Differential gene expression analysis is widely used to study changes in gene expression profiles between two or more groups of samples (e.g., physiological versus pathological conditions, pre-treatment versus post-treatment, and infected versus non-infected tissues). This protocol aims to identify gene expression changes in a pre-selected set of genes associated with severe acute respiratory syndrome coronavirus 2 viral infection and host cell antiviral response, as well as subsequent gene expression association with phenotypic features using samples deposited in public repositories.
Protocol

Analysis workflow of publicly available RNA-sequencing datasets

Pablo Sanchis,1,2,3,4,* Rosario Lavignolle,1,2,3,* Mercedes Abbate,1,2 Sofía Lage-Vickers,1,2 Elba Vazquez,1,2 Javier Cotignola,1,2 Juan Bizzotto,1,2 and Geraldine Gueron1,2,5,*

1Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, Departamento de Química Biológica, Buenos Aires C1428EGA, Argentina
2CONICET - Universidad de Buenos Aires, Instituto de Química Biológica de la Facultad de Ciencias Exactas y Naturales (IQUIBICEN), Buenos Aires C1428EGA, Argentina
3These authors contributed equally
4Technical contact
5Lead contact
*Correspondence: pabloasanchis@gmail.com (P.S.), rosario.lavignolle@gmail.com (R.L.), ggueron@iquibicen.fcen.uba.ar (G.G.)

https://doi.org/10.1016/j.xpro.2021.100478

SUMMARY

Differential gene expression analysis is widely used to study changes in gene expression profiles between two or more groups of samples (e.g., physiological versus pathological conditions, pre-treatment versus post-treatment, and infected versus non-infected tissues). This protocol aims to identify gene expression changes in a pre-selected set of genes associated with severe acute respiratory syndrome coronavirus 2 viral infection and host cell antiviral response, as well as subsequent gene expression association with phenotypic features using samples deposited in public repositories.

For complete details on the use and outcome of this informatics analysis, please refer to Bizzotto et al. (2020).

BEFORE YOU BEGIN

Download R and RStudio

هج Timing: 1 h

1. R is a free software environment for statistical computing and graphics. It runs on UNIX, Windows and MacOS.
   a. To download and install R go to https://www.r-project.org/ (R Core Team, 2013). The current pipeline was performed using R version 3.6.2.
2. RStudio is an integrated development environment (IDE) for R. It allows to easily execute the R codes, plot graphics, and manage the workspace in a multipanel interphase.
   a. To download and install RStudio go to https://rstudio.com/products/rstudio/ (RStudio Team, 2020).

Pause point: Since this protocol follows a bioinformatic pipeline exclusively, by saving the executed steps the protocol can be paused at any time.

Download required packages in RStudio

هج Timing: 1 h
3. Users must first download the required packages (listed in the key resources table). They can be downloaded through Bioconductor, which provides tools for the analysis and comprehension of high-throughput genomic data. BiocManager::install() is the recommended command to install packages (for detailed information on why BiocManager::install() is preferred to the standard R packages installation please read https://www.bioconductor.org/install/#why-biocmanagerinstall):

   a. Install the packages needed for the analysis. See troubleshooting 1:

   ```r
   if (!requireNamespace("BiocManager", quietly = TRUE))
     install.packages("BiocManager")
   BiocManager::install(c("DESeq2", "GEOquery", "canvasXpress",
                         "ggplot2", "clinfun", "Ggally", "factoextra"))
   ```

   b. Once all packages are installed, they need to be loaded:

   ```r
   library(DESeq2)
   library(GEOquery)
   library(canvasXpress)
   library(ggplot2)
   library(clinfun)
   library(Ggally)
   library(factoextra)
   ```

4. When using datasets from public repositories, the key step is to identify a dataset (or datasets) that comply with the eligibility criteria and that contains the sample information required for the analysis.

   a. We suggest browsing Gene Expression Omnibus (GEO: https://www.ncbi.nlm.nih.gov/gds, (Barrett et al., 2012)) and ArrayExpress (https://www.ebi.ac.uk/arrayexpress/,(Atha et al., 2019)) repositories because they gather multiple high-throughput genomics datasets. However, there are several public repositories that might be more suitable for other types of studies. These repositories allow to download raw sequencing data (.fastq sequencing files) and/or pre-processed files (tab-delimited.txt files containing matrices with sequence read counts after trimming and alignment to the reference genome). The pre-processed files may contain a raw-counts matrix (non-normalized) or a normalized counts matrix (see below for more details). Sample information is also available to download. Finally, the platform used, and pre-processing algorithm (when data are pre-processed) are specified. We strongly recommend researchers to thoroughly evaluate the type of data submitted, study design, number of samples and any other relevant information that might help to analyze the samples and draw statistically valid conclusions. Troubleshooting 2.

