BMC Bioinformatics

SOFTWARE Open Access

ideal: an R/Bioconductor package for interactive differential expression analysis

Federico Marini[1,2*], Jan Linke[1,2] and Harald Binder[3]

*Correspondence: marinif@uni-mainz.de
[1] Center for Thrombosis and Hemostasis (CTH), University Medical Center of the Johannes Gutenberg University Mainz, Langenbeckstr. 1, 55131 Mainz, Germany
Full list of author information is available at the end of the article

Abstract

Background: RNA sequencing (RNA-seq) is an ever increasingly popular tool for transcriptome profiling. A key point to make the best use of the available data is to provide software tools that are easy to use but still provide flexibility and transparency in the adopted methods. Despite the availability of many packages focused on detecting differential expression, a method to streamline this type of bioinformatics analysis in a comprehensive, accessible, and reproducible way is lacking.

Results: We developed the `ideal` software package, which serves as a web application for interactive and reproducible RNA-seq analysis, while producing a wealth of visualizations to facilitate data interpretation. `ideal` is implemented in R using the Shiny framework, and is fully integrated with the existing core structures of the Bioconductor project. Users can perform the essential steps of the differential expression analysis workflow in an assisted way, and generate a broad spectrum of publication-ready outputs, including diagnostic and summary visualizations in each module, all the way down to functional analysis. `ideal` also offers the possibility to seamlessly generate a full HTML report for storing and sharing results together with code for reproducibility.

Conclusion: `ideal` is distributed as an R package in the Bioconductor project (http://bioconductor.org/packages/ideal/), and provides a solution for performing interactive and reproducible analyses of summarized RNA-seq expression data, empowering researchers with many different profiles (life scientists, clinicians, but also experienced bioinformaticians) to make the *ideal* use of the data at hand.

Keywords: RNA-Seq, Differential expression, Interactive data analysis, Data visualization, Transcriptomics, R, Bioconductor, Shiny, Web application, Reproducible research

Background

Over the last decade, RNA sequencing (RNA-seq, [1]) has become the standard experimental approach for accurately profiling gene expression. Complex biological questions can be addressed, also thanks to the development of specialized software for data analysis; these aspects are, e.g., reviewed in the works of Conesa et al. [2] and Van den Berge et al. [3], which cover a broad spectrum of the possible applications.

Differential expression analysis is a very commonly used workflow [4–7], whereby researchers seek to define the mechanisms for transcriptional regulation, enabled by the comparisons between, for example, different conditions, genotypes, tissues, cell types, or time points. The ultimate aim is to determine robust sets of genes that display changes in expression, and to contextualize them at the level of molecular pathways, in a way that can explain the biological systems under investigation and provide actionable insights in basic research and clinical settings [8].

Established end-to-end analysis procedures (such as [9–11]) are nowadays available, yet often bioinformatics analyses can be a challenging and time-consuming bottleneck, especially for users whose programming skills do not suffice to flexibly combine and customize the steps (and software modules) of a complete analysis pipeline.

Current software implementations for quality assessment (e.g. FastQC, https://www.bioinformatics.babraham.ac.uk/projects/fastqc/), preprocessing, alignment [12], and quantification [13–16] have streamlined the generation of large matrices of the transcriptome profiles. These intermediate results have to be provided as input to software for differential expression analysis [7, 17, 18], which constitute core components of the R/Bioconductor project [19, 20].

In our previous work [21], we reviewed a selection of interfaces for RNA-seq analysis from the perspective of a life scientist, defining criteria that cover many essential aspects of every software/framework, including e.g. installation, usability, flexibility, hardware requirements, and reproducibility. Building upon these results, we subsequently developed a tool that satisfies a broad set of requirements for differential expression analysis and is presented in the following.

As a result of close collaborations with wet-lab life scientists and clinicians, we developed our proposal as an interactive Shiny [22] web based application in the `ideal` R/Bioconductor package, which guides the user through all operations in a complete differential expression analysis. `ideal` provides an integrated platform for extracting, visualizing, interpreting, and sharing RNA-seq datasets, similar to what our Bioconductor `pcaExplorer` package does for the fundamental step of exploratory data analysis [23].

The `ideal` package takes as input a count matrix and the experimental design information, for allowing to also analyze complex designs (such as multifactorial experimental setups), while making it easy to reproduce and share the analysis sessions, promoting effective collaboration between scientists with different skill sets, an open research culture [24], and the adherence to the FAIR Guiding Principles for scientific data [25]. Moreover, `ideal` delivers a wide range of information-rich visualizations, charts, and tables, both for diagnostic and downstream steps, which taken together form a comprehensive, transparent, and reproducible analysis of RNA-seq data.

`ideal` reprises and expands some design choices of `pcaExplorer`, with an improved documentation system based on tooltips and on the adoption of self-paced learning tours of the main functionality (with the `rintrojs` library [26]). State saving and automated HTML report generation via knitr and R Markdown, following a template bundled in the package itself (which can be edited by the experienced users to address specific questions), ensure code reproducibility, which has received increasing attention in recent years [27–32].

There is a multitude of software packages, developed to operate on tabular-like summarized expression data, or on formats which might derive from their results [33–49]. We provide a comprehensive overview of their functionality and characteristics in Additional file 1: Table S1.

Similar to `ideal`, many of the existing tools accept count data in tabular format, and proceed to compute differentially expressed genes, accompanying this with visualizations (both focused on samples and on features) and sometimes downstream operations such as functional analysis for the identified subset. In most of the existing software, single genomic features of interest can be inspected, with some support for identifier conversion. The majority of these solutions are distributed as standalone web applications (commonly in R/Shiny, although some are written in Javascript); still, not all of these can be easily distributed as packages, or deployed seamlessly to private local instances. While still underrepresented, some of the tools allow the generation of an analysis product, which in many cases is based on a report in R Markdown [50], dynamically generated at runtime.

Overall, the existence of many such software packages highlights the need for a user-friendly framework to generate rich outputs for assisting analysis and interpretation, yet currently none of the existing proposals is offering the complete set of features we implemented in our work, with a full integration in the Bioconductor environment, and a seamless combination of interactivity and reproducibility.

The `ideal` package integrates and connects a number of R/Bioconductor packages, wrapping the current best practices in RNA-seq data analysis with a coherent user interface, and can deliver multiple types of outputs and visualizations to easily translate transcriptomic datasets into knowledge and insights. By leveraging the efficient core structures of Bioconductor, `ideal` allows flexible additional visualizations, as it is possible with custom scripts or with other GUI-based tools such as the `iSEE` package [51, 52].

`ideal` is available at http://bioconductor.org/packages/ideal/, and the application can additionally be deployed as a standalone web-service, as we did for the publicly hosted version available at http://shiny.imbei.uni-mainz.de:3838/ideal, where the readers can explore the functionality of the app.

Implementation

General design of ideal

`ideal` is written in the R programming language, wiring together the functionality of a number of widely used packages available from Bioconductor. `ideal` uses the framework of the `DESeq2` package to generate the results for the Differential Expression (DE) step, as it was found to be among the best performing in many experimental settings for simple and complex eukaryotes [53, 54]. Internally, this framework includes the estimation of size factors (with the median ratio method) and of the dispersion parameters, followed by the generalized linear model fitting and testing itself.

