MutationAnalysis

Theo-60985751, Hannah-88788427 , Jingxuan-85741635

2023-11-23

Data processing

Including the libraries

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```
library(pheatmap)
```

```
## Warning: package 'pheatmap' was built under R version 4.2.3
```

```
library("TCGAbiolinks")
library("survival")
```

```
## Warning: package 'survival' was built under R version 4.2.3
```

```
library("survminer")
```

```
## Warning: package 'survminer' was built under R version 4.2.3
```

```
## Loading required package: ggpubr
```

```
## Warning: package 'ggpubr' was built under R version 4.2.3
```

```
##
## Attaching package: 'survminer'
```

```
## The following object is masked from 'package:survival':
##
##     myeloma
```

```
library("SummarizedExperiment")
```

```
## Loading required package: MatrixGenerics
```

```
## Loading required package: matrixStats
```

```
## Warning: package 'matrixStats' was built under R version 4.2.3
```

```
##
## Attaching package: 'MatrixGenerics'
```

```
## The following objects are masked from 'package:matrixStats':
##
```

```
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: GenomicRanges

## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min

## Loading required package: S4Vectors

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

##
## Attaching package: 'IRanges'
```

```
## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomeInfoDb

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians

library(DESeq2)
library("gridExtra")

## Warning: package 'gridExtra' was built under R version 4.2.3

##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:Biobase':
##
##     combine

## The following object is masked from 'package:BiocGenerics':
##
##     combine

library("AnnotationDbi")
library("org.Hs.eg.db")

##

library(pathview)

##
#############################################################################
#
## Pathview is an open source software package distributed under GNU General
## Public License version 3 (GPLv3). Details of GPLv3 is available at
```

```
## http://www.gnu.org/licenses/gpl-3.0.html. Particullary, users are required
to
## formally cite the original Pathview paper (not just mention it) in
publications
## or products. For details, do citation("pathview") within R.
##
## The pathview downloads and uses KEGG data. Non-academic uses may require a
KEGG
## license agreement (details at http://www.kegg.jp/kegg/legal.html).
##
############################################################################
#

library(gage)

##
```

Opening the datafiles

```
data_mutation = read.delim("data_mutations.txt", header = TRUE, sep = '\t')
data_clinical = read.table("data_clinical_patient.txt",header = TRUE, sep =
'\t')
data_rna = read.csv("RNAseq_LIHC.csv", header = TRUE, row.names = "X")
```

Finding the patients that we have data for clinical, genome, and RNAseq data

```
unique_patient_clinical = unique(data_clinical$PATIENT_ID)
unique_patient_rna = unique(colnames(data_rna))
unique_patient_mutation = unique(data_mutation$Tumor_Sample_Barcode)

shortened_rna = substr(unique_patient_rna, start = 1, stop = 12)
shortened_rna = gsub("\\.", "-", shortened_rna)

shortened_mutation = substr(unique_patient_mutation, start = 1, stop = 12)

common_names1 <- intersect(unique_patient_clinical, shortened_mutation)
common_names2 <- intersect(common_names1, shortened_rna)
```

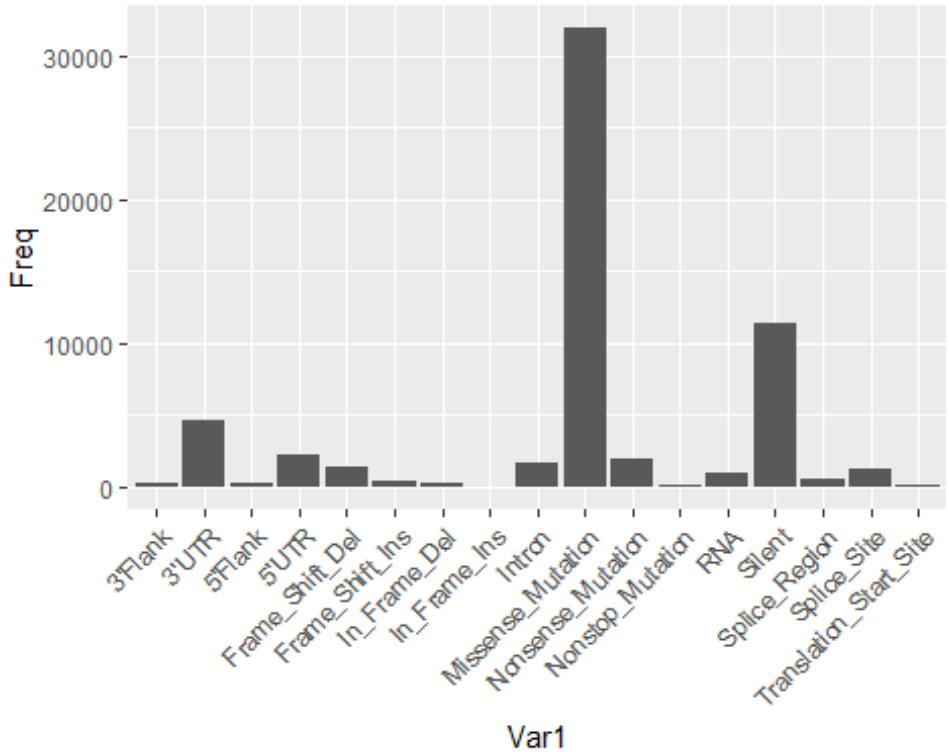
Adding the patient ID at the last column and get the common datas

```
data_clinical_common = subset(data_clinical, PATIENT_ID %in% common_names2)

data_mutation$PATIENT_ID = substr(data_mutation$Tumor_Sample_Barcode, start =
1, stop = 12)
data_mutation_common = subset(data_mutation, PATIENT_ID %in% common_names2)

data_mutation_common$Tumor_Sample_Barcode = data_mutation_common$PATIENT_ID

data_rna_shortened = data_rna
colnames(data_rna_shortened) = substr(colnames(data_rna_shortened), start =
1, stop = 12)
colnames(data_rna_shortened) = gsub("\\.", "-", colnames(data_rna_shortened))
data_rna_common = data_rna_shortened[,common_names2]
```

Analysis of mutation data

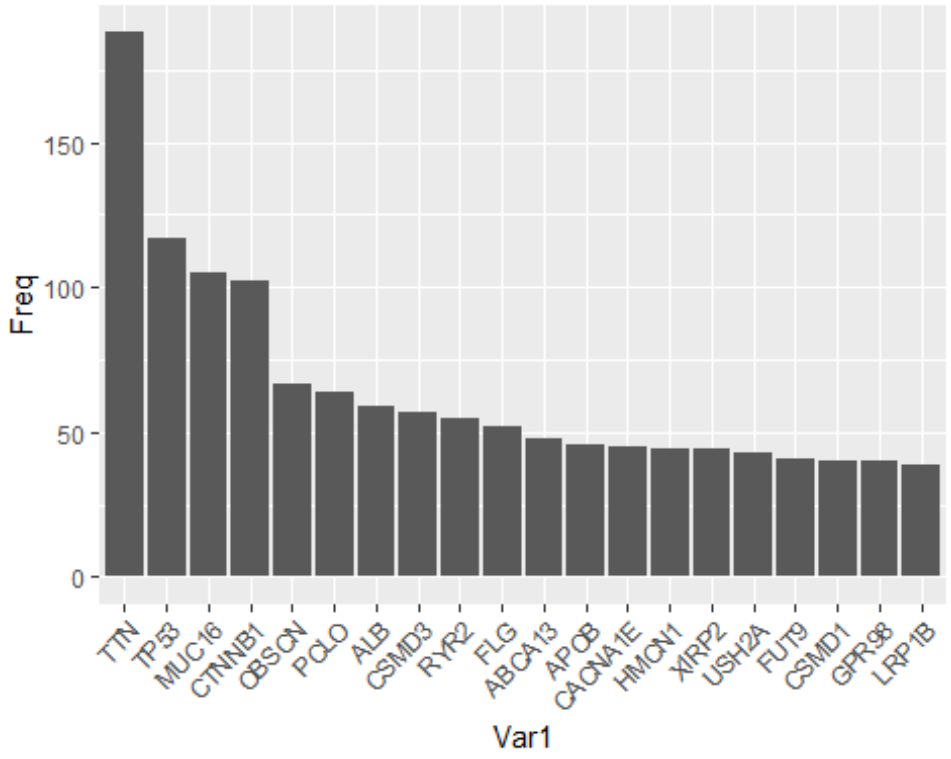
Generate a plot for distribution of variant classifications

```
data_oncoplot = data_mutation_common

hugo <- as.data.frame(table(data_oncoplot$Hugo_Symbol))
var.class <- as.data.frame(table(data_oncoplot$Variant_Classification))
ggplot(data=var.class, aes(x=Var1, y=Freq))+
  geom_col()+
  theme(axis.text.x = element_text(angle = 45,hjust=1))
```



Generate a plot for distribution of mutation events

```
hugo <- as.data.frame(table(data_mutation_common$Hugo_Symbol))

hugo.ordered <- hugo[order(-hugo$Freq),]

ggplot(data=hugo.ordered[1:20,], aes(x=Var1, y=Freq))+
  geom_col()+
  theme(axis.text.x = element_text(angle = 45,hjust=1))+
  scale_x_discrete(limits = hugo.ordered[1:20,]$Var1)
```



Generate an oncoplot matrix of all genes

```
cnv_events = unique(data_oncoplot$Variant_Classification)
oncomat = reshape2::dcast(
  data = data_oncoplot,
  formula = Hugo_Symbol ~ Tumor_Sample_Barcode,
  fun.aggregate = function(x, cnv = cnv_events) {
    x = as.character(x) # >= 2 same/distinct variant classification =
Multi_Hit
    xad = x[x %in% cnv]
    xvc = x[!x %in% cnv]

    if (length(xvc) > 0) {
      xvc = ifelse(test = length(xvc) > 1,
                   yes = 'Multi_Hit',
                   no = xvc)
    }

    x = ifelse(
      test = length(xad) > 0,
      yes = paste(xad, xvc, sep = ';'),
      no = xvc
    )
    x = gsub(pattern = ';$',
             replacement = '',
             x = x)
    x = gsub(pattern = '^;',
```

```
            replacement = '',
            x = x)
    return(x)
  },
  value.var = 'Variant_Classification',
  fill = '',
  drop = FALSE
)

rownames(oncomat) = oncomat$Hugo_Symbol
oncomat <- oncomat[,-1]

oncomat.ordered <- oncomat[order(-hugo$Freq),]
```

Transform the matrix into a binary matrix

```
mat <- oncomat.ordered
mat[mat== "Silent"] = 0
mat[mat == "Intron"] = 0
mat[mat == "Missense_Mutation"] = 0
mat[mat == ""] = 0

mat <- apply(mat, 2 ,as.numeric)
mat <- as.matrix(mat)
mat[is.na(mat)]=1

rownames(mat)  <-  row.names(oncomat.ordered)
```

Finding the top 20 most mutated genes

