Visual Analytic Report
Visual Exploration and Analysis of Pancreatic Adenocarcinoma Expression in Platelets with Differential Co-Expression Analysis DCETool

Ali Toccacieli,1799754

***Abstract*—Differential co-expression (DCE) analysis (or network analysis) , over Tumor Educated Platelet (TEPs) trascriptome data, improves selection of potential biomarker for Pancreatic Adenocarcinoma (PAAD) diagnosis. Differential Expression (DE) Analysis over TEPs trascriptome data , followed by classification, gave confirm on their importance [4]. While this is already known , same is not for Differential Co-Expression (DCE) Network Analysis. Differential co-expression analysis studies diseases and phenotypic variations by finding modules of genes whose co-expression patterns vary across conditions. In fact DCE analysis over DE genes resulted in an increase of scores in classification. This project wants to focus the visualization of RNA-seq data , giving the possibility to the user to interact with the genes in order to further compute DCE network analysis to extract significant genes for further classification. The initial genes will be selected by Principal Component Analysis (PCA) and further filtered by interacting with the visualizations in order to finally compute and visualize the most important genes coming out from the respective DCE Network Analysis over the previous gene selection.**

I. Introduction

THIS project was created with the intention to let the users visualize what they are working with , when speaking about RNA-seq data, in order to have more clarity over data and analysis results meaning. Gene expression and co-expression analysis require a lot of computation and data manipulation that is usually studied by looking at the resulting tables. Usually this is done in program languages such as Python or R. Different packages can help the user by plot results in various charts, but still , being static, they are still not enough. Because this packages needs to satisfy different data, usually the hard job of "personalizing" the chart is left to the end user, that takes time to learn the various packages. This is why this project , with the intent to help the users to have a clear and fast visualization over significant data about RNA-seq transcriptome, let the user, not only visualize, but compute directly analysis by interacting with the visualization of its data (DCE in this version). This is because the aim of this tool is to add Differential Co-Expression Network Analysis over DE genes in order to obtain a restricted number of genes (possible biomarkers) that have better performance than DE genes ranked by their P-Value.

A. *Intended Users*

Being a BioInformatics tool , the application can be used by BioInformatics in order to have a visualization over RNA-seq data while applying actual analysis on its data (DCE) and interacting with it. Same is applied to Scientist that are not well integrated with Informatics (e.g. program languages such as Python or R) and want to study their data just by interacting with guided visualizations. Both groups can take advantage of this tool in order to improve Precision Medicine and improve classification over Diseases such as Cancer , even in specific cases such as Tumor Educated Platelet analysis. Main attention though is given to users searching for biomarkers in cancer, in fact given the possibility to compute Differential co-expression Network Analysis over RNA-seq data from Deferentially expressed genes, the user is able to retrieve a list of genes that can significantly reduce the previous amount of genes obtaining the same, if not even more precise, accuracy over pan-cancer classification. This is why the resulting genes, that will be displayed, will be the genes , with a connection degree, above the 90 percentile of the total degrees greater than zero.

II. Implementation

A. *Data*

mRNA sequencing of tumor-educated blood platelets were obtained from the Gene Expression Omnibus (GEO) [1] database under accession code GSE68086 [2] using the GEOquery R package function getGEO. R was the program language used in order to obtain the data and export it as csv files in order to manipulate data. The data was composed by 57736 genes and 285 samples. The samples used for this project are the ones related to healthy controls(n=55) and pancreatic adenocarcinoma tumor (n = 39). Genes that did not satisfy the condition for which any sample related to that specific gene has a read count greater than 5 were removed (n = 14411 genes left). Furthermore genes with a value of logCPM $<$ 3 and abs(logFC) $<$ 1 were removed (1319 genes left). Gene expression was computed over the genes in order to determine their respective regulation. The data that has been used for the visualization comprehend the following : count matrix (1319 genes x 94 samples for a total of 123986 reads) , DGE results for each gene (1319 x 5 for a total of 6595 values) , patient class (94 x 2 for a total of 188 values).