The web application and all its features are provided by a call to the `ideal()` function, which fully exploits the Shiny reactive programming paradigm to efficiently (re-)generate the rendered components and outputs upon detection of changes in the input widgets.

The layout of the user interface is built on the shinydashboard package [55], with a sidebar containing widgets for the general options, and the main panel structured in different tabs that mirror the different steps to undertake to perform a comprehensive differential expression analysis, from data setup to generating a full report. The task menu in the dashboard header contains buttons for state saving, either as binary RData files or as environments in the interactive workspace, accessible after closing the app.

Alongside features like tooltips, based on the bootstrap components in the shinyBS package [56], ideal uses collapsible elements containing text to quickly introduce the functionality of the diverse modules, and guided tours of the user interface via the rintrojs package [26], which provide means for learning-by-doing by inviting the user to perform actions that reflect typical use cases in each section. The *Quick viewer* widget in the sidebar keeps track of the essential objects, which are either provided upon launch, or computed at runtime, while valueBox elements (whose color turns from red to green once the corresponding object is available) above the main panel display a brief summary of each.

We invested particular attention in designing the application to guide the user through the different workflow steps (Fig. 1). This can be appreciated in the steps for the *Data Setup* panel, which appear dynamically once the required input and parameters are provided. Moreover, we used conditional panels to activate the functionality of each tab only if the underlying objects are available.

The base and ggplot2 [57] graphics systems are used to generate static visualizations, enabling interactions by brushing or clicking on them in the Shiny framework. Interactive heatmaps are generated with the d3heatmap [58] package, and tables are displayed as interactive objects for efficient navigation via the DT package [59].

We provide an R Markdown template for a complete DE analysis together with the package, and users can customize its contents by editing or adding chunks in the embedded editor (based on the shinyAce package [60]). Combining this object with the current status of the reactive widgets in the main tabs of the application, an HTML report is generated for preview at runtime, and can later be exported, shared with colleagues, or simply stored (Fig. 1, bottom section).

ideal has been tested on macOS, Linux, and Windows. It can be downloaded from the Bioconductor project page (http://bioconductor.org/packages/ideal/), and its development version can be found at https://github.com/federicomarini/ideal/. Alternatively, ideal is also provided as a Bioconda recipe [61], simplifying the installation procedure in isolated software environments e.g. in combined use with Snakemake [62], with binaries available at https://anaconda.org/bioconda/bioconductor-ideal.

Since ideal is normally installed on local systems, its speed and performance will vary depending on the hardware specifications available. In our experience, a typical modern laptop or workstation with at least 8/16 GB RAM is sufficient to run ideal on a variety of datasets. For example, in the analysis of the experimental dataset described in Additional File S2 (24 samples, with 4 treatments on 6 cell lines from different donors), the core functionality of DESeq2 required slightly less than 2 GB of RAM and less than one minute on a MacBook Pro with 2,9 GHz Intel Core i7 and 16 GB of memory, and required resources can be expected to scale approximately linearly with the increase of sample numbers - as measured with the profvis package [63]. The routines for

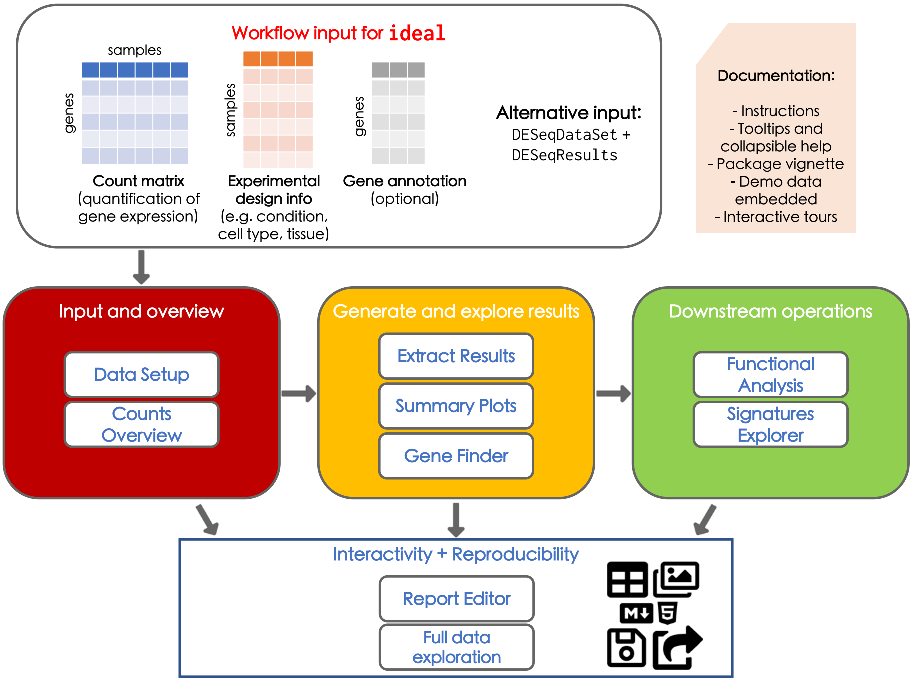


Fig. 1 Overview of the `ideal` workflow. Top section: The typical analysis with `ideal` starts by providing the count matrix for the samples of interest, together with the corresponding experimental design information. The optional gene annotation information can also be retrieved at runtime. The combination of a `DESeqDataSet` and a `DESeqResults` objects can be given as an alternate input. Package documentation includes tooltips, collapsible help elements, and instructions in the app. Together with the vignette as a detailed reference, the interactive tours guide users through the fundamental components in each step, coupled to the embedded demo dataset. Middle section: The interactive session spans from the overview on the provided input, to the generation of differential expression analysis results and their visualization, while supporting downstream operations such as functional analysis, to assist in the interpretation of the data. Bottom section: All the generated output elements can be downloaded (images, tables), as well as exported in form of a R Markdown/HTML report, a document that guarantees reproducible analyses and can be readily shared or stored. (Icons contained in this figure are contained in the collections released by Font Awesome under the CC BY 4.0 license)

functional annotation have a peak of allocated memory of ca. 4 GB, and take less than a minute to complete.

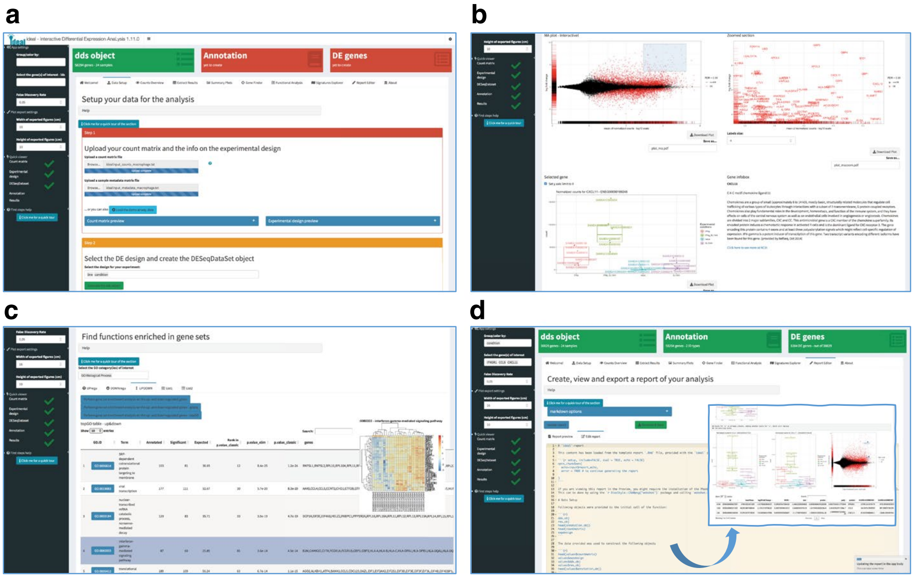
The desired depth of exploration after performing the backbone of the DE analysis is the main factor influencing the time required for completing a session with `ideal`, after familiarizing with its interface, e.g. by following the introductory tours on the demo dataset included.

