IOBR (Immuno-Oncology Biological Research)

Dongqiang Zeng, Yiran Fang

2024-01-11

Contents

Introduction

Preface

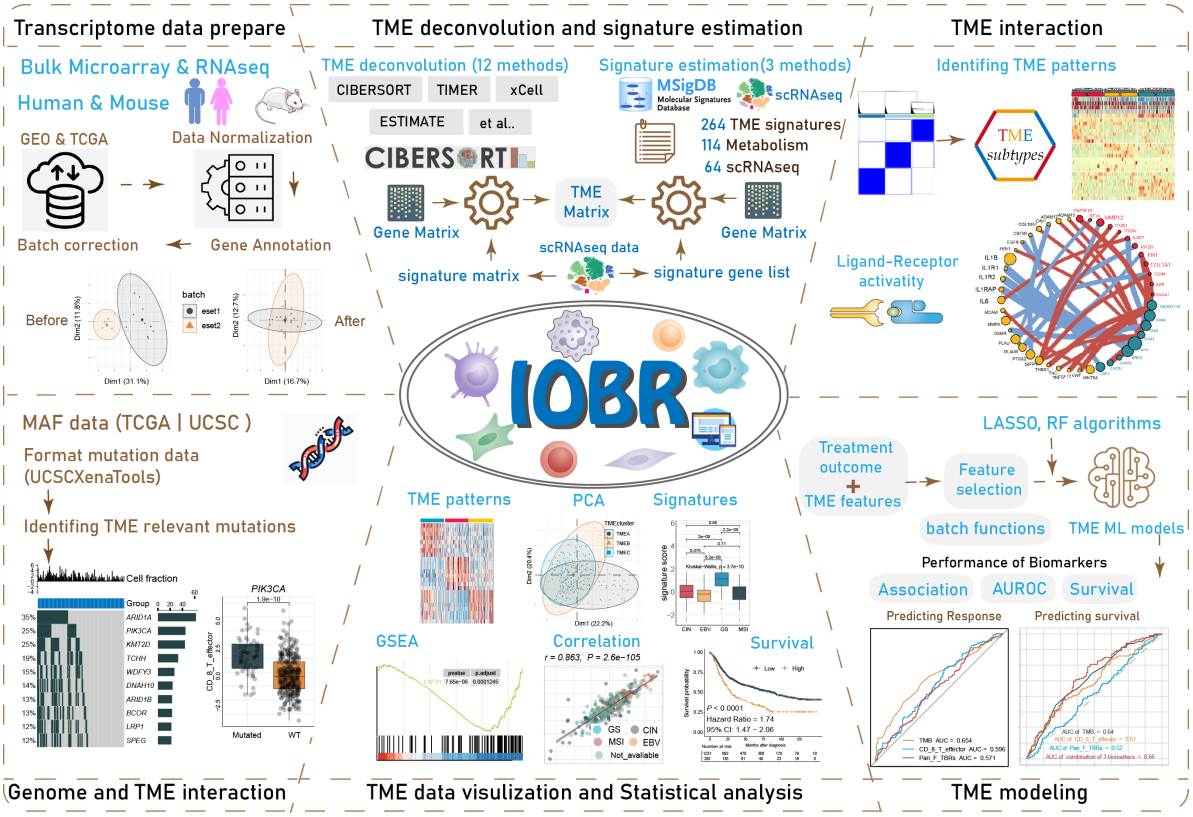


Figure 1: The workflow of IOBR

0.1 Introduction

IOBR is the acronym for Immuno-Oncology Biological Research. Recent advances in next-generation sequencing have triggered the rapid accumulation of publicly available multi-omics data. The application of integrated omics to explore robust signatures for clinical translation is increasingly highlighted in immuno-oncology, but poses computational and biological chal-

lenges. This vignette aims to demonstrate how to use the package named IOBR to perform multi-omics immuno-oncology biological research to decipher tumour microenvironment and signatures for clinical translation.

This R package integrates 8 published methods for decoding the tumour microenvironment (TME) context: `CIBERSORT`, `TIMER`, `xCell`, `MCPcounter`, `ESITMATE`, `EPIC`, `IPS`, `quanTIseq`. In addition, 264 published signature gene sets have been collected by IOBR covering tumour microenvironment, tumour metabolism, m6A, exosomes, microsatellite instability and tertiary lymphoid