1) Gene Expression Analysis: The data was subject to multiple preprocessing, because the data needed to be seen under different features. The first preprocessing regards the data about the gene regulation with respect to the condition. The program language used in order to reach this goal was R Studio. The data , in a matrix form, has been add to a DGEList object (edgeR object). edgeR is the package used for make Differential Expression Analysis. After the object was ready I computed the normalization using the R function *calcNormFactors*. The method used for normalization was Trimmed Mean of M-values (TMM) . The reason of this decision is that TMM is an effective method for estimating relative RNA production levels from RNA-seq data [3]. The TMM method estimates scale factors between samples that can be incorporated into currently used statistical methods for DE (Differential Expression) analysis. In essence, TMM normalization assumes that the majority of genes, common to both samples, are not differentially expressed. After normalization, a design matrix was built and a contrast was created (Pancreatic Adenocarciroma - Healthy Control). Common, tagwise and trended dispersion were estimated. After fitting of negative binomial models, differentially expressed transcripts were determined using a generalized linear model (GLM) likelihood ratio test (LRT). The results were saved and imported in my project. Note that this step does not actually intend to filter genes but rather to remove genes that are not significant for the visualization.

B. Dimensionality Reduction : PCA

The main work was made by the Principal Component Analysis (PCA) . In fact PCA was used to determine the key genes that had a big influence in putting healthy control and pancreatic adenocarcinoma into two different clusters. PCA was computed over all the 1319 genes and the genes that had the biggest influence were used to our final goal (Differential co-expression Analysis). In fact PCA not only shows some important information about our data (Tumor Educated Platelet can discriminate conditions), but will help to select those genes with which we will interact in all our visualizations. For this purpose the package sklearn was used in a back-end server using Flask. The features were the genes while the sample were the patients. In order to make it work, I passed the count matrix to the back-end server, which transposed the matrix (because genes were on rows) and compute the Pricipal Component Analysis over this matrix. Genes with the largest variation between samples have the most influence on the principal components, I.E. genes highly expressed in some patients and not expressed in others will have a lot of variation and influence on the Principal Components. The 1^{st} PC captures the most variation in the data, the 2^{nd} PC captures the second most variation in the data. Samples with similar transcripts patterns will cluster together. In fact from the results we could clearly see that the 1^{st} PC had the biggest influence in putting Healthy Control patients (samples) on the left and Pancreatic Adenocarcinoma patients (samples) on the right, that is why I looked at the influence scores in PC1 in order to select the top genes to use for further analysis.

C. Charting

Charts were completely made with D3.js into a single Javascript File in order to manage width and height on window resize.

D. Back-End

Back-End server was made with Flask. Flask is a micro-framework Web wrote in Python. The server runs in local and make use of Post and Get functions to communicate with Javascript (where all the visualization is built). All analysis such as Principal Component Analysis and Differential Co-Expression Network Analysis was made here by calls from the Javascript file.

E. Styling

Styling was completely made with CSS. The CSS file presents more than 800 lines of code in order to manage not only color and styiling , but manage resizing and responsivness with media queries.

III. Visual Enviroment

This tool can be used for multiple purposes, but the inteded one is Differential Co-Epression Network Analysis. For this goal the component were named by their order of use. Each of this components were built using D3.js with a significant help of CSS.

A. Responsive Design

Responsive design was possible thanks to CSS. The tool was inside a single div of type container. The multiple div inside were made responsive by applying a display of type grid , changing order and size using media queries. In total 5 div were built inside the parent container. Each of this div contains more components such as a legend, a title section (were the user can interact with tooltips and buttons) and the chart. To make the text responsive, global variables were added to the *body:root* in order to change them globally by using media queries. The charts instead went through more work. Each chart was built in D3.js (for more details see next subsections) , for make them responsive , parent width and height were passed as props. A listener of type *window.addEventListener('resize',render)* was used. The function called by the listener re-render each chart in order to update the width and height based on the new size of its parent div, such that the charts satisfy complete responsiveness to resizing being compatible with any kind of device (e.g. mobile , tablet, notebook).

B. *User Support*

User support was something really important to implement. Each component comes with a tooltip, that will help the user understand what the chart is about and how to interact with it. Even if some explanations are straight forward to understand without any hint, more information are better than less. Each tooltip works with any kind of resizing, the right position were studied such that in any kind of device , the tooltip will be visible. Other than this , there is a main button that shows a brief tutorial on one of the multiple studies the user can make out of this tool.

