August 9, 2017

# Scanpy for analysis of large-scale single-cell gene expression data

F. Alexander Wolf[1,†], Philipp Angerer[1] & Fabian J. Theis[1,2,‡]
**1** Helmholtz Zentrum München – German Research Center for Environmental Health, Institute of Computational Biology, Neuherberg, Munich, Germany.
**2** Department of Mathematics, Technische Universität München, Munich, Germany.
† falexwolf.de ‡ fabian.theis@helmholtz-muenchen.de

We present Scanpy, a scalable toolkit for analyzing single-cell gene expression data. It includes preprocessing, visualization, clustering, pseudotime and trajectory inference, differential expression testing and simulation of gene regulatory networks. The Python-based implementation efficiently deals with datasets of more than one million cells and enables easy interfacing of advanced machine learning packages. Code is available from `https://github.com/theislab/scanpy`.

Simple integrated analysis workflows for single-cell transcriptomic data (Stegle *et al.*, 2015) have been enabled by frameworks such as Seurat (Satija *et al.*, 2015), MAST (Finak *et al.*, 2015), Monocle (Trapnell *et al.*, 2012), Scater (McCarthy *et al.*, 2017), Cell Ranger (Zheng *et al.*, 2017), Scran (Lun *et al.*, 2016) and SCDE (Kharchenko *et al.*, 2014). However, they do not scale to the increasingly available large-scale datasets with up to one million cells. Here, we present a framework that overcomes this limitation and provides similar analysis possibilities (Fig. 1a). In addition, in contrast to the existing R-based frameworks, Scanpy's Python-based implementation allows to easily integrate advanced machine learning packages, such as Tensorflow (Abadi *et al.*, 2015, Suppl. Note 1).

Scanpy provides preprocessing comparable to Seurat (Macosko *et al.*, 2015) and Cell Ranger (Zheng *et al.*, 2017) and visualization through tSNE (Coifman *et al.*, 2005; Amir *et al.*, 2013), graph-drawing (Fruchterman and Reingold, 1991; Csardi and Nepusz, 2006; Weinreb *et al.*, 2017), Diffusion Maps (Coifman *et al.*, 2005; Haghverdi *et al.*, 2015; Angerer *et al.*, 2015) and principal component analysis (Fig. 1a). It provides clustering similar to Phenograph (Blondel *et al.*, 2008; Levine *et al.*, 2015) and allows identifying clusters with cell types by finding marker genes using differential expression testing. Scanpy provides pseudotemporal-ordering and the reconstruction of branching trajectories via Diffusion Pseudotime (DPT, Haghverdi *et al.*, 2016)[1] and allows simulating single cells governed by gene regulatory networks (Suppl. Note 2, Wittmann *et al.*, 2009). Scanpy provides its tools with speedups between 4 and 16 and much higher memory efficiency (about a factor 10) than comparable frameworks (Fig. 1b, Suppl. Note 2). This enables the analysis of datasets with over a million cells and allows an *interactive* analysis of about hundred thousand cells (Fig. 1c, Suppl. Note 2).

Scanpy is implemented in a highly modular fashion and can hence be easily further developed by a community (Suppl. Note 3). Its data storage formats and objects allow a simple cross-language and cross-platform transfer of results (Suppl. Note 3). Scanpy integrates well into the existing Python ecosystem, where no comparable toolkit yet exists (Suppl. Note 4).

## Acknowledgements

We thank the authors of Seurat for sharing their great tutorials. We thank S. Tritschler, L. Simon, D. S. Fischer, F. Buettner and G. Eraslan for commenting on the software package and the paper. F.A.W. acknowledges support by the Helmholtz Postdoc Programme, Initiative and Networking Fund of the Helmholtz Association.

