Wolf *et al. Genome Biology* (2018) 19:15
https://doi.org/10.1186/s13059-017-1382-0

Genome Biology

SOFTWARE Open Access

SCANPY: large-scale single-cell gene expression data analysis

F. Alexander Wolf[1*], Philipp Angerer[1] and Fabian J. Theis[1,2*]

Abstract

SCANPY is a scalable toolkit for analyzing single-cell gene expression data. It includes methods for preprocessing, visualization, clustering, pseudotime and trajectory inference, differential expression testing, and simulation of gene regulatory networks. Its Python-based implementation efficiently deals with data sets of more than one million cells (https://github.com/theislab/Scanpy). Along with SCANPY, we present ANNDATA, a generic class for handling annotated data matrices (https://github.com/theislab/anndata).

Keywords: Single-cell transcriptomics, Machine learning, Scalability, Graph analysis, Clustering, Pseudotemporal ordering, Trajectory inference, Differential expression testing, Visualization, Bioinformatics

Background

Simple integrated analysis work flows for single-cell transcriptomic data [1] have been enabled by frameworks such as SEURAT [2], MONOCLE [3], SCDE/PAGODA [4], MAST [5], CELL RANGER [6], SCATER [7], and SCRAN [8]. However, these frameworks do not scale to the increasingly available large data sets with up to and more than one million cells. Here, we present a framework that overcomes this limitation and provides similar analysis possibilities. Moreover, in contrast to the existing R-based frameworks, SCANPY's Python-based implementation is easy to interface with advanced machine-learning packages, such as TENSORFLOW [9].

Results

SCANPY integrates canonical analysis methods in a scalable way

SCANPY integrates the analysis possibilities of established R-based frameworks and provides them in a scalable and modular form. Specifically, SCANPY provides preprocessing comparable to SEURAT [10] and CELL RANGER [6], visualization through TSNE [11, 12], graph-drawing [13–15] and diffusion maps [11, 16, 17], clustering similar to PHENOGRAPH [18–20], identification of marker genes for clusters via differential expression tests and pseudotemporal ordering via diffusion pseudotime [21], which compares favorably [22] with MONOCLE 2 [22], and WISHBONE [23] (Fig. 1a).

SCANPY is benchmarked in comparisons with established packages

In a detailed clustering tutorial of 2700 peripheral blood mononuclear cells (PBMCs), adapted from one of SEURAT's tutorials (http://satijalab.org/seurat/pbmc3k_tutorial.html) [2], all steps starting from raw count data to the identification of cell types are carried out, providing speedups between 5 and 90 times in each step (https://github.com/theislab/scanpy_usage/tree/master/170505_seurat). Benchmarking against the more run-time optimized CELL RANGER R kit [6], we demonstrate a speedup of 5 to 16 times for a data set of 68,579 PBMCs (Fig. 1a,b, https://github.com/theislab/scanpy_usage/tree/master/170503_zheng17) [6]. Moreover, we demonstrate the feasibility of analyzing 1.3 million cells without subsampling in a few hours of computing time on eight cores of a small computing server (Fig. 1c, https://github.com/theislab/scanpy_usage/tree/master/170522_visualizing_one_million_cells). Thus, SCANPY provides tools with speedups that enable an analysis of data sets with more than one million cells and an interactive analysis with run times of the order of seconds for about 100,000 cells.