The functionality of the `ideal` package is extensively described in the package vignette, regularly generated via the Bioconductor build system, and also embedded in the Welcome tab. Documentation for each function is provided, with examples rendered at the Github project page https://federicomarini.github.io/ideal/, generated with the `pkgdown` package [64].

Typical usage workflow

During the typical usage session of `ideal`, users need to provide (or upload) two essential components: (1) a gene-level count matrix (`countmatrix`), a common intermediate result after quantifying the expression measures in widely used workflows ([9–11]), and (2), the metadata table (`expdesign`) with the experimental variables for the samples of interest, as illustrated in the top panel of Fig. 1. `ideal` can accept any tabular text files, and uses simple heuristics to detect the delimiter used to separate the distinct fields; a preview on the uploaded files is shown in the collapsible boxes in the Step 1 of the *Data Setup* panel (Fig. 2a). A modal dialog informs the users about the formatting expected in the input files, with matched sample names and gene identifiers specified as column or row names. We strongly advise to perform a thorough exploratory data analysis on the input high-dimensional data, as this is a fundamental requirement in each rigorous analysis workflow. Users can refer to the `pcaExplorer` package for this purpose if an interactive approach is desired.

In the context of differential expression testing, the design argument has also to be specified, and this is normally a subset of the variables in the experimental metadata, which constitute the main source of information when submitting the data to a public repository such as the NCBI Gene Expression Omnibus (National Center for Biotechnology Information, https://www.ncbi.nlm.nih.gov/geo/). All other parameters (the corresponding `DESeqDataSet`, `DESeqResults`, and a data frame containing matched identifiers for the features of interest) for the `ideal()` function can be also constructed



ideal - Interactive Differential Expression AnaLysis 1.11.0

dds object
58294 genes - 24 samples

Annotation
yet to create

DE genes
yet to create

Welcome! | Data Setup | Counts Overview | Extract Results | Summary Plots | Gene Finder | Functional Analysis | Signatures Explorer | Report Editor | About

Setup your data for the analysis

Step 1

Upload your count matrix and the info on the experimental design

Upload a count matrix file: idealInput_counts_macrophage.txt

Upload a sample metadata matrix file: idealInput_metadata_macrophage.txt

Step 2

Select the DE design and create the DESeqDataSet object

Select the design for your experiment: line condition

Find functions enriched in gene sets

Select the GO category(ies) of interest: GO Biological Process

Create, view and export a report of your analysis

Group/color by: condition

Select the gene(s) of interest: IFNGR1 CCL5 CXCL11

dds object
58829 genes - 24 samples

Annotation
58294 genes - 2 ID types

DE genes
6384 DE genes - out of 58829

Fig. 2 Selected screenshots of the `ideal` application. **a** Data Setup panel, after uploading the expression matrix and experimental design information, here displayed with the macrophage dataset illustrated as use case in Additional file 2. The setup steps appear consecutively once the required input elements and information are provided by the user. **b** Overview of the Summary Plots panel, with an MA plot displaying expression change (as $\log_2$ Fold Change) versus average expression level. Upon interaction via brushing, a zoomed plot appears, where features are labeled to facilitate further exploration, e.g. by displaying expression plots and information fetched from the NCBI Entrez database. **c** Results from the Functional Analysis panel, with an interactive table for the enriched Biological Processes. When clicking on a row of interest, a signature heatmap is shown to better show the behavior of the features annotated to the particular Gene Ontology term. **d** The Report Editor panel, accessed after computing all required objects (as shown by the green value boxes in the upper part of the subfigure), contains a text editor which displays the default comprehensive template provided with the package - which can additionally be edited by the user. In the framed content, a screenshot of the rendered report is included, e.g. with an annotated MA plot highlighting a set of specified genes

manually on the command line and provided optionally, otherwise they will be computed at runtime.

For demonstration purposes, we include a primary human airway smooth muscle cell lines dataset [65], which can be loaded in the *Data Setup* tab. For each module in the main application, ideal gives a text introduction to the typical operations, and then encourages the user to perform these in a guided manner by following the provided rintrojs tours, which can be started by clicking on a button. Descriptions of the user interface elements are anchored to the widgets themselves, and are highlighted in sequence while the interaction with them is enabled.

When the analysis session is terminated, binary RData objects and environments in the R session can store the exported reactive values. Additional analyses can be performed on the exported values, enabling e.g. alternative methods for functional enrichment, as illustrated in a section of the package vignette. While all result files and figures generated and displayed in the user interface can also be saved locally with few mouse clicks, the generation of a full interactive HTML report is the intended concluding step. This report is created by combining the values of reactive elements with the provided template, which can be extended by experienced users. Such a literate programming approach (conceived by [66] and perfected in the knitr package) is one of the preferred methods to ensure the technical reproducibility of computational analyses [67, 68].

Additionally, users can continue exploring interactively the exported objects, if some representations are not included in ideal directly. A flexible interface to do so is represented by the iSEE Bioconductor package [51], which also fully tracks the code of the generated outputs, and we support this with a dedicated export function to a SummarizedExperiment object, with annotated rowData and colData slots filled with the results of the differential expression analysis.

Deploying ideal on a Shiny server

While we anticipate that the ideal package will typically be installed on local machines, it can also be deployed as a web application on a Shiny server, simplifying the workflow for users who want to analyze and explore their data without installing software. Deployment of an instance shared among lab members of the same research group is an exemplary use case; our proposal also supports protected instances behind institutional firewalls, e.g. if sensitive patient data is to be handled.

We describe the full procedure to set up ideal on a server and document the required steps in the GitHub repository https://github.com/federicomarini/ideal_serveredition, which can be particularly useful for bioinformaticians or IT-system administrators. Following this approach, a publicly available instance has been created and is accessible at http://shiny.imbei.uni-mainz.de:3838/ideal for demonstration purposes, where users can either explore the airway dataset or upload their own data.

Results

The functionality of ideal is described in the next sections, and is illustrated in detail for the analysis of a human RNA-seq data of macrophage immune stimulation (published in [69]) in Additional file 2 (complete use case as HTML document structured like a vignette, with text, code, and output chunks).

Data input and overview

The setup for the data analysis is carried out in the *Data Setup* panel (Fig. 2a). To guide the user through the mandatory steps without an exceeding burden of interface elements, we designed this tab in a compact way, with boxes encapsulating related widgets, appearing consecutively once the upstream actions are completed. One of the fundamental data structures for the `ideal` app is a `DESeqDataSet` object, used in the workflow based on the `DESeq2` package [9]. This is complemented by an optional annotation object, i.e. a simple data frame where different key types (e.g. ENTREZ, ENSEMBL, HGNC-based gene symbols) are matched to the identifiers for the features of interest. While this is not mandatory, it is recommended as some of the package functionality relies on the interconversion across such identifiers; `ideal` suggests the corresponding orgDb Bioconductor packages and makes it immediate to create such an object directly at runtime. Once the initial selections are finalized, the `DESeq()` command runs the necessary steps of the pipeline, displaying a textual summary and a mean-dispersion plot as diagnostic tools.

