Deconvolution Analysis with CIBERSORT

Cristiane Esteves, Mariana Boroni - Bioinformatics and Computational Biology Lab (LBBC/INCA-RJ)

```
#Load libPaths
.libPaths(c("~/deconv_cibersort/deconv_cibersort/lib/","/home/manager/R/x86_64-pc-linux-gnu-library/4.2"))
pkgs <- c("survival", "survminer", "data.table", "dplyr", "ggplot2", "e1071", "parallel", "preprocessCore", "corr
plot", "RColorBrewer", "parallel", "ggdendro")
#install.packages(pkgs)
```

```
#Load Packages
suppressPackageStartupMessages({
  library(tibble)
  library(dplyr)
  library(ggplot2)
  library(survival)
  library(survminer)
  library(e1071)
  library(parallel)
  library(preprocessCore)
  library(data.table)
  library(corrplot)
  library(RColorBrewer)
  library(readr)
})

#Load script CIBERSORT and barplot function
source('CIBERSORT.R')
source('barplot_cibersort.R')
```

#Load signature matrix (LM22) and bulk RNA matrix (SKCM-Metastasis)

LM22 is the signature genes file we used for Cibersort analyses (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4739640/). The file contains expression counts for 547 signature genes (547 rows) for 22 distinct human immune cells (22 columns).

```
lm22_signatures <- as.data.frame(fread("data/lm22.txt"))
```

```
## Warning in fread("data/lm22.txt"): Detected 22 column names but the data has
## 23 columns (i.e. invalid file). Added 1 extra default column name for the first
## column which is guessed to be row names or an index. Use setnames() afterwards
## if this guess is not correct, or fix the file write command that created the
## file to create a valid file.
```

```
print(head(lm22_signatures[,1:4]))
```

```
##       V1 B cells naive B cells memory Plasma cells
## 1  ABCB4     555.71345       10.74423     7.225819
## 2  ABCB9      15.60354       22.09479   653.392328
## 3  ACAP1     215.30595      321.62102    38.616872
## 4   ACHE      15.11795       16.64885    22.123737
## 5   ACP5     605.89738     1935.20148  1120.104684
## 6 ADAM28    1943.74270     1148.12014   324.780800
```

```
lm22_signatures <- tibble::column_to_rownames(lm22_signatures, "V1")

#Bulk TCGA-SKCM metastatic
skcm_bulk <- as.data.frame(fread("data/bulk.txt"))
```

```
## Warning in fread("data/bulk.txt"): Detected 366 column names but the data has
## 367 columns (i.e. invalid file). Added 1 extra default column name for the first
## column which is guessed to be row names or an index. Use setnames() afterwards
## if this guess is not correct, or fix the file write command that created the
## file to create a valid file.
```

```
skcm_bulk <- tibble::column_to_rownames(skcm_bulk, "V1")
print(head(skcm_bulk[,1:4]))
```

```
##        TCGA-EB-A5VV-06A-11R-A32P-07 TCGA-GN-A263-01A-11R-A18T-07
## ABCB4                        8.3243                       8.8439
## ABCB9                        1.1392                       0.6741
## ACAP1                      123.8629                       0.6941
## ACHE                        23.7050                       0.0790
## ACP5                        78.3980                      76.1972
## ADAM28                      43.5955                       0.5815
##        TCGA-HR-A2OG-06A-21R-A18U-07 TCGA-FS-A4F4-06A-12R-A266-07
## ABCB4                        1.8307                       1.3882
## ABCB9                        2.0190                       2.5264
## ACAP1                        6.3677                       2.7528
## ACHE                         1.1324                       1.1902
## ACP5                        66.2074                     132.7469
## ADAM28                       2.8839                       0.7676
```