[1] DPT compares favorably (Qiu *et al.*, 2017) with Monocle 2 (Trapnell *et al.*, 2014; Qiu *et al.*, 2017), Wanderlust (Bendall *et al.*, 2014) and Wishbone (Setty *et al.*, 2016)

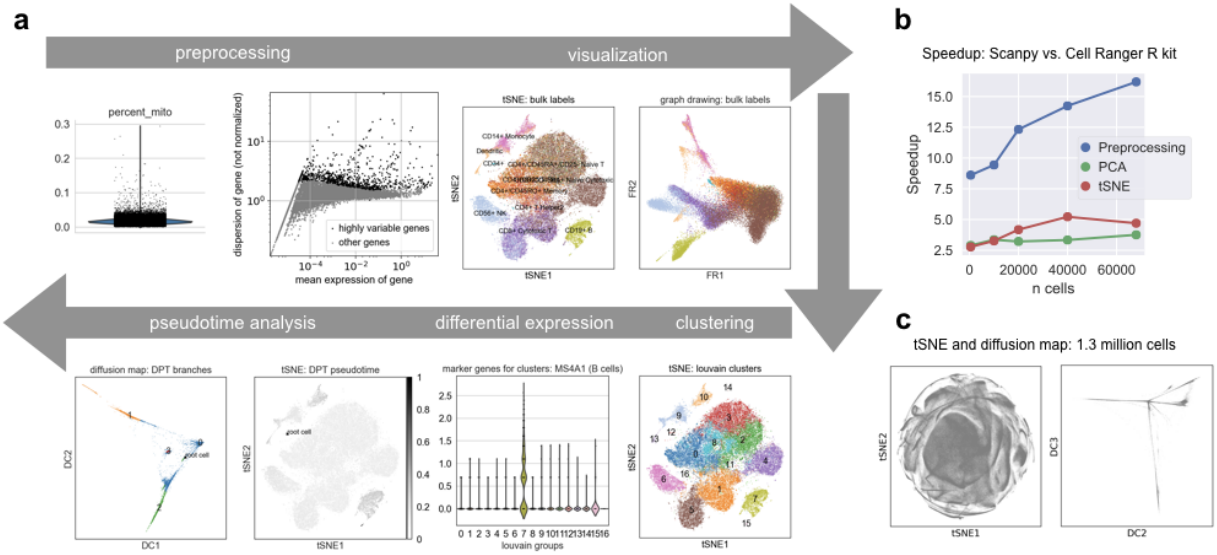


**Figure 1 | a, Scanpy's analysis features.** Using the example of 68,579 PBMC cells of Zheng *et al.* (2017). Regressing-out confounding variables, normalizing and identifying highly-variable genes. tSNE and graph-drawing (Fruchterman-Reingold) visualizations show cell types obtained by comparing with bulk expression (Zheng *et al.*, 2017). Louvain clustering. Ranking differentially-expressed genes in clusters identifies the MS4A1 marker gene for B cells in cluster 7, which agrees with the bulk labels. Pseudotemporal ordering from a root cell in the CD34+ cluster and detection of a branching trajectory, visualized with tSNE and Diffusion Maps. **b, Speedup over Cell Ranger R kit.** For representative steps of the analysis (Zheng *et al.*, 2017). **c, Visualizing 1.3 million cells.** Using tSNE and Diffusion Maps. The data, brain cells from E18 mice, is publicly available from 10X Genomics.

## Supplemental Note 1: Scanpy's technological foundations

Scanpy is on the Python packaging index: `https://pypi.python.org/pypi/scanpy`.

Scanpy's Python-based implementation allows easily interfacing advanced machine learning packages such as Tensorflow (Abadi *et al.*, 2015) for Deep Learning (LeCun *et al.*, 2015), Limix for linear mixed models (Lippert *et al.*, 2014) and GPy/GPflow for Gaussian Processes (GPy, 2012; Matthews *et al.*, 2017). See Suppl. Note 2 for an example of combining Deep Learning and Scanpy (Eulenberg *et al.*, 2016).

Scanpy's core relies on Numpy (van der Walt *et al.*, 2011), Scipy (Jones *et al.*, 2001), Matplotlib (Hunter, 2007) and h5py (Collette, 2013). Parts of the toolkit rely on Pandas (McKinney, 2010), scikit-learn (Pedregosa *et al.*, 2011), statsmodels (Seabold and Perktold, 2010), Seaborn (Waskom *et al.*, 2016), NetworkX (Hagberg *et al.*, 2008), igraph (Csardi and Nepusz, 2006), the tSNE package of Ulyanov (2016) and the Louvain clustering package of Traag (2017).

