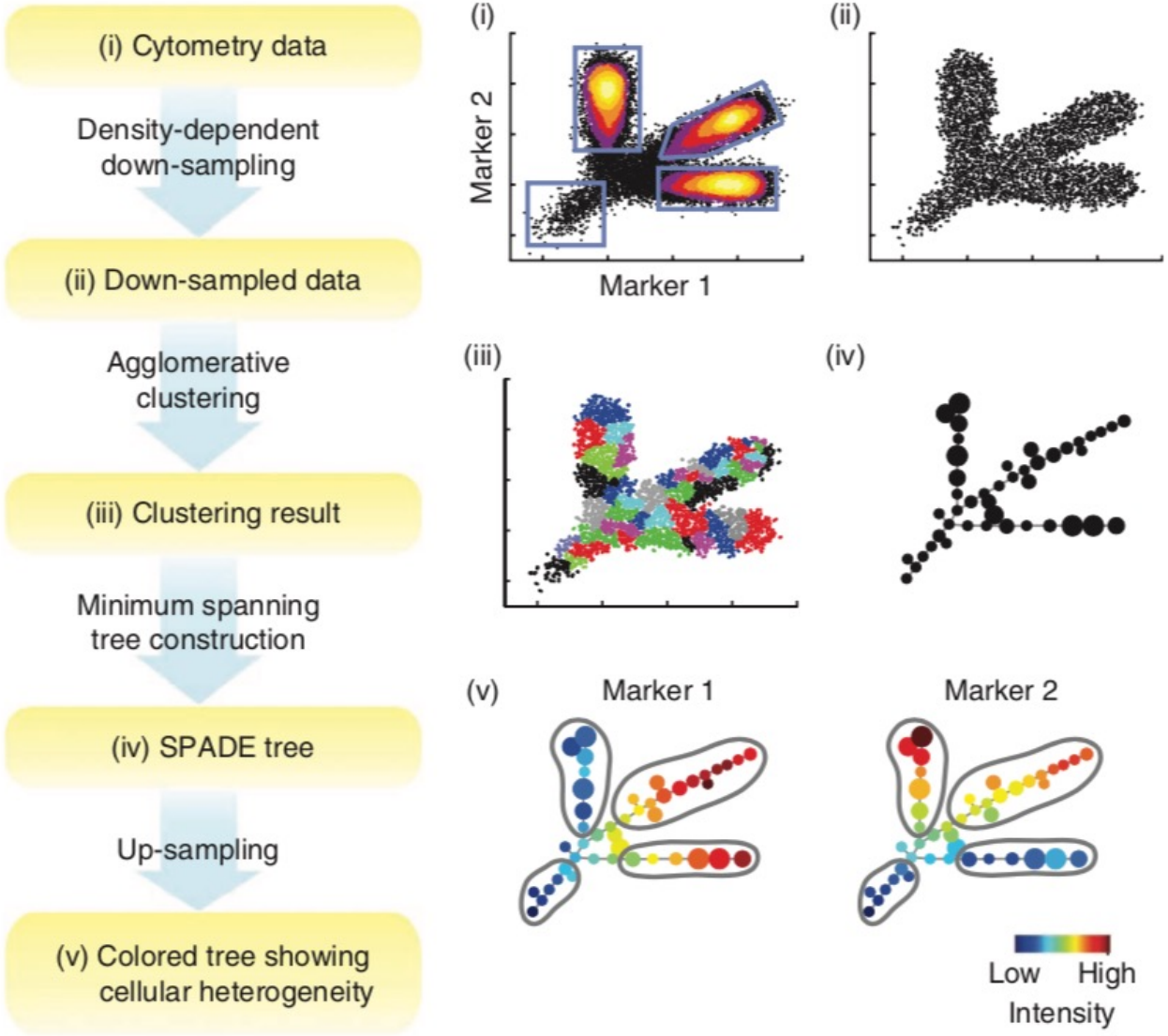


Change in phospho-protein levels



BUT WHERE ARE THE P VALUES???

How does SPADE work?



Minimum spanning tree:
All vertices are connected without any cycles, and with the minimum possible edge weight (distance).

Additional complexity:
SPADE uses the L1 distance for all steps, whereas most other CyTOF tools I've seen use the L2 (Euclidean) distance.

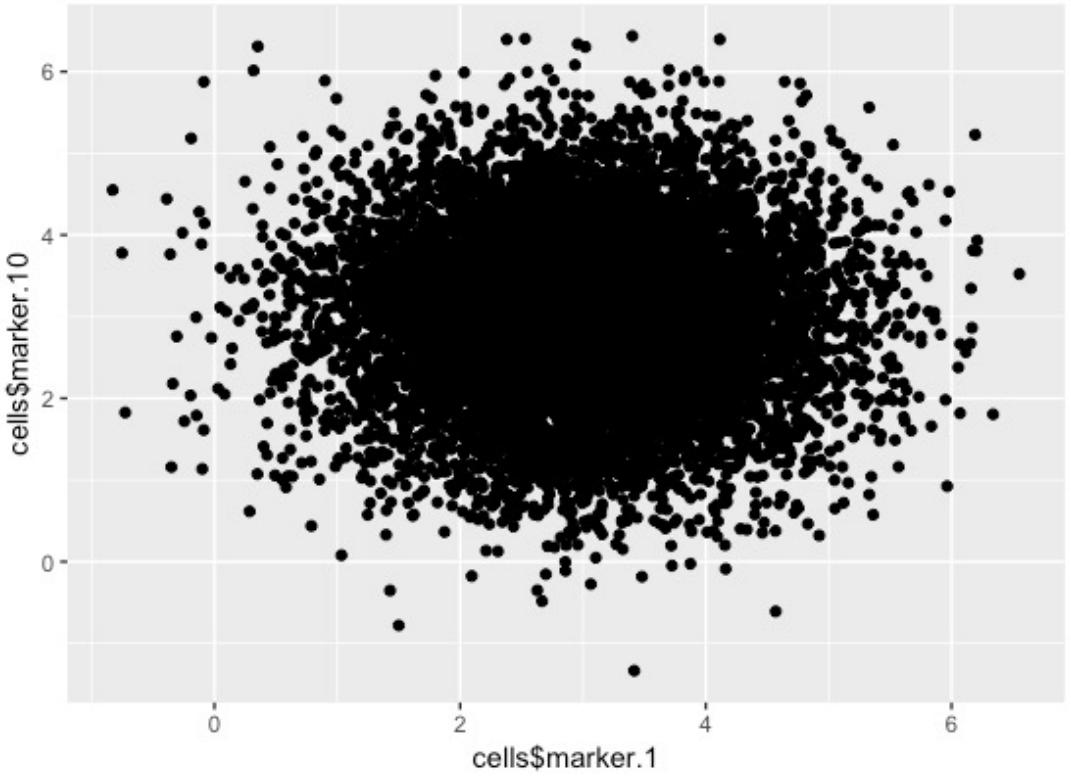
What happens when you run SPADE on random data?