#Deconvolution Analysis - CIBERSORT

i. perm = No. permutations; set to >=100 to calculate p-values (default = 0)
   ii. QN = Quantile normalization of input mixture (default = TRUE) - (disabling is recommended for RNA-Seq data)
   iii. absolute = Run CIBERSORT in absolute mode (default = FALSE)
      - note that cell subsets will be scaled by their absolute levels and will not be represented as fractions (to derive the default output, normalize absolute levels such that they sum to 1 for each mixture sample)
      - the sum of all cell subsets in each mixture sample will be added to the ouput.

```
set.seed(42)
h1 <- Sys.time()
results.cibersort <- CIBERSORT(lm22_signatures, skcm_bulk, perm = 100, absolute = F, QN = F)
h2 <- Sys.time()
print(h2 - h1)
```

```
## Time difference of 20.61991 mins
```

```
results.sign = as.data.frame(results.cibersort)[which(as.data.frame(results.cibersort)$`P-value` <= 0.05),]
results.sign = results.sign[1:22]
```

```
#Save Cibersort results to directory
saveRDS(results.sign, "~/deconv_cibersort/cancer_genome_analysis_africa/modules/RNA deconvolution/Data_Deconvolut
ion/deconv_cibersort/results_cibersort.rds")
```

```
library(readr)

# Load Metastatic Melanoma (SKCM-TCGA) Clinical and Survival dataset

dados_SKCM = readRDS("data/Dados_SKCM.rds")
subtipos <- read_csv("data/subtipos.csv")
```

```
## New names:
## Rows: 7734 Columns: 11
## ── Column specification
## ──────────────────────────────────────────────────────── Delimiter: "," chr
## (9): pan.samplesID, cancer.type, Subtype_mRNA, Subtype_DNAmeth, Subtype_... dbl
## (2): ...1, Subtype_protein
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## • `` -> `...1`
```

```
#Identify the quartile of each sample in each cell type

rownames(results.sign) <- substr(rownames(results.sign),1,12)

results.sign1 <- results.sign
for (i in 1:length(colnames(results.sign))) {
  for (j in 1:5) {
    quant <- quantile(results.sign1[,i])
    results.sign[which(results.sign1[,i] > quant[j]),i] <- j
  }
}

results.sign$Mixture <- rownames(results.sign)

#Aggregate Cibersort result with clinical and survival data according to patient ID

forest_data <- left_join(results.sign,dados_SKCM$survival_met[,c(1,8,16,17,2,5)], by= c("Mixture" = "bcr_patient_
barcode"))
forest_data <- left_join(forest_data,subtipos[,c(2,10)], by= c("Mixture" = "pan.samplesID"))

#rename column
colnames(forest_data)[29] = "Subtype_Mutation"
```

