Bioinformatics, YYYY, 0–0
doi: 10.1093/bioinformatics/xxxxx
Advance Access Publication Date: DD Month YYYY
Manuscript Category

Subject Section

KimLabIDV: Application for Interactive RNA-Seq Data Analysis and Visualization

Qin Zhu[1], Stephen A Fisher[1], Hannah Dueck[1], Sarah Middleton[1], Mugdha Khaladkar[1], Young-Ji Na[1] and Junhyong Kim[1,*]

[1]Department of Biology, University of Pennsylvania, Philadelphia, PA, USA

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Summary: We developed the KimLabIDV package (IDV) to facilitate fast and interactive RNA-Seq data analysis and visualization. IDV supports routine analysis including differential expression analysis, correlation analysis, dimension reduction, clustering and classification. With the graphical user interface IDV provides, users can easily obtain statistical test results and publication-quality graphs with their data. IDV further supports program state saving and report generation, so that all analysis can be saved, shared and reproduced.

Availability and implementation: IDV is implemented in R and is distributed as an R package. It is developed based on the Shiny framework, multiple R packages and a collection of scripts written by members of Junhyong Kim's Lab at University of Pennsylvania. IDV supports any system that has R and a modern web browser installed. It can be downloaded from Kim Lab Software Repository (http://kim.bio.upenn.edu/software).

Contact: junhyong@upenn.edu

1 Introduction

Next generation sequencing brought digital gene counts and present them as high-dimensional expression matrixes for downstream analysis. Various statistical packages have been developed for the analysis of count-based gene expression data, including DESeq (Love et al., 2014), edgeR (McCarthy et al., 2012), SCDE (Kharchenko et al., 2014) and monocle (Trapnell et al., 2014). These packages are written in R and expect a significant understanding of R programming language of the user. Unfortunately, learning a programming language and associated statistical procedures are often a daunting task, especially for researchers who spend most of the time working in the wet lab. On the other hand, these researchers often have first-hand knowledge about the data, and it would be very helpful if they can conveniently visualize the outcome and assess the experiment. One solution would be to build a user-friendly interface for current RNA-Seq data analysis packages and scripts, while also providing the user the power of data filtering, transformation and parameter setting.

The Shiny package elegantly bridges the gap between R and JavaScript-based web applications (Chang et al., 2016). It translates user-end commands such as pressing buttons and entering parameters into R interpretable reactive data objects, and present R computed results as dynamic web contents. In addition, multiple interactive graphics packages such as networkD3 (Gandrud et al., 2015) and ggvis (Chang and Wickham, 2015) have been developed in recent years, which leverages the power of JavaScript libraries such as d3.js and brought interactivity to R-based data visualization.

The KimLabIDV package (IDV) we present here builds upon the above mentioned techniques and R packages. It also harbors scripts written by our lab members to facilitate routine RNA-Seq data analyses. With a few button clicks, users can navigate through various analysis modules and obtain high-quality graphs and tabulated analysis results.

2 Description

Current IDV workflow is specifically designed for count based gene expression data produced by tools such as HTSeq (Anders et al., 2014) and featureCounts (Liao et al., 2014). A counts matrix or data folder can be directly uploaded into IDV and optionally normalized by DESeq (Fig. 1). The uploaded data can be further transformed and filtered, based on various criteria such as minimum expression level or a marker gene list. Users can also upload the experiment design information such as groups and batches, which can be visualized as distinctive color points/sidebars and be used for analysis such as differential expression (DE).

The analysis modules of IDV include basic summary statistics, DE analyses, clustering, correlation analyses, heatmap generation, dimension reduction, cell state ordering and classification. If samples contain spike-

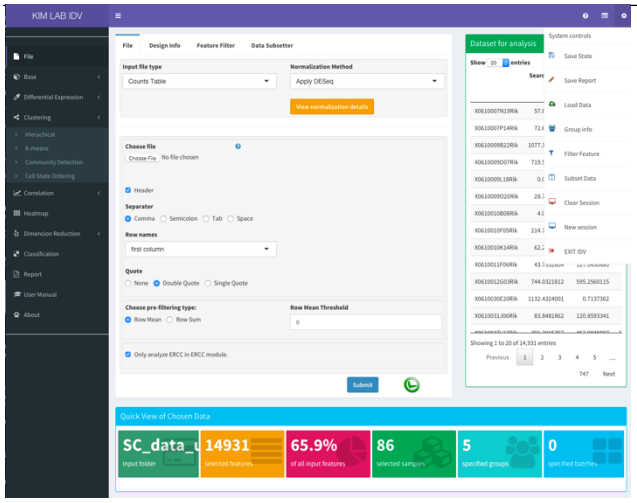

Fig. 1. KimLabIDV file input and system control panel. IDV supports input of data folder, count matrix and IDV states. It allows the user to filter or subset the data based on various criteria. Group/batch information can be uploaded in the "design info" section for differential expression analysis. Through the left side panel users can navigate through various analysis modules, including summary statistics, differential expression (DE) analysis, clustering, correlation analysis, heatmap, dimension reduction and classification. The top right corner contains a dropdown menu for system control, where users can save and load the program state, launch additional sessions and generate analysis reports.