When dealing with large numbers of samples and more complex designs (entailing the computation of many coefficients), it is possible to take advantage of parallelized computation, as it is implemented for the `DESeq2` package. `ideal` provides a slider to select the number of cores to use for running the main analysis, depending on the available resources (Step 3 in the *Data Setup* panel).

A first overview on the dataset, including a set of basic summary statistics on the expressed genes, as well as the (log transformed) normalized values can be retrieved in the *Counts Overview* tab, together with pairwise scatter plots of the values. Thresholds can be introduced to subset the original dataset by keeping only genes with robust expression levels, either based on the detection in at least one sample, or on the average normalized value.

Generating and exploring the results for differential expression analysis

The *Extract Results* tab provides the functionality to generate the other fundamental data structure, namely the `DESeqResults` object. After setting the FDR threshold in the sidebar, users are prompted to define the contrast of interest for their data, selecting one of the experimental factors included in the design. When the factor of interest has three or more levels available (e.g. the cell type in the `airway` demonstration dataset), the likelihood ratio test can be used instead of the Wald test, to allow for an ANOVA-like analysis across groups.

Further refinements to the results can be obtained by activating independent filtering [70], or selecting the more powerful Independent Hypothesis Weighting (IHW) framework [71], to ameliorate the multiple testing issue by incorporating an informative covariate, e.g. the mean gene expression [72]. Shrinkage of the effect sizes is also optionally performed on the log fold change estimates, to reflect the higher levels of uncertainty for lowly expressed genes. Interactive tables for the results are shown, with embedded links to the ENSEMBL browser and to the NCBI Gene portal to facilitate deeper exploration of shortlisted genes. Moreover, a number of diagnostic plots are generated, including histograms for unadjusted p-values, also using small multiples to stratify them on different mean expression value classes, a Schweder-Spjøtvoll plot [73], and a histogram of the estimated log fold changes.

More visualizations are included in the *Summary Plots* tab (Fig. 2b), where users can zoom in the MA plot (M, $\log_2$ fold change vs A, average expression value) representation by brushing an area on the element. From the magnified subset, by clicking close to a selected gene, it is possible to obtain a gene expression boxplot (with the individual jittered observations superimposed), together with an info-box with details retrieved from the NCBI resource portal [74]. Heatmaps (both static and interactive) and volcano plots (log fold change vs $\log_{10}$ of the p-value) deliver alternative views of the underlying result table, or interesting subsets of it.

Iterations oriented towards exploring a set of features of interest are made easier by the *Gene Finder* tab. Genes can be shortlisted on the fly, adding them from the sidebar selectize widget. For each feature of interest, a plot comparing the normalized values is displayed, and these are included in an annotated MA plot, where the selected subsets are highlighted on the plot and their values are shown in a corresponding table. Alternatively, a gene list can be uploaded directly as text file to obtain the same output, with the ease of providing entire sets of genes (e.g. a file with all cytokines, or a curated list of genes affected by a particular transcription factor) in one step.

Putting results into biological context

Many times it is challenging to make sense out of a carefully derived table of DE results, since it is not straightforward to identify the common biological themes that might be underlying the observed phenotypes. `ideal` offers different means to help researchers in meaningfully interpreting their RNA-seq data. The *Functional Analysis* panel offers three alternatives for gene set overrepresentation analysis, relying on `topGO` [75], `goseq` [76], or the `goana()` function in the `limma` package [7]; users can perform the enrichment tests on genes that are significantly differentially regulated, either split by direction of expression change, or combined in one list (Fig. 2c). Additionally, users can upload up to two custom lists of genes, which can be compared to the one derived from the result object, in order to detect significant overlaps among the sets of interest, which can be represented via Venn diagrams or UpSet plots [77].

The Gene Ontology (GO, [78]) terms enriched in each list can be interactively displayed, with links to the AmiGO database (http://amigo.geneontology.org/), as well as heatmaps displaying the expression values for all the DE genes annotated to a particular signature.

Expanding on this functionality, `ideal` provides a *Signatures Explorer* panel, where a signature heatmap can be generated for any gene set provided in the gmt format, common to many sources of curated databases (MSigDB, WikiPathways). Conversion between identifier types is guided in the user interface, and so is the aspect of the final heatmap, where rows and columns can be clustered to better display existing patterns in the data, or transformations (such as mean centering or row standardization) help to bring the feature expression levels to a similar scale, for a better display in the final output.

Generating reproducible and transparent results

The focus in the development of ideal was on combining interactivity and reproducibility of the analysis. Therefore, we implemented the *Report Editor* as the toolset for enabling reproducible reports in the DE analysis step (Fig. 2d). The predefined template, embedded in the package, fetches the values of the reactive elements and the input widgets, thus capturing a snapshot of the ongoing session. Text, code, and results are all combined in an interactive HTML report, which can be previewed in the app, or subsequently exported.

This functionality is particularly appealing for less experienced users due to its automated simplicity, but experienced users can also take full benefit of it, by expanding the R Markdown document by adding or editing specific chunks of code.

State saving, activated by the buttons in the task menu in the header, stores the content of the current session into binary data objects or environments accessible from the global workspace.

As an additional feature, we leverage the flexibility of iSEE, the interactive SummarizedExperiment Explorer, another tool which fully supports intuitive and reproducible analyses, by assembling a serialized rds object that can be directly fed to the main iSEE() function for bespoke visualizations.

Discussion

The guiding principle for the development of our package ideal was the effective combination of usability and reproducibility, applied to one of the most widely adopted workflows in transcriptomics, i.e. the analysis of differentially expressed genes, followed by the downstream analyses based on functional enrichment among the subset of detected features.

Several software packages have been developed to operate on this tabular-like summarized expression data, or on formats which might derive from their results, and a comprehensive comparison of their features is presented in Additional file 1: Table S1. Notably, these tools differ by their set of included features (ranging from first exploration to downstream analysis steps), implementation (with R/Shiny, python, and JavaScript as main choices), format of distribution (packages, local web app, webserver), and ease of implementation in existing pipelines (e.g. by leveraging widely adopted class structures, or requiring and providing text files for portability across systems). The comparison with other tools is also available online (https://federicomarini.github.io/ideal_supplement), linked to a Google Sheet where the individual characteristics of each tool will be updated, in order to provide a tool for users who might be seeking advice on which solution to adapt for their needs (accessible at https://docs.google.com/spreadsheets/d/167XV0w18P0FSld1dt6owN4C2Esxl5FU2QTo4D-wclz0/edit?usp=sharing).