```
genes = rowSums(mat)
genes.ordered = sort(genes, decreasing = TRUE)



genes.ordered.top = genes.ordered[1:20]
genes.ordered.top.names = names(genes.ordered.top)

data_oncoplot.top = subset(data_oncoplot, Hugo_Symbol %in%
genes.ordered.top.names)
```

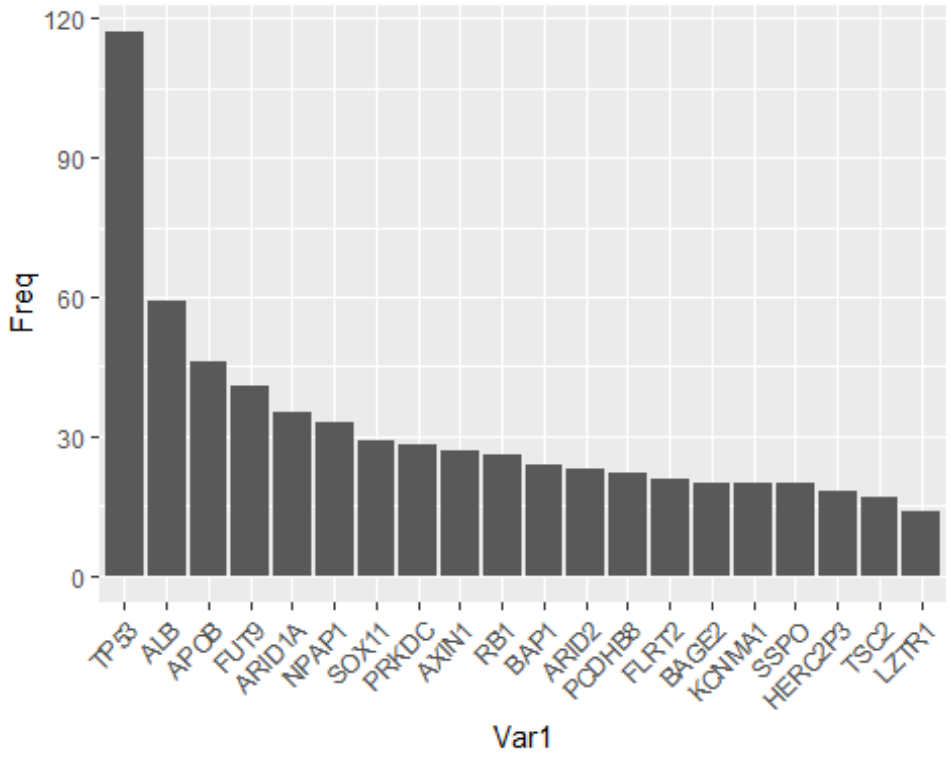
Making a matrix of the top 20 mutated genes

```
cnv_events = unique(data_oncoplot.top$Variant_Classification)
oncomat.top = reshape2::dcast(
  data = data_oncoplot.top,
  formula = Hugo_Symbol ~ Tumor_Sample_Barcode,
  fun.aggregate = function(x, cnv = cnv_events) {
    x = as.character(x) # >= 2 same/distinct variant classification =
Multi_Hit
    xad = x[x %in% cnv]
```

```
    xvc = x[!x %in% cnv]

    if (length(xvc) > 0) {
      xvc = ifelse(test = length(xvc) > 1,
                   yes = 'Multi_Hit',
                   no = xvc)
    }

    x = ifelse(
      test = length(xad) > 0,
      yes = paste(xad, xvc, sep = ';'),
      no = xvc
    )
    x = gsub(pattern = ';$',
             replacement = '',
             x = x)
    x = gsub(pattern = '^;',
             replacement = '',
             x = x)
    return(x)
  },
  value.var = 'Variant_Classification',
  fill = '',
  drop = FALSE
)
hugo <- as.data.frame(table(data_oncoplot.top$Hugo_Symbol))

rownames(oncomat.top) = oncomat.top$Hugo_Symbol
oncomat.top <- oncomat.top[,-1]
oncomat.top.ordered <- oncomat.top[order(-hugo$Freq),]
```

Generate a plot for distribution of the top 20 most mutated genes

```
hugo.ordered <- hugo[order(-hugo$Freq),]
ggplot(hugo.ordered, aes(x=Var1, y=Freq))+
  geom_col()+
  theme(axis.text.x = element_text(angle = 45,hjust=1))+
  scale_x_discrete(limits = hugo.ordered$Var1)
```

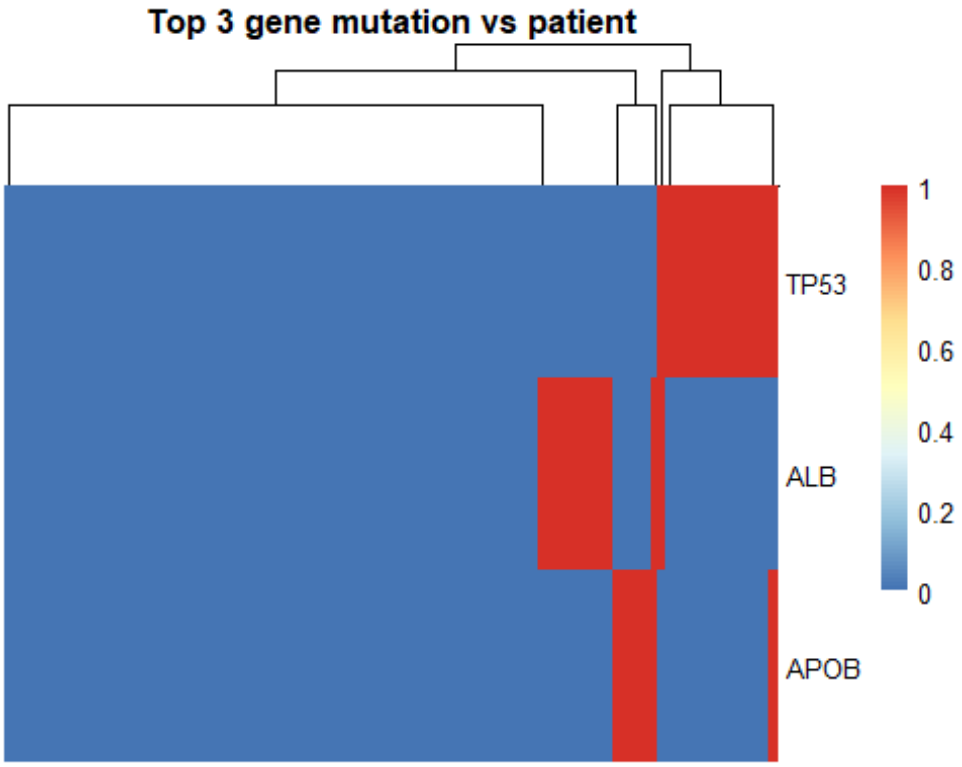


Transforming the top 20 gene matrix into binary

```
mat.top <- oncomat.top.ordered
mat.top[mat.top== "Silent"] = 0
mat.top[mat.top == "Intron"] = 0
mat.top[mat.top == "Missense_Mutation"] = 0

mat.top[mat.top == ""] = 0



mat.top <- apply(mat.top, 2 ,as.numeric)
mat.top <- as.matrix(mat.top)
mat.top[is.na(mat.top)]=1

rownames(mat.top)  <-  row.names(oncomat.top.ordered)
```

Generate a pheatmap for top 3 mutated genes

```
reduce.mat <- mat.top[1:3,]
res <- pheatmap(reduce.mat,
         cluster_rows = FALSE,
         show_colnames = FALSE, main = "Top 3 gene mutation vs patient")
```



Performing k-mean clustering

```
cluster = as.data.frame(cutree(res$tree_col, k = 2))
# cluster
```

Finding the patientID in each group

```
mutation_group1_patientID = subset(cluster,`cutree(res$tree_col, k = 2)` == 1)
mutation_group2_patientID = subset(cluster,`cutree(res$tree_col, k = 2)` == 2)

data_clinical_common_group1 = subset(data_clinical_common,PATIENT_ID %in% rownames(mutation_group1_patientID))

data_clinical_common_group2 = subset(data_clinical_common,PATIENT_ID %in% rownames(mutation_group2_patientID))

mutation_TP53_patientID = data_mutation_common$PATIENT_ID[data_mutation_common$Hugo_Symbol == "TP53"]

data_clinical_common_TP53 = subset(data_clinical_common, PATIENT_ID %in% mutation_TP53_patientID)
```

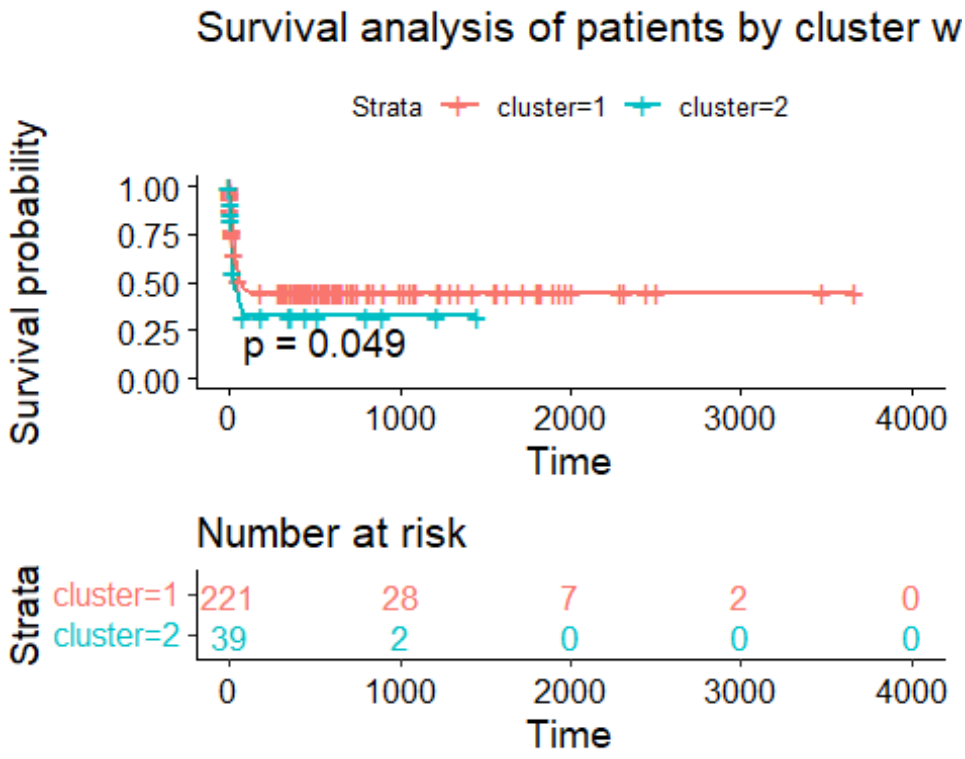
Survival analysis

survival analysis (SA) on clinical data

```
data_clinical_common$deceased = data_clinical_common$PFS_STATUS ==
"1:PROGRESSION"

