Anàlisis de dades òmiques (ADO) – Informe PAC2

TAULA DE CONTINGUTS

ABSTRACT

La transcriptomica permet estudiar com responen les cèl·lules davant diferents condicions fisiopatològiques mitjançant la quantificació del RNA. Aquest tipus d'estudis són clau per poder entendre els mecanismes implicats en processos fisiològics, patològics i infecciosos (1).

Aquest treball descriu l'anàlisi d'un conjunt de dades d'RNAseq provinents de mostres de sang de pacients amb COVID-19, infecció bacteriana o sans obtingudes en l'estudi GSE161731 (2-4). Mitjançant l'ús de R i Bioconductor, s'ha realitzat una anàlisis d'expressió gènica diferencial per identificar gens amb canvis significatius en la seva expressió, així com una anàlisi funcional per detectar quins processos biològics es troben afectats. Els resultats mostres diferències transcriptòmiques significatives entre els grups COVID-19, infecció bacteriana i control sa, associades a diferents funcions biològiques. Aquest tipus d'estudis poden ajudar a comprendre millor la resposta de l'hoste a infeccions i contribuir a la identificació de possibles dianes terapèutiques o biomarcadors.

HIPÒTESIS I OBJECTIUS

HIPÒTESI

El pacients amb COVID-19 presenten un perfil d'expressió gènica diferent en comparació amb els individus sans i els pacients que pateixen una infecció bacteriana. Aquests canvis transcriptòmics poden reflectir processos biològics específics associats a la resposta immunitària davant la infecció per SARS-CoV-2.

OBJECTIUS

L'objectiu principal d'aquest treball és analitzar l'expressió gènica diferencial en sang perifèrica de pacients amb COVID-19, infecció bacteriana i controls sans, mitjançant l'ús de dades d'RNA-seq. Per tal d'assolir aquest objectiu, es plantegen els següents objectius específics:

1. Descarregar, processar i explorar el conjunt de dades GSE161731.
2. Dur a terme un estudi bioestadístic i bioinformàtic per identificar gens diferencialment expressants entre els diferents grups d'estudi (COVID-19, Bacterial, Healthy).
3. Dur a terme un estudi bioinformàtic de sobrerepresentació per identificar les funcions enriquides entre els gens sobreexpressats en pacients amb COVID19 en comparació amb els controls sans.
4. Interpretar els resultats per identificar quins processos biològics es troben enriquits en els gens diferencialment expressats en pacients amb COVID-19 per comprendre la resposta immunitària contra la infecció per SARS-CoV-2.

MATERIALS I MÈTODES

El codi utilitzat per cada un dels apartats i en la metodologia es troba detallat en l'annex d'aquest informe.

DADES

El conjunt de dades utilitzat en aquest estudi prové de l'estudi publicat per McClain et al. (3) i està disponible públicament al respositori del *Gene Expression Omnibus* (GEO) sota l'identificador GSE161731 (2). Aquest conjunt de dades conté informació d'expressió gènica obtinguda mitjançant RNA-seq a partir de 77 mostres de sang perifèrica de 46 individus classificats en diferents cohorts (COVID-19, infecció coronavirus, grip, pneumònia bacteriana, controls sans). Les metadades proporcionen informació clínica i demogràfica de les mostres, i les dades transcriptòmiques han estat anotades amb coordenades gèniques utilitzant la base de dades EnsDb.Hsapiens.v86. Les mostres van ser seqüenciades mitjançant la tecnologia Illumina NovaSeq 6000 (5).

Per la realització d'aquest treball, s'han seleccionat 75 mostres de manera aleatòria, corresponents exclusivament a les cohorts de pacients amb COVID-19, infeccions bacterianes i controls sans.

PROCESSAMENT DE LES DADES

Pre-processament i preparació de l'entorn de treball

Les dades transcriptòmiques utilitzades es van obtenir del repositori Gene Expression Omnibus (GEO). Els arxius associats a les dades d'expressió gènica i a la informació clínica associada es van descarregar mitjançant el paquet *GEOquery* (6). Per poder prepara les dades de la anàlisis es van carregar a R els fitxers corresponents a la matriu de *counts* i a la metadata (*counts_key*) de l'estudi. Les dades d'aquests dos fitxers es van llegir com a dataframes independents.

Seguidament, es van extreure els identificadors dels gens presents a la matriu d'expressió per tal de recuperar la seva informació genòmica mitjançant el paquet *EnsDb.Hsapiens.v86.* Aquesta informació és la que ens va permetre construir les coordenades genètiques emmagatzemades a l'objecte *GRanges* i que s'associen a cada gen. Mitjançant aquesta informació, es va poder filtrar la matriu perquè contingués només aquells gens que tenien la informació completa.

Per assegurar la consistència entre la matriu d'expressió i la metadata, es van identificar i seleccionar les mostres comuns entre les dues fonts, i es van ordenar per poder garantir que les dades es fusionaven correctament amb la funció *intersect()*. Un cop realitzats aquests canvis, es va procedir a la construcció de l'objecte *SummarizedExperiment* (7), que integra la matriu amb els *counts*, és a dir, les dades d'expressió, la informació de les mostres continguda a *colData* i les coordenades genòmiques a *rowRanges*. Aquest objecte ens permet manipular i analitzar de manera més eficient les dades derivades del RNA-seq amb el que estem treballant.

Posteriorment, es va realitzar un procés de neteja i filtratge per obtenir un conjunt de dades apte per l'anàlisi. Aquesta va incloure els següents passos:

- Selecció de cohorts d'interès, que en aquest cas eren *COVID-19, Bacterial* i *healthy*, eliminant la resta de condicions present en l'estudi.
- Eliminar mostres duplicades, ja que com es menciona hi ha individus que es troben repetits en l'atribut *subject_id*. Es detecten 31 individus duplicats i, per evitar baixos, es conserva només la primera mostra associada a cada individu.
- Revisió i transformació de les classes de les variables perquè sigui coherent. En aquest procés *age* es va canviar a *numèric* i *cohort* a factor.
- Eliminar símbols no desitjats i garantir una nomenclatura coherent en les dades. Tot els símbols "conflictius" van ser substituïts per "_".

Finalment, per tal de reduir la càrrega computacional i facilitar la anàlisis, es van seleccionar aleatòriament 75 mostres amb *myseed()* amb una llavor definida pel nom complet de l'autor del treball (ID: carlapaniselloaranda). Amb les mostres seleccionades, es va emmagatzemar la informació en un nou objecte *SummarizedExperiment*.

Control de qualitat

L'exploració de les dades i la realització d'un control de qualitat adient són essencials per assegurar-nos que les dades es troben degudament representades i compleixen condicions concretes per poder realitzar els anàlisis posteriors de manera adient. En aquest estudi, es van dur a terme diferents accions per garantir la fiabilitat i detectar possibles problemes tècnics o biològics.

- Aplicació d'un filtratge per eliminar aquells gens amb nivells d'expressió baixos mitjançant l'ús del criteri de counts-per-million (CPM) amb un llindar de 0.5 CPM en alemanys dues llibreries. Aquesta estratègia permet reduir el soroll i evitar la inclusió de gens no informatius o bé que poden donar lloc a biaixos (8).
- Càlcul del nombre total de lectures per mostra, expressat en milions, mitjançant la suma dels valors d'expressió. Aquesta informació es representa mitjançant un diagrama de barres amb les mostres agrupades segons el seu grup cohort (COVID-19, Bacterial o healthy). Aquesta visualització permet detectar la variabilitat entre les mostres i identificar aquelles amb menys de 20 milions de lectures, que podrien tenir algun efecte en la anàlisi.

Aquesta primera exploració de les dades, proporciona una base sòlida per considerar si és necessari aplicar transformacions o normalitzacions abans de procedir amb l'anàlisi de l'expressió gènica diferencial.

Normalització de les dades i control de qualitat

La normalització de les dades és un pas fonamental per la anàlisi transcriptòmica per corregir les diferències sistemàtiques entre mostres, especialment derivades de la mida de la llibreria i altres efectes tècnics. En aquest estudi, es va aplicar la normalització mitjançant el mètode *Trimmed Mean of M Values* (TMM) (8). Aquest mètode es pot utilitzar mitjançant la funció *calcNormFactors()* que es troba en el paquet *edgeR* (8,10). Aquest mètode ajusta les diferències de la composició entre mostres i permet una comparació més fiable dels nivells d'expressió gènica.

Un cop realitzada la normalització, es va procedir a transformar les dades en *logCPM*, que bàsicament és fer el logaritme dels CPM mitjançant la funció *cpm(..., log = TRUE)*. Aquesta transformació permet estabilitzar la variància i reduir l'impacte dels valors extrems.

Per avaluar l'impacte de la normalització, es van generar diversos gràfics comparatius entre les dades brutes (no normalitzades ni transformades) i les dades normalitzades:

- MA-plot permet mostrar la diferència M en l'expressió gènica respecte la mitjana A entre dues mostres. La normalització hauria de reduir la dispersió.
- Boxplot amb Relative Log Expression (RLE) que permet observar si la distribució d'expressió per mostra és centrada i comparable entre condicions.

ANÀLISI EXPLORATÒRIA DE LES DADES (EDA)

Un cop transformades i normalitzades les variables, es van avaluar les dades per detectar mostres *outliers* o bé factors que podrien afectar a les conclusions de l'estudi. Per tal de fer aquest *screening*, es van dur a terme diverses tècniques d'exploració:

- Multi-dimensional scaling (MDS) plot per visualitzar la similitud global entre mostres i identificar agrupacions segons la cohort (COVID-19, bacterial, healthy).
- Anàlisis de components principals (PCA) per avaluar la variabilitat global i corroborar les agrupacions entre mostres.
- Clustering jeràrquic per identificar patrons similars en les mostres basant-se en la distància de Pearson entre els diferents perfils d'expressió.
- Heatmap seleccionant el subconjunt de 500 gens amb major variabilitat per intentar determinar patrons diferencials d'expressió entre les diferents cohorts de l'estudi.

En el cas d'identificar outliers en l'anàlisi exploratòria anterior, es realitzarà un gràfic MDS on veure el nom de cada subjecte. D'aquesta manera, es podrà identificar la mostra outlier i seguidament es podrà procedir a la seva eliminació tant dels DEGs com del data set inicial *Covid_75*. Un cop eliminat d'ambdós conjunts de dades, podem procedir a fer una exploració de les dades per veure com es distribueixen. Es pot triar qualsevol dels gràfics anteriors, però en aquest cas es realitzarà un MDS plot que és bastant informatiu.

Per realitzar l'estudi de les confusion variables, mirarem quins factors hi ha en el data set descarregat que puguin afectar a les conclusions de l'estudi. Realitzant *colnames(colData(Covid_75))* podrem veure quins factors hi ha al data set original. En base a aquests, realitzarem gràfics MDS de cada factor sobreposat amb la cohort en el cas de variables categòriques i boxplots per les variables numèriques. En aquest cas, el color indicarà els diferents nivells de cada factor o variable analitzat, i la forma (quadrat, triangle, cercle) el nivell de la cohort. A més, es realitzarà un estudi estadístic amb Chi-quadrat per les variables categòriques i ANOVA per les numèriques. Aquests gràfics ens permetran observar si hi ha algun factor que pugui estar afectant a les conclusions, i que per tant, calgui tenir en compte en la generació de la matriu de disseny i de contrastos pels posteriors anàlisis.

MATRIU DE DISSENY I CONTRASTOS I ANÀLISIS DE L'EXPRESSIÓ DIFERENCIAL DE GENS (DEGS)

Per identificar els gens diferencialment expressats entre grups, s'utilitza el paquet edgeR seleccionat aleatòriament entre diversos enfocaments d'anàlisis d'expressió diferencial per evitar biaixos en l'elecció metodològica.

Tenint en compte la anàlisis exploratòria es construeix una matriu de disseny mitjançant la funció *model.matrix* incorporant els factores explicatius del grup experimental (cohort) i les possibles variables confussores. S'agafen els counts

normalitzats i es crea una DGEList que permet estimar la dispersió de les dades amb la funció *estimateDisp*, que s'utilitza per ajustar el model lineal generalitzar amb *glmQLFit* per modelar l'expressió gènica en funció del disseny experimental (8).

En base a la matriu generada, es poden realitzar les matrius de contrastos per veure l'expressió diferencial entre *Bacterial vs healthy* i *COVID-19 vs healthy*. Aquests es poden especificar amb *makeConstrasts* aplicant *glmQLFTest* per obtenir estadístics de contrast per cada gen. D'aquest resultat, obtenim els *topTags*, on tenim els gens amb valor de log fold change (logFC) superior a 1.5 i FDR inferior a 0.05. En base a aquests resultats, es realitzen volcano plots amb la funció *EnhancedVolcano* per observar l'expressió diferencial de gens (11).

COMPARACIÓ DE CONTRASTOS

Per identificar gens diferencialment expressats compartits i específics entre les dues condicions analitzades (Bacterial vs healthy i COVID-19 vs healthy) s'extreuen gens significatius de cada comparació. Es seleccionen aquells amb un valor de FDR inferior a 0.05 i un valor absolut de logFC superior a 1.5. A partir d'aquests conjunt de gens, es construeixen un diagrama de Venn amb la funció *venn.diagram* del paquet *VennDiagram* i un gràfic Upset del paquet *ComplexUpset* per poder veure quants gens son comuns, quants es troben només en bacteris i quants només en Covid dintre dels diferencialment sobreexpressats.

ANÀLISIS D'ENRIQUIMENT FUNCIONAL

Per identificar processos biològics significativament associats amb els gens sobreexpressats en cada condició, es realitza un anàlisis d'enriquiment funcional utilitzant les anotacions de Gene Ontology (GO), centrant-nos exclusivament en la categoria de processos biològics (Biological processes, BP).

Per poder treballar amb GO, es converteixen els identificadors ENSEMBL a ENTREZ ID utilitzant *bitr* del paquet *clusterProfiler*, amb la base de dades d'anotació humana *org.Hs.eg.db*. Un cop tenim aquesta conversió, passem a fer l'enriquiment funcional amb *enrichGO*, especificant el mètode de correcció per múltiples comparacions de Benjamini-Hochberg (BH) amb valors de p ajustats a >0.05 i q-value a >0.2 com a llindars de significància. Els resultats es visualitzen amb *dotplot* mostrant els 15 processos amb més significància (12).

RESULTATS

PROCESSAMENT I CONTROL DE QUALITAT DE LES DADES CRUES

Les dades de GSE161731 es van obtenir correctament mitjançant GEOquery. Es va realitzar de manera eficient el pre-processament de les dades seguint els passos mencionats anteriorment. Inicialment, es tenia metadata de 57602 gens, però en canvi, la matriu d'expressió descarregada contenia 60675 gens. Per aquest motiu, es van filtrar els gens perquè es generés una matriu amb tota la informació, eliminant aquells que no es trobaven en ambdues.

Un cop teníem aquesta informació, es va generar l'objecte *SummarizedExperiment* amb el que es treball durant tot l'estudi. Aquest objecte també va ser sotmès a canvis, filtrant només els elements d'interès com era la cohort COVID-19, Bacterial i healthy. Eliminem correctament els elements duplicats, que com es pot veure a l'annex amb el codi equival a 31. A part, també canviem la classe d'edat a numèric i de cohort a factor per poder procedir amb la anàlisi. Finalment, es seleccionen aleatòriament 75 mostres.

Amb aquestes dades satisfactòriament processades, es va realitzar un filtratge de gens poc expressats tenint en compte els count-per-million (CPM) (Fig.1). Com podem observar, només 24652 gens es trobaven per sobre el llindar establert. Son aquests amb el que es treballarà durant l'estudi.

```
Covid_75_dge <- DGEList(counts = assay(Covid_75))
keep <- rowSums(cpm(Covid_75_dge) > 0.5) >= 2
table(keep)

## keep
## FALSE  TRUE
## 32950 24652

Covid_75_dge <- Covid_75_dge[keep,]
```

Figura 1. Codi per eliminar gens amb CPM baix.

Com a últim pas del control de qualitat, es realitza una observació de les dades segons els counts amb un gràfic de barres. En aquest, observem com es distribueixen les mostres, i que totes elles tenen un nombre de reads elevat. Tot i així, observem que hi ha una que potser hauríem d'analitzar, ja que té menys de 20M reads. Sospitem que aquesta mostra podria ser un outlier, però cal realitzar un estudi més precís per determinar si ho és o no, i si cal eliminar-ho (Fig. 2). Aquest canvi ho farem amb l'estudi de qualitat posterior a la normalització.

