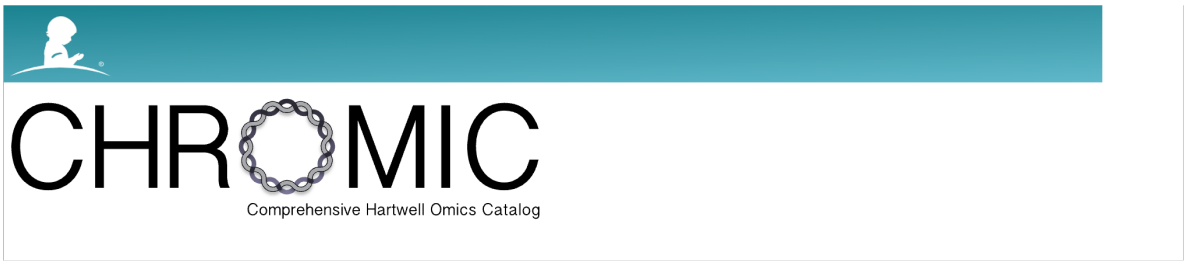

Create cohort

Antonia Chroni achroni@stjude.org for St. Jude Children's Research Hospital BioHackathon Team 1

Contents


```
## The following object is masked _by_ .GlobalEnv:
##
##     root_dir
```

Project: Comprehensive Omics Catalogue for Hartwell

St. Jude Children's Research Hospital BioHackathon Team 1

Date started: 09/04/2024 Date completed: 9/06/2024 Report generated: 18:12:28 CDT 09/11/2024

1 Information about this notebook

This is an exploratory analysis of the data availability in terms of assays in the Comprehensive Omics Catalogue for Hartwell.

For demo purposes, we use dummy data cohort and subset by human brain tumor samples. In addition, we generate random fake SJUID per brain cancer type as this information is not contained in the demo cohort. This cohort will be used for further analysis.

2 Set up

```
suppressPackageStartupMessages({
  library(tidyverse)
})
```

3 Directories and paths to file Inputs/Outputs

```
attach(params)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     root_dir
```

```
analysis_dir <- file.path(root_dir, "analyses", "data-exploratory-analysis")
input_dir <- file.path(analysis_dir, "input")

# We will first read in metadata file as we need to define sample_name
metadata_file <- file.path(input_dir, input_file) # metadata input file

source(paste0(analysis_dir, "/util/generate-fake-SJUID.R"))
```

4 Read metadata file

We will subset by human brain tumor samples.

```
# Read metadata
project_df <- read.csv(metadata_file, stringsAsFactors=FALSE) %>%

  # Add cancer_type_brain: Ependymoma, HGG, LGG, Medulloblastoma
  add_column(cancer_type_brain = "other") %>%
  mutate(cancer_type_brain = case_when(grepl("Ependymoma", Disease) ~ "Ependymoma",
                                       grepl("HGG", Disease) ~ "HGG",
                                       grepl("LGG", Disease) ~ "LGG",
                                       grepl("MedulloBlastoma", Disease) ~ "Medulloblastoma"),
         Assay = Omics.Method) %>%
  mutate(across(where(is.character), ~ na_if(.,""))) %>% # Omics Method for NA
  filter(!cancer_type_brain == "other",
         !is.na(cancer_type_brain),
         !is.na(Omics.Method),
         Source == "Human") %>%
  select(Source, Disease, Assay, Omics.Method.Detail, Site, Sub.Group, cancer_type_brain)
```

