Zappia and Theis *Genome Biology* (2021) 22:301
https://doi.org/10.1186/s13059-021-02519-4

Genome Biology

REVIEW Open Access

Over 1000 tools reveal trends in the single-cell RNA-seq analysis landscape

Luke Zappia[1,2] and Fabian J. Theis[1,2,3*]

* Correspondence: fabian.theis@helmholtz-muenchen.de
[1]Institute of Computational Biology, Helmholtz Zentrum München, 85764 Neuherberg, Germany
[2]Department of Mathematics, Technical University of Munich, 85748 Garching bei München, Germany
Full list of author information is available at the end of the article

Abstract

Recent years have seen a revolution in single-cell RNA-sequencing (scRNA-seq) technologies, datasets, and analysis methods. Since 2016, the scRNA-tools database has cataloged software tools for analyzing scRNA-seq data. With the number of tools in the database passing 1000, we provide an update on the state of the project and the field. This data shows the evolution of the field and a change of focus from ordering cells on continuous trajectories to integrating multiple samples and making use of reference datasets. We also find that open science practices reward developers with increased recognition and help accelerate the field.

Introduction

Developments in single-cell technologies over the last decade have drastically changed the way we study biology. From measuring genome-wide gene expression in a few cells in 2009 [1], researchers are now able to investigate multiple modalities in thousands to millions of cells across tissues, individuals, species, time, and conditions [2, 3]. The commercialisation of these techniques has improved their robustness and made them available to a greater number of biological researchers. Although single-cell technologies have now extended to other modalities including chromatin accessibility [4, 5], DNA methylation [6, 7], protein abundance [8], and spatial location [9, 10], much of the focus of the single-cell revolution has been on single-cell RNA sequencing (scRNA-seq). Single-cell gene expression measurements are cell type-specific (unlike DNA), more easily interpretable (compared to epigenetic modalities), and scalable to thousands of features (unlike antibody-based protein measurements) and thousands of cells. These features mean that scRNA-seq can be used as an anchor, often measured in parallel and used to link other modalities.

While single-cell assays of all kinds are now more readily available, the ability to extract meaning from them ultimately depends on the quality of computational and statistical analysis. With the rise of new technologies, we have seen a corresponding boom in the development of analytic methods. After years of rapid growth, the sheer number of possible analysis options now available can be bewildering to researchers faced with an scRNA-seq dataset for the first time. Efforts have also been made to benchmark

common tasks such as clustering of similar cells [11, 12], differential expression between cells [13, 14] or integration of multiple samples [15, 16] in an attempt to establish which approaches are consistently good performers and in which situations they fail. Building on these benchmarks, the community has now produced tutorials [17], workshops, and best practices recommendations for approaching a standard analysis [18, 19].

Several projects exist which attempt to chart the progression of scRNA-seq technologies, datasets, and analysis tools. For example, the single-cell studies database tracks the availability and size of scRNA-seq datasets and has been used to show trends in technologies and analysis [20]. The Awesome Single Cell repository is a community-curated list of software packages, resources, researchers, and publications for various single-cell technologies [21] and Albert Villela's SingleCell Omics spreadsheet [22] tracks a range of information including technologies, companies, and software tools. While these are all very useful resources, they either have different focuses or are less structured and detailed than the scRNA-tools database.

The scRNA-tools database focuses specifically on the cataloging and manual curation of software tools for analyzing scRNA-seq data [23]. When tools become available (usually through a bioRxiv preprint), we classify them according to the analysis tasks they can be used for and record information such as associated preprints and publications, software licenses, code location, software repositories, and a short description. Most tools are added to the database within 30 days of the first preprint or publication (Additional file 1: Figure S1). All the recorded information is publicly available in an interactive format at https://www.scrna-tools.org/ [24]. As the number of tools in the database has moved past 1000, we have taken this opportunity to provide an update on the current state of the database and explore trends in scRNA-seq analysis across the past 5 years. We find that the focus of tool developers has moved on from continuous ordering of cells to methods for integrating samples and classifying cells. The database also shows us that more new tools are built using Python while the relative usage of R is declining. We also examine the role of open science in the development of the field and find that open source practices lead to increased citations. While the scRNA-tools database does not record every scRNA-seq analysis tool, the large proportion it does include over the history of what is still a young field make these analyses possible and a reasonable estimate of trends across all tools.

Results

The current state of scRNA-seq analysis tools

We first started cataloging software tools for analyzing scRNA-seq data in 2016 and the scRNA-tools database currently contains 1059 tools as of September 26, 2021 (Fig. 1A). This represents a more than tripling of the number of available tools since the database was first published in June 2018. The continued growth of the number of available tools reflects the growth in the availability of and interest in single-cell technologies. It also demonstrates the continued need for new methods to extract meaning from them. This trend has continued for more than 5 years, and if it continues at the current rate, we can expect to see around 1500 tools by the end of 2022 and more than 3000 by the end of 2025 (Additional file 1: Figure S2).

Fig. 1 Overview of the scRNA-tools database. **A** Line plot of the number of tools in the scRNA-tools database over time. The development of tools for analyzing scRNA-seq data has continued to accelerate with more than 1000 tools currently recorded. **B** Publication status of tools in the scRNA-tools database. Around 70% of tools have at least one peer-reviewed publication while more than 20% have an associated preprint. **C** Bar charts showing the distribution of platforms, software licenses, and software repositories for tools in the scRNA-tools database. Colors indicate proportions of tools using R or Python. **D** Bar chart showing the proportion of tools in the database assigned to each analysis category. Categories are grouped by broad phases of a standard scRNA-seq analysis workflow

Publication status

The scRNA-tools database records preprints and publications associated with each tool, currently including more than 1500 references. Over two thirds of tools have at least one peer-reviewed publication while around another quarter have been described in a preprint but are not yet peer-reviewed (Fig. 1B). The remaining less than 10% currently have no associated references and have come to our attention through other means such as Twitter, software and code repositories or submissions via the scRNA-tools website. The overall number of tools with peer-reviewed publications has increased over time, which is to be expected as tools make their way through the publication process. For more discussion of the delay in publication and the effect of preprints, see the open science section.

Software platforms, licenses, and repositories

Tool developers must make a choice of which platform to use, and this is also recorded in the scRNA-tools database. In most cases, the platform is a programming language but a minority of tools is built around an existing framework including workflow managers such as Snakemake [25, 26] and Nextflow [27]. R [28] and Python [29] continue to be the dominant platforms for scRNA-seq analysis, just as they are for more general data science applications (Fig. 1C). C++ continues to play a role, particularly for including compiled code in R packages to improve computational efficiency. Although it is a

minority (less than 7%), there is a consistent community of developers focussed on MATLAB. The popularity of interpreted languages (R, Python, MATLAB) with commonly used interactive interfaces rather than compiled languages (C++) is consistent with relatively few tools being developed for low-level, computationally intensive tasks such as alignment and quantification where compiled languages are more commonly seen (Additional file 1: Figure S3)

Around two thirds of tools are not available from a standard centralized software repository (CRAN, Bioconductor or PyPI) and are mostly available only from GitHub. While this makes it harder for other members of the community to install and use these tools, it is perhaps unsurprising given the large amount of time and effort required to maintain a software package. Many of these tools may also be primarily intended as example implementations of a method rather than a tool designed for reuse by the community. A higher proportion of Python-based tools are available from PyPI (the primary Python package repository) when compared to those built using R. This may reflect the lower submission requirements of PyPI which do not enforce checks for documentation or testing, unlike R repositories. Of R packages available from central repositories, the majority of developers have chosen to submit their tools to the biology-focussed Bioconductor [30] repository rather than the more general CRAN. The Bioconductor community is well-established and provides centralized infrastructure (such as the commonly used SingleCellExperiment class) designed to allow small, specialized packages to work together.

