Acute Myeloid Leukemia Heatmap

CCDL for ALSF - Adapted for this repository by Candace Savonen

October 2021

Contents

This analysis has been adapted from this refine.bio-examples notebook

Purpose of the analysis

In this analysis, we will use this acute myeloid leukemia sample dataset from @Shih2017 and pre-processed by @refinebio.

The data that we downloaded from refine.bio for this analysis has 19 samples (obtained from 19 mice with acute myeloid leukemia (AML)), containing RNA-sequencing results for types of AML under controlled treatment conditions.

This dataset can be downloaded from this page on refine.bio. `00-download-data.py` downloads it already processed and quantile normalized.

Set up analysis folders

```
# Create the data folder if it doesn't exist
if (!dir.exists("data")) {
  dir.create("data")
}

# Define the file path to the plots directory
plots_dir <- "plots"

# Create the plots folder if it doesn't exist
if (!dir.exists(plots_dir)) {
  dir.create(plots_dir)
}
```

```r
# Define the file path to the results directory
results_dir <- "results"

# Create the results folder if it doesn't exist
if (!dir.exists(results_dir)) {
  dir.create(results_dir)
}
```

```r
# Define the file path to the data directory
data_dir <- file.path("data", "SRP070849")

# Declare the file path to the gene expression matrix file
data_file <- file.path(data_dir, "SRP070849.tsv")

# Declare the file path to the metadata file
# inside the directory saved as `data_dir`
metadata_file <- file.path(data_dir, "metadata_SRP070849.tsv")
```

Clustering Heatmap - RNA-seq

Install libraries

We will use `pheatmap` [@Slowikowski2017] for clustering and creating a heatmap.

```r
if (!("pheatmap" %in% installed.packages())) {
  # Install pheatmap
  install.packages("pheatmap", update = FALSE)
}
```

Attach the `pheatmap` and `magrittr` libraries:

```r
# Attach the `pheatmap` library
library(pheatmap)

# We will need this so we can use the pipe: %>%
library(magrittr)

# Set the seed so our results are reproducible:
set.seed(12345)
```

Import and set up data

This chunk of code will read in both TSV files and add them as data frames to your environment.

```r
# Read in metadata TSV file
metadata <- readr::read_tsv(metadata_file, show_col_types = FALSE)

# Read in data TSV file
expression_df <- readr::read_tsv(data_file, show_col_types = FALSE) %>%
  # Here we are going to store the gene IDs as row names so that
  # we can have only numeric values to perform calculations on later
  tibble::column_to_rownames("Gene")
```

Let's take a look at the metadata object that we read into the R environment.

```
head(metadata)
```

```
## # A tibble: 6 x 23
##   refinebio_accession_c~1 experiment_accession refinebio_age refinebio_cell_line
##   <chr>                   <chr>                <lgl>         <lgl>
## 1 SRR3189679              SRP070849            NA            NA
## 2 SRR3189680              SRP070849            NA            NA
## 3 SRR3189681              SRP070849            NA            NA
## 4 SRR3189682              SRP070849            NA            NA
## 5 SRR3189683              SRP070849            NA            NA
## 6 SRR3189684              SRP070849            NA            NA
## # i abbreviated name: 1: refinebio_accession_code
## # i 19 more variables: refinebio_compound <lgl>, refinebio_disease <lgl>,
## #   refinebio_disease_stage <lgl>, refinebio_genetic_information <lgl>,
## #   refinebio_organism <chr>, refinebio_platform <chr>,
## #   refinebio_processed <lgl>, refinebio_processor_id <dbl>,
## #   refinebio_processor_name <chr>, refinebio_processor_version <chr>,
## #   refinebio_race <lgl>, refinebio_sex <lgl>, ...
```

Now let's ensure that the metadata and data are in the same sample order.

```
# Make the data in the order of the metadata
expression_df <- expression_df %>%
  dplyr::select(metadata$refinebio_accession_code)

# Check if this is in the same order
all.equal(colnames(expression_df), metadata$refinebio_accession_code)
```

```
## [1] TRUE
```

Now we are going to use the `pheatmap` package to look at how are samples and genes are clustering.