4.1 Generate fake SJUID

We will generate random fake SJUID per brain cancer type as this information is not contained in the demo cohort.

```
# Create a smaller set of unique strings, each starting with "SJH"
unique_strings <- paste0("SJH", sapply(1:80, function(x) generate_string(8)))

# Sample from this set to create a vector of 100 strings, allowing duplicates
SJUID <- sample(unique_strings, 100, replace = TRUE)

# Generate vector for `cancer_type_brain`
cancer_type_brain_vec <- c("Ependymoma", "HGG", "LGG", "Medulloblastoma")
n <- 25 # random number
cancer_type_brain <- rep(cancer_type_brain_vec, each=n)

# Assign `SJUID` to `cancer_type_brain`
bind_df <- cbind(SJUID, cancer_type_brain)  %>%
  as.data.frame()

# Merge both df
df <- project_df %>%
  left_join(bind_df, by = "cancer_type_brain", relationship = "many-to-many") %>%
  unique() %>%
  mutate(match_id_assay = paste(SJUID, Assay, sep = "_")) %>%
  distinct(match_id_assay, .keep_all = TRUE) %>%
  add_column(samples_drop = "keep") %>%
  mutate(samples_drop = case_when(grepl("SJH5HCKPC97", SJUID) & grepl("WES", Assay) ~ "drop",
                                  grepl("SJH98QNKIUU", SJUID) & grepl("WES", Assay) ~ "drop",
                                  grepl("SJH0H5WYREP", SJUID) & grepl("RNAseq", Assay) ~ "drop",
                                  grepl("SJHKYUIYAME", SJUID) & grepl("Methylation", Assay) ~ "drop",
                                  grepl("SJHFGGJRHYO", SJUID) & grepl("RNAseq", Assay) ~ "drop",
                                  grepl("SJHKVQEBOCP", SJUID) & grepl("Methylation", Assay) ~ "drop",
                                  grepl("SJHUMKP2L6V", SJUID) & grepl("ATACseq", Assay) ~ "drop",
                                  grepl("SJHCNO3RCVD", SJUID) & grepl("ATACseq", Assay) ~ "drop",
                                  grepl("SJHBNJSZHW6", SJUID) & grepl("RNAseq", Assay) ~ "drop",
                                  grepl("SJHDDTEOSYL", SJUID) & grepl("Methylation|ATACseq", Assay) ~ "
                                  grepl("SJHWSONRZVA", SJUID) & grepl("WGS", Assay) ~ "drop",
                                  grepl("SJHOYO5OJJN", SJUID) & grepl("WGS", Assay) ~ "drop",
                                  grepl("SJHHW23UJ9P", SJUID) & grepl("ATACseq|WGS", Assay) ~ "drop",
                                  grepl("SJHBISK5KBU", SJUID) & grepl("WGS", Assay) ~ "drop",
                                  grepl("SJHROJUQZAP", SJUID) & grepl("RNAseq", Assay) ~ "drop",
                                  grepl("SJHC1QST5GR", SJUID) & grepl("ATACseq", Assay) ~ "drop",
                                  grepl("SJHHZH67WMF", SJUID) & grepl("ATACseq|ChIPseq", Assay) ~ "drop
                                  grepl("SJH8HIE3POP", SJUID) & grepl("Methylation|ChIPseq|RNAseq", Assa
                                  grepl("SJHZZLRCGJ6", SJUID) & grepl("RNAseq", Assay) ~ "drop",
                                  grepl("SJHOYO5OJJN", SJUID) & grepl("ChIPseq", Assay) ~ "drop",
                                  grepl("SJHNPJTQHIT", SJUID) & grepl("RNAseq", Assay) ~ "drop",
                                  grepl("SJHEQG3P4FK", SJUID) & grepl("RNAseq|WES", Assay) ~ "drop",
                                  TRUE ~ samples_drop)) %>%
  filter(samples_drop == "keep") %>%
  select(!samples_drop)

# Save file
write.csv(df, file.path(input_dir, "cohort.csv"), row.names = FALSE)
```