Our proposal complements the existing pcaExplorer package, where the main exploratory data analysis steps are performed, and provides a platform for performing a complete differential expression analysis with the ease of interactivity, accompanied by a number of diagnostic plots, often overseen in other software tools. The combination of interactivity with reproducibility (Fig. 1, bottom panel) is an essential aspect to consider for generating robust and transparent analyses, substantiated by code which can also be used for didactic purposes, learning the best current practices with the state-of-the-art methods included in our package.

ideal fully supports widely used standard classes from Bioconductor, and thus allows seamless integration with many R packages for further downstream processing and within existing data analysis pipelines, while also benefiting from a thriving community of developers. ideal itself is part of Bioconductor, and thus is integrated into a build system that continuously checks all of its components and their interoperability, guaranteeing that the available set of features is correctly interfacing with the latest version of the package dependencies. Notably, the Bioconductor project enforces a number of best practices to enhance the usability of its components, with both internal and external documentation (for the individual functions and as complete tutorials, in form of vignettes), as well as providing unit test sets to ensure the software is working as expected: ideal adheres to these guidelines, which can be essential to define robust software [79] that can be adopted by a wide range of users.

A possible use case for deploying ideal is tightly related to data and result sharing. Distributing data in raw and processed format, together with a set of results, is becoming a practice that enables efficient data mining and can help ensure their reproducibility and reusability [25]. Stemming from a close collaboration with life and medical scientists, our tool allows researchers to share their work with other interested parties, starting from the operations during the collaboration phase, and continuing after publication, where broader audiences can effectively digest contents as they are presented from the authors.

The faster turnover in generating insights, thanks to the accessible interface and the multiple outputs, constitutes a significant advantage for reducing the time to new results, or alternatively for re-analyzing publicly available RNA-seq datasets. ideal provides a platform to facilitate discoveries in a standardized way, which at the same time improves the transparency and the reproducibility of the analyses. Indeed, one possible use case that we envision is the submission of a comprehensive notebook/report as Supplementary Material for a manuscript, so that the results are presented in a transparent manner, thus facilitating the contribution of reviewers, as well as the re-usability of analysis code. A rich technical description of parameters and software used would also greatly facilitate the writing of the "Material and Methods" sections, in a way that fully captures steps, parameters, code, and software versions.

Conclusion

The infrastructure provided by the ideal R/Bioconductor package delivers a web browser application that guarantees ease of use through interactivity and a dynamic user interface, together with reproducible research, for the essential step of differential expression investigation in RNA-seq analysis. The combination of these two features is a key factor for efficient, quick, and robust extraction of information, while leveraging the facilities available in the Bioconductor project in terms of classes and statistical methods.

The wealth of information that can be extracted while running the app may play a critical role when choosing the tools to adopt in a project. Still, to ensure the proper interpretation of the output results, the interaction of wet-lab scientists with collaborators with additional bioinformatics/biostatistics expertise is essential. The design choices for ideal aim at making this communication as robust and easy as possible, possibly defining this tool as the *ideal* way of approaching this step.

Following the criteria used in our previous overview on RNA-seq analysis interfaces [21], our package reaches out to the life/medical scientist, being simple to install and use, based on robust statistical methods, and offering multiple levels of documentation. ideal allows scientists to easily take control of the analysis of RNA-seq data, while providing an accessible framework for reproducible research, which can be extended according to the user's needs.

Availability and requirements

Project name: ideal
Project home page:http://bioconductor.org/packages/ideal/ (release) and https://github.com/federicomarini/ideal/ (development version)
Archived version:https://doi.org/10.5281/zenodo.4056654, package source as gzipped tar archive of the version reported in this article
Project documentation: rendered at https://federicomarini.github.io/ideal/
Operating systems: Linux, Mac OS, Windows
Programming language: R
Other requirements: R 3.4 or higher, Bioconductor 3.5 or higher
License: MIT
Any restrictions to use by non-academics: none.

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1186/s12859-020-03819-5.

Additional file 1: Comparison of software for analyzing interactively RNA-Seq data, including link to the related publications (if available) and to the source code repositories. Evaluation criteria are included in the dedicated sheet. The information contained in this table are also available online at https://federicomarini.github.io/ideal_supplement, displaying the content of the ideal supplement Google Sheet, accessible at https://docs.google.com/spreadsheets/d/167XV0w18P0FSld1dt6owN4C2Esxl5FU2QTo4D-wclz0/edit?usp=sharing.

Additional file 2: Complete use case for the ideal package, based on the macrophage immune stimulation dataset (Interferon Gamma treatment vs naive cells).

Abbreviations

ANOVA: Analysis of variance; DE: Differential expression; FDR: False discovery rate; GO: Gene ontology; HGNC: HUGO (Human Genome Organisation) Gene Nomenclature Committee; IHW: Independent hypothesis weighting; MA plot: M (log ratio) versus A (mean average) plot; MSigDB: Molecular signatures database; NCBI: National Center for Biotechnology Information; RNA-seq: RNA sequencing.

Acknowledgements

We thank Sebastian Schubert and Carina Santos of the Ruf lab (CTH Mainz) for fruitful discussions and their feedback as early adopters of the ideal package, as well as the users' community for their helpful suggestions. We also thank Charlotte Soneson (FMI, Basel), Miguel Andrade, Wolfram Ruf, Franziska Härtner, Gerrit Toenges, and Konstantin Strauch for their helpful comments on the manuscript.

Authors' contributions

FM: Conceptualization, Methodology, Software Implementation, Writing-Original Draft Preparation, Writing-Review & Editing. JL: Software Implementation, Writing-Review & Editing. HB: Supervision, Writing-Review & Editing. All authors read and approved the final manuscript.

Funding
Open Access funding enabled and organized by Projekt DEAL. The work of FM is supported by the German Federal Ministry of Education and Research (BMBF 01EO1003).

Availability of data and materials
Data used in the described use cases (as demo dataset and in Additional file 2) is available from the following articles: The airway smooth muscle cell RNA-seq is included in PubMed ID: 24926665 (https://doi.org/10.1371/journal.pone.0099625). GEO entry: GSE52778, accessed from the Bioconductor experiment package `airway` (http://bioconductor.org/packages/airway/, version 1.7.0). The data set on the macrophage immune stimulation is included in PubMed ID: 29379200 (https://doi.org/10.1038/s41588-018-0046-7). Dataset deposited at the ENA (ERP020977, project id: PRJEB18997) and accessed from the Bioconductor experiment package `macrophage` package (http://bioconductor.org/packages/macrophage/, version 1.3.1)
The `ideal` package can be downloaded from its Bioconductor page http://bioconductor.org/packages/ideal/ or the GitHub development page https://github.com/federicomarini/ideal/. `ideal` is also provided as a recipe in Bioconda (https://anaconda.org/bioconda/bioconductor-ideal).

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Center for Thrombosis and Hemostasis (CTH), University Medical Center of the Johannes Gutenberg University Mainz, Langenbeckstr. 1, 55131 Mainz, Germany. [2] Institute of Medical Biostatistics, Epidemiology and Informatics (IMBEI), University Medical Center of the Johannes Gutenberg University Mainz, Obere Zahlbacher Str. 69, 55131 Mainz, Germany. [3] Institute of Medical Biometry and Statistics, Faculty of Medicine and Medical Center, University of Freiburg, Stefan-Meier-Str. 26, 79104 Freiburg, Germany.