C. *Visualizations*

*1) **V1 - Principal Component Analysis**:* For the Principal Component Analysis a scatter plot was used (Figure 1). The data was passed through a back end server. The data was then called by a fetch function directly in javascript. The x and y axis were built using a linear scale. The x and y coordinates represented the two Principal Components. In order to make the axis display the data as more centralized as possible, the axis domain and range were determined using the d3 function *d3.max* and *d3.min* for the domain. A color variable was created using *d3.scaleOrdinal()* with domain as patient class (condition) and range of 2 colours. For draw the data , circles were appended and filled with the previously mentioned color variable. In order to visualize the patients id , a tooltip was added by appending a div in proximity (absolute position) to the corresponding circle. After this a zoom was implemented. This was done in order to focus where dots used to overlap. In order to give feedback on which dot is selected, each selected dot on mouse hover has an increase of radius. The zoom was implemented by adding a rect over which the user , with the help of the scroll of the mouse, can zoom in and zoom out. When we call the function zoom, a function *updateChart* is called which will update the axis and the circles that are included in the zoomed area. An image of a scroll with the text ”zoom” was added below the legend , in order to give the user an intuitive guide to interaction. Thanks to a selector , from this visualization we can further select the top genes by their significance in variance , and those genes will be used through out all the analysis flow.

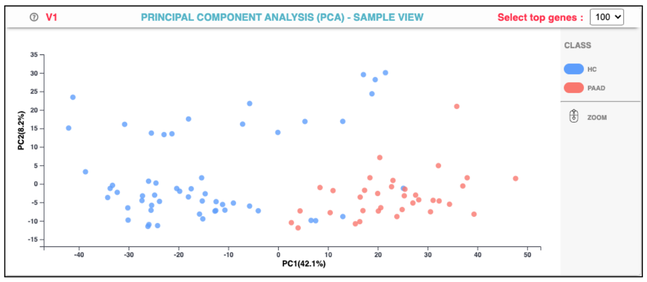


Fig. 1: V1 — Principal Component Analysis Visualization

*2) **V2 - Mean Difference Scatterplot**:* This visualization [Figure 2] is a crucial visualization for interact and select the genes that will be used for the final analysis (DCE). The following visualization follows the same concept of the first one but the intention and interaction are totally different. The intentions are to exploit to the user the genes under the aspect of their respective log fold change with respect to their log counts per million value on Pancreatic Adenocarcinoma against Healthy Donors. With this visualization the user have an easier visualization over logarithmic counts per million and is able to select those genes satisfying a specific value for both. Here the interaction plays an important role. In fact a brush area (that correspond to the width and height of the plot) was added , in order to let the user select a set of genes. When selected, this genes goes into a list that will be passed to the table, that will immediately show the ensembl and symbol of the selected genes. In order to give a straight forward feedback on the selected genes to the user, the opacity and stroke of the selected circles (genes) changes, to be precise the stroke color is black. Axis grid was added to help the user select a specific range. In fact without the grid it resulted difficult to select a specific range of values , for both, log fold change and logarithmic counts per million.

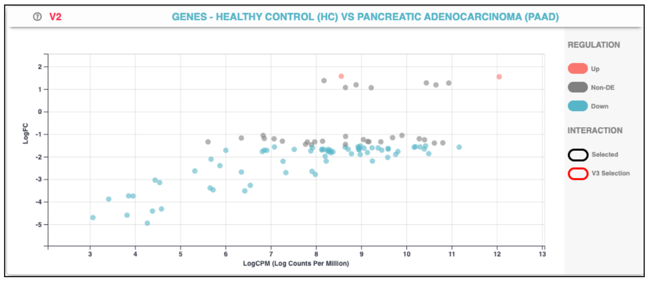


Fig. 2: V2 — Mean Difference PAAD - HC Scatterplot

*3) **V3 - Parallel Coordinates**:* As Visualization V2, the data are the genes, and to be precise, they are the same genes displayed in visualization V2. In order to give full control to the user over gene selection, a parallel coordinate was added to the tool. In fact thanks to this visualization [Figure 3] the user is able to select not only the already available log fold change, but the False Discovery Rate (FDR), LR , and the module value of the log fold change, such that he can select the log fold change value, without caring about its sing. The parallel coordinated was implemented by creating 4 axis. Important here, is to mention that one of the four axis had to use a different scale : *scaleLog*. This is because the FDR axis needed log scale values to be displayed. This was implemented by adding an if condition over the dimensions loop, already created for make the axis. The lines were added and divided into background lines and foreground lines. To each axis a brush was appended , the brush calls an update function that update foreground to only display selected values. Parallel data is then saved to send it to the table that will be , as in the case of V2 , updated with the selected genes. Important is to say that a button of type reset selection was added. This is because for the user is not intuitive to deselect the previous selection by clicking on each of the selected axis. This is why a ***”Reset Selection”*** button was added. This button will trigger a function that will remove the brushes and remove the stroke color over the previously selected data.