   b. Our eligibility criteria for (Bizzotto et al., 2020) was: (i) publicly available transcriptome data; ii) detailed sample/patient information; (iii) detailed protocol information; (iv) ≥ 60 samples. We selected the GSE152075 dataset from GEO which contained RNA-seq data from 430 SARS-CoV-2 positive and 54 negative patients (Lieberman et al., 2020). We downloaded the gene expression aligned data matrix (tab-delimited .txt file with reads pseudo-aligned to the human reference transcriptome). Clinico-pathological information included age, gender, and viral load (expressed as cycle threshold (Ct) by RT-qPCR for the N1 viral gene at time of diagnosis). The interpretation for viral load was as follows: the lower the Ct, the higher the viral load. This phenotypic data can be downloaded directly in RStudio as explained in the “RNA-seq data organization and counts normalization” section.
KEY RESOURCES TABLE

| REAGENT or RESOURCE | SOURCE | IDENTIFIER |
|----------------------|--------|------------|
| Deposited data       |        |            |
| In vivo antiviral host response to SARS-CoV-2 by viral load, sex, and age [dataset I] | NCBI Gene Expression Omnibus | GSE152075 |
| Raw_code_Sanchis_et_al.R | GitHub repository | https://github.com/lab-inflamacionycancer/STAR-protocol-Sanchis.et.al/blob/main/raw_code_Sanchis_et_al.R |

Software and algorithms

| R software | (R Core Team, 2013) | https://www.r-project.org/ |
| RStudio | (RStudio Team, 2020) | https://rstudio.com/ |
| DEseq2 v1.28.1 package | (Love et al., 2014) | https://bioconductor.org/packages/release/bioc/html/DESeq2.html |
| ggplot2 package | (Wickham, 2016) | https://ggplot2.tidyverse.org/ |
| GGally package | (Schloerke et al., 2020) | https://github.com/ggobi/ggally/ |
| canvasXpress package | (Neuhaus and Brett, 2020) | http://www.canvasxpress.org/ |
| factoextra package | (Kassambara and Mundt, 2020) | https://rpkgs.datanovia.com/factoextra/index.html |
| clinfun package | (Seshan, 2018) | https://github.com/veseshan/clinfun |
| GEOquery package | (Davis and Meltzer, 2007) | https://www.bioconductor.org/packages/release/bioc/html/GEOquery.html |

Note: detailed information on the usage of the different packages may be found in the links provided in the identifier column.

MATERIALS AND EQUIPMENT

For this bioinformatics analysis we used a laptop with an Intel Core i5 8th generation processor, 8 GB RAM memory and Windows 10. No high-performance computing clusters were needed for the analysis of the data. Internet connection is required for downloading R packages and data matrices.

STEP-BY-STEP METHOD DETAILS

The flow chart for data processing is included in Figure 1.

![Flow chart for data processing](Figure 1)
Download and prepare the data matrix for analysis

© Timing: 2 h

You can download the experiment information and clinical data directly from GEO using the GEO-query package:

1. The series matrix file is a text file that includes a tab-delimited value-matrix for each sample containing the phenotypic/clinical and experimental data of a given dataset. In the GEO webtool, there is a hyperlink to the series matrix, called “Series Matrix File(s)”. To download the series matrix file directly to the R environment use the getGEO command:

   ```r
   data <- getGEO(GEO = c("GSE152075"))
   # replace the text between [] with the GSE of your choice and remove the []
   # print the first five rows of the matrix to see matrix information
   head(data)
   # output (do not run this piece of script)
   > head(data)
   GSM152075_series_matrix.txt.gz
   ExpressionSet (storageMode: lockedEnvironment)
   assayData: 0 features, 488 samples
   elementNames: exprs
   pData: none
   phenoData
   sampleNames: GSM4602241 GSM4602242 ... GSM4602725 (484 total)
   varLabels: title geo_accession ... sequencing_batch:ch1 (43 total)
   varMetadata: labelDescription
   featureData: none
   experimentData: use `experimentData(object)`
   pubMedIds: 32898168
   Annotation: GPL18573
   ```