4.2 Generate fake longitudinal data

We will generate random longitudinal data per SJUID and per brain cancer type as this information is not contained in the demo cohort.

```
# Generate vector for `cancer_type_brain`
longitudinal_vec <- c("Diagnosis", "Remission", "Relapse", "Deceased")
n <- length(df$SJUID)
longitudinal <- rep(longitudinal_vec, each=n)

# Assign `longitudinal` TO df
#longitudinal_bind_df <- cbind(df, longitudinal) %>%
#  as.data.frame()

longitudinal_df <- cbind(df, longitudinal) %>%
  unique() %>%
  mutate(match_id_longitudinal_assay = paste(SJUID, longitudinal, Assay, sep = "_")) %>%
  mutate(match_id_longitudinal = paste(SJUID, longitudinal, sep = "_")) %>%
  distinct(match_id_longitudinal_assay, .keep_all = TRUE)

# Select random SJUID and longitudinal and create uneven longitudinal information/SJUID
# list <- unique(longitudinal_df$SJUID)
# list

filter_longitudinal_df <-longitudinal_df %>%
  add_column(samples_drop = "keep") %>%
  mutate(samples_drop = case_when(grepl("SJH5HCKPC97", SJUID) & grepl("Diagnosis", longitudinal) ~ "drop
                                  grepl("SJH98QNKIUU", SJUID) & grepl("Diagnosis", longitudinal) ~ "drop
                                  grepl("SJHOH5WYREP", SJUID) & grepl("Diagnosis", longitudinal) ~ "drop
                                  grepl("SJHKYUIYAME", SJUID) & grepl("Remission", longitudinal) ~ "drop
                                  grepl("SJHFGGJRHYO", SJUID) & grepl("Diagnosis", longitudinal) ~ "drop
                                  grepl("SJHKVQEBOCP", SJUID) & grepl("Remission", longitudinal) ~ "drop
                                  grepl("SJHUMKP2L6V", SJUID) & grepl("Relapse", longitudinal) ~ "drop"
                                  grepl("SJHCNO3RCVD", SJUID) & grepl("Relapse", longitudinal) ~ "drop"
                                  grepl("SJHBNJSZHW6", SJUID) & grepl("Diagnosis", longitudinal) ~ "drop
                                  grepl("SJHDDTEOSYL", SJUID) & grepl("Remission|Deceased", longitudinal
                                  grepl("SJHWSONRZVA", SJUID) & grepl("Deceased", longitudinal) ~ "drop
                                  grepl("SJHOYO5OJJN", SJUID) & grepl("Deceased", longitudinal) ~ "drop
                                  grepl("SJHHW23UJ9P", SJUID) & grepl("Diagnosis|Deceased", longitudinal
                                  grepl("SJHBISK5KBU", SJUID) & grepl("Deceased", longitudinal) ~ "drop
                                  grepl("SJHROJUQZAP", SJUID) & grepl("Deceased", longitudinal) ~ "drop
                                  grepl("SJHC1QST5GR", SJUID) & grepl("Diagnosis", longitudinal) ~ "drop
                                  grepl("SJHHZH67WMF", SJUID) & grepl("Diagnosis|Remission", longitudina
                                  grepl("SJH8HIE3POP", SJUID) & grepl("Diagnosis|Remission|Deceased", lo
                                  grepl("SJHZZLRCGJ6", SJUID) & grepl("Deceased", longitudinal) ~ "drop
                                  grepl("SJHOYO5OJJN", SJUID) & grepl("Remission", longitudinal) ~ "drop
                                  grepl("SJHNPJTQHIT", SJUID) & grepl("Remission", longitudinal) ~ "drop
                                  grepl("SJHEQG3P4FK", SJUID) & grepl("Remission|Deceased", longitudinal
                                  TRUE ~ samples_drop)) %>%
  filter(samples_drop == "keep") %>%
  select(!samples_drop)

# Save file
write.csv(filter_longitudinal_df, file.path(input_dir, "cohort-longitudinal.csv"), row.names = FALSE)
```

5 Session Info

```
## R version 4.4.0 (2024-04-24)
## Platform: x86_64-pc-linux-gnu
## Running under: Red Hat Enterprise Linux 8.8 (Ootpa)
##
## Matrix products: default
## BLAS:   /research/rgs01/applications/hpcf/authorized_apps/rhel8_apps/lapack/3.10.1/install/lib64/libl
## LAPACK: /research/rgs01/applications/hpcf/authorized_apps/rhel8_apps/lapack/3.10.1/install/lib64/libl
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## time zone: America/Chicago
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] lubridate_1.9.3 forcats_1.0.0   stringr_1.5.1   dplyr_1.1.4
##  [5] purrr_1.0.2     readr_2.1.5     tidyr_1.3.1     tibble_3.2.1
##  [9] ggplot2_3.5.1   tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] gtable_0.3.5      jsonlite_1.8.8    compiler_4.4.0    tidyselect_1.2.1
##  [5] jquerylib_0.1.4   scales_1.3.0      yaml_2.3.10       fastmap_1.2.0
##  [9] mime_0.12         R6_2.5.1          generics_0.1.3    knitr_1.48
## [13] munsell_0.5.1     bslib_0.8.0       pillar_1.9.0      tzdb_0.4.0
## [17] rlang_1.1.4       utf8_1.2.4        stringi_1.8.4     cachem_1.1.0
## [21] xfun_0.47         sass_0.4.9        timechange_0.3.0  cli_3.6.3
## [25] withr_3.0.1       magrittr_2.0.3    digest_0.6.37     grid_4.4.0
## [29] hms_1.1.3         lifecycle_1.0.4   vctrs_0.6.5       evaluate_0.24.0
## [33] glue_1.7.0        fansi_1.0.6       colorspace_2.1-1  rmarkdown_2.28
## [37] tools_4.4.0       pkgconfig_2.0.3   htmltools_0.5.8.1
```