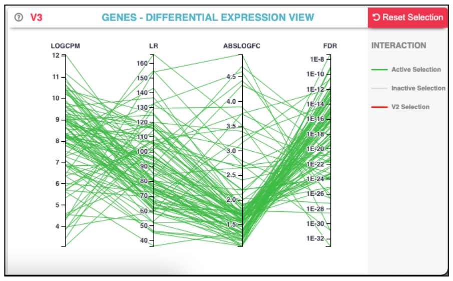
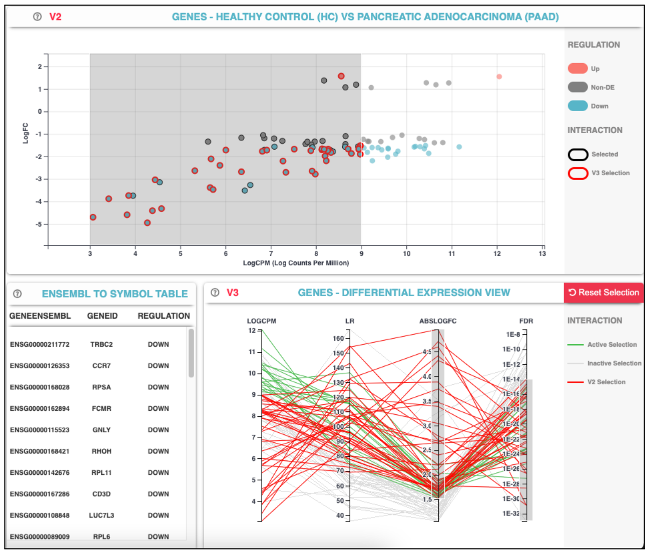


Fig. 3: V3 — Differential Expression Analysis — Genes View

*4) **V2 and V3 - Coordinated Visualizations**:* Visualization V2 and Visualization V3 are coordinated [Figure 4]. In fact when the brush is applied to either one of the two charts, the selection over the same data will be shown in the other chart. This could be done , because the data on both chart is the same, but with different axis (except for logFC which is present in both). The coordinate visualization works as follows : **(i)** When brush is applied only in V2 , the circles have the usual black scatter , while the parallel lines becomes red. This is done thanks to a list that extract the selected values from V2 and update the stroke color (red) of the foreground lines on visualization V3. **(ii)** When brush is applied only in V3 then the lines that are selected are shown as foreground lines as usual, while in visualization V2 the circles , with the same technique used from V2 to V3, changes stroke color into red. **(iii)** When there is one or more brush on both visualizations, the red stroke on circles, and red stroke on the foreground lines of V3 are the intersection of the gene selected on all brushes. This is the third and last list that save the intersection of the two selections, this list will be used to send the intersected data to the table.



ENSEMBL TO SYMBOL TABLE

GENEENSEMBL	GENEID	REGULATION
ENSG00000211772	TRBC2	DOWN
ENSG00000126353	CCR7	DOWN
ENSG00000168028	RPSA	DOWN
ENSG00000162894	FCMR	DOWN
ENSG00000115523	GNLY	DOWN
ENSG00000168421	RHOH	DOWN
ENSG00000142676	RPL11	DOWN
ENSG00000167286	CD3D	DOWN
ENSG00000108848	LUC7L3	DOWN
ENSG00000089009	RPL6	DOWN

Fig. 4: Coordinated Visualizations

*5) **V4 - Differential Co-Expression Analysis**:* This last chart represent a bar-plot [Figure 5] . To this bar-plot, it has been added a tool tip. The same tool tip that was used for the principal component analysis but with different data. The bar-plot was created using function *map* of the selected data. Important is to specify that the data, that is used for generate the bar-plot, is the data selected by V2 or V3 or the intersection of them (if both brushes are active). A grid was added too, in order to help the user understand the degree of the respective gene. In order to let the user compute the analysis, a button is provided. Two other buttons are available too , that are the download network button and the download genes button.