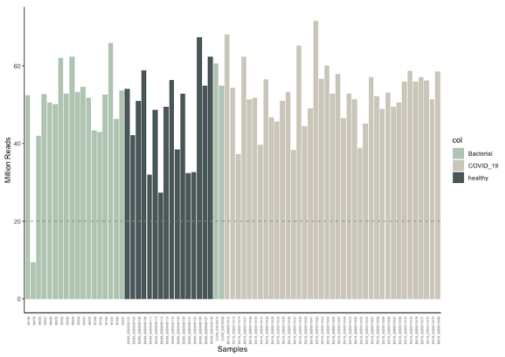


Figura 2. Barplot CPM gens en mostres individus.

NORMALITZACIÓ I CONTROL QUALITAT

Quan fem servir el paquet *edgeR* (el que ha tocat segons "carlapaniselloaranda"), una manera típica de treballar amb les dades és normalitzar-les amb el mètode trimmed mean of M vàlues (TMM) mitjançant la funció *calcNormFactors*, i posteriorment realitzar una transformació logarítmica (logCPM).

Per comprovar si les nostres dades estan correctament normalitzades, el que fem és una sèrie d'anàlisis que ens permetran valorar-ho gràficament i comparar-ho amb la *raw data* sense normalitzar (Fig.3). Podem observar tant per maPlot (Fig.3A) com per boxplots RLE (Fig.3B) que no hi ha massa canvis entre la Raw data i les dades normalitzades. Això pot venir donar perquè el factor de normalització és proper a 1. Tot i així, si ens fixem en el Boxplot observem que les mostres estan més properes a 0 i el rang és de 1 a 1. És un canvi subtil, però s'està aplicant correctament.

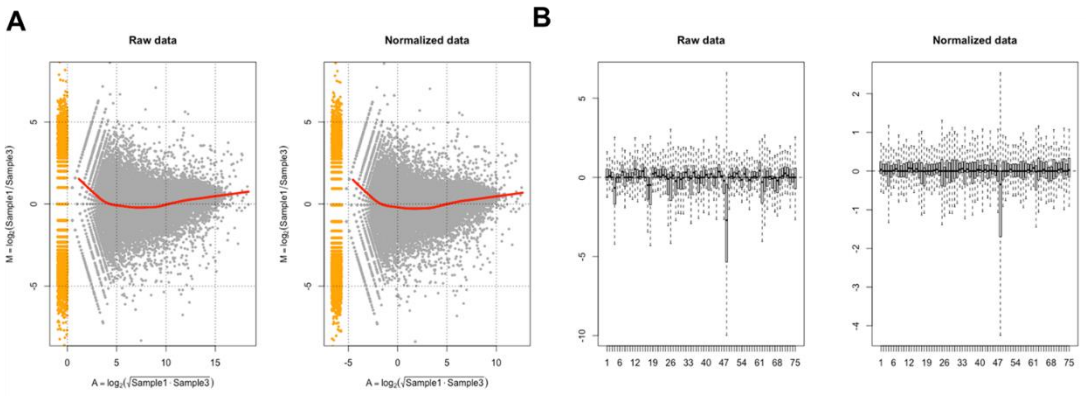


Figura 3. Control de qualitat de la normalització de Covid_75 mitjançant maPlot (A) i boxplot RLE (B).

Aquest estudi rebel·la que la transformació / normalització de dades logarítmiques ha funcionat, i que per tant, es pot procedir amb els diferents anàlisis estadístics i de funció biològica.

ANÀLISIS EXPLORATORI DE LES DADES

L'anàlisi exploratori de les dades ens permetrà veure com es distribueixen les dades segons la cohort determinada (COVID-19, bacterial i healthy). Mitjançant els plots de MDS (Fig.4A), PCA (Fig.4B), clustering jeràrquic (Fig.4C) i heatmap (Fig.4D) podem observar que les mostres de healthy i Covid-19 sembla que s'agrupen properes, mentre que les de bacerial es troben més separades, semblen més diferents. Per altra banda, observem que una mostra de bacterial es troba allunyada de tot, podria ser un outlier. Probablement és la mostra amb pocs CPM observada en el barplot de la Figura 2.

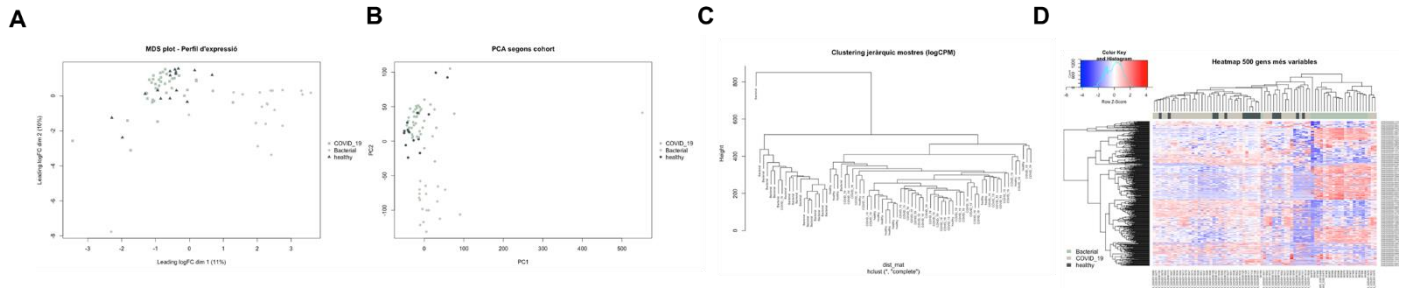


Figura 4. Anàlisis exploratori de les mostres segons la cohort per MDS plot (A), PCA (B), clustering jeràrquic (C) i heatmap (D).

Per poder veure a quina mostra correspon el valor outlier, podem generar el plot MDS però perquè ens mostri el ID de les mostres. el ID de les mostres. Mitjançant aquest gràfic, podem observar com la mostra de la cohort Bacterial que actua com outlier és la 94478 (Fig.5A). Aquesta mostra cal eliminar-la dels nostres anàlisis, ja que podria provocar la inferència de conclusions errònies.

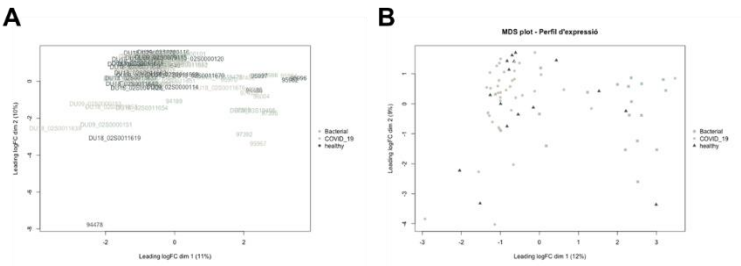


Figura 5. MDS plot per determinar ID de l'outlier (A) i MDS post eliminació de l'outlier (B).

Un cop eliminat outlier, al realitzar el gràfic de MDS, observem que les mostres es troben distribuïdes de manera més homogènia. Tot i així, seguim observant que bacterial forma un clúster més diferenciat.

Seguidament, realitzem un estudi de les possibles variables confusores. Aquestes poden ser variables que poden donar lloca a inferències errònies. Entre elles ens trobem amb les variables age, gender, race, time_since_onset, hospitalized i batch. Cal veure si hi ha algun patró en elles que pugui afectar a la distribució de la cohort, i que per tant calgui tenir en compte.

Observant cada factor un per un, podem veure que la variable age sembla que distribueix de manera diferencial en funció de l'edat. Podem veure que els que són més grans, tenen més tendència a tenir infeccions bacterianes (Fig.6A). Aquest fet ens permet determinar que aquest podria ser una variable confusora. En quant al gènere, tant estadísticament per Chi-quadrat com gràficament, no s'observa una distribució que porti a confusió (Fig.6B). Per raça, observem que sembla que els Black_African_American podrien portar a confusió, i a part, estadísticament és significatiu (Fig.6C). Per hospitalització (Fig.6D) i per onset (Fig.6E) no observem tampoc camp impacte important. En canvi, si veiem que hi ha efecte batch (Fig.6F) i que és estadísticament significatiu com es pot veure en l'Annex del treball.

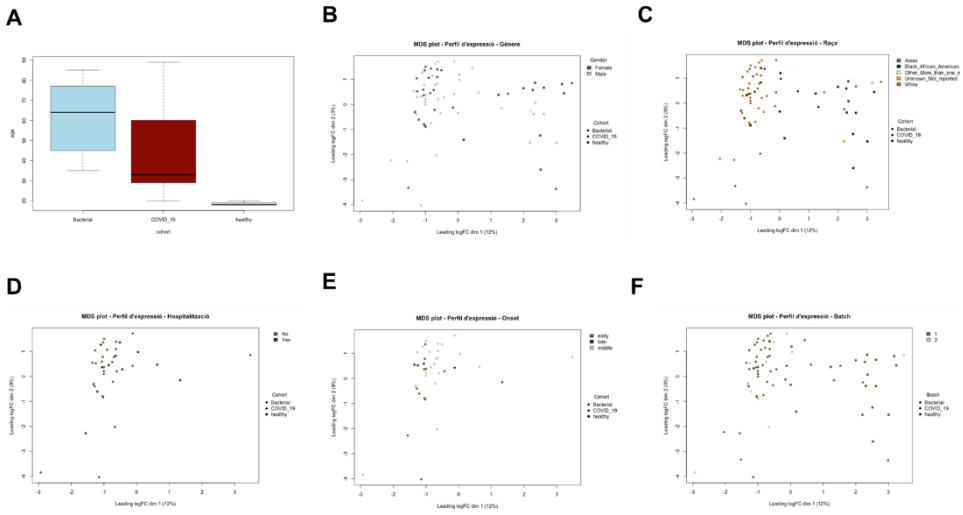


Figura 6. Estudi de les possibles variables confussores d'edat (A), gènere (B), raça (C), hospitalització (D), onset (E) i batch (F).

Amb aquest primer anàlisis exploratori, podem determinar que sembla que les mostres bacterial es comporten diferent a les cohorts de COVID-19 i healthy. A més, hem identificat un outlier i l'hem pogut eliminar. Finalment, hem vist com poden haver diverses variables confussores com serien la edat, la raça i l'efecte batch. Aquests últims paràmetres son necessaris per fer la matriu de disseny i contrastos i l'anàlisi d'expressió diferencial.

MATRIU DE DISSENY I CONTRASTOS I ANÀLISI DE L'EXPRESSIÓ DIFERENCIAL DE GENS (DEGS)

Generem la matriu tenint en compte la edat dels individus, la seva raça i l'efecte bacth per evitar conclusions errònies. Seguidament fem les estimacions del disseny i ajustem al model lineal. En base a aquesta matriu de disseny, establim la matriu de contrastos que en aquest cas contindrà *Bacterial vs healthy* i *COVID19 vs healthy.*

Els volcano plots derivats d'aquests anàlisis mostren que hi ha molts gens que es troben diferencialment expressats en mostres bacterianes en comparació a sanes (Fig.7A). Pel COVID19, també observem gens diferencialment expressats, tot i que en menys mesura (Fig.7B).

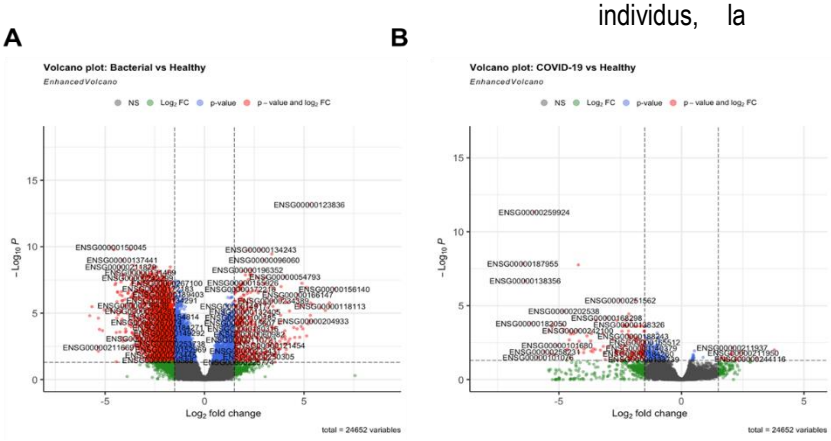


Figura 7. Volcano plots amb l'expressió diferencial de mostres de pacients amb infeccions bacterianes (A) o COVID-19 (B) en comparació a individus sans.

COMPARACIÓ DE CONTRASTOS

Tenint en compte els gens diferencialment expressats per cada comparació, podem veure quins d'aquests son comuns i quins difereixen. Mitjançant la realització de un diagrama de Venn (Fig.8A) i Upset (Fig.8B), observem que hi ha 3173 gens només expressats en infeccions bacterianes, 101 en COVID i 80 que són comuns.

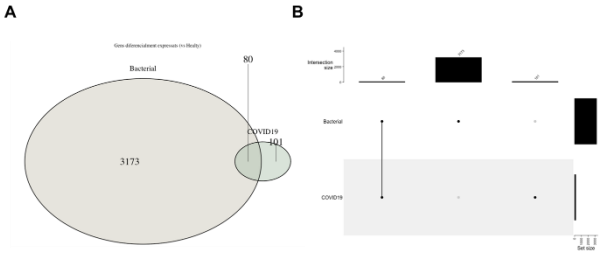


Figura 8. Diagrama de Venn (A) i gràfic UpSet (B) mostrant expressió diferencial de gens en Bacterial i COVID-19.

ANÀLISI D'ENRIQUIMENT FUNCIONAL

Al realitzar estudi de l'enriquiment funcional, observem que els gens sobreexpressats en COVID-19 es relacionen amb regulació positiva de la resposta immunitària com ara la producció de citocines, regulació de la resposta inlfamatòria, regulació negativa de processos immunes, etc (Fig.9A). Té a veure amb una resposta més pro-inflamatòria. En canvi, quan mirem aquells relacionats amb la resposta en infeccions bacterianes, veiem que també son mediadors de la resposta immunitària, però hi ha un gran component humoral amb producció de immunoglobulines, divisió nuclear, i processos més intrínsecament cel·lulars (Fig.9B).

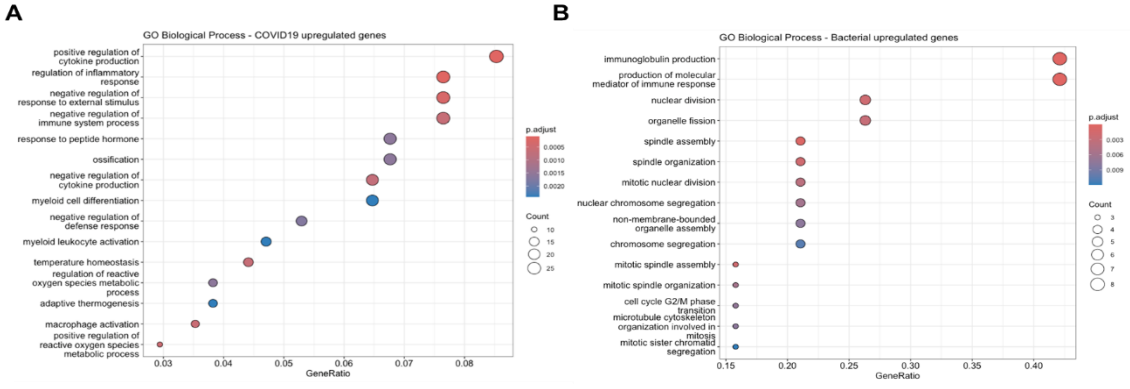


Figura 9. Anàlisis d'enriquiment funcional en COVID19 (A) i infeccions bacterianes (B).

DISCUSSIÓ

Els resultats obtinguts rebel·len diferències clares i significatives en el perfil d'expressió gènica entre els pacients amb COVID-19, aquells amb infecció bacteriana i el control sa. Aquestes diferències no només es veuen reflectides en el nombre de gens diferencialment expressats, sinó que també es donen en el tipus de funcions biològiques implicades en cada tipus d'infecció.

En primer lloc, observem que hi ha més gens diferenciats en les infeccions bacterianes que no pas en la de COVID19 quan es compara amb individus sans. Si mirem concretament el tipus de resposta que es dona, podem observar que les infeccions bacterianes activen processos cel·lulars i respostes humorals i innates. En canvi, la infecció de COVID19 presenta un patró de gens regulats implicats en respostes pro-inflamatòries, producció de citocines i respostes antivirals.

