OXFORD (GIGA)n SCIENCE

GigaScience, 2025, **14**, 1–18
DOI: 10.1093/gigascience/giae107
TECH NOTE

Galaxy as a gateway to bioinformatics: Multi-Interface Galaxy Hands-on Training Suite (MIGHTS) for scRNA-seq

Camila L. Goclowski [1,†], Julia Jakiela [2,†], Tyler Collins [3], Saskia Hiltemann [4], Morgan Howells [5], Marisa Loach [6], Jonathan Manning [7], Pablo Moreno [8], Alex Ostrovsky [3], Helena Rasche [4], Mehmet Tekman [9], Graeme Tyson [6], Pavankumar Videm [10], and Wendi Bacon [6,*]

[1]Eccles Institute of Human Genetics, University of Utah, Salt Lake City, UT, 84112, USA
[2]School of Chemistry, University of Edinburgh, Edinburgh, EH9 3FJ, UK
[3]Department of Computer Science, John Hopkins Medical Institution, Baltimore, MD, 21224, USA
[4]Erasmus Medical Center, Rotterdam, Zuid-Holland, 3015 GD, Netherlands
[5]School of Computing & Communications, The Open University, Milton Keynes, Buckinghamshire, MK7 6AA, UK
[6]School of Life, Health & Chemical Sciences, The Open University, Milton Keynes, Buckinghamshire, MK7 6AA, UK
[7]European Bioinformatics Institute, European Molecular Biology Laboratory, Hinxton, CB10 1SD, UK
[8]Early Computational Oncology, AstraZeneca, Cambridge, CB2 0AA, UK
[9]Division of Pharmacology and Toxicology, University of Freiburg, Freiburg im Breisgau, Baden-Württemberg, 79098, Germany
[10]Department of Computer Science, University of Freiburg, Freiburg im Breisgau, Baden-Württemberg, 79098, Germany
*Correspondence address. Wendi Bacon, School Admin, Office School of Life, Health & Chemical Sciences, The Open University Walton Hall Milton Keynes MK7 6AA United Kingdom; E-mail: wendi.bacon@open.ac.uk
[†]Co-first authors C.G. and J.J. contributed equally to this publication's curation and manuscript preparation. As such, they receive equal credit in the publication. Co-first authors and middle authors have been ordered alphabetically.

Abstract

Background: Bioinformatics is fundamental to biomedical sciences, but its mastery presents a steep learning curve for bench biologists and clinicians. Learning to code while analyzing data is difficult. The curve may be flattened by separating these two aspects and providing intermediate steps for budding bioinformaticians. Single-cell analysis is in great demand from biologists and biomedical scientists, as evidenced by the proliferation of training events, materials, and collaborative global efforts like the Human Cell Atlas. However, iterative analyses lacking reinstantiation, coupled with unstandardized pipelines, have made effective single-cell training a moving target.

Findings: To address these challenges, we present a Multi-Interface Galaxy Hands-on Training Suite (MIGHTS) for single-cell RNA sequencing (scRNA-seq) analysis, which offers parallel analytical methods using a graphical interface (buttons) or code. With clear, interoperable materials, MIGHTS facilitates smooth transitions between environments. Bridging the biologist–programmer gap, MIGHTS emphasizes interdisciplinary communication for effective learning at all levels. Real-world data analysis in MIGHTS promotes critical thinking and best practices, while FAIR data principles ensure validation of results. MIGHTS is freely available, hosted on the Galaxy Training Network, and leverages Galaxy interfaces for analyses in both settings. Given the ongoing popularity of Python-based (Scanpy) and R-based (Seurat & Monocle) scRNA-seq analyses, MIGHTS enables analyses using both.

Conclusions: MIGHTS consists of 11 tutorials, including recordings, slide decks, and interactive visualizations, and a demonstrated track record of sustainability via regular updates and community collaborations. Parallel pathways in MIGHTS enable concurrent training of scientists at any programming level, addressing the heterogeneous needs of novice bioinformaticians.

Keywords: training, STEM education, Galaxy project, single-cell RNA-seq analysis, scRNA-seq, bioinformatics, reproducibility, sustainability

Introduction

Although bioinformatics is critical to basic biological and applied biomedical research, there remains a shortage of scientists with bioinformatics expertise [1]. As computational domains of biology continue to grow, bioinformatics play an important role in groundbreaking discoveries [2–5]. Thinking computationally about biological processes has been shown to produce more accurate models [6] and enhance problem-solving [7]. However, it is important to note that bioinformatics often requires many expensive resources, such as computational infrastructure, maintenance, and training [8]. Financial barriers can limit access to training and research [9–13]. As such, many bioinformaticians rarely receive formal training in the field [8], and teaching bioinformatics is notably difficult.

Integrating bioinformatics into undergraduate curricula may address this gap [1, 14]. Bioinformatics has been introduced in high schools, where it was shown to improve awareness, engagement, and self-efficacy of students, leading to increased interest in STEM careers [15]. Pharmaceutical companies need biomedical analysts [16], most employers in the life sciences prefer some competency in software analyses [17], and the use of bioinformatic analyses to characterize novel cell types and lineages [18]

Received: August 13, 2024. **Revised:** October 28, 2024. **Accepted:** November 26, 2024

has surged. In response, institutes are beginning to teach foundational computing skills to biologists [14, 19].

Materials that focus on problem-solving, interactivity, and cooperative learning have demonstrated enhanced learning outcomes [20], and bioinformatics has effectively been taught by emphasizing interdisciplinary problem-solving [21]. To standardize training, a list of "rules" was identified to teach scientists to program: beginning with the end in mind, taking small steps forward, and focusing on individual tasks [20]. The "end in mind" requires domain-specific understanding (i.e., identifying cell types via marker genes) while the individual tasks require programming skills (R, Python, troubleshooting, etc.). This duality forces new biofinformaticians to learn and apply 2 new skill sets simultaneously [22–24]. The need to embed computing into science is not novel [25], but blending skills across disciplines is not without challenges [26].

The Galaxy Training Network (GTN) boasts tutorials for analysis across a range of fields, all publicly available and accessible by URL [27]. The GTN provides free training infrastructure to fast-track trainees via live courses in which trainers are available to monitor and assist participants [28]. This supports all, but especially low-resource institutions', engagement with bioinformatic training and has additionally been tested for native Spanish speakers [28]. Integrating these free resources into undergraduate curricula has been successful [27], as training materials include interactive features based on research-backed pedagogies. Separation of learning components has previously been suggested as an effective method [29] but raises the question: how can coding and complex bioinformatic analyses be isolated from one another?

Here, we directly address the need to separate the two for training. Leveraging the Galaxy Graphical User Interface (GUI) and the GTN, we present the Multi-Interface Galaxy Hands-on Training Suite (MIGHTS): a single-cell RNA sequencing (scRNA-seq) tutorial suite enabling a smooth transition from data analysis in a button-based, user-friendly environment [30] to a more advanced, flexible programming environment. Using a sample dataset, MIGHTS guides users through the steps necessary to accomplish commonly published scRNA-seq analyses and visualizations, including generating a matrix, combining datasets, filtering, plotting, and general exploration of the data, as well as trajectory inference of a dataset known to represent a developmental spectrum. The sample dataset provided for use with the suite reveals a delay in thymic maturation in growth-restricted neonatal mice [31]. MIGHTS offers multiple routes of scRNA-seq analysis, allowing a button-based or coding-based version of the same commonly published workflows. Notably, MIGHTS offers opportunities for a heterogeneous student population ranging from programming-friendly to programming-fearful, expanding access to critical skills required for effective bioinformatic analyses and biomedical and life science research.

Methods

Multienvironment

MIGHTS consists of 11 tutorials: 6 button-based (BB) and 5 in a programming environment (PE) each making use of common analysis packages (Table 1).

The Galaxy GUI features "click-to-run" buttons that execute programming functions [30].Users select andset parameters from drop-down lists and input boxes (Fig. 1, left column: button-based). Each tool includes help text to guide users anddescribe the flexibility ofthe tool's function.

Galaxy's interactive programming environments [61] are where the PE tutorials take place. Tutorials may be downloaded as RMarkdown or Jupyter notebooks [62, 63], or users may copy, paste, and run each executable code-containing cell from the PE text (Fig. 1, right column: programming environment). Jupyter and RMarkdown notebooks may be exported at the conclusion of each coded tutorial for easy reference or repetition.

Multilevel

MIGHTS caters to the following 3 learning pathways: BB to PE, straight to PE, and PE with BB (Fig. 2).

In the first case, BB tutorials guide beginners through the key steps of scRNA-seq analysis, establishing familiarity with the methods and learning to interpret results. Next, users may repeat the analysis in the PE, focusing on programming skills while becoming familiar with the languages and libraries commonly used for scRNA-seq analysis (Fig. 2A). If a user has experience programming and wants a more flexible analysis, they may begin with the PE tutorials, learning methods with more advanced functionality (Fig. 2B). Alternatively, experienced bioinformaticians may utilize Galaxy's Interactive Environments to learn new analyses or run computationally demanding steps that they are unable to do locally (Fig. 2C).

Multilanguage

scRNA-seq analysis is commonly performed in both R-based (using Seurat [46–50] & Monocle [56]) and Python-based (using Scanpy [41]) environments. Therefore, parallel analyses were created across BB and PE as well as across programming languages—effectively demonstrating multiple methods of analysis and data validation (Fig. 3). Users may conduct a typical, full scRNA-seq analysis workflow in R or Python in addition to on a GUI or in a PE.

Research-relevant skills

MIGHTS demonstrates the use of many frequently used data types and packages for scRNA-seq analyses (Table 1), preparing users with research-relevant skills. Broadly applicable use of programming functions, algorithms, and troubleshooting lends itself to increased creativity and critical thinking [64, 65]. This also improves users' employability and reaches scientists in various research groups, regardless of the method they prefer.

Tutorials

Each tutorial begins with data import. The data used in MIGHTS come from a published study by Bacon et al. [31], describing a mouse model of fetal growth restriction that is publicly available from the EMBL-EBI ArrayExpress under accession number E-MTAB-6945 and may additionally be explored in the Single Cell Expression Atlas [66]. Tutorials in MIGHTS work with the same data throughout to demonstrate analyses using different methods and tools. Tutorials use real, uncurated data, which have simply been subsampled to enhance computational efficiency. The source data are the same, but each analysis allows import of a unique data file to start. Tutorials are designed to be completed in order but may be performed out of order—if a user wishes to learn how to cluster cells using Scanpy (RRID:SCR_018139), for example, they may select the dedicated tutorial and start with the provided, preprocessed file(s).

Table 1: MIGHTS tutorials with used packages and data types

Analysis	Environment | tutorial (Package/Language)	Packages	Data types
Preprocessing	BB | Generating a single-cell matrix using Alevin	Salmon [32] with Alevin [33]	FASTQ
	BB | Combining single-cell datasets after preprocessing	dropletUtils [34, 35] (emptyDrops [34])	FASTA
	PE | Generating a single-cell matrix using Alevin and combining datasets (bash + R [36])	atlas-gene-annotation-manipulation [37]	GTF
		tximeta [38] (PE)	SingleCellExperiment Object
		biomaRt [39, 40] (PE)	SummarizedExperiment; AnnData
Plotting and interpretation	BB | Filter, plot, and explore single-cell RNA-seq data (Scanpy)	Scanpy [41]	AnnData
	PE | Filter, plot, and explore single-cell RNA-seq data (Scanpy, Python [42])	igraph [43] (PE)	
		louvain [44] (PE)	
		pandas [45] (PE)	
	BB | Filter, plot, and explore single-cell RNA-seq data (Seurat)	Seurat [46–50]	AnnData (for conversion to Seurat)
	PE | Filter, plot, and explore single-cell RNA-seq data (Seurat, R)	Matrix [51] (PE)	Seurat Object
		dplyr [52] (PE)	
Trajectories	BB | Inferring single-cell trajectories (Scanpy)	Scanpy [41]	AnnData
	PE | Inferring single-cell trajectories (Scanpy, Python)	fa2 [53] (PE)	
		igraph [43] (PE)	
		louvain [44] (PE)	
		numpy [54] (PE)	
		matplotlib [55] (PE)	
	BB | Inferring single-cell trajectories (Monocle3)	Monocle [56]	Cell Data Set
	PE | Inferring single-cell trajectories (Monocle3, R)	anndata [57] (PE)	AnnData (for conversion to Cell Data Set)
		viridislite [58] (PE)	
		magrittr [59] (PE)	
		Rcpp [60] (PE)	
		biomaRt [39, 40] (PE)	

MIGHTS's full workflow consists of three sequential analyses aligning with standard scRNA-seq pipelines [67] and allowing users to compare results across methods.