# create an "overall survival" variable that is equal to days_to_death
# for dead patients, and to days_to_last_follow_up for patients who
# are still alive
data_clinical_common$overall_survival = ifelse(data_clinical_common$deceased,
                                    data_clinical_common$OS_MONTHS,
                                    data_clinical_common$DAYS_LAST_FOLLOWUP)
```

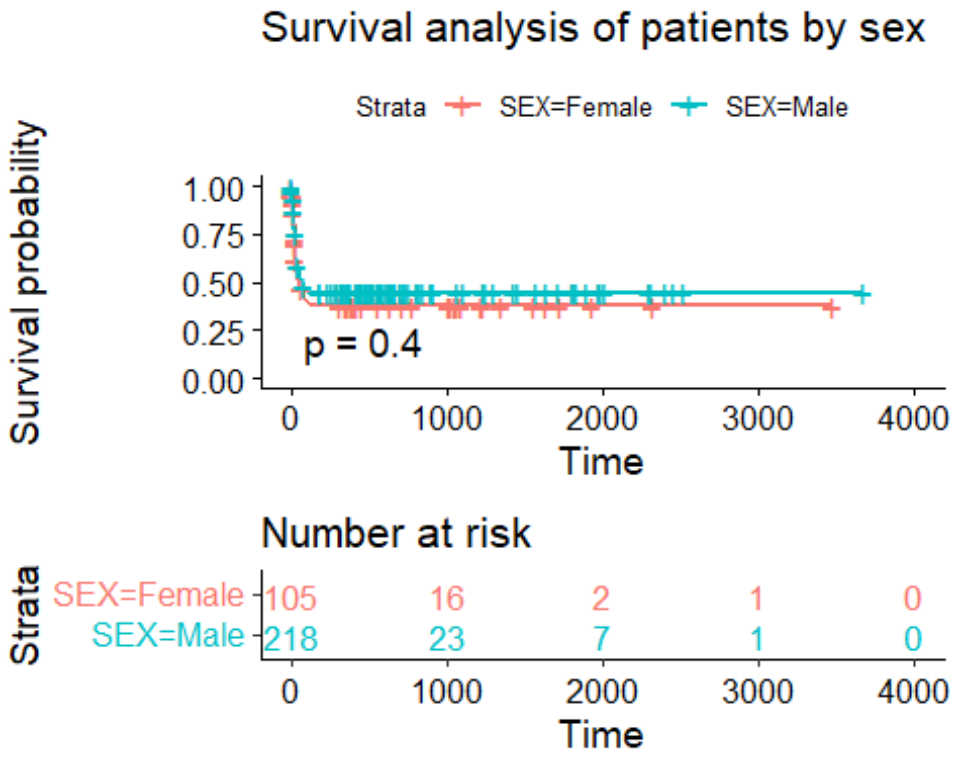
SA by cluster

```
# Adding a cluster annotation column

data_clinical_common$cluster = cluster[data_clinical_common$PATIENT_ID,]

Surv(data_clinical_common$overall_survival, data_clinical_common$deceased) ~
data_clinical_common$cluster

## Surv(data_clinical_common$overall_survival, data_clinical_common$deceased)
~
##     data_clinical_common$cluster

table(data_clinical_common$cluster)

##
##   1   2
## 240  45

fit = survfit(Surv(overall_survival, deceased) ~ cluster, data =
data_clinical_common)

pval = surv_pvalue(fit, data=data_clinical_common)$pval
print(pval)

## [1] 0.04853337

ggsurvplot(fit, data=data_clinical_common, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35, title="Survival analysis of
patients by cluster with mutation data")
```



SA by sex

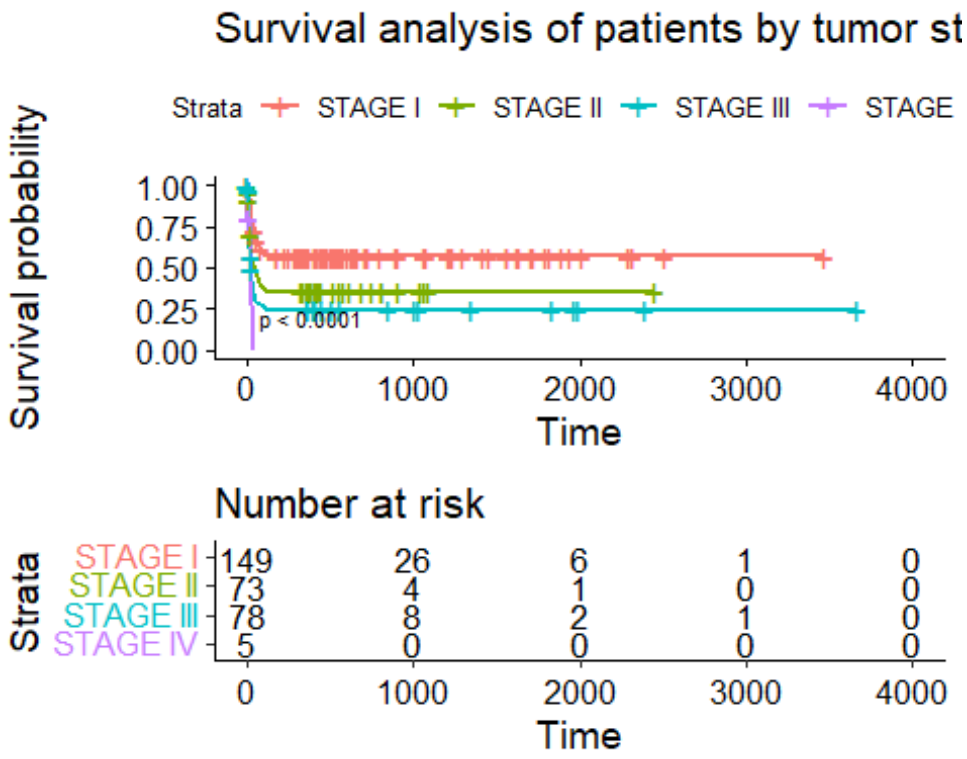
```
Surv(data_clinical_common$overall_survival, data_clinical_common$deceased) ~
data_clinical_common$SEX

## Surv(data_clinical_common$overall_survival, data_clinical_common$deceased)
~
##     data_clinical_common$SEX

table(data_clinical_common$SEX)

##
## Female   Male
##    117    236

fit = survfit(Surv(overall_survival, deceased) ~ SEX, data =
data_clinical_common)

pval = surv_pvalue(fit, data=data_clinical_common)$pval
print(pval)

## [1] 0.4000524

ggsurvplot(fit, data=data_clinical_common, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35, title = "Survival analysis
of patients by sex")
```



SA by Tumor Stage

```
# remove any of the letters "a", "b" or "c", but only if they are at the end
# of the name, eg "stage iiia" would become simply "stage iii"
data_clinical_common$AJCC_PATHOLOGIC_TUMOR_STAGE = gsub("[ABC]$", "",
data_clinical_common$AJCC_PATHOLOGIC_TUMOR_STAGE)


data_clinical_common[which(data_clinical_common$AJCC_PATHOLOGIC_TUMOR_STAGE
== ""), "AJCC_PATHOLOGIC_TUMOR_STAGE"] = NA

table(data_clinical_common$AJCC_PATHOLOGIC_TUMOR_STAGE)

##
##   STAGE I  STAGE II STAGE III  STAGE IV
##       166        81        82         6

fit = survfit(Surv(overall_survival, deceased) ~ AJCC_PATHOLOGIC_TUMOR_STAGE,
data=data_clinical_common)

# we can extract the survival p-value and print it
pval = surv_pvalue(fit, data=data_clinical_common)$pval
print(pval)

## [1] 9.488618e-09

ggsurvplot(fit, data=data_clinical_common, pval=T,pval.size = 3
,risk.table=T, risk.table.height=0.39,legend.lab = c("STAGE I", "STAGE II",
```

```
"STAGE III", "STAGE IV"), title = "Survival analysis of patients by tumor
stage", title.fontsize = 2)
```



Survival analysis on patients in each cluster (produced by mutation data) ### SA on cluster 1 by Tumor Stage

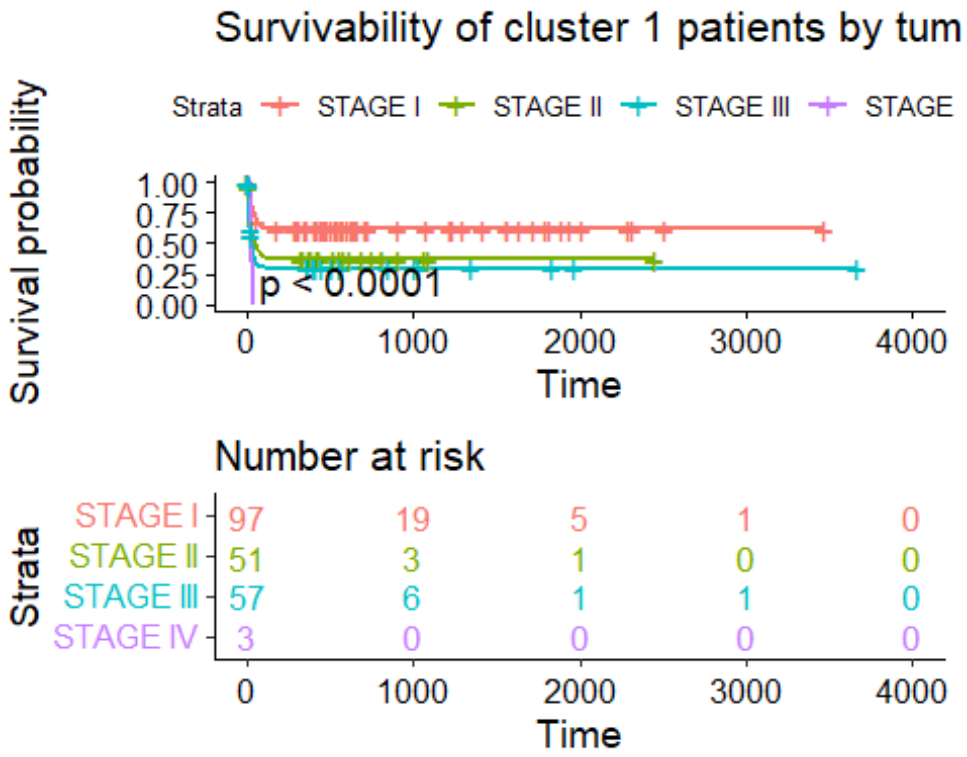
```
data_clinical_common_group1$deceased = data_clinical_common_group1$PFS_STATUS
== "1:PROGRESSION"

# create an "overall survival" variable that is equal to days_to_death
# for dead patients, and to days_to_last_follow_up for patients who
# are still alive
data_clinical_common_group1$overall_survival =
ifelse(data_clinical_common_group1$deceased,
                                    data_clinical_common_group1$OS_MONTHS,

data_clinical_common_group1$DAYS_LAST_FOLLOWUP)

# remove any of the letters "a", "b" or "c", but only if they are at the end
# of the name, eg "stage iiia" would become simply "stage iii"
data_clinical_common_group1$AJCC_PATHOLOGIC_TUMOR_STAGE = gsub("[ABC]$", "",
data_clinical_common_group1$AJCC_PATHOLOGIC_TUMOR_STAGE)

data_clinical_common_group1[which(data_clinical_common_group1$AJCC_PATHOLOGIC
_TUMOR_STAGE == ""), "AJCC_PATHOLOGIC_TUMOR_STAGE"] = NA

table(data_clinical_common_group1$AJCC_PATHOLOGIC_TUMOR_STAGE)
```

```
##
##   STAGE I  STAGE II STAGE III  STAGE IV
##       108        58        58         3
```

```
fit = survfit(Surv(overall_survival, deceased) ~ AJCC_PATHOLOGIC_TUMOR_STAGE
, data=data_clinical_common_group1)

# we can extract the survival p-value and print it
pval = surv_pvalue(fit, data=data_clinical_common_group1)$pval
print(pval)
```

```
## [1] 7.978726e-06
```

```
ggsurvplot(fit, data=data_clinical_common_group1, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.45,legend.lab = c("STAGE I",
"STAGE II", "STAGE III", "STAGE IV"), title = "Survivability of cluster 1
patients by tumor stage")
```



SA on cluster 1 by Sex

```
Surv(data_clinical_common_group1$overall_survival,
data_clinical_common_group1$deceased) ~ data_clinical_common_group1$SEX
```

```
## Surv(data_clinical_common_group1$overall_survival,
data_clinical_common_group1$deceased) ~
##     data_clinical_common_group1$SEX
```