La anàlisi de coincidència entre els conjunts de gens diferencialment expressats, ha permès identificar gens que son exclusius per cada tipus de condició i altres que son compartits. En concret, per bacteris trobem que hi ha 3173 gens exclusius sobreexpressats, mentre que per COVID19 hi ha només 101 i en comú només en tenen 80. Aquest fet suggereix que les respostes fisiològiques que s'activen davant ambdues infeccions són notòriament diferents, probablement per la naturalesa de la infecció (bacteriana o vírica) (13,14). Les infeccions bacterianes donen lloc a respostes del tipus humoral, amb la producció d'anticossos i immunoglobulines. En canvi, les víriques com la COVID, presenten un component més cel·lular i pro-inflamatori. Aquest fet pot tenir implicacions importants a l'hora de triar el tractament a aplicar als pacients i alhora a realitzar un bon diagnòstic en base a dades transcriptòmiques. Cal tenir en compte, que els processos comuns també poden tenir rellevància, ja que poden ser dianes comuns per els dos tipus d'infeccions.

Aquestes observacions poden tenir una gran rellevància cínica. L'ús de signatures transcriptòmiques per distingir entre infeccions bacterianes i víriques podria contribuir a l'hora de prendre decisions en el tractament de les diferents infeccions. Un dels problemes que va provocar la infecció per la COVID-19 en pacients crítics era que hi havia una sobre-activació de la resposta immunitària que no era compensada pel *feedback* negatiu de la resposta immunitària. En molts casos, aquesta sobre-activació era la que provocava fibrosis pulmonar i l'*exitus* de mols pacients (15) . Conèixer els mecanismes que tenen lloc en pacients en COVID-19 en base a dades transcriptòmiques, pot permetre conèixer quines dianes atacar i trobar estratègies per aturar aquesta sobre-activació de la immunitat.

Tanmateix, l'estudi presenta certes limitacions. En primer lloc, la presa aleatòria de la mostra pot tenir un efecte en els resultats. A més, hi ha diversos pacients que cal eliminar de l'estudi perquè no tenen prou informació clínica que pot ser rellevant per l'estudi. Aquest fet, pot afectar a la interpretació biològica dels resultats.

En conclusió, aquest estudi ens permet estudiar els patrons d'expressió diferencial entre COVID-19 i infeccions bacterianes, i obre la porta a entendre com funciona la resposta immunitària envers ambdós tipus d'infeccions, i a trobar marcadors immunològics útils per la pràctica clínica. Tot i així, cal validar aquestes dades amb altres subconjunts de pacients, ampliant la cohort i incorporant altres variables que puguin afectar als models d'anàlisi.

CONCLUSIÓ

Després d'analitzar el conjunt de dades GSE161731, realitzar el processament de les dades i dur a terme l'anàlisi d'expressió gènica diferencial, podem concloure que la nostra hipòtesi inicial es confirma: el perfil transcriptòmic dels pacients amb COVID-19 difereix clarament del dels pacients amb infeccions bacterianes i el grup control sa. El resultats obtinguts mostren diferències significatives tant en el nombre de gens expressats diferencialment, com en les funcions biològiques associades a diferents processos immunitaris per cada tipus d'infecció.

Aquests resultats identifiquen la transcriptòmica com una bona eina per identificar possibles biomarcadors i processos biològics associats a infeccions o malalties que poden ser útils per la gestió clínica. A més a més, ens permet entendre la biologia de les resposta immunitària per cada tipus d'infecció. Tot i així, per poder treure conclusions sòlides, cal validar els resultats amb altres cohorts i ampliar la mostra. Això ens permetrà aprofundir de manera més fiable en la anàlisis d'expressió diferencial i funcional.

REFERÈNCIES

Repositori GitHub: https://github.com/cpani6/Panisello-Aranda-Carla-PEC2

1. Wang, Z., Gerstein, M. & Snyder, M. RNA-Seq: a revolutionary tool for transcriptomics. Nat Rev Genet. 2009. 10, 57–63.
2. McClain MT, Constantine FJ, Nicholson BP, et al. (2021). Host responses to SARS-CoV-2 compared to other viral and bacterial infections in human blood transcriptomes. NCBI Gene Expression Omnibus, accession number GSE161731. Available at: https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE161731
3. McClain MT, Constantine FJ, Henao R, Liu Y, Tsalik EL, Burke TW, Steinbrink JM, Petzold E, Nicholson BP, Rolfe R, Kraft BD, Kelly MS, Saban DR, Yu C, Shen X, Ko EM, Sempowski GD, Denny TN, Ginsburg GS, Woods CW. Dysregulated transcriptional responses to SARS-CoV-2 in the periphery. Nat Commun. 2021 Feb 17;12(1):1079.
4. Wang L, Balmat TJ, Antonia AL, Constantine FJ, Henao R, Burke TW, Ingham A, McClain MT, Tsalik EL, Ko ER, Ginsburg GS, DeLong MR, Shen X, Woods CW, Hauser ER, Ko DC. An atlas connecting shared genetic architecture of human diseases and molecular phenotypes provides insight into COVID-19 susceptibility. Genome Med. 2021 May 17;13(1):83.
5. NovaSeq 6000 Sequencing System. Illumina. Available from: https://www.illumina.com/systems/sequencing-platforms/novaseq.html
6. Davis S. Using the GEOquery Package. Bioconductor. 2014. Available from: https://bioconductor.org/packages/devel/bioc/vignettes/GEOquery/inst/doc/GEOquery.html
7. Morgan M, Obenchain V, Hester J, Pagès H. SummarizedExperiment for Coordinating Experimental Assays, Samples, and Regions of Interest. Bioconductor. 2023. Available from: https://www.bioconductor.org/packages/release/bioc/vignettes/SummarizedExperiment/inst/doc/SummarizedExperiment.html
8. Chen Y, Lun ATL, Smyth GK. From reads to genes to pathways: differential expression analysis of RNA-Seq experiments using Rsubread and the edgeR quasi-likelihood pipeline. F1000 Research. 2016; 5:1438.
9. Robinson MD, Oshlack A. A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biology. 2010;11, R25.
10. Normalisation. RNAseq analysis with R. Qfab-bioinformatics. Github. Available from: https://qfab-bioinformatics.github.io/workshops-RNAseq-analysis-with-R/normalisation.html
11. Blighe K, Rana S, Lewis M. EnhancedVolcano: publication-ready volcano plots with enhanced colouring and labeling. Bioconductor. 2025. https://bioconductor.org/packages/devel/bioc/vignettes/EnhancedVolcano/inst/doc/EnhancedVolcano.html
12. Overview of enrichment analysis. Biomedical Knowledge Mining using GOSemSim and clusterProfiler. Available from: https://yulab-smu.top/biomedical-knowledge-mining-book/enrichment-overview.html#ora-algorithm
13. Justiz Vaillant AA, Sabir S, Jan A. Physiology, Immune Response. [Updated 2024 Jul 27]. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing; 2025 Jan-. Available from: https://www.ncbi.nlm.nih.gov/books/NBK539801/
14. Melenotte C, Silvin A, Goubet AG, Lahmar I, Dubuisson A, Zumla A, Raoult D, Merad M, Gachot B, Hénon C, Solary E, Fontenay M, André F, Maeurer M, Ippolito G, Piacentini M, Wang FS, Ginhoux F, Marabelle A, Kroemer G, Derosa L, Zitvogel L. Immune responses during COVID-19 infection. Oncoimmunology. 2020 Aug 25;9(1):1807836.
15. Alahdal M, Elkord E. Exhaustion and over-activation of immune cells in COVID-19: Challenges and therapeutic opportunities. Clinical Immunology. 2022;245; 109177.

ANNEX

Treballem amb el dataset GSE161731 del treball de McClain et al., realitzant un anàlisis d'expressió gènica diferencial en R/Bioconductor en pacients amb COVID_19 en comparació amb altres patologies. A continuació es troba el codi emprat per la realització del treball.

LOAD PACKAGES THAT WILL BE USED IN THIS CODE

```
suppressPackageStartupMessages(library(GEOquery))

## Warning: package 'Biobase' was built under R version 4.3.1

## Warning: package 'BiocGenerics' was built under R version 4.3.1

suppressPackageStartupMessages(library(AnnotationDbi))

## Warning: package 'AnnotationDbi' was built under R version 4.3.1

## Warning: package 'IRanges' was built under R version 4.3.1

## Warning: package 'S4Vectors' was built under R version 4.3.2

suppressPackageStartupMessages(library(EnsDb.Hsapiens.v86))

## Warning: package 'ensembldb' was built under R version 4.3.1

## Warning: package 'GenomicRanges' was built under R version 4.3.1

## Warning: package 'GenomeInfoDb' was built under R version 4.3.1

## Warning: package 'GenomicFeatures' was built under R version 4.3.1

suppressPackageStartupMessages(library(GenomicRanges))
suppressPackageStartupMessages(library(SummarizedExperiment))

## Warning: package 'MatrixGenerics' was built under R version 4.3.1

## Warning: package 'matrixStats' was built under R version 4.3.3

suppressPackageStartupMessages(library(edgeR))
suppressPackageStartupMessages(library(ggplot2))

## Warning: package 'ggplot2' was built under R version 4.3.3

suppressPackageStartupMessages(library(reshape2))
suppressPackageStartupMessages(library(gplots))

## Warning: package 'gplots' was built under R version 4.3.3

suppressPackageStartupMessages(library(EnhancedVolcano))

## Warning: package 'ggrepel' was built under R version 4.3.3

suppressPackageStartupMessages(library(VennDiagram))
suppressPackageStartupMessages(library(ComplexHeatmap))
suppressPackageStartupMessages(library(clusterProfiler))
suppressPackageStartupMessages(library(org.Hs.eg.db))
```

1. DOWNLOAD THE FILES FROM GSE161731

```
# gse <- getGEOSuppFiles("GSE161731") # Només ho fem servir el primer cop, després es crea una car
peta "GSE161731" on estem treballant.
```

```
# Llegim els arxius de la carpeta.
gse<-list.files(path = "GSE161731",full.names = TRUE)

# Els arxius contenen les dades d'expressió i la metadata.
gse

## [1] "GSE161731/GSE161731_counts_key.csv.gz"
## [2] "GSE161731/GSE161731_counts.csv.gz"
## [3] "GSE161731/GSE161731_key.csv.gz"
## [4] "GSE161731/GSE161731_xpr_nlcpm.csv.gz"
## [5] "GSE161731/GSE161731_xpr_tpm_geo.txt.gz"

# Counts
expr_matrix <- read.csv(gse[grepl("counts.csv.gz",gse)], row.names = 1,check.names = FALSE)

# Counts Key
meta_info <- read.csv(gse[grepl("counts_key.csv.gz",gse)])

# Busquem els features per generar les coordenades genètiques.
gene_ids <- rownames(expr_matrix)

# Obtenim les coordenades genètiques juntament amb els identificadors del gen.
gene_ranges <- genes(EnsDb.Hsapiens.v86, filter = GeneIdFilter(gene_ids))

# Tenim informació de 57602 gens, i la matriu d'expressió conté 60675. Per tant, cal filtrar els g
ens de la matriu d'expressió per poder analitzar aquells que tenim tota la informació.
expr_matrix <- expr_matrix[gene_ranges$gene_id,]

# El número de columnes de la matriu d'expressió i el número de files de la metadata no coincideix
. Per aquest motiu, seleccionem mostres analitzades on tenim tota la informació.
common_samples <- intersect(colnames(expr_matrix), meta_info$rna_id)

# Filtrem i ordenem mostres dels dos objectes
expr_matrix_final <- expr_matrix[, common_samples]
meta_info_final <- meta_info[meta_info$rna_id %in% common_samples, ]
```

1.1 GENERATE SUMMARIZEDEXPERIMENT

```
# Mirem la dimensió dels objectes finals per comprovar que tot està correcte i que quadra.
dim(expr_matrix_final)

## [1] 57602   198

dim(meta_info_final)

## [1] 198   9

length(gene_ranges)

## [1] 57602

# Mirem si hi ha diferències en el IDs.
setdiff(colnames(expr_matrix_final),meta_info_final$rna_id)

## character(0)

setdiff(rownames(expr_matrix_final),gene_ranges$gene_id)

## character(0)

# Creem objecte SummarizedExperiment
Covid_se <- SummarizedExperiment(
  assays = list(counts = expr_matrix_final),
```

```
  colData = meta_info_final,
  rowRanges = gene_ranges
)

# Comprovem que objecte s'ha generat correctament.
Covid_se

## class: RangedSummarizedExperiment
## dim: 57602 198
## metadata(0):
## assays(1): counts
## rownames(57602): ENSG00000223972 ENSG00000227232 ... ENSG00000231514
##   ENSG00000235857
## rowData names(6): gene_id gene_name ... symbol entrezid
## colnames(198): 94189 DU09-03S0000604 ... 105847 105848
## colData names(9): rna_id subject_id ... hospitalized batch
```

2. CLEAN DATA

```
# Seleccionem només les dades d'interès que corresponen a COVID_19, Bacterial i healthy.
Covid_se_filter <- Covid_se[,Covid_se$cohort %in% c("COVID-19","Bacterial","healthy")]

table(Covid_se_filter$cohort)

##
## Bacterial  COVID-19   healthy
##        24        77        19

# Filtrem individus duplicats
table(duplicated(Covid_se_filter$subject_id))

##
## FALSE  TRUE
##    89    31

# Ens surt que hi ha 31, els eliminem i només ens quedem amb el primer valor.
Covid_se_clean <- Covid_se_filter[,!duplicated(Covid_se_filter$subject_id)]
table(duplicated(Covid_se_clean$subject_id))

##
## FALSE
##    89

# Mirem que les classes dels diferents paràmetres estiguin ben establertes.
str(colData(Covid_se_clean))

## Formal class 'DFrame' [package "S4Vectors"] with 6 slots
##   ..@ rownames       : chr [1:89] "94189" "DU18-02S0011619" "DU18-02S0011623" "DU18-02S0011627"
 ...
##   ..@ nrows          : int 89
##   ..@ elementType    : chr "ANY"
##   ..@ elementMetadata: NULL
##   ..@ metadata       : list()
##   ..@ listData       :List of 9
##   .. ..$ rna_id          : chr [1:89] "94189" "DU18-02S0011619" "DU18-02S0011623" "DU18-02S0011
627" ...
##   .. ..$ subject_id      : chr [1:89] "A1BD46" "450905" "0B943B" "7085CA" ...
##   .. ..$ age             : chr [1:89] "57" "60" "33" "28" ...
##   .. ..$ gender          : chr [1:89] "Female" "Male" "Male" "Male" ...
##   .. ..$ race            : chr [1:89] "Black/African American" "White" "White" "White" ...
##   .. ..$ cohort          : chr [1:89] "Bacterial" "COVID-19" "COVID-19" "COVID-19" ...
##   .. ..$ time_since_onset: chr [1:89] NA "early" "early" "early" ...
##   .. ..$ hospitalized    : chr [1:89] NA "No" "No" "No" ...
##   .. ..$ batch           : int [1:89] 1 1 2 1 1 1 1 1 1 1 ...
```

```
# Canviem edat (de caràcter a numèric) i cohort (de caràcter a factor)
colData(Covid_se_clean)$age <- gsub(">","", colData(Covid_se_clean)$age)
colData(Covid_se_clean)$age <- as.numeric(colData(Covid_se_clean)$age)

colData(Covid_se_clean)$cohort <- as.factor(colData(Covid_se_clean)$cohort)

# Netegem noms treient els símbols indicats a l'enunciat i substituïn per "_".

rownames_colData <- gsub("[\\/\\.()\\-]+", "_", rownames(colData(Covid_se_clean)))

Covid_coldata <- lapply(colData(Covid_se_clean), function(x) {
  if (is.character(x)|| is.factor(x)) {
    gsub("[^[:alnum:]]", "_", x)
  } else {
    x
  }
})

Covid_coldata <- as.data.frame(Covid_coldata)
rownames(Covid_coldata) <- rownames_colData
colData(Covid_se_clean) <- as(Covid_coldata, "DFrame")
colData(Covid_se_clean)$cohort <- as.factor(colData(Covid_se_clean)$cohort)

# I els colnames del summarizedexperiment
colnames(Covid_se_clean) <- gsub("[\\/\\.()\\-]+", "_", colnames(Covid_se_clean))

# Agafem 75 mostres de manera aleatòria.
myseed <- sum(utf8ToInt("carlapaniselloaranda"))
set.seed(myseed)
selected_samples <- sample(colnames(Covid_se_clean), 75)

Covid_75<- Covid_se_clean[,selected_samples]
Covid_75

## class: RangedSummarizedExperiment
## dim: 57602 75
## metadata(0):
## assays(1): counts
## rownames(57602): ENSG00000223972 ENSG00000227232 ... ENSG00000231514
##   ENSG00000235857
## rowData names(6): gene_id gene_name ... symbol entrezid
## colnames(75): DU18_02S0011631 DU18_02S0011646 ... DU18_02S0011634 95996
## colData names(9): rna_id subject_id ... hospitalized batch
```