*Correspondence: alex.wolf@helmholtz-muenchen.de; fabian.theis@helmholtz-muenchen.de
[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

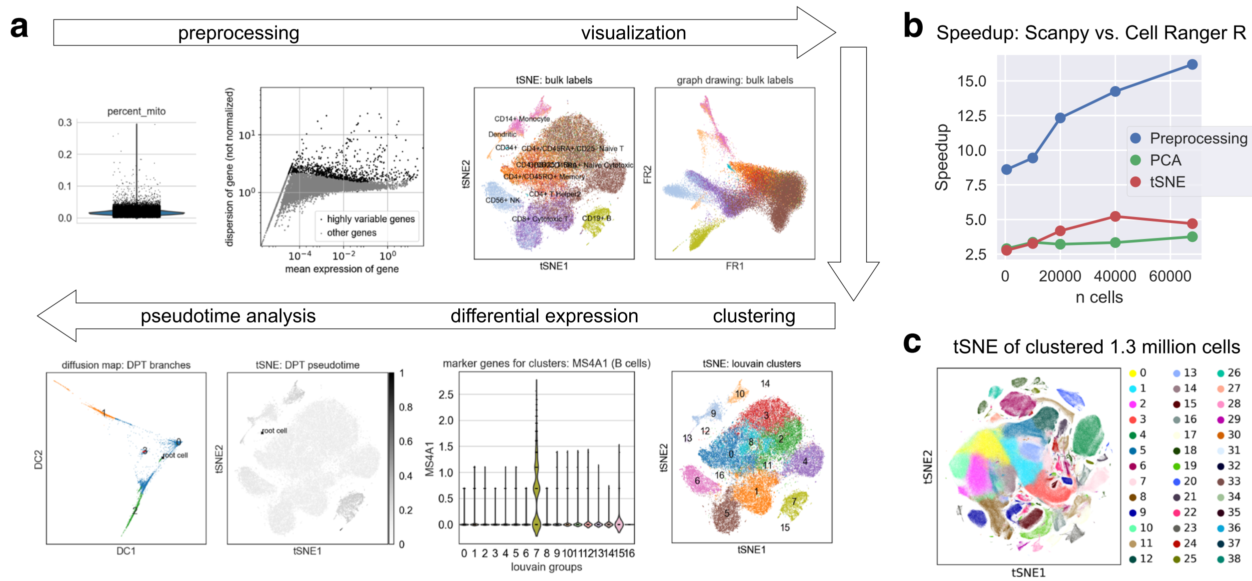

Fig. 1 **a** Scanpy's analysis features. We use the example of 68,579 peripheral blood mononuclear cells of [6]. We regress out confounding variables, normalize, and identify highly variable genes. TSNE and graph-drawing (Fruchterman–Reingold) visualizations show cell-type annotations obtained by comparisons with bulk expression. Cells are clustered using the Louvain algorithm. Ranking differentially expressed genes in clusters identifies the MS4A1 marker gene for B cells in cluster 7, which agrees with the bulk labels. We use pseudotemporal ordering from a root cell in the CD34+ cluster and detect a branching trajectory, visualized with TSNE and diffusion maps. **b** Speedup over Cell Ranger R kit. We consider representative steps of the analysis [6]. **c** Visualizing and clustering 1.3 million cells. The data, brain cells from E18 mice, are publicly available from 10x Genomics. PCA = principal component analysis, DC = diffusion component

In addition to the mentioned standard clustering-based analyses approaches, we demonstrate the reconstruction of branching developmental processes via diffusion pseudotime [21] as in the original paper (https://github.com/theislab/scanpy_usage/tree/master/170502_haghverdi16), the simulation of single cells using literature-curated gene regulatory networks based on the ideas of [24] (https://github.com/theislab/scanpy_usage/tree/master/170430_krumsiek11), and the analysis of deep-learning results for single-cell imaging data [25] (https://github.com/theislab/scanpy_usage/tree/master/170529_images).

Scanpy introduces efficient modular implementation choices

With Scanpy, we introduce the class AnnData—with a corresponding package anndata—which stores a data matrix with the most general annotations possible: annotations of observations (samples, cells) and variables (features, genes), and unstructured annotations. As Scanpy is built around that class, it is easy to add new functionality to the toolkit. All statistics and machine-learning tools extract information from a data matrix, which can be added to an AnnData object while leaving the structure of AnnData unaffected. AnnData is similar to R's ExpressionSet [26], but supports sparse data and allows HDF5-based backing of AnnData objects on disk, a format independent of platform, framework, and language. This allows operating on an AnnData object without fully loading it into memory—the functionality is offered via AnnData's backed mode as opposed to its memory mode. To simplify memory-efficient pipelines, Scanpy's functions operate in-place by default but allow the optional non-destructive transformation of objects. Pipelines written this way can then also be run in backed mode to exploit online-learning formulations of algorithms. Almost all of Scanpy's tools are parallelized.