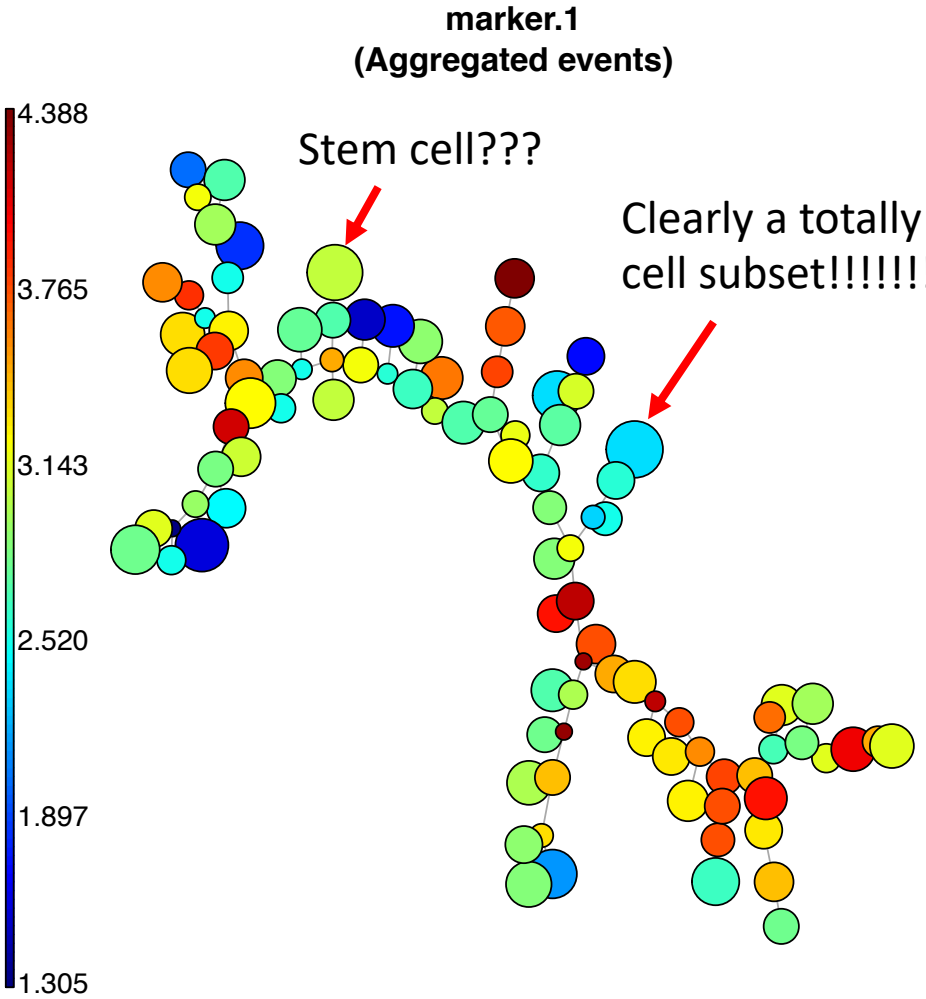
```r
library(tidyverse)

# pipeline ----------------------------------------

cells <- lapply(seq(30), function(i) {
    curr <- rnorm(10000, mean = 3, sd = 1)
    return(curr)
}) %>%
    do.call(cbind, .) %>%
    as_tibble()

names(cells) <- paste("marker", seq(30), sep = ".")

write.csv(cells, "bogus_cell_data.csv")
```

```
# A tibble: 10,000 x 30
   marker.1 marker.2 marker.3 marker.4 marker.5 marker.6 marker.7 marker.8 marker.9 marker.10 marker.11
      <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl>     <dbl>     <dbl>
 1    3.75      2.37     2.74     2.03     3.34     3.10     3.14     1.93     2.90      2.68      1.77
 2    0.602     4.26     2.96     2.82     0.883    3.48     2.22     2.08     1.39      1.95      3.00
 3    2.41      2.71     3.60     4.74     2.64     4.30     1.75     3.57     3.15      5.39      3.01
 4    3.39      4.24     3.36     5.42     5.03     1.05     4.26     3.94     3.84      2.19      3.00
 5    2.76      3.43     2.48     4.83     1.55     1.00     3.93     4.63     4.22      3.60      2.82
 6    2.78      2.34     3.03     2.06     1.97     4.06     4.56     2.67     4.56      3.10      4.01
 7    2.59      4.23     3.99     2.45     3.48     4.08     2.38     3.45     3.44      3.66      4.23
 8    2.90      2.15     4.06     2.02     1.85     2.48     4.81     2.53     1.42      2.51      1.19
 9    1.98      3.16     3.19     3.65     4.27     3.91     2.61     3.32     2.43      3.20      2.17
10    3.50      3.89     4.19     2.84     2.13     2.75     4.14     5.45     4.78      4.47      4.35
# … with 9,990 more rows, and 19 more variables: marker.12 <dbl>, marker.13 <dbl>, marker.14 <dbl>,
#   marker.15 <dbl>, marker.16 <dbl>, marker.17 <dbl>, marker.18 <dbl>, marker.19 <dbl>,
#   marker.20 <dbl>, marker.21 <dbl>, marker.22 <dbl>, marker.23 <dbl>, marker.24 <dbl>,
#   marker.25 <dbl>, marker.26 <dbl>, marker.27 <dbl>, marker.28 <dbl>, marker.29 <dbl>,
#   marker.30 <dbl>
```

What happens when you run SPADE on random data?

30 dimensional hairball





There are probably ways to evaluate the MINIMUM-NESS of the spanning tree, to avoid seeing meaning where there is none

And thus began the clustering holy wars…

Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE

Peng Qiu[1,2], Erin F Simonds[3], Sean C Bendall[3], Kenneth D Gibbs Jr[3], Robert V Bruggner[3], Michael D Linderman[4], Karen Sachs[3], Garry P Nolan[3] & Sylvia K Plevritis[1]

Cell

Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis

Graphical Abstract

Authors

Jacob H. Levine, Erin F. Simonds, Sean C. Bendall, ..., James R. Downing, Dana Pe'er, Garry P. Nolan

Published in final edited form as:
Nat Methods. 2016 June ; 13(6): 493–496. doi:10.1038/nmeth.3863.

Automated Mapping of Phenotype Space with Single-Cell Data

Nikolay Samusik[1], Zinaida Good[1,2], Matthew H. Spitzer[1,2], Kara L. Davis[1,3], and Garry P. Nolan[1,*]

[1]Department of Microbiology and Immunology, Stanford University School of Medicine, Stanford, California, USA

[2]Department of Pathology, Stanford University School of Medicine, Stanford, California, USA

[3]Department of Pediatric Hematology and Oncology, Stanford University School of Medicine, Stanford, California, USA

Methodology article | Open Access

Data reduction for spectral clustering to analyze high throughput flow cytometry data

Habil Zare, Parisa Shooshtari, Arvind Gupta and Ryan R Brinkman

BMC Bioinformatics 2010 **11**:403

https://doi.org/10.1186/1471-2105-11-403 | © Zare et al; licensee BioMed Central Ltd. 2010

Received: 21 December 2009 | **Accepted:** 28 July 2010 | **Published:** 28 July 2010

Cytometry PART A — Journal of Quantitative Cell Science — ISAC

Original Article | Free Access

*immuno*Clust—An automated analysis pipeline for the identification of immunophenotypic signatures in high-dimensional cytometric datasets

Till Sörensen, Sabine Baumgart, Pawel Durek, Andreas Grützkau, Thomas Häupl

Automatic Classification of Cellular Expression by Nonlinear Stochastic Embedding (ACCENSE)

Karthik Shekhar, Petter Brodin, Mark M. Davis, and Arup K. Chakraborty

PNAS January 7, 2014 111 (1) 202-207; published ahead of print December 16, 2013
https://doi.org/10.1073/pnas.1321405111

Contributed by Mark M. Davis, November 19, 2013 (sent for review October 5, 2013)

Cytometry PART A — Journal of Quantitative Cell Science — ISAC

Original Article | Free Access

FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data

Sofie Van Gassen, Britt Callebaut, Mary J. Van Helden, Bart N. Lambrecht, Piet Demeester, Tom Dhaene, Yvan Saeys

How should we evaluate these clustering tools? asked Lukas Weber and Mark Robinson.

Clustering algorithm

Expert manual gating

TP = True Positives
TN = True Negatives
FP = False Positives
FN = False Negatives

	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

$$\textbf{Precision} = \frac{TP}{TP + FP}$$

$$\textbf{Recall} = \frac{TP}{TP + FN}$$

$$\textbf{F1-score} = \frac{2 * precision * recall}{precision + recall}$$

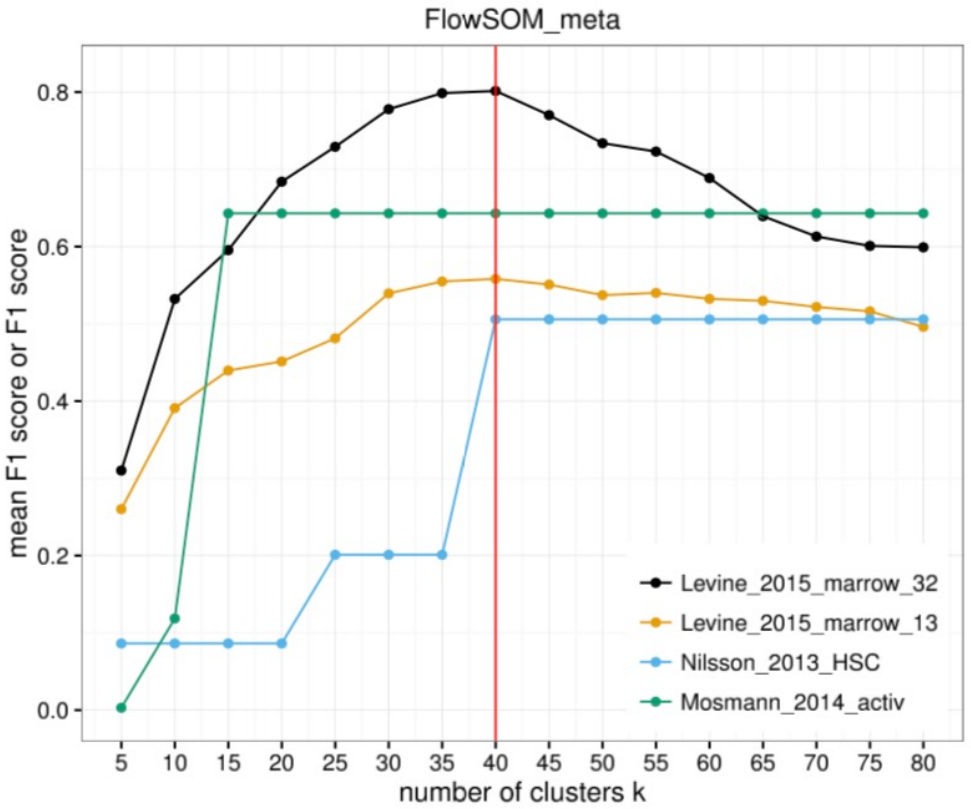
Comparison of Clustering Methods for High-Dimensional Single-Cell Flow and Mass Cytometry Data