3. QUALITY CONTROL

3.1. FILTERING

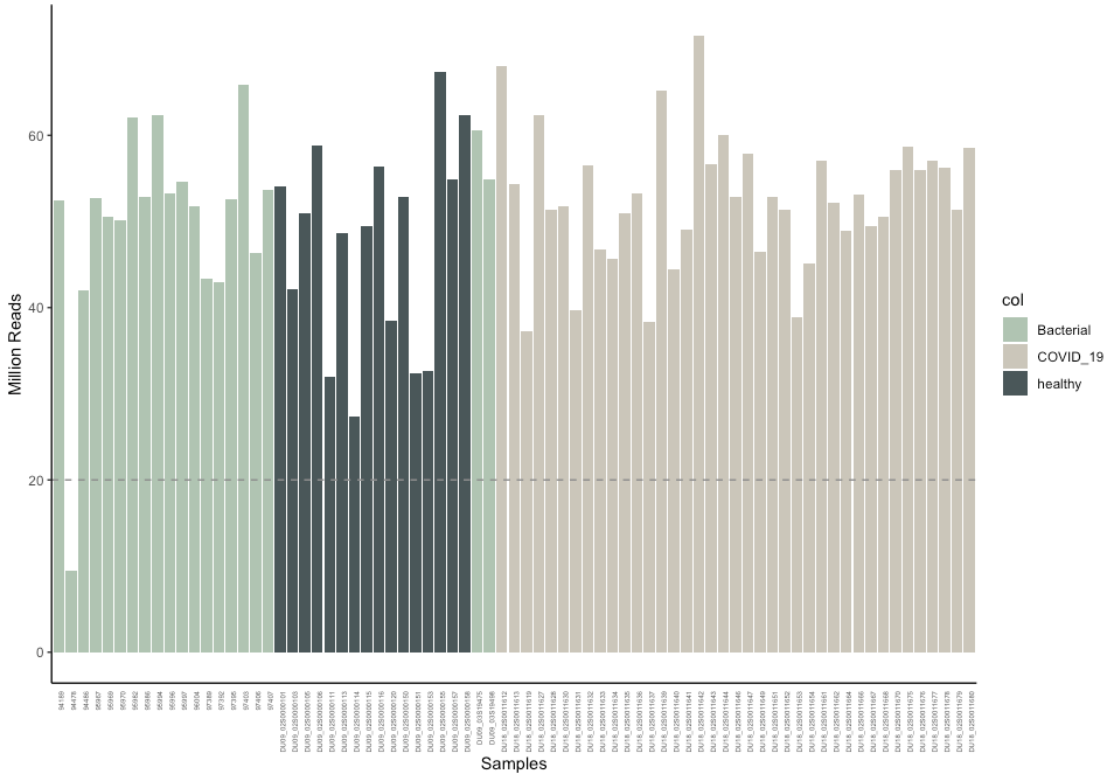
```
# Eliminem els gens que tenen low counts. Ho fem amb count-per-million (CPM) per evitar l'efecte d
e gens expressats en llibreries més grans en comparació a més petites. Fiquem 0.5 de cut-off en co
m a mínim dues llibreries.
Covid_75_dge <- DGEList(counts = assay(Covid_75))
keep <- rowSums(cpm(Covid_75_dge) > 0.5) >= 2
table(keep)

## keep
## FALSE  TRUE
## 32950 24652

Covid_75_dge <- Covid_75_dge[keep,]
```

```
# Filtrem el summarized experiment també
Covid_75 <- Covid_75[keep,]
```

Observem que 32950 gens es perden. En l'estudi treballarem amb 24652.

3.2. EXPLORING

```
# Primer calculem el nombre total de counts per mostra
sumdata <- data.frame(
  value = round(colSums(assay(Covid_75)) / 1e6, 1)  # total en milions, amb un decimal
)
sumdata$key <- colnames(Covid_75)
sumdata$col <- as.character(Covid_75$cohort) # Fiquem color per columna

ggplot(data = sumdata, aes(x = key, y = value, fill = col)) +
  geom_bar(stat = "identity") +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1,size = 4.5), axis.ticks.x = element_blank
()) +
  labs(x = "Samples", y = "Million Reads") +
  geom_hline(yintercept = 20, linetype = 2, color = "gray56") +
  scale_fill_manual(values = c("COVID_19" = "#CBC6B9", "Bacterial" = "#b0c4b1", "healthy" = "#4a57
59"))
```

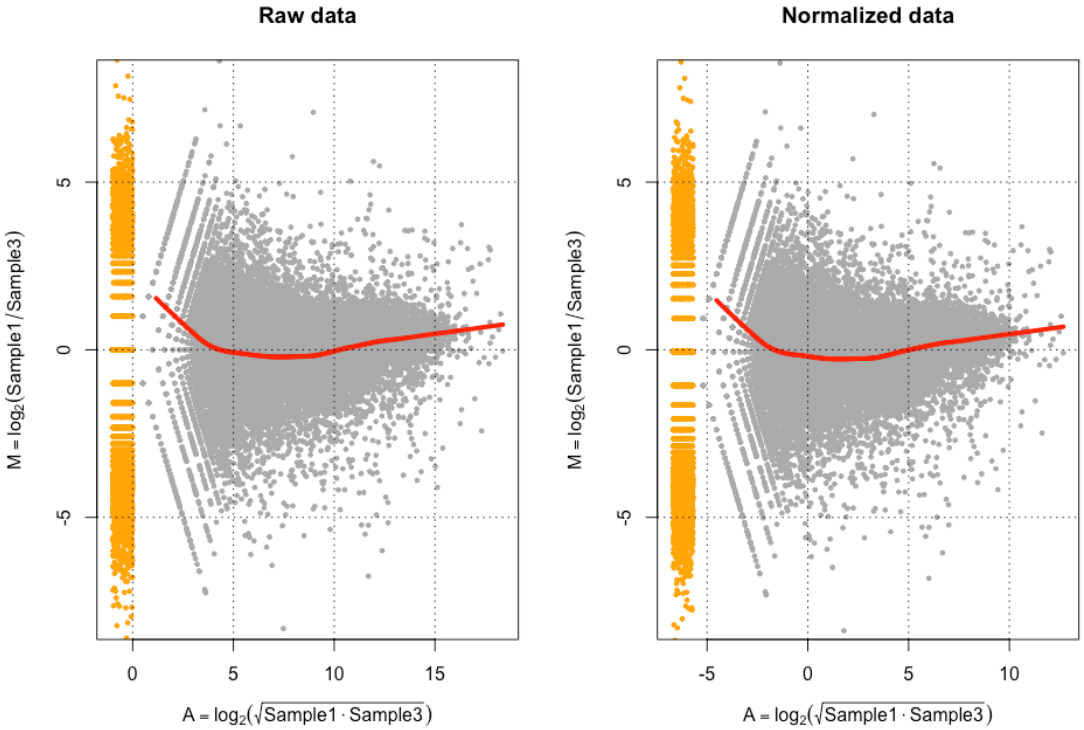


Observem la distribució de les diferents mostres. Observem com es distribueixen. Per poder fer DEGs cal que normalitzem les mostres.

3.3. NORMALITZATION

```
# Normalitzarem amb el mètode trimmed mean of M values (TMM) fent servir 'calcNormFactors'.
Covid_75_dge_norm <- calcNormFactors(Covid_75_dge, method = "TMM")

# Guardem counts normalitzats (no log)
norm_counts <- cpm(Covid_75_dge_norm)
assays(Covid_75)[["normalized"]] <- norm_counts
```

```
# Guardem logCPM per separat per si calen
logCPM <- cpm(Covid_75_dge_norm, log = TRUE, normalized.lib.sizes = TRUE)

# MA plots
par(mfrow = c(1, 2))
maPlot(assay(Covid_75,"counts")[,1], assay(Covid_75,"counts")[,3], pch=19, cex=.5, ylim=c(-8,8),
       allCol="darkgray", lowess=TRUE,
       xlab=expression( A == log[2] (sqrt(Sample1 %.% Sample3))  ),
       ylab=expression(M == log[2](Sample1/Sample3)))
grid(col="black")
title("Raw data")

maPlot(assay(Covid_75,"normalized")[,1], assay(Covid_75,"normalized")[,3], pch=19, cex=.5, ylim=c(
-8,8),
       allCol="darkgray", lowess=TRUE,
       xlab=expression( A == log[2] (sqrt(Sample1 %.% Sample3))),
       ylab=expression(M == log[2](Sample1/Sample3)))
grid(col="black")
title("Normalized data")
```



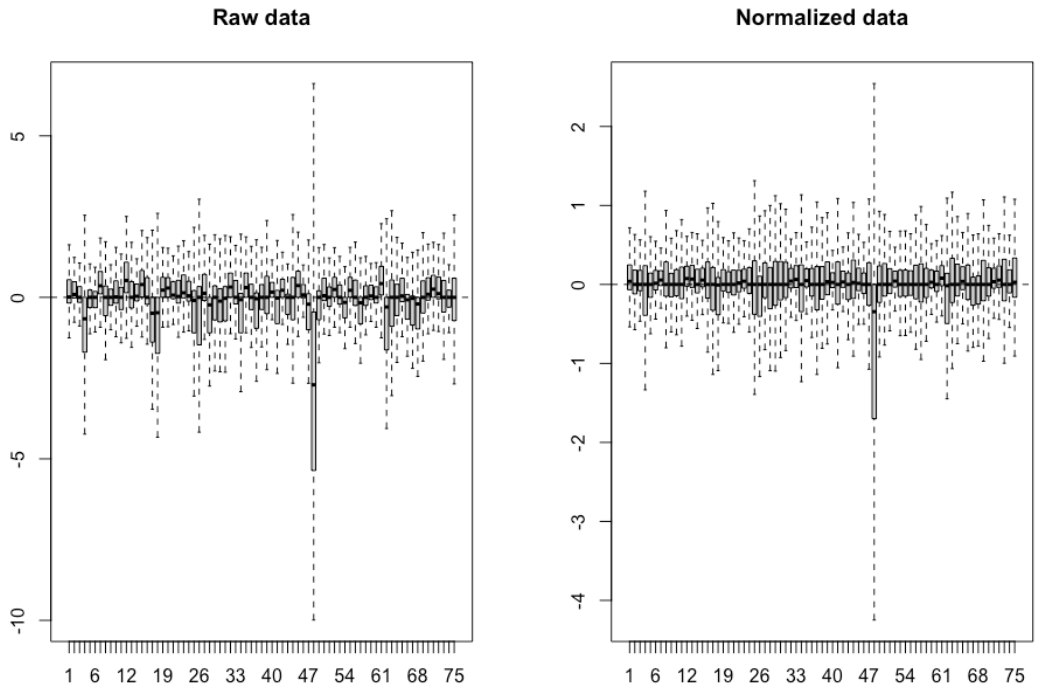

```
# No hi ha grans canvis però norm factor és proper a 1 sempre, pe tant, no cal gran normalització
head(Covid_75_dge_norm$samples)
```

```
##                 group lib.size norm.factors
## DU18_02S0011631     1 39668161    1.2723302
## DU18_02S0011646     1 52795404    1.2674028
## DU18_02S0011667     1 49451505    1.0673963
## DU09_02S0000151     1 32402192    0.7956480
## DU18_02S0011641     1 48971810    1.1383898
## DU18_02S0011649     1 46458834    0.9349759
```

```
summary(Covid_75_dge_norm$samples$norm.factors)
```

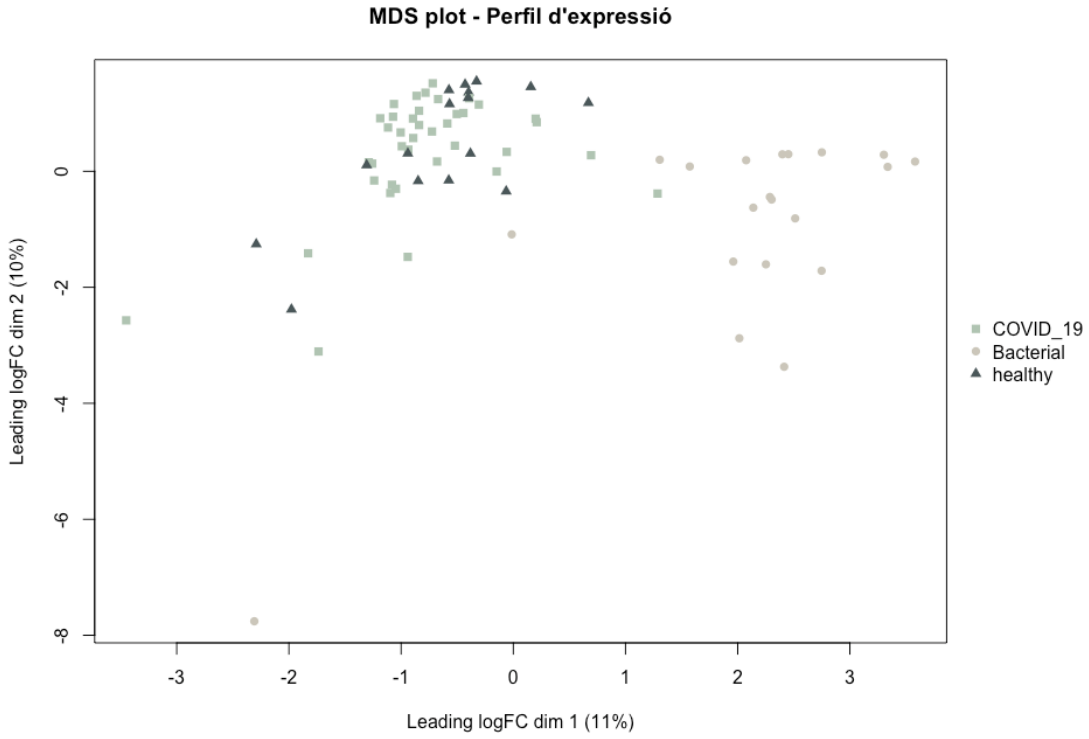
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.5788  0.8837  1.0497  1.0194  1.1607  1.3538
```

```
# Boxplots RLE
par(mfrow = c(1, 2))
EDASeq::plotRLE(as.matrix(assay(Covid_75,"counts")), outline=FALSE, names=NULL)
title("Raw data")

EDASeq::plotRLE(as.matrix(assay(Covid_75,"normalized")), outline=FALSE, names=NULL)
title("Normalized data")
```



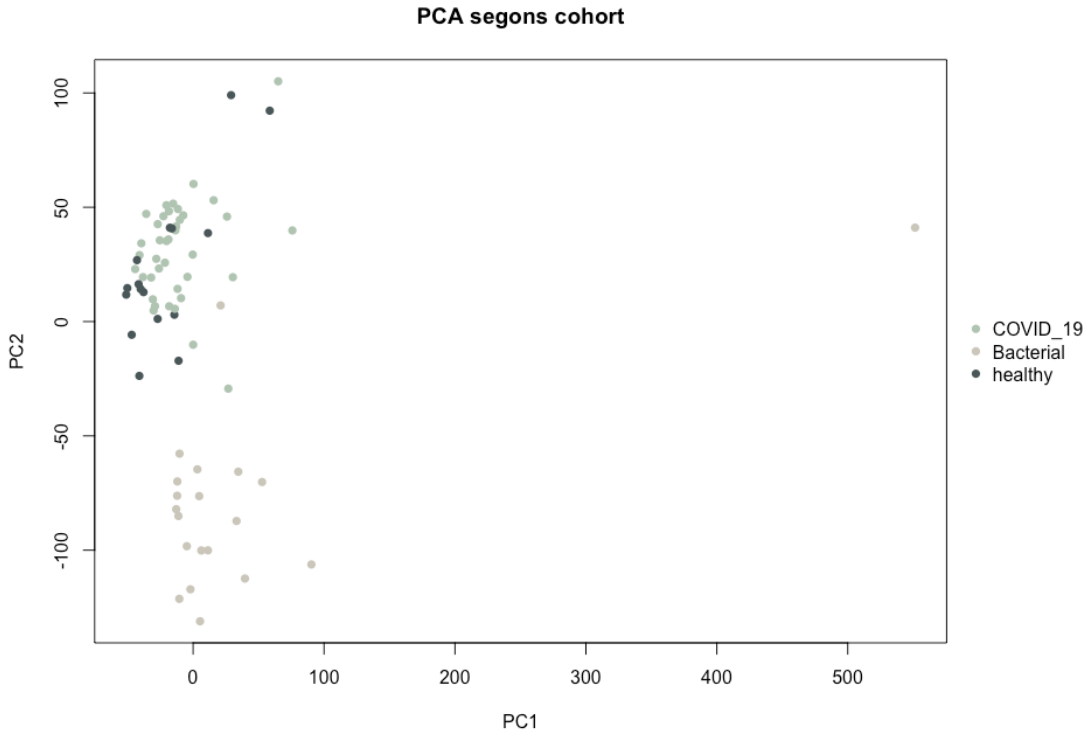
4. EXPLORATORY ANALYSIS