Received: 9 January 2020 Accepted: 15 October 2020
Published online: 09 December 2020

References

1. Wang Z, Gerstein M, Snyder M. RNA-Seq: a revolutionary tool for transcriptomics. Nat Rev Genet. 2009;10(1):57–63. https://doi.org/10.1038/nrg2484.
2. Conesa A, Madrigal P, Tarazona S, Gomez-Cabrero D, Cervera A, McPherson A, Szcześniak MW, Gaffney DJ, Elo LL, Zhang X, Mortazavi A. A survey of best practices for RNA-seq data analysis. Genome Biol. 2016;17(1):13. https://doi.org/10.1186/s13059-016-0881-8.
3. Van den Berge K, Hembach KM, Soneson C, Tiberi S, Clement L, Love MI, Patro R, Robinson MD. RNA sequencing data: Hitchhiker's guide to expression analysis. Ann Rev Biomed Data Sci. 2019;2(1):139–73. https://doi.org/10.1146/annurev-biodatasci-072018-021255.
4. Oshlack A, Robinson MD, Young MD. From RNA-seq reads to differential expression results. Genome Biol. 2010;11(12):220. https://doi.org/10.1186/gb-2010-11-12-220.
5. Love MI, Anders S, Kim V, Huber W. RNA-Seq workflow: gene-level exploratory analysis and differential expression. F1000Research. 2015;4:1070. https://doi.org/10.12688/f1000research.7035.1.
6. Soneson C, Love MI, Robinson MD. Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. F1000Research. 2015;4(0):1521. https://doi.org/10.12688/f1000research.7563.2.
7. Ritchie ME, Phipson B, Wu D, Hu Y, Law CW, Shi W, Smyth GK. Limma powers differential expression analyses for RNA-sequencing and microarray studies. Nucleic Acids Res. 2015;43(7):47. https://doi.org/10.1093/nar/gkv007.
8. Beigh M. Next-generation sequencing: the translational medicine approach from "bench to bedside to population". Medicines. 2016;3(2):14. https://doi.org/10.3390/medicines3020014.
9. Anders S, McCarthy DJ, Chen Y, Okoniewski M, Smyth GK, Huber W, Robinson MD. Count-based differential expression analysis of RNA sequencing data using R and Bioconductor. Nat Protoc. 2013;8(9):1765–86. https://doi.org/10.1038/nprot.2013.099.
10. Pertea M, Kim D, Pertea GM, Leek JT, Salzberg SL. Transcript-level expression analysis of RNA-seq experiments with HISAT. StringTie and Ballgown. Nat Protoc. 2016;11(9):1650–67. https://doi.org/10.1038/nprot.2016-095.
11. Love MI, Soneson C, Patro R. Swimming downstream: statistical analysis of differential transcript usage following salmon quantification, vol. 7, p. 952 (2018). https://doi.org/10.12688/f1000research.15398.1. https://f1000research.com/articles/7-952/v1
12. Dobin A, Davis CA, Schlesinger F, Drenkow J, Zaleski C, Jha S, Batut P, Chaisson M, Gingeras TR. STAR: ultrafast universal RNA-seq aligner. Bioinformatics. 2013;29(1):15–21. https://doi.org/10.1093/bioinformatics/bts635.
13. Anders S, Pyl PT, Huber W. HTSeq-a Python framework to work with high-throughput sequencing data. Bioinformatics. 2015;31(2):166–9. https://doi.org/10.1093/bioinformatics/btu638.
14. Liao Y, Smyth GK, Shi W. featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. Bioinformatics. 2014;30(7):923–30. https://doi.org/10.1093/bioinformatics/btt656.
15. Patro R, Duggal G, Love MI, Irizarry RA, Kingsford C. Salmon provides fast and bias-aware quantification of transcript expression. Nature Methods (2017). https://doi.org/10.1038/nmeth.4197. arXiv:1505.02710