2. You may now extract the phenotypic/clinical data matrix from the series matrix:

   ```r
   clndata <- data[["GSM152075_series_matrix.txt.gz"]][phenoData$data]
   # replace the GSE with the one of your choice. Do not remove the [] in this line.
   # print the first five rows of the matrix to see the information included in columns of interest
   head(clndata[,c(1,2,8,40,39,42)])
   # output (do not run this piece of script)
   > head(clndata[,c(1,2,8,40,39,42)])
   title geo_accession source_name_ch1 gender:ch1 age:ch1 sars-cov-2
   positivity:ch1
   GSM4602241 POS_001 GSM4602241 Nasopharyngeal Swab H 64 pos
   GSM4602242 POS_002 GSM4602242 Nasopharyngeal Swab H 30 pos
   GSM4602243 POS_003 GSM4602243 Nasopharyngeal Swab H 47 pos
   GSM4602244 POS_004 GSM4602244 Nasopharyngeal Swab H 67 pos
   GSM4602245 POS_005 GSM4602245 Nasopharyngeal Swab H 62 pos
   GSM4602246 POS_006 GSM4602246 Nasopharyngeal Swab H 52 pos
   ```

3. Download and save on your computer the raw-counts matrix from GEO website. This matrix is a tab-delimited txt. file containing the counts for every gene aligned from a RNA-seq experiment. After downloading it, load the matrix into RStudio:

   ```r
   raw_counts <-
   ```
Note: The sequencing data for GSE152075 was submitted as raw-counts in a separate file from the experimental and clinical data. (https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE152075; GSE152075_raw_counts_GEO.txt.gz); therefore, it was downloaded separately. For datasets with pre-processed/normalized data, the counts matrix might be included in the series matrix file downloaded in step 1. Troubleshooting 3.

RNA-seq data organization and count normalization

© Timing: 1 h

Before performing differential gene expression analysis, it is required to normalize the read counts if the raw-counts matrix was downloaded. This normalization step allows to compare gene expression (read counts) among samples (Evans et al., 2018). It is also recommended to correct for batch effect if multiple batches of experiments were performed. If the user’s dataset is already normalized, then go directly to step 6.

4. Gene expression normalization:
   a. Before sample normalization, data should be converted and organized to the format required for further analysis (data format and organization might vary for different packages). Troubleshooting 4.

   ```r
   raw_counts <- as.matrix(raw_counts)
   rownames(clindata) <- clindata$[title]
   # replace the rownames of clindata (sampleID) with the same sample name [title] of [raw_counts]. This will help to match sample names in both matrixes.
   Remove the []
   all(rownames(clindata) %in% colnames(raw_counts))
   # the outcome should be TRUE
   all(colnames(raw_counts) %in% rownames(clindata))
   # the outcome should be TRUE
   ```

   b. Make sure that the grouping variables are factors. We also changed the original names of the columns containing the relevant variables to make them shorter and easier to work with.
c. Merge the read counts and clinical data matrixes into a DESeqDataSet object using the DESeq2 package:

```r
dds <- DESeqDataSetFromMatrix(countData = raw_counts,
    colData = clindata,
    design = ~ [positivity + batch])
```

# the design argument is a formula that expresses how the counts for each gene depend on the variables in colData. The ‘design’ function sets the important variables to take into consideration when performing expression analyses but they are not taken into account for the normalization step. Replace the variables between [] with the variables of your choice and remove the [].

# print first five rows of the merged data to check how it is organized
head(dds)

# output (do not run this piece of script)
> head(dds)
class: DESeqDataSet
dim: 6 484
metadata(1): version
assays(1): counts
rownames(6): A1BG A1CP ... A2MPL A3GALT2
rowData names(0):
colnames(484): POS_001 POS_002 ... NEG_063 NEG_065
colData names(44): title geo_accession ... batch sizeFactor

d. Normalization by estimation of size factor:

```r
dds <- estimateSizeFactors(dds)
```

# print first five rows of the normalized data
head(dds)

# output (do not run this piece of script)
> head(dds)
class: DESeqDataSet
dim: 6 484
metadata(1): version
assays(1): counts
rownames(6): A1BG A1CP ... A2MPL A3GALT2
rowData names(0):
colnames(484): POS_001 POS_002 ... NEG_063 NEG_065
colData names(45): title geo_accession ... viral_load sizeFactor

e. Create a new table with the normalized read counts (gene expression) for all genes:
Note: There are alternative methods to normalize and extract the counts such as rlog and vst that would fit this analysis (Love et al., 2014). However, this STAR Protocol replicates the bio-informatics analysis described in Bizzotto et al. (Bizzotto et al., 2020) where the command “normalized=TRUE” was used.