Most tools are covered by a standard open-source software license, although there remains a consistent minority of tools (around 20%) for which no clear software license is available. The lack of a license can severely restrict how tools can be used and the ability of the community to learn from and extend existing code. We strongly encourage authors to clearly license their code and for reviewers and journal editors to include checks for a software license into the peer-review process to avoid this problem. Among those tools that do have a license, variants of the copy-left GNU Public License (GPL) are most common (particularly among R-based tools). The MIT license is also used by many R tools but is more common for Python tools, as are BSD-like and Apache licenses. These licenses all allow the reuse of code but may impose some conditions such as retaining the original copyright notice [31]. GPL licenses also require that any derivatives of the original work are also covered by a GPL license.

Analysis categories

A unique feature of the scRNA-tools database is the classification of tools according to which analysis tasks they can be used for. These categories have been designed to capture steps in a standard scRNA-seq workflow, and as new tasks emerge, additional categories can be created. While they have limitations, these categories should provide some guidance to analysts looking to complete a particular task and can be used to filter tools on the scRNA-tools website. Categories that are applicable to many stages of analysis (visualization, dimensionality reduction) are among the most common, as is clustering which has been the focus of much tool development but is also required as an input to many other tasks (Fig. 1D). Other stages of a standard analysis form the next most common categories including integration of multiple samples, batches or

modalities, ordering of cells into a lineage or pseudotime trajectory, quality control of cells, normalization, classification of cell types or states, and differential expression testing. Tools that either construct or make use of gene networks are also common. Some tools such as Seurat [32] and Scanpy [33] are general analysis toolboxes that can complete many tasks, while others are more specialized and focus on one problem. While the number of categories per tool is variable there is no clear trend in tools becoming more general or specialized over time (Additional file 1: Figure S4).

Other categories in the scRNA-tools database capture some of the long tail of possible scRNA-seq analyses. For example, analysis of alternative splicing or allele-specific expression, stem cells, rare cell types, and immune receptors may not be relevant for all experiments, but when they are having methods for those specific tasks can be invaluable. As biologists use scRNA-seq to investigate more phenomena, developers will create methods and tools for more specific tasks. We plan to update the categories in the scRNA-tools database to reflect this and have recently added a category for tools designed to work with perturbed data such as drug screens or gene editing experiments including MELD [34], scTenifoldKnK [35], and scGen [36].

Trends in scRNA-seq analysis tools

Over 5 years of data in the scRNA-tools database on new tools and their associated publications allows us to track some of the trends in scRNA-seq analysis over that time. Here, we focus on trends in analysis tasks as well as the choice of development platform.

An increasing proportion of tools use Python

Figure 2A shows how the proportion of tools using the most common programming languages has changed over time as more tools are added to the database. The clear trend here is the increasing popularity of Python and the corresponding decrease in the proportion of tools built using R. There are several possible explanations for this trend.

As the size and complexity of scRNA-seq datasets have increased, the potential memory and computational efficiency of Python has become more relevant. Another possible catalyst is the development of Python-based infrastructure for the community to build around such as the AnnData and Scanpy packages [33] which play a similar central role in the Python environment as the SingleCellExperiment and Seurat packages do in the R environment (Additional file 1: Figure S5). These standard representations improve interoperability between packages and allow developers to focus on analysis methods rather than how to store their data which may have previously been a barrier for Python developers.

While bulk transcriptomics typically focused on the statistical analysis of a designed experiment, scRNA-seq analysis is often more exploratory and employs more machine learning techniques such as unsupervised clustering and more recently various neural-network architectures. This shift in analysis focus may have triggered a corresponding shift in demographics with more researchers from a computer science background turning their attention to developing scRNA-seq analysis methods and bringing with them a preference for Python over R. If this trend continues at the current rate, we can expect Python to overtake R as the most common platform for scRNA-seq analysis

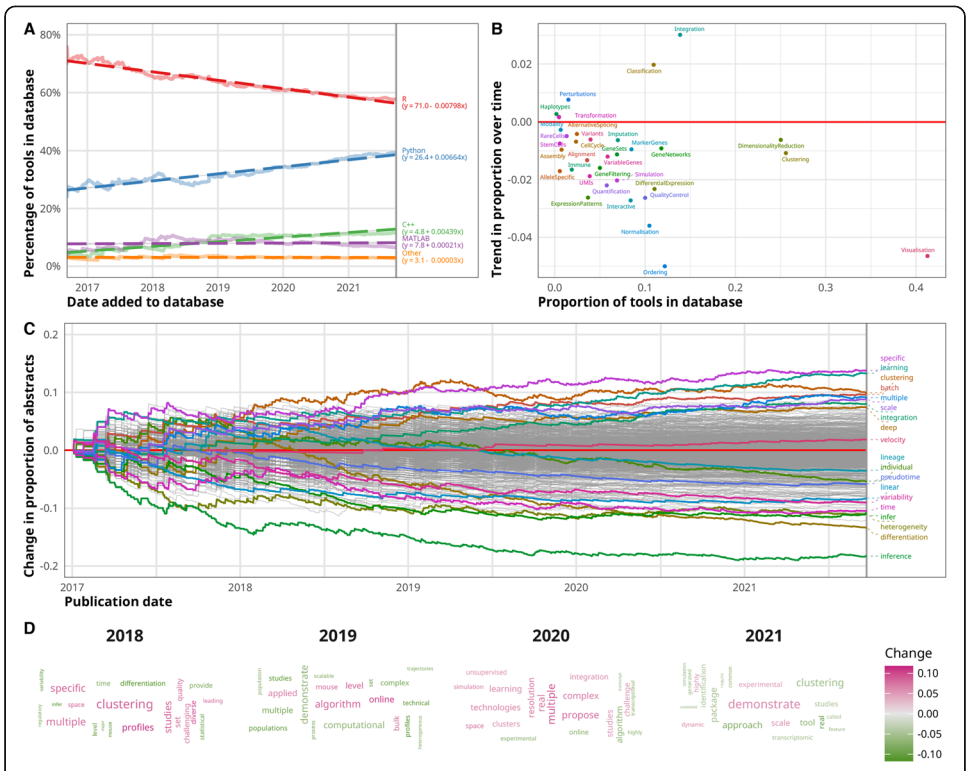

Fig. 2 Trends in scRNA-seq analysis tools. **A** Line plot of platform usage of tools in the scRNA-tools database over time. Python usage has increased over time while R usage has decreased. Darker dashed lines show linear fits with coefficients given in the legend. **B** Scatter plot of trends in scRNA-tools analysis categories over time. The current proportion of tools in the database is shown on the *x*-axis, and the trend in proportion change is shown on the *y*-axis. **C** Line plot of trend in word use in scRNA-seq analysis tool publication abstracts over time. Publication date is shown on the *x*-axis and change in the proportion of abstracts containing a word on the *y*-axis. Some important and highly variable terms are highlighted. **D** Word clouds of abstract terms by year. Word size indicates the proportion of abstracts that included the term in that year. The color of words shows the change in proportion compared to the previous year with pink indicating an increase and green indicating decreases. The 20 words with the greatest change in proportion are shown for each year

tools by mid-2025; however, R will continue to be an important platform for the community. It is also important to note that this trend represents the preferences of developers and may not reflect how commonly these platforms are used by analysts.