Scanpy introduces a class for representing a graph of neighborhood relations among data points. The computation of neighborhood relations is much faster than in the popular reference package [27]. This is achieved by aggregating rows (observations) in a data matrix to submatrices and computing distances for each submatrix using fast parallelized matrix multiplication. Moreover, the class provides several functions to compute random-walk-based metrics that are not available in other graph software [14, 28, 29]. Typically, Scanpy's tools reuse a once-computed, single graph representation of data and hence, avoid the use of different, potentially inconsistent, and computationally expensive representations of data.

Conclusions

Scanpy's scalability directly addresses the strongly increasing need for aggregating larger and larger data sets [30] across different experimental setups, for example within challenges such as the Human Cell Atlas [31]. Moreover, being implemented in a highly modular fashion, SCANPY can be easily developed further and maintained by a community. The transfer of the results obtained with different tools used within the community is simple, as SCANPY's data storage formats and objects are language independent and cross-platform. SCANPY integrates well into the existing Python ecosystem, in which no comparable toolkit yet exists.

During the revision of this article, the loom file format (https://github.com/linnarsson-lab/loompy) was proposed for HDF5-based storage of annotated data. Within a joint effort of facilitating data exchange across different labs, ANNDATA now supports importing and exporting to loom (https://github.com/linnarsson-lab/loompy). In this context, we acknowledge the discussions with S. Linnarson, which motivated us to extend ANNDATA's previously static to a dynamic HDF5 backing. Just before submission of this manuscript, a C++ library that provides simple interfacing of HDF5-backed matrices in R was made available as a preprint [32].

Methods

SCANPY's technological foundations

SCANPY's core relies on NUMPY [33], SCIPY [34], MATPLOTLIB [35], PANDAS [36], and H5PY [37]. Parts of the toolkit rely on SCIKIT-LEARN [27], STATSMODELS [38], SEABORN [39], NETWORKX [28], IGRAPH [14], the TSNE package of [40], and the Louvain clustering package of [41]. The ANNDATA class—available within the package ANNDATA—relies only on NUMPY, SCIPY, PANDAS, and H5PY.

SCANPY's Python-based implementation allows easy interfacing to advanced machine-learning packages such as TENSORFLOW [9] for deep learning [42], LIMIX for linear mixed models [43], and GPY/GPFLOW for Gaussian processes [44, 45]. However, we note that the Python ecosystem comes with less possibilities for classical statistical analyses compared to R.

Comparison with existing Python packages for single-cell analysis

Aside from the highly popular SCLVM (https://github.com/PMBio/scLVM) [46, 47], which uses Gaussian process latent variable models for inferring hidden sources of variation, there are, among others, the visualization frameworks FASTPROJECT (https://github.com/YosefLab/FastProject) [48], ACCENSE (http://www.cellaccense.com/) [49], and SPRING (https://github.com/AllonKleinLab/SPRING) [15]—the latter uses the JavaScript package (http://d3js.org D3.js for the actual visualization and Python only for preprocessing—the trajectory inference tool SCIMITAR (https://github.com/dimenwarper/scimitar), the clustering tool PHENOGRAPH (https://github.com/jacoblevine/PhenoGraph) [19], the single-cell experiment design tool MIMOSCA (https://github.com/asncd/MIMOSCA)[50], UMIS (https://github.com/vals/umis) for handling raw read data [51], the tree-inference tool ECLAIR (https://github.com/GGiecold/ECLAIR) [52], and the framework FLOTILLA (https://github.com/yeolab/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 are the core of SCANPY. Also, FLOTILLA is built around a complicated class STUDY, which contains data, tools, and plotting functions. SCANPY, by contrast, is built around a simple HDF5-backed class ANNDATA, which makes SCANPY both scalable and extendable (law of Demeter).