5. For our study, we converted the continuous variables (e.g., age, viral load) into factor/strata variables (e.g., 10-year age ranges, low/medium/high viral load). The code below shows an example on how to stratify the viral load and age (after duplicating the original variable in order to not overwrite the original data).

```r
#stratify viral load
{  
  clindata$viral_load <- clindata$ct
  clindata$viral_load[clindata$viral_load == "N/A"] <- "Negative"
  clindata$viral_load[clindata$viral_load > 24 & clindata$viral_load != "Unknown" & clindata$viral_load != "Negative"] <- "LOW"
  clindata$viral_load[clindata$viral_load <= 24 & clindata$viral_load >= 19] <- "MEDIUM"
  clindata$viral_load[clindata$viral_load < 19] <- "HIGH"
  clindata$viral_load <- as.factor(clindata$viral_load)
  clindata$viral_load <- factor(clindata$viral_load, levels = c("Negative", "LOW", "MEDIUM", "HIGH", "Unknown"))
  clindata$positivity <- factor(clindata$positivity, levels = c("HEALTHY", "COVID19"))
}

#stratify age
{  
  clindata$age_cat <- clindata$'age:chi'
  clindata$age_cat[clindata$'age:chi' < 30] = "< 30"
  clindata$age_cat[clindata$'age:chi' >= 30 & clindata$'age:chi' < 40] = "30s"
  clindata$age_cat[clindata$'age:chi' >= 40 & clindata$'age:chi'< 50] = "40s"
  clindata$age_cat[clindata$'age:chi' >= 50] = "50s"
  clindata$age_cat <- as.factor(clindata$'age:chi')
}
```
Note: This is an optional step depending on your variables and analysis.

Differential gene expression analysis across different strata

**Timing:** 1 day

*Note:* As mentioned in Bizzotto et al. (Bizzotto et al., 2020), some samples were removed from the analysis since they were considered to have low quality read sequencing (>70% genes with 0 counts). Because this might not apply to all protocols, we did not include the code for this filtering in the main manuscript, but it was included in the troubleshooting 5. Therefore, the outcomes for this protocol might slightly differ from the outcome published on Bizzotto et al. (Bizzotto et al., 2020), but this omission does not change the results and interpretation of the study.

This step aims to compare gene expression across different strata. Below we show, as an example, the differential MX1 expression analysis for SARS-CoV-2 positive vs. negative patients.

6. As described in our publication, we selected specific genes that could potentially be linked to SARS-CoV-2 infection. For this section, we selected MX1 as an example to show. Plot MX1 expression for SARS-CoV-2 positive and negative patients (Figure 2A.i) using the following code:

```r
clindata$age_cat[clindata$'age:ch1' <= 60 & clindata$'age:ch1' < 70] = '50s'
clindata$age_cat[clindata$'age:ch1' >= 70] = '70+
clindata$age_cat[clindata$'age:ch1' == "Unknown"] = NA

Note: This is an optional step depending on your variables and analysis.

Differential gene expression analysis across different strata

**Timing:** 1 day

*Note:* As mentioned in Bizzotto et al. (Bizzotto et al., 2020), some samples were removed from the analysis since they were considered to have low quality read sequencing (>70% genes with 0 counts). Because this might not apply to all protocols, we did not include the code for this filtering in the main manuscript, but it was included in the troubleshooting 5. Therefore, the outcomes for this protocol might slightly differ from the outcome published on Bizzotto et al. (Bizzotto et al., 2020), but this omission does not change the results and interpretation of the study.

This step aims to compare gene expression across different strata. Below we show, as an example, the differential MX1 expression analysis for SARS-CoV-2 positive vs. negative patients.

6. As described in our publication, we selected specific genes that could potentially be linked to SARS-CoV-2 infection. For this section, we selected MX1 as an example to show. Plot MX1 expression for SARS-CoV-2 positive and negative patients (Figure 2A.i) using the following code:

```r
MX1 <- ggplot(NULL, aes(x=clindata$positivity, y=log2(t(norm_counts["MX1",]+1)))) + geom_jitter(aes(shape=clindata$positivity, color=clindata$positivity), size=3)+ xlab(NULL) + ylab("MX1 expression \n log2 (norm counts +1") + theme(legend.position = "bottom") + theme_bw() + theme(axis.text = element_text(size = 15), axis.title = element_text(size = 15), plot.title =element_text(size = 25), legend.position = ‘none’) + stat_summary(fun=mean, geom="point", shape='._', size=14, colour=c(‘#b53d35’, ‘#066e70’))
MX1
```
CRITICAL: For the y argument you must specify a data frame containing the samples in the rows and the variables (genes) in the columns; therefore, we used the t() argument to transpose the data frame "norm_counts". Gene expression is expressed as the log2(counts); therefore, and because some samples have 0 counts, it is necessary to add 1 count to all genes for all samples to avoid errors due to log2(0).

7. Export the plot in high quality (complying with most publication standards):