Greater focus on integration and classification

Trends also exist in the tasks that new tools perform. In Fig. 2B, we can see the overall proportion of tools in the database assigned to each category against the trend in proportion over time. Two categories stand out as increasing in focus over time: integration and classification. Both of these trends reflect the growing scale, complexity and availability of scRNA-seq datasets. While early scRNA-seq experiments usually consisted of a single sample or a few samples from a single lab, it is now common to see experiments with multiple replicates, conditions and sources. For example, studies have benchmarked single-cell protocol across multiple centers [37], measured hundreds of cell lines at multiple time points [38], and compared immune cells between cancer types [39]. Handling batch effects between samples is vital to producing meaningful

results but can be extremely challenging due to the balance of removing technical effects while preserving biological variation. The importance of this task is demonstrated by the more than 140 tools that perform some kind of integration and is particularly relevant for global atlas building projects like the Human Cell Atlas [40] which attempt to bring together researchers and samples from around the world to map cell types in whole tissues or organisms. Recent technological advances have made it more feasible to measure biological signals other than gene expression in individual cells. Some tools tackle this more challenging task, either using one modality to inform analysis of another or by bringing modalities together for a fully integrated analysis. Combining multiple data types can provide additional insight by confirming a signal that is unclear in one modality (for example protein expression confirming gene expression) or revealing another aspect of a biological process (chromatin accessibility used to show how genes are regulated).

The increased interest in classification can also be seen as an attempt to tackle the increasing scale of scRNA-seq data. Rather than performing the computationally and labor-intensive task of merging datasets and jointly analyzing them to get consistent labels, classifier tools make use of public references to directly label cells with cell types or states. This approach is a shortcut for analysts which allows them to skip many early analysis steps but is limited by the completeness and reliability of the reference. For this reason, integration and classification are intimately linked, with effective integration required to produce high-quality references for classification. Some tools address both sides of this problem, integrating datasets to produce a reference and providing methods to classify new query datasets while considering batch effects in the query.

Decrease in ordering and common tasks

The category with the biggest decreasing trend over time is the ordering category, which refers to tools that determine a continuous order for cells, usually related to a developmental process or another perturbation such as the onset of disease or the effect of a drug treatment. This category represents perhaps the biggest promise of the single-cell revolution, the ability to interrogate continuous biological processes at the level of individual cells. Some of the earliest scRNA-seq analysis tools addressed this task but the proportion of new tools containing ordering methods has significantly decreased since. It is unclear why this is the case. It may simply be that high-performing methods have been established [41] and adopted by the community, reducing the need for further development. Alternatively, it could be that the initial excitement was overcome by limitations revealed when these techniques were applied to real datasets [42]. A subset of the ordering category is RNA velocity methods [43, 44] which offer an alternative approach to analyzing continuous processes but have resulted in the development of relatively few new tools.

Many of the other categories show some trend toward decreasing proportions with normalization and visualization having the biggest reductions. A plausible explanation for these changes is the consolidation of common tasks into analysis toolboxes with each of the major software repositories having standard workflows based around a few core tools. With these available and accessible through the use of standard data

structures, developers no longer need to implement each stage of an analysis workflow and can focus on particular tasks.

Trends in publication abstracts

Similar trends can be seen in the text of publications associated with analysis tools. Figure 2C shows how the proportion of abstracts associated with scRNA-seq tools containing keywords has changed since 2017. Highly variable words are highlighted as well as some related to trends discussed above. Both "batch" and "integration" have become more common, mirroring the increase in tools performing integration, as have related terms like "scale" and "multiple." Machine learning terms "deep" and "learning" have also become more common, consistent with the increased use of Python which is the primary language for deep learning. In contrast "lineage", "pseudotime," and "differentiation" have all decreased consistently with the reduction of tools in the ordering category. The "velocity" term has seen a small increase in use over the last 2 years, but as these abstracts only come from publications associated with tools (and not publications that focus on the analysis of scRNA-seq data), it is difficult to say anything about the take-up of these methods in the community and whether they have replaced the earlier generation of ordering techniques.

Figure 2D shows this the same data as a series of word clouds by year. The twenty words with the biggest change in proportion compared to the previous year are shown with color indicating the change and size indicating the proportion in that year. These terms are less specific than those found by looking at the whole timespan of the database but still show changes in important terms like clustering and deep learning.

Open science accelerates scRNA-seq tool development

Researchers must make a conscious choice about when and how to share their work and for developers of software tools—there are several options. A tool could be made available during development, when there is a stable version ready for users, accompanying a preprint or only after a peer-reviewed publication. Here, we touch on the decision around open science practices and the effect they have on scRNA-seq analysis tools.

GitHub is the primary home for scRNA-seq analysis tools

The vast majority of tools in the scRNA-tools database (over 90%) have a presence on the social coding website GitHub. Like GitLab, BitBucket, and other similar services, GitHub provides an all-in-one service for open-source software development which has been embraced by the scRNA-seq community. Being available on GitHub allows the community to ask questions, raise issues, suggest enhancements, and contribute features. Across the scRNA-tools database, there are 991 associated GitHub repositories from 734 owners (Fig. 3A). To these repositories, over 1800 contributors have made more than 165,000 commits and opened over 29,000 issues. If each of these commits and issues represents just 10 min of work on average, this corresponds to more than 32,000 person-hours of work or three and a half person-years. This is a tremendous amount of effort from the community but is likely still a large underestimate as this does not capture many of the tasks involved in software development and maintenance.