## Supplemental Note 2: Scanpy's analysis features

The following links allow to reproduce Figure 1, give detailed background information on the benchmark computations and provide further use cases.

- The analysis of 68,579 PBMC cells of Figure 1 and the comparison with the Cell Ranger R kit (Zheng *et al.*, 2017): scanpy_usage/170503_zheng17.
- A detailed clustering tutorial, adapted from Seurat's tutorial, walks the user from raw data through all steps of the analysis to the identification of cell types: scanpy_usage/170505_seurat.
- Visualizing 1.3 mio cells as in Figure 1c: scanpy_usage/170522_visualizing_one_million_cells.

- Examples for reconstructing branching processes via Diffusion Pseudotime (Haghverdi *et al.*, 2016): scanpy_usage/170502_haghverdi16.
- Simulating single cells using gene regulatory networks (Wittmann *et al.*, 2009); here, myeloid differentiation (Krumsiek *et al.*, 2011): scanpy_usage/170430_krumsiek11.
- Analyzing deep learning results for single-cell images (Eulenberg *et al.*, 2016): scanpy_usage/170529_images.

## Supplemental Note 3: Scanpy's technological concepts

Scanpy tools operate on a class *AnnData*, which simply stores the annotated data matrix. While Scanpy is in large parts object oriented, by building *AnnData*, we chose a functional-programming oriented design to enable a modular development of Scanpy: adding new functionality to the toolkit is easy as any new tool leaves the structure of *AnnData* unaffected. *AnnData* is similar to R's ExpressionSet (Huber *et al.*, 2015), but supports sparse data and file iterators and provides simple control of the underlying data types (van der Walt *et al.*, 2011). In addition, *AnnData*'s simple structure allowed us to design a corresponding *hdf5* file format (Collette, 2013), which enables writing and reading objects to disk in a highly efficient and platform-, framework- and language-independent way. This allows easily transferring data and analysis results from and to existing R packages (see also Suppl. Note 3).

Further technological concepts are as follows.

- Support of reading a wide variety of data file formats and their simple cache in fast *hdf5* files; similar to caching full *AnnData* objects.
- A central class *DataGraph* whose focus is the efficient representation of a graph of neighborhood relations in data; their computation is parallelized and much faster than in existing packages (Pedregosa *et al.*, 2011). The class provides functions to compute quantities on the graph, which are not available in other graph packages (Hagberg *et al.*, 2008; Csardi and Nepusz, 2006). Storage is again platform- and language-independent via CSR sparse matrices, which appear as data annotation in *AnnData*.
- Scanpy functions by default operate "inplace" and thereby encourage and enable easily building memory-efficient pipelines.
- Computations are monitored by profiling information so that users develop an intuition for waiting times. In addition, this encourages performance-aware development.
- A modular design of the toolkit with user submodules for *preprocessing*, *tools*, *plotting*, *settings* and correspondence in naming conventions between the modules.
- A command-line interface that parallels the usage of the API allows easily submitting jobs to remote computing infrastructure.

Just before submission of this manuscript we became aware of an alternative approach to tackling large-scale data in statistical computing. Lun *et al.* (2017) provide a C++ library that simplifies interfacing large-scale matrices for R-package developers. This approach is therefore an alternative to only a small subset of Scanpy's features — interfacing hdf5-backed large-scale matrices.

## Supplemental Note 4: Python packages for single-cell analysis

Aside from the highly popular scLVM (Buettner *et al.*, 2015, 2016), which uses Gaussian Process latent variable models for inferring hidden sources of variation, there are, among others, the visualization frameworks FastProject (DeTomaso and Yosef, 2016), ACCENSE (Shekhar *et al.*, 2013)

and SPRING (Weinreb *et al.*, 2017),[2] the trajectory inference tool SCIMITAR, the clustering tool PhenoGraph (Levine *et al.*, 2015), the single-cell experiment design tool MIMOSCA (Dixit *et al.*, 2016), the tree-inference tool ECLAIR (Giecold *et al.*, 2016) and the framework flotilla, which comes with modules for simple visualization, simple clustering and differential expression testing. Hence, only the latter provides a data analysis framework that solves more than one specific task. In contrast to Scanpy, however, flotilla is neither targeted at single-cell nor at large-scale data and does not provide any graph-based methods, which build the core of Scanpy. Also, flotilla is built around a complicated class *Study* that contains data, tools and plotting functions, which orthogonal to the design choice of Scanpy, which is built around a simple class *AnnData* and hence easily extendable (Suppl. Note 3).