```r
ggsave(filename="[Name of the file.png]", plot= [name of the plot in RStudio i.e.: MX1], device="png", dpi=600, height=10, width=14, units="in")
#replace the text between [] with the filename to be saved. Remove the [].
```

8. Perform a Wilcoxon test to assess the statistical significance of MX1 expression differences

```r
MX1stat <- wilcox.test(norm_counts["MX1"], ~ clindata$positivity, paired = FALSE)
MX1stat
```

Note: Please note that Limma and DESeq2 would be more appropriate statistical packages to run when analyzing whole transcriptomes. In (Bizzotto et al., 2020), we analyzed a pre-selected set of genes; therefore, the Wilcoxon test can be used to analyze mean differences between groups.

9. In addition, we performed the Jonckheere-Terpstra (Arif et al., 2015) trend test to evaluate gene expression trends across ordered strata (e.g., age stratified by 10-year ranges). As an example, here we show
the trend test for MX1 expression across age categories in SARS-CoV-2 negative and positive patients (Figure 2B, left and right panels, respectively). The continuous age variable was stratified in 10-year ranges as described in “RNA-seq data organization and counts normalization - step 5” section.

```
{ 
  p_trend_age <- jonckheere.test(x = log2(t(norm_counts[,] + 1)))
  clindata$positivity == "COVID19",
  g = factor(clindata$age_cat
  [clindata$positivity == "COVID19"],
  ordered = TRUE),
  alternative = "decreasing",
  nperm = 500)
  p_trend_age
  
注：在这段代码中，列“clindata$age_cat”包含按10年年龄段划分的年龄。

注：要对SARS-CoV-2阴性患者执行此测试，然后将“COVID19”替换为“HEALTHY”。

注：参数alternative可以是“two.sided”，“increasing”或“decreasing”。根据您的假设检验选择最佳参数。

Correlation analysis

© Timing: 1 day

The aim of this step is to analyze gene expression correlation in the different categorical variables. We also provide the code to perform a pairwise gene expression correlation including the viral load as a third variable plotted in a color scale.

10. Spearman correlation analysis between gene expression levels:
   a. Calculate the Spearman coefficients and plot all pairwise correlations. The example below shows the correlation analysis for four genes (MX1, MX2, ACE2 and TMPRSS2) in SARS-CoV-2 positive and negative patients (Figure 3A):

```
{
  pairwise_corr <- ggpairs(as.data.frame(log2(t(norm_counts+1)))),
  columns = c("MX1", "MX2", "ACE2", "TMPRSS2"),
  upper = list(continuous = wrap('cor',
  method = "spearman", size = 3),
  combo = "box_no_facet",
  discrete = "count",
  na = "na"),
  ggrepel::aes(colour=clindata$positivity,
  shape=clindata$positivity, alpha = 0.01))
  pairwise_corr <- pairwise_corr + theme(strip.placement = "outside",
  text = element_text(size = 9, face = "bold") +
  ggtitle("Gene correlation") +
  theme(plot.title = element_text(size = 15,
  hjust = 0.5)) +
  ylab("log2(counts +1") +
  xlab("log2 (counts +1")
  pairwise_corr
  
注：Spearman相关分析用于基因表达水平之间。
Any continuous variable (e.g. viral load expressed as Ct) could be included in the analysis of correlation with gene expression (Bizzotto et al., 2020).

11. Plot gene expression correlation between two genes and include viral load (expressed as Ct) in a color scale (Figure 3B.i) and calculate the Spearman correlation coefficient (Figure 3B.ii).

**Figure 3. Correlation analysis**

(A) Pairwise Spearman correlation matrix analysis between all genes of interest (MX1, MX2, ACE2 and TMPRSS2), considering different infection status (all patients together (Corr.), SARS-CoV-2 negative (HEALTHY), or SARS-CoV-2 positive patients (COVID19)). Alternative hypothesis: \( \rho \neq 0 \), which indicates that there is a lineal correlation between the variables under study. Statistical significance: *\( p < 0.05 \); **\( p < 0.01 \); ***\( p < 0.001 \).

(B) Scatter plot of MX1 and MX2 gene expression levels, color-coded by viral load (i) and Spearman correlation output in RStudio (ii). Figures 3A and 3B.i were adapted with permission from Bizzotto et al., 2020.

**Note:** Any continuous variable (e.g. viral load expressed as Ct) could be included in the analysis of correlation with gene expression (Bizzotto et al., 2020).

11. Plot gene expression correlation between two genes and include viral load (expressed as Ct) in a color scale (Figure 3B.i) and calculate the Spearman correlation coefficient (Figure 3B.ii)