Choose genes of interest

For this example, we will sort genes by variance and select genes in the upper quartile, but there are many alternative criterion by which you may want to sort your genes, e.g. fold change, t-statistic, membership in a particular gene ontology, so on.

```
# Calculate the variance for each gene
variances <- apply(expression_df, 1, var)

# Determine the upper quartile variance cutoff value
upper_var <- quantile(variances, 0.75)

# Filter the data choosing only genes whose variances are in the upper quartile
df_by_var <- data.frame(expression_df) %>%
  dplyr::filter(variances > upper_var)
```

Let's save these to our results folder as a TSV file.

```
readr::write_tsv(df_by_var, file.path(results_dir, "top_75_var_genes.tsv"))
```

Prepare metadata for annotation

From the accompanying paper, we know that the mice with `IDH2` mutant AML were treated with vehicle or AG-221 (the first small molecule in-vivo inhibitor of IDH2 to enter clinical trials) and the mice with `TET2` mutant AML were treated with vehicle or 5-Azacytidine (Decitabine, hypomethylating agent). [@Shih2017]

```
# Let's prepare the annotation for the uncollapsed `DESeqData` set object
# which will be used to annotate the heatmap
annotation_df <- metadata %>%
  # Create a variable to store the cancer type information
  dplyr::mutate(
    mutation = dplyr::case_when(
      startsWith(refinebio_title, "TET2") ~ "TET2",
      startsWith(refinebio_title, "IDH2") ~ "IDH2",
      startsWith(refinebio_title, "WT") ~ "WT",
      # If none of the above criteria are satisfied,
      # we mark the `mutation` variable as "unknown"
      TRUE ~ "unknown"
    )
  ) %>%
  # select only the columns we need for annotation
  dplyr::select(
    refinebio_accession_code,
    mutation,
    refinebio_treatment
  ) %>%
  # The `pheatmap()` function requires that the row names of our annotation
  # data frame match the column names of our `DESeaDataSet` object
  tibble::column_to_rownames("refinebio_accession_code")
```

Create annotated heatmap

The following code generates the heatmap. To interpret the results, look for the following features in the resulting plot:

The Color Scale: We are using a deepskyblue (low) to yellow (high) color scale. Because we set scale = "row", the colors represent the Z-score (standard deviation from the mean) for each gene.