Availability and requirements

SCANPY's and ANNDATA's open-source code are maintained on GITHUB (https://github.com/theislab/scanpy, https://github.com/theislab/anndata) and published under the BSD3 license.

SCANPY and ANNDATA are released via the Python packaging index: https://pypi.python.org/pypi/scanpy and https://pypi.python.org/pypi/anndata.

Demonstrations and benchmarks discussed in the main text are all stored at https://github.com/theislab/scanpy_usage and summarized here:

- Analyzing 68,579 PBMCs (Fig. 1) in a comparison with the CELL RANGER R kit [6]: https://github.com/theislab/scanpy_usage/tree/master/170503_zheng17.
- Clustering and identifying cell types, adapted from and benchmarked with http://satijalab.org/seurat/pbmc3k_tutorial.html and one of SEURAT's tutorials [2]: https://github.com/theislab/scanpy_usage/tree/master/170505_seurat.
- Visualizing and clustering 1.3 million cells (Fig. 1c): https://github.com/theislab/scanpy_usage/tree/master/170522_visualizing_one_million_cells.
- Reconstructing branching processes via diffusion pseudotime [21]: https://github.com/theislab/scanpy_usage/tree/master/170502_haghverdi16.
- Simulating single cells using gene regulatory networks [24]: https://github.com/theislab/scanpy_usage/tree/master/170430_krumsiek11.
- Analyzing deep-learning results for single-cell images [25]: https://github.com/theislab/scanpy_usage/tree/master/170529_images.

The data sets used in demonstrations and benchmarks are three data sets from 10x Genomics.

Programming language: Python

Operating system: Linux, Mac OS and Windows

Acknowledgements

We thank the authors of SEURAT, CELL RANGER, and SPRING for sharing their great tutorials. We are grateful to Sten Linnarson for discussions on HDF5-backing of data on disk. We thank S. Tritschler, L. Simon, D. S. Fischer, and M. Büttner for commenting on the software package. We thank M. Lotfollahi for clustering the 1.3-million-cell data set and N. K. Chlis for setting up installation instructions for Windows.

Funding

FAW acknowledges the support of the Helmholtz Postdoc Programme, Initiative and Networking Fund of the Helmholtz Association.
FJT acknowledges support from the German Research Foundation (DFG) within the Collaborative Research Centre 1243, Subproject A17.

Authors' contributions

FAW conceived the project and developed the software. PA co-developed the software, mainly in regard to architecture and maintainability. FJT supervised the project and helped interpret and present the results. FAW wrote the manuscript with the help of PA and FJT. All authors read and approved the final manuscript.

Ethics approval and consent to participate

Ethics approval was not applicable for this study.

Competing interests

None of the authors declare competing interests.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Received: 16 August 2017 Accepted: 20 December 2017
Published online: 06 February 2018