## References

Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng (2015), *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, Software available from tensorflow.org.

Amir, E.-a. D., K. L. Davis, M. D. Tadmor, E. F. Simonds, J. H. Levine, S. C. Bendall, D. K. Shenfeld, S. Krishnaswamy, G. P. Nolan, and D. Pe'er (2013), *viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia*, Nature Biotechnology **31**, 545.

Angerer, P., L. Haghverdi, M. Buettner, F. Theis, C. Marr, and F. Buettner (2015), *destiny – diffusion maps for large-scale single-cell data in R*, Bioinformatics **32**, 1241.

Bendall, S. C., K. L. Davis, E.-a. D. Amir, M. D. Tadmor, E. F. Simonds, T. J. Chen, D. K. Shenfeld, G. P. Nolan, and D. Pe'er (2014), *Single-Cell Trajectory Detection Uncovers Progression and Regulatory Coordination in Human B Cell Development*, Cell **157**, 714.

Blondel, V. D., J.-L. Guillaume, R. Lambiotte, and E. Lefebvre (2008), *Fast unfolding of communities in large networks*, J. Stat. Mech. **2008**, P10008.

Buettner, F., K. N. Natarajan, F. P. Casale, V. Proserpio, A. Scialdone, F. J. Theis, S. A. Teichmann, J. C. Marioni, and O. Stegle (2015), *Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells*, Nature Biotechnology **33**, 155.

Buettner, F., N. Pratanwanich, J. C. Marioni, and O. Stegle (2016), *Scalable latent-factor models applied to single-cell RNA-seq data separate biological drivers from confounding effects*, bioRxiv doi: 10.1101/087775.

Coifman, R. R., S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker (2005), *Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps*, Proceedings of the National Academy of Sciences **102**, 7426.

Collette, A. (2013), *Python and HDF5* (O'Reilly).

Csardi, G., and T. Nepusz (2006), *The igraph software package for complex network research*, InterJournal Complex Systems **2006**, 1695.

DeTomaso, D., and N. Yosef (2016), *FastProject: a tool for low-dimensional analysis of single-cell RNA-Seq data*, BMC Bioinformatics **17**, 315.

[2] The latter uses the JavaScript package D3.js for the actual visualization and Python only for preprocessing.

Dixit, A., O. Parnas, B. Li, J. Chen, C. P. Fulco, L. Jerby-Arnon, N. D. Marjanovic, D. Dionne, T. Burks, R. Raychowdhury, B. Adamson, T. M. Norman, E. S. Lander, J. S. Weissman, N. Friedman, and A. Regev (2016), *Perturb-Seq: Dissecting Molecular Circuits with Scalable Single-Cell RNA Profiling of Pooled Genetic Screens.*, Cell **167**, 1853.

Eulenberg, P., N. Köhler, T. Blasi, A. Filby, A. E. Carpenter, P. Rees, F. J. Theis, and F. A. Wolf (2016), *Deep Learning for Imaging Flow Cytometry: Cell Cycle Analysis of Jurkat Cells*, bioRxiv, accepted in Nat. Comms. doi: 10.1101/081364.

Finak, G., A. McDavid, M. Yajima, J. Deng, V. Gersuk, A. K. Shalek, C. K. Slichter, H. W. Miller, M. J. McElrath, M. Prlic, and et al. (2015), *MAST: A flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA-seq data.*, bioRxiv doi: 10.1101/020842.

Fruchterman, T. M. J., and E. M. Reingold (1991), *Graph drawing by force-directed placement*, Software: Practice and Experience **21**, 1129.

Giecold, G., E. Marco, S. P. Garcia, L. Trippa, and G.-C. Yuan (2016), *Robust lineage reconstruction from high-dimensional single-cell data*, Nucleic acids research **44**, e122.

GPy, (2012), *GPy: A Gaussian process framework in python*.