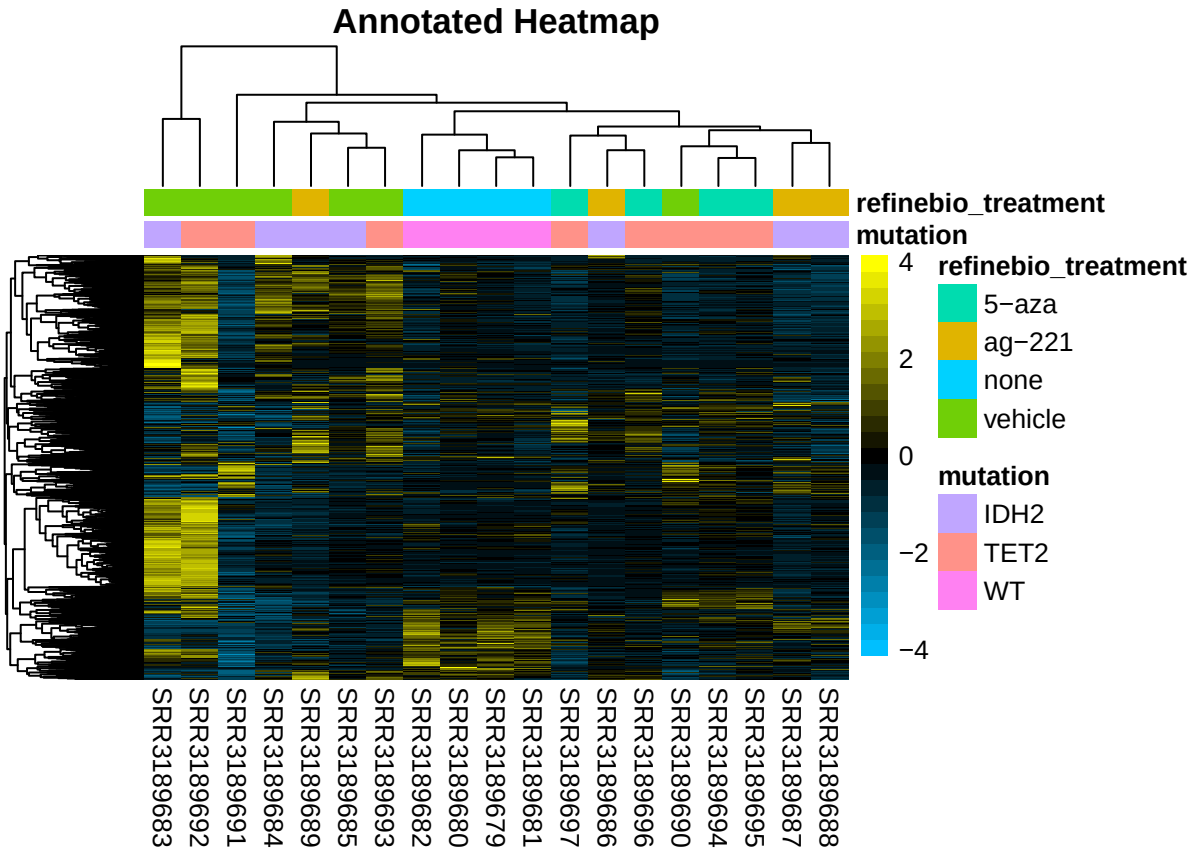
- Yellow: Indicates that the gene is expressed higher in that specific sample compared to the average of that gene across all samples.
- Blue: Indicates that the gene is expressed lower than the average.
- Black: Indicates expression near the average.

The Dendrograms (Trees): The branching structures on the top and left sides represent hierarchical clustering.

- Top Dendrogram (Columns/Samples): Samples that are connected by short branches are mathematically similar to each other. If the biological replicates (e.g., all IDH2 mutants) cluster together, it suggests the mutation drives a distinct gene expression profile.
- Left Dendrogram (Rows/Genes): Genes that cluster together likely behave similarly and may be co-regulated or part of the same biological pathway.

The Annotation Tracks: Look at the colored bars at the very top of the columns. These represent the mutation and refinebio_treatment metadata. Check if the "blocks" of yellow or blue gene expression patterns align vertically with specific mutations or treatments.

```
# Create and store the annotated heatmap object
heatmap_annotated <-
  pheatmap(
    df_by_var,
    cluster_rows = TRUE,
    cluster_cols = TRUE,
    show_rownames = FALSE,
    annotation_col = annotation_df, # Specify our annotation here
    main = "Annotated Heatmap",
    colorRampPalette(c(
      "deepskyblue",
      "black",
      "yellow"
    ))(25
    ),
    scale = "row" # Scale values in the direction of genes (rows)
  )
```



Save annotated heatmap as a PNG

You can switch this to save to a JPEG or TIFF by changing the function and file name within the function to the respective file suffix.

```
# Open a PNG file
png(file.path(
  plots_dir,
  "aml_heatmap.png" # Replace with a relevant file name
))
```

```r
# Print the heatmap
heatmap_annotated

# Close the PNG file:
dev.off()
```

```
## pdf
##   2
```