in control mixes such as ERCC (Lemire et al., 2011), IDV will also separately analyze the ERCC count distribution. In the summary statistics module, a user can quickly assess sample quality by looking at bar plots showing the number of detected genes, total read counts and sequencing depths. The rank-frequency plots and mean-variability plots provide a general overview of gene expression and dispersion pattern across samples. For DE analysis, IDV implements a graphical user interface (GUI) for the Mann-Whitney U test and several popular DE packages such as DESeq and SCDE. The results are presented as dynamic tables including all essential statistics such as maximum likelihood estimation and confidence intervals. Each gene entry in the table can be clicked and visualized as violin plots or box plots, showing the actual expression level across groups or batches.

One useful feature of IDV is that it provides users multiple visualization options by exploiting the power of various plotting packages. For example, users have the choice of visualizing gene heatmaps using either the heatmap.2 function in the gplots package (Warnes et al., 2015), which produce publication-quality graphs, or the d3heatmap package (Cheng et al., 2015), which allows users to zoom in on the heatmap and thus is more suitable for exploratory data analysis. For principal component analysis (PCA), IDV uses three different packages to present the 2D and 3D projections. The ggvis package allow sample names and relevant information to be revealed on the plot as mouse-over labels, while the ggbiplot (Vu, 2011) presents the loadings of each genes on the graph as vectors. The threejs package (Lewis, 2015) fully utilize the power of WebGL and outputs rotatable 3D projections.

IDV provides users extensive control over parameter choices, as an R programmer would have. Each analysis module contains multiple control widgets where users can adjust the parameters and obtain updated analysis results on the fly. On occasions when results in one module may be used as inputs for another, IDV effectively bridges these modules and load the results in memory once it's been computed. For example, the cell-to-cell distances computed in the SCDE module can be used as distance measures for hierarchical clustering or tree generation. Likewise, after community detection has been performed, users can conveniently color samples by community in modules such as heatmap, PCA and t-SNE (van der Maaten, 2013; Krijthe, 2015).

Users can generate analysis reports in IDV by simply choosing from a list of analysis modules. The last state of selected analysis modules will be captured and be converted to R-markdown codes (Allaire et al., 2016), which in turn produce dynamic reports or presentations. IDV states are automatically saved in cases of browser refresh, crash or user exit, and can also be manually exported, shared and loaded. Thus, all analysis performed in IDV are fully reproducible.

3 Summary

IDV is an extensive R-based GUI for the visualization and analysis of RNA-Seq data. It builds upon various statistical packages and data visualization techniques, and provides non-programmers a convenient way to explore their data and perform reproducible research.

References

Allaire,J. et al. (2016) rmarkdown: Dynamic Documents for R. *CRAN*.

Anders,S. et al. (2014) HTSeq--a Python framework to work with high-throughput sequencing data. *Bioinformatics*, 31, 166-169.

Chang,W. et al. (2016) shiny: Web Application Framework for R. *CRAN*.

Chang,W. and Wickham,H. (2015) ggvis: Interactive Grammar of Graphics. *CRAN*.

Cheng,J. et al. (2015) d3heatmap: Interactive Heatmaps Using 'htmlwidgets' and 'D3.js'. *CRAN*.

Gandrud,C. et al. (2015) networkD3: D3 JavaScript Network Graphs from R. *CRAN*.

Kharchenko,P. et al. (2014) Bayesian approach to single-cell differential expression analysis. *Nature Methods*, 11, 740-742.

Krijthe,J. (2015) Rtsne: T-Distributed Stochastic Neighbor Embedding using Barnes-Hut Implementation. *CRAN*.

Lemire,A. et al. (2011) Development of ERCC RNA spike-in control mixes. *Journal of biomolecular techniques: JBT* 22(Suppl).

Lewis,B. (2015) threejs: Interactive 3D Scatter Plots and Globes. *CRAN*.

Liao,Y. et al. (2014) featureCounts: an efficient general purpose program for assigning sequence reads to genomic features. *Bioinformatics* 30(7): 923-930.

Love,M. et al. (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol*, 15.

McCarthy,D. et al. (2012) Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation. *Nucleic Acids Research*, 40, 4288-4297.

Trapnell,C. et al. (2014) The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nat Biotechnol*, 32, 381-386.

van der Maaten,L. (2013) Barnes-Hut-SNE. *In Proceedings of the International Conference on Learning Representations*.

Vu,V. (2011) ggbiplot: A ggplot2 based biplot. *CRAN*.

Warnes,G. et al. (2015) gplots: Various R Programming Tools for Plotting Data. *CRAN*.