```
# Podem fer-ho MDS que ens permet fer exploració global i control de qualitat
group <- Covid_75$cohort
pch <- c(15, 16, 17)
colors <- rep(c("#b0c4b1", "#CBC6B9", "#4a5759"), length.out=length(levels(group)))
group <- factor(group, levels = c("COVID_19", "Bacterial", "healthy"))

# MDS plot
par(mar = c(5, 4, 4, 8), xpd = TRUE)
plotMDS(Covid_75_dge_norm, col = colors[group], pch = pch[group], main = "MDS plot - Perfil d'expr
essió")
legend("right",inset = c(-0.17, 0), legend = levels(group), col = colors, pch = pch,bty = "n")
```



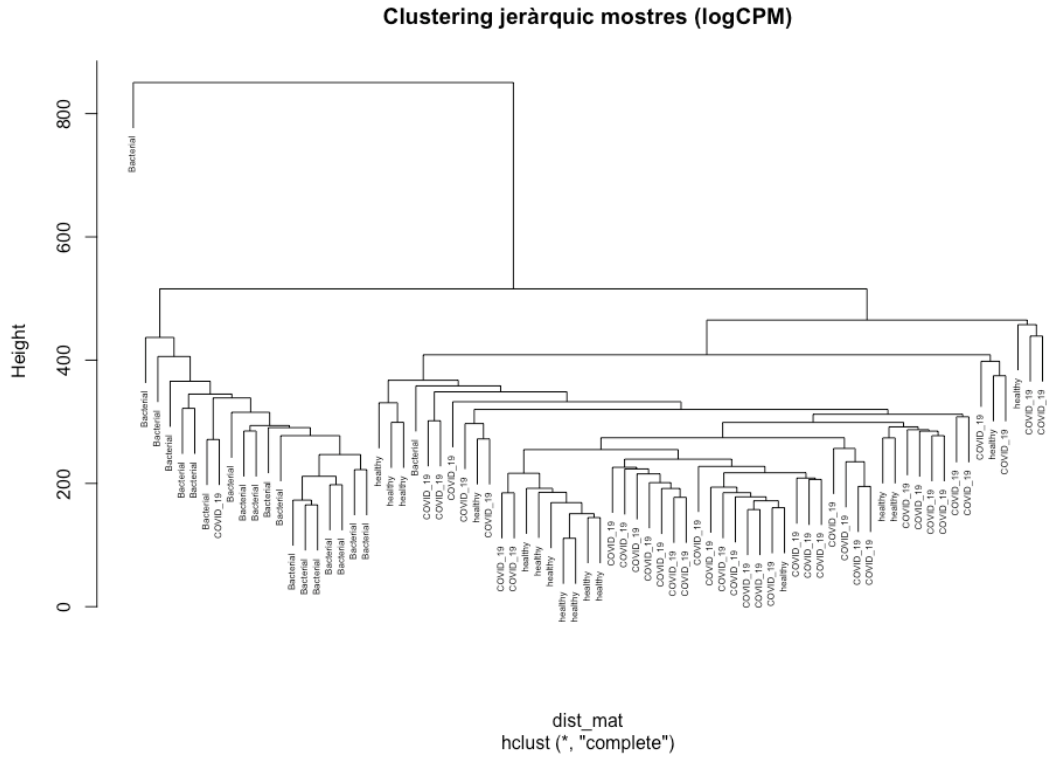
Observem que sembla que hi ha un outlier dintre del grup de Bacterial. A part, dona la sensació que les mostres de COVID_19 i healthy s'agrupen similar, mentre que Bacterial més diferent.

```
#Podem fer també PCA
par(mar = c(5, 4, 4, 8), xpd = TRUE)
pca <- prcomp(t(logCPM), scale. = TRUE)
cols <- c("COVID_19" = "#b0c4b1", "Bacterial" = "#CBC6B9", "healthy" = "#4a5759")
plot(pca$x[,1], pca$x[,2], col=cols[group], pch=16,
     xlab="PC1", ylab="PC2", main="PCA segons cohort")
legend("right", inset = c(-0.17, 0),legend=levels(group), col=cols, pch=16,bty = "n")
```

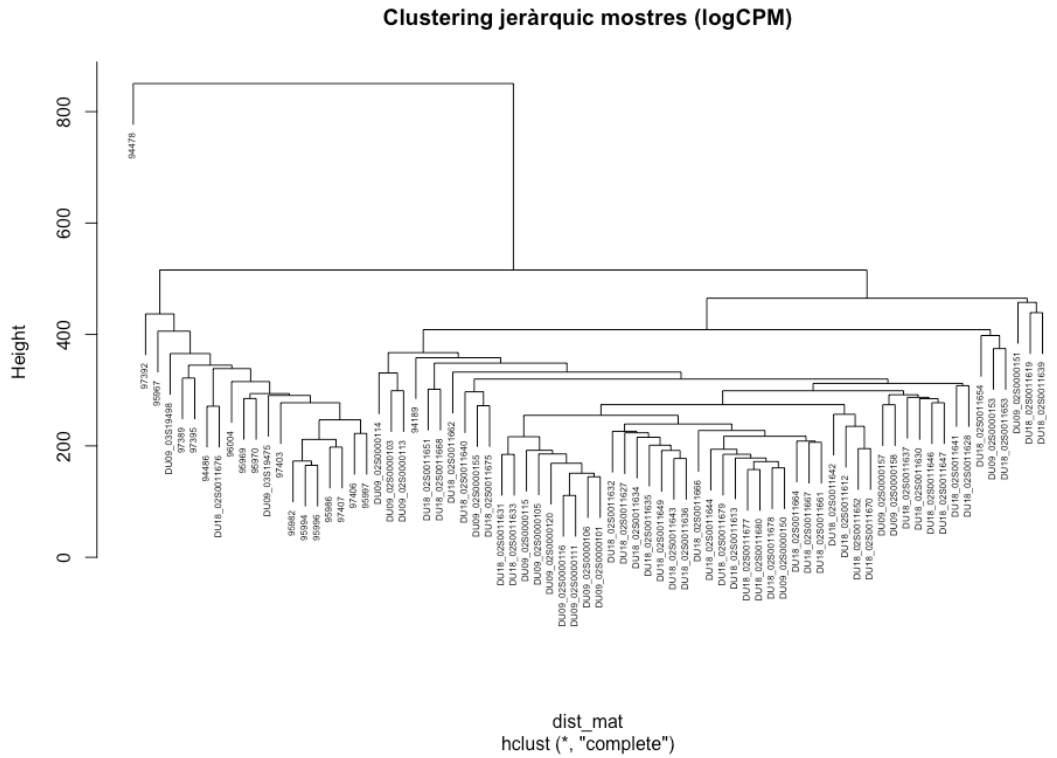


Observem el mateix que en el cas anterior, sembla que Bacterial és el més diferenciat, i que hi ha un outlier.

```
# Clustering
dist_mat <- dist(t(logCPM))
hc <- hclust(dist_mat)
  plot(hc, labels=Covid_75$cohort, main="Clustering jeràrquic mostres (logCPM)", cex=0.5)
```

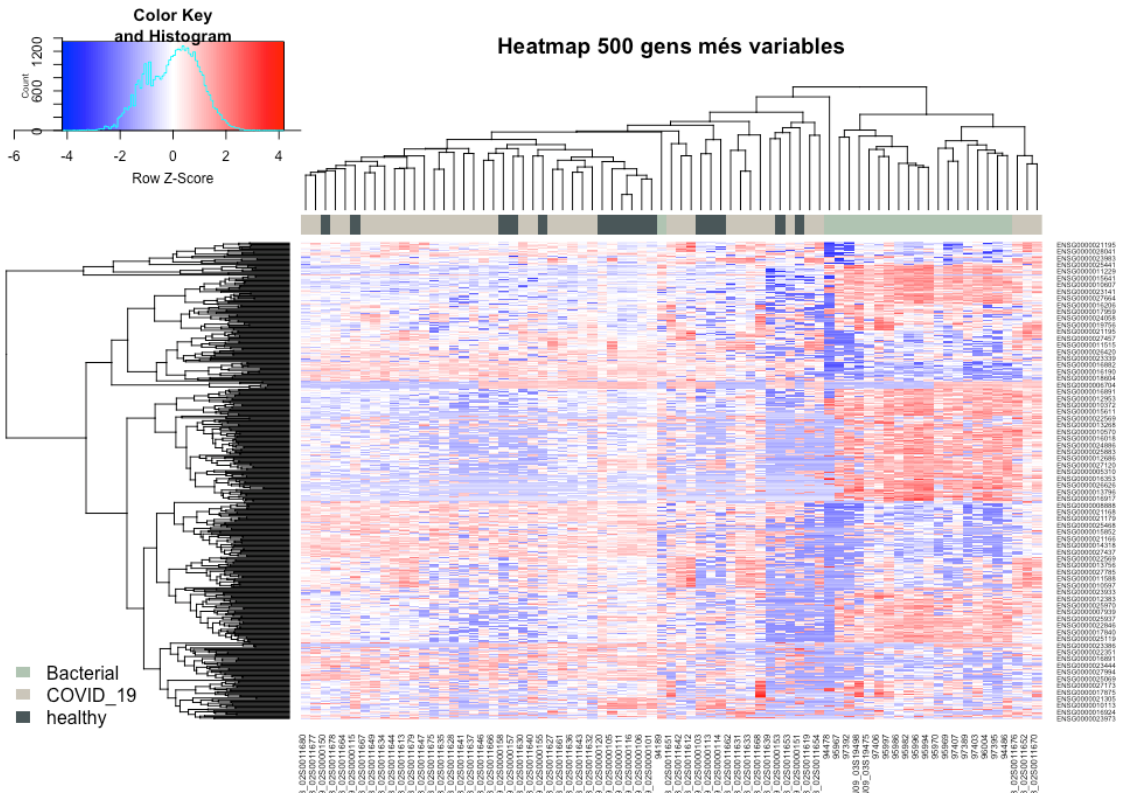


```
  plot(hc, labels=Covid_75$rna_id, main="Clustering jeràrquic mostres (logCPM)", cex=0.5)
```



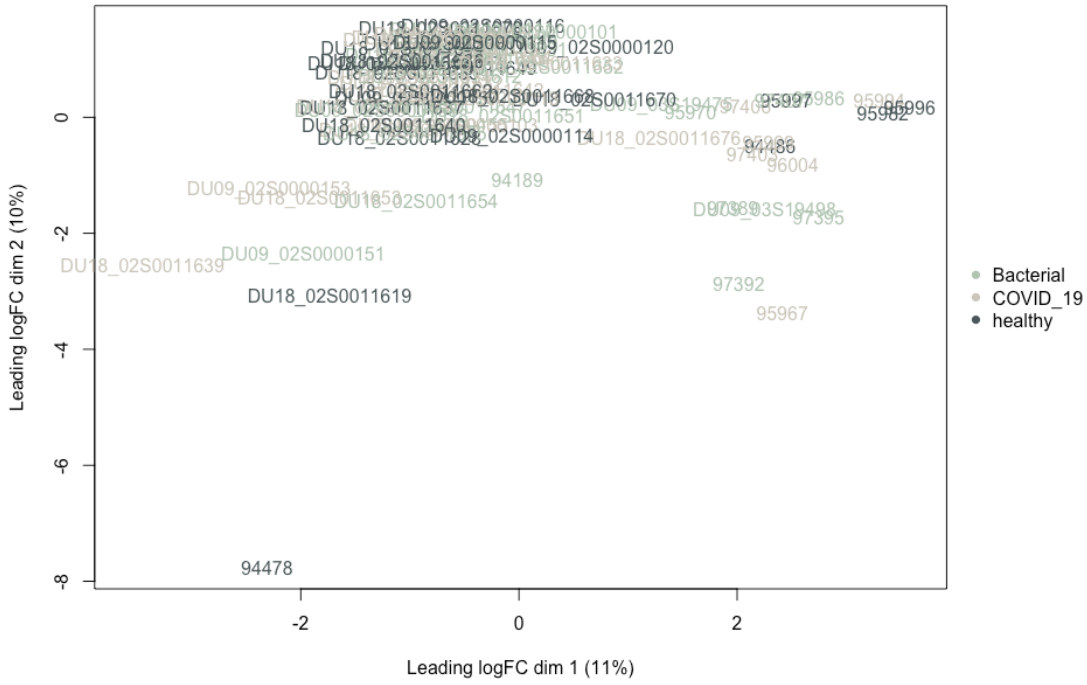
Seguim observant que les mostres de healhty i covid s'agrupen juntes, mentre que bacterial es situa per separat. A part, a bacterial seguim observant l'outlier.

```
var_genes <- apply(logCPM, 1, var)
top_var_genes <- names(sort(var_genes, decreasing=TRUE))[1:500]

# Colors per cohorts
library(RColorBrewer)
cols <- c("COVID_19" = "#b0c4b1", "Bacterial" = "#CBC6B9", "healthy" = "#4a5759")
group <- Covid_75$cohort
sample_colors <- cols[group]

# Heatmap amb heatmap.2
par(mar = c(5, 4, 4, 8), xpd = TRUE)
heatmap.2(logCPM[top_var_genes, ], trace="none", col=bluered(100),
          ColSideColors=sample_colors,
          main="Heatmap 500 gens més variables",
          scale="row")
legend("bottomleft", inset = c(-0.1, -0.1), legend=levels(group), fill=cols, border=NA, bty="n")
```



Com hem vist amb la anàlisis exploratòria, tenim un outlier. Per aquest motiu, caldrà eliminar-ho dels anàlisis. Podem detectar-ho per MDS i PCA.

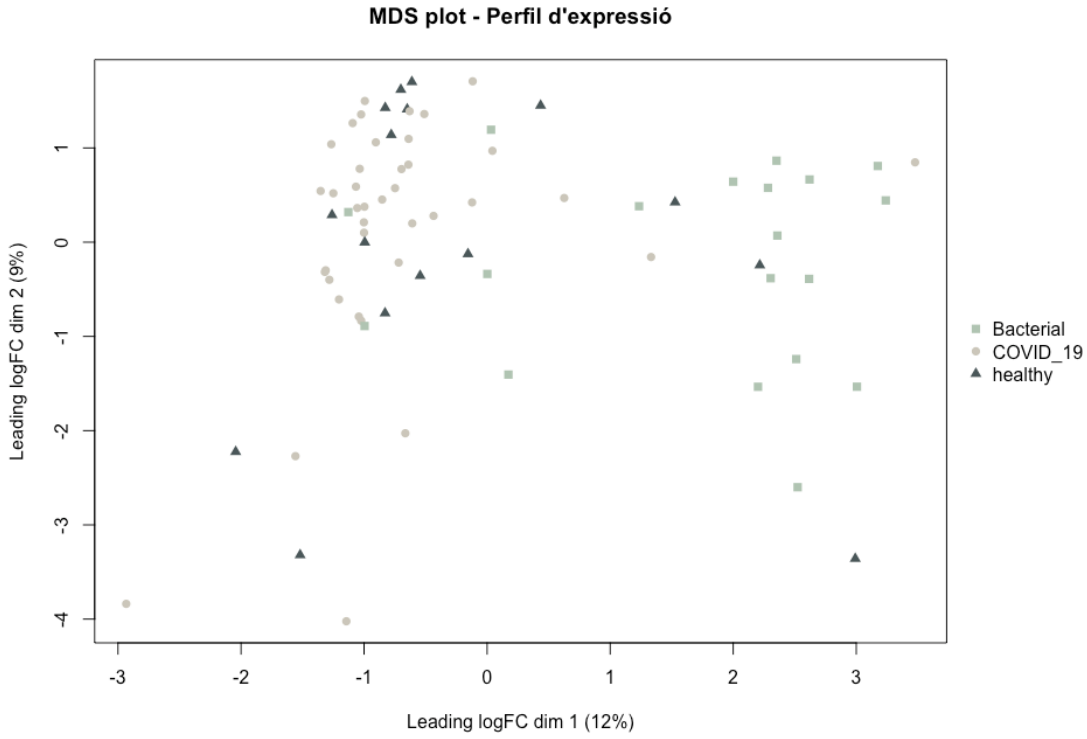
```
# Plotegem per veure quin valor té l'outlier.
par(mar = c(5, 4, 4, 8), xpd = TRUE)
plotMDS(Covid_75_dge_norm, col=c("COVID_19" = "#b0c4b1", "Bacterial" = "#CBC6B9", "healthy" = "#4a
5759"))
legend("right", inset = c(-0.17, 0),legend=levels(group), col=cols, pch=16,bty = "n")
```




```
# Veiem que outlier és 94478, el podem eliminar.

outlier_sample <- "94478"
Covid_75_dge_clean <- Covid_75_dge_norm[, colnames(Covid_75_dge_norm) != outlier_sample]
Covid_75_clean <- Covid_75[, colnames(Covid_75) != outlier_sample]
```

Un cop ho hem eliminat, podem repetir els gràfics anteriors per veure si la distribució és millor. Com a mode d'exemple, farem el de MDS.