16. Bray NL, Pimentel H, Melsted P, Pachter L. Near-optimal probabilistic RNA-seq quantification. Nat Biotechnol. 2016;34(5):525–7. https://doi.org/10.1038/nbt.3519. arXiv:1505.02710.
17. Love MI, Huber W, Anders S. Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. 2014;15(12):550. https://doi.org/10.1186/s13059-014-0550-8.
18. McCarthy DJ, Chen Y, Smyth GK. Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. Nucleic Acids Res. 2012;40(10):4288–97. https://doi.org/10.1093/nar/gks042.
19. Gentleman RC, Carey VJ, Bates DM, Bolstad B, Dettling M, Dudoit S, Ellis B, Gautier L, Ge Y, Gentry J, Hornik K, Hothorn T, Huber W, Iacus S, Irizarry R, Leisch F, Li C, Maechler M, Rossini AJ, Sawitzki G, Smith C, Smyth G, Tierney L, Yang JYH, Zhang J. Bioconductor: open software development for computational biology and bioinformatics. Genome Biol. 2004;5(10):80. https://doi.org/10.1186/gb-2004-5-10-r80.
20. Huber W, Carey VJ, Gentleman R, Anders S, Carlson M, Carvalho BS, Bravo HC, Davis S, Gatto L, Girke T, Gottardo R, Hahne F, Hansen KD, Irizarry Ra, Lawrence M, Love MI, MacDonald J, Obenchain V, Oleś AK, Pagès H, Reyes A, Shannon P, Smyth GK, Tenenbaum D, Waldron L, Morgan M. Orchestrating high-throughput genomic analysis with Bioconductor. Nat Methods. 2015;12(2):115–21. https://doi.org/10.1038/nmeth.3252.
21. Poplawski A, Marini F, Hess M, Zeller T, Mazur J, Binder H. Systematically evaluating interfaces for RNA-seq analysis from a life scientist perspective. Briefings Bioinf. 2016;17(2):213–23. https://doi.org/10.1093/bib/bbv036.
22. Chang W, Cheng J, Allaire JJ, Xie Y, McPherson J. shiny: Web Application Framework for R (2016). https://cran.r-project.org/package=shiny
23. Marini F, Binder H. pcaExplorer: an R/Bioconductor package for interacting with RNA-seq principal components. BMC Bioinform. 2019;20(1):331. https://doi.org/10.1186/s12859-019-2879-1.
24. Nosek BA, Alter G, Banks GC, Borsboom D, Bowman SD, Breckler SJ, Buck S, Chambers CD, Chin G, Christensen G, Contestabile M, Dafoe A, Eich E, Freese J, Glennerster R, Goroff D, Green DP, Hesse B, Humphreys M, Ishiyama J, Karlan D, Kraut A, Lupia A, Mabry P, Madon T, Malhotra N, Mayo-Wilson E, McNutt M, Miguel E, Paluck EL, Simonsohn U, Soderberg C, Spellman BA, Turitto J, VandenBos G, Vazire S, Wagenmakers EJ, Wilson R, Yarkoni T. Promoting an open research culture. Science. 2015;348(6242):1422–5. https://doi.org/10.1126/science.aab2374.
25. Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten J-W, da Silva Santos LB, Bourne PE, Bouwman J, Brookes AJ, Clark T, Crosas M, Dillo I, Dumon O, Edmunds S, Evelo CT, Finkers R, Gonzalez-Beltran A, Gray AJG, Groth P, Goble C, Grethe JS, Heringa J, 't Hoen PAC, Hooft R, Kuhn T, Kok R, Kok J, Lusher SJ, Martone ME, Mons A, Packer AL, Persson B, Rocca-Serra P, Roos M, van Schaik R, Sansone S-A, Schultes E, Sengstag T, Slater T, Strawn G, Swertz MA, Thompson M, van der Lei J, van Mulligen E, Velterop J, Waagmeester A, Wittenburg P, Wolstencroft K, Zhao J, Mons B. The FAIR guiding principles for scientific data management and stewardship. Sci Data. 2016;3:160018. https://doi.org/10.1038/sdata.2016.18.
26. Ganz C. rintrojs: a Wrapper for the Intro.js Library. J Open Source Softw. 2016;1(6):2016. https://doi.org/10.21105/joss.00063.
27. Peng RD. Reproducible research in computational science. Science. 2011;334(6060):1226–7. https://doi.org/10.1126/science.1213847.
28. McNutt M. Journals unite for reproducibility. Science. 2014;346(6210):679–679. https://doi.org/10.1126/science.aaa1724.
29. Stodden BV, Mcnutt M, Bailey DH, Deelman E, Hanson B, Heroux MA, Ioannidis JPA, Taufer M. Enhancing reproducibility for computational methods. Science. 2016;354(6317):1240–1. https://doi.org/10.1126/science.aah6168.
30. Marini F, Binder H. Development of applications for interactive and reproducible research: a case study. Genom Comput Biol. 2016;3(1):1–4.
31. Eglen SJ, Marwick B, Halchenko YO, Hanke M, Sufi S, Gleeson P, Silver RA, Davison AP, Lanyon L, Abrams M, Wachtler T, Willshaw DJ, Pouzat C, Poline J-B. Toward standard practices for sharing computer code and programs in neuroscience. Nat Neurosci. 2017;20(6):770–3. https://doi.org/10.1038/nn.4550.
32. Perkel JM. Data visualization tools drive interactivity and reproducibility in online publishing. Nature. 2018;554(7690):133–4. https://doi.org/10.1038/d41586-018-01322-9.
33. Younesy H, Möller T, Lorincz MC, Karimi MM, Jones SJ. VisRseq: R-based visual framework for analysis of sequencing data. BMC Bioinf. 2015;16(Suppl 11):2. https://doi.org/10.1186/1471-2105-16-S11-S2.
34. Nelson JW, Sklenar J, Barnes AP, Minnier J. The START App: a web-based RNAseq analysis and visualization resource. Bioinformatics. 2016;33(3):624. https://doi.org/10.1093/bioinformatics/btw624.
35. Su S, Law CW, Ah-Cann C, Asselin-Labat M-L, Blewitt ME, Ritchie ME. Glimma: interactive graphics for gene expression analysis. Bioinformatics. 2017;33(13):2050–2. https://doi.org/10.1093/bioinformatics/btx094.
36. Harshbarger J, Kratz A, Carninci P. DEIVA: a web application for interactive visual analysis of differential gene expression profiles. BMC Genom. 2017;18(1):47. https://doi.org/10.1186/s12864-016-3396-5.
37. Gardeux V, David FPA, Shajkofci A, Schwalie PC, Deplancke B. ASAP: A web-based platform for the analysis and interactive visualization of single-cell RNA-seq data. Bioinformatics. 2017;33(19):3123–5. https://doi.org/10.1093/bioinformatics/btx337.
38. Lim JH, Lee SY, Kim JH. TRAPR: R package for statistical analysis and visualization of RNA-Seq data. Genom Inf. 2017;15(1):51. https://doi.org/10.5808/gi.2017.15.1.51.
39. Li Y, Andrade J. DEApp: an interactive web interface for differential expression analysis of next generation sequence data. Source Code Biol Med. 2017;12(1):10–3. https://doi.org/10.1186/s13029-017-0063-4.
40. Zhu Q, Fisher SA, Dueck H, Middleton S, Khaladkar M, Kim J. PIVOT: platform for interactive analysis and visualization of transcriptomics data. BMC Bioinf. 2018;19(1):6. https://doi.org/10.1186/s12859-017-1994-0.
41. Ge SX, Son EW, Yao R. iDEP: an integrated web application for differential expression and pathway analysis of RNA-Seq data. BMC Bioinf. 2018;19(1):534. https://doi.org/10.1186/s12859-018-2486-6.
42. Monier B, McDermaid A, Zhao J, Fennell A, Ma Q. IRIS-EDA: an integrated RNA-Seq interpretation system for gene expression data analysis. bioRxiv, 283341 (2018). https://doi.org/10.1101/283341
43. McDermaid A, Monier B, Zhao J, Liu B, Ma Q. Interpretation of differential gene expression results of RNA-seq data: review and integration. Briefings Bioinf. 2018;00(April):1–11. https://doi.org/10.1093/bib/bby067.