Generating a single-cell matrix using Alevin and combining datasets

The first two tutorials demonstrate the transformation of a FASTA sequencing file into a count matrix (Fig. 4, Supplementary Figs. S1, S2). The BB tutorial describes principles of transcriptome quantification, while the PE tutorial introduces users to the many means of installing required packages. This tutorial will take users from aligned read counts to a single-cell experiment (SCE) object, which may be further analyzed and converted in RStudio (RRID:SCR_000432) or Jupyter Notebook.

Users first generate a transcript-to-gene map using FASTQ files, a GTF file, and a reference FASTA transcriptome. A Salmon index of the transcriptome is created, and a cell-by-gene count matrix is built using Alevin. The BB tutorial combines these two steps using one Galaxy tool and demonstrates basic quality control checks, including a description of the barcode rank plot "knee detection" method.

The PE tutorial identifies empty droplets, adds cell and gene-level metadata, and flags empty droplets based on transcript count. Droplet annotation is corrected for false discovery, and the matrix is filtered before combining the datasets manually. PE users save and export files while converting formats to SCE so they are compatible with downstream analyses.

The BB tutorial incorporates metadata straight from a GTF file using a tool to extract gene names and IDs and to label mitochondrial transcripts. The generated gene information is assigned to the matrix, which is subsequently transposed for compatibility with tools meant for 10x Genomics software. EmptyDrops is then used to remove empty droplets.

Half of the remaining suite emphasizes use of AnnData-compatible packages. To prepare users, tutorials conclude with 1 final format conversion from SCE to AnnData with the SCEasy tool. Once each object has been converted, the BB user concatenates them with a Galaxy tool. The BB tutorial sets the user and their objects up for the next tutorial by adding a number of useful metrics to help visualize the data in the coming tutorial(s). Workflows for each tutorial are shown in Fig. 4.

Filter, plot, and explore with Scanpy

These tutorials filter and analyze a preprocessed scRNA-seq matrix using Scanpy (Fig. 5). PE users leverage Python via a Jupyter Notebook to filter the data for noise, as well as accomplish common visualizations and differential expression analysis across clusters for the purpose of cell type labeling.

The PE tutorial imports a raw AnnData file and demonstrates storage as a pandas dataframe, while users iteratively visualize data with violin and scatter plots to determine filtering thresholds. Users filter the data to remove technical artifacts and poor-quality cells. The PE alternatively uses Boolean indexing for this rather than Scanpy's built-in functions. Users remove transcripts

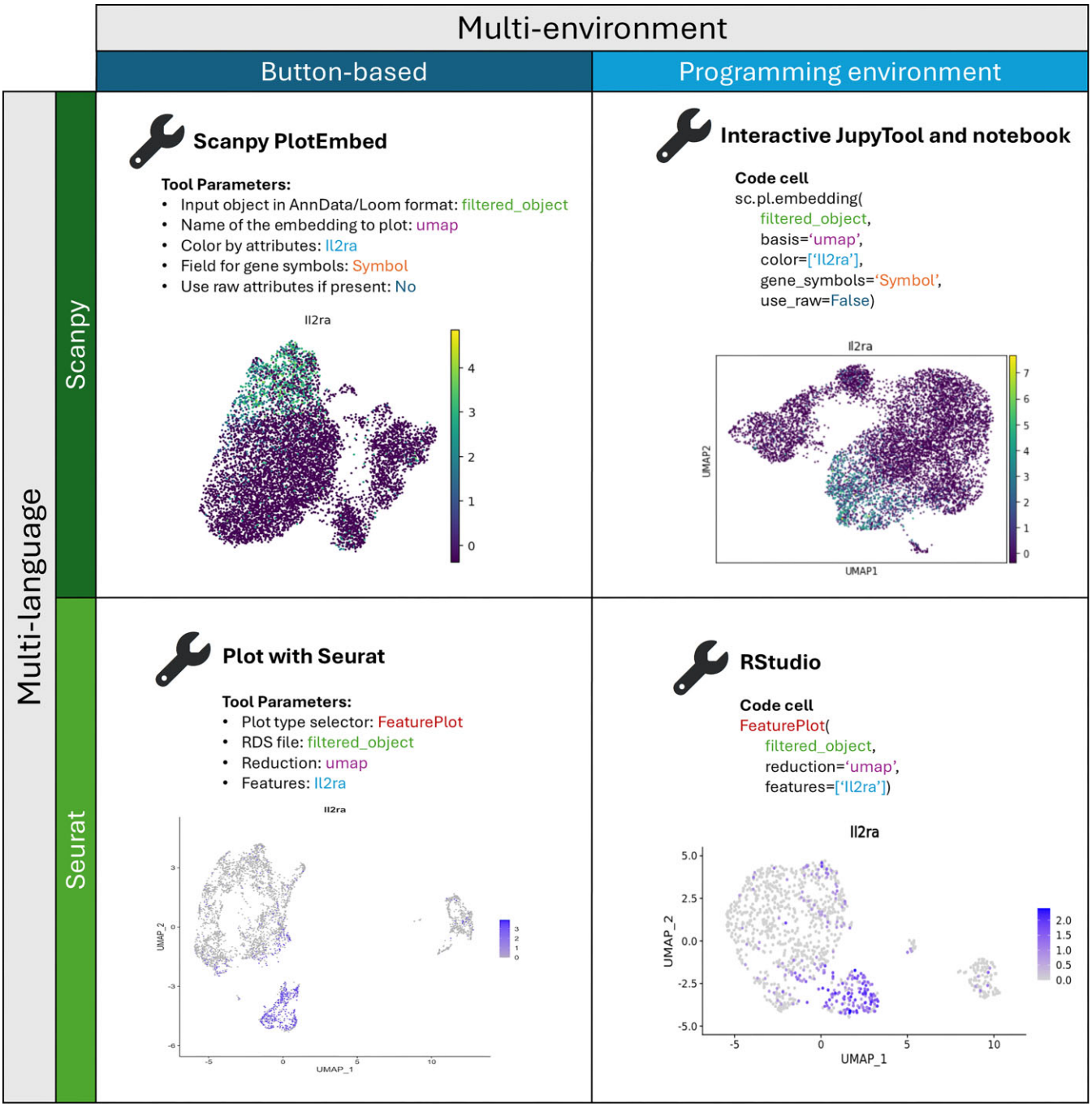

Figure 1: Performing the same step (plotting a marker gene: Il2ra on UMAP embedding) using Galaxy GUI and programming environment with both Python-based Scanpy and R-based Seurat packages. The resulting plots, although slightly different, represent the same biological information, no matter if the BB or PE method was used.

no longer expressed in more than 3 cells and are prompted to compare different thresholds for the filtering of genes.

Log normalization aligns gene expression along a normal distribution. The PE tutorial includes a description of how normalization works and what other methods exist. Variable genes are flagged for use in more computationally demanding steps. Scaling the data ensures all genes have equal variance and a zero mean, creating a matrix that is compatible with downstream analyses.

Users next reduce the dimensionality of the matrix to allow visualization and interpretation. Principal component analysis (PCA) is performed to calculate the most descriptive principal components (PCs). Users plot PCs against the standard variation they describe, visualizing how PCs relate to variance in their data. The PCs are used to compute a k-nearest-neighbors graph, storing a representation of connections between and across cells. Final dimensionality reductions are performed with t-distributed stochastic neighbor embedding (tSNE) [68] and Uniform Manifold Approximation and Projection (UMAP) [69]—both methods reducing the data down to 2 dimensions for visualization.

Scanpy's clustering function(s) assign each cell to a cluster based on transcriptomic similarity. The tutorials describe clustering algorithms and prompt users to experiment with multi-

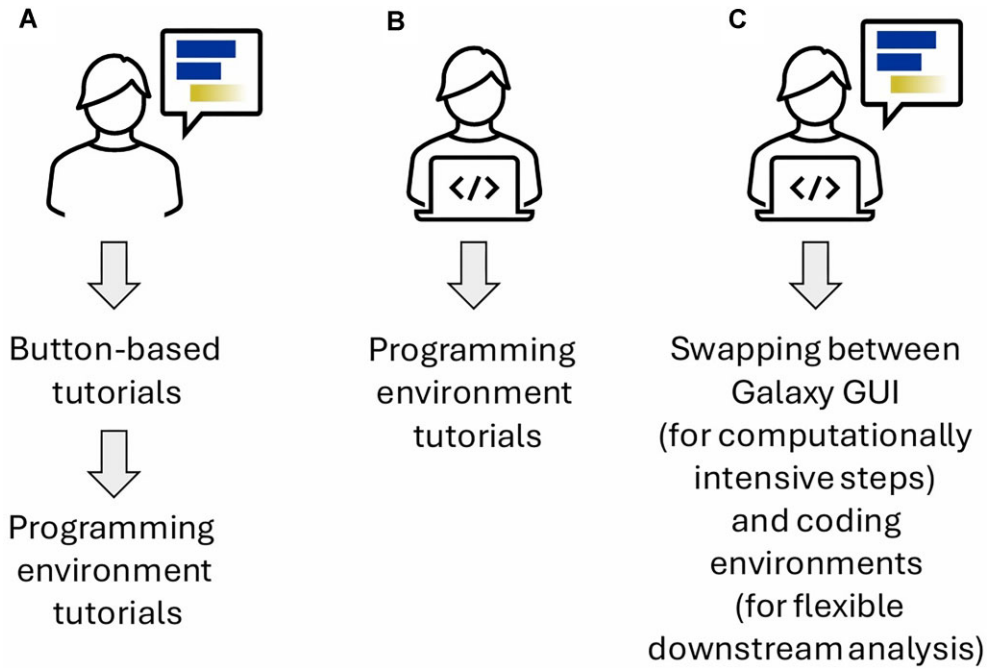

Figure 2: Representation of 3 possible user journeys using MIGHTS. (A) A beginner starting from button-based (BB) tutorials who can then move to the programming environment (PE). (B) An experienced programmer who can start the analysis directly from the PE, skipping introductory BB tutorials. (C) A skilled user who can optimize analyses by swapping between Galaxy GUI to perform computationally intensive steps and a programming environment for more flexible analyses.

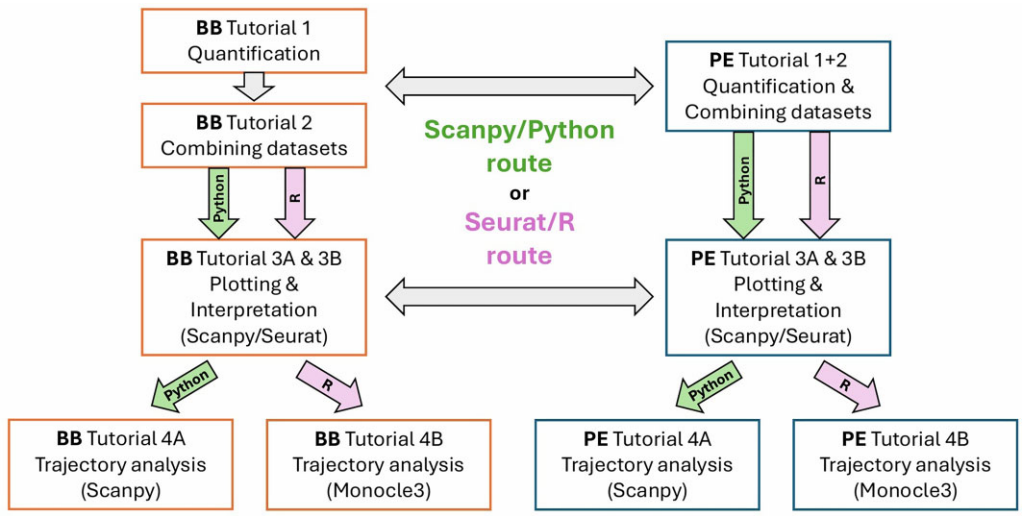

Figure 3: A diagram of the connections of tutorials. It highlights that the languages and packages used in BB and PE tutorials are consistent and allow moving between them easily.

ple clustering resolutions, adjusting such that the assigned clusters visually represent what is understood to be biologically accurate. Scanpy's rank_genes_groups identifies the most representative transcripts for each cluster and genotype, and PE users transform the output into a data frame.