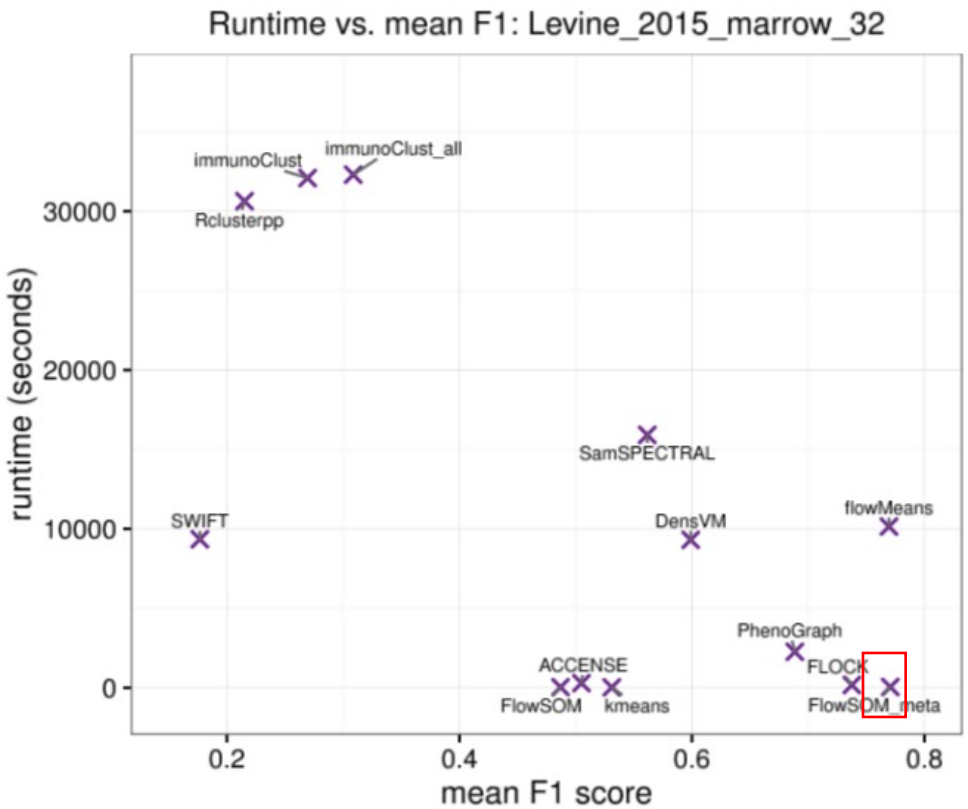
Lukas M. Weber[1,2], Mark D. Robinson[1,2,*] (Our heroes in the story)

[1] Institute of Molecular Life Sciences, University of Zurich, Switzerland

[2] SIB Swiss Institute of Bioinformatics, University of Zurich, Switzerland

Turn this into a brute-force program for single cell data





FlowSOM wins!

General temporal progression of CyTOF clustering algorithm development

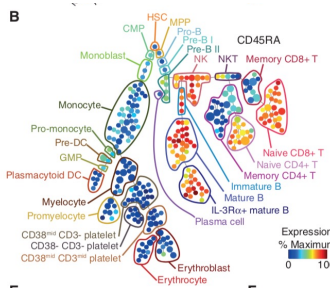
New way of analyzing data type

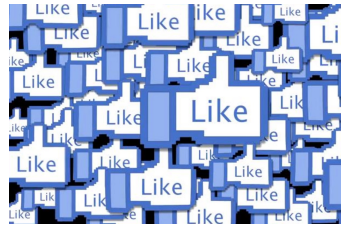
People see pitfalls and room for improvement

Everyone makes a better version at the same time

Someone develops/uses an evaluation metric for the algorithms and identifies the best ones

Particular algorithms become MUCH more popular





Lessons learned

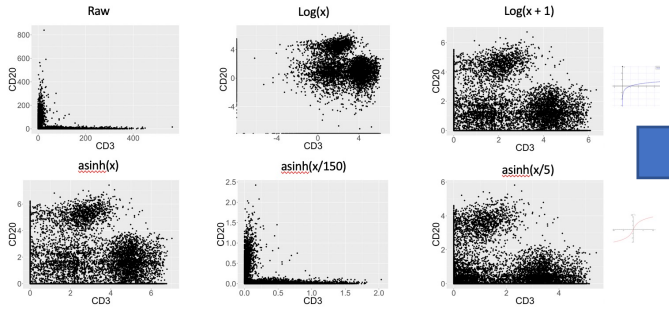
- If something is radically new, it doesn’t have to be PERFECT to make it into Cell/Science/Nature. It’s a prototype.
- Every bioinformatic tool has its assumptions and limitations. Break it. Benchmark it.
- Evaluation metrics of unsupervised learning algorithms and dataset reanalysis is underexplored.
- Think of a CyTOF pipeline as an interaction between the cell expression matrix, cell cluster frequency table, and a gating/visualization tool

Outline

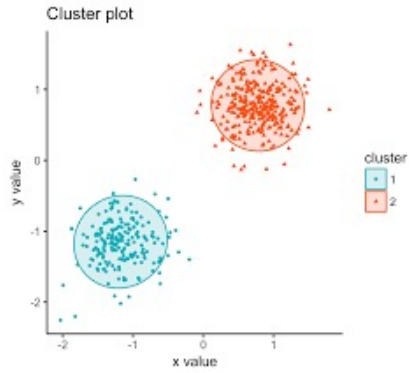
- Part 1: History of CyTOF analysis
- Part 2: How to develop a robust analytical pipeline

CyTOF analysis: general principles relevant to the DRFZ

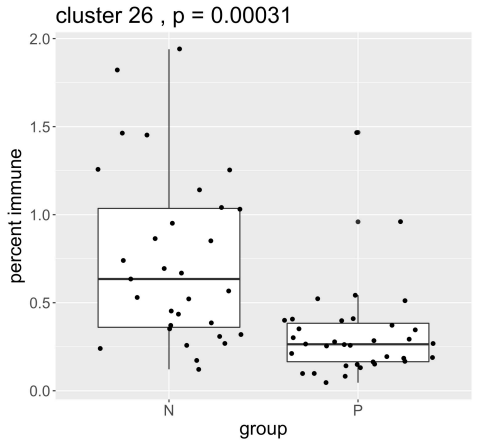
Pre-processing



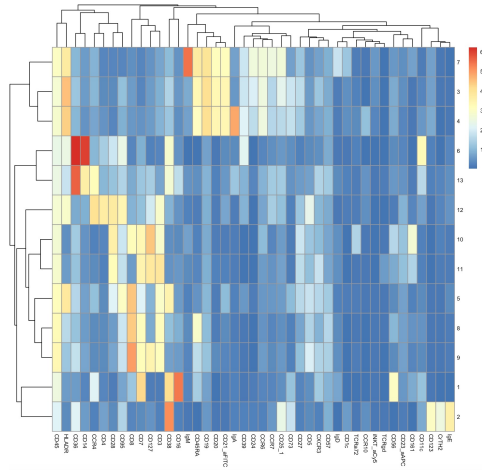
Grouping



Per-group statistics



Visualizations



First, process the data

```
# Make the flow set
fs <- FlowsetWrapper(total_files, subsample = sub_sample)
```

Read in every fcs file
Asinh transfer of markers within fcs file
Subsample each fcs file to user-defined number of cells

```
Subsampling flow set
> fs
A flowSet with 20 experiments.

  column names:
  89Y_CD15 102Pd 103Rh_live-dead 104Pd_barcode 105Pd_barcode 106Pd_barcode 108Pd_barcode 110Pd_barcode 113In_
CD66b 115In_Siglec8 127I 130Ba 140Ce_CD14 142Nd_cleaved_caspase_3 143Nd_CD19 144Nd_pPLCg2 145Nd_CD4 146Nd_CD4
5RO 147Sm_CD20 148Nd_IgA 149Sm_Syk 150Nd_pSTAT5 151Eu_CD123 152Sm_CD45RA 153Eu_pSTAT1 154Sm_CD1c 155Gd_CD27 1
56Gd_p38 158Gd_pSTAT3 159Tb_pMAPKAPK2 160Gd_CD11c 161Dy_CD7 162Dy_IgM 163Dy_CCR7 164Dy_IkBa 165Ho_pCREB 166Er
_pNFkBp65 167Er_CD38 168Er_CD16 169Tm_CD25 170Er_Siglec1 171Yb_ZAP70_Syk 172Yb_pS6 173Yb_IgD 174Yb_HLA-DR 175
Lu_CXCR3 176Yb_CD56 190BCKG 191Ir_DNA 193Ir_DNA 194Pt_barcode 195Pt_CD3 196Pt_CD8 198Pt_CD45 208Pb 209Bi_CD11
b
```