Barplot

Proportions of the expression predicted by Cibersort pf each celltype

```
# Check the columns (variable names) present in the clinical dataset
names(forest_data)
```

```
##  [1] "B cells naive"                   "B cells memory"
##  [3] "Plasma cells"                    "T cells CD8"
##  [5] "T cells CD4 naive"               "T cells CD4 memory resting"
##  [7] "T cells CD4 memory activated"    "T cells follicular helper"
##  [9] "T cells regulatory (Tregs)"      "T cells gamma delta"
## [11] "NK cells resting"                "NK cells activated"
## [13] "Monocytes"                       "Macrophages M0"
## [15] "Macrophages M1"                  "Macrophages M2"
## [17] "Dendritic cells resting"         "Dendritic cells activated"
## [19] "Mast cells resting"              "Mast cells activated"
## [21] "Eosinophils"                     "Neutrophils"
## [23] "Mixture"                         "Subtype_DNAmeth"
## [25] "OS"                              "OS.time"
## [27] "gender"                          "age_at_initial_pathologic_diagnosis"
## [29] "Subtype_Mutation"
```

```
#Filter for columns sample and Stage
data_barplot = forest_data[,c(23,29)]

#Make patient IDs unique values
data_barplot$Mixture <- make.names(data_barplot$Mixture, unique = T)
data_barplot$Mixture <- gsub("\\.", "-", data_barplot$Mixture)

# Put patient IDs in rownames
rownames(data_barplot) = data_barplot$Mixture
data_barplot$Mixture = NULL

# Add `NA` in empty fields
data_barplot$Subtype_Mutation[which(is.na(data_barplot$Subtype_Mutation))] <- "nan"
data_barplot$Subtype_Mutation[which(data_barplot$Subtype_Mutation == "-")] <- "nan"

# Make Mixture column (Patient IDs from cibersort result table) as first column

res_cibersort = forest_data[, c("Mixture", colnames(forest_data)[1:22])]
res_cibersort$Mixture <- make.names(res_cibersort$Mixture, unique = T)
res_cibersort$Mixture <- gsub("\\.", "-", res_cibersort$Mixture)

#Plot the barplot in which each column is a patient with the clinical informations on the first row (Stage, for e
xample) (colored according to legend colors) and each bar
#is divided by the proportion o immune cells types described also in the legend.

plot.ciber.heat(ciber.obj = res_cibersort, ann_info = data_barplot, sample.column = 1)
```

```
## Loading required package: ggdendro
```

```
## Loading required package: gridExtra
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

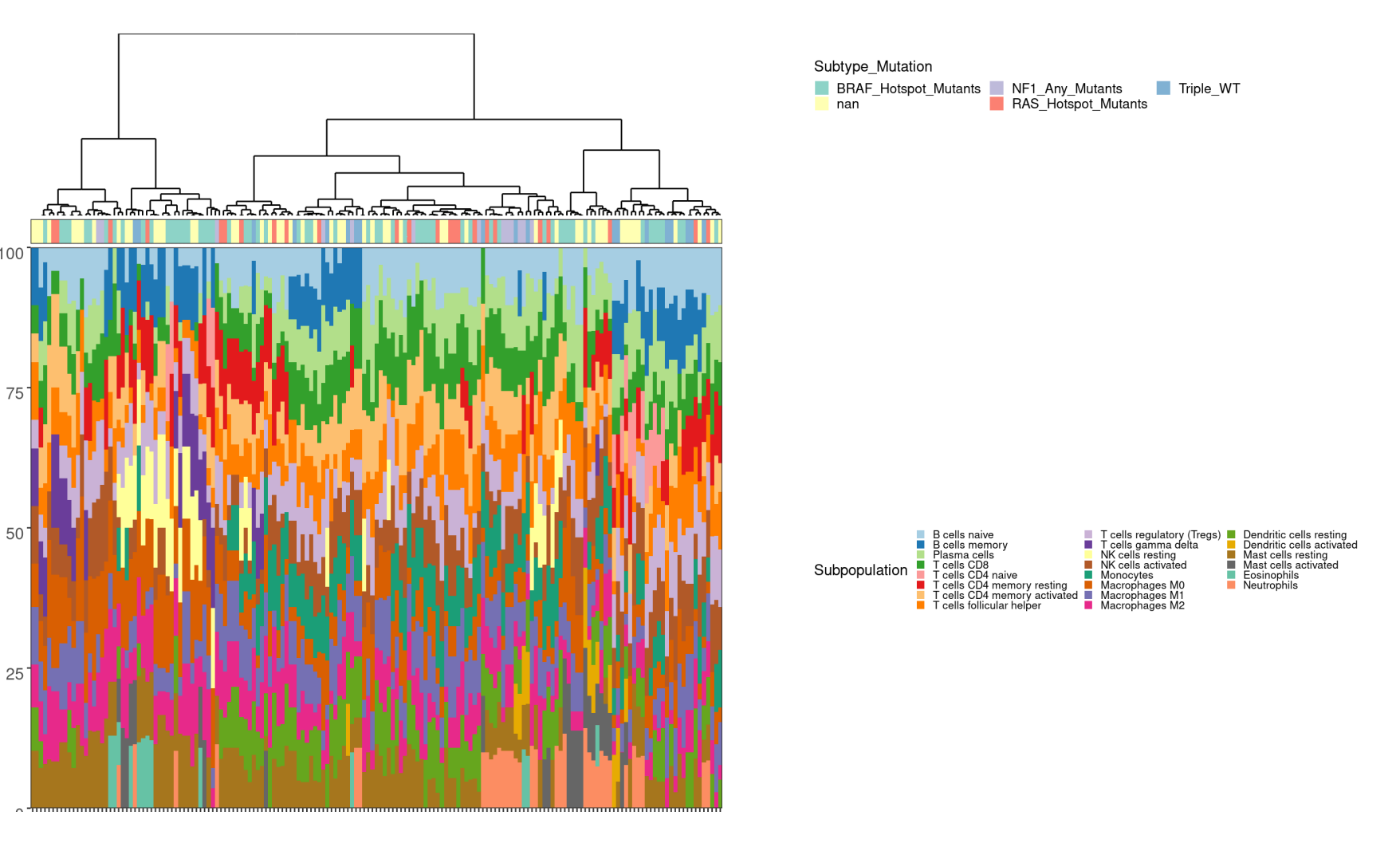
```
## Loading required package: grid
```