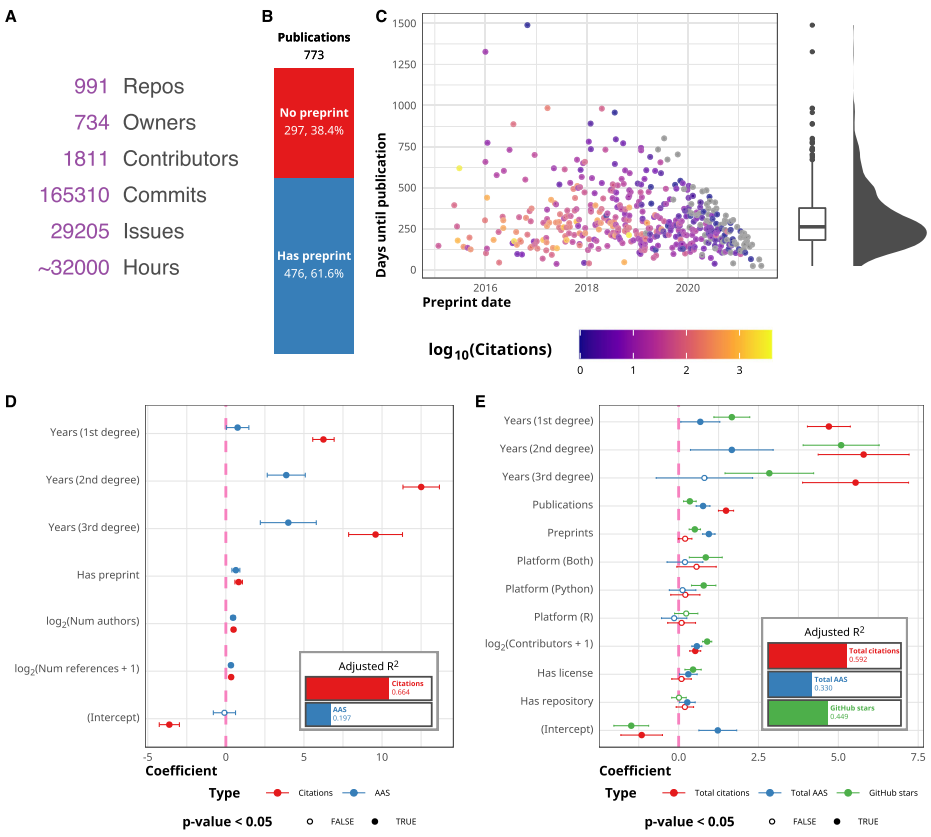

Fig. 3 Open science in scRNA-seq tools development. **A** GitHub summary statistics for scRNA-seq tool repositories. **B** Stacked bar plot showing the proportion of publications with and without an associated preprint. **C** Scatter plot showing preprint date against the number of days until publication, colors indicate the number of citations (log scale). Box plot and density on the right show the distribution of time delay in publication. **D** Coefficients for log-linear models predicting citations and Altmetric Attention Score (AAS) for publications. Years since publication are modeled as a cubic spline with three degrees of freedom. Error bars show a 95% confidence interval. The inlaid bar chart shows the adjusted R^2 for each model. **E** Coefficients for log-linear models predicting total citations, total AAS, and GitHub stars for tools. Error bars show a 95% confidence interval. The inlaid bar chart shows the adjusted R^2 for each model

Preprints allow the rapid development of scRNA-seq methods

Around 60% of the 773 publications associated with tools in the scRNA-tools database were preceded by a preprint (Fig. 3B). Figure 3C shows the number of days between preprints and the publication of the final peer-reviewed article. The average delay in publication is around 250 days, but the biggest gap is around six times that long at almost 1500 days. The willingness of the scRNA-seq community to share their work in preprints and make code implementing it on GitHub is a big contributor to the rapid development of the field. Without early sharing of ideas, we would still be waiting for tools that were released a year ago and that delay would likely be much longer if we consider the compounding effect of early access over time.

Open science practices lead to increased citations

To quantify the effect of open science practices such as posting preprints and sharing code, we modeled citations using a method based on that suggested by Fu and Hughey

[45]. We acknowledge that citations are a flawed and limited metric for assessing software tools, for example, they only indirectly measure the effectiveness and usability of a tool. Despite these flaws, citations are still the metric most commonly used to judge researchers and their work so it is important to see how they are affected by open science practices. The same model is also used to predict Altmetric Attention Score (AAS) [46] and GitHub stars (for tools).

Figure 3D shows coefficients for a log-linear model predicting metrics for publications. Unsurprisingly, the biggest predictor of the number of citations is years since publication (included in the model as a spline with three degrees of freedom) (Additional file 1: Table S1). Whether or not a publication has an associated preprint was also a significant positive predictor (coefficient = 0.82, 95% CI = (0.58, 1.1), p value = 4.2e−11). As we modeled metrics on a $\log_2$ scale, a coefficient close to one indicates a two-fold increase in citations for publications with a preprint. The number of authors and number of references were also significant in this model but with smaller effect sizes. This same effect was observed by Fu and Hughey more generally across fields and as well as in other similar studies [47]. That preprints both help the community and result in more citations should encourage more researchers to share their work in this way and outweigh the fear of being "scooped."

Results for AAS are broadly similar to those seen for citations, but it is important to note that the model fit for AAS is significantly worse than for citations. While the two metrics are correlated ($\rho = 0.55$) (Additional file 1: Figure S6), AAS captures a much wider range of sources, and many of which could be more highly associated with preprints than publications.

Coefficients for a similar model for predicting metrics at the tool level are shown in Fig. 3E. In this model, we replaced having a preprint with whether or not the tool is available from one of the major software repositories (CRAN, Bioconductor, PyPI). Similarly, the number of GitHub contributors has replaced the number of authors. We also included the tool platform, the number of publications, the number of preprints, and the presence of a software license as possible predictors. As well as total citations and total AAS for tools, we also model the number of GitHub stars which has been used by other studies of GitHub repositories and found to be closely related to more complex measures of GitHub popularity [48]. Similar to publication metrics, these values are also weakly correlated but are not entirely predictive of each other (Additional file 1: Figure S6). After accounting for the age of GitHub repositories, the number of publications had the largest effect on total citations (Additional file 1: Table S2). Interestingly, the number of preprints was a bigger predictor of total AAS suggesting that preprints attract attention and are responsible for the initial interest in a tool. The usage of Python was a significant predictor for the number of GitHub stars, and the presence of a software license predicted GitHub stars and AAS while the availability of tools from a software repository was only significant for AAS.

The other significant predictor for all three metrics was the number of GitHub contributors. While this is similar to the number of authors of a publication, an important distinction is that members of the community can contribute code over time (unlike publications where the number of authors is fixed). This makes it difficult to establish the direction of the relationship to the metrics modeled here, i.e., do more contributors lead to better tools that get more citations and stars or do tools that are already highly

used attract more contributors? While including GitHub contributors result in models with better fits, it is possible that it also masks the smaller effects of other predictors such as the availability of tools from a software repository.

Conclusions

Interest in single-cell RNA-sequencing and other technologies continues to increase with more datasets being produced with more complex designs, greater numbers of cells, and multiple modalities. To keep pace with the increase in datasets, there has been a corresponding increase in the development of computational methods and software tools to help make sense of them. We have cataloged this development in the scRNA-tools database which now records more than 1000 tools. While we try to record every new scRNA-seq analysis tool, the database is likely incomplete. We warmly welcome contributions to the database and encourage the community to submit new tools or updates via the scRNA-tools website. This includes suggestions for new categories to capture aspects of analysis that are not currently represented.

Despite the incompleteness of the database, it is a large sample of the scRNA-seq analysis landscape, and using it, we can observe trends in the field. We see that there has been a decrease in the development of methods for ordering cells into continuous trajectories and an increased focus on methods for integrating multiple datasets and using public reference datasets to directly classify cells. Once more comprehensive references atlases are available; these reference-based workflows will likely replace the current unsupervised clustering approach for many analyses. The classification of tools by analysis task is an important feature of the scRNA-tools database, and we plan to expand these categories to cover more aspects of scRNA-seq analysis. We also examined trends in development platforms and found that more new tools are being built using Python than R. While it is exciting to see the field develop, the continued increase in the number of tools presents some concerns. There is still a need for new tools, but if growth continues at this rate, it is a risk that the community will start repeating work and approaches that are already available. By providing a catalog of tools in a publicly available website, the scRNA-tools database makes it easier to find current tools and we encourage developers to contribute to existing projects where that is a good fit. Equally important are continued, high-quality benchmarking studies to rigorously evaluate the performance of methods and we hope to include this information in future versions of the database. This would include published benchmarks but also results from continuous benchmarking efforts such as those proposed by the Open Problems in Single-Cell Analysis project [49].