Output: A flow set
Package: FlowCore

…containing our expression matrices

```
> exprs(fs[[1]]) %>% as.tibble()
# A tibble: 100,000 x 56
   `89Y_CD15` `102Pd` `103Rh_live-dea… `104Pd_barcode` `105Pd_barcode` `106Pd_barcode` `108Pd_barcode`
        <dbl>   <dbl>            <dbl>           <dbl>           <dbl>           <dbl>           <dbl>
 1     0.435     418.            0.790            341.            409.            58.6            38.5
 2     0         239.            0.769            259.            294.            44.5            22.4
 3     0.0209    260.            0                222.            258.            33.3            12.9
 4     0.573     326.            0                257.            243.            41.1            23.2
 5     2.26      239.            0                201.            228.            32.9            24.1
 6     1.34      415.            0                310.            356.            32.0            28.2
 7     0.703     323.            3.40             268.            218.            25.6            27.0
 8     0.313     324.            1.14             290.            378.            21.3            24.0
 9     0.490     256.            1.49             226.            225.            16.6            25.0
10     0         164.            0                169.            152.            15.9             6.46
# … with 99,990 more rows, and 49 more variables: `110Pd_barcode` <dbl>, `113In_CD66b` <dbl>,
#   `115In_Siglec8` <dbl>, `127I` <dbl>, `130Ba` <dbl>, `140Ce_CD14` <dbl>,
#   `142Nd_cleaved_caspase_3` <dbl>, `143Nd_CD19` <dbl>, `144Nd_pPLCg2` <dbl>, `145Nd_CD4` <dbl>,
#   `146Nd_CD45RO` <dbl>, `147Sm_CD20` <dbl>, `148Nd_IgA` <dbl>, `149Sm_Syk` <dbl>,
#   `150Nd_pSTAT5` <dbl>, `151Eu_CD123` <dbl>, `152Sm_CD45RA` <dbl>, `153Eu_pSTAT1` <dbl>,
#   `154Sm_CD1c` <dbl>, `155Gd_CD27` <dbl>, `156Gd_p38` <dbl>, `158Gd_pSTAT3` <dbl>,
#   `159Tb_pMAPKAPK2` <dbl>, `160Gd_CD11c` <dbl>, `161Dy_CD7` <dbl>, `162Dy_IgM` <dbl>,
#   `163Dy_CCR7` <dbl>, `164Dy_IkBa` <dbl>, `165Ho_pCREB` <dbl>, `166Er_pNFkBp65` <dbl>,
```

…of flow frames
Package: FlowCore

```
> fs[[1]]
flowFrame object 'c01_ExpT29_HC_SLE_pool2_SLE16_SLE.fcs'
with 100000 cells and 56 observables:
                      name                 desc range minRange maxRange
$P3                89Y_CD15             89Y_CD15  8192        0     8191
$P4                   102Pd                102Pd  4096        0     4095
$P5         103Rh_live-dead      103Rh_live-dead  4096        0     4095
$P6           104Pd_barcode        104Pd_barcode  4096        0     4095
$P7           105Pd_barcode        105Pd_barcode  4096        0     4095
$P8           106Pd_barcode        106Pd_barcode  4096        0     4095
$P9           108Pd_barcode        108Pd_barcode  4096        0     4095
$P10          110Pd_barcode        110Pd_barcode  4096        0     4095
$P11            113In_CD66b          113In_CD66b  4096        0     4095
$P12          115In_Siglec8        115In_Siglec8  4096        0     4095
```

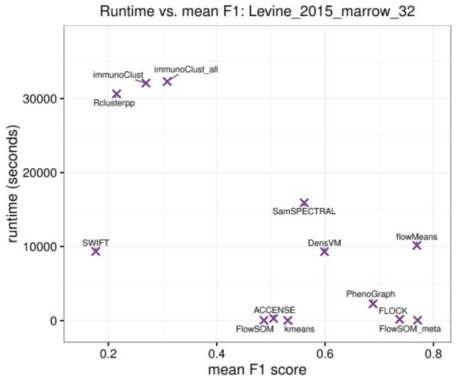
FlowSOM clustering

Cytometry PART A | Journal of Quantitative Cell Science | ISAC

Original Article | Free Access

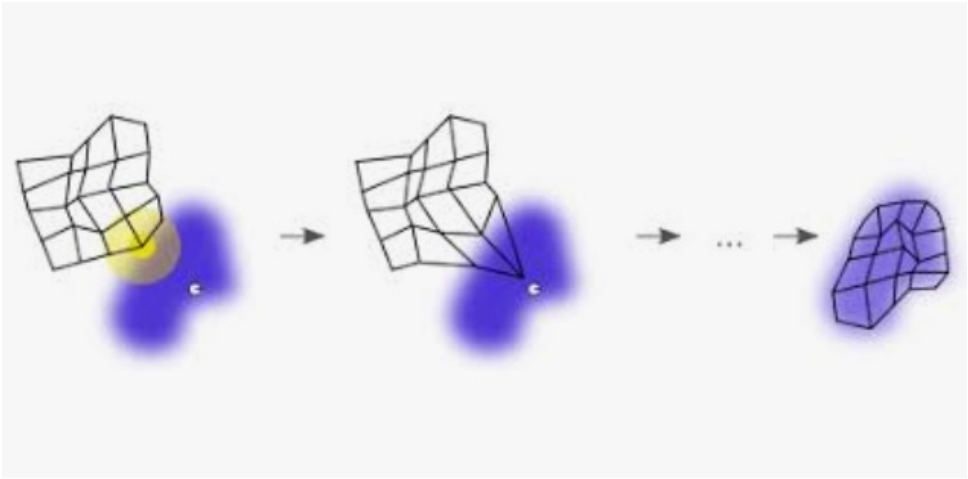
FlowSOM: Using self-organizing maps for visualization and interpretation of cytometry data

Sofie Van Gassen, Britt Callebaut, Mary J. Van Helden, Bart N. Lambrecht, Piet Demeester, Tom Dhaene, Yvan Saeys

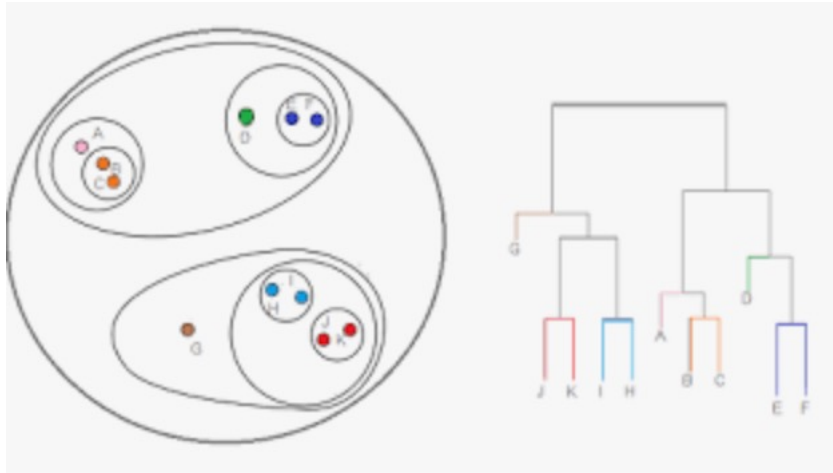


Note: try taking your favorite clustering algorithm and using it at the consensus step (eg. Louvain, Mean-shift).

Self organizing map (similar output to k-means)
Package: FlowSOM



Hierarchical clustering of the clusters
Package: ConsensusClusterPlus (within FlowSOM)



FlowSOM clustering

Newer versions use the FlowSOM package directly

```
clustering <- cytofkit::cytof_cluster(xdata = cells[,surface],
                                      method = clust_choice,
                                      Rphenograph_k = rphenograph_k, # Default 30
                                      FlowSOM_k = flowsom_numclust) # Default 40
```

Cluster only on surface markers

Make this chart in excel

```
> markers
          surface                functional
1       89Y_CD15 142Nd_cleaved_caspase_3
2     113In_CD66b             144Nd_pPLCg2
3  115In_Siglec8              149Sm_Syk
4      140Ce_CD14             150Nd_pSTAT5
5      143Nd_CD19             153Eu_pSTAT1
6       145Nd_CD4              156Gd_p38
7    146Nd_CD45RO             158Gd_pSTAT3
8      147Sm_CD20           159Tb_pMAPKAPK2
9       148Nd_IgA              164Dy_IkBa
10    151Eu_CD123             165Ho_pCREB
11   152Sm_CD45RA           166Er_pNFkBp65
12     154Sm_CD1c           171Yb_ZAP70_Syk
13     155Gd_CD27              172Yb_pS6
14    160Gd_CD11c
15      161Dy_CD7
16      162Dy_IgM
```

```
# A tibble: 2,000,000 x 60
   `89Y_CD15` `102Pd` `103Rh_live-dea… `104Pd_barcode` `105Pd_barcode` `106Pd_barcode` `108Pd_barcode`
        <dbl>   <dbl>            <dbl>           <dbl>           <dbl>           <dbl>           <dbl>
 1     0.435     418.            0.790            341.            409.            58.6            38.5
 2     0         239.            0.769            259.            294.            44.5            22.4
 3     0.0209    260.            0                222.            258.            33.3            12.9
 4     0.573     326.            0                257.            243.            41.1            23.2
 5     2.26      239.            0                201.            228.            32.9            24.1
 6     1.34      415.            0                310.            356.            32.0            28.2
 7     0.703     323.            3.40             268.            218.            25.6            27.0
 8     0.313     324.            1.14             290.            378.            21.3            24.0
 9     0.490     256.            1.49             226.            225.            16.6            25.0
10     0         164.            0                169.            152.            15.9             6.46
# … with 1,999,990 more rows, and 53 more variables: `110Pd_barcode` <dbl>, `113In_CD66b` <dbl>,
#   `115In_Siglec8` <dbl>, `127I` <dbl>, `130Ba` <dbl>, `140Ce_CD14` <dbl>,
#   `142Nd_cleaved_caspase_3` <dbl>, `143Nd_CD19` <dbl>, `144Nd_pPLCg2` <dbl>, `145Nd_CD4` <dbl>,
#   `146Nd_CD45RO` <dbl>, `147Sm_CD20` <dbl>, `148Nd_IgA` <dbl>, `149Sm_Syk` <dbl>,
#   `150Nd_pSTAT5` <dbl>, `151Eu_CD123` <dbl>, `152Sm_CD45RA` <dbl>, `153Eu_pSTAT1` <dbl>,
#   `154Sm_CD1c` <dbl>, `155Gd_CD27` <dbl>, `156Gd_p38` <dbl>, `158Gd_pSTAT3` <dbl>,
#   `159Tb_pMAPKAPK2` <dbl>, `160Gd_CD11c` <dbl>, `161Dy_CD7` <dbl>, `162Dy_IgM` <dbl>,
#   `163Dy_CCR7` <dbl>, `164Dy_IkBa` <dbl>, `165Ho_pCREB` <dbl>, `166Er_pNFkBp65` <dbl>,
#   `167Er_CD38` <dbl>, `168Er_CD16` <dbl>, `169Tm_CD25` <dbl>, `170Er_Siglec1` <dbl>,
#   `171Yb_ZAP70_Syk` <dbl>, `172Yb_pS6` <dbl>, `173Yb_IgD` <dbl>, `174Yb_HLA-DR` <dbl>,
#   `175Lu_CXCR3` <dbl>, `176Yb_CD56` <dbl>, `190BCKG` <dbl>, `191Ir_DNA` <dbl>, `193Ir_DNA` <dbl>,
#   `194Pt_barcode` <dbl>, `195Pt_CD3` <dbl>, `196Pt_CD8` <dbl>, `198Pt_CD45` <dbl>, `208Pb` <dbl>,
#   `209Bi_CD11b` <dbl>, condition <chr>, sampleID <chr>, file <chr>, cluster <int>
```