```
par(mar = c(5, 4, 4, 8), xpd = TRUE)
# MDS plot
plotMDS(Covid_75_dge_clean, col = colors[group], pch = pch[group], main = "MDS plot - Perfil d'exp
ressió")
legend("right", inset = c(-0.17, 0), legend = levels(group), col = colors, pch = pch, bty = "n")
```



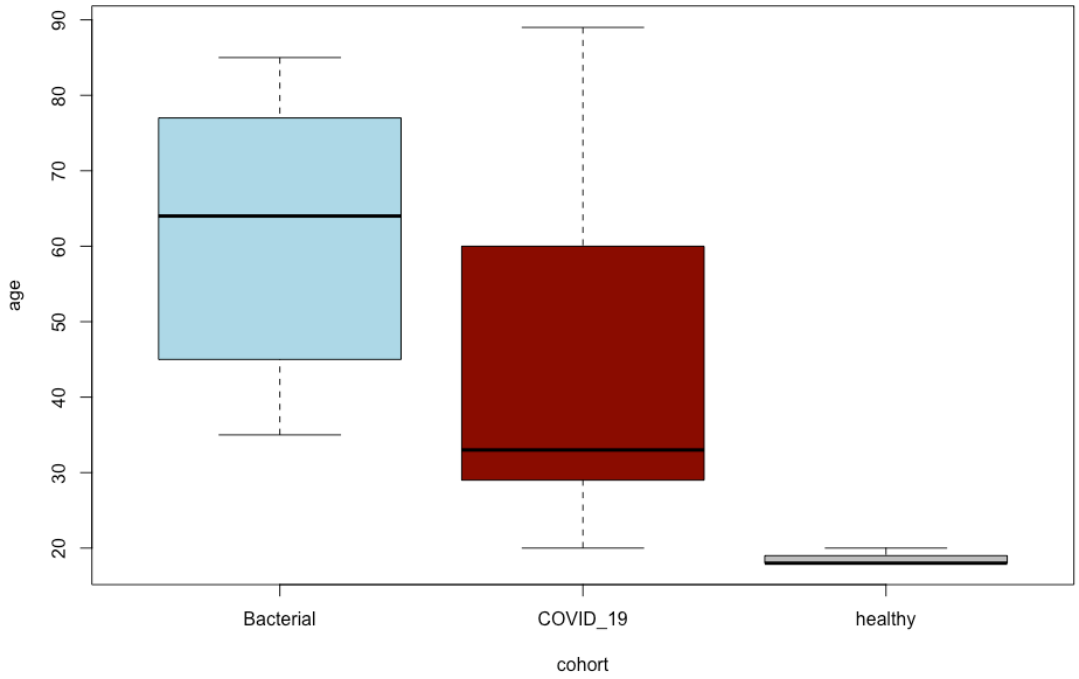
Ara observem que les mostres es distribueixen millor, i que les agrupacions no depenen de l'outlier anterior.

4.1. CONFUSION VARIABLES

```
# Primer de tot mirem quines possibles variables confusores pot haver
colnames(colData(Covid_75))
```

```
## [1] "rna_id"           "subject_id"       "age"              "gender"
## [5] "race"             "cohort"           "time_since_onset" "hospitalized"
## [9] "batch"
```

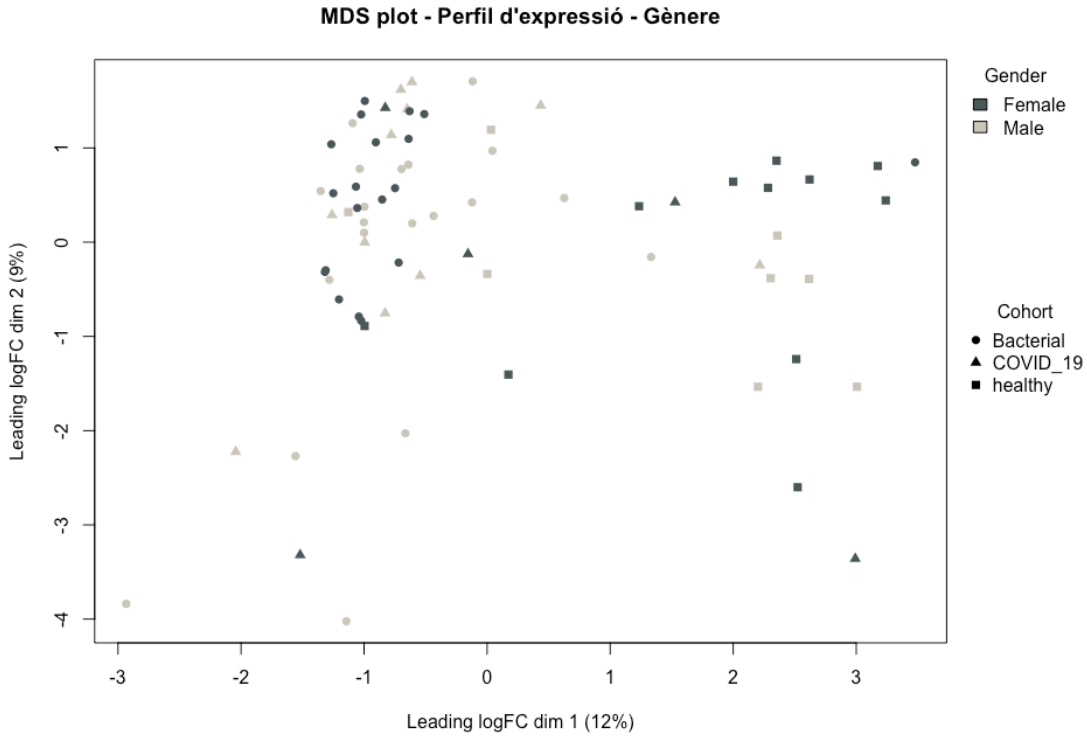
```
# De les que tenim, algunes podem ser Age, gender, race, time_since_onset, hospitalized. Fem gràfi
cs MDS (boxplots per variables numeriques) per cada un d'aquests. Preparem les variables primer.

# Edat
boxplot(age ~ cohort, data = colData(Covid_75), col=c("lightblue", "darkred","grey"))
```




```
anova_result <- aov(age ~ cohort, data = colData(Covid_75))
summary(anova_result)

##             Df Sum Sq Mean Sq F value   Pr(>F)
## cohort       2  16504    8252   30.37 2.73e-10 ***
## Residuals   72  19564     272
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Pot ser variable confusora, bacterians son més grans.

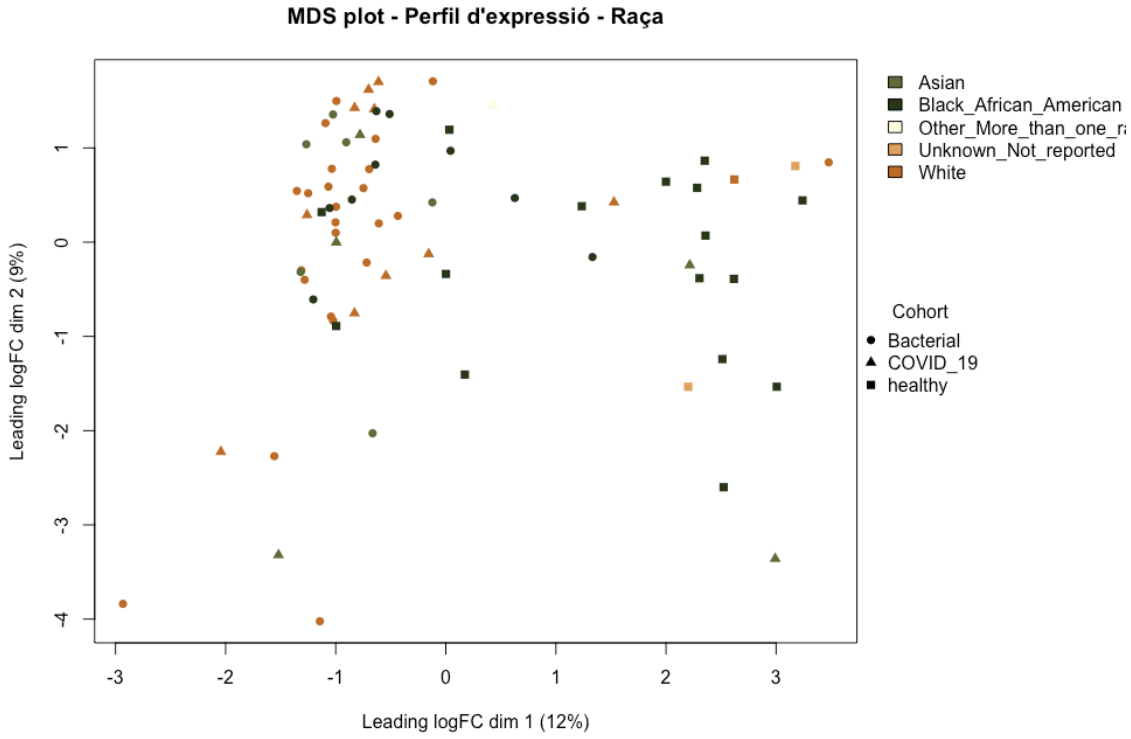
```
# Gender
gender <- colData(Covid_75)$gender
gender <- as.factor(gender)
colors_gender <- c("#4a5759", "#CBC6B9")[1:nlevels(gender)]
col_vector <- colors_gender[gender]
par(mar = c(5, 4, 4, 8), xpd = TRUE)
plotMDS(Covid_75_dge_clean, col = col_vector, pch = pch[group], main = "MDS plot - Perfil d'expres
sió - Gènere")
legend("topright",inset = c(-0.15, 0), legend = levels(gender), fill = colors_gender, title = "Gen
der",bty = "n")
legend("right",inset = c(-0.17, 0), legend = levels(factor(group)), pch = unique(pch[group]),title
 = "Cohort",bty = "n")
```




```
table(colData(Covid_75)$cohort, colData(Covid_75)$gender)

##
##             Female Male
##   Bacterial     12    8
##   COVID_19      19   20
##   healthy        5   11

chisq.test(table(colData(Covid_75)$cohort, colData(Covid_75)$gender))

##
##  Pearson's Chi-squared test
##
## data:  table(colData(Covid_75)$cohort, colData(Covid_75)$gender)
## X-squared = 2.9604, df = 2, p-value = 0.2276
```

No sembla que porti a confusió.

```
# Race
race <- colData(Covid_75)$race
race <- as.factor(race)
colors_race <- c("#606c38", "#283618", "#fefae0", "#dda15e", "#bc6c25" )[1:nlevels(race)]
col_vector_race <- colors_race[race]
par(mar = c(5, 4, 4, 12), xpd = TRUE)
plotMDS(Covid_75_dge_clean, col = col_vector_race, pch = pch[group], main = "MDS plot - Perfil d'e
xpressió - Raça")
legend("topright", inset = c(-0.4, 0),legend = levels(race), fill = colors_race,bty = "n")
legend("right", inset = c(-0.17, 0),legend = levels(factor(group)), pch = unique(pch[group]),title
 = "Cohort",bty = "n")
```

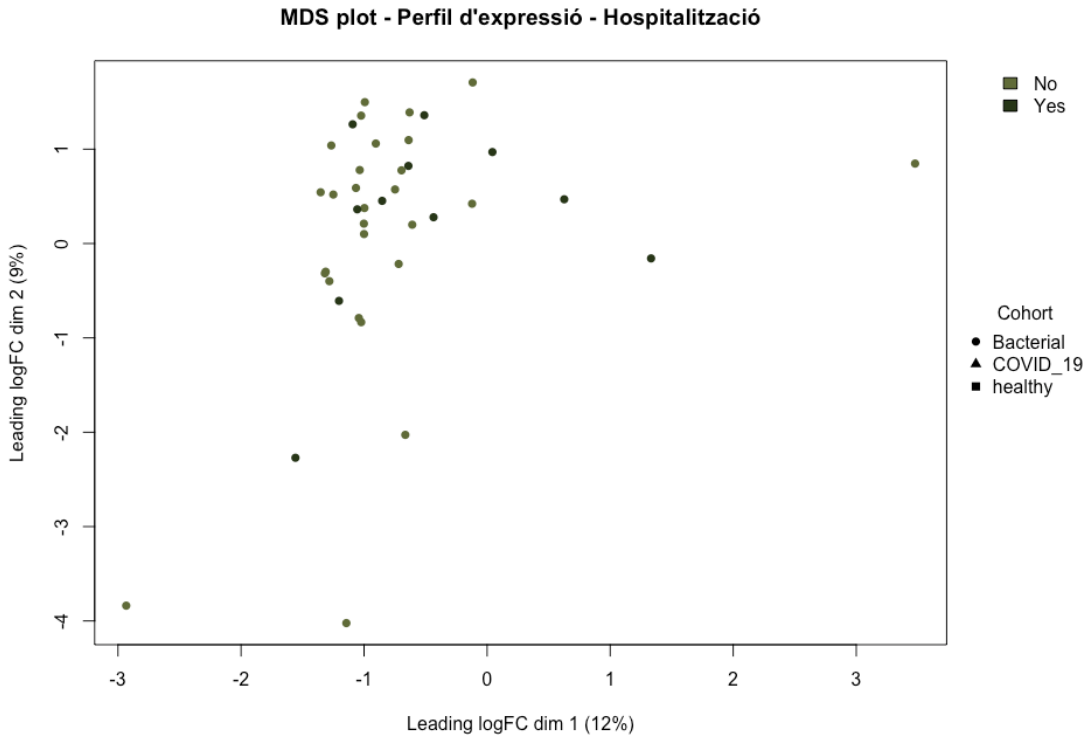



```
table(colData(Covid_75)$cohort, colData(Covid_75)$race)

##
##             Asian Black_African_American Other_More_than_one_race
##   Bacterial     0                    16                        0
##   COVID_19      6                     9                        0
##   healthy       5                     0                        1
##
##             Unknown_Not_reported White
##   Bacterial                    2     2
##   COVID_19                     0    24
##   healthy                      0    10

chisq.test(table(colData(Covid_75)$cohort, colData(Covid_75)$race))

## Warning in chisq.test(table(colData(Covid_75)$cohort, colData(Covid_75)$race)):
## Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  table(colData(Covid_75)$cohort, colData(Covid_75)$race)
## X-squared = 42.972, df = 8, p-value = 8.891e-07
```