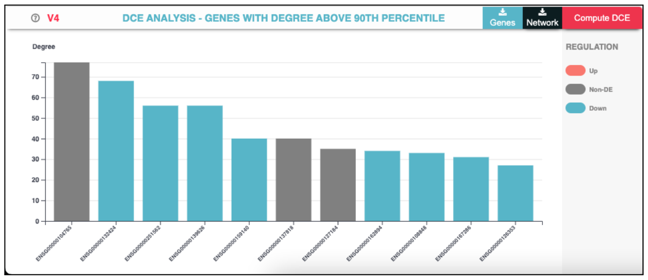


Fig. 5: V4 — Differential Co-Expression Analysis

IV. Analytic

This tool is centered in one specific analysis that is the Differential Co-Expression Network Analysis. Because the computations behind this analysis are many, it is necessary to devote a section for it.

*1) **Input**:* The input required is a count matrix of RNA-seq data. The count matrix that is used as input is composed by samples and genes. The genes in this case are the genes selected by the user from visualization V2 and V3. If none of those visualization has any active selection, than all the genes are used. For GET and POST data, a text/javascript script was added to the HTML file. In fact when we click on the button *compute DCE*, a function *dceRequest* is called. In this function it was defined a function request that made use of the proxy *Promise*. A new ***XMLHttpRequest()*** is created and data is sent to the back end url. Now the data has moved to the ***flask server*** where we call the function that uses the data in order to make the analysis.

*2) **Differential Co-Expression Analysis**:* The function takes as input a count matrix (gene as rows and patient samples as columns). As first I filter the global data with the genes that were selected (the ones present in the new count matrix passed as input). Then the matrix is divided into two matrices , one with only healthy control samples and one with only Pancreatic Adenocarcinoma samples. The genes are the same in both matrices. Now the data start to be manipulated. (i) To Each matrix a correlation function over the transposed matrix is applied. The method used for correlation is Pearson. Now we will have 2 matrices (n x n where n = number of genes). To this matrix a new function is applied. This function is called Fisher-Transformation , and it makes use of the following formula that will be applied to each of the values of the matrices :

$$z_{1or2} = \frac{1}{2} ln \left(\frac{1 + p_{1or2}}{1 - p_{1or2}} \right) \tag{1}$$

The function was applied to each element using the function apply. (ii) Now that we have the updated matrices , we go for another computation , the z-scores. Here is more complicate because now we need to take each element of each matrix and use both of this elements to create a new single element in a new matrix. In fact the function to make this new matrix takes as input two values , one from the healthy control matrix (after correlation and fisher-transformation) and one from the Pancreatic Adenocarcinoma matrix (such that the genes correspond) . This matrix describes us the connection between genes. The formula of the function is the following one (n1 and n2 are respectively the healthy control and tumor sample size) :

$$Z = \frac{z_1 - z_2}{\sqrt{\frac{1}{n_1 - 3} + \frac{1}{n_2 - 3}}} \tag{2}$$

Now that we have the final z-score matrix, we define a threshold (3) with which we compute the adjacency matrix. In fact the adjacency matrix will be built as follows, for each value v of the z-score matrix , given the threshold t (where $t > 0$ is always true) :

$$A(v) = \begin{cases} 1 & \text{se } v > t \\ -1 & \text{se } v < -t \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Now that we know which gene has a strong connection with other genes , and know the sign of this connection, we can define the degree for each gene. Because the degree does not depend on the sign of the connection, we will use the function *rowSums()* on the module of the number, such that both 1 and -1 will be considered as a connection to add to the final degree. The final matrix will be of length n (n = genes). Because some genes have zero connection, we remove them. Now we will have only genes with connection greater than one. Of this genes we are interested on genes that have a degree above the 90th percentile. After the 90th percentile is computed we return the gene satisfying the value (degree value greater than the 90th percentile). It may be possible that the result is empty (no enough genes , or not connection between genes at all) , in this case a string ”empty” is returned , that will be managed in the JavaScript, otherwise, as most of the cases, the genes are returned with their degree and regulation , ready to be displayed as a bar-plot in the JavaScript file.