Vector of Cluster ID, attached to the end of the tibble

```
> cells$cluster
  [1]  4  4  4  4  4  4  4  4  4  9  4  4  8  4  4  4  4  4  3  4  4  4  4 17  4  4  4  4 13  4 23  4
 [33]  4  4  4  4  4  4  4  4  4  4  4  4  4  1  4  4 39  4  4  4  4  4 15  4  1  4  4  4  4  4 22  4  4
 [65]  4  4  4  4  4  4  4  4  4  4  4  4  4  4 30  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  1
 [97]  4  4  4  4  4  4  4  4  4  4  4  4 23  4  4 11  4  4 37  4  4  4  4  4  4 15  4  4  4  4  4  4
[129]  4 15  4  4  4  4  4  4  1  4  4  4  4  4  4  4  4  4  4  4  2 17  4  4  4  4  4  4  4  2  4  4  4
[161]  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4 14  4  4  4  4  4  4  4 13  4  4  4 13  4  4  4
[193]  4  4  4 13  4  7 30  4  4  4  4  4  4  4  8  4 40  4  4  5  4 40  4  4  4  4  4  9  4  4 13  4  4  4
[225] 25  4 11  4  4  4  4  4  4  4 30  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4 36  4  4  4
[257]  4  4  4 25  4  3 36  4  4  4  4  4  4 15  4  4  4 17  4  4  4  4 36  2  4  4  4  4  4  4  4  4
```

From number of clusters to cell frequency table

package: pheatmap

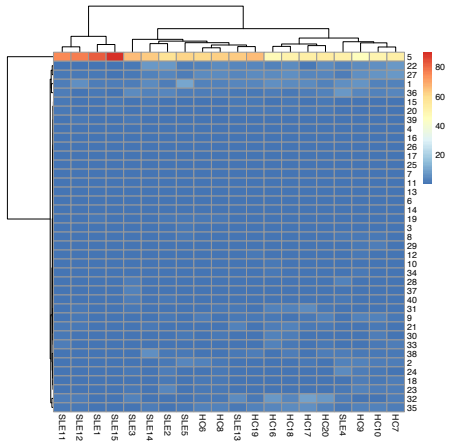
```
# Freq table for single instance of clustering, named by SampleID
counts <- GetFreqTable(clusters = cells$cluster,
                       cells = cells,
                       comp.conds = comp.conds)
```

Number of cells per cluster, per file
File summarized as “Sample ID” for aesthetics

```
> counts
    SLE1 SLE11 SLE12 SLE13 SLE14 SLE15  SLE2  SLE3  SLE4  SLE5  HC10  HC16  HC17  HC18  HC19  HC20
1   2355  2517  5980  4244  4817  1841  2922  2617  6485 10721  5776  6014  5907  5365  4596  3254
2    785   559   292   935   446   775   627   236   715  1022   313   462   400   705   716   529
3    740   683   368   578   568   413   943  2605  1137   647   562   572   641   299   209   622
4  82898 74776 77242 63202 63228 90177 58566 66551 51495 61380 49636 47626 52586 51145 66214 54251
5    316   390    99   227   287    92   182   285   147   141   172   201   129    92   121   192
6    141   102   447   437   133     6   478   173  1056   398  1076  1133  1041   781   591   657
7    474   294   250    53   332    58  1398   398  2902  3079   481    48    90    19    28    17
8    153   144   163   170   147   177   139   169   138   148   129   145   161   158   142   169
9    378   206   214   375    64    99   667    54  1094  1516  1073  1028   648   574  1033   737
10    75    51     9    17    23    36    32    47   117   153    74    56    27    33    16    51
11   176   146    46   274   215   101   197   107   390   514   212   258   207   258   205   193
12   232    99    23    76    69    85   195   131   137   165    96    96    60    99    55    93
13   536   668  1626  1359    56   158  1233   364   505   928  3427  1948  1663  1156  2147  2903
14   536   404   111   455   285   488   399   350   299   425   328   351   299   425   439   385
15  1369   264    17   321   285   331  1092   756   844   990   665   220   188   292   302   299
16   634  1141   171  1341  1079  1422   492   460   524   550   359   962   954  1234  1090  1097
17   450   831   132   381   200   303   309   335   218   278   168   250   208   214   195   182
18   111    84    18   213    83    37   229    77   173   186   154   152   110   193   220   257
19   260   140   526   137   263   127   208   358   604   261   241    86   121   136   100   128
20   271   190   432   247   371    28   196   469   183   155   421   160   269   295   194   205
```

```
# Convert to percentages
counts <- apply(counts, 2, function(j) {
    return(j/sum(j))
}) %>% as.tibble
counts <- counts * 100
```

```
# All frequencies
freq_heatmap <- pheatmap(counts)
```



```
> counts
     SLE1  SLE11  SLE12  SLE13  SLE14  SLE15   SLE2   SLE3   SLE4   SLE5   HC10   HC16   HC17   HC18
1   2.355  2.517  5.980  4.244  4.817  1.841  2.922  2.617  6.485 10.721  5.776  6.014  5.907  5.365
2   0.785  0.559  0.292  0.935  0.446  0.775  0.627  0.236  0.715  1.022  0.313  0.462  0.400  0.705
3   0.740  0.683  0.368  0.578  0.568  0.413  0.943  2.605  1.137  0.647  0.562  0.572  0.641  0.299
4  82.898 74.776 77.242 63.202 63.228 90.177 58.566 66.551 51.495 61.380 49.636 47.626 52.586 51.145
5   0.316  0.390  0.099  0.227  0.287  0.092  0.182  0.285  0.147  0.141  0.172  0.201  0.129  0.092
6   0.141  0.102  0.447  0.437  0.133  0.006  0.478  0.173  1.056  0.398  1.076  1.133  1.041  0.781
7   0.474  0.294  0.250  0.053  0.332  0.058  1.398  0.398  2.902  3.079  0.481  0.048  0.090  0.019
8   0.153  0.144  0.163  0.170  0.147  0.177  0.139  0.169  0.138  0.148  0.129  0.145  0.161  0.158
9   0.378  0.206  0.214  0.375  0.064  0.099  0.667  0.054  1.094  1.516  1.073  1.028  0.648  0.574
10  0.075  0.051  0.009  0.017  0.023  0.036  0.032  0.047  0.117  0.153  0.074  0.056  0.027  0.033
11  0.176  0.146  0.046  0.274  0.215  0.101  0.197  0.107  0.390  0.514  0.212  0.258  0.207  0.258
12  0.232  0.099  0.023  0.076  0.069  0.085  0.195  0.131  0.137  0.165  0.096  0.096  0.060  0.099
13  0.536  0.668  1.626  1.359  0.056  0.158  1.233  0.364  0.505  0.928  3.427  1.948  1.663  1.156
14  0.536  0.404  0.111  0.455  0.285  0.488  0.399  0.350  0.299  0.425  0.328  0.351  0.299  0.425
15  1.369  0.264  0.017  0.321  0.285  0.331  1.092  0.756  0.844  0.990  0.665  0.220  0.188  0.292
16  0.634  1.141  0.171  1.341  1.079  1.422  0.492  0.460  0.524  0.550  0.359  0.962  0.954  1.234
17  0.450  0.831  0.132  0.381  0.200  0.303  0.309  0.335  0.218  0.278  0.168  0.250  0.208  0.214
18  0.111  0.084  0.018  0.213  0.083  0.037  0.229  0.077  0.173  0.186  0.154  0.152  0.110  0.193
19  0.260  0.140  0.526  0.137  0.263  0.127  0.208  0.358  0.604  0.261  0.241  0.086  0.121  0.136
```

Running statistical testing on the frequency table, across conditions

```
# The full statistics wrapper
pvalues <- StatsWrapper(conds = comp_conds,
                        counts = counts,
                        test_type = test,
                        paired = paired_test,
                        fdr = use_fdr)
```

Define conditions of interest, stat test of interest, whether to use FDR
Later versions use regression-based modeling with EdgeR (package: Diffcyt)