The scRNA-seq community has largely embraced open science practices, and we sought to quantify their effect on the field. We found an average delay of 250 days between preprints and peer-reviewed publications with some examples being much longer. The willingness of researchers to share early versions of their work has likely been a major contributor to the rapid development of the field. We also found that open science did not conflict with recognition of work with open science practices being a positive predictor of citations and AAS.

We hope that the scRNA-tools database is a valuable resource for the community, both for helping analysts find tools for a particular task and tracking the development of the field over time.

Methods

Curation of the scRNA-tools database

The main sources of tools and updates for the scRNA-tools database are Google Scholar and bioRxiv alerts for scRNA-seq specific terms (Table 1). Other sources include social media, additions to similar projects like the Awesome Single Cell page [21] and submissions via the scRNA-tools website. Once a potential new tool is found, it is checked if it fulfills the criteria for inclusion in the database, namely that it can be used to analyze scRNA-seq data and that the tool is available to users to install and run locally (this excludes tools and resources that are only available online). Most of the information for new tools (description, license, categories) comes from code repositories (GitHub) and package documentation.

We believe that the categorization of tools according to the tasks they perform is an important feature of the scRNA-tools database. The current categories have been designed to cover the main stages of a standard scRNA-seq analysis workflow but may not fully represent alternative or new analysis approaches. When the need for a new category becomes apparent, it can be added to the database as we have done for perturbations. Other categories under consideration include cell-cell interactions, infrastructure, time series, and cancer. The major bottleneck to adding new categories is the time required to re-categorize tools already in the database, and we welcome support from the community for this task.

A command-line application written in R (v4.0.5) [28] and included in the main scRNA-tools repository is used to make changes to the database. Using this interface rather than editing files allows input to be checked and some information to be automatically retrieved from Crossref (using the *rcrossref* package (v1.1.0) [50]), arXiv (using the *aRxiv* package (v0.5.19) [51]) and GitHub (using the *gh* package (v1.2.0) [52]). This application also contains functionality for performing various consistency checks including identifying new or deprecated software packages and code repositories, updated licenses and new publications linked to preprints. Identification of new software package repositories is done by fuzzy matching of the tool name using the *stringdist* package (v0.9.6.3) [53]. Linking of preprints uses an R implementation of the algorithm by Cabanac, Oikonomidi and Boutron [54] available in the *doilinker* package (v0.1.0) [55]. Software licenses are standardized using the SPDX License List [56]. Data files and visualizations shown on the scRNA-tools website are also produced by the command line application using *ggplot2* (v3.3.3) [57] and *plotly* (v4.9.2.2) [58]. Dependencies for the application are managed using the *renv* package (v0.13.0) [59] and more details on its use and functionality can be found at https://github.com/scRNA-tools/scRNA-tools/wiki.

Table 1 Alert terms for Google Scholar and bioRxiv

Google Scholar	bioRxiv
"scRNAseq" OR "scRNA-seq" OR "sc-RNA-seq" OR "(sc)RNA-seq"	scRNA-seq scRNAseq
"single-cell gene expression"	single-cell RNA-sequencing
"single-cell transcriptomics" OR "single-cell transcriptome"	
"single-cell RNA sequencing"	
"single-cell rna-seq"	

Contributing to scRNA tools

Contributions to the scRNA-tools database from the community are welcomed and encouraged. The easiest way to contribute is by submitting a new tool or update using the form on the scRNA-tools website (https://www.scrna-tools.org/submit). For those comfortable using Git and GitHub changes can be made directly to the database and submitted as a pull request. Suggestions for changes or enhancements to the website can be made by opening issues on the scRNA-tools GitHub repository (https://github.com/scRNA-tools/scRNA-tools) or using the contact form on the website (https://www.scrna-tools.org/contact).

Data acquisition and analysis

The main source of data for the analysis presented here is the scRNA-tools database as of September 26, 2021. This was read into R directly from the GitHub repository using the *readr* package (v1.4.0) [60] and manipulated using other *tidyverse* (v1.3.1) packages [61], particularly *dplyr* (v1.0.6) [62], *tidyr* (v1.1.3) [63], *forcats* (v0.5.1) [64], and *purrr* (v0.3.4) [65]. Additional information about references was obtained from the Crossref and Altmetric.com APIs using the *rcrossref* (v1.1.0.99) [50] and *rAltmetric* (v0.7.0) [66] packages. Dates for publications can vary depending on what information journals have submitted to Crossref. We used the online publication date where available, followed by the print publication date and the issued date. When a date was incomplete, it was expanded to an exact date by setting missing days to the first of the month and missing months to January, so that an incomplete date of 2021-08 would become 2021-08-01 and 2021 would become 2021-01-01. Dependencies for CRAN packages were obtained using the base R *package_dependencies()* function and for Bioconductor packages using the *BiocPkgTools* package (v1.10.1) [67]. Python package dependencies were found using the Wheelodex API [68] and the *johnnydep* package (v1.8) [69]. All of the data acquisition and analysis was organized using a *targets* (v0.4.2) [70] pipeline with dependencies managed using *renv* (v0.13.2) [59].

Modeling trends

The trend in platform usage over time was simply modeled by calculating the proportion of tools using each of the main platforms on each day since the creation of the database. This proportion was then plotted over time to display the trend.

A more complex approach was taken to modeling the trend in categories. Time since the start of the database was divided into calendar quarters (3-month periods), and for each category, the proportion of tools added during each quarter was calculated. These quarter proportions were used as the input to a linear model (base R *lm()* function), and the calculated slope taken as the trend for each category. These trends were then plotted against the overall proportion in the database to show the relationship between this and the trend over time.

Modeling of publication term usage over time started with abstracts obtained from Crossref. Abstracts were available for 1055 references. Each abstract was converted to a bag of words using the *tidytext* package (v0.3.1) [71] and URLs, and numbers and common stop words were removed. We also excluded a shortlist of uninformative common scRNA-seq terms and those that appeared in less than 10 abstracts. For each word, we

calculated the cumulative proportion of abstracts that continued that word for each day since the first publication. The proportion as of the start of 2017 was taken as a baseline, and the change since then was plotted. A set of top words to display was chosen based on their variability over time, high absolute change in proportion and a few relevant to other parts of the analysis.

Word clouds were created by calculating the proportion of abstracts published in each year that contained each word. The change in proportion compared to the previous year was then calculated, and the 20 words with the greatest change were selected for plotting using the *ggwordcloud* package (v0.5.0) [72].

Modeling the effect of open science

To model the effect of a previous preprint on citations and Altmetric Attention Score for a publication, we used a simplified version of the log-linear model proposed by Fu and Hughey [45]:

$$\log_2(\text{Metric}+1) \sim \log_2(\text{Num.references}+1) + \log_2(\text{Num.authors}) + \text{Preprint} + \text{spline}(\text{Years}, \text{df}=3)$$

Here, *Metric* is either citations or AAS, and *Preprint* is a Boolean indicator variable showing whether or not the publication has an associated preprint and $spline(Years, df = 3)$ is a natural cubic spline fit to years since publication with three degrees of freedom. We excluded additional author terms from the original model including whether an author had a US affiliation, whether an author had a Nature Index affiliation and the publication age of the last author. Information for these terms is difficult to collect, and in the original publication, they were shown to have a small effect compared to the presence of a preprint. We also fit all publications together rather than for each journal individually.