Session info

```r
# Print session info
sessioninfo::session_info()
```

```
## - Session info -----------------------------------------------------------------
##  setting  value
##  version  R version 4.3.1 (2023-06-16)
##  os       macOS 26.1
##  system   aarch64, darwin20
##  ui       X11
##  language (EN)
##  collate  en_US.UTF-8
##  ctype    en_US.UTF-8
##  tz       Europe/Berlin
##  date     2026-01-04
##  pandoc   3.1.1 @ /Applications/RStudio.app/Contents/Resources/app/quarto/bin/tools/ (via rmarkdown)
##  quarto   1.3.353 @ /Applications/RStudio.app/Contents/Resources/app/quarto/bin/quarto
##
## - Packages ---------------------------------------------------------------------
##  ! package      * version date (UTC) lib source
##    bit            4.6.0   2025-03-06 [1] RSPM (R 4.3.0)
##    bit64          4.6.0-1 2025-01-16 [1] RSPM (R 4.3.0)
##  P cli            3.6.1   2023-03-23 [?] CRAN (R 4.3.0)
##    crayon         1.5.3   2024-06-20 [1] RSPM (R 4.3.0)
##  P digest         0.6.35  2024-03-11 [?] CRAN (R 4.3.1)
##    dplyr          1.1.4   2023-11-17 [1] RSPM (R 4.3.0)
##  P evaluate       0.23    2023-11-01 [?] CRAN (R 4.3.1)
##    farver         2.1.2   2024-05-13 [1] RSPM (R 4.3.0)
##  P fastmap        1.1.1   2023-02-24 [?] CRAN (R 4.3.0)
##    generics       0.1.4   2025-05-09 [1] RSPM (R 4.3.0)
##  P glue           1.6.2   2022-02-24 [?] CRAN (R 4.3.0)
##    gtable         0.3.6   2024-10-25 [1] RSPM (R 4.3.0)
##  P highr          0.10    2022-12-22 [?] CRAN (R 4.3.0)
##    hms            1.1.4   2025-10-17 [1] RSPM (R 4.3.0)
##  P htmltools      0.5.7   2023-11-03 [?] CRAN (R 4.3.1)
##  P knitr          1.45    2023-10-30 [?] CRAN (R 4.3.1)
##  P lifecycle      1.0.3   2022-10-07 [?] CRAN (R 4.3.0)
##  P magrittr     * 2.0.4   2025-09-12 [?] RSPM
##    pheatmap     * 1.0.13  2025-06-05 [1] RSPM (R 4.3.0)
##    pillar         1.11.1  2025-09-17 [1] RSPM (R 4.3.0)
##    pkgconfig      2.0.3   2019-09-22 [1] RSPM (R 4.3.0)
##  P R6             2.5.1   2021-08-19 [?] CRAN (R 4.3.0)
```

```
##    RColorBrewer   1.1-3   2022-04-03 [1] RSPM (R 4.3.0)
##  P readr          2.1.6   2025-11-14 [?] RSPM (R 4.3.0)
##    renv           1.1.4   2025-03-20 [1] CRAN (R 4.3.3)
##  P rlang          1.1.1   2023-04-28 [?] CRAN (R 4.3.0)
##  P rmarkdown      2.26    2024-03-05 [?] CRAN (R 4.3.1)
##  P rstudioapi     0.15.0  2023-07-07 [?] CRAN (R 4.3.0)
##    scales         1.4.0   2025-04-24 [1] RSPM (R 4.3.0)
##  P sessioninfo    1.2.3   2025-02-05 [?] RSPM (R 4.3.0)
##    tibble         3.3.0   2025-06-08 [1] RSPM (R 4.3.0)
##    tidyselect     1.2.1   2024-03-11 [1] RSPM (R 4.3.0)
##    tzdb           0.5.0   2025-03-15 [1] RSPM (R 4.3.0)
##    utf8           1.2.6   2025-06-08 [1] RSPM (R 4.3.0)
##  P vctrs          0.6.5   2023-12-01 [?] RSPM
##    vroom          1.6.7   2025-11-28 [1] RSPM (R 4.3.0)
##  P withr          3.0.2   2024-10-28 [?] RSPM
##  P xfun           0.42    2024-02-08 [?] CRAN (R 4.3.1)
##  P yaml           2.3.8   2023-12-11 [?] CRAN (R 4.3.1)
##
##  [1] /Users/christausch/Library/Mobile Documents/com~apple~CloudDocs/Projects/heatmap-example/renv/li
##  [2] /Users/christausch/Library/Caches/org.R-project.R/R/renv/sandbox/R-4.3/aarch64-apple-darwin20/a
##
##  * -- Packages attached to the search path.
##  P -- Loaded and on-disk path mismatch.
##
## ------------------------------------------------------------------------------
```