The point: just get your data into this frequency table format

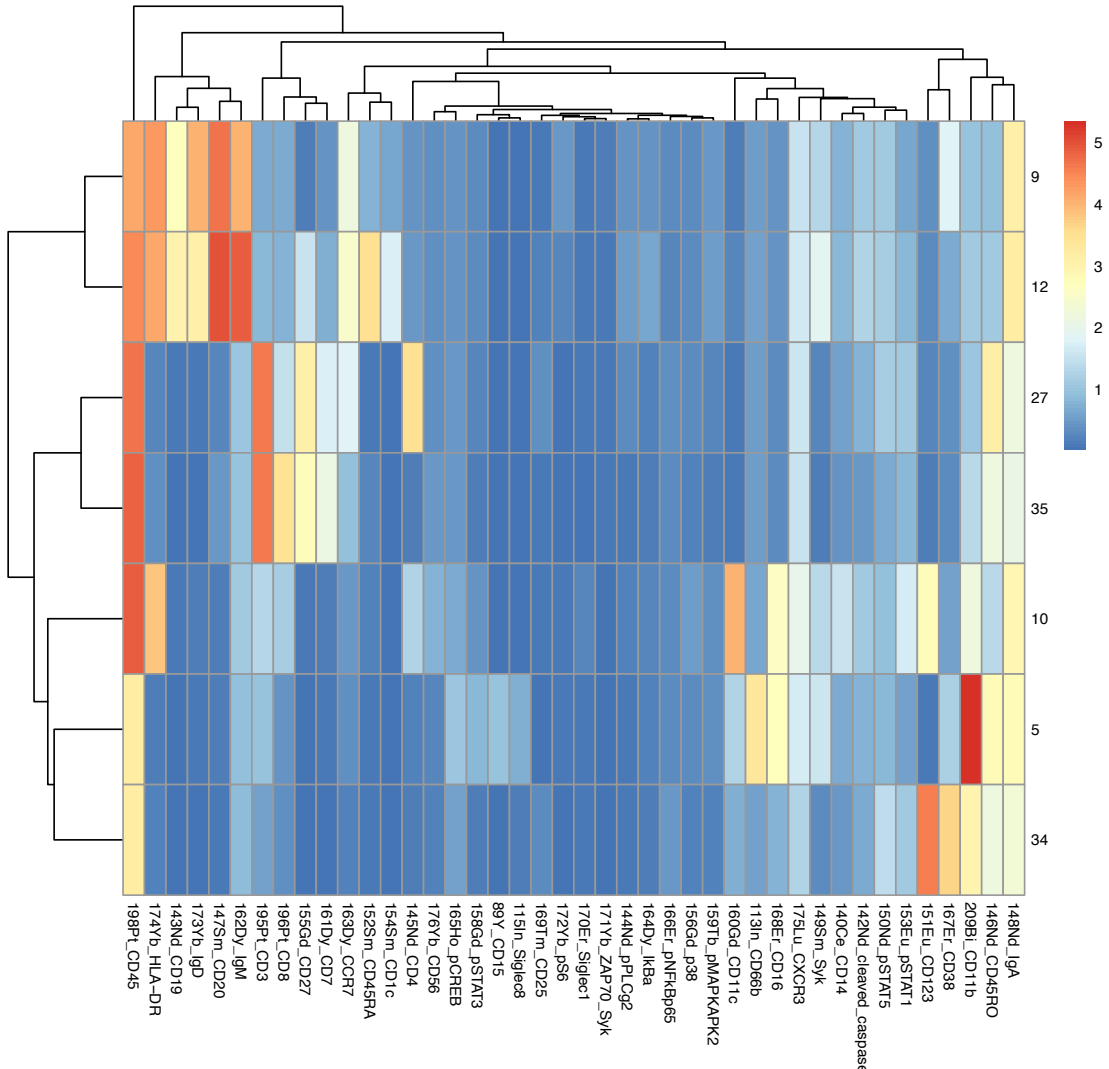
```
> counts
     SLE1  SLE11  SLE12  SLE13  SLE14  SLE15   SLE2   SLE3   SLE4   SLE5   HC10   HC16   HC17   HC18
1   2.355  2.517  5.980  4.244  4.817  1.841  2.922  2.617  6.485 10.721  5.776  6.014  5.907  5.365
2   0.785  0.559  0.292  0.935  0.446  0.775  0.627  0.236  0.715  1.022  0.313  0.462  0.400  0.705
3   0.740  0.683  0.368  0.578  0.568  0.413  0.943  2.605  1.137  0.647  0.562  0.572  0.641  0.299
4  82.898 74.776 77.242 63.202 63.228 90.177 58.566 66.551 51.495 61.380 49.636 47.626 52.586 51.145
5   0.316  0.390  0.099  0.227  0.287  0.092  0.182  0.285  0.147  0.141  0.172  0.201  0.129  0.092
6   0.141  0.102  0.447  0.437  0.133  0.006  0.478  0.173  1.056  0.398  1.076  1.133  1.041  0.781
7   0.474  0.294  0.250  0.053  0.332  0.058  1.398  0.398  2.902  3.079  0.481  0.048  0.090  0.019
8   0.153  0.144  0.163  0.170  0.147  0.177  0.139  0.169  0.138  0.148  0.129  0.145  0.161  0.158
9   0.378  0.206  0.214  0.375  0.064  0.099  0.667  0.054  1.094  1.516  1.073  1.028  0.648  0.574
10  0.075  0.051  0.009  0.017  0.023  0.036  0.032  0.047  0.117  0.153  0.074  0.056  0.027  0.033
11  0.176  0.146  0.046  0.274  0.215  0.101  0.197  0.107  0.390  0.514  0.212  0.258  0.207  0.258
12  0.232  0.099  0.023  0.076  0.069  0.085  0.195  0.131  0.137  0.165  0.096  0.096  0.060  0.099
13  0.536  0.668  1.626  1.359  0.056  0.158  1.233  0.364  0.505  0.928  3.427  1.948  1.663  1.156
14  0.536  0.404  0.111  0.455  0.285  0.488  0.399  0.350  0.299  0.425  0.328  0.351  0.299  0.425
15  1.369  0.264  0.017  0.321  0.285  0.331  1.092  0.756  0.844  0.990  0.665  0.220  0.188  0.292
16  0.634  1.141  0.171  1.341  1.079  1.422  0.492  0.460  0.524  0.550  0.359  0.962  0.954  1.234
17  0.450  0.831  0.132  0.381  0.200  0.303  0.309  0.335  0.218  0.278  0.168  0.250  0.208  0.214
18  0.111  0.084  0.018  0.213  0.083  0.037  0.229  0.077  0.173  0.186  0.154  0.152  0.110  0.193
19  0.260  0.140  0.526  0.137  0.263  0.127  0.208  0.358  0.604  0.261  0.241  0.086  0.121  0.136
```

Output: p values ordered by cluster

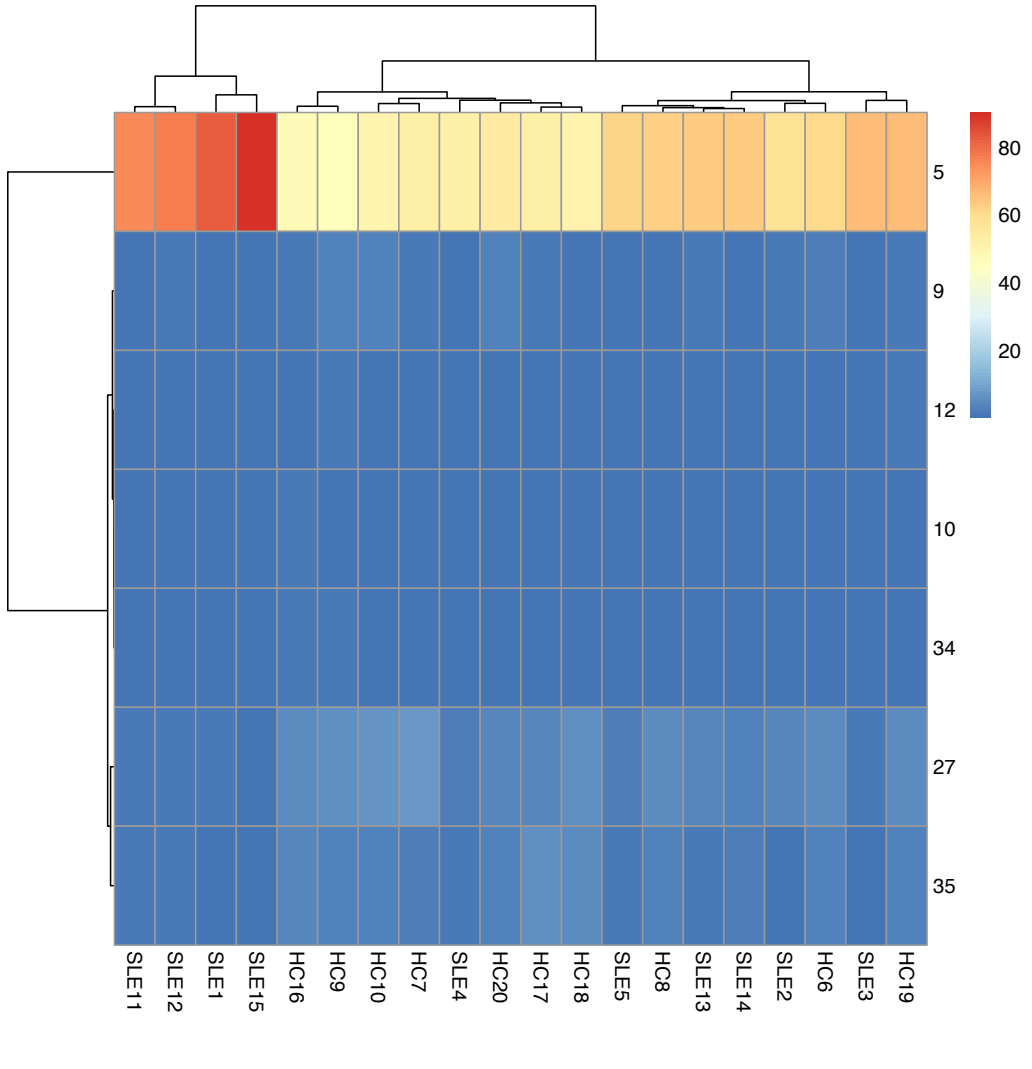
```
> as.data.frame(pvalues)
         t.SLE
1  0.486638136
2  0.406943432
3  0.241516597
4  0.025389661
5  0.092914460
6  0.004701097
7  0.213056814
8  0.364272324
9  0.191262148
10 0.486638136
11 0.698766900
12 0.642796844
13 0.004701097
14 0.877846761
15 0.698766900
16 0.877846761
17 0.156628859
18 0.086690006
19 0.092914460
```