Users visualize all three dimensionality reductions, different clustering resolutions, and the expression of marker genes. A table of marker genes per cell type from the literature is provided so that the user may inspect their expression patterns and map them to the correct cluster(s). Users label each cluster with a cell type, and plots are saved to the Galaxy history or notebook to be exported. BB users are additionally introduced to the CELLxGENE (RRID:SCR_021059) [70] tool: an interactive environment for visualizing and exploring scRNA-seq data. Tutorial workflows are shown in Fig. 5 and Supplementary Fig. S3.

Filter, plot, and explore with Seurat

These tutorials closely resemble the workflows of the preceding Scanpy ones, this time making use of the R package Seurat. The workflows teach users the basics of scRNA-seq data analysis, including typical preprocessing; basic visualization with FeaturePlots, DimPlots, and UMAPs; and exploration of differentially expressed genes across clusters and experimental variables (Fig. 6 and Supplementary Fig. S4).

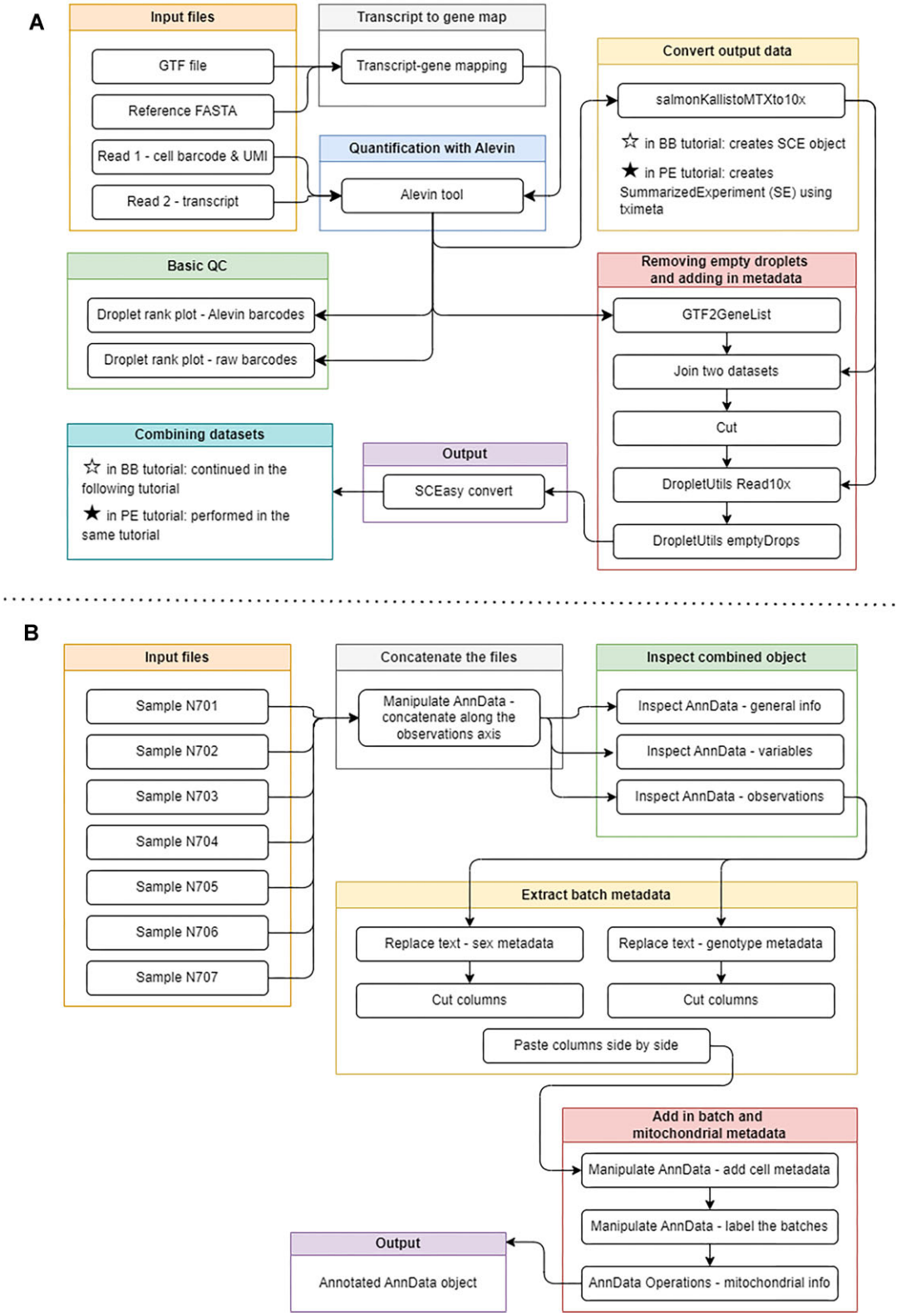

Figure 4: A diagram of workflows in the preprocessing tutorials. (A) Workflow for tutorial "Generating a single-cell matrix using Alevin." Solid stars denote steps specific to the PE tutorial while unfilled stars represent BB-specific ones. (B) Workflow for tutorial "Combining single-cell datasets after pre-processing." Both workflows featured in the BB tutorial are combined in the PE. A figure of an extracted Galaxy workflow is available in Supplementary Figs. S1 and S2.

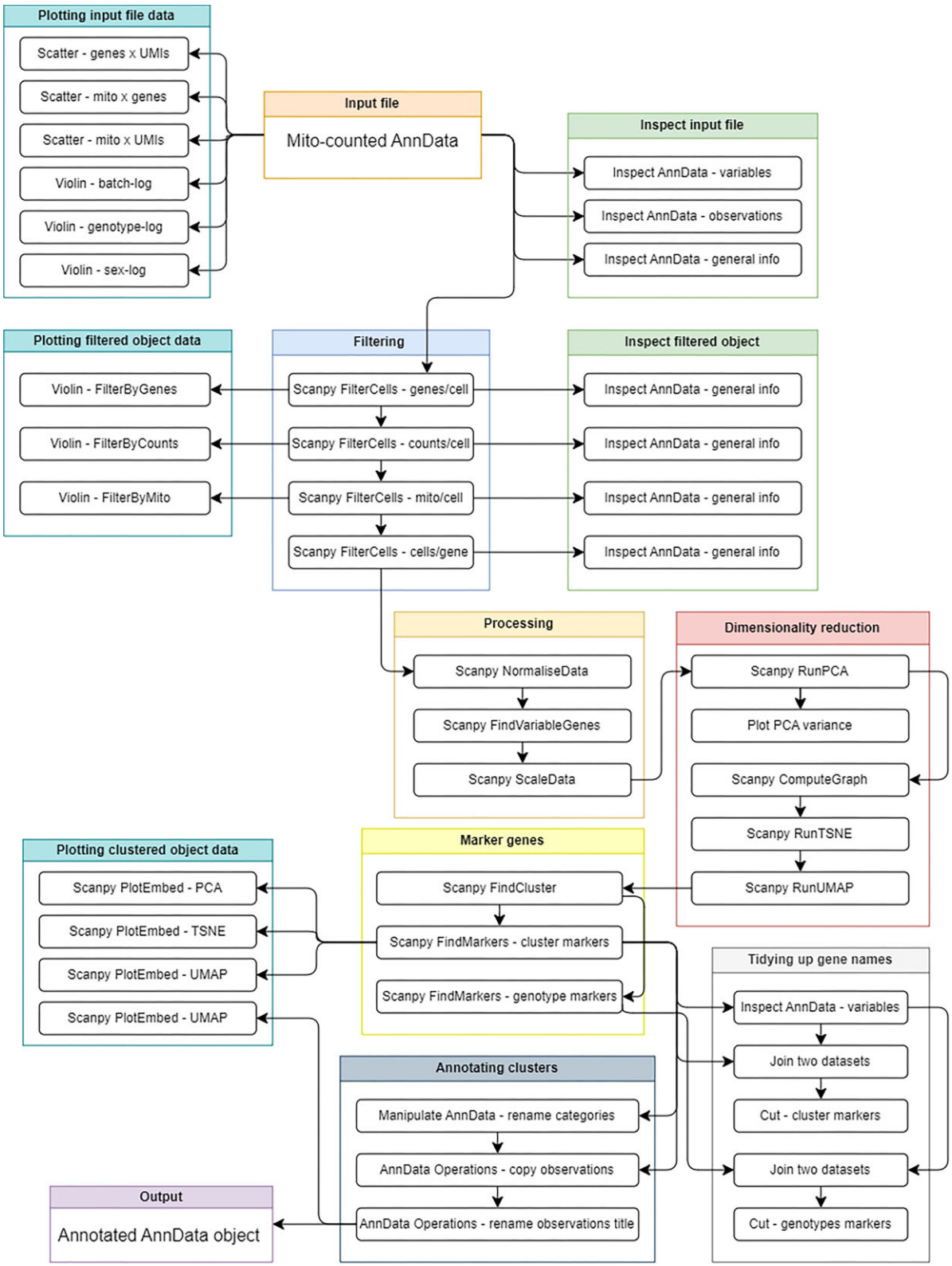

Figure 5: Workflows of plotting and interpretation tutorials: filter, plot, and explore with Scanpy. Features creation of single-cell objects, normalizing data, identifying variable genes, performing dimensionality reduction, identifying clusters, finding marker genes, and interpreting plots. A figure of the extracted Galaxy workflow is available in Supplementary Fig. S3.

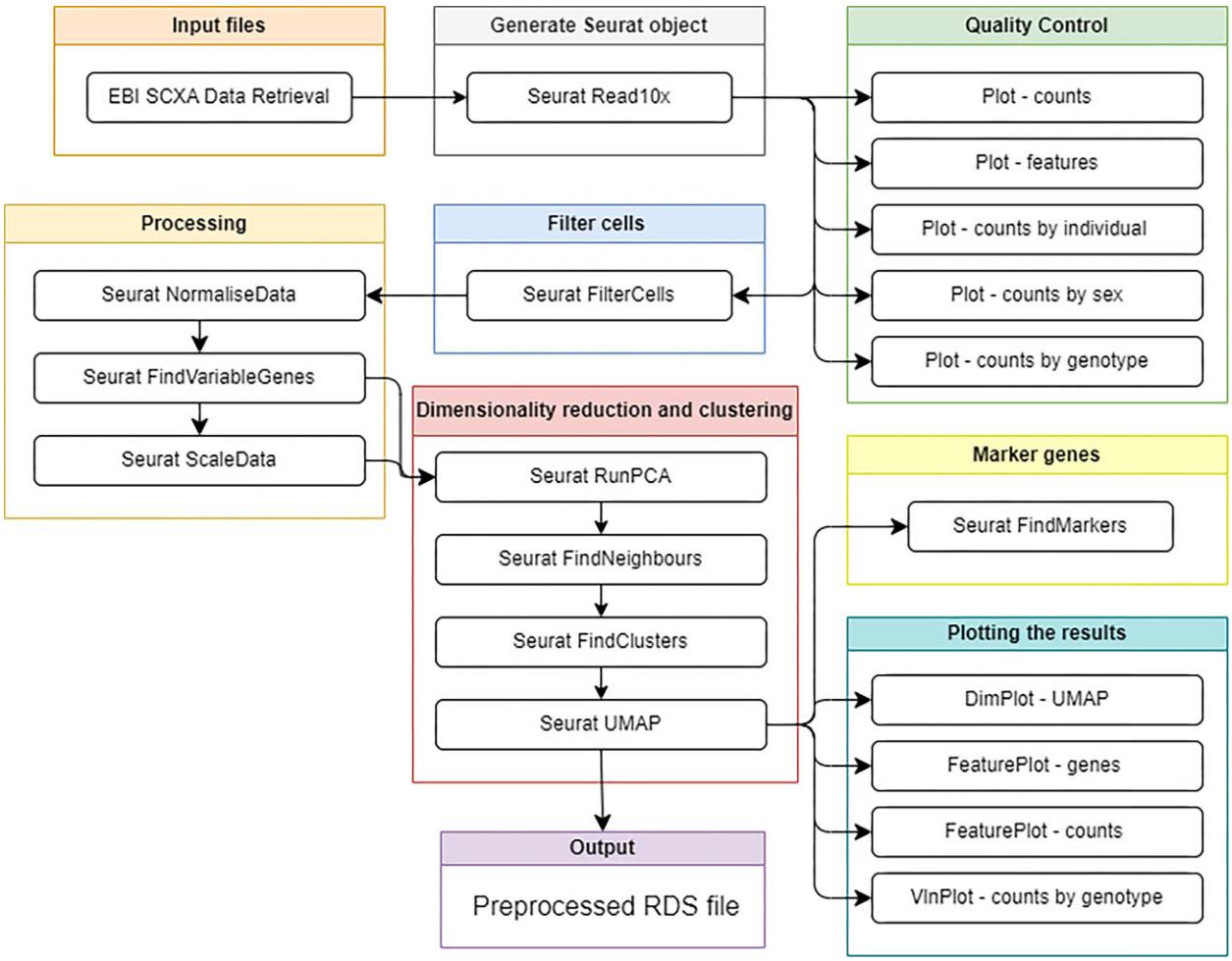

Figure 6: Workflow of the Filter, Plot, and Explore tutorial with Seurat. Features generation of a Seurat object, quality control plots, filtering cells, processing, dimensionality reduction, clustering, finding marker genes, and creating many plots to analyze the results. A figure of the extracted Galaxy workflow is available in Supplementary Fig. S4.