Hagberg, A. A., D. A. Schult, and P. J. Swart (2008), *Exploring network structure, dynamics, and function using NetworkX*, in *Proceedings of the 7th Python in Science Conference (SciPy2008)* (Pasadena, CA USA) pp. 11–15.

Haghverdi, L., F. Buettner, and F. J. Theis (2015), *Diffusion maps for high-dimensional single-cell analysis of differentiation data*, Bioinformatics **31**, 2989.

Haghverdi, L., M. Büttner, F. A. Wolf, F. Buettner, and F. J. Theis (2016), *Diffusion pseudotime robustly reconstructs branching cellular lineages*, Nature Methods **13**, 845.

Huber, W., V. J. Carey, R. Gentleman, S. Anders, M. Carlson, B. S. Carvalho, H. C. Bravo, S. Davis, L. Gatto, T. Girke, R. Gottardo, F. Hahne, K. D. Hansen, R. A. Irizarry, M. Lawrence, M. I. Love, J. MacDonald, V. Obenchain, A. K. Oleś, H. Pagès, A. Reyes, P. Shannon, G. K. Smyth, D. Tenenbaum, L. Waldron, and M. Morgan (2015), *Orchestrating high-throughput genomic analysis with Bioconductor*, Nature Methods **12**, 115.

Hunter, J. D. (2007), *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering **9**, 90.

Jones, E., T. Oliphant, P. Peterson, *et al.* (2001), *SciPy: Open source scientific tools for Python*.

Kharchenko, P. V., L. Silberstein, and D. T. Scadden (2014), *Bayesian approach to single-cell differential expression analysis*, Nature Methods **11**, 740.

Krumsiek, J., C. Marr, T. Schroeder, and F. J. Theis (2011), *Hierarchical Differentiation of Myeloid Progenitors Is Encoded in the Transcription Factor Network*, PLoS ONE **6**, e22649.

LeCun, Y., Y. Bengio, and G. Hinton (2015), *Deep learning*, Nature **521**, 436.

Levine, J. H., E. F. Simonds, S. C. Bendall, K. L. Davis, E.-a. D. Amir, M. D. Tadmor, O. Litvin, H. G. Fienberg, A. Jager, E. R. Zunder, R. Finck, A. L. Gedman, I. Radtke, J. R. Downing, D. Pe'er, and G. P. Nolan (2015), *Data-Driven Phenotypic Dissection of AML Reveals Progenitor–like Cells that Correlate with Prognosis*, Cell **162**, 184.

Lippert, C., F. P. Casale, B. Rakitsch, and O. Stegle (2014), *LIMIX: genetic analysis of multiple traits*, bioRxiv doi: 10.1101/003905.

Lun, A., D. McCarthy, and J. Marioni (2016), *A step-by-step workflow for low-level analysis of single-cell RNA-seq data with Bioconductor [version 2; referees: 3 approved, 2 approved with reservations]*, F1000Research **5**, doi: 10.12688/f1000research.9501.2.

Lun, A. T. L., H. Pagès, and M. L. Smith (2017), *beachmat: a Bioconductor C++ API for accessing single-cell genomics data from a variety of R matrix types*, bioRxiv doi: 10.1101/167445.

Macosko, E. Z., A. Basu, R. Satija, J. Nemesh, K. Shekhar, M. Goldman, I. Tirosh, A. R. Bialas, N. Kamitaki, E. M. Martersteck, and et al. (2015), *Highly Parallel Genome-wide Expression Profiling of Individual Cells Using Nanoliter Droplets*, Cell **161**, 1202.

Matthews, A. G. d. G., M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman (2017), *GPflow: A Gaussian process library using TensorFlow*, Journal of Machine Learning Research **18**(40), 1.

McCarthy, D., Q. Wills, and K. Campbell (2017), *scater: Single-cell analysis toolkit for gene expression data in R*, Bioinformatics **33**, 1179.

McKinney, W. (2010), *Data Structures for Statistical Computing in Python*, in *Proceedings of the 9th Python in Science Conference*, edited by S. van der Walt and J. Millman, pp. 51 – 56.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay (2011), *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research **12**, 2825.