Sembla que raça pot portar a confusió, cal tenir en compte.

```
# Hospitalitzats
hospitalized <- colData(Covid_75)$hospitalized
hospitalized <- as.factor(hospitalized)
colors_hosp <- c("#606c38", "#283618", "#fefae0")[1:nlevels(hospitalized)]
col_vector_hosp <- colors_hosp[hospitalized]
par(mar = c(5, 4, 4, 8), xpd = TRUE)
plotMDS(Covid_75_dge_clean, col = col_vector_hosp, pch = pch[group], main = "MDS plot - Perfil d'e
xpressió - Hospitalització")
legend("topright", inset = c(-0.15, 0), legend = levels(hospitalized), fill = colors_hosp, bty = "
n")
```

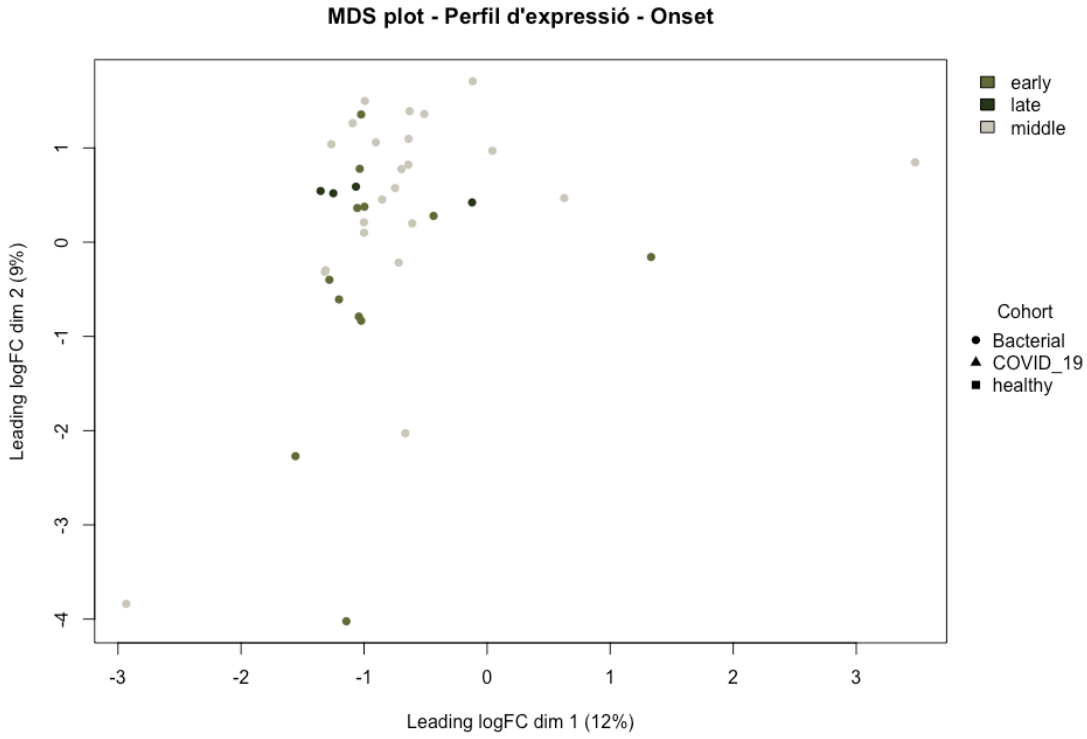
```
legend("right",inset = c(-0.17, 0),  legend = levels(factor(group)), pch = unique(pch[group]),titl
e = "Cohort",bty = "n")
```




```
table(colData(Covid_75)$cohort, colData(Covid_75)$hospitalized)

##
##             No Yes
##   Bacterial  0   0
##   COVID_19  28  11
##   healthy    0   0

chisq.test(table(colData(Covid_75)$cohort, colData(Covid_75)$hospitalized))

## Warning in chisq.test(table(colData(Covid_75)$cohort,
## colData(Covid_75)$hospitalized)): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  table(colData(Covid_75)$cohort, colData(Covid_75)$hospitalized)
## X-squared = NaN, df = 2, p-value = NA
```

Observem que hospitalització tampoc és un factor a tenir en compte.

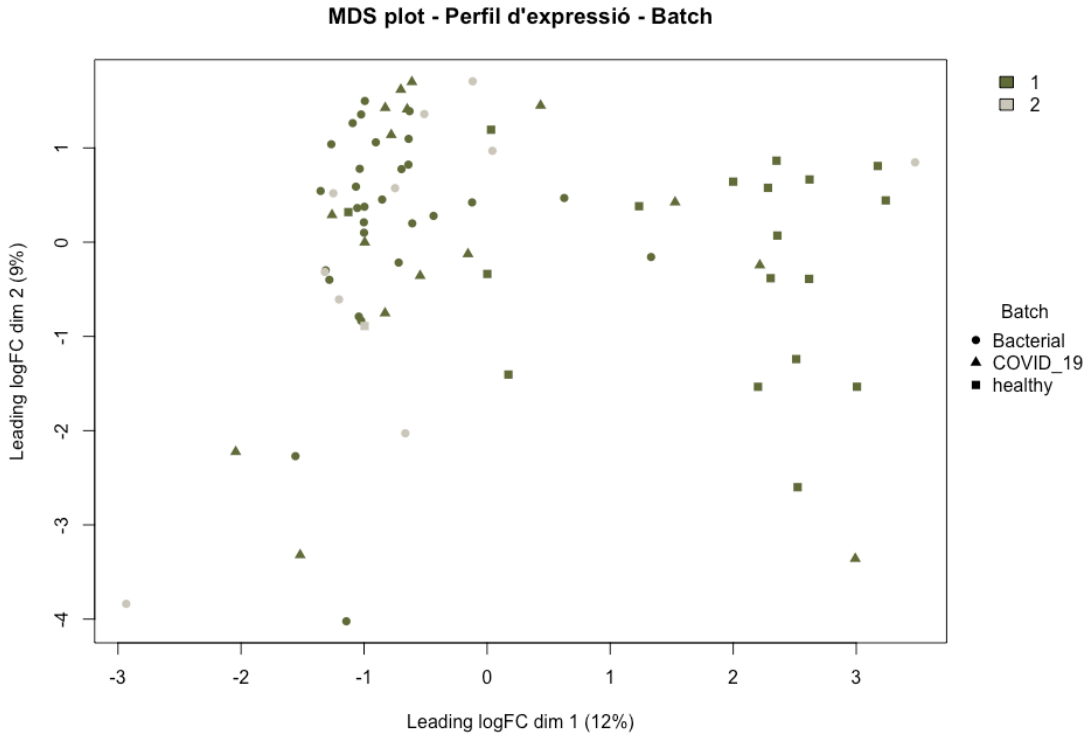
```
# Days since onset
onset <- colData(Covid_75)$time_since_onset
onset <- as.factor(onset)
colors_onset <- c("#606c38", "#283618", "#CBC6B9")[1:nlevels(onset)]
col_vector_onset <- colors_onset[onset]
par(mar = c(5, 4, 4, 8), xpd = TRUE)
plotMDS(Covid_75_dge_clean, col = col_vector_onset, pch = pch[group], main = "MDS plot - Perfil d'
expressió - Onset")
legend("topright", inset = c(-0.15, 0),legend = levels(onset), fill = colors_onset,bty = "n")
legend("right",inset = c(-0.17, 0), legend = levels(factor(group)), pch = unique(pch[group]),title
 = "Cohort",bty = "n")
```




```
table(colData(Covid_75)$cohort, colData(Covid_75)$time_since_onset)

##
##             early late middle
##   Bacterial     0    0      0
##   COVID_19     12    4     23
##   healthy       0    0      0

chisq.test(table(colData(Covid_75)$cohort, colData(Covid_75)$time_since_onset))

## Warning in chisq.test(table(colData(Covid_75)$cohort,
## colData(Covid_75)$time_since_onset)): Chi-squared approximation may be
## incorrect

##
##  Pearson's Chi-squared test
##
## data:  table(colData(Covid_75)$cohort, colData(Covid_75)$time_since_onset)
## X-squared = NaN, df = 4, p-value = NA
```

Observem que aquest paràmetre tampoc és una variable confusora.

```
# Batch
onset <- colData(Covid_75)$batch
onset <- as.factor(onset)
colors_onset <- c("#606c38", "#CBC6B9")[1:nlevels(onset)]
col_vector_onset <- colors_onset[onset]
par(mar = c(5, 4, 4, 8), xpd = TRUE)
plotMDS(Covid_75_dge_clean, col = col_vector_onset, pch = pch[group], main = "MDS plot - Perfil d'
expressió - Batch")
legend("topright", inset = c(-0.12, 0),legend = levels(onset), fill = colors_onset,bty = "n")
legend("right",inset = c(-0.17, 0), legend = levels(factor(group)), pch = unique(pch[group]),title
 = "Batch",bty = "n")
```




```
table(colData(Covid_75)$cohort, colData(Covid_75)$batch)

##
##              1  2
##   Bacterial 19  1
##   COVID_19  29 10
##   healthy   16  0

chisq.test(table(colData(Covid_75)$cohort, colData(Covid_75)$batch))

## Warning in chisq.test(table(colData(Covid_75)$cohort,
## colData(Covid_75)$batch)): Chi-squared approximation may be incorrect

##
##  Pearson's Chi-squared test
##
## data:  table(colData(Covid_75)$cohort, colData(Covid_75)$batch)
## X-squared = 7.9962, df = 2, p-value = 0.01835
```

Hi ha efecte batch que cal tenir en compte en el disseny. Tenint això en compte, no tindrem en compte cap d'aquests variables per fer la matriu de disseny i de contrast en el següent pas.

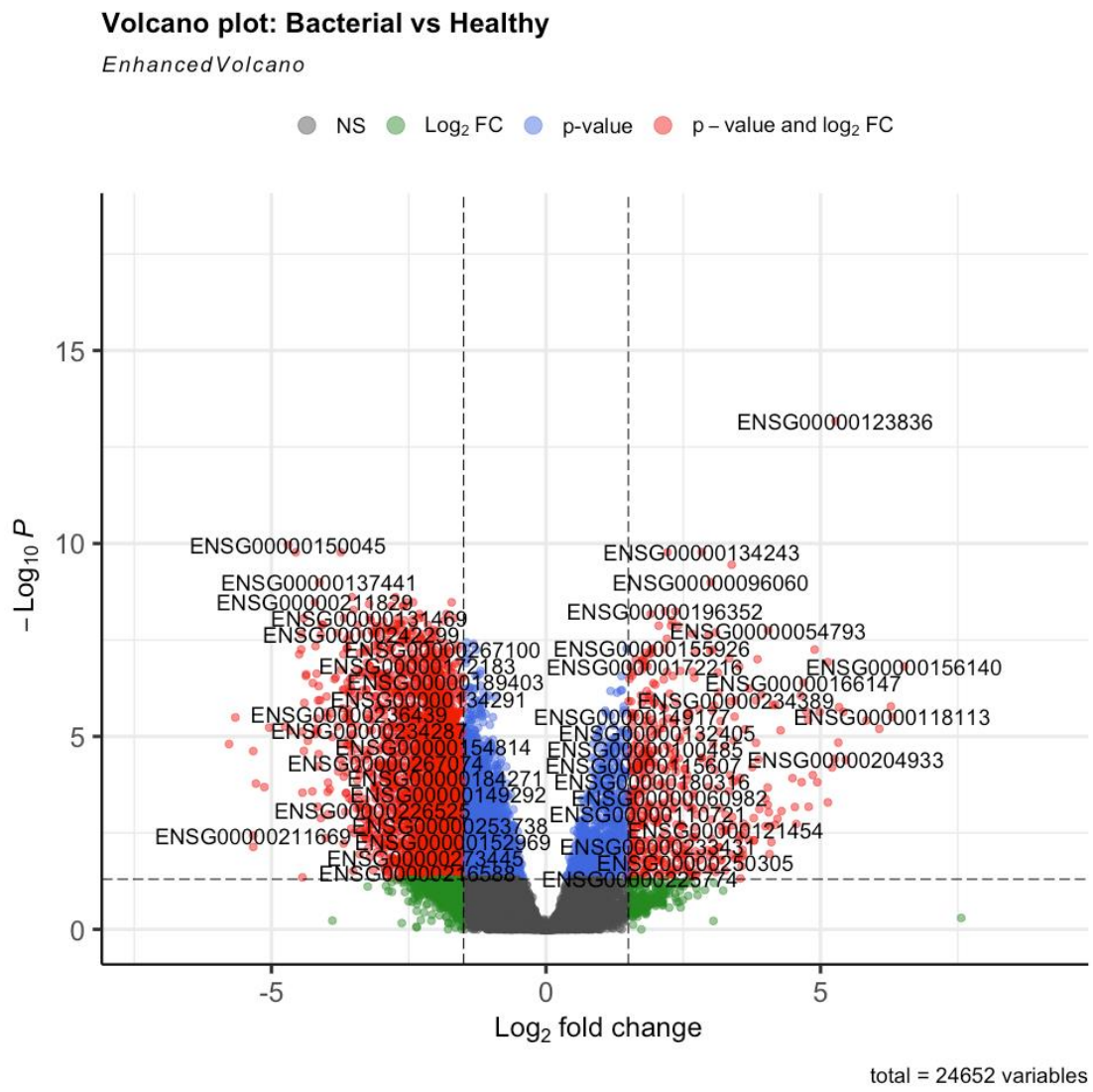
DESIGN AND CONTRAST MATRIX

```
set.seed(myseed)
sample(c("edgeR", "voom+limma", "DESeq2"), size = 1)

## [1] "edgeR"

# Farem servir edgeR.

# Creem matriu disseny amb grup cohort i les variables confusores.
design <- model.matrix(~ 0 + group + age + race + batch, data = colData(Covid_75))
```

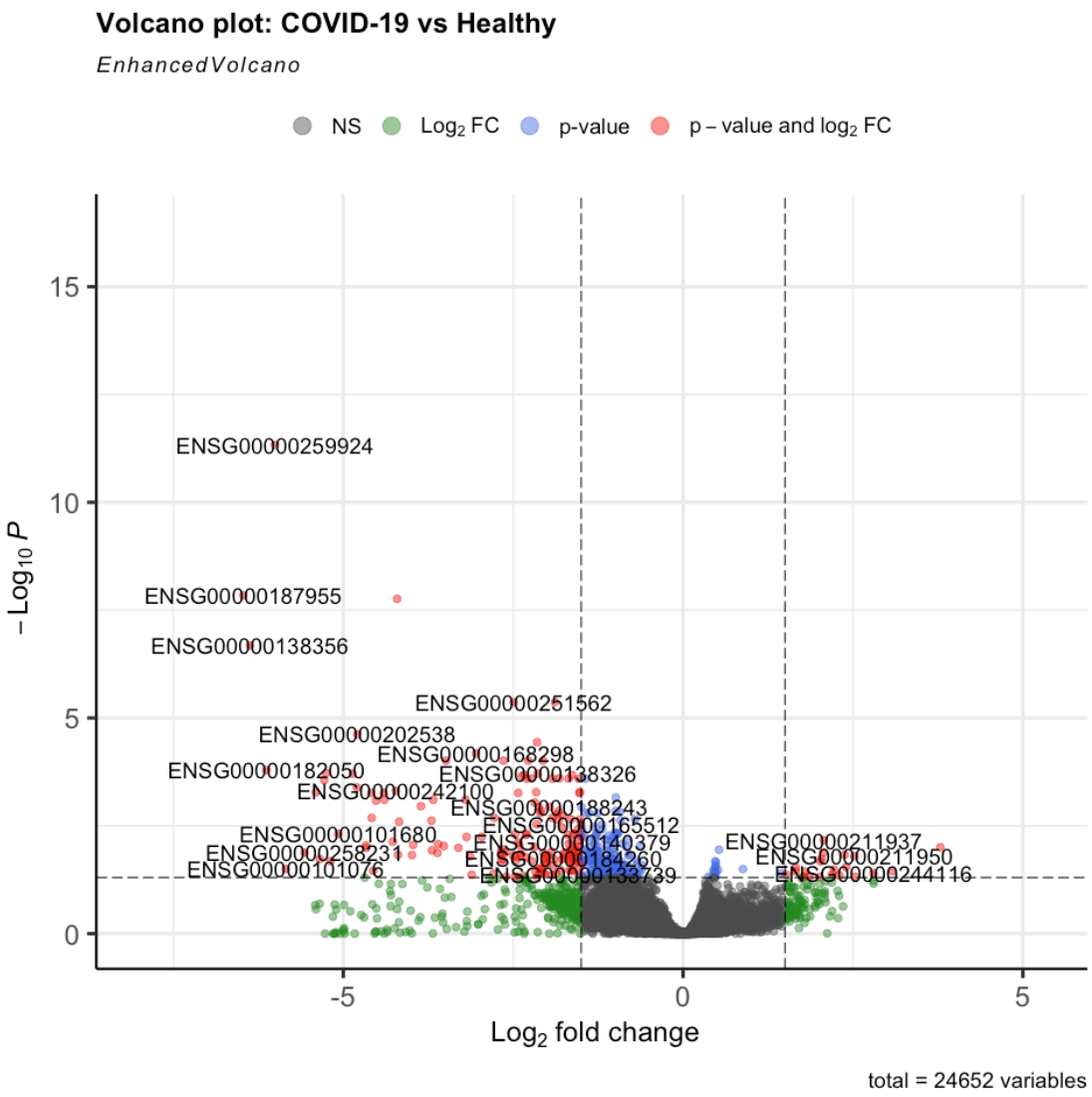
```r
# Assignem noms dels coeficients del disseny
colnames(design)[1:3] <- levels(group)

dge <- DGEList(counts = assay(Covid_75,"normalized"), group = group)

# Estimem la dispersió
dge <- estimateDisp(dge, design)

# Ajustem el model lineal
fit <- glmQLFit(dge, design)
```