What to do with your statistical output: heatmaps

Significant clusters, per-cluster marker expression

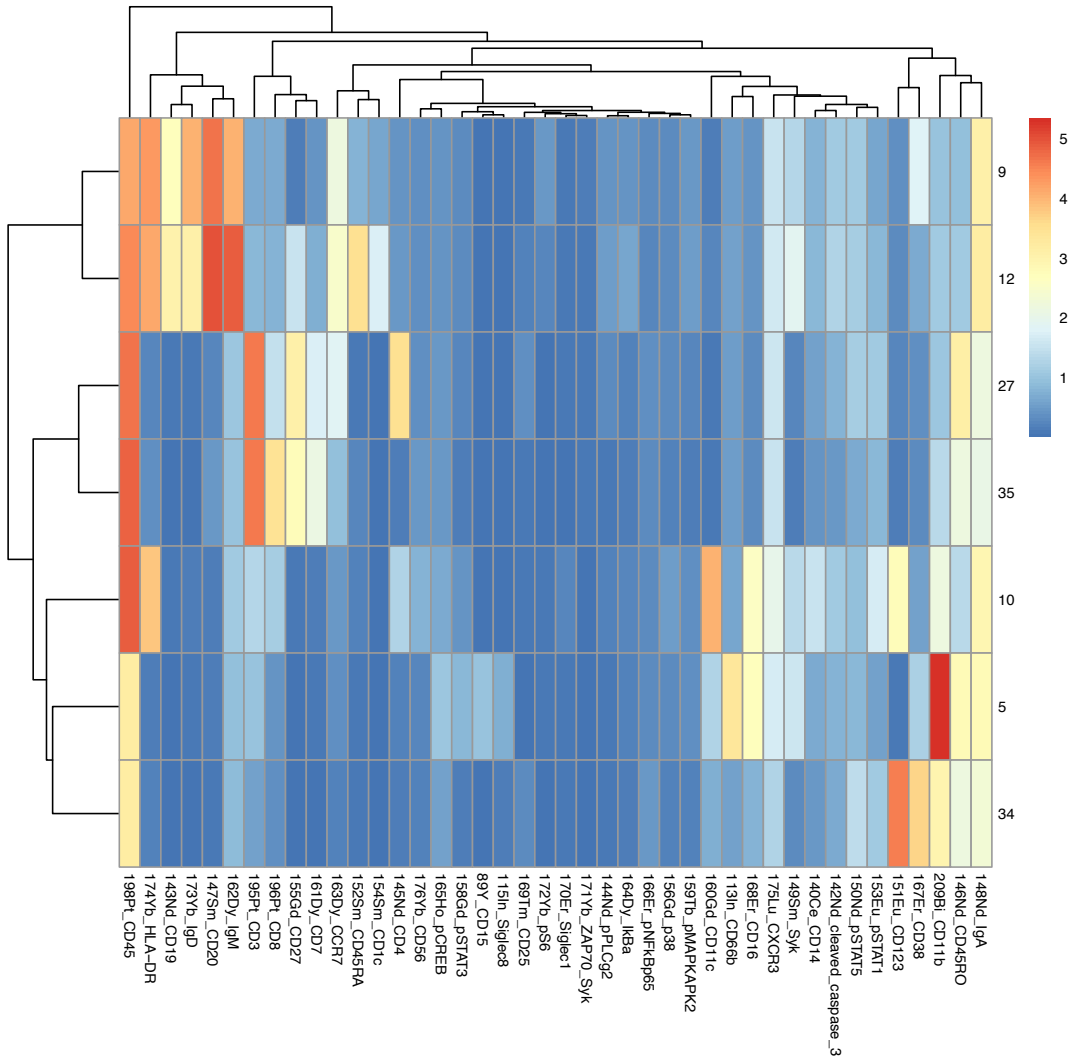


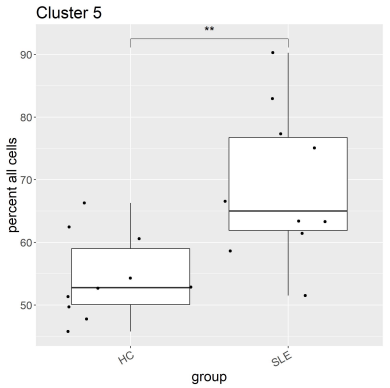
Significant clusters, per-marker frequency



What to do with your statistical output: plots

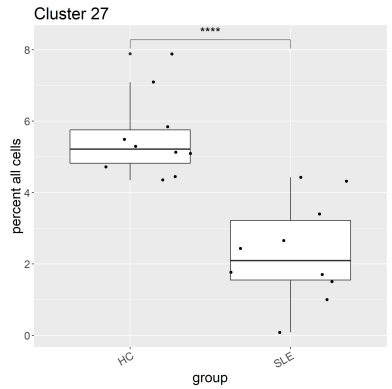
Significant clusters, per-marker expression

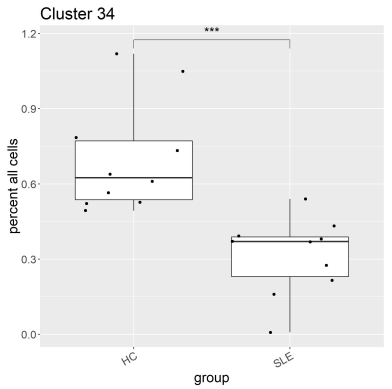


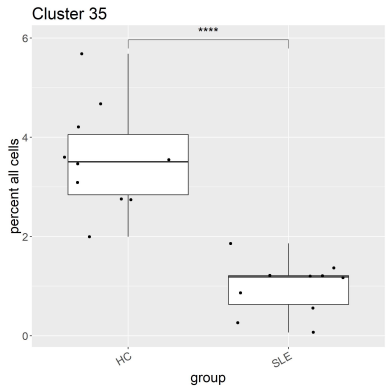


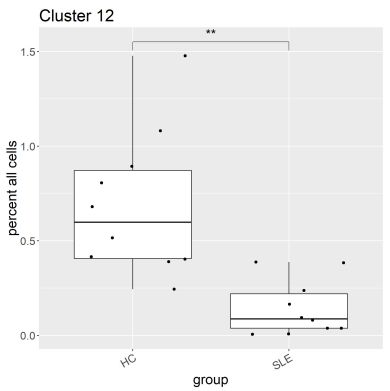
Package: ggplot2, ggsignif, gginnards

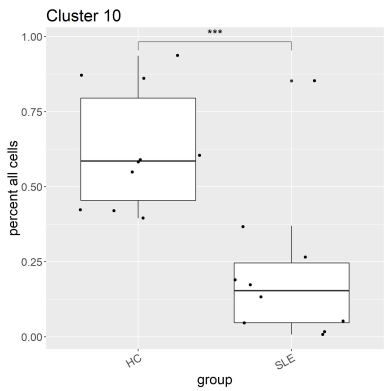
```
# Production and saving of plots
stat_plots <- PlotWrapper(Sig_rows = sig_rows,
                          Counts = counts,
                          test = stat_test,
                          fdr = fdr_adj,
                          Add_p = add_p,
                          to_save = TRUE,
                          Comp_conds = comp_conds,
                          Control_cond = control_cond)
```