We then adapted this model to consider tools rather than individual publications:

$$\log_2(\text{Metric}+1) \sim \text{Platform} + \text{Repository} + \text{License} + \text{Publications} + \text{Preprints} + \log_2(\text{Contributors}+1) + \text{spline}(\text{Years}, \text{df}=3)$$

Here, *Metric* is either total citations for all publications and preprints associated with a tool, total AAS for all publications and preprints or the number of GitHub stars. The *Platform* variable is an indicator showing whether the tool uses R, Python, both or some other platform (baseline). *Repository* is a Boolean indicator showing whether the tool is available from any of the major software package repositories (CRAN, Bioconductor or PyPI). *License* is a Boolean indicator variable showing whether the tool has an associated software license. *Publications* and *Preprints* are the numbers of publications and preprints associated with a tool, and *Contributors* is the number of GitHub contributors.

These variables were selected after assessing the fit of a range of models predicting total citations. Although other values obtained from GitHub such as the number of issues or forks are also good predictors, we choose to exclude them because while they are correlated we do not consider them to have a causal relationship. It is unlikely that

a tool is cited because it has many GitHub issues, rather that whatever causes a tool to be used will result in both more citations and more GitHub issues.

Models were fit using the base R *lm()* function and summary statistics including confidence intervals, and *p* values were extracted using the *ggstatsplot* package (v0.8.0) [73]. Spearman's correlation coefficient was calculated for each pair of metrics using the base R *cor()* function.

Visualization

Plots and other figures were produced in R using the *ggplot2* package (v3.3.5) [57]. Various extension packages were also used including *ggtext* (v0.1.1) [74] for complex formatting of text and *ggrepel* (v0.9.1) [75] for labeling points and lines. Labels for bar charts and other plots were constructed using the *glue* package (v1.4.2) [76]. The final figures were assembled using the *cowplot* (v1.1.1) [77] and *patchwork* (v1.1.1) [78] packages.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s13059-021-02519-4.

Additional file 1: Supplementary Figure 1. Delay in adding tools to the database. **Supplementary Figure 2**. Fit of the number of tools over time. **Supplementary Figure 3.** Platform proportions by category. **Supplementary Figure 4.** Number of categories per tool. **Supplementary Figure 5.** Dependencies between tools. **Supplementary Figure 6.** Correlations between publications and tools metrics. **Supplementary Table 1.** Coefficients for publications models. **Supplementary Table 2.** Coefficients for tools models.

Additional file 2. Review history.

Acknowledgements

We would like to thank everyone in the community who has contributed to scRNA-tools since it began, including those who submitted new tools or updates, contributed code or provided advice and discussion. We also want to acknowledge the work of the developer community who continue to produce new tools and methods to drive the field forward. Lastly, thank you to Fabiola Curion, Carlos Talavera-Lopez, Malte Lücken, Lukas Huemos and Karin Hrovatin for their comments on drafts of the manuscript.

Review history

The review history is available as Additional file 2.

Peer review information

Barbara Cheifet was the primary editor of this article and managed its editorial process and peer review in collaboration with the rest of the editorial team.

Authors' contributions

LZ designed the study, performed the analysis, created the figures and wrote the manuscript. LZ is also responsible for updating and maintaining the scRNA-tools database and website (with contributions from the community). FJT supervised the study and contributed to the manuscript. The authors read and approved the final manuscript.

Authors' information

Twitter @_lazappi_ (Luke Zappia); Twitter @fabian_theis (Fabian J. Theis)

Funding

This work was supported by the Bavarian Ministry of Science and the Arts in the framework of the Bavarian Research Association "ForInter" (Interaction of human brain cells). Open Access funding enabled and organized by Projekt DEAL.

Availability of data and materials

The scRNA-tools website and database are publicly available at https://www.scrna-tools.org/ [24]. The raw database files, as well as code for managing the database and website, are available on GitHub under an MIT license [79]. Code and data files for the analysis presented here can be found on GitHub [80] and from Zenodo [81], also under an MIT license.

Declaration

Competing interests

FJT reports receiving consulting fees from ImmunAI and an ownership interest in Dermagnostix. The other author declares no competing interests.

Author details
[1]Institute of Computational Biology, Helmholtz Zentrum München, 85764 Neuherberg, Germany. [2]Department of Mathematics, Technical University of Munich, 85748 Garching bei München, Germany. [3]TUM School of Life Sciences Weihenstephan, Technical University of Munich, 85354 Freising, Germany.

Received: 24 August 2021 Accepted: 14 October 2021
Published online: 29 October 2021