5.1. DEG

```r
# Creem matriu de constrastos
contrasts <- makeContrasts( Bacterial_vs_healthy = Bacterial - healthy,
                            COVID19_vs_healthy = `COVID_19` - healthy,levels = design)

# Bacterial vs healthy
lrt_bact_vs_healthy <- glmQLFTest(fit, contrast = contrasts[, "Bacterial_vs_healthy"])

# Extraiem resultats amb logFC > 1.5 i FDR < 0.05
res_bact <- topTags(lrt_bact_vs_healthy, n = Inf)$table

# Volcano plot
EnhancedVolcano(res_bact,
                lab = rownames(res_bact),
                x = 'logFC',
                y = 'FDR',
                pCutoff = 0.05,
                FCcutoff = 1.5, # Cutoff de FC
                title = 'Volcano plot: Bacterial vs Healthy')
```

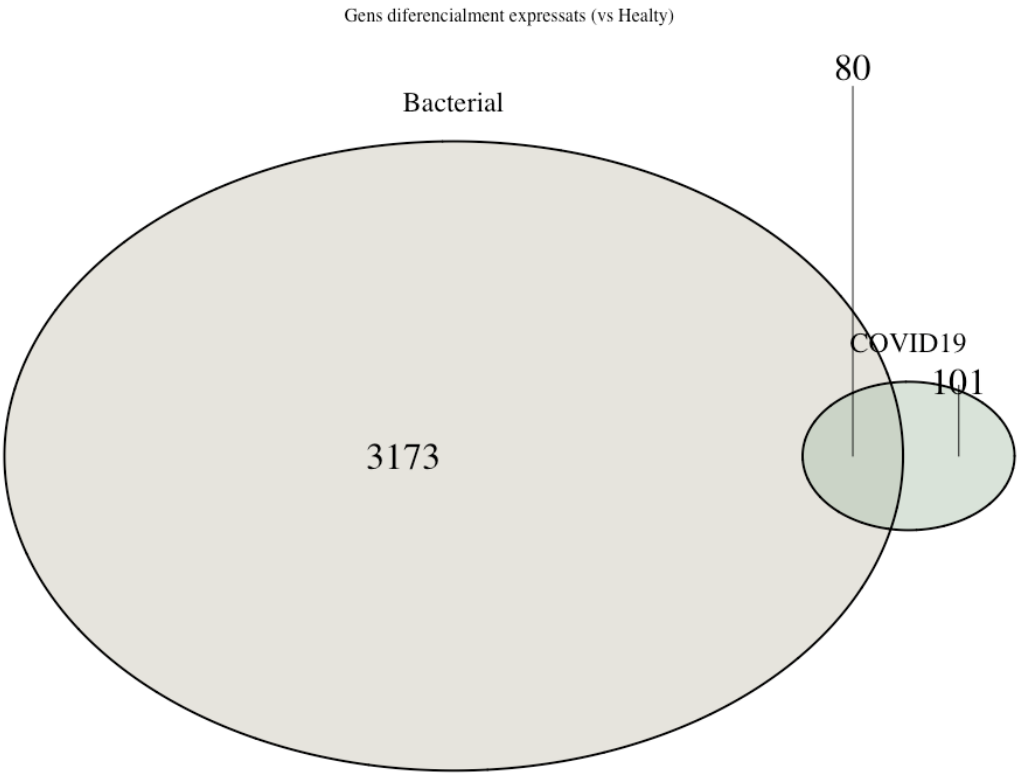



```r
# COVID19 vs healthy
# Prova d'hipòtesis per cada contrast
lrt_covid_vs_healthy <- glmQLFTest(fit, contrast = contrasts[, "COVID19_vs_healthy"])

res_covid <- topTags(lrt_covid_vs_healthy, n = Inf)$table

# Volcano plot
EnhancedVolcano(res_covid,
                lab = rownames(res_covid),
                x='logFC',
                y='FDR',
                pCutoff = 0.05,
                FCcutoff = 1.5, # Cutoff de FC
                title = 'Volcano plot: COVID-19 vs Healthy')
```

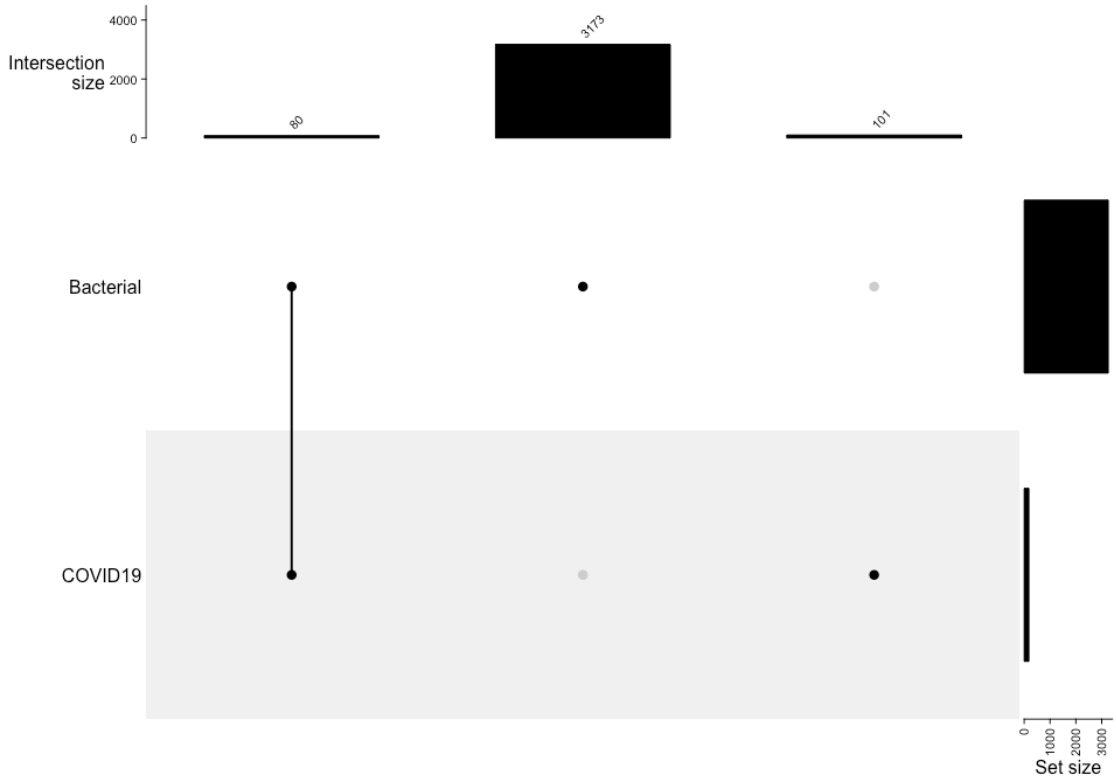


6. BACTERIAL VDS HEALTHY / COVID19 VS HEALTHY

```
genes_bact <- rownames(res_bact)[res_bact$FDR < 0.05 & abs(res_bact$logFC) > 1.5]
genes_covid <- rownames(res_covid)[res_covid$FDR < 0.05 & abs(res_covid$logFC) > 1.5]

venn.plot <- venn.diagram(
  x = list(Bacterial = genes_bact, COVID19 = genes_covid),
  filename = NULL,
  fill = c("#CBC6B9", "#b0c4b1"),
  alpha = 0.5,
  cex = 2,
  cat.cex = 1.5,
  cat.pos = 0,
  main = "Gens diferencialment expressats (vs Healty)"
)

grid.newpage()
grid.draw(venn.plot)
```

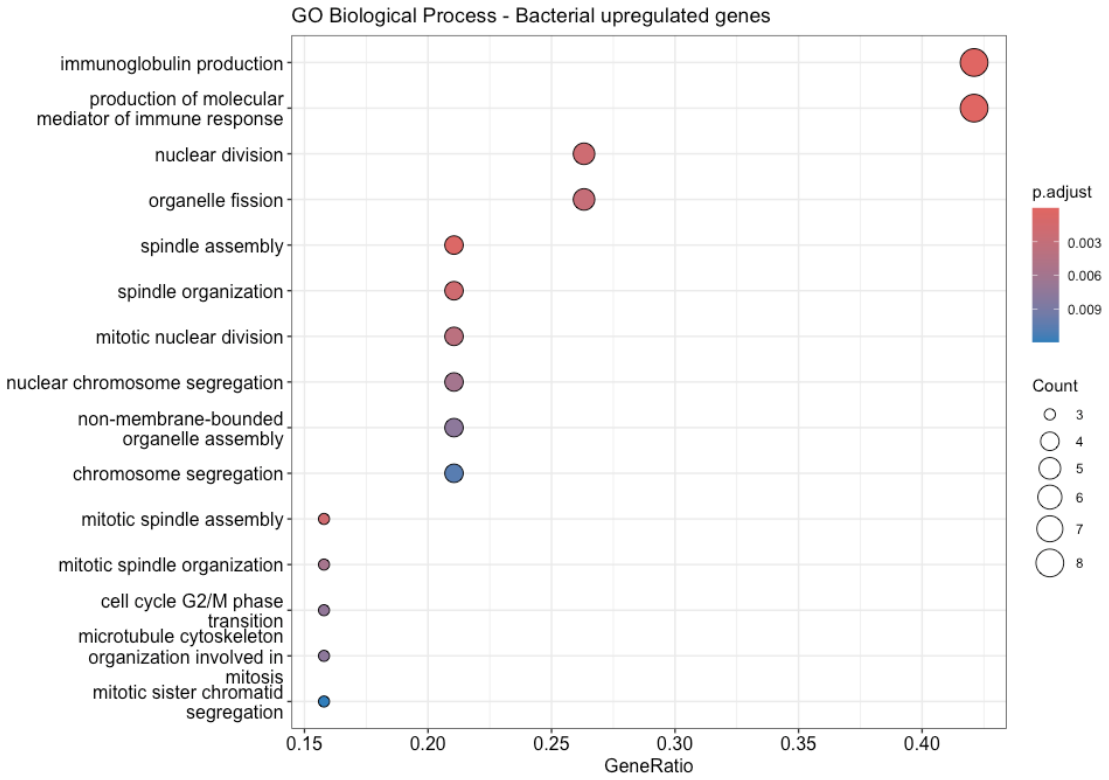



```
#  UpSetR
data_upset <- list(Bacterial= genes_bact,
     COVID19 =genes_covid)
data_upset_matrix <- make_comb_mat(list_to_matrix(data_upset))

UpSet(data_upset_matrix,top_annotation = upset_top_annotation(data_upset_matrix, add_numbers = TRU
E))
```



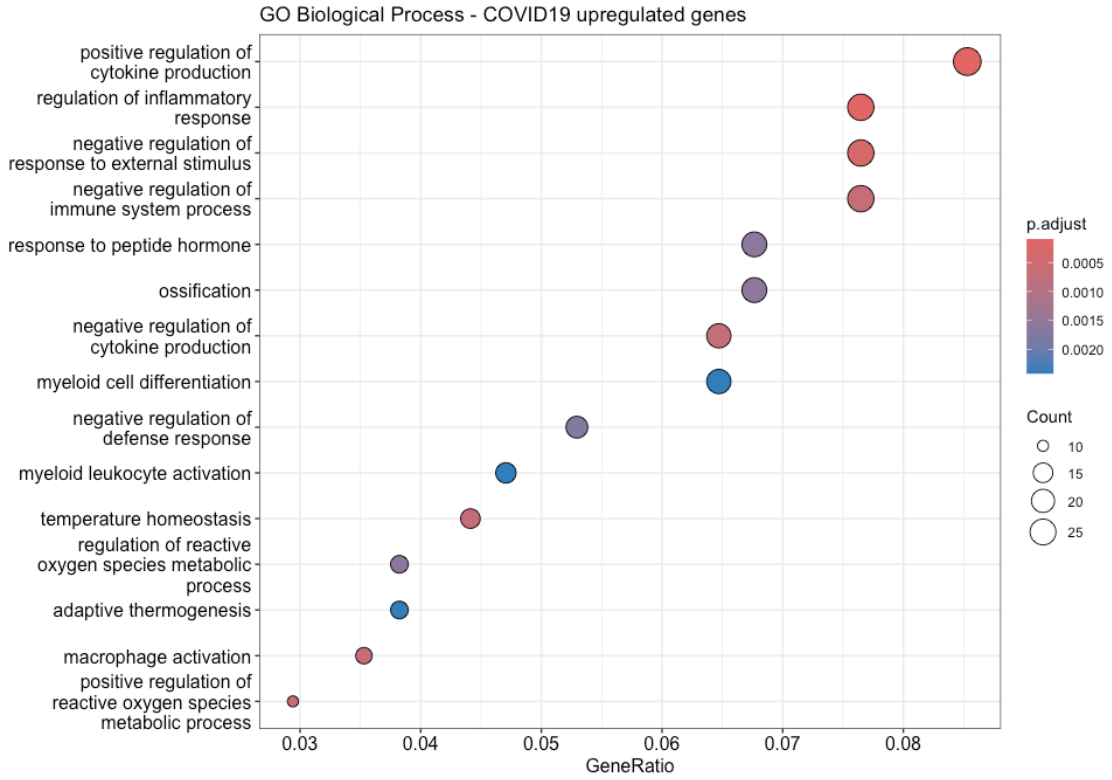
7. GSEA

```
#Covid
genes_covid <- rownames(res_covid)[res_covid$FDR < 0.05 & res_covid$logFC > 1.5]
entrez_up <- bitr(genes_covid, fromType = "ENSEMBL", toType = "ENTREZID", OrgDb = org.Hs.eg.db)

## 'select()' returned 1:1 mapping between keys and columns

ego_up <- enrichGO(gene          = entrez_up$ENTREZID,
                   OrgDb         = org.Hs.eg.db,
                   ont           = "BP",          # Només Biological Process
                   pAdjustMethod = "BH",
                   pvalueCutoff  = 0.05,
                   qvalueCutoff  = 0.2,
                   readable      = TRUE)
# Dotplot
dotplot(ego_up, showCategory = 15, title = "GO Biological Process - Bacterial upregulated genes")
```




```
# Bacterial
genes_bact <- rownames(res_bact)[res_bact$FDR < 0.05 & res_bact$logFC > 1.5]

entrez_up <- bitr(genes_bact, fromType = "ENSEMBL", toType = "ENTREZID", OrgDb = org.Hs.eg.db)

## 'select()' returned 1:many mapping between keys and columns

## Warning in bitr(genes_bact, fromType = "ENSEMBL", toType = "ENTREZID", OrgDb =
## org.Hs.eg.db): 15.13% of input gene IDs are fail to map...

ego_up <- enrichGO(gene          = entrez_up$ENTREZID,
                   OrgDb         = org.Hs.eg.db,
                   ont           = "BP",          # Solo Biological Process
                   pAdjustMethod = "BH",
                   pvalueCutoff  = 0.05,
                   qvalueCutoff  = 0.2,
                   readable      = TRUE)
```

```
# Dotplot
dotplot(ego_up, showCategory = 15, title = "GO Biological Process - COVID19 upregulated genes")
```



SAVE THE OBJECT

```
saveRDS(Covid_75, "Covid_75.Rds")
```