*3) **Output:*** Once the output is ready, the dictionary is built and passed to the JavaScript from where a function for draw the bar-plot is called. In case the result is the string ”empty” mentioned before, then I first call a *d3.selectAll.remove()* in order to remove the actual bar plot and display a message , where the user is guided to select more genes in order to display a non empty result. When the new selection is made and the result is not empty, then the text is removed using the same *d3.selectAll.remove()*) and the bar-plot is displayed. Important is the resizing. Because every time the page is resized each chart is drawn again, this would make the request to the server repeat multiple times, requiring to much time from request to response. That is why the data is saved in an external variable, and if the resize is active then the data to draw will be re drawn without requesting anymore from the back-end, in order to avoid multiple request and fetch for the same data. Other than the degree-gene relation output, the full network in a list format is provided too. In fact the user will have the possibility to download the edge list with its weights (in this case negative or positive) in order to further study the network (e.g Cytoscape).

V. INSIGHTS

The questions we want to answer are the followings :

1) Can Tumor Educated Platelets (TEPs) be a valid tool for cancer diagnostic ?
2) How can Differential Co-Expression Network Analysis increase accuracy scores in pan-cancer Classification?

This tool is meant to produce genes that can improve sensitivity, specificity and so accuracy in classification of pan-cancer. This could be applied to multi-class too , with a simple update that is not intended for this version. The problem of gene expression classification is the amount of noise. In fact is really hard to understand which gene contributes to discriminate the samples between conditions (e.g. disease and healthy). Usually we work with thousands of genes , and a simple filter could remove important genes or keep genes, that not only do not contribute to discriminate our samples, but they confuse the classification. This is why Principal Component Analysis helps in selecting genes that have the largest influence over variation between samples . Other than that Differential Co-Expression Analysis let the genes , that were considered not significant under Differential Gene Expression , be considered again for their possible strong connectivity.

A. Answear to Question 1

With PCA it was clear that Tumor Educated Platelet (TEPs) are indeed a valid tool for cancer Diagnostic. In fact we can see how the samples are clustered into 2 conditions pretty well (except for some exceptions). Overall this is an incredible result. It tells us that there could exist a non-invasive procedure for cancer diagnostic. Nowadays only invasive cancer diagnostic is available such as Biopsy. This usually happens when the symptoms already showed. Instead

TEPs would just require a blood sample from the patient. It would be less invasive and easy to do. The problem left is the biomarker to study. In fact if I have 50'000 genes , it would make the procedure not feasible. This is why is important to obtain genes that can increase classification accuracy. Our aim is to find key genes that could explain signs of a normal or abnormal process(biomarker).

B. Answear to Question 2

Let's see what Differential Co-Expression Analysis does. Differential Co-Expression (DCE) is the alteration of gene co-expression patterns observed in different conditions. Gene Co-expression Networks find genes that have the tendency to exhibit similar expressions in a group of samples. It is a way to discover functions of unknown genes and their associations with diseases. As we can see the approach is different from the most known Differential Gene Expression Analysis, but we believe that a cooperation between this two can decrease the number of genes (removing noise) and so improve accuracy over pan-cancer classification.

C. Classification results

Some tests were made. To make the classification an SVM machine was used. The sample training and test sets were respectively of 70:30. Each time we select some genes set g with size n to study, we compare the results with the mean results of 100 different gene sets with size n , so we know for sure if the set g is the only set with size n to score better, or if any other set with same size n would actually score the same if not even better. This could deny or confirm the fact that genes coming out from DCE analysis provide better scores.

*1) **Scores**:* Genes with degree , after DCE analysis, greater than the 90th percentile between PCA top genes were selected. The resulting scores showed a sensitivity of 94% , a specificity of 90% and an accuracy of 92%. The mean of 100 different sets with same size but with genes below the 90th percentile showed a mean sensitivity of 86% , a specificity of 89% and an accuracy of 87%.