```
## Loading required package: cowplot
```

```
##
## Attaching package: 'cowplot'
```

```
## The following object is masked from 'package:ggpubr':
##
##     get_legend
```

```
## Note: Using an external vector in selections is ambiguous.
## ℹ Use `all_of(sample.column)` instead of `sample.column` to silence this message.
## ℹ See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This message is displayed once per session.
## Using Mixture as id variables
##
## Using Mixture as id variables
##
## Joining, by = "Mixture"
```



Univariate and Multivariate (Cox Regression)/Survival analysis

The Cox proportional-hazards model (Cox, 1972) is essentially a regression model commonly used statistical in medical research for investigating the association between the survival time of patients and one or more predictor variables.

In clinical investigations, there are many situations, where several known quantities (known as covariates), potentially affect patient prognosis.

For instance, suppose two groups of patients are compared: those with and those without a specific genotype. If one of the groups also contains older individuals, any difference in survival may be attributable to genotype or age or indeed both. Hence, when investigating survival in relation to any one factor, it is often desirable to adjust for the impact of others.

Statistical model is a frequently used tool that allows to analyze survival with respect to several factors simultaneously. Additionally, statistical model provides the effect size for each factor.

The cox proportional-hazards model is one of the most important methods used for modelling survival analysis data. The next section introduces the basics of the Cox regression model.

References: http://www.sthda.com/english/wiki/cox-proportional-hazards-model https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2394262/

```
library(dplyr)
library(survival)
library(survminer)


# Univariate Cox
#Cox univariate analysis estimated the impact on survival of each cell type.
#For this, it is necessary to have the survival time of each patient and the event variable (in this case, death)
and provide this information as input to the surv() function
surv_object <- Surv(time = forest_data$OS.time, event = forest_data$OS)

colnames(forest_data)[1:22] <- gsub(" ", "_", colnames(forest_data)[1:22])
colnames(forest_data)[9] <- "Treg"
colnames(forest_data)
```

```
##  [1] "B_cells_naive"                   "B_cells_memory"
##  [3] "Plasma_cells"                    "T_cells_CD8"
##  [5] "T_cells_CD4_naive"               "T_cells_CD4_memory_resting"
##  [7] "T_cells_CD4_memory_activated"    "T_cells_follicular_helper"
##  [9] "Treg"                            "T_cells_gamma_delta"
## [11] "NK_cells_resting"                "NK_cells_activated"
## [13] "Monocytes"                       "Macrophages_M0"
## [15] "Macrophages_M1"                  "Macrophages_M2"
## [17] "Dendritic_cells_resting"         "Dendritic_cells_activated"
## [19] "Mast_cells_resting"              "Mast_cells_activated"
## [21] "Eosinophils"                     "Neutrophils"
## [23] "Mixture"                         "Subtype_DNAmeth"
## [25] "OS"                              "OS.time"
## [27] "gender"                          "age_at_initial_pathologic_diagnosis"
## [29] "Subtype_Mutation"
```

```
# Get names of each column (immune cells)
covariables <- colnames(forest_data)[c(1:22,27:29)]
univ_formulas <- sapply(covariables, function(x) as.formula(paste('surv_object ~', x)))

univ_models <- lapply(univ_formulas, function(x){coxph(x, data = forest_data)})

univ_results <- lapply(univ_models,
                       function(x){
                         x <- summary(x)
                         p.value<-signif(x$wald["pvalue"], digits=2)
                         wald.test<-signif(x$wald["test"], digits=2)
                         beta<-signif(x$coef[1], digits=2);#coeficient beta
                         HR <-signif(x$coef[2], digits=2);#exp(beta)
                         HR.confint.lower <- signif(x$conf.int[,"lower .95"], 2)
                         HR.confint.upper <- signif(x$conf.int[,"upper .95"],2)
                         HR <- paste0(HR, " (",
                                      HR.confint.lower, "-", HR.confint.upper, ")")
                         res<-c(beta, HR, wald.test, p.value)
                         names(res)<-c("beta", "HR (95% CI for HR)", "wald.test",
                                       "p.value")
                         return(res)
                         #return(exp(cbind(coef(x),confint(x))))
                       })

res.bisque = as.data.frame(t(do.call(cbind, univ_results)))
```

```
## Warning in (function (..., deparse.level = 1) : number of rows of result is not
## a multiple of vector length (arg 1)
```