```
fit = survfit(Surv(overall_survival, deceased) ~ SEX, data =
data_clinical_common_group1)
```

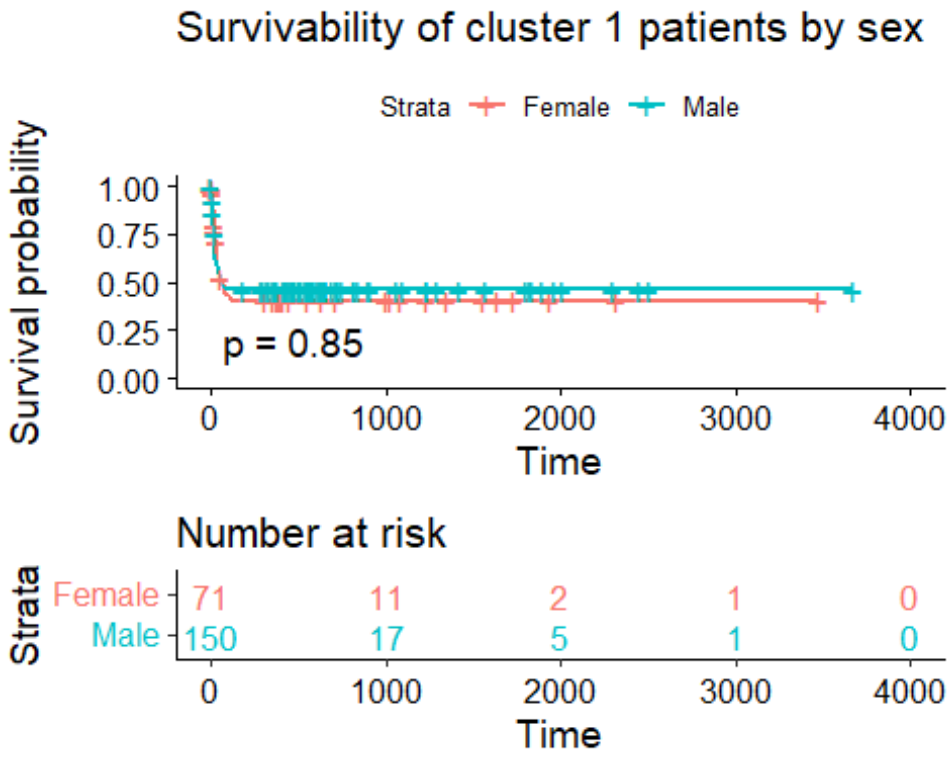
```r
table(data_clinical_common_group1$SEX)
```

```
##
## Female   Male
##     80    160
```

```r
pval = surv_pvalue(fit, data=data_clinical_common_group1)$pval
print(pval)
```

```
## [1] 0.8544197
```

```r
ggsurvplot(fit, data=data_clinical_common_group1, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35, legend.lab = c("Female",
"Male"), title = "Survivability of cluster 1 patients by sex")
```



SA on cluster 2 Tumor stage

```r
data_clinical_common_group2$deceased = data_clinical_common_group2$PFS_STATUS
== "1:PROGRESSION"

# create an "overall survival" variable that is equal to days_to_death
# for dead patients, and to days_to_last_follow_up for patients who
# are still alive
data_clinical_common_group2$overall_survival =
ifelse(data_clinical_common_group2$deceased,
                                    data_clinical_common$OS_MONTHS,
                                    data_clinical_common$DAYS_LAST_FOLLOWUP)
```

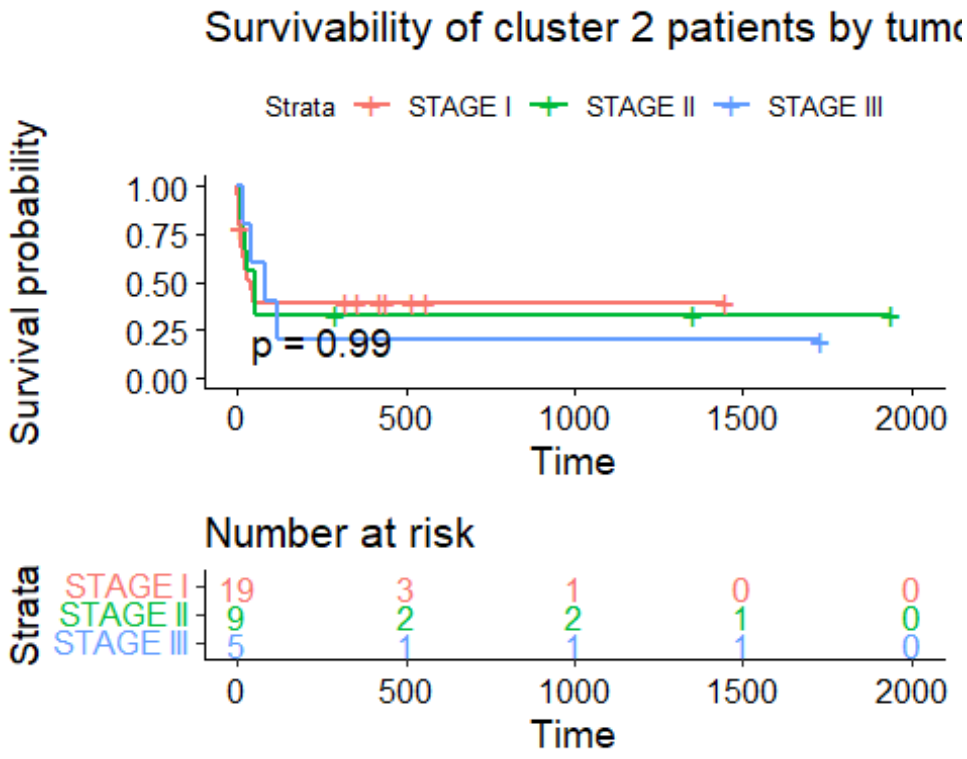
```r
# remove any of the letters "a", "b" or "c", but only if they are at the end
# of the name, eg "stage iiia" would become simply "stage iii"
data_clinical_common_group2$AJCC_PATHOLOGIC_TUMOR_STAGE = gsub("[ABC]$", "",
data_clinical_common_group2$AJCC_PATHOLOGIC_TUMOR_STAGE)

data_clinical_common_group2[which(data_clinical_common_group2$AJCC_PATHOLOGIC
_TUMOR_STAGE == ""), "AJCC_PATHOLOGIC_TUMOR_STAGE"] = NA

table(data_clinical_common_group2$AJCC_PATHOLOGIC_TUMOR_STAGE)
```

```
##
##   STAGE I  STAGE II STAGE III
##        25        12         7
```

```r
fit = survfit(Surv(overall_survival, deceased) ~ AJCC_PATHOLOGIC_TUMOR_STAGE,
data=data_clinical_common_group2)

# we can extract the survival p-value and print it
pval = surv_pvalue(fit, data=data_clinical_common_group2)$pval
print(pval)
```

```
## [1] 0.9941215
```

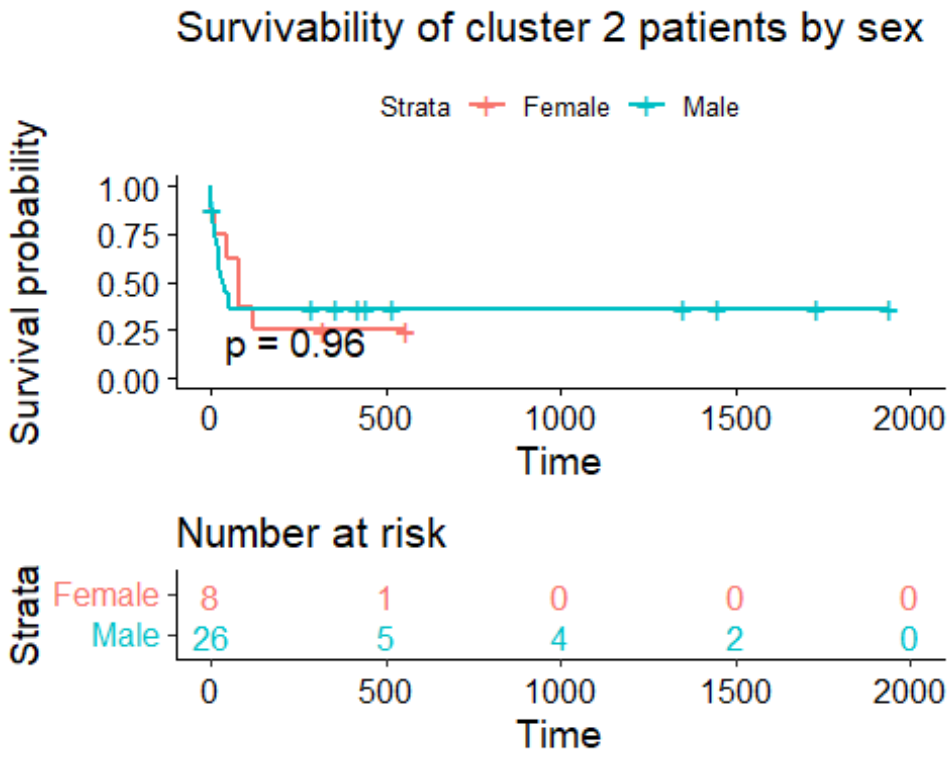
```r
ggsurvplot(fit, data=data_clinical_common_group2, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35,legend.lab = c("STAGE I",
"STAGE II", "STAGE III"), title = "Survivability of cluster 2 patients by
tumor stage")
```



SA on cluster 2

Sex

```
Surv(data_clinical_common_group2$overall_survival,
data_clinical_common_group2$deceased) ~ data_clinical_common_group2$SEX

## Surv(data_clinical_common_group2$overall_survival,
data_clinical_common_group2$deceased) ~
##     data_clinical_common_group2$SEX

fit = survfit(Surv(overall_survival, deceased) ~ SEX, data =
data_clinical_common_group2)

table(data_clinical_common_group2$SEX)

##
## Female   Male
##     10     35

pval = surv_pvalue(fit, data=data_clinical_common_group2)$pval
print(pval)

## [1] 0.9592107

ggsurvplot(fit, data=data_clinical_common_group2, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35, legend.lab = c("Female",
"Male"), title = "Survivability of cluster 2 patients by sex")
```



SA on the most mutated gene (TP53) ### SA by sex

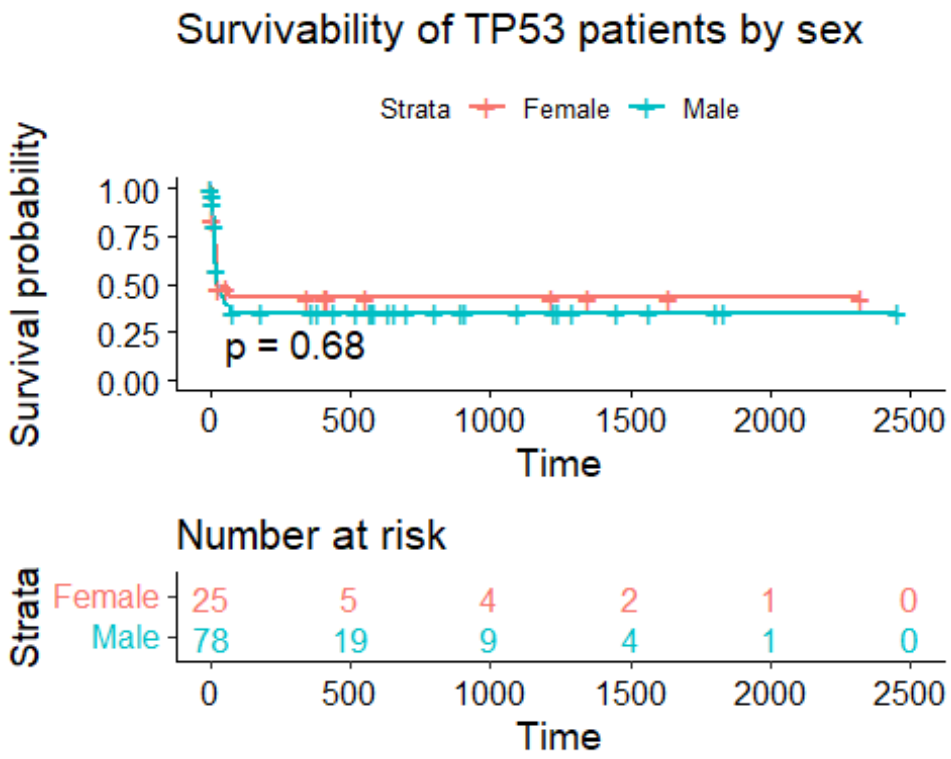
```
data_clinical_common_TP53$deceased = data_clinical_common_TP53$PFS_STATUS ==
"1:PROGRESSION"