References

1. Tang F, Barbacioru C, Wang Y, Nordman E, Lee C, Xu N, et al. mRNA-Seq whole-transcriptome analysis of a single cell. Nat Methods. 2009;6:377–82. Available from:. https://doi.org/10.1038/nmeth.1315.
2. Angerer P, Simon L, Tritschler S, Wolf FA, Fischer D, Theis FJ. Single cells make big data: new challenges and opportunities in transcriptomics. Curr Opin Syst Biol. 2017;4:85–91 Available from: http://www.sciencedirect.com/science/article/pii/S245231001730077X.
3. Svensson V, Vento-Tormo R, Teichmann SA. Exponential scaling of single-cell RNA-seq in the past decade. Nat Protoc. 2018;13:599. Available from. https://doi.org/10.1038/nprot.2017.149.
4. Cusanovich DA, Daza R, Adey A, Pliner HA, Christiansen L, Gunderson KL, et al. Multiplex single cell profiling of chromatin accessibility by combinatorial cellular indexing. Science. 2015;348:910–4. Available from:. https://doi.org/10.1126/science.aab1601.
5. Buenrostro JD, Wu B, Litzenburger UM, Ruff D, Gonzales ML, Snyder MP, et al. Single-cell chromatin accessibility reveals principles of regulatory variation. Nature. 2015;523:486–90. Available from:. https://doi.org/10.1038/nature14590.
6. Hu Y, Huang K, An Q, Du G, Hu G, Xue J, et al. Simultaneous profiling of transcriptome and DNA methylome from a single cell. Genome Biol. 2016;17:88. Available from:. https://doi.org/10.1186/s13059-016-0950-z.
7. Mulqueen RM, Pokholok D, Norberg SJ, Torkenczy KA, Fields AJ, Sun D, et al. Highly scalable generation of DNA methylation profiles in single cells. Nat Biotechnol. 2018; Available from:. https://doi.org/10.1038/nbt.4112.
8. Stoeckius M, Hafemeister C, Stephenson W, Houck-Loomis B, Chattopadhyay PK, Swerdlow H, et al. Simultaneous epitope and transcriptome measurement in single cells. Nat Methods. 2017;14:865–8. Available from:. https://doi.org/10.1038/nmeth.4380.
9. Vickovic S, Ståhl PL, Salmén F, Giatrellis S, Westholm JO, Mollbrink A, et al. Massive and parallel expression profiling using microarrayed single-cell sequencing. Nat Commun. 2016;7:13182. Available from:. https://doi.org/10.1038/ncomms13182.
10. Lubeck E, Cai L. Single-cell systems biology by super-resolution imaging and combinatorial labeling. Nat Methods. 2012;9:743–8. Available from:. https://doi.org/10.1038/nmeth.2069.
11. Duò A, Robinson MD, Soneson C. A systematic performance evaluation of clustering methods for single-cell RNA-seq data. F1000Res. 2018;7 [cited 2018 Jul 27]. Available from: https://f1000research.com/articles/7-1141/v1/pdf.
12. Freytag S, Tian L, Lönnstedt I, Ng M, Bahlo M. Comparison of clustering tools in R for medium-sized 10x Genomics single-cell RNA-sequencing data. F1000Res. 2018;7 [cited 2018 Aug 17]. Available from: https://f1000research.com/articles/7-1297/v1/pdf.
13. Soneson C, Robinson MD. Bias, Robustness and scalability in differential expression analysis of single-cell RNA-Seq data. bioRxiv. 2017:143289 [cited 2017 May 29]. Available from: http://biorxiv.org/content/early/2017/05/28/143289.figures-only.
14. Baik B, Yoon S, Nam D. Benchmarking RNA-seq differential expression analysis methods using spike-in and simulation data. PLoS One. 2020;15:e0232271. Available from:. https://doi.org/10.1371/journal.pone.0232271.
15. Tran HTN, Ang KS, Chevrier M, Zhang X, Lee NYS, Goh M, et al. A benchmark of batch-effect correction methods for single-cell RNA sequencing data. Genome Biol. 2020;21:12. Available from:. https://doi.org/10.1186/s13059-019-1850-9.
16. Luecken MD, Buttner M, Chaichoompu K, Danese A, Interlandi M, Mueller MF, et al. Benchmarking atlas-level data integration in single-cell genomics. bioRxiv. 2020; [cited 2020 May 25]. p. 2020.05.22.111161. Available from: https://www.biorxiv.org/content/10.1101/2020.05.22.111161v1.abstract?%3Fcollection=.
17. Andrews TS, Kiselev VY, McCarthy D, Hemberg M. Tutorial: guidelines for the computational analysis of single-cell RNA sequencing data. Nat Protoc. 2020; Available from:; 10.1038/s41596-020-00409-w.
18. Luecken MD, Theis FJ. Current best practices in single-cell RNA-seq analysis: a tutorial. Mol Syst Biol. 2019;15 [cited 2019 Jun 20]. Available from: https://www.embopress.org/doi/full/10.15252/msb.20188746.
19. Amezquita RA, Lun ATL, Becht E, Carey VJ, Carpp LN, Geistlinger L, et al. Orchestrating single-cell analysis with Bioconductor. Nat Methods. 2019:1–9 [cited 2019 Dec 3]. Available from: https://www.nature.com/articles/s41592-019-0654-x.
20. Svensson V, da Veiga BE, Pachter L. A curated database reveals trends in single-cell transcriptomics. Database. 2020;2020. Available from:. https://doi.org/10.1093/database/baaa073.
21. Davis S, Kutum R, Zappia L, Sorenson J, Kiselev V, Olivier P, et al. Awesome single cell. [cited 2018 Jun 20]. Available from: https://zenodo.org/record/1294021
22. Vilella AJ. SingleCell Omics spreadsheet [Internet]. Available from: bit.ly/scellmarket. Accessed 26 Sept 2021.
23. Zappia L, Phipson B, Oshlack A. Exploring the single-cell RNA-seq analysis landscape with the scRNA-tools database. PLoS Comput Biol. 2018;14:e1006245. Available from:. https://doi.org/10.1371/journal.pcbi.1006245.
24. Zappia L, Phipson B, Oshlack A, Theis FJ, The scRNA-tools community. scRNA-tools: A catalogue of tools for scRNA-seq analysis. The scRNA-tools website. Available from: https://www.scrna-tools.org/. Accessed 26 Sept 2021.
25. Köster J, Rahmann S. Snakemake - a scalable bioinformatics workflow engine. Bioinformatics. 2012;28:2520–2. Available from:. https://doi.org/10.1093/bioinformatics/bts480.
26. Mölder F, Jablonski KP, Letcher B, Hall MB, Tomkins-Tinch CH, Sochat V, et al. Sustainable data analysis with Snakemake. F1000Res. 2021;10:33 [cited 2021 Jan 22]. Available from: https://f1000research.com/articles/10-33/v1/pdf.
27. Di Tommaso P, Chatzou M, Floden EW, Barja PP, Palumbo E, Notredame C. Nextflow enables reproducible computational workflows. Nat Biotechnol. 2017;35:316–9. Available from:. https://doi.org/10.1038/nbt.3820.