Qiu, X., Q. Mao, Y. Tang, L. Wang, R. Chawla, H. Pliner, and C. Trapnell (2017), *Reversed graph embedding resolves complex single-cell developmental trajectories*, bioRxiv doi: 10.1101/110668.

Satija, R., J. A. Farrell, D. Gennert, A. F. Schier, and A. Regev (2015), *Spatial reconstruction of single-cell gene expression data*, Nature Biotechnology **33**, 495.

Seabold, S., and J. Perktold (2010), *Statsmodels: Econometric and statistical modeling with python*, in *9th Python in Science Conference*.

Setty, M., M. D. Tadmor, S. Reich-Zeliger, O. Angel, T. M. Salame, P. Kathail, K. Choi, S. Bendall, N. Friedman, and D. Pe'er (2016), *Wishbone identifies bifurcating developmental trajectories from single-cell data*, Nature Biotechnology **34**, 637.

Shekhar, K., P. Brodin, M. M. Davis, and A. K. Chakraborty (2013), *Automatic Classification of Cellular Expression by Nonlinear Stochastic Embedding (ACCENSE)*, Proceedings of the National Academy of Sciences **111**, 202.

Stegle, O., S. A. Teichmann, and J. C. Marioni (2015), *Computational and analytical challenges in single-cell transcriptomics*, Nature Reviews Genetics **16**, 133.

Traag, V. (2017), *Louvain*, GitHub doi: 10.5281/zenodo.35117.

Trapnell, C., D. Cacchiarelli, J. Grimsby, P. Pokharel, S. Li, M. Morse, N. J. Lennon, K. J. Livak, T. S. Mikkelsen, and J. L. Rinn (2014), *The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells*, Nature Biotechnology **32**, 381.

Trapnell, C., A. Roberts, L. Goff, G. Pertea, D. Kim, D. R. Kelley, H. Pimentel, S. L. Salzberg, J. L. Rinn, and L. Pachter (2012), *Differential gene and transcript expression analysis of RNA-seq experiments with TopHat and Cufflinks*, Nature Protocols **7**, 562.

Ulyanov, D. (2016), *Multicore-TSNE*.

van der Walt, S., S. C. Colbert, and G. Varoquaux (2011), *The NumPy Array: A Structure for Efficient Numerical Computation*, Computing in Science & Engineering **13**, 22.

Waskom, M., O. Botvinnik, drewokane, P. Hobson, David, Y. Halchenko, S. Lukauskas, J. B. Cole, J. Warmenhoven, J. de Ruiter, S. Hoyer, J. Vanderplas, S. Villalba, G. Kunter, E. Quintero, M. Martin, A. Miles, K. Meyer, T. Augspurger, T. Yarkoni, P. Bachant, M. Williams, C. Evans, C. Fitzgerald, Brian, D. Wehner, G. Hitz, E. Ziegler, A. Qalieh, and A. Lee (2016), *Seaborn*.

Weinreb, C., S. Wolock, and A. Klein (2017), *SPRING: a kinetic interface for visualizing high dimensional single-cell expression data*, bioRxiv doi: 10.1101/090332.

Wittmann, D. M., J. Krumsiek, J. Saez-Rodriguez, D. A. Lauffenburger, S. Klamt, and F. J. Theis (2009), *Transforming Boolean models to continuous models: methodology and application to T-cell receptor signaling*, BMC Syst. Biol. **3**, 98.

Zheng, G. X. Y., J. M. Terry, P. Belgrader, P. Ryvkin, Z. W. Bent, R. Wilson, S. B. Ziraldo, T. D. Wheeler, G. P. McDermott, J. Zhu, M. T. Gregory, J. Shuga, L. Montesclaros, J. G. Underwood, D. A. Masquelier, S. Y. Nishimura, M. Schnall-Levin, P. W. Wyatt, C. M. Hindson, R. Bharadwaj, A. Wong, K. D. Ness, L. W. Beppu, H. J. Deeg, C. McFarland, K. R. Loeb, W. J. Valente, N. G. Ericson, E. A. Stevens, J. P. Radich, T. S. Mikkelsen, B. J. Hindson, and J. H. Bielas (2017), *Massively parallel digital transcriptional profiling of single cells*, Nature Communications **8**, 14049.