D. Interactive Gene Analysis

The most important feature of this tool, is that the user see hes data under different angles without the necessity to over study tables of thousands of rows and columns. For example , after Differentially Gene Expression (DGE) Analysis, a data analyst should plot the results in order to obtain deeper information from it. But if we have thousands of genes, this can result difficult. That is why usually packages in R are used in order to see DGE results under different plots. The problem is that most of the times plots that are mostly used, are static plots such as network plots, the problem of this static plots is that when the genes are thousands the network is not usable at all, and require to be exported and studied in deeper software's such as Cytoscape. Instead what it can be done in this tool is : spot the genes of interest and directly use them for DCE Analysis. The selected genes are exposed in a table such that the user is not forced to compute the DCE. It is true in the other side that packages with interactivity exist in R Studio. The problem with DGE is that usually we want to consider more factors such as FDR, logFC and logCPM. But having just a table with thousands of rows does not help to really value each of them. We could sort by column, but this would give the representation focused in only one aspect. Instead with a parallel coordinate we can select the desired value for each factor of interest, while seeing the relationship between log fold change and logarithmic count per millions. I personally worked with this analysis , and I can tell that I hardly ever had such clear visualization about my analysis just by using R Studio. Not only, for DCE I needed to filter every time different genes and then re run hundreds of code, instead here I can select genes seeing them (less error rate) and click a single button to compute hundreds lines of code.

VI. Related Proposals

RNA-seq data exploration is critical to the comprehension of the relation between samples and diseases. That is why recently more tools and software's are starting to be created for this purpose. I personally did not took inspiration from other tools , but rather from analysis experience. But there exist tool's that make similar visualization even if I have never seen a Differentially Co-Expression Network Analysis tool for RNA-seq data. The next subsection will briefly describe the other tool's and a closer look to their difference with my tool will be mentioned.

*A. **pcaExplorer***

pcaExplorer is a web application, developed in the Shiny framework, which allows the user to efficiently explore and visualize the wealth of information contained in RNA-seq datasets with PCA, performed for visualizing relationships either among samples or genes [5]. pcaExplorer additionally provides other tools typically needed during exploratory data analysis, including normalization, heatmaps, boxplots of shortlisted genes and functional interpretation of the principal components. The tool is built in order to give users the possibility to directly export plots about their data results and use them for scientific reports. The tool gives, both sample and gene, data view. Interactivity is present, giving the user the possibility to visualize selected data under different relationships. An example is the possibility to select a range of genes and then study the heatmap related to it, and the possibility to further focus on a restricted number of genes in the heatmap. pcaExplorer is entirely written in the R programming language and relies on several other widely used R packages available from Bioconductor. The main functionality can be accessed by a single call to the pcaExplorer() function, which starts the web application. The data if not normalized, will be normalized by the tool itself. It is possible to see , as a table, the full DGE Results. The tool require as input a count matrix and a sample/condition matrix.

1) Component Differences with DCETool: Let's start with the things that are in common. Both tools , pcaExplorer and DCETool, takes as input a count matrix of RNA-seq data (possibily already normalized) and a matrix containing samples information (e.g. conidition). Pca exploration is available in both tools, except that gene view is not available in DCETool. Heatmap is another difference, in fact DCETool do not let the user have a heatmap visualization over the data. This is because DCETool focuses more on DCE Analysis. In general pcaExplorer gives a more complete overview on RNA-seq data , giving the user almost all information needed. This is because the main difference between the two tools is that one focuses on giving an overview over data (pcaExplorer), and the other to compute the Differential Co-Expression Analysis over an easy-interact gene selection (DCETool). What is interesting though, is the possibility to see different Principal Components in pcaExplorer. What DCETool has that pcaExplorer do not provide is DCE Analysis. In fact there is no possibility to compute nor to show this functionality in pcaExplorer. This is because DCE Analysis is something not widely used in Gene Expression as DGE Analysis.

2) Interaction Differences with DCETool: Here the difference becomes more evident. Starting from the Principal Component Analysis (PCA), in DCETool is possible to go over each sample in order to see the patient Id and its condition. While this is true for pcaExplorer , it is not possible to see the names when the samples increase in size unless the user zoom in. This is not the only problem, the second problem is that if for instance we are studying cancer such as Lung Cancer and we want to add informations such as years of smoke prior to diagnosis , would be really easy to add in a tooltip as implemented in DCETool, thing that would not be possible instead in pcaExplorer for how is built. Another detail is that the zoom implemented in pcaExplorer is shown in a second component , while the zoom implemented in DCETool is done directly on the same component of the original size one. For what it concern the analysis flow , DCETool is not only a tool of visualization but a real guided tool for analysis. DCETool can be used instead of coding DCE Analysis in a manner of seconds. In fact while on pceExplorer the main interaction is for visualize data, DCETool let the user not only visualize and interact, but interactively make hes own personalized analysis.