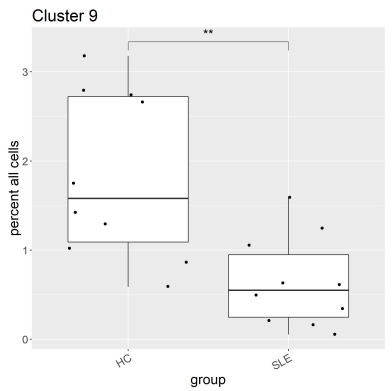












How to use dimension reduction effectively: visualizing output per-cell

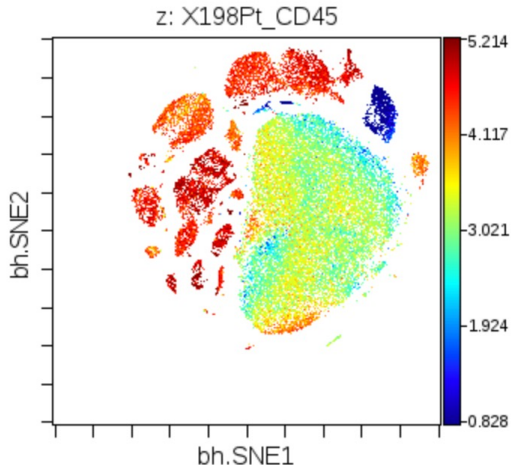
```
# Places the pvalues into the subsampled cell data object
final <- PvaluesToCells(stat_output = pvalues,
                        cells = sub_cells,
                        clusters = sub_cells[[names(sub_cells)[grep("cluster", names(sub_cells))]]],
                        log_transform = TRUE)
```

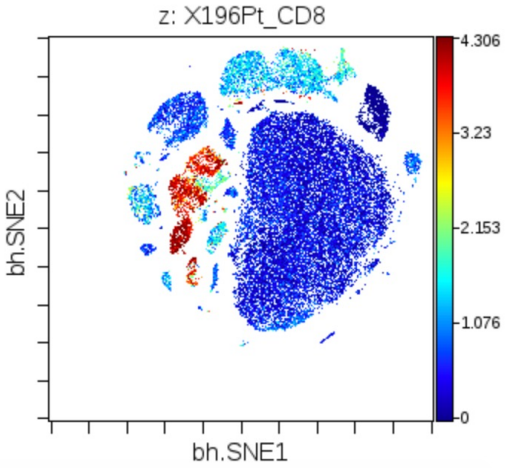
```
# Performs t-SNE to a desired number of cells
final <- Sconify:::AddTsne(dat = final, input = surface)
write.csv(final, paste("final_output_t_test", ncells, "csv", sep = "."))
```

Cytobank Premium

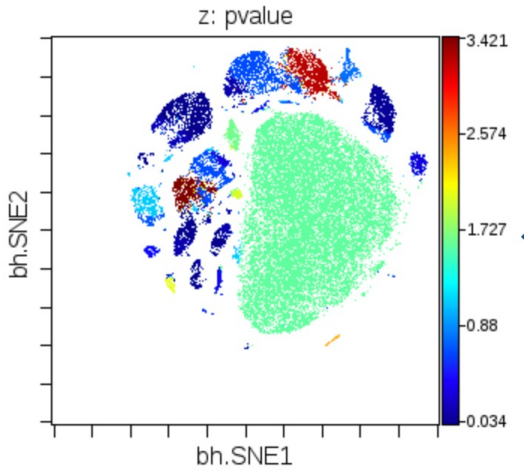
(or your favorite per-cell visualization tool)

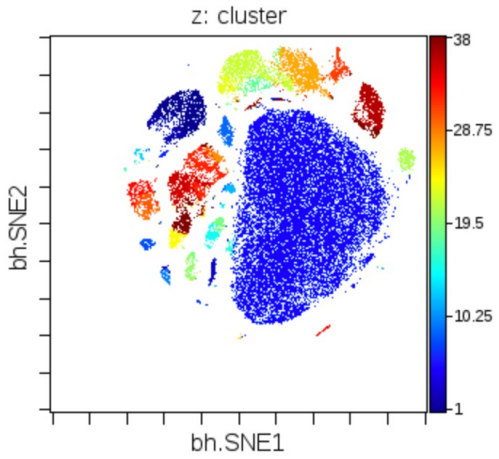
Try visualizing distance to cluster centroid relative to the nearest neighboring cluster centroid

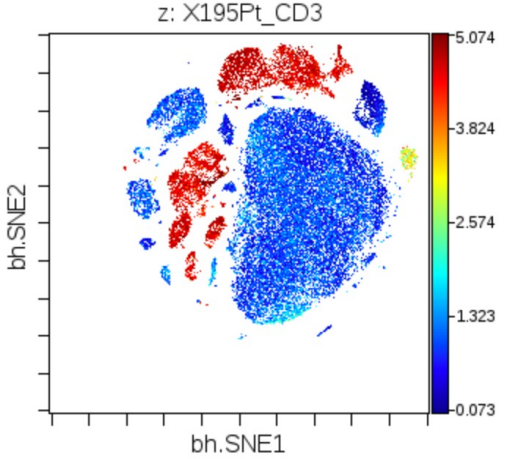


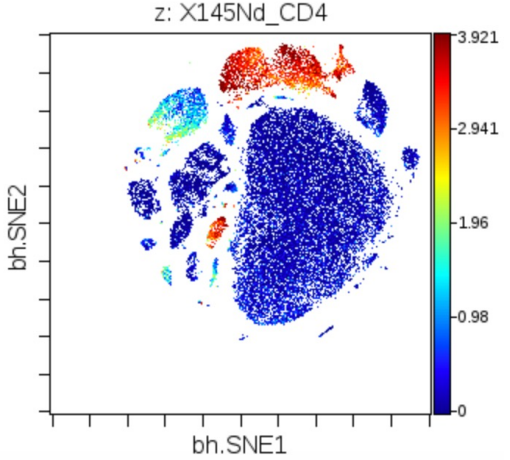


-log10 * pvalue



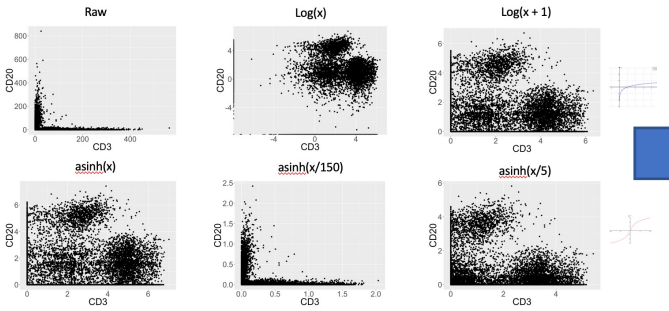






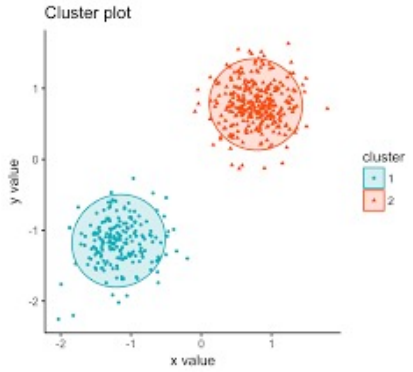
CyTOF analysis: general principles relevant to the DRFZ

Pre-processing



Package: FlowCore
Manual gating tools:
FlowJo, Cytobank

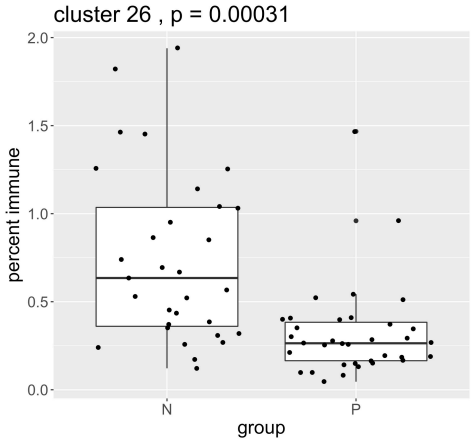
Grouping



Package: FlowSOM

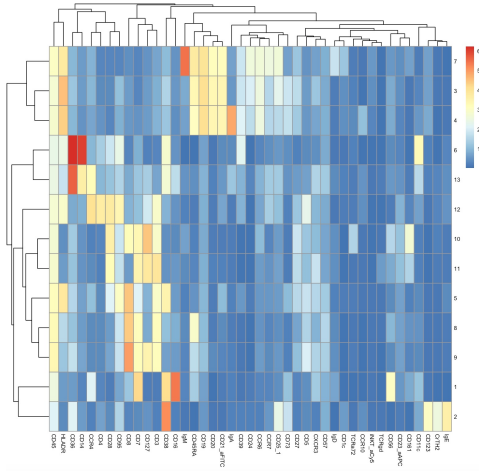
Others to try: Phenograph,
X-shift

Per-group statistics



Package: ggplot2, ggsignif,
gginnards

Visualizations



Package: pheatmap,
Rtsne
Others to try: UMAP

Visualization tools:
FlowJo, Cytobank

Key takeaways from the pipeline

- Make an expression matrix of cells, including File ID.
- Whatever method I do, I add to the matrix above (cluster ID, p-values)
  - Makes the pipeline robust to new tools
- I visualize my data and results by any possible means: plots, heatmaps, dimension reduction, etc
- I always check my results at every step, to make sure they make sense (Interactive programming languages are good for this)

Thank You!

Department of Immune Monitoring

Andreas Grützkau

Marie Urbicht

Department of Mass Cytometry

Henrik Mei

Axel Schulz

Sabine Baumgart

Heike Hirseland

Silke Stanislawiak

Antonia Niedobitek

Julia Schulze

Edward Rullmann

Sarah Gräßle

Eva Holzhäuser

GerMaNet
German Mass Cytometry Network

DRFZ BERLIN
Deutsches Rheuma-Forschungszentrum
Ein Institut der Leibniz-Gemeinschaft