```r
MX1_MX2 <- ggplot(NULL, aes(x = log2(t(norm_counts['MX1'],+1)[which(clindata$positivity=="COVID19" & clindata$ct != "Unknown")]),
```
Patient segregation based on gene expression

**Figure 4. Patient segregation based on gene expression**

(A) Principal component analysis (PCA) output in RStudio, showing the first two principal components explaining most part of the variance among samples.

(B) PCA biplot of gene expression data showing a rough segregation of SARS-CoV-2 positive and negative samples.

(C) 3D scatter plot for the expression of three genes of interest (BSG, MX1 and ACE2). Samples are colored by SARS-CoV-2 status (positive or negative). Ellipsoids represent the 95% confidence interval. Figures 4B and 4C were adapted with permission from Bizzotto et al., 2020.

\[
y = \log_2(t(n_{\text{norm counts}}["MX2",]+1)) \text{[which(clindata$positivity=="COVID19" & clindata$ct == "Unknown")],}
\]
\[
color = \text{as.integer(clindata$ct[which(clindata$positivity=="COVID19" & clindata$ct == "Unknown")])}) +
geom_point(size = 4, na.rm = TRUE) +
scale_color_gradientn(colours=c("red","white","blue"), name = "Viral load (ct)") +
ylab("MX2 expression RNA-seq \n log2 (norm counts +1)") +
xlab("MX1 expression RNA-seq \n log2 (norm counts +1)") +
theme(legend.position = "bottom") +
theme_low() +
theme(axis.text = element_text(size = 15),
axis.title = element_text(size = 15),
plot.title = element_text(size = 25))
\]

MX1_MX2stat <- cor.test(norm_counts["MX1",]
[which(clindata$positivity=="COVID19")),
norm_counts["MX2",]
[which(clindata$positivity=="COVID19")),
method = "spearman")
MX1_MX2stat
Principal-component analysis (PCA) is a dimensionality reduction technique used to increase the interpretability of a given dataset, minimizing information loss and maximizing variance.

12. Principal component analysis based on disease status:
   a. Calculate the principal component on the log2 transformed gene expression data. The output is a table as shown in Figure 4A, containing the weight of each gene in the variance of the samples for Principal Component 1 (PC1; the largest component of variance in the data set) and Principal Component 2 (PC2; the second most important component influencing the variance):

```r
res.pca <- prcomp(t(log2(norm_counts[c("gene1","gene2","geneN"),]+1)),
                 scale = TRUE)
#replace gene1, gene2, geneN by the list of genes of your interest
res.pca
```

Note: For PCA we only considered expression of the candidate genes, but you might also include any independent variable you consider to affect the variability among samples.

b. Plot PC1 vs. PC2 (Figure 4B):

```r
p<- fviz_pca_biplot(res.pca, col.ind = clinda$data$positivity,
                      geom = "point",
                      addEllipses = TRUE,
                      palette = c("#F8766D", '#00BFC4'),
                      title = "Principal Component Analysis")
p
#replace the variable between [] for any variable of your interest, and remove the []
```

13. 3D graphs showing expression for the selected genes can be plotted as follows. The output is shown in Figure 4C:

```r
data=t(log2(norm_counts[c("MX1","ACE2","BSG"),]+1)),
varAnnot=as.data.frame(clinda$data$positivity,
row.names=rownames(clinda)),
axisTickScaleFontFactor=0.6,
axisTitleScaleFontFactor=0.6,
ellipseBy="clinda$data$positivity",
colorBy="clinda$data$positivity",
colorKey=list("clinda$data$positivity"=list("COVID19"=#F8766D,
"HEALTHY"=#00BFC4)),
graphType="Scatter3D",
title="3D scatter plot",
xAxis=list("ACE2"),
yAxis=list("BSG"),
zAxis=list("MX1"),
showLoessFit = FALSE)
```

**EXPECTED OUTCOMES**

Plots in Figure 2 depict dot plots of gene expression separating patients according to different phenotypic data, such as disease status (Figure 2A) and age (Figure 2B), and the output of the
statistical tests. The color code for these analyses is: red = SARS-CoV-2 negative patients, and blue = SARS-CoV-2 positive patients.

In order to evaluate the association between different parameters, such as gene expression and viral load, we performed pairwise correlation analysis explained in the “Correlation Analysis” section. The outcome is a scatter plot and the Spearman coefficient (rho) with the associated P-value for each comparison. Figure 3A shows the pairwise correlation plots and statistics. Figure 3B shows the scatter plot for the correlation between two genes and viral load as color-coded dots.

Finally, it is possible to visualize patient segregation by PCA using the selected genes (Bizzotto et al., 2020) as independent variables (Figures 4A and 4B). 3D scatter plots can be done. These plots depict gene expression and 95% confidence ellipsoids (Figure 4C).

QUANTIFICATION AND STATISTICAL ANALYSIS
Eligibility criteria, statistical tests and software used for this protocol are properly described in the “before you begin” and “step-by-step methods details” sections.

LIMITATIONS
This protocol relies on the accuracy of the clinical records submitted to the public repository by the original authors. For some datasets, RNA-sequencing raw counts data may not be available, and data could have been pre-processed by the original authors, thus limiting the decision making of which sequence quality will comply with the own standards, usage of different alignment software and algorithms, reference genome, etc. However, there are still some quality controls that might be done. As we mentioned in (Bizzotto et al., 2020), we detected that some samples had 0 counts for most genes (>70% genes), suggesting a very low quality RNA isolation and/or sequencing. Therefore, we were able to remove them from the analysis. We strongly suggest performing all possible quality controls before analyzing the data.

TROUBLESHOOTING
It is important that when working in R, and following this protocol, take into consideration the critical statements and notes for each step provided in the method details.

Problem 1
When running a piece of code, the output is not the expected (i.e., error message in the console or an unexpected feature in the plot).

Potential solution
For the proper execution of the piece of code, the syntax must be accurate, so it is important to check whether there are no syntax mistakes, such as:

- Forgotten comma.
- Unclosed bracket, parenthesis, or quotes.
- Misspelled function or filenames.

Problem 2
There is no count matrix available. The only accessible information are the FASTQ files thus requiring additional processing (e.g., alignment to reference transcriptome) which is not described in the current protocol.

Potential solution
Since there are already publications explaining this matter further, we suggest visiting the manuals which describe how to process and align sequencing data before performing expression analyses.
using HISAT2 (Kim et al., 2019), TopHat2 (Kim et al., 2013), STAR (Dobin et al., 2013) and Salmon (Patro et al., 2017).

**Problem 3**
The raw counts for a specific dataset are available as a separate file, but it is not clear for the user whether they are raw counts or pre-processed data.

**Potential solution**
It is important to verify that the data represent raw counts. This can be checked by reading the GEO submission description and the Methods section within the corresponding citation.

**Problem 4**
The phenotypic and count matrices do not contain the same list of samples.

**Potential solution**
It is important to verify that the phenotypic and counts matrices contain the same list of samples in order to merge phenotypic and gene expression data. If not, discard incomplete samples. This can be achieved using the following command, which creates a table only with the samples included in both matrices:

```r
common_names = intersect(rownames(clidata), colnames(raw_counts))
clidata = clidata[rownames(clidata) %in% common_names,]
raw_counts = raw_counts[,colnames(raw_counts) %in% common_names]
```

**Problem 5**
There is a great number of genes with 0 counts.

**Potential solution**
Samples considered to have low quality read sequencing should be removed from the analysis since they may introduce noise to the results. The following code is an example of how this can be achieved in RStudio when samples in the dataset have >70% genes with 0 counts:

```r
#see how many genes per sample have 0 reads
zero <- colSums(norm_counts == 0)/nrow(norm_counts)
hist(zero, breaks = 50)
abline(v = 0.7, col="red")
summary(zero)
#select only those samples with less than 70% of genes with zero
good_samples <- zero <0.7
norm_counts <- norm_counts[,good_samples]
```

**RESOURCE AVAILABILITY**

**Lead contact**
Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Geraldine Gueron, ggueron@iquibicen.fcen.uba.ar.

**Materials availability**
This study did not generate new unique reagents.

**Data and code availability**
The code generated during this study is available at GitHub repository accessible using the following link: https://github.com/lab-inflamacionycancer/STAR-protocol-Sanchis.et.al.
ACKNOWLEDGMENTS
This work was supported by grants from AGENCIA-PICT-2016-0056 (Argentina) and AGENCIA-PICT-RAICES-2018-02639 (Argentina).

AUTHOR CONTRIBUTIONS
Conceptualization, P.S., R.L., M.A., S.L.V., E.V., J.C., J.B., and G.G.; formal analysis, P.S., R.L., M.A., and J.B.; investigation, P.S., R.L., M.A., S.L.V., E.V., J.C., J.B., and G.G.; resources, J.C. and G.G.; data curation, P.S., R.L., M.A. S.L.V., E.V., J.C., J.B., and G.G.; writing – review and editing, P.S., R.L., S.L.V., E.V., J.C., J.B., and G.G.; funding, E.V., J.C., and G.G.; all authors have read and agreed to the published version of the manuscript.

DECLARATION OF INTERESTS
The authors declare no competing interests.

REFERENCES
Arif, A., Abdur, R., Afaq Ahmed, S., Maliha, N., Saba, W., and Waseem, A. (2015). Non-parametric test for ordered medians: the jonckheere terpstra test. https://doi.org/10.6000/1929-6029.2015.04.02.8.

Athar, A., Fulgrabe, A., George, N., Iqbal, H., Huerta, L., Ali, A., Snow, C., Fonseca, N.A., Petryszak, R., Papatheodorou, I., et al. (2019). ArrayExpress update – from bulk to single-cell expression data. Nucleic Acids Res. 47, D711–D715, https://doi.org/10.1093/nar/gky964.

Barrett, T., Wilhite, S.E., Ledoux, P., Evangelista, C., Kim, I.F., Tomashevsky, M., Marshall, K.A., Phillippy, K.H., Sherman, P.M., Holko, M., et al. (2012). NCBI GEO: archive for functional genomics data sets—update. Nucleic Acids Res. 41, D991–D995, https://doi.org/10.1093/nar/gks1193.

Bizzotto, J., Sanchis, P., Abbate, M., Lage-Vickers, S., Lavignolle, R., Toro, A., Olszewicki, S., Sabater, A., Cascardo, F., Vazquez, E., Cotignola, J., and Gueron, G. (2020). SARS-CoV-2 infection boosts MX1 antiviral effector in COVID-19 patients. iScience 23, 101585, https://doi.org/10.1016/j.isci.2020.101585.

Davis, S., and Meltzer, P.S. (2007). GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor. Bioinformatics 23, 1846–1847, https://doi.org/10.1093/bioinformatics/btm254.

Dobin, A., Davis, C.A., Schlesinger, F., Drenkow, J., Zaleski, C., Jha, S., Batut, P., Chaisson, M., and Gingeras, T.R. (2013). STAR: ultrafast universal RNA-seq aligner. Bioinformatics 29, 15–21, https://doi.org/10.1093/bioinformatics/bts355.

Evens, C., Hardin, J., and Stoebel, D.M. (2018). Selecting between-sample RNA-Seq normalization methods from the perspective of their assumptions 17. https://doi.org/10.1093/bib/bby2008.

Kassambara, A., and Mundt, F. (2020). factoextra: Extract and Visualize the Results of Multivariate Data Analyses (R package).

Kim, D., Paggi, J.M., Park, C., Bennett, C., and Salzberg, S.L. (2019). Graph-based genome alignment and genotyping with HISAT2 and HISAT-genotype. Nat. Biotechnol. 37, 907–915, https://doi.org/10.1038/s41587-019-0201-4.

Kim, D., Pertea, G., Trapnell, C., Pimentel, H., Kelley, R., and Salzberg, S.L. (2013). TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions. Genome Biol. 14, R36, https://doi.org/10.1186/gb-2013-14-4-r36.

Liebermann, N.A.P., Peddu, V., Xie, H., Shrestha, L., Huang, M.-L., Mears, M.C., Cajimat, M.N., Bente, D.A., Shi, P.-Y., Bovier, F., Roychoudhury, P., Jerome, K.R., Moscona, A., Porotto, M., and Greninger, A.L. (2020). In vivo antiviral host transcriptional response to SARS-CoV-2 by viral load, sex, and age. PLoS Biol. 18, e3000849, https://doi.org/10.1371/journal.pbio.3000849.

Love, M.I., Huber, W., and Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol. 15, 520, https://doi.org/10.1186/s13059-014-0550-8.

Neuhaus, I., and Brett, C. (2020). canvasXpress: Visualization Package for CanvasXpress in R (R package).

Patro, R., Duggal, G., Love, M.I., Irizarry, R.A., and Kingsford, C. (2017). Salmon provides fast and bias-aware quantification of transcript expression. Nat. Methods 14, 417–419, https://doi.org/10.1038/nmeth.4197.

R Core Team (2013). R: A Language and Environment for Statistical Computing (R Foundation for Statistical Computing).

RStudio Team (2020). RStudio: Integrated Development for R. RStudio (Boston, MA: PBC), http://www.rstudio.com/.

Seshan, V. (2018). clinfun: Clinical Trial Design and Data Analysis Functions (R package).

Wickham, H. (2016). ggplot2: Elegant Graphics for Data Analysis (Springer).

Schloerke, B., Cook, D., Larmarange, J., Briatte, F., Marbach, M., Thoen, E., Elberg, A., and Crowley, J. (2020). Ggally: Extension to ‘ggplot2’. R package.