B. *DiffCor*

DiffCor is a simple method for identifying pattern changes between 2 experimental conditions in correlation networks, which builds on a commonly used association measure, such as Pearson's correlation coefficient [6] . DiffCor tests differential correlation between the 2 groups based on Fisher's z-test. The R program DiffCorr contains a set of functions for identifying differential correlations in a correlation network derived from large-scale omics data. Functions in DiffCorr package can be divided into 3 main categories: (1) module detection, (2) visualization of eigen-molecule networks; and (3) export of the results of testing based on Fisher's z-test.

1) Component Differences with DCETool: DiffCor gives an easy computation of Differential Co-Expression Analysis for R. The main difference between DCETool and DiffCor is that interaction is close to zero for the second one. In fact DiffCor provides visualizations over the analysis , but it does not provide a preprocessing of data for it , for which is supposed to be done by the user, and the plots are static. This brings to the conclusion that when doing an analysis with DiffCor you can just be ok with it or redo from zero. Instead with DCETool you are able to manipulate the data in a manner of seconds and recompute the analysis having at the same time the visibility on everything that is happening.

C. *iSEE*

iSEE (Interactive SummarizedExperiment Explorer) is a tool that provides a general visual interface for exploring data. iSEE is implemented in R using the Shiny framework [7]. A feature in common with DCETool is that iSEE has the ability to dynamically transmit information between panels. This feature , as in DCETool , facilitates exploration of the relationships between different aspects of the data. iSEE automatically memorises the exact R code that was used to generate every plot. This code is fully accessible to users at any time during the run-time of the interface. By integrating the code reported by iSEE into their own scripts, users can easily reproduce the results of any exploratory analysis. This is probably one of the most interesting features, and it could be added in future for the DCE analysis on DCETool too, since the function is completely written by hand. As in DCETool a PCA is available , but with difference that a t-SNE is available too.

1) Features differences with DCETool: The first feature that differ is that in iSEE there is the possibility for the user to order the plots as he wants. In fact this can be useful when a lot of plots are presented, to let the user compare two plots that could have been distant from each other, close to each other. Another feature is the color freedom. In fact as for DCETool the inital colors are taken by literature or by more famous packages such as ggplot2. But because iSEE let more different data be studied, color significance can change and thus a color freedom is provided to the user. In DCETool this is not needed since the aim of this tool is specific , but was worth to mention.

VII. Conclusion

While it is true that this tool let the user customize the set of genes in order to compute differential expression analysis, what it miss here is an additional visualization over the resulting network. This was not the main goal of the project, but it would definitely give an additional help , giving the user an all in one tool.

A. *Future Work*

As a future work, network visualization and interaction could be applied. To give the user an idea of the clusters of genes resulted from differential co-expression network analysis. Other than that, more dimensionality reduction could be added such as Multi Dimensional Scaling and t-SNE.

References

[1] Gene Expression Omnibus, *NCBI*.

[2] Best MG, Wurdinger T , RNA-seq of tumor-educated platelets enables blood-based pan-cancer, multiclass and molecular pathway cancer diagnostics, Amsterdam , Netherlands: Apr 21, 2015.

[3] Robinson and Oshlack, A scaling normalization method for differential expression analysis of RNA-seq data 2010.

[4] Best Et al.RNA-Seq of Tumor-Educated Platelets Enables Blood-Based Pan-Cancer , Amsterdam , Netherlands: Apr 21, 2015.

[5] Kevin Rue-Albrecht Et al., iSEE: Interactive SummarizedExperiment Explorer, 2018 Jun 14 .

[6] Atsushi Fukushima, DiffCorr: an R package to analyze and visualize differential correlations in biological networks , 2012 Dec 13.

[7] Federico Marini Harald Binder, pcaExplorer: an R/Bioconductor package for interacting with RNA-seq principal components, 2019. https://pubmed.ncbi.nlm.nih.gov/23246976/