Users import raw counts in both the PE and BB pathways. PE users transition to Galaxy's Interactive RStudio environment, where they are shown how to set up an environment and given an explanation of how and why packages must be loaded prior to use, as well as how to use Galaxy's gx_get() function for data import. Users manually change the column names of the experimental design data for compatibility with Seurat.

Users next generate a Seurat object: BB users with Seurat's Read10X function and PE users by manually applying barcode and feature labels to the matrix for input to Seurat's CreateSeuratObject function. Each method is accompanied by descriptions of the alternatives for creating the same Seurat object.

Users apply cell-level metadata to their objects. PE users incorporate percent of gene expression (per cell) mapping to the mitochondrial genome—a commonly used parameter for quality control and filtering. Tools are currently being updated to enable BB users to do the same.

Users produce and interpret quality control plots to identify filtering thresholds: assessing potential confounders in the data and developing an understanding of how different variables drive this process. The purpose and theory behind commonly used filtering parameters are described so that users may bring the same (or different) strategies to their own analyses. PE users are additionally shown how to preview the number of cells that would be included based on their choice of filtering parameters.

Both users subset their Seurat object–removing cells outside the chosen threshold(s). PE users additionally remove genes that are now expressed at such low frequencies that they will not contribute biological insight.

Next, users process their filtered object. In the BB, processing of the data includes sequentially normalizing the data, identifying variable features, and scaling. In a more recent update to Seurat's workflow, the SCTransform function [71, 72] was introduced, which combinatorially conducts the three steps in a manner optimized for downstream analyses. SCTransform is used in the PE tutorial while the BB tutorial follows a similar workflow to the one originally published by Seurat. Both users subsequently cover dimensionality reduction via PCA, deciding on the number of PCs to use, finding neighbors, identifying clusters, and using UMAP before guided visualization and exploration of the data.

Inferring single-cell trajectories with Scanpy

Trajectory inferences (TIs), or pseudotime analysis, provide an alternative means of grouping cells based on gradients of expression. It is worth noting that not all TI algorithms are fit for all datasets—these tutorials begin to explore the reasons why and guide users through the decision-making process. These parallel tutorials conduct the typical TI pipeline using Galaxy buttons or in

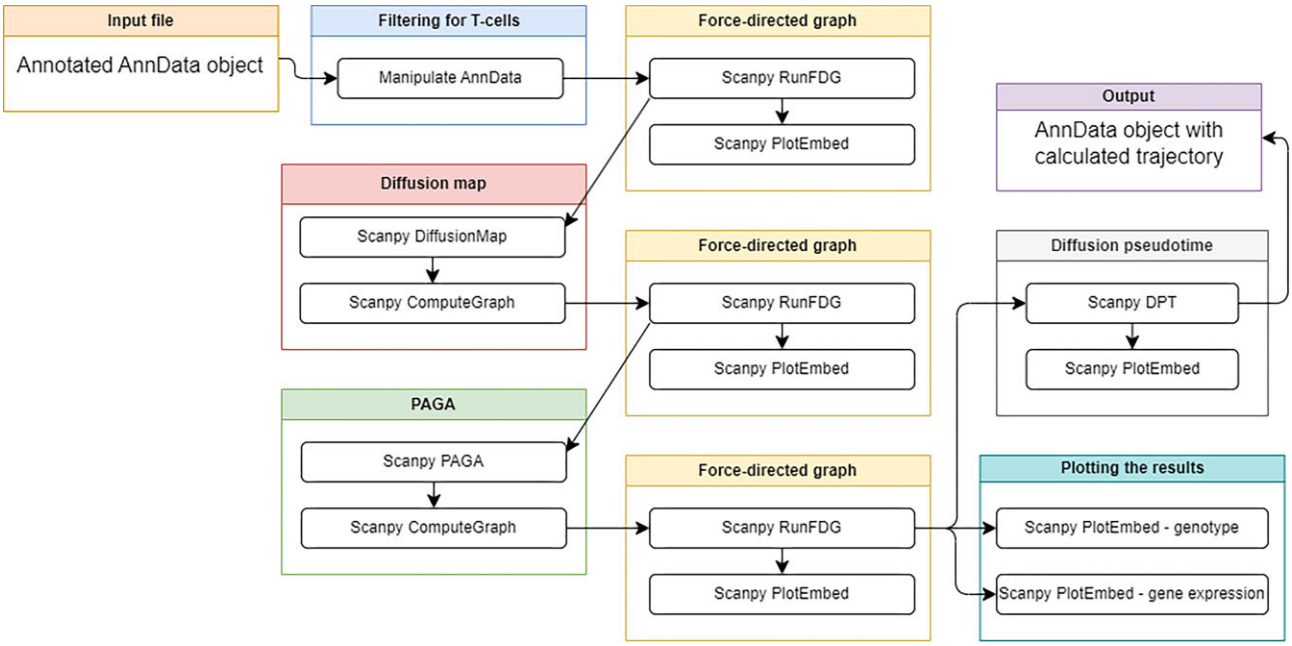

Figure 7: Workflow of inferring trajectories with Scanpy tutorial. Features methods such as force-directed graphs, diffusion maps, and PAGA used to infer the cells' trajectory in pseudotime. A figure of the extracted Galaxy workflow is available in Supplementary Fig. S5.

a Python coded environment to characterize transitions between cell states using Scanpy.

Tutorials are significantly based on Scanpy documentation, beginning with import of an annotated AnnData object into Galaxy. Users filter the data to retain a single cell type. The PE tutorial additionally demonstrates installation of modules before transferring their h5ad data to their Jupyter Notebook with the Galaxy–Jupyter cross-talk feature.

Users calculate force directed graphs (FDGs), which represent the data more appropriately for TI than the previously generated tSNE or UMAP visualizations [73]. Optionally, they may create diffusion maps, which can be used in place of PCs to recompute the nearest neighbors visualized in the FDGs.

Both BB and PE users order cells in pseudotime using Scanpy's diffusion approach, which accepts root cluster assignment indicating to the algorithm which population of cells the trajectory begins with. Users visualize inferred trajectories colored by pseudotime, as well as save and export their data, plots, and notebook. Users are encouraged to consider other changes across the identified trajectories beyond the scope of the tutorial. Tutorial workflows are shown in Fig. 7 and Supplementary Fig. S5.

Inferring single-cell trajectories with Monocle3

Similarly to the aforementioned, the Monocle3 tutorials teach users to conduct TI (Fig. 8 and Supplementary Fig. S6). These tutorials demonstrate the variability that may arise when trajectories are inferred by different algorithms—this time using the algorithms employed by Monocle3. PE users may implement RStudio or Jupyter Notebook through Galaxy's Interactive Environments. In collaboration with the Scanpy TI tutorial, users accomplish another TI method to additionally validate their results.

PE users are shown the installation of necessary libraries and modules, and they import a filtered AnnData object and familiarize themselves with the data's structure. They extract the expression matrix, cell, and gene metadata and prepare them for generation of a Cell Data Set (CDS) object—Monocle's preferred data type—with format and column name changes, as well as transposition. BB users may import a CDS file ready for downstream analysis in Monocle or the precursor files to create a CDS manually.

PE users utilize the BioMart database to retrieve gene symbols and associated gene IDs. Although not necessary to complete the tutorial's workflow, this ability is of use to users analyzing their own data.

Users preprocess with Monocle3, beginning with dimensionality reduction. PCA is the method used in these tutorials, although latent semantic indexing (LSI), UMAP, and tSNE options are also available. PE users visualize each PC in relation to gene variance: to identify how many PCs are needed to capture appropriate variability. Users are provided with visualizations of the output data given different choices in PC.

BB users plot the data in a PCA space, visualizing the effects of various experimental design variables. PE users may optionally correct for batch effects and enjoy customizable plots for a more tailored analysis prior to final dimensionality reduction.

Users cluster the data using Monocle3 as the tutorial describes the differences between clusters and partitions. The PE tutorial additionally demonstrates manual partitioning of cells: an important step for reliable trajectory inference.

The PE tutorial demonstrates three combined means of assigning cell types to the clusters—a supervised, an unsupervised, and an automated method. Users next infer trajectories relying on Monocle's trajectory graph. Once cells have been ordered in pseudotime, starting from the user-directed root cell, cells are visualized colored by pseudotime. BB users end here, comparing the results of the Monocle3-derived trajectory with the Scanpy algorithms.

PE users are presented with more options for differential expression analysis, visualizing results, identifying the visualization method best suited for them, and exporting plots, data, and their Python or RStudio notebook.

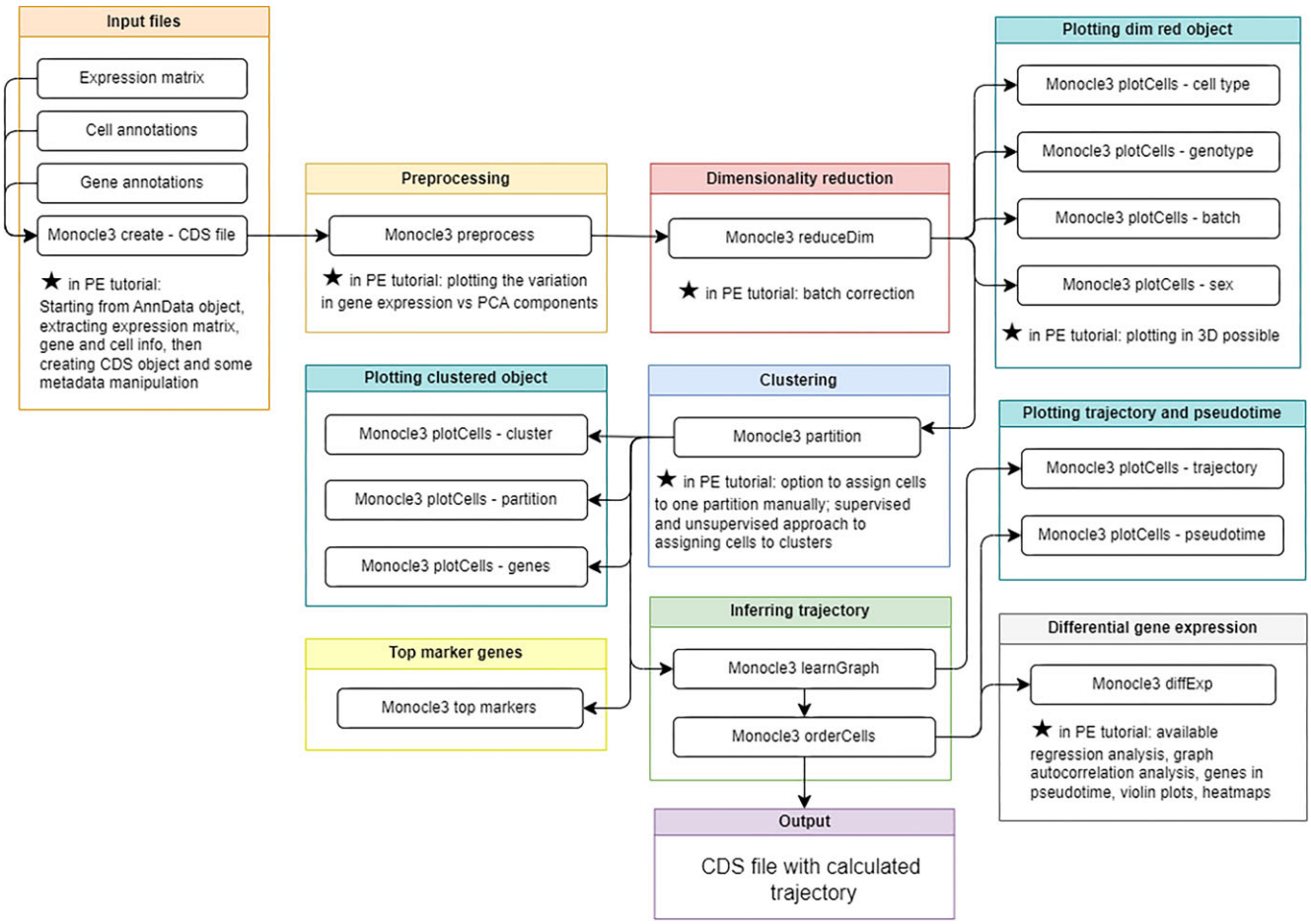

Figure 8: Workflow of Monocle3 inferring trajectories tutorial. Features data type changes for package compatibility, Monocle-specific preprocessing, and trajectory inference on a CDS object, followed by differential gene expression. A figure of an extracted Galaxy workflow is available in Supplementary Fig. S6.