# create an "overall survival" variable that is equal to days_to_death
# for dead patients, and to days_to_last_follow_up for patients who
# are still alive
data_clinical_common_TP53$overall_survival =
ifelse(data_clinical_common_TP53$deceased,
                                    data_clinical_common_TP53$OS_MONTHS,

data_clinical_common_TP53$DAYS_LAST_FOLLOWUP)


Surv(data_clinical_common_TP53$overall_survival,
data_clinical_common_TP53$deceased) ~ data_clinical_common_TP53$SEX

## Surv(data_clinical_common_TP53$overall_survival,
data_clinical_common_TP53$deceased) ~
##     data_clinical_common_TP53$SEX

fit = survfit(Surv(overall_survival, deceased) ~ SEX, data =
data_clinical_common_TP53)

table(data_clinical_common_TP53$SEX)
```

```
##
## Female   Male
##     26     84
```

```
pval = surv_pvalue(fit, data=data_clinical_common_TP53)$pval
print(pval)
```

```
## [1] 0.6845213
```

```
ggsurvplot(fit, data=data_clinical_common_TP53, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35,legend.lab = c("Female",
"Male"), title = "Survivability of TP53 patients by sex" )
```



SA by tumor stage

```
# remove any of the letters "a", "b" or "c", but only if they are at the end
# of the name, eg "stage iiia" would become simply "stage iii"
data_clinical_common_TP53$AJCC_PATHOLOGIC_TUMOR_STAGE = gsub("[ABC]$", "",
data_clinical_common_TP53$AJCC_PATHOLOGIC_TUMOR_STAGE)
```

```
data_clinical_common_TP53[which(data_clinical_common_TP53$AJCC_PATHOLOGIC_TUM
OR_STAGE == ""), "AJCC_PATHOLOGIC_TUMOR_STAGE"] = NA
```

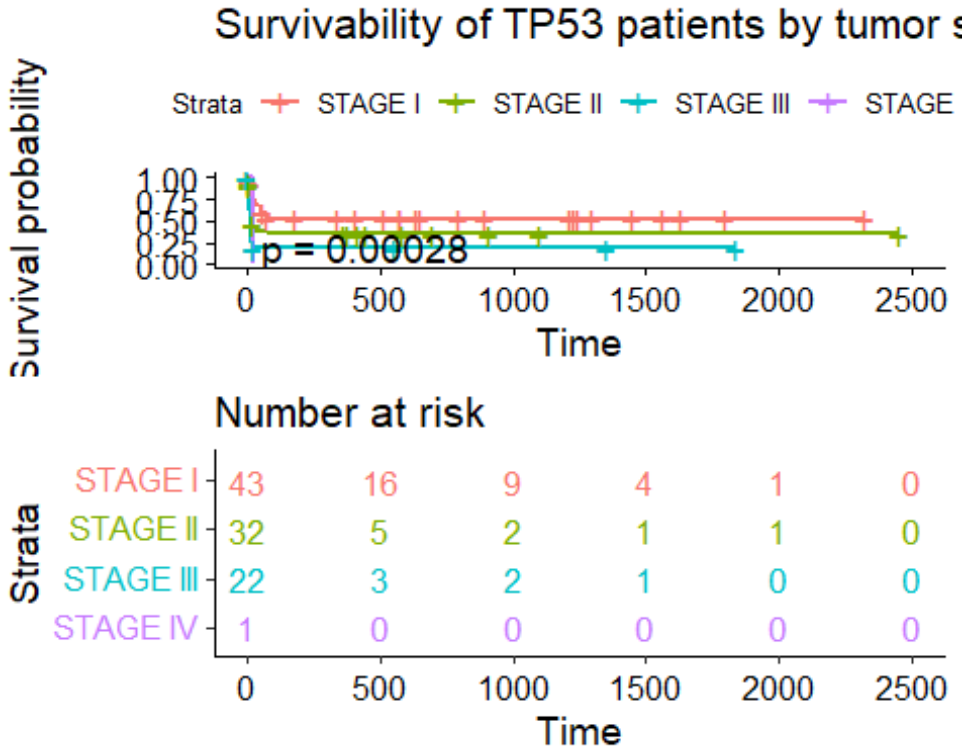
```
table(data_clinical_common_TP53$AJCC_PATHOLOGIC_TUMOR_STAGE)
```

```
##
##   STAGE I  STAGE II STAGE III  STAGE IV
##        47        33        24         1
```

```
fit = survfit(Surv(overall_survival, deceased) ~ AJCC_PATHOLOGIC_TUMOR_STAGE,
data=data_clinical_common_TP53)

# we can extract the survival p-value and print it
pval = surv_pvalue(fit, data=data_clinical_common_TP53)$pval
print(pval)
```

```
## [1] 0.0002790916
```

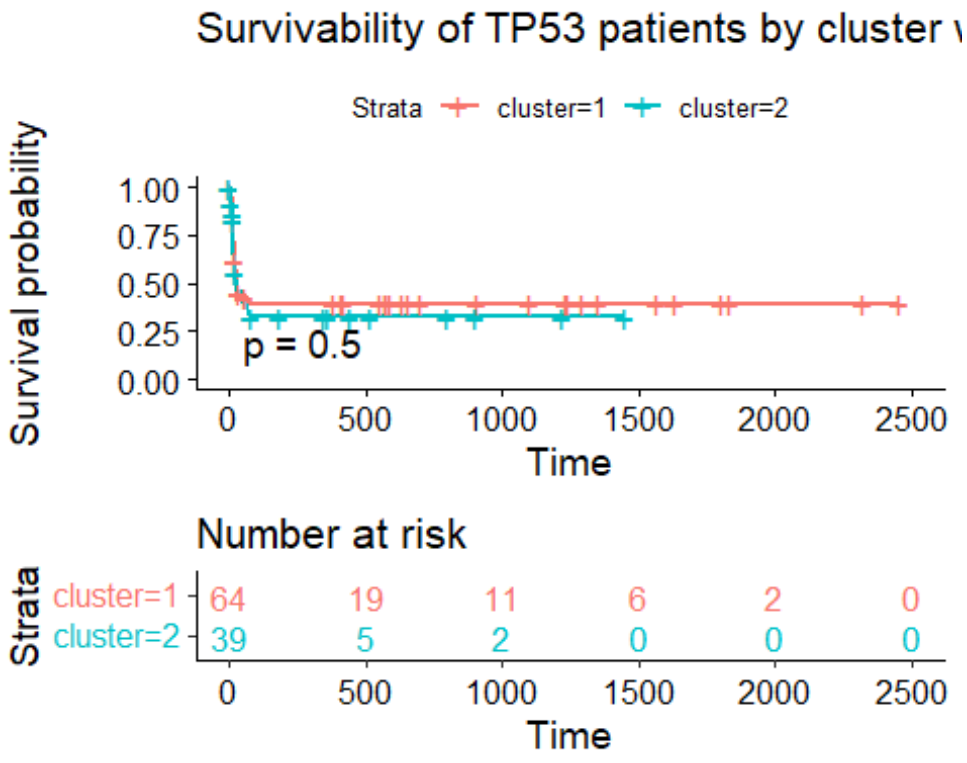
```
ggsurvplot(fit, data=data_clinical_common_TP53, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.50, legend.labs = c("STAGE I",
"STAGE II", "STAGE III", "STAGE IV"), title = "Survivability of TP53 patients
by tumor stage")
```



SA by cluster with mutation data

```
data_clinical_common_TP53$cluster =
cluster[data_clinical_common_TP53$PATIENT_ID,]

Surv(data_clinical_common_TP53$overall_survival,
data_clinical_common_TP53$deceased) ~ data_clinical_common_TP53$cluster
```

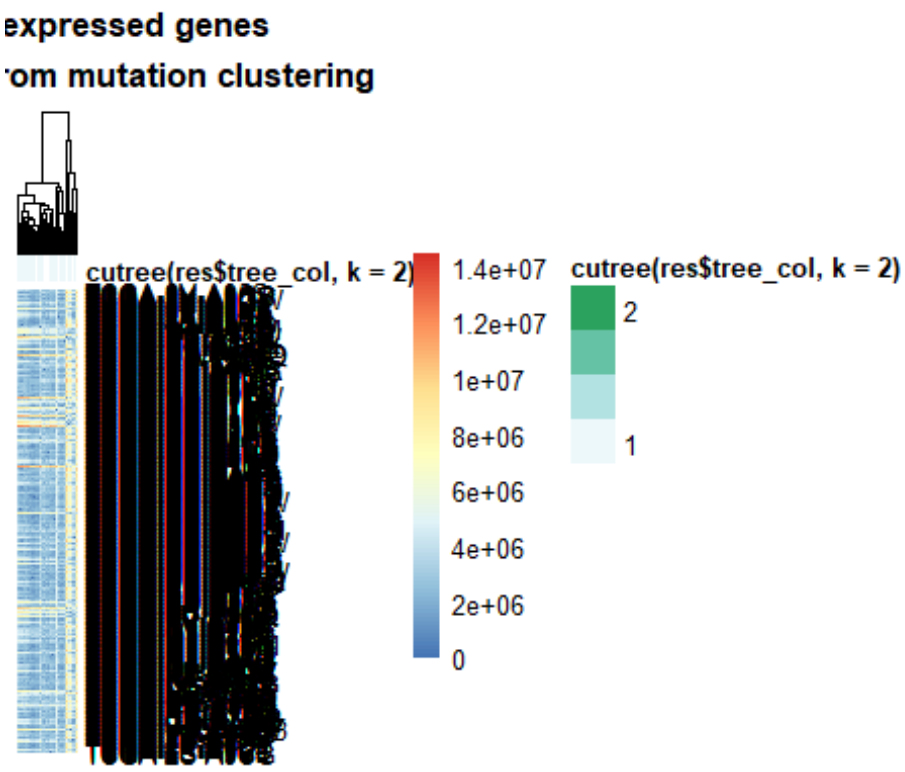
```
## Surv(data_clinical_common_TP53$overall_survival,
data_clinical_common_TP53$deceased) ~
##     data_clinical_common_TP53$cluster

fit = survfit(Surv(overall_survival, deceased) ~ cluster, data =
data_clinical_common_TP53)

table(data_clinical_common_TP53$cluster)

##
##  1  2
## 65 45

pval = surv_pvalue(fit, data=data_clinical_common_TP53)$pval
print(pval)

## [1] 0.5026988

ggsurvplot(fit, data=data_clinical_common_TP53, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35, title = "Survivability of
TP53 patients by cluster with mutation data")
```



Differential Expression analysis

```
data_rna_common <- data_rna_common[rowSums(data_rna_common)>1,]
# head(data_rna_common)

sampleDists = dist(t(data_rna_common),upper = TRUE)
# sampleDists
```

```
# annot_col = data.frame(colData$condition)
# row.names(annot_col) <- rownames(colData)

sampleDistMatrix = as.matrix( sampleDists )
rownames(sampleDistMatrix) = colnames(data_rna_common)
colnames(sampleDistMatrix) = colnames(data_rna_common)

res_rna = pheatmap(sampleDistMatrix,
         clustering_distance_rows = sampleDists,
         clustering_distance_cols = sampleDists,
         cluster_rows=FALSE, show_rownames=TRUE,
         cluster_cols=TRUE, shor_colnames =  FALSE, show_colnames = FALSE,
         annotation_col=cluster, main = "Distance of expressed genes \n with
annotation from mutation clustering",)
```

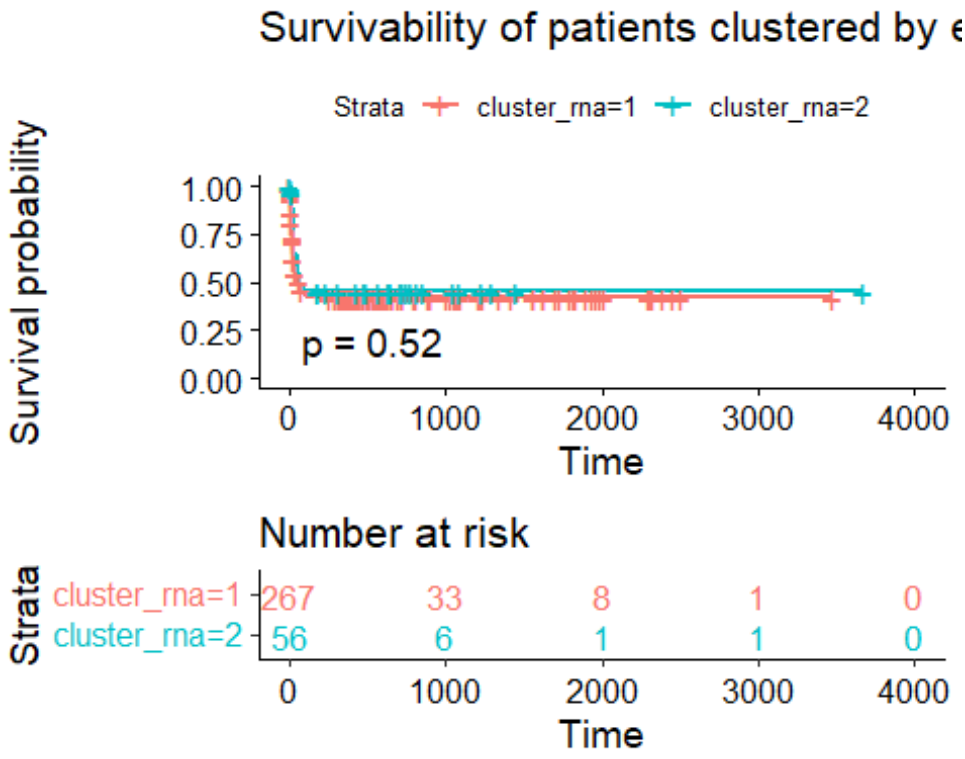



```
cluster_rna = as.data.frame(cutree(res_rna$tree_col, k = 2))
# cluster_rna
```

Adding cluster_rna cluster and survival analysis

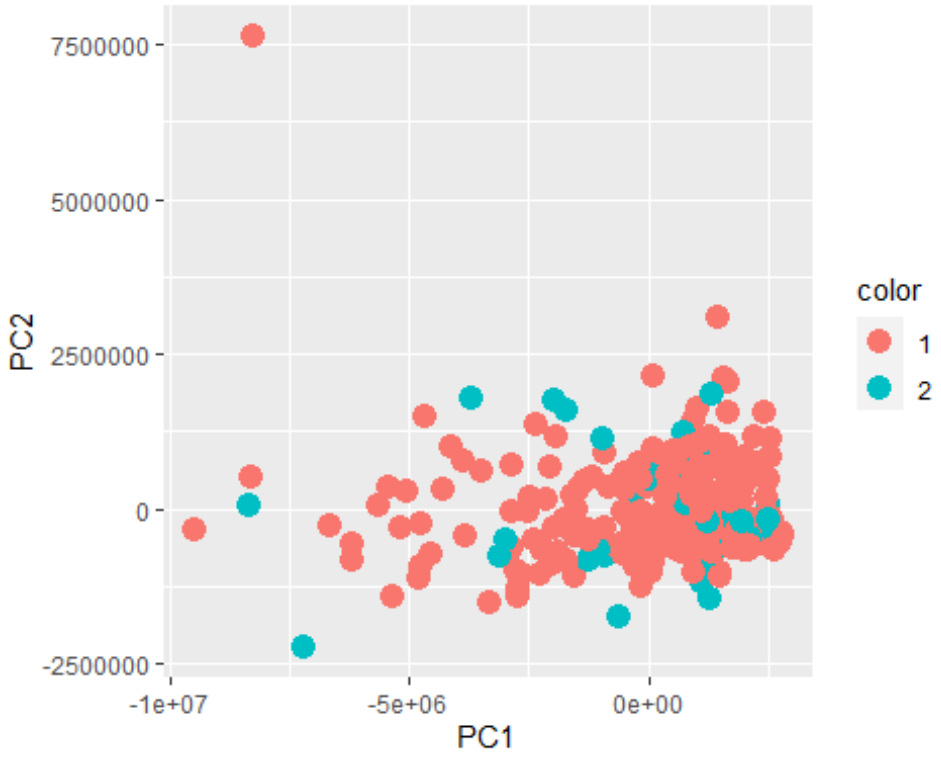
```
data_clinical_common$deceased = data_clinical_common$PFS_STATUS ==
"1:PROGRESSION"

# create an "overall survival" variable that is equal to days_to_death
# for dead patients, and to days_to_last_follow_up for patients who
# are still alive
data_clinical_common$overall_survival = ifelse(data_clinical_common$deceased,
```

```
                                    data_clinical_common$OS_MONTHS,
                                    data_clinical_common$DAYS_LAST_FOLLOWUP)

data_clinical_common$cluster_rna =
cluster_rna[data_clinical_common$PATIENT_ID,]

Surv(data_clinical_common$overall_survival, data_clinical_common$deceased) ~
data_clinical_common$cluster_rna

## Surv(data_clinical_common$overall_survival, data_clinical_common$deceased)
~
##     data_clinical_common$cluster_rna

table(data_clinical_common$cluster_rna)

##
##   1   2
## 291  62

fit = survfit(Surv(overall_survival, deceased) ~ cluster_rna, data =
data_clinical_common)

pval = surv_pvalue(fit, data=data_clinical_common)$pval
print(pval)

## [1] 0.518244

ggsurvplot(fit, data=data_clinical_common, pval=T, risk.table=T,
risk.table.col="strata", risk.table.height=0.35, title = "Survivability of
patients clustered by expression data")
```



PCA plot

```
rownames(cluster) = substr(rownames(cluster), start = 1, stop = 12)
common_cluster <- row.names(cluster)

data_rna_common_cluster <- data_rna_common[,common_cluster]

colnames(data_rna_common_cluster) = rownames(cluster)

pca_res <- prcomp(t(data_rna_common_cluster), scale. = FALSE)
score <- pca_res$x

score = as.data.frame(score)
score$color <- as.factor(cluster$`cutree(res$tree_col, k = 2)`)

ggplot(score, aes(x=PC1, y=PC2,  color=color)) +
  geom_point(size = 4)
```



Actually running the DESeq pipeline

```
rownames(cluster) = substr(rownames(cluster), start = 1, stop = 12)
common_cluster <- row.names(cluster)

data_rna_common_cluster <- data_rna_common[,common_cluster]

colnames(data_rna_common_cluster) = rownames(cluster)

metadata <- data.frame(
  patientID = row.names(cluster),
  condition = cluster$`cutree(res$tree_col, k = 2)`
)

### making metadata a factor so DESeq2 runs faster
row.names(metadata) = metadata$patientID
metadata$patientID = NULL
metadata$condition = factor(metadata$condition)

dds <- DESeqDataSetFromMatrix(countData = data_rna_common_cluster, colData =
metadata, design = ~condition)

dds = DESeq(dds)

## estimating size factors

## estimating dispersions
```

```
## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 5788 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

dds

## class: DESeqDataSet
## dim: 54571 285
## metadata(1): version
## assays(6): counts mu ... replaceCounts replaceCooks
## rownames(54571): ENSG00000000003.15 ENSG00000000005.6 ...
##   ENSG00000288674.1 ENSG00000288675.1
## rowData names(23): baseMean baseVar ... maxCooks replace
## colnames(285): TCGA-2V-A95S TCGA-2Y-A9GS ... TCGA-ZS-A9CF TCGA-ZS-A9CG
## colData names(3): condition sizeFactor replaceable

print("Result table of DESeq")

## [1] "Result table of DESeq"

res_de = results(dds, contrast=c(1, 2))
mcols(res_de, use.names = TRUE)

## DataFrame with 6 rows and 2 columns
##                         type            description
##                  <character>            <character>
## baseMean       intermediate mean of normalized c..
## log2FoldChange      results log2 fold change (ML..
## lfcSE               results  standard error: +1,+2
## stat                results  Wald statistic: +1,+2
## pvalue              results Wald test p-value: +..
## padj                results   BH adjusted p-values

summary(res_de)

##
## out of 54493 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 25010, 46%
## LFC < 0 (down)     : 1934, 3.5%
## outliers [1]       : 0, 0%
```

```
## low counts [2]     : 11625, 21%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
print("\n Number of genes with adjusted p-value that is less than 0.05")
```

```
## [1] "\n Number of genes with adjusted p-value that is less than 0.05"
```

```
res_de.05 <- results(dds, alpha = 0.05)
table(res_de.05$padj < 0.05)
```

```
##
## FALSE  TRUE
## 29379  1873
```

```
print("\n number of genes with log2 fold more than doubling (p-value < 0.1)")
```

```
## [1] "\n number of genes with log2 fold more than doubling (p-value < 0.1)"
```

```
resLFC1 <- results(dds, lfcThreshold=1)
table(resLFC1$padj < 0.1)
```

```
##
## FALSE  TRUE
## 27003    23
```

P-values

```
res_de <- res_de[order(res_de$pvalue),]
summary(res_de)
```

```
##
## out of 54493 with nonzero total read count
## adjusted p-value < 0.1
## LFC > 0 (up)       : 25010, 46%
## LFC < 0 (down)     : 1934, 3.5%
## outliers [1]       : 0, 0%
## low counts [2]     : 11625, 21%
## (mean count < 0)
## [1] see 'cooksCutoff' argument of ?results
## [2] see 'independentFiltering' argument of ?results
```

```
## number of adjusted p-values less than 0.1
sum(res_de$padj < 0.1, na.rm=TRUE)
```

```
## [1] 26944
```

multiple testing

```
# sum(res_de$pvalue < 0.05, na.rm=TRUE)
# sum(!is.na(res_de$pvalue))
# sum(res_de$padj < 0.05, na.rm=TRUE)
```

```r
resSig <- subset(res_de, padj < 0.06)
resSig <- subset(res_de, padj < 0.06)
head(resSig[order( resSig$log2FoldChange ), ])
```

```
## log2 fold change (MLE): +1,+2
## Wald test p-value: +1,+2
## DataFrame with 6 rows and 6 columns
##                     baseMean log2FoldChange     lfcSE      stat
pvalue
##                    <numeric>      <numeric> <numeric> <numeric>
<numeric>
## ENSG00000250385.2    1.81257       -5.85375   2.05799  -2.84441 4.44944e-
03
## ENSG00000285933.1    5.59978       -5.55253   1.46623  -3.78694 1.52516e-
04
## ENSG00000142515.15  14.61962       -5.46463   2.17753  -2.50955 1.20884e-
02
## ENSG00000249641.2    1.12196       -5.15249   2.15537  -2.39054 1.68238e-
02
## ENSG00000284779.2    1.47193       -4.86638   1.62221  -2.99985 2.70116e-
03
## ENSG00000251630.1    1.77364       -4.76246   1.17470  -4.05420 5.03066e-
05
##                          padj
##                     <numeric>
## ENSG00000250385.2  8.07708e-03
## ENSG00000285933.1  2.94388e-04
## ENSG00000142515.15 2.12836e-02
## ENSG00000249641.2  2.92012e-02
## ENSG00000284779.2  4.96885e-03
## ENSG00000251630.1  9.87475e-05
```

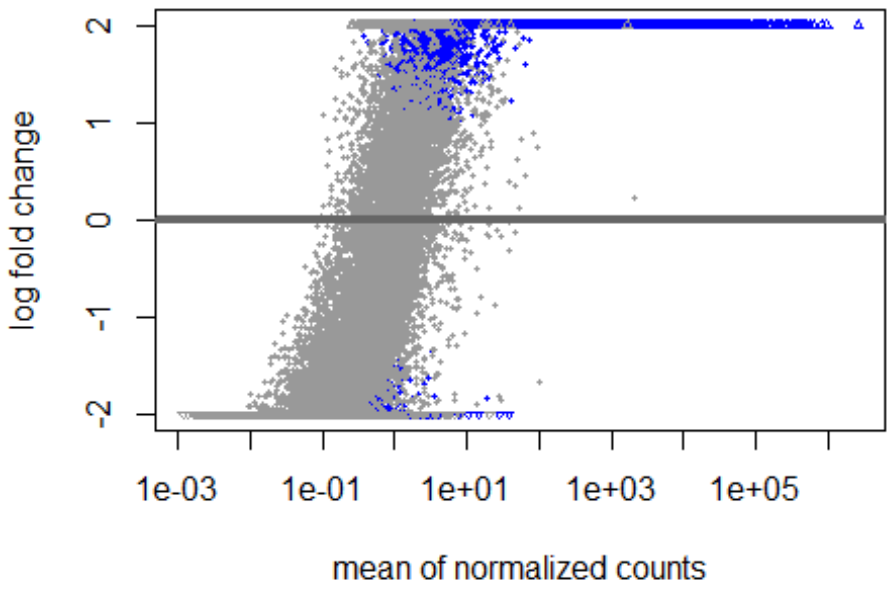
```r
head(resSig[order(resSig$log2FoldChange, decreasing=TRUE),], n= 20)
```

```
## log2 fold change (MLE): +1,+2
## Wald test p-value: +1,+2
## DataFrame with 20 rows and 6 columns
##                     baseMean log2FoldChange     lfcSE      stat
pvalue
##                    <numeric>      <numeric> <numeric> <numeric>
<numeric>
## ENSG00000163631.17   2610593        21.1939  0.446255   47.4927
0
## ENSG00000087086.15    545819        20.2335  0.327027   61.8712
0
## ENSG00000197249.14    963226        19.6484  0.347743   56.5027
0
## ENSG00000198804.2     809312        19.4548  0.279884   69.5104
0
```

```
## ENSG00000158874.11    558465        19.4243  0.458847   42.3328
0
## ...                      ...           ...       ...       ...
...
## ENSG00000198888.2     232394        17.9615  0.283928   63.2608
0.00000e+00
## ENSG00000171560.16    506679        17.8472  0.427564   41.7415
0.00000e+00
## ENSG00000257017.9     406341        17.8178  0.565548   31.5054 7.33251e-
218
## ENSG00000106927.12    284987        17.7434  0.375081   47.3056
0.00000e+00
## ENSG00000198786.2     151522        17.7019  0.360167   49.1491
0.00000e+00
##                           padj
##                      <numeric>
## ENSG00000163631.17           0
## ENSG00000087086.15           0
## ENSG00000197249.14           0
## ENSG00000198804.2            0
## ENSG00000158874.11           0
## ...                        ...
## ENSG00000198888.2   0.00000e+00
## ENSG00000171560.16  0.00000e+00
## ENSG00000257017.9  3.64601e-217
## ENSG00000106927.12  0.00000e+00
## ENSG00000198786.2   0.00000e+00
```

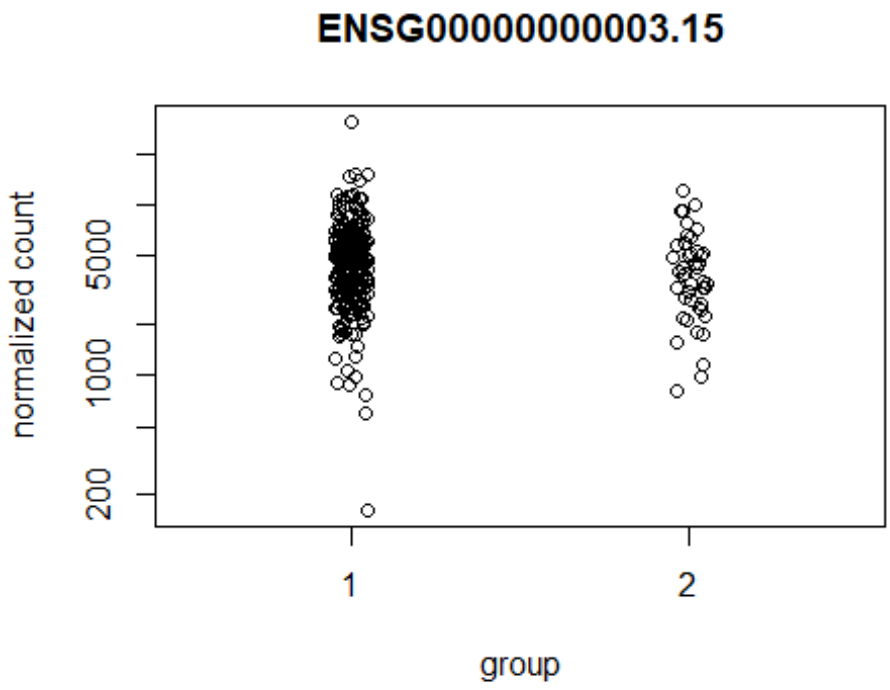
MA-plot

```
plotMA(res_de, ylim=c(-2,2))
```



Plot counts

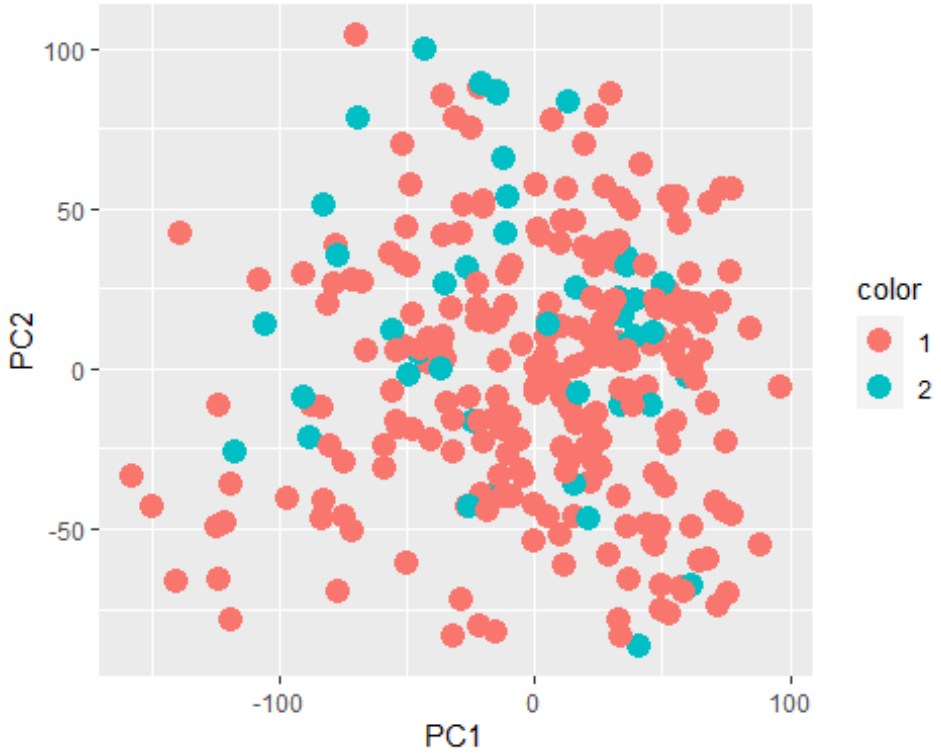
```
plotCounts(dds, gene=which.min(res_de$padj), intgroup="condition")
```



Effect of transformations on the variance

```
# this gives log2(n + 1)
ntd <- normTransform(dds)
# Variance stabilizing transformation
vsd <- vst(dds)

# Regularized log transformation
# The blind=TRUE argument results in a transformation unbiased to sample
condition information.
# rld <- vst(dds, blind=FALSE)

# sampleDists = dist(t(assay(rld)),upper = TRUE)
#
# # annot_col = data.frame(cluster$`cutree(res$tree_col, k = 2)`)
# # row.names(annot_col) <- rownames(clusters)
#
# sampleDistMatrix = as.matrix( sampleDists )
# rownames(sampleDistMatrix) = colnames(data_rna_common)
# colnames(sampleDistMatrix) = colnames(data_rna_common)
#
# pheatmap(sampleDistMatrix,
#          clustering_distance_rows = sampleDists,
#          clustering_distance_cols = sampleDists,
#          cluster_rows=FALSE, show_rownames=TRUE,
#          cluster_cols=TRUE,
#          annotation_col=cluster)
```

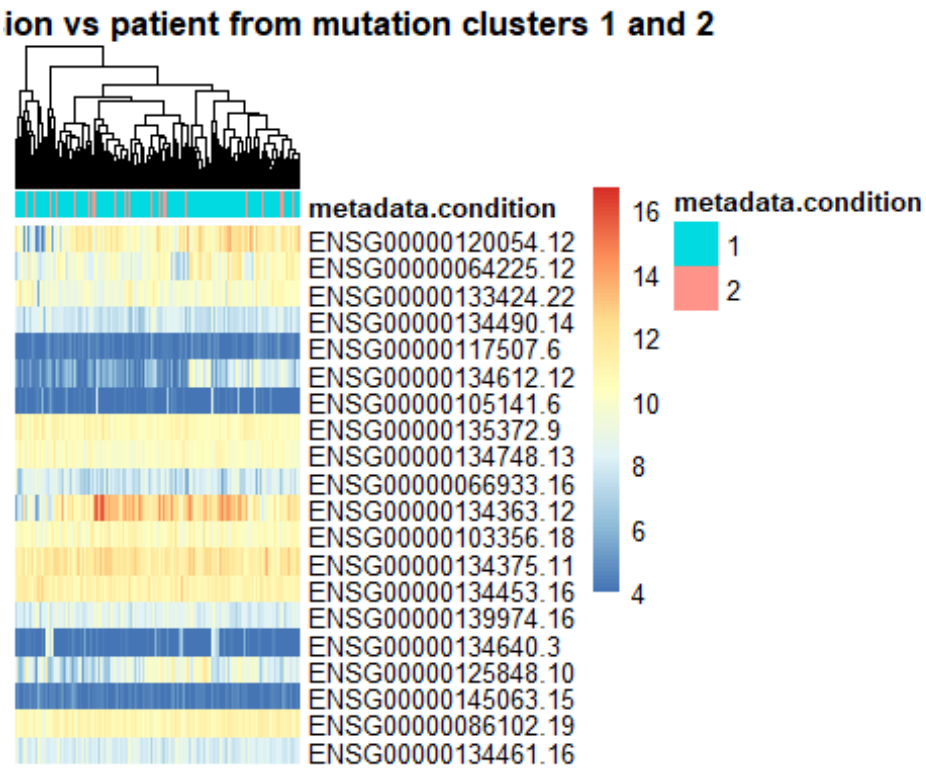
```
pca_res <- prcomp(t(assay(vsd)), scale. = FALSE)
score <- pca_res$x

score = as.data.frame(score)
score$color <- as.factor(cluster$`cutree(res$tree_col, k = 2)`)


ggplot(score, aes(x=PC1, y=PC2,  color=color)) +
  geom_point(size = 4)
```



Selecting the top 20 upregulated (or top 20 downregulated)

```
# Top20genes = head(resSig[order(resSig$log2FoldChange, decreasing=TRUE),], n= 20)

# genes <- order(res$padj, decreasing = TRUE) [1:20]

# we can select a subset of genes to plot.let's choose the 20 genes with the largest positive log2fold change.
genes <- order(res_de$log2FoldChange,decreasing = TRUE)[1:20]

# or largest negative log2fold change
# genes <- order(res$log2FoldChange, decreasing = FALSE)[1:20]

# or we can select the top 20 significant genes
```

```
annot_col = data.frame(metadata$condition)
row.names(annot_col) <- rownames(metadata)

sampleMatrix <- assay(vsd)[genes,]

rownames(sampleMatrix) = rownames(data_rna_common_cluster[genes,])
colnames(sampleMatrix) = colnames(data_rna_common_cluster)

pheatmap(sampleMatrix , cluster_rows=FALSE, show_rownames=TRUE, show_colnames
= FALSE,
         cluster_cols=TRUE, annotation_col=annot_col, main = "Heatmap of Gene
expression vs patient from mutation clusters 1 and 2")
```



Pathway analysis

Adding gene notation

```
columns(org.Hs.eg.db)

##  [1] "ACCNUM"       "ALIAS"        "ENSEMBL"      "ENSEMBLPROT"
"ENSEMBLTRANS"
##  [6] "ENTREZID"     "ENZYME"       "EVIDENCE"     "EVIDENCEALL"
"GENENAME"
## [11] "GENETYPE"     "GO"           "GOALL"        "IPI"          "MAP"
## [16] "OMIM"         "ONTOLOGY"     "ONTOLOGYALL"  "PATH"         "PFAM"
```

```
## [21] "PMID"         "PROSITE"      "REFSEQ"       "SYMBOL"       "UCSCKG"
## [26] "UNIPROT"

row.names(res_de)= substr(row.names(res_de), start = 1, stop = 15)
res_de$symbol = mapIds(org.Hs.eg.db,
                    keys=row.names(res_de),
                    column="SYMBOL",
                    keytype="ENSEMBL",
                    multiVals="first")

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

res_de$entrez = mapIds(org.Hs.eg.db,
                    keys=row.names(res_de),
                    column="ENTREZID",
                    keytype="ENSEMBL",
                    multiVals="first")

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

res_de$name =   mapIds(org.Hs.eg.db,
                    keys=row.names(res_de),
                    column="GENENAME",
                    keytype="ENSEMBL",
                    multiVals="first")

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

head(res_de, 10)

## log2 fold change (MLE): +1,+2
## Wald test p-value: +1,+2
## DataFrame with 10 rows and 9 columns
##                   baseMean log2FoldChange     lfcSE      stat    pvalue
##                  <numeric>      <numeric> <numeric> <numeric> <numeric>
## ENSG00000000003  4974.658       11.84570  0.246020   48.1494         0
## ENSG00000000419  1245.483       10.23732  0.139225   73.5307         0
## ENSG00000000457   542.066        8.78990  0.170892   51.4354         0
## ENSG00000000971 58558.734       15.32732  0.352084   43.5331         0
## ENSG00000001036  2891.948       11.94004  0.224317   53.2283         0
## ENSG00000001084  4340.614       12.74366  0.270735   47.0707         0
## ENSG00000001167   926.293       10.05109  0.205238   48.9728         0
## ENSG00000001461   444.561        8.95119  0.188333   47.5284         0
## ENSG00000001497  2030.143       11.09935  0.160704   69.0672         0
## ENSG00000001617   923.542       10.10906  0.204536   49.4244         0
##                      padj      symbol      entrez                  name
##                 <numeric> <character> <character>           <character>
## ENSG00000000003         0      TSPAN6        7105          tetraspanin 6
## ENSG00000000419         0        DPM1        8813 dolichyl-phosphate m..
## ENSG00000000457         0       SCYL3       57147 SCY1 like pseudokina..
## ENSG00000000971         0         CFH        3075    complement factor H
## ENSG00000001036         0       FUCA2        2519   alpha-L-fucosidase 2
```

```
## ENSG00000001084         0        GCLC        2729 glutamate-cysteine l..
## ENSG00000001167         0        NFYA        4800 nuclear transcriptio..
## ENSG00000001461         0      NIPAL3       57185 NIPA like domain con..
## ENSG00000001497         0       LAS1L       81887 LAS1 like ribosome b..
## ENSG00000001617         0      SEMA3F        6405          semaphorin 3F
```

```
library("org.Hs.eg.db")
library(pathview)
library(gage)
library(gageData)
kegg.sets.hs = data(kegg.sets.hs)
sigmet.idx.hs = data(sigmet.idx.hs)

# Focus on signaling and metabolic pathways only
kegg.sets.hs = kegg.sets.hs[sigmet.idx.hs]
kegg.sets.hs = list(kegg.sets.hs)
# Examine the first 3 pathways
# head(kegg.sets.hs, 3)
```

```
foldchanges = res_de$log2FoldChange
names(foldchanges) = res_de$entrez
head(foldchanges)
```

```
##      7105      8813     57147      3075      2519      2729
## 11.845697 10.237323  8.789901 15.327318 11.940039 12.743663
```

```
# Get the results
keggres = gage(foldchanges, gsets=kegg.sets.hs)
```

```
attributes(keggres)
```

```
## $names
## [1] "greater" "less"    "stats"
```

```
# Look at the first few up (greater) pathways
head(keggres$greater)
```

```
##      p.geomean stat.mean p.val q.val set.size exp1
## [1,]        NA       NaN    NA    NA        1   NA
```

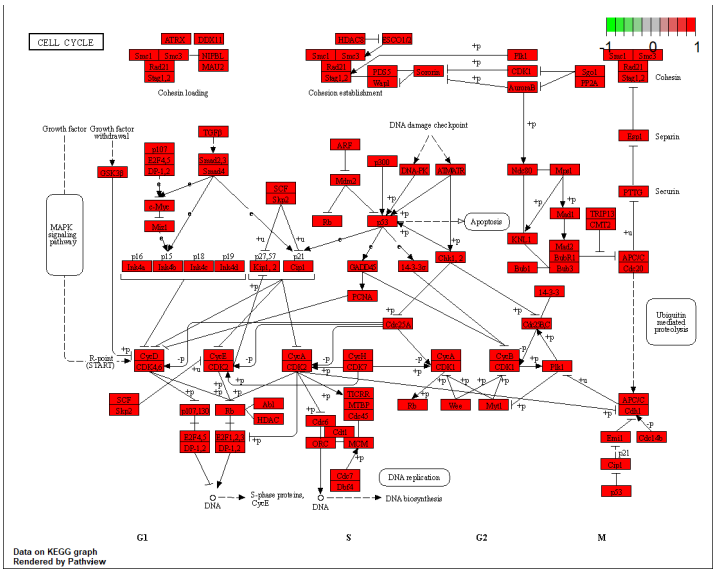
```
pathview(gene.data=foldchanges, pathway.id="hsa04110")
```

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

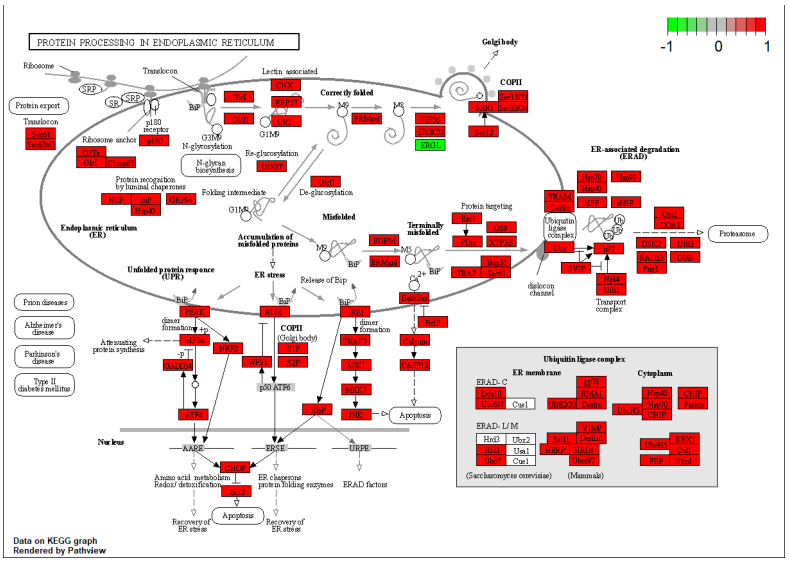
```
## Info: Working in directory D:/Classes/BMEG 310/BMEG310_FinalProject
```

```
## Info: Writing image file hsa04110.pathview.png
```

```
knitr::include_graphics("hsa04110.pathview.png")
```




```
## Focus on top 5 upregulated pathways here for demo purposes only
keggrespathways <- rownames(keggres$greater)[1:1]

# Extract the 8 character long IDs part of each string
keggresids = substr(keggrespathways, start=1, stop=8)
keggresids

## character(0)

pathview(gene.data=foldchanges, pathway.id=keggresids, species="hsa")

## Info: Downloading xml files for hsaNA, 1/1 pathways..

## Warning in download.file(xml.url, xml.target, quiet = T): cannot open URL
## 'https://rest.kegg.jp/get/hsaNA/kgml': HTTP status was '400 Bad Request'

## Warning: Download of hsaNA xml file failed!
## This pathway may not exist!

## Info: Downloading png files for hsaNA, 1/1 pathways..

## Warning: Download of hsaNA png file failed!
## This pathway may not exist!

## Warning: Failed to download KEGG xml/png files, hsa skipped!

knitr::include_graphics("hsa04141.pathview.png")
```


PROTEIN PROCESSING IN ENDOPLASMIC RETICULUM
-1
0
1
Golgi body
Ribosome
Translocon
Lectin associated
COPII
Protein export
Correctly folded
Misfolded
Terminally misfolded
ER-associated degradation (ERAD)
Endoplasmic reticulum (ER)
Accumulation of misfolded proteins
Protein targeting
Ubiquitin ligase complex
Proteasome
Unfolded protein response (UPR)
ER stress
Release of Bip
Prion diseases
Alzheimer's disease
Parkinson's disease
Type II diabetes mellitus
Attenuating protein synthesis
dimer formation
COPII (Golgi body)
Nucleus
AARE
ERSE
UPRE
Amino acid metabolism
Redox/ detoxification
ER chaperone protein folding enzymes
ERAD factors
Apoptosis
Recovery of ER stress
Ubiquitin ligase complex
ER membrane
Cytoplasm
ERAD- C
ERAD- L/M
(Saccharomyces cerevisiae)
(Mammals)
Data on KEGG graph
Rendered by Pathview