```
res.bisque <- as.data.frame(res.bisque)
res.bisque$p.value <- as.character(res.bisque$p.value)
res.bisque$p.value <- as.numeric(res.bisque$p.value)
```

```
## Warning: NAs introduced by coercion
```

```
#Filter for pval =< 0.05
res.bisque_filt <- res.bisque[which(res.bisque$p.value <= 0.05),]

#res.bisque_filt

#Check the immune cells that significantly impact each patient's survival (p val =<0.05)
rownames(res.bisque_filt)
```

```
## [1] "T_cells_CD8"                         "T_cells_CD4_memory_activated"
## [3] "Macrophages_M1"                      "Dendritic_cells_resting"
## [5] "age_at_initial_pathologic_diagnosis"
```

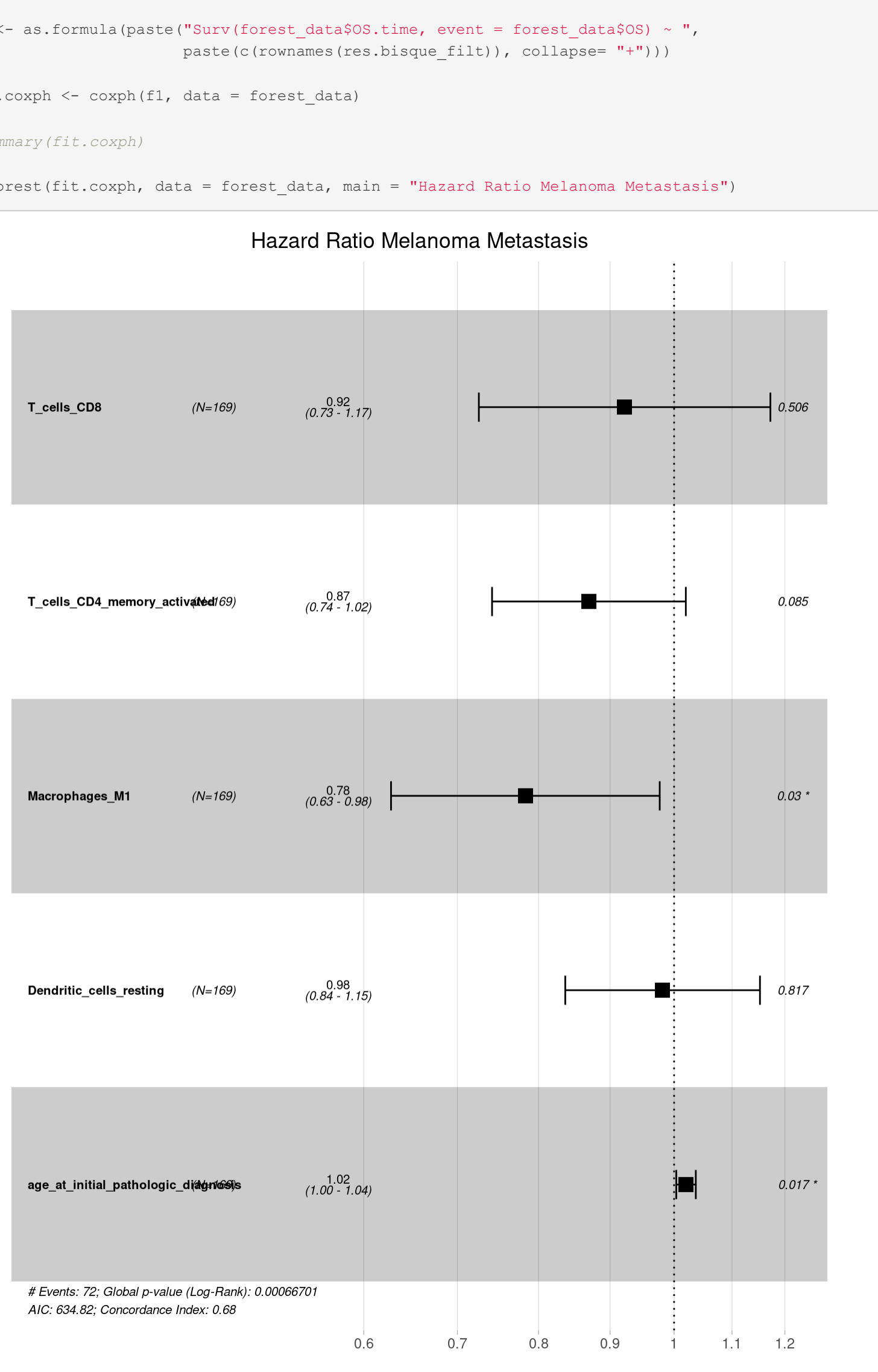
```
#Multivariate Analysis
#Now, we want to describe how the factors jointly impact on survival. To answer to this question, we'll perform a
multivariate Cox regression analysis.