Discussion

We present MIGHTS, a Multi-Interface Galaxy Hands-on Training Suite, where users may embark on three possible learning trajectories: (i) first learning to analyze scRNA-seq data with buttons in a GUI and subsequently performing the same, more flexible analysis in a programming environment; (ii) learning to run the code behind commonly published scRNA-seq analyses; or (iii) supplementing their preexisting analyses and skills with Galaxy tools.

MIGHTS performs analysis from raw reads, guiding users through filtering, normalization, dimensionality reduction, data quality assessment, and biological interpretations. The suite demonstrates filtering, clustering, annotation, and trajectory inference for a well-rounded scRNA-seq skill set. Each analysis is demonstrated using methods based on different packages, libraries, and programming languages with the hope that MIGHTS will prepare users to conduct their own, more complex, analyses.

Training features

Users of MIGHTS may start at any step by importing preprocessed input files, using output files from the preceding tutorial or their own data. Regardless, the analyses will be replicated across languages, methods, and starting points (Fig. 9), allowing users to follow the trajectory best suited for their skill level and analysis goals. To facilitate choosing the correct starting point depending on experience and goals, single-cell-oriented Learning Pathways were introduced. "Applying single-cell RNA-seq analysis" [74] and "Applying single-cell RNA-seq analysis in Coding Environments" [75] pathways are based on BB and PE tutorials, respectively, and may be used to facilitate a smooth transition between button-based tutorials and a programming environment (Fig. 2A) or a direct start in the programming environment (Fig. 2B). Additionally, to allow for easy identification of the tutorials described here, each tutorial has been tagged and can be found by entering "MIGHTS" in the GTN search box to get access to the relevant materials.

Each tutorial builds on the preceding, with no behind-the-scenes data formatting or annotation required between tutorials. With visual examples and analysis of various data types, live training courses found that trainees who performed tutorials during the day could successfully apply the analyses to their own data in the evenings [27, 28].

Learning how to set parameters has long been a difficulty in bioinformatics training [76]. By highlighting parameters that are adjusted often, users learn to prioritize what would otherwise be a never-ending list of decision-making. These "Decision-Time" features enable training for individuals and groups, with the option to vary parameter values and compare results (Fig. 10).

Testing of this feature has shown that, broadly, results remain the same regardless of parameter choice, demonstrating the relevance of robust, iterative analyses and data validation [27, 28].

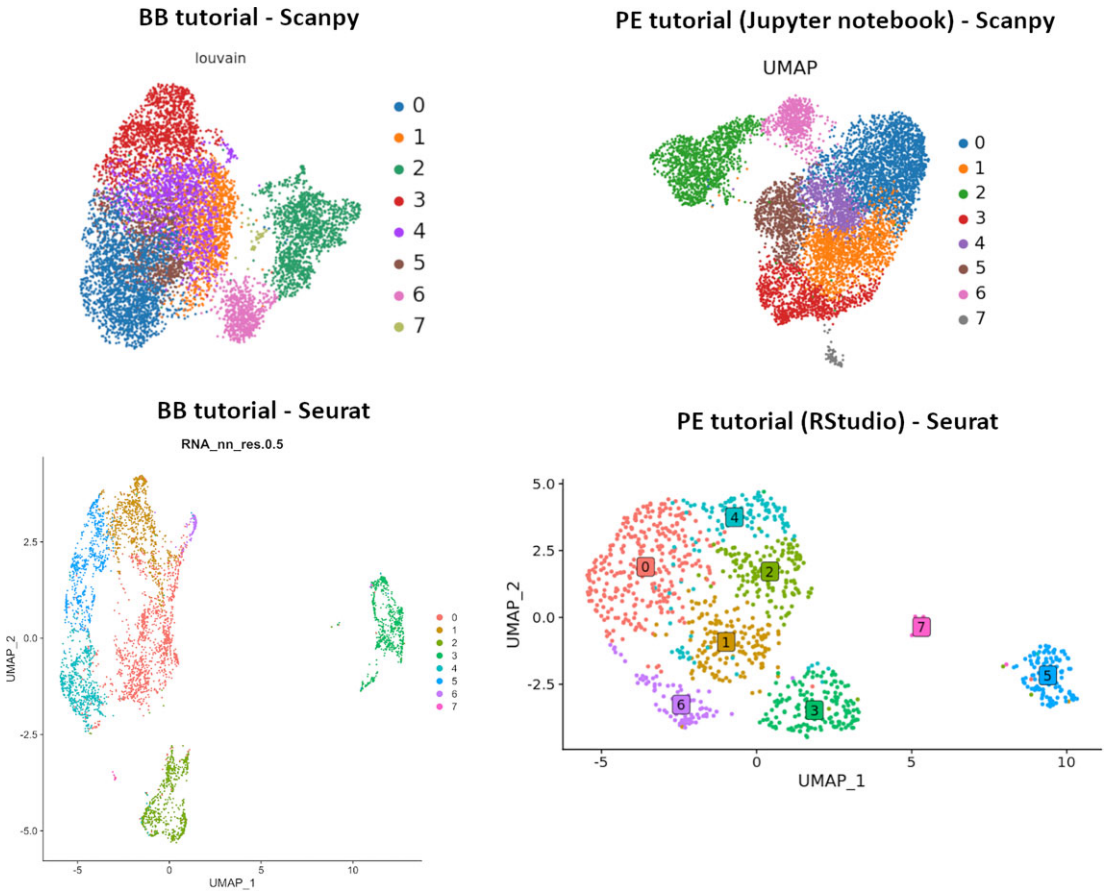

Figure 9: The final "cluster plot" as an output of plotting and interpretation tutorials across 4 paths: BB tutorial with Scanpy, PE (Jupyter notebook) with Scanpy, BB tutorial with Seurat, and PE (RStudio) tutorial with Seurat. The numbers correspond to the identified clusters in the dataset. No matter which method (BB vs. PE) or language (Scanpy vs. Seurat) is used, the biological interpretation is consistent in identifying 7 clusters.

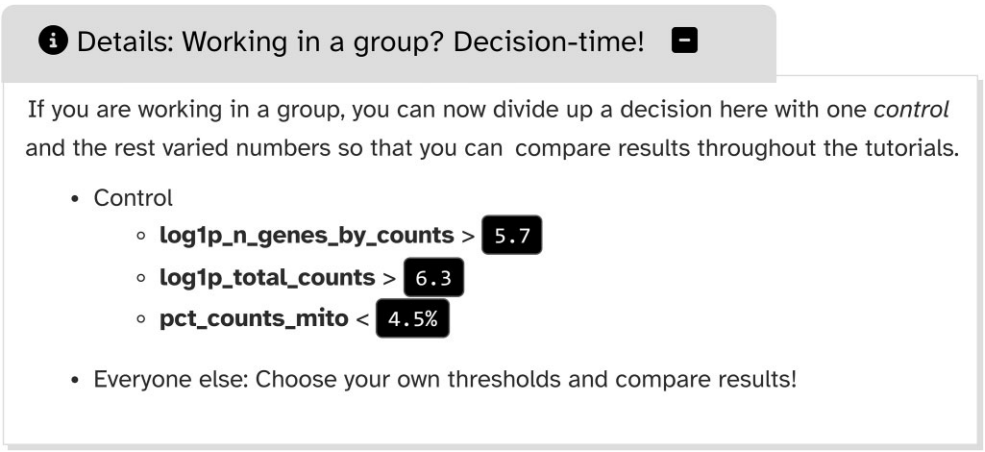

Figure 10: Exemplary "Decision-Time" feature box in tutorial "Filter, plot and explore single-cell RNA-seq data (Scanpy)."

To facilitate effective comprehension and a self-led learning environment, tutorials are interspersed with question boxes and collapsible solutions, allowing users to solidify their understanding of the material while they learn.

MIGHTS additionally pilots multiple import strategies, ensuring reliability for live training events. This includes direct import from Zenodo [77], import tools linked to data atlases [66], and import from "input" and "answer-key" Galaxy histories—which led to the development of a new feature within the GTN to signpost the option as supporting material (Fig. 11).

"Answer-key" histories follow datasets along every step of analyses, providing a final contingency for delivering live training and protecting users from frustration. Tutorials are additionally accompanied by slide decks (which can act as a general introduction to the topic), as well as recordings of the step-by-step analysis performed by an instructor.

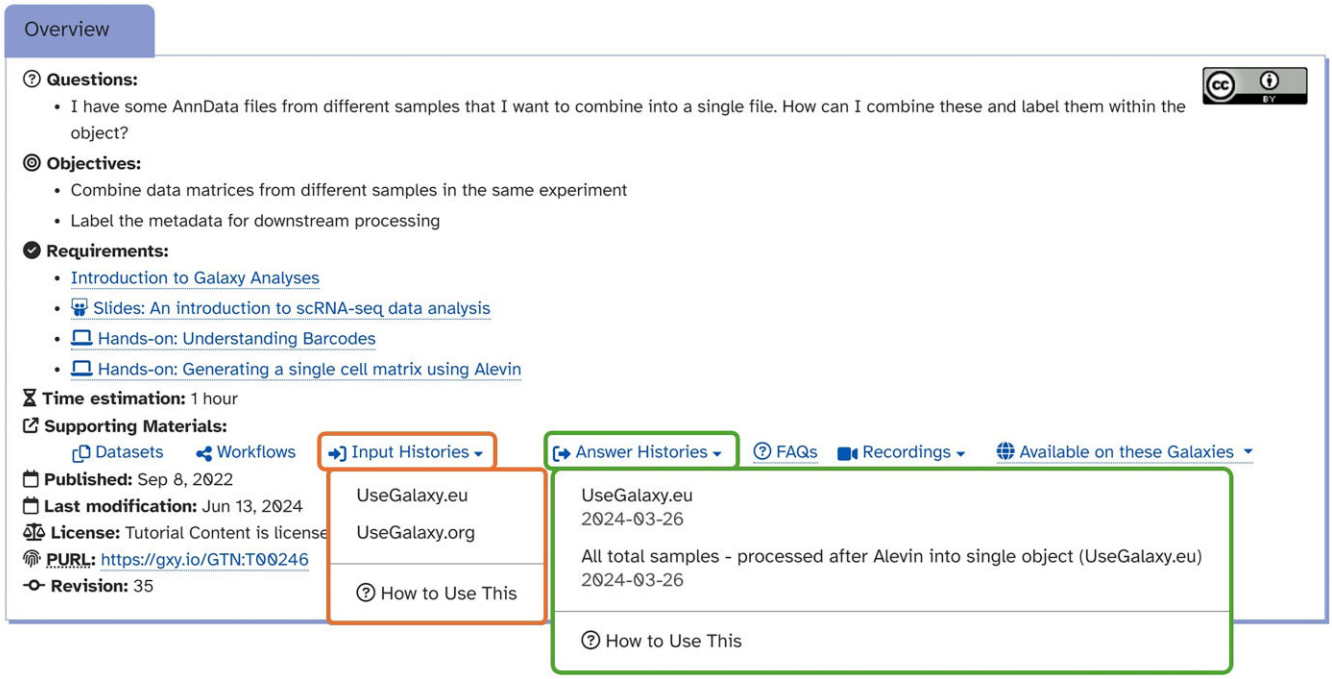

Figure 11: An overview box found at the beginning of the BB tutorial "Combining single cell datasets after pre-processing." It showcases a header feature, which allows for a quick access to the input histories (orange frame) and answer histories (green frame).