References

1. Wagner A, Regev A, Yosef N. Revealing the vectors of cellular identity with single-cell genomics. Nat Biotechnol. 2016;34:1145–60.
2. Satija R, Farrell JA, Gennert D, Schier AF, Regev A. Spatial reconstruction of single-cell gene expression data. Nat Biotechnol. 2015;33:495–502.
3. Trapnell C, et al. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. Nat Biotechnol. 2014;32:381–6.
4. Kharchenko PV, Silberstein L, Scadden DT, Bayesian approach to single-cell differential expression analysis. Nat Methods. 2014;11:740–2.
5. Finak, G, et al. MAST: a flexible statistical framework for assessing transcriptional changes and characterizing heterogeneity in single-cell RNA sequencing data. Genome Biol. 2015;16:278.
6. Zheng GXY, et al. Massively parallel digital transcriptional profiling of single cells. Nat Commun. 2017;8:14049.
7. McCarthy D, Wills Q, Campbell K. SCATER: single-cell analysis toolkit for gene expression data in R. Bioinformatics. 2017;33:1179.
8. Lun A, McCarthy D, Marioni J. A step-by-step workflow for low-level analysis of single-cell RNA-seq data with BIOCONDUCTOR. F1000Research. 2016;5:2122.
9. Abadi M, et al. TENSORFLOW: large-scale machine learning on heterogeneous systems. 2015. https://www.tensorflow.org/about/bib.
10. Macosko EZ, et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. Cell. 2015;161:1202–14.
11. Coifman RR, et al. Geometric diffusions as a tool for harmonic analysis and structure definition of data: diffusion maps. Proc Natl Acad Sci. 2005;102: 7426–31.
12. Amir EAD, Davis KL, Tadmor MD, Simonds EF, Levine JH, Bendall SC, et al. VISNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia. Nat Biotechnol. 2013;31: 545–52.
13. Reingold EM. Graph drawing by force-directed placement. Softw Pract Exp. 1991;21:1129–64.
14. Csardi G, Nepusz T. The IGRAPH SOFTWARE PACKAGE FOR COMPLEX NETWORK RESEARCH. InterJournal Compl Syst. 2006;2006:1695.
15. Weinreb C, Wolock S, Klein A. SPRING: a kinetic interface for visualizing high dimensional single-cell expression data. bioRxiv. 2017. https://doi.org/10.1093/bioinformatics/btx792.
16. Buettner F, Theis FJ. Diffusion maps for high-dimensional single-cell analysis of differentiation data. Bioinformatics. 2015;31:2989–98.
17. Angerer P, et al. DESTINY: diffusion maps for large-scale single-cell data in R. Bioinformatics. 2015;32:1241.
18. Blondel VD, Guillaume J-L, Lambiotte R, Lefebvre E. Fast unfolding of communities in large networks. J Stat Mech. 2008;2008:P10008.
19. Levine JH, et al. Data-driven phenotypic dissection of AML reveals progenitor-like cells that correlate with prognosis. Cell. 2015;162:184–97.
20. Xu C, Su Z. Identification of cell types from single-cell transcriptomes using a novel clustering method. Bioinformatics. 2015;31:1974–80.
21. Haghverdi L, Bü, Wolf FA, Buettner F, Theis FJ. Diffusion pseudotime robustly reconstructs branching cellular lineages. Nat Methods. 2016;13: 845–8.
22. Qiu X, et al. Reversed graph embedding resolves complex single-cell trajectories. Nat Methods. 2017;14:979–82.
23. Setty, M, et al. WISHBONE identifies bifurcating developmental trajectories from single-cell data. Nat Biotechnol. 2016;34:637–45.
24. Wittmann, DM, et al. Transforming Boolean models to continuous models: methodology and application to T-cell receptor signaling. BMC Syst Biol. 2009;3:98.
25. Eulenberg P, et al. Reconstructing cell cycle and disease progression using deep learning. Nat Commun. 2017;8:463.
26. Huber, W, et al. Orchestrating high-throughput genomic analysis with BIOCONDUCTOR. Nat Methods. 2015;12:115–21.
27. Pedregosa F, et al. SCIKIT-LEARN: machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
28. Hagberg AA, Schult DA, Swart PJ. Exploring network structure, dynamics, and function using NETWORKX. In: Proceedings of the 7th Python in Science Conference (SciPy2008). Pasadena; 2008. p. 11–15.
29. Bastian M, Heymann S, Jacomy M. GEPHI: an open source software for exploring and manipulating networks. International AAAI Conference on Weblogs and Social Media. 2009.
30. Angerer, P, et al. Single cells make big data: new challenges and opportunities in transcriptomics. Curr Opin Syst Biol. 2017;4:85–91.
31. Regev A, et al. Science forum: the human cell atlas. eLife. 2017;6:e27041.
32. Lun ATL, Pagè H, Smith ML. BEACHMAT: a BIOCONDUCTOR C++ API for accessing single-cell genomics data from a variety of R matrix types. bioRxiv. 2017. https://doi.org/10.1101/167445.
33. van der Walt S, Colbert SC, Varoquaux G. The NUMPY array: a structure for efficient numerical computation. Comput Sci Eng. 2011;13:22–30.
34. Jones E, Oliphant T, Peterson P, et al. SCIPY: open source scientific tools for Python. 2001. https://www.scipy.org/citing.html.
35. Hunter JD. MATPLOTLIB: a 2D graphics environment. Comput Sci Eng. 2007;9:90–5.
36. McKinney W. Data structures for statistical computing in Python. In: van der Walt S, Millman J, editors. Proceedings of the 9th Python in Science Conference; 2010. p. 51–6.
37. Collette A. Python and HDF5. Sebasto pol: O'Reilly; 2013.
38. Seabold S, Perktold J. STATSMODELS: econometric and statistical modeling with Python. 9th Python in Science Conference. 2010.
39. Waskom, M, et al. In: Varoquaux G, Vaught T, Millman J, editors. SEABORN; 2016. http://doi.org/10.5281/zenodo.12710, https://networkx.github.io/documentation/networkx-1.10/reference/citing.html.
40. Ulyanov D. MULTICORE-TSNE. 2016. https://github.com/DmitryUlyanov/Multicore-TSNE.
41. Traag V, Louvain. GITHUB. 2017. https://doi.org/10.5281/zenodo.595481.
42. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521: 436–44.
43. Lippert C, Casale FP, Rakitsch B, Stegle O. In: van der Walt S, Millman J, editors. LIMIX: genetic analysis of multiple traits; 2014. https://doi.org/10.1101/003905, http://conference.scipy.org/proceedings/scipy2010/mckinney.html. bioRxiv.
44. Matthews AGdeG, van der Wilk M, Nickson T, Fujii K, Boukouvalas A, Le'on-Villagr'a P, Ghahramani Z, Hensman J. GPflow: A Gaussian process