28. R Core Team. R: A Language and Environment for Statistical Computing. Vienna: R Foundation for Statistical Computing; 2021. Available from: https://www.R-project.org/
29. Van Rossum G, Drake FL Jr. Python tutorial. Centrum voor Wiskunde en Informatica Amsterdam; 1995.
30. Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, et al. Orchestrating high-throughput genomic analysis with Bioconductor. Nat Methods. 2015;12:115–21. Available from:. https://doi.org/10.1038/nmeth.3252.
31. Open Source Initiative. Licenses & Standards. Open Source Initiative. [cited 2021 Aug 9]. Available from: https://opensource.org/licenses. Accessed 9 Aug 2021.
32. Satija R, Farrell JA, Gennert D, Schier AF, Regev A. Spatial reconstruction of single-cell gene expression data. Nat Biotechnol. 2015;33:495–502. [cited 2016 May 10]. Available from:. https://doi.org/10.1038/nbt.3192.
33. Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. Genome Biol. 2018;19:15. Available from:. https://doi.org/10.1186/s13059-017-1382-0.
34. Burkhardt DB, Stanley JS 3rd, Tong A, Perdigoto AL, Gigante SA, Herold KC, et al. Quantifying the effect of experimental perturbations at single-cell resolution. Nat Biotechnol. 2021; Available from:. https://doi.org/10.1038/s41587-020-00803-5.
35. Osorio D, Zhong Y, Li G, Xu Q, Hillhouse AE, Chen J, et al. scTenifoldKnk: a machine learning workflow performing virtual knockout experiments on single-cell gene regulatory networks. Cold Spring Harbor Laboratory. 2021; [cited 2021 Mar 24]. p. 2021.03.22.436484. Available from: https://www.biorxiv.org/content/10.1101/2021.03.22.436484v1?ct=.
36. Lotfollahi M, Wolf FA, Theis FJ. scGen predicts single-cell perturbation responses. Nat Methods. 2019;16:715–21. Available from:. https://doi.org/10.1038/s41592-019-0494-8.
37. Mereu E, Lafzi A, Moutinho C, Ziegenhain C, McCarthy DJ, Álvarez-Varela A, et al. Benchmarking single-cell RNA-sequencing protocols for cell atlas projects. Nat Biotechnol. 2020;38:747–55. Available from:. https://doi.org/10.1038/s41587-020-0469-4.
38. Jerber J, Seaton DD, Cuomo ASE, Kumasaka N, Haldane J, Steer J, et al. Population-scale single-cell RNA-seq profiling across dopaminergic neuron differentiation. Nat Genet. 2021;53:304–12. Available from:. https://doi.org/10.1038/s41588-021-00801-6.
39. Cheng S, Li Z, Gao R, Xing B, Gao Y, Yang Y, et al. A pan-cancer single-cell transcriptional atlas of tumor infiltrating myeloid cells. Cell. 2021;184:792–809.e23. Available from. https://doi.org/10.1016/j.cell.2021.01.010.
40. Regev A, Teichmann SA, Lander ES, Amit I, Benoist C, Birney E, et al. The Human Cell Atlas. Elife. 2017;6. Available from:. https://doi.org/10.7554/eLife.27041.
41. Saelens W, Cannoodt R, Todorov H, Saeys Y. A comparison of single-cell trajectory inference methods. Nat Biotechnol. 2019; Available from:. https://doi.org/10.1038/s41587-019-0071-9.
42. Tritschler S, Büttner M, Fischer DS, Lange M, Bergen V, Lickert H, et al. Concepts and limitations for learning developmental trajectories from single cell genomics. Development. 2019;146. Available from:. https://doi.org/10.1242/dev.170506.
43. La Manno G, Soldatov R, Zeisel A, Braun E, Hochgerner H, Petukhov V, et al. RNA velocity of single cells. Nature. 2018;560:494–8. Available from. https://doi.org/10.1038/s41586-018-0414-6.
44. Bergen V, Lange M, Peidli S, Wolf FA, Theis FJ. Generalizing RNA velocity to transient cell states through dynamical modeling. Nat Biotechnol. 2020; Available from:. https://doi.org/10.1038/s41587-020-0591-3.
45. Fu DY, Hughey JJ. Releasing a preprint is associated with more attention and citations for the peer-reviewed article. Elife. 2019;8. Available from:. https://doi.org/10.7554/eLife.52646.
46. Altmetric [Internet]. [cited 2021 Sep 27]. Available from: https://www.altmetric.com/
47. Fraser N, Momeni F, Mayr P, Peters I. The relationship between bioRxiv preprints, citations and altmetrics. Quantitative Sci Stud. 2020:1–21 Available from: https://direct.mit.edu/qss/article/1/2/618-638/96153.
48. Al-Rubaye A, Sukthankar G. Scoring popularity in GitHub. arXiv [cs.SI]. 2020; Available from: http://arxiv.org/abs/2011.04865.
49. Open Problems. Aggregating and benchmarking open problems in single cell analysis [Internet]. Open Problems in Single Cell Analysis. Available from: https://openproblems.bio/
50. Chamberlain S, Zhu H, Jahn N, Boettiger C, Ram K. rcrossref: client for various "CrossRef" "APIs". 2020. Available from: https://CRAN.R-project.org/package=rcrossref
51. Ram K, Broman K. aRxiv: interface to the arXiv API. 2019. Available from: https://CRAN.R-project.org/package=aRxiv
52. Bryan J, Wickham H. gh: "GitHub" "API". 2021. Available from: https://CRAN.R-project.org/package=gh
53. van der Loo MPJ. The stringdist package for approximate string matching. R J. 2014;6:111. Available from:. https://doi.org/10.32614/rj-2014-011.
54. Cabanac G, Oikonomidi T, Boutron I. Day-to-day discovery of preprint-publication links. Scientometrics. 2021:1–20. Available from:. https://doi.org/10.1007/s11192-021-03900-7.
55. Zappia L. doilinker: link preprints and publications by DOI. 2021. Available from: https://github.com/lazappi/doilinker
56. The Linux Foundation. SPDX License List [Internet]. The Software Package Data Exchange. [cited 2021 Aug 11]. Available from: https://spdx.org/licenses/
57. Wickham H. ggplot2: elegant graphics for data analysis. New York: Springer; 2010.
58. Sievert C. Interactive web-based data visualization with R, plotly, and Shiny: CRC Press, Taylor and Francis Group; 2020. Available from: https://play.google.com/store/books/details?id=0fs1vAEACAAJ
59. Ushey K. renv: project environments. 2021. Available from: https://CRAN.R-project.org/package=renv
60. Wickham H, Hester J. readr: read rectangular text data. 2020. Available from: https://CRAN.R-project.org/package=readr
61. Wickham H, Averick M, Bryan J, Chang W, McGowan L, François R, et al. Welcome to the Tidyverse. JOSS. 2019;4:1686 Available from: https://joss.theoj.org/papers/10.21105/joss.01686.
62. Wickham H, François R, Henry L, Müller K. dplyr: a grammar of data manipulation. 2021. Available from: https://CRAN.R-project.org/package=dplyr
63. Wickham H. tidyr: tidy Messy Data. 2021. Available from: https://CRAN.R-project.org/package=tidyr
64. Wickham H. forcats: tools for Working with Categorical Variables (Factors). 2021. Available from: https://CRAN.R-project.org/package=forcats
65. Henry L, Wickham H. purrr: Functional Programming Tools. 2020. Available from: https://CRAN.R-project.org/package=purrr

66. Ram K. rAltmetric: retrieves Altmerics Data for Any Published Paper from "Altmetric.com". 2017. Available from: https://CRAN.R-project.org/package=rAltmetric
67. Su S, Davis S. BiocPkgTools: Collection of simple tools for learning about Bioc Packages. 2021. Available from: https://github.com/seandavi/BiocPkgTools
68. Wodder JT. Wheelodex [Internet]. Wheelodex. [cited 2021 Aug 13]. Available from: https://www.wheelodex.org/
69. Glenn W. johnnydep: display dependency tree of Python distribution. GitHub. [cited 2021 Aug 13]. Available from: https://github.com/wimglenn/johnnydep.
70. Landau W. The targets R package: a dynamic Make-like function-oriented pipeline toolkit for reproducibility and high-performance computing. J Open Source Softw. 2021;6:2959 Available from: https://joss.theoj.org/papers/10.21105/joss.02959.
71. Silge J, Robinson D. Tidytext: text mining and analysis using tidy data principles in R. J Open Source Softw. 2016;1:37 Available from: http://joss.theoj.org/papers/10.21105/joss.00037.
72. Le Pennec E, Slowikowski K. ggwordcloud: a Word Cloud Geom for "ggplot2". 2019. Available from: https://CRAN.R-project.org/package=ggwordcloud
73. Patil I. Visualizations with statistical details: the "ggstatsplot" approach. J Open Source Softw. 2021;6:3167 Available from: https://joss.theoj.org/papers/10.21105/joss.03167.
74. Wilke CO. ggtext: Improved Text Rendering Support for "ggplot2". 2020. Available from: https://CRAN.R-project.org/package=ggtext
75. Slowikowski K. ggrepel: automatically position non-overlapping text labels with "ggplot2". 2021. Available from: https://CRAN.R-project.org/package=ggrepel
76. Hester J. glue: interpreted string Literals (2017). Available from: https://CRAN.R-project.org/package=glue
77. Wilke CO. cowplot: streamlined plot theme and plot annotations for "ggplot2". 2020. Available from: https://CRAN.R-project.org/package=cowplot
78. Pedersen TL. patchwork: the composer of plots. 2020. Available from: https://CRAN.R-project.org/package=patchwork
79. Zappia L, Wells D, Wolf A, Gitter A, et al. scRNA-tools: table of software for the analysis of single-cell RNA-seq data: Github. Available from: https://github.com/scRNA-tools/scRNA-tools
80. Zappia L, Theis FJ. 1000-tools-paper: code and analysis for the 1000 tools paper. GitHub. Available from: https://github.com/scRNA-tools/1000-tools-paper.
81. Zappia L, Theis FJ. 1000 tools paper: Zenodo; 2021. Available from: https://zenodo.org/record/5195628

Publisher's Note