Learn to code in a beginner-friendly way

As sequencing strategies and tools continue to advance, it is important that the field of bioinformatics "trains the trainer" in response to continued growth. To support comprehension, each tutorial provides detailed explanations of biological and computational concepts, including simplified troubleshooting and multiple interactive elements. By showing alternative methods to perform a single analysis, users become familiar with the most common programming languages used in the life sciences: Python and R, as well as command language Bash. This provides users with well-rounded examples of how to analyze scRNA-seq data and how they may begin to leverage analyses (and Galaxy) as a means to learn new programming skills [78]. These PE tutorials introduce users to relevant packages, functions, and data types used in today's published bioinformatic analyses (Table 1).

The transition from Galaxy-button tutorials into the coded environment is facilitated by interactive tools such as RStudio or Jupyter Notebook, such that all the analysis may be completed within Galaxy as opposed to on local computers. Importantly, there is no need for any software installation—all tutorials provide necessary tools to complete them, including example datasets, slides, videos, workflows, and public Galaxy servers where the analysis may be performed. Internet access is the only additional necessary resource [79]. This approach specifically facilitates accessible bioinformatics analyses by eliminating installation hang-ups, minimizing the time spent setting one's environment, and increasing computing capacity for users.

Additionally, if users embrace programming such that they are looking to program their own button-based tools or create new training material, opportunities to do so exist on GTN pages dedicated to development in Galaxy [80] and contributing to the Galaxy Training Material [81].

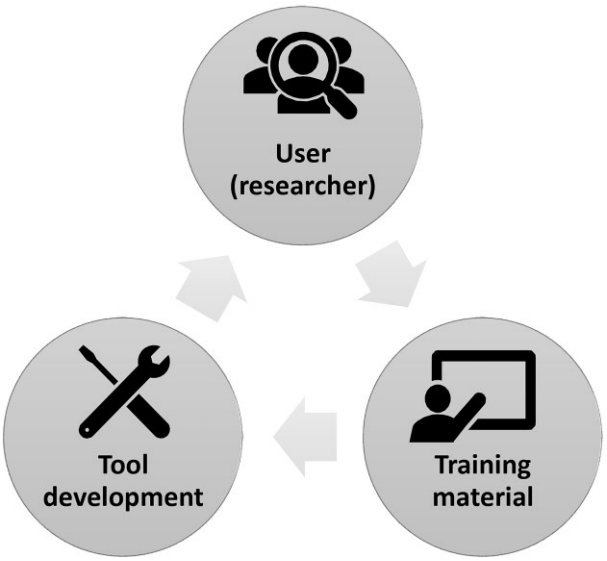

Figure 12: Galaxy Circle of Life demonstrating the interdisciplinary, multilevel sustainability practiced by the GTN. Users report changes they wish to see made in the training material, prompting new tool development and updates that can be sustainably utilized and tested by researchers.

FAIR data

MIGHTS tutorials were created on an interface with embedded findable, accessible, interoperable, and reusable (FAIR) data usage [80]. The FAIR principles can and should be applied in all life science domains where large amounts of data are produced. FAIR data management is particularly important in scRNA-seq analysis, which looks at large expression matrices. Unfortunately, it is often the case that published datasets come with missing, or

Table 2: Number of revisions made to each tutorial featured in MIGHTS as of August 2024

Tutorial topic	BB tutorials		PE tutorials	
	Months since tutorial publication	Number of revisions	Months since tutorial publication	Number of revisions
Generating a single-cell matrix using Alevin	41	16	8	2
Combining single-cell datasets after preprocessing	23	16		
Filter plot explore scRNA-seq data with Scanpy	40	18	11	9
Filter plot and explore scRNA-seq data with Seurat	4	3	10	8
Inferring single-cell trajectories with Scanpy	8	5	40	15
Inferring single-cell trajectories with Monocle3	22	19	15	10

incomplete, metadata—rendering the dataset less useful than it would be with complete annotation(s). By completing MIGHTS tutorials, users become equipped with the skills helpful in formatting such demanding datasets.

Sustainable

An important feature characterizing MIGHTS is its sustainability. As previously reported, the evolving nature of bioinformatics requires a sustainable bridge between the fields of biology and informatics [81]. Therefore, collaboration between developers and domain experts is critical. The GTN emphasizes that users be included in this collaboration, whereby users have the opportunity to report issues and request additional resources. This facilitates involvement of developers who are aware of user needs and users who are actively contributing to the improvement of materials. Any issues may be reported back to tutorial developers themselves, demonstrating the sustainability of Galaxy's Circle of Life (Fig. 12).

Tutorials on the GTN are, at minimum, updated annually in preparation for the Galaxy Community Conference (GCC). The Galaxy Circle of Life functions such that tutorials will continue to meet evolving user needs—largely thanks to the commitment of the growing Galaxy Single-Cell & Spatial Omics Community. Notably, MIGHTS tutorials have been updated, on average, 7 times a year since their respective publication (Table 2).

The number of revisions demonstrates continued sustainability of tutorials featured in the suite.

Notably, the GTN offers tutorials on additional bioinformatic analyses in a variety of fields. These tutorials are similarly monitored and revised, although the rate of growth specifically for single-cell tutorials is worth noting.

Addressing modern challenges in bioinformatics

MIGHTS addresses many broad challenges of bioinformatics training, emphasizing that effective bioinformatics involves understanding key principles and gaining experience [22] with real-world data, problem-solving [24], reproducibility [82], and validation [83].

One challenge in bioinformatics is the application of analyses from training courses to real, messy, lab-generated data. Uniquely, MIGHTS uses raw, unannotated data from a published analysis—Bacon et al. [31]—and guides users through reformatting, annotating, and conducting biological analyses.

Reproducibility and reinstantiation, keystones of quality bioinformatic analyses, are ensured by MIGHTS thanks to published workflows and answer key histories for each dataset (Fig. 11). Workflows are available on the Galaxy servers, providing a stable way to perform a particular analysis in an identical environment. By linking detailed tool versions to the tutorial workflows themselves, it is possible to submit new input files, adjust parameter thresholds, and wait for an output. This is particularly helpful for analyzing multiple samples that require the same pipeline, allowing reproducible results, minimizing time spent rerunning code, and eliminating the need for complex coding skills to develop pipelines.

To address another challenge of the field, MIGHTS emphasizes the importance of validating one's results: to determine whether results reflect an actual biological process versus artifacts of the pipeline. Using tools based in various programming languages and multiple algorithms allows users to feel confident that their results are uncovering true biological insights no matter the analysis method used (Fig. 9). MIGHTS can act as a guide on how to validate results. The suite may additionally be used directly by novice to intermediate bioinformaticians to check whether their results are consistent via alternative methods without needing to learn another programming language.

Because bioinformatics combines numerous STEM fields, it faces interdisciplinary and intergenerational challenges [84]. Software developers often do not understand the underlying biology their programs analyze, and biologists often do not know how analytical algorithms function [85]. MIGHTS aims to fill this gap by introducing step-by-step analyses, while simultaneously demonstrating the biological interpretation of results and how they were uncovered. Coded tutorials provide additional opportunities to become familiar with the algorithms behind the analyses. The suite can act as a resource to educate and inspire future generations of bioinformaticians: ones who are able to speak across disciplines, effectively identify areas for improvement, and build flexible, long-term solutions. Tutorials are available by link on Galaxy [86–96]

Limitations and further steps

The main limitation of the Galaxy GUI tutorials is that the analyses are limited to the packages and functions that have been wrapped into tools. As such, some analysis steps might be limited in the BB tutorials. However, users have the opportunity to submit "tool requests": an ongoing effort to mitigate this limitation.

Additionally, tool versions must be compatible with one another. To mitigate this limitation, tools are regularly tested and updated to allow for analysis using the most recent versions and ensuring outputs are compatible inputs for downstream steps. Issues with tools may be reported on Galaxy forums, where experts and developers respond quickly to issues.

The main limitations of the PE tutorials are limited resources allocated to Interactive Environments and inconsistencies between the notebooks on different public Galaxy servers (.eu vs. .org vs. .au). However, the educational purpose of the coded-tutorials is to familiarize users with coding environments, so

downsampled data provide the same benefits and enable most analyses to be done within the resource limit. Even so, should a user need or want more resources allocated, they can request that from the Galaxy admins.

There are ongoing efforts to expand the functionality of MIGHTS to enable more bespoke analyses of datasets, in response to community needs.

Availability of Source Code and Requirements

- Project name: Multi-Interface Galaxy Hands-on Training Suite for scRNA-seq
- Project homepage: https://github.com/galaxyproject/training-material/tree/main/topics/single-cell/tutorials
- Operating system(s): web-based, platform independent
- Programming languages: R, Python, Bash
- License: MIT

Additional Files

Supplementary Fig. S1. Galaxy workflow for tutorial "Generating a single-cell matrix using Alevin." The tools used for the analysis are shown, together with their outputs and connectivities, as well as high-level descriptors of performed steps. Solid stars denote steps specific to the PE tutorial while unfilled stars represent BB-specific ones.

Supplementary Fig. S2. Galaxy workflow for tutorial "Combining single-cell datasets after pre-processing." All tools used for the analysis are shown, as well as their connectivities and high-level descriptors of performed steps. Solid star denotes steps specific to the PE tutorial.

Supplementary Fig. S3. Galaxy workflow for tutorial "Filter, plot and explore single-cell RNA-seq data (Scanpy)." All tools used for the analysis are shown, as well as their connectivities and high-level descriptors of performed steps.

Supplementary Fig. S4. Galaxy workflow for tutorial "Filter, plot and explore single-cell RNA-seq data (Seurat)." All tools used for the analysis are shown, as well as their connectivities and high-level descriptors of performed steps.

Supplementary Fig. S5. Galaxy workflow for tutorial "Inferring single-cell trajectories (Scanpy)." All tools used for the analysis are shown, as well as their connectivities and high-level descriptors of performed steps.

Supplementary Fig. S6. Galaxy workflow for tutorial "Inferring single-cell trajectories (Monocle3)." All tools used for the analysis are shown, as well as their connectivities and high-level descriptors of performed steps.

Abbreviations

BB: button based; CDS: cell data set; DPT: diffusion pseudotime; EMBL-EBI: European Molecular Biology Laboratory–European Bioinformatics Institute; FAIR: findable, accessible, interoperable, and reusable; FDG: force-directed graph; GCC: Galaxy Community Conference; GTF: gene transfer format; GTN: Galaxy Training Network; GUI: graphical user interface; LSI: latent semantic indexing; MIGHTS: Multi-Interface Galaxy Hands-on Training Suite for scRNA-seq; PAGA: partition-based graph abstraction; PC: principal component; PCA: principal component analysis; PE: programming environment; QC: quality control; SCE: single-cell experiment; scRNA-seq: single-cell RNA sequencing; SE: summarized experiment; STEM: Science, Technology, Engineering, and Mathematics; TI: trajectory inference; tSNE: t-distributed stochastic neighbor embedding; UMAP: Uniform Manifold Approximation and Projection.

Acknowledgments

The authors extend their gratitude to the Galaxy community for supporting the testing of workflows as well as the development of tools. They thank Gareth Price for support testing on the Australian Galaxy instance, and training course participants for testing tutorials in live user settings.