library using TensorFlow. J Mach Learn Res. 2017;18(40):1–6. http://jmlr.org/papers/v18/16-537.html.
45. Matthews de, G, Alexander G, et al. GPFLOW: a Gaussian process library using TENSORFLOW. J Mach Learn Res. 2017;18:1–6. https://github.com/SheffieldML/GPy.
46. Buettner F, et al. Computational analysis of cell-to-cell heterogeneity in single-cell RNA-sequencing data reveals hidden subpopulations of cells. Nat Biotechnol. 2015;33:155.
47. Buettner F, Pratanwanich N, McCarthy DJ, Marioni JC, Stegle O. F-SCLVM: scalable and versatile factor analysis for single-cell RNA-seq. Genome Biol. 2017;18:212.
48. DeTomaso D, Yosef N. FASTPROJECT: a tool for low-dimensional analysis of single-cell RNA-seq data. BMC Bioinform. 2016;17:315.
49. Shekhar K, Brodin P, Davis MM, Chakraborty AK. Automatic classification of cellular expression by nonlinear stochastic embedding (ACCENSE); 2013. p 202–7.
50. Dixit A, et al. PERTURB-SEQ: dissecting molecular circuits with scalable single-cell RNA profiling of pooled genetic screens. Cell. 2016;167:1853–66.e17.
51. Svensson V, et al. Power analysis of single cell RNA-sequencing experiments. Nat Methods. 2017;14:381.
52. Giecold G, Marco E, Garcia SP, Trippa L, Yuan G-C. Robust lineage reconstruction from high-dimensional single-cell data. Nucleic Acids Res. 2016;44:e122.