44. Schultheis H, Kuenne C, Preussner J, Wiegandt R, Fust A, Bentsen M, Looso M. WIlsON: web-based interactive Omics VisualizatioN. Bioinformatics. 2018;33(17):2699–705. https://doi.org/10.1093/bioinformatics/bty711. arXiv:103549.
45. Kucukural A, Yukselen O, Ozata DM, Moore MJ, Garber M. DEBrowser: interactive differential expression analysis and visualization tool for count data. BMC Genom. 2019;20(1):6. https://doi.org/10.1186/s12864-018-5362-x.
46. Choi K, Ratner N. iGEAK: an interactive gene expression analysis kit for seamless workflow using the R/shiny platform. BMC Genom. 2019;20(1):177. https://doi.org/10.1186/s12864-019-5548-x.
47. Price A, Caciula A, Guo C, Lee B, Morrison J, Rasmussen A, Lipkin WI, Jain K. DEvis: an R package for aggregation and visualization of differential expression data. BMC Bioinf. 2019;1–7: https://doi.org/10.1186/s12859-019-2702-z.
48. Tintori SC, Golden P, Goldstein B. Differential expression gene explorer (DrEdGE): a tool for generating interactive online visualizations of gene expression datasets. Bioinformatics. 2020;8(5):55. https://doi.org/10.1093/bioinformatics/btz972.
49. Su W, Sun J, Shimizu K, Kadota K. TCC-GUI: a Shiny-based application for differential expression analysis of RNA-Seq count data. BMC Res Notes. 2019;12(1):133. https://doi.org/10.1186/s13104-019-4179-2.
50. Allaire J, Xie Y, McPherson J, Luraschi J, Ushey K, Atkins A, Wickham H, Cheng J, Chang W. Rmarkdown: Dynamic Documents for R. (2018). R package version 1.10. https://CRAN.R-project.org/package=rmarkdown
51. Rue-Albrecht K, Marini F, Soneson C, Lun ATL. iSEE: interactive SummarizedExperiment Explorer. F1000Research. 2018;7(0):741. https://doi.org/10.12688/f1000research.14966.1.
52. Amezquita RA, Carey VJ, Carpp LN, Geistlinger L, Lun AT, Marini F, Rue-Albrecht K, Risso D, Soneson C, Waldron L, Pages H, Smith M, Huber W, Morgan M, Gottardo R, Hicks SC. Orchestrating Single-Cell Analysis with Bioconductor. bioRxiv. 2019;590562: https://doi.org/10.1101/590562.
53. Schurch NJ, Schofield P, Gierliński M, Cole C, Sherstnev A, Singh V, Wrobel N, Gharbi K, Simpson GG, Owen-Hughes T, Blaxter M, Barton GJ. How many biological replicates are needed in an RNA-seq experiment and which differential expression tool should you use? RNA. 2016;22(6):839–51. https://doi.org/10.1261/rna.053959.115.
54. Froussios K, Schurch NJ, Mackinnon K, Gierliński M, Duc C, Simpson GG, Barton GJ. How well do RNA-Seq differential gene expression tools perform in a complex eukaryote? A case study in Arabidopsis thaliana. Bioinformatics, 090753 (2019). https://doi.org/10.1093/bioinformatics/btz089
55. Chang W, Borges Ribeiro B. Shinydashboard: Create Dashboards with 'Shiny'. (2018). R package version 0.7.0. https://CRAN.R-project.org/package=shinydashboard
56. Bailey E. shinyBS: Twitter Bootstrap Components for Shiny. (2015). R package version 0.61. https://CRAN.R-project.org/package=shinyBS
57. Wickham H. Ggplot2: Elegant Graphics for Data Analysis. Springer (2016). http://ggplot2.org
58. Cheng J, Galili T. D3heatmap: Interactive Heat Maps Using 'htmlwidgets' and 'D3.js'. (2018). R package version 0.6.1.2. https://CRAN.R-project.org/package=d3heatmap
59. Xie Y. DT: A Wrapper of the JavaScript Library 'DataTables'. (2018). R package version 0.4. https://CRAN.R-project.org/package=DT
60. Nijs V, Fang F, Trestle Technology LLC, Allen J. shinyAce: Ace Editor Bindings for Shiny. (2018). R package version 0.3.2. https://CRAN.R-project.org/package=shinyAce
61. Grüning B, Dale R, Sjödin A, Chapman BA, Rowe J, Tomkins-Tinch CH, Valieris R, Köster J. Bioconda: sustainable and comprehensive software distribution for the life sciences. Nat Methods. 2018;15(7):475–6. https://doi.org/10.1038/s41592-018-0046-7.
62. Köster J, Rahmann S. Snakemake-a scalable bioinformatics workflow engine. Bioinformatics. 2012;28(19):2520–2. https://doi.org/10.1093/bioinformatics/bts480.
63. Chang W, Luraschi J, Mastny T. Profvis: Interactive Visualizations for Profiling R Code. (2019). R package version 0.3.6. https://CRAN.R-project.org/package=profvis
64. Wickham H, Hesselberth J. Pkgdown: Make Static HTML Documentation for a Package. (2018). R package version 1.1.0. https://CRAN.R-project.org/package=pkgdown
65. Himes BE, Jiang X, Wagner P, Hu R, Wang Q, Klanderman B, Whitaker RM, Duan Q, Lasky-Su J, Nikolos C, Jester W, Johnson M, Panettieri Ra, Tantisira KG, Weiss ST, Lu Q. RNA-Seq transcriptome profiling identifies CRISPLD2 as a glucocorticoid responsive gene that modulates cytokine function in airway smooth muscle cells. PLoS ONE. 2014;9(6):99625. https://doi.org/10.1371/journal.pone.0099625.
66. Knuth DE. Literate programming. Comput J. 1984;27(2):97–111. https://doi.org/10.1093/comjnl/27.2.97.
67. Sandve GK, Nekrutenko A, Taylor J, Hovig E. Ten simple rules for reproducible computational research. PLoS Comput Biol. 2013;9(10):1003285. https://doi.org/10.1371/journal.pcbi.1003285.
68. Stodden V, Miguez S. Best practices for computational science: software infrastructure and environments for reproducible and extensible research. J Open Res Softw. 2014;2(1):21. https://doi.org/10.5334/jors.ay.
69. Alasoo K, Rodrigues J, Mukhopadhyay S, Knights AJ, Mann AL, Kundu K, Hale C, Dougan G, Gaffney DJ. Shared genetic effects on chromatin and gene expression indicate a role for enhancer priming in immune response. Nat Genet. 2018;50(3):424–31. https://doi.org/10.1038/s41588-018-0046-7.
70. Bourgon R, Gentleman R, Huber W. Independent filtering increases detection power for high-throughput experiments. Proc Nat Acad Sci. 2010;107(21):9546–51. https://doi.org/10.1073/pnas.0914005107.
71. Ignatiadis N, Klaus B, Zaugg JB, Huber W. Data-driven hypothesis weighting increases detection power in genome-scale multiple testing. Nat Methods. 2016;13(7):577–80. https://doi.org/10.1038/nmeth.3885.
72. Korthauer K, Kimes PK, Duvallet C, Reyes A, Subramanian A, Teng M, Shukla C, Alm EJ, Hicks SC. A practical guide to methods controlling false discoveries in computational biology. Genome Biol. 2019;20(1):118. https://doi.org/10.1186/s13059-019-1716-1.
73. Schweder T, Spjøtvoll E. Plots of P-values to evaluate many tests simultaneously. Biometrika. 1982;69(3):493–502. https://doi.org/10.1093/biomet/69.3.493.
74. Agarwala R, Barrett T, Beck J, Benson DA, Bollin C, Bolton E, Bourexis D, Brister JR, Bryant SH, Canese K, Charowhas C, Clark K, DiCuccio M, Dondoshansky I, Feolo M, Funk K, Geer LY, Gorelenkov V, Hlavina W, Hoeppner M, Holmes B, Johnson M, Khotomlianski V, Kimchi A, Kimelman M, Kitts P, Klimke W, Krasnov S, Kuznetsov A, Landrum MJ, Landsman D, Lee JM, Lipman DJ, Lu Z, Madden TL, Madej T, Marchler-Bauer A, Karsch-Mizrachi I, Murphy T, Orris R, Ostell J, O'Sullivan C, Palanigobu V, Panchenko AR, Phan L, Pruitt KD, Rodarmer K, Rubinstein W, Sayers EW, Schneider V, Schoch CL, Schuler

GD, Sherry ST, Sirotkin K, Siyan K, Slotta D, Soboleva A, Soussov V, Starchenko G, Tatusova TA, Todorov K, Trawick BW, Vakatov D, Wang Y, Ward M, Wilbur WJ, Yaschenko E, Zbicz K. Database resources of the national center for biotechnology information. Nucleic Acids Res. 2017;45(D1):12–7. https://doi.org/10.1093/nar/gkw1071.

75. Alexa A, Rahnenführer J, Lengauer T. Improved scoring of functional groups from gene expression data by decorrelating GO graph structure. Bioinformatics. 2006;22(13):1600–7. https://doi.org/10.1093/bioinformatics/btl140.
76. Young MD, Wakefield MJ, Smyth GK, Oshlack A. Gene ontology analysis for RNA-seq: accounting for selection bias. Genome Biol. 2010;11(2):14. https://doi.org/10.1186/gb-2010-11-2-r14.
77. Lex A, Gehlenborg N, Strobelt H, Vuillemot R, Pfister H. UpSet: visualization of intersecting sets. IEEE Trans Visual Comput Graphics. 2014;20(12):1983–92. https://doi.org/10.1109/TVCG.2014.2346248.
78. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene ontology: tool for the unification of biology. Nat Genet. 2000;25(1):25–9. https://doi.org/10.1038/75556. arXiv:10614036.
79. Taschuk M, Wilson G. Ten simple rules for making research software more robust. PLoS Comput Biol. 2017;13(4):1005412. https://doi.org/10.1371/journal.pcbi.1005412.

Publisher's Note

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