Author Contributions

Camila Goclowski (Formal analysis [equal], Investigation, Methodology [equal], Validation [equal], Visualization [equal], Writing—original draft [equal], Writing—review & editing [equal]), Julia Jakiela (Formal analysis [equal], Investigation [equal], Methodology [equal], Validation [equal], Visualization [equal], Writing—original draft [equal], Writing—review & editing [equal]), Tyler Collins (Resources [equal], Software [equal]), Saskia Hiltemann (Resources [equal], Software [equal]), Morgan Howells (Formal analysis [equal], Investigation [equal], Methodology [equal], Validation [equal], Visualization [equal]), Marisa Loach (Investigation [equal], Methodology [equal], Validation [equal], Visualization [equal], Writing—review & editing [equal]), Jonathan Manning (Formal analysis [equal], Investigation [equal], Methodology [equal], Validation [equal], Visualization [equal]), Pablo Moreno (Resources [equal], Software [equal]), Alex Ostrovsky (Resources [equal], Software [equal]), Helena Rasche (Data curation [equal], Resources [equal], Software [equal]), Mehmet Tekman (Resources [equal], Software [equal], Validation [equal]), Graeme Tyson (Validation [equal]), and Pavankumar Videm (Resources [equal], Software [equal], Validation [equal], Writing—review & editing [equal]).

Wendi Bacon (Conceptualization, Data curation, Formal Analysis, Investigation, Funding Acquisition, Methodology, Project Administration, Resources, Supervision, Validation, Visualization, Writing-review & editing

Funding

Internships were funded in part by the Engineering & Physical Sciences Research Council (UK) as well as Hobart and William Smith Colleges (Geneva, NY, USA). Additionally, part of the materials was created thanks to the Third Training Open Call issued by EOSC-Life, which has received funding from the European Union's Horizon 2020 program under grant agreement number 824087.

Data Availability

All the tutorials are available at the dedicated Single Cell subpage of the Galaxy Training Network (GTN) [97].

The used experimental data come from a published study by Bacon et al. [31], which is publicly available from the EMBL-EBI ArrayExpress under accession number E-MTAB-6945 and can also be browsed from the Single Cell Expression Atlas [66]. The input datasets used in tutorials are stored at Zenodo [98–103], and all generated data files are available in the shared Galaxy histories, included in each tutorial.

The tutorials comprise many different tools that can be freely used at the Galaxy public servers, such as Galaxy Europe [104],

Galaxy US [105], Galaxy Australia [106], and others. The tool wrappers with detailed information are stored at the Galaxy ToolShed [107].

Competing Interests

The author(s) declare that they have no competing interests.

References

1. Attwood TK, Schneider MV, Brazas MD. A global perspective on evolving bioinformatics and data training needs. Briefing Bioinformatics 2017;20:398–404. https://doi.org/10.1093/bib/bbx100..
2. Goodman N. Biological data becomes computer literate: new advances in bioinformatics. Curr Opin Biotechnol 2002;13:68–71. https://doi.org/10.1016/S0958-1669(02)00287-2.
3. Mitra D, Bensaad MS, Sinha S, et al. Evolution of bioinformatics and its impact on modern bio-science in the twenty-first century: special attention to pharmacology, plant science and drug discovery. Computational Toxicol 2022;24:100248. https://doi.org/10.1016/j.comtox.2022.100248.
4. Singh S, Pandey AK, Prajapati VK. Chapter one—from genome to clinic: the power of translational bioinformatics in improving human health. Adv Protein Chem Structural Biol 2024;139:1–25. https://doi.org/10.1016/bs.apcsb.2023.11.010.
5. Dhiman K, Dhiman H. Unveiling the world of bioinformatics. In: Kumar Ued. Applying machine learning techniques to bioinformatics: few-shot and zero-shot methods. Hershey, Pennsylvania: IGI Global; 2024;181–200. https://doi.org/10.4018/979-8-3693-1822-5.ch010.
6. Wright AM, Schwartz JR, Newman CE. The why, when, and how of computing in biology classrooms. F1000Res 2019;8:1854. https://doi.org/10.12688/f1000research.20873.2.
7. Dabholkar S. Computational thinking in biology: part 1. In: Priami C.ed. Bridging interdisciplinary gaps in education sciences. Berlin, Heidelberg: Springer; 2021. https://doi.org/10.1007/978-3-540-76639-1_4.
8. Ras V, Carvajal-Lopez P, Gopalasingam P, et al. Challenges and considerations for delivering bioinformatics training in LMICs: perspectives from Pan-African and Latin American bioinformatics networks. Frontiers in Education 2021;6. https://doi.org/10.3389/feduc.2021.710971.
9. Chasapi A, Promponas VJ, Ouzounis CA. The bioinformatics wealth of nations. Bioinformatics 2020;36:2963–65. https://doi.org/10.1093/bioinformatics/btaa132.
10. Erxleben-Eggenhofer A, Batut B, Müller T. FAIR and scalable education the galaxy training network (GTN) and a training infrastructure as a service (TIaaS). Proceedings of the Conference on Research Data Infrastructure 2023;1. https://doi.org/10.52825/cordi.v1i.422.
11. Forero D, Wonkam A, Wang W, et al. Current needs for human and medical genomics research infrastructure in low and middle income countries. J Med Genet 2016;53:438–40. https://doi.org/10.1136/jmedgenet-2015-103631.
12. Mendy M, Caboux E, Sylla BS, et al. Infrastructure and facilities for human biobanking in low- and middle-income countries: a situation analysis. Pathobiology 2014;81:252–60. https://doi.org/10.1159/000362093.
13. Pérez-Wohlfeil E, Torreno O, Bellis L, et al. Training bioinformaticians in high performance computing. Heliyon 2018;4:e01057. https://doi.org/10.1016/j.heliyon.2018.e01057.
14. Wilson SMA, Hauser C, Sierk M, et al. Bioinformatics core competencies for undergraduate life sciences education. PLoS One 2018;13:e0196878. https://doi.org/10.1371/journal.pone.0196878.
15. Williams JJ, Drew JC, Galindo-Gonzalez S, et al. Barriers to integration of bioinformatics into undergraduate life sciences education: a national study of US life sciences faculty uncover significant barriers to integrating bioinformatics into undergraduate instruction. PLoS One 2019;14:e0224288. https://doi.org/10.1371/journal.pone.0224288.
16. Katara P. Role of bioinformatics and pharmacogenomics in drug discovery and development process. Network Model Anal Health Inform Bioinform 2013;2:225–230. https://doi.org/10.1007/s13721-013-0039-5.
17. Levine AG. An explosion of bioinformatics careers. Science 2014;344:1303–6.https://doi.org/10.1126/science.344.6189.1303.
18. Hwang B, Lee JH, Bang D. Single-cell RNA sequencing technologies and bioinformatics pipelines. Experimental & Molecular Medicine 2018;50:1–14. https://doi.org/10.1038/s12276-018-0071-8.
19. Navlakha S, Bar-Joseph Z. Algorithms in nature: the convergence of systems biology and computational thinking. Molecular Systems Biology 2011;7:546https://doi.org/10.1038/msb.2011.78.
20. Carey MA, Papin J. Ten simple rules for biologists learning to program. PLoS Comput Biol 2018;14:e1005871. https://doi.org/10.1371/journal.pcbi.1005871.
21. Via A, Blicher T, Bongcam-Rudloff E, et al. Best practices in bioinformatics training for life scientists. Briefings Bioinf 2013;14:528–37. https://doi.org/10.1093/bib/bbt043.
22. Dudley J, Butte AJ. A quick guide for developing effective bioinformatics programming skills. PLoS Comput Biol 2009;5:e1000589. https://doi.org/10.1371/journal.pcbi.1000589.
23. Jazayeri M. Combining mastery learning with project-based learning in a first programming course: an experience report. In: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering. Florence, Italy: Institute of Electrical and Electronics Enigneers; 2015. https://doi.org/10.1109/ICSE.2015.163.
24. Perkins DN, Hancock C, Hobbs R, et al. Conditions of learning in novice programmers. J Educ Comput Res 1986;2:37–55. https://doi.org/10.2190/GUJT-JCBJ-Q6QU-Q9PL.
25. Shafto SA. Programming for learning in mathematics and science. SIGCSE Bull 1986;18:296–302. https://doi.org/10.1145/953055.5635.
26. Johnston IG, Slater M, Cazier JB. Interdisciplinary and transferable concepts in bioinformatics education: observations and approaches from a UK MSc course. Frontiers in Education 2022;7. https://doi.org/10.3389/feduc.2022.826951.
27. Hiltemann S, Rasche H, Gladman S, et al. Galaxy Training: a powerful framework for teaching! PLoS Comput Biol 2023;19:e1010752. https://doi.org/10.1371/journal.pcbi.1010752.
28. Rasche H, Hyde C, Davis J, et al. Training infrastructure as a service. Gigascience 2022;12:giad048. https://doi.org/10.1093/gigascience/giad048.
29. Bacon W, Holinski A, Pujol M, et al. Ten simple rules for leveraging virtual interaction to build higher-level learning into bioinformatics short courses. PLoS Comput Biol 2022;18:e1010220. https://doi.org/10.1371/journal.pcbi.1010220.
30. Moreno P, Huang N, Manning JR, et al. User-friendly, scalable tools and workflows for single-cell RNA-seq analysis. Nature

Methods 2021;18:327–328. https://doi.org/10.1038/s41592-021-01102-w.
31. Bacon WA, Hamilton RS, Yu Z, et al. Single-cell analysis identifies thymic maturation delay in growth-restricted neonatal mice. Front Immunol 2018;9. https://doi.org/10.3389/fimmu.2018.02523.
32. Patro R, Duggal G, Love M, et al. Salmon provides fast and bias-aware quantification of transcript expression. Nat Methods 2017;14:417–419. https://doi.org/10.1038/nmeth.4197.
33. Srivastava A, Malik L, Smith T, et al. Alevin efficiently estimates accurate gene abundances from dscRNA-seq data. Genome Biol 2019;20. https://doi.org/10.1186/s13059-019-1670-y.
34. Lun ATL, Riesenfeld S, Andrews T, et al. EmptyDrops: distinguishing cells from empty droplets in droplet-based single-cell RNA sequencing data. Genome Biol 2019;20. https://doi.org/10.1186/s13059-019-1662-y.
35. Griffiths JA, Richard AC, Bach K, et al. Detection and removal of barcode swapping in single-cell RNA-seq data. Nat Commun 2018;9. https://doi.org/10.1038/s41467-018-05083-x.
36. R Core Team. R: a language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing; 2021. https://www.R-project.org/.
80. atlas-gene-annotation-manipulation. EBI Gene Expression Group. GitHub. https://github.com/ebi-gene-expression-group/atlas-gene-annotation-manipulation.Accessed 13 August 2024.
38. Love MI, Soneson C, Hickey PF, et al. Tximeta: reference sequence checksums for provenance identification in RNA-seq. PLoS Comput Biol 2020;16:e1007664. https://doi.org/10.1371/journal.pcbi.1007664.
39. Durinck S, Moreau Y, Kasprzyk A, et al. BioMart and Bioconductor: a powerful link between biological databases and microarray data analysis. Bioinformatics 2005;21:3439–40. https://doi.org/10.1093/bioinformatics/bti525.
40. Durinck S, Spellman P, Birney E, et al. Mapping identifiers for the integration of genomic datasets with the R/bioconductor package biomaRt. Nat Protoc 2009;4:1184–91. https://doi.org/10.1038/nprot.2009.97.
41. Wolf FA, Angerer P, Theis FJ. SCANPY: large-scale single-cell gene expression data analysis. Genome Biol 2018;19. https://doi.org/10.1186/s13059-017-1382-0.
42. Van Rossum G, Van Rossum D, Fred L.Python reference manual Centrum voor Wiskunde en Informatica Amsterdam. 1995.
43. Csárdi G, Nepusz T, Traag V, et al. igraph: network analysis and visualization in R. 2024. https://doi.org/10.32614/CRAN.package.igraph.Accessed 13 August 2024.
44. Blondel VD, Guillaume J, Lambiotte R, et al. Fast unfolding of communities in large networks. J Stat Mech 2008;2008:P10008. https://doi.org/10.1088/1742-5468/2008/10/P10008.
45. McKinney W, Data structures for statistical computing in python. In: der van Millman Jeds. Proceedings of the 9th Python in Science Conference. Vol. 445. Austin, TX 2010:56–61. http://conference.scipy.org.s3.amazonaws.com/proceedings/scipy2010/pdfs/mckinney.pdf.
46. Satija R, Farrell JA, Gennert D, et al. Spatial reconstruction of single-cell gene expression data. Nat Biotechnol 2015;33:495–502. https://doi.org/10.1038/nbt.3192.
47. Hao Y, Stuart T, Kowalski MH, et al. Dictionary learning for integrative, multimodal and scalable single-cell analysis. Nature Biotechnology 2023;42:293–304. https://doi.org/10.1038/s41587-023-01767-y
48. Hao Y, Hao S, Andersen-Nissen E, et al. Integrated analysis of multimodal single-cell data. Cell 2021;184:3573–3587. https://doi.org/10.1016/j.cell.2021.04.048.
49. Stuart T, Butler A, Hoffman P, et al. Comprehensive integration of single-cell data. Cell 2019;177:1888–1902. https://doi.org/10.1016/j.cell.2019.05.031.
50. Butler A, Hoffman P, Smibert P, et al. Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat Biotechnol 2018;36:411–420. https://doi.org/10.1038/nbt.4096.
51. Maechler M. Matrix: sparse and dense Matrix classes and methods. 2024. https://doi.org/10.32614/CRAN.package.Matrix.Accessed 13 August 2024.
52. Wickham H, François R, Henry L, et al. dplyr: a grammar of data manipulation. 2023. https://dplyr.tidyverse.org. Accessed 13 August 2024.
53. Razavi K, Luthra M, Koldehofe B, et al. FA2: fast, accurate autoscaling for serving deep learning inference with SLA guarantees. In: 2022 IEEE 28th Real-Time and Embedded Technology and Applications Symposium (RTAS). Milano, Italy: Institute of Electrical and Electronics Engineers (IEEE); 2022:146–59. https://doi.org/10.1109/RTAS54340.2022.00020.
54. Harris CR, Millman KJ, van der Walt SJ, et al. Array programming with NumPy. Nature 2020;585:357–62. https://doi.org/10.1038/s41586-020-2649-2.
55. Hunter JD. Matplotlib: a 2D graphics environment. Comput Sci Eng 2007;9(3):90–95.https://doi.org/10.1109/MCSE.2007.55.
56. Cao J, Spielmann M, Qiu X, et al. The single-cell transcriptional landscape of mammalian organogenesis. Nature 2019;566:496–502https://doi.org/10.1038/s41586-019-0969-x.
57. Virshup I, Rybakov S, Theis FJ, et al. anndata: access and store annotated data matrices. J Open Source Softw 2024;9(101):4371. https://doi.org/10.21105/joss.04371.
58. Garnier S, Ross N, Rudis R, et al. viridis(Lite)—colorblind-friendly color maps for R. viridisLite package version 0.4.2. 2023. https://sjmgarnier.github.io/viridis/, https://doi.org/10.5281/zenodo.4678327.Accessed 13 August 2024.
59. Bache S, Wickham H. magrittr: a forward-pipe operator for R. 2022. https://magrittr.tidyverse.org, https://github.com/tidyverse/magrittr. Accessed 13 August 2024.
60. Eddelbuettel D, Francois R, Allaire J, et al. Rcpp: seamless R and C++ integration. R package version 1.0.12. 2024. https://doi.org/10.32614/CRAN.package.Rcpp.Accessed 13 August 202.4
61. Gruning B, Rasche E, Rebolledo-Jaramillo B, et al. Jupyter and Galaxy: easing entry barriers into complex data analyses for biomedical researchers. PLoS Comput Biol 2017;13:e1005425. https://doi.org/10.1371/journal.pcbi.1005425.
62. Baumer B, Udwin D. R markdown. WIREs Computational Stats 2015;7:167–77. https://doi.org/10.1002/wics.1348.
63. Ragan-Kelley M, Perez F, Granger B, et al. The Juyter/iPython architecture: a unified view of computational research, from interactive exploration to communication and publication. San Fransisco, CA: American Geophysical Union; 2016. https://ui.adsabs.harvard.edu/abs/2014AGUFM.H44D..07R/abstract.
64. Scherer R, Siddiq F, Sánchez-Scherer B. Some evidence on the cognitive benefits of learning to code. Front Psychol 2021;12:559424. https://doi.org/10.3389/fpsyg.2021.559424.
65. Shute VJ, Sun C, Asbell-Clarke J. Demystifying computational thinking. Educ Res Rev 2017;22:142–158. https://doi.org/10.1016/j.edurev.2017.09.003.

66. Papatheodorou I, Moreno P, Manning J, et al. Expression Atlas update: from tissues to single cells. Nucleic Acids Res 2020; 48:D77–D83. https://doi.org/10.1093/nar/gkz947.
67. He J, Lihui L, Chen J. Practical bioinformatics pipeline for single-cell RNA-seq data analysis. Biophys Rep 2022;8:158–69. https://doi.org/10.52601/bpr.2022.210041.
68. van der Maaten L, Hinton G. Visualizing data using t-SNE. J Mach Learn Res 2008;9(86):2579–605. https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf?fbcl.
69. McInnes L, Healy J, Saul N, et al. UMAP: Uniform Manifold Approximation and Projection. J Open Source Softw 2018;3:861. https://doi.org/10.21105/joss.00861.
70. Megill C, Martin B, Weaver C, et al. cellxgene: a performant, scalable exploration platform for high dimensional sparse matrices. Biorxiv. 2021. Accessed August 2024. https://doi.org/10.1101/2021.04.05.438318.
71. Hafemeister C, Satija R. Normalization and variance stabilization of single-cell RNA-seq data using regularized negative binomial regression. Genome Biol 2019;20https://doi.org/10.1186/s13059-019-1874-1.
72. Choudhary S, Satija R. Comparison and evaluation of statistical error models for scRNA-seq. Genome Biol 2022;23. https://doi.org/10.1186/s13059-021-02584-9.
73. Ko ME, Williams CM, Fread KI, et al. FLOW-MAP: a graph based, force directed layout algorithm for trajectory mapping in single-cell time course datasets. Nat Protoc 2020;15:398–420. https://doi.org/10.1038/s41596-019-0246-3.
74. Galaxy Training. Applying single-cell RNA-seq analysis. https://gxy.io/GTN:P00020. Accessed 13 August 2024.
75. Galaxy Training. Applying single-cell RNA-seq analysis in coding environments. https://gxy.io/GTN:P00024.Accessed 13 August 2024.
76. Tractenberg RE, Lindvall JM, Attwood TK, et al. The Mastery Rubric for bioinformatics: a tool to support design and evaluation of career-spanning education and training. PLoS One 2019;14:e0225256. https://doi.org/10.1371/journal.pone.0225256.
77. Wareham J, Pujol Priego L, Zenodo—Open science monitor case study. European Commission, Directorate-General for Research and Innovation. 2019. https://data.europa.eu/doi/10.2777/298228.Accessed 13 August 2024.
78. Goclowski C. From GTN Intern to tutorial author to bioinformatician. Galaxy Training. 2024. https://gxy.io/GTN:N00087 .Accessed 13 August 2024.
79. Goecks J, Nekrutenko A, Taylor J. Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 2010;11. https://doi.org/10.1186/gb-2010-11-8-r86.
80. Ouwerkerk J, Rasche H, Spalding JD, et al. FAIR data retrieval for sensitive clinical research data in Galaxy. Gigascience 2024;13:giad099. https://doi.org/10.1093/gigascience/giad099.
81. Aron S, Chauke PA, Ras V, et al. The development of a sustainable bioinformatics training environment within the H3Africa Bioinformatics Network. Front Educ 2021;6. https://doi.org/10.3389/feduc.2021.725702.
82. Cokelaer T, Cohen-Boulakia S, Lemoine F. Reprohackathons: promoting reproducibility in bioinformatics through training. Bioinformatics 2023;39:i11–i20. https://doi.org/10.1093/bioinformatics/btad227.
83. Yang A, Troup M, Ho JWK. Scalability and validation of big data bioinformatics software. Comput Struct Biotechnol J 2017;15:379–386. https://doi.org/10.1016/j.csbj.2017.07.002.
84. Bartlett A, Lewis J, Williams M. Generations of interdisciplinarity in bioinformatics. New Genet Soc 2016;25:186–209. https://doi.org/10.1080/14636778.2016.1184965.
85. Garmire DG, Xun Z, Aravind M, et al. GranatumX: a community-engaging, modularized, and flexible webtool for single-cell data analysis. Genomics Proteomics Bioinformatics 2021;19:452–460. https://doi.org/10.1016/j.gpb.2021.07.005.
86. Bacon W, Manning J. Generating a single cell matrix using Alevin (Galaxy Training Materials). https://gxy.io/GTN:T00245 .Accessed 13 August 2024.
87. Bacon W, Manning J. Combining single cell datasets after preprocessing (Galaxy Training Materials). https://gxy.io/GTN:T00246. Accessed 13 August 2024.
88. Jakiela J, Bacon W. Generating a single cell matrix using Alevin and combining datasets (bash + R) (Galaxy Training Materials). https://gxy.io/GTN:T00378.Accessed 13 August 2024.
89. Bacon W. Filter, plot and explore single-cell RNA-seq data with Scanpy (Galaxy Training Materials). https://gxy.io/GTN:T00247 .Accessed 13 August 2024.
90. Howells M, Bacon W. Filter, plot and explore single-cell RNA-seq data with Scanpy (Python) (Galaxy Training Materials). https://gxy.io/GTN:T00358.Accessed 13 August 2024.
91. Goclowski C, Moreno P. Filter, plot, and explore single cell RNA-seq data with Seurat (Galaxy Training Materials). https://gxy.io/GTN:T00438. Accessed 13 August 2024.
92. Goclowski C. Filter, plot, and explore single cell RNA-seq data with Seurat (R) (Galaxy Training Materials). https://gxy.io/GTN:T00366. Accessed 13 August 2024.
93. Loach M, Bacon W, Jakiela J, et al. Inferring single cell trajectories with Scanpy (Galaxy Training Materials). https://gxy.io/GTN:T00379.Accessed 13 August 2024.
94. Bacon W, Jakiela J, Tekman M. Inferring single cell trajectories with Scanpy (Python) (Galaxy Training Materials). https://gxy.io/GTN:T00244.Accessed 13 August 2024.
95. Jakiela J. Inferring single cell trajectories with Monocle3 (Galaxy Training Materials). https://gxy.io/GTN:T00249.Accessed 13 August 2024.
96. Jakiela J. Inferring single cell trajectories with Monocle3 (R) (Galaxy Training Materials). https://gxy.io/GTN:T00336. Accessed 13 August 2024.
97. Single Cell subpage of Galaxy Training Network (GTN). 2024. https://training.galaxyproject.org/training-material/topics/single-cell. Accessed 20 November 2024.
98. Bacon WA. Pre-processing scRNA-seq data using Alevin in Galaxy [Data set]. Zenodo. 2021. https://doi.org/10.5281/zenodo.4574153.Accessed 13 August 2024.
99. Jakiela J. Combining datasets after Alevin pre-processing—Galaxy Training Material [Data set]. Zenodo. 2024. https://doi.org/10.5281/zenodo.10852529.Accessed 13 August 2024.
100. Bacon WA. AnnData object for case study tutorials [Data set]. Zenodo. 2022. https://doi.org/10.5281/zenodo.7053673 .Accessed 13 August 2024.
101. Bacon WA. Trajectories_Jupyter_Tutorial [Data set]. Zenodo. 2021. https://doi.org/10.5281/zenodo.7075718.Accessed 13 August 2024.
102. Jakiela J. CDS input for Monocle3 tutorial—Galaxy Training Material [Data set]. Zenodo. 2023. https://doi.org/10.5281/zenodo.10397366.Accessed 13 August 2024.
103. Jakiela J. Trajectory analysis: monocle3 in RStudio—Galaxy training material [Data set]. Zenodo. 2022. https://doi.org/10.5281/zenodo.7455590. Accessed 13 August 2024.

104. Galaxy Europe server. https://usegalaxy.eu. Accessed 20 November 2024.
105. Galaxy US server. https://usegalaxy.org. Accessed 20 November 2024.
106. Galaxy Australia server. https://usegalaxy.org.au. Accessed 20 November 2024.
107. Galaxy ToolShed. https://toolshed.g2.bx.psu.edu. Accessed 20 November 2024.

Received: August 13, 2024. **Revised:** October 28, 2024. **Accepted:** November 26, 2024