f1 <- as.formula(paste("Surv(forest_data$OS.time, event = forest_data$OS) ~ ",
                       paste(c(rownames(res.bisque_filt)), collapse= "+")))

fit.coxph <- coxph(f1, data = forest_data)

#summary(fit.coxph)

ggforest(fit.coxph, data = forest_data, main = "Hazard Ratio Melanoma Metastasis")
```



The Cox regression results can be interpreted as follow:

Statistical significance. The column marked "z" gives the Wald statistic value. It corresponds to the ratio of each regression coefficient to its standard error (z = coef/se(coef)). The wald statistic evaluates, whether the beta (β) coefficient of a given variable is statistically significantly different from 0. From the output above, we can conclude that the variable sex have highly statistically significant coefficients.

The regression coefficients. The second feature to note in the Cox model results is the the sign of the regression coefficients (coef). A positive sign means that the hazard (risk of death) is higher, and thus the prognosis worse, for subjects with higher values of that variable.

Hazard ratios. The exponentiated coefficients (exp(coef) = exp(-0.53) = 0.59), also known as hazard ratios, give the effect size of covariates. For example, being macrophages M1 reduces the hazard by a factor of 0.78. Being Macrophages M1 is associated with good prognostic.

Confidence intervals of the hazard ratios. The summary output also gives upper and lower 95% confidence intervals for the hazard ratio (exp(coef)).

Global statistical significance of the model. Finally, the output gives p-values for three alternative tests for overall significance of the model: The likelihood-ratio test, Wald test, and score logrank statistics. These three methods are asymptotically equivalent. For large enough N, they will give similar results. For small N, they may differ somewhat. The Likelihood ratio test has better behavior for small sample sizes, so it is generally preferred.

```
######## Levels expressions M1 macrophages in survival Analysis (Kaplan-Meier plot)


library(ggplot2)
library(survival)
library(survminer)


forest_data$Macrophages_M1_group = ifelse(forest_data$Macrophages_M1 >= mean(forest_data$Macrophages_M1), "High",
"Low")

#Having fit a Cox model to the data, it's possible to visualize the predicted survival proportion at any given po
int in time for a particular risk group. The function survfit() estimates the survival proportion, by default at
 the mean values of covariates.

fit <- survfit(Surv(OS.time, OS) ~ Macrophages_M1_group, data = forest_data)
ggsurvplot(fit, palette = c( "#DB7093","#20b2aa"), xlab = "Survival time in years",
           surv.median.line = c("hv"), cumcensor = F,  conf.int = F ,risk.table = TRUE, pval = T,
           title = 'Overall survival: TCGA-SKCM (Macrophages M1)', risk.table.y.text.col = T, # colour risk table
text annotations.
           risk.table.y.text = FALSE, font.main = c(10), font.legend = c(10), font.y = c(10),font.x = c(10), fon
t.caption = c(10),
           font.tickslab = c(10),legend.labs=c("Macrophages M1 High","Macrophages M1 Low"), fontsize = 3,risk.tab
le.height = 0.3, pval.size = 4, censor.size = 2,
           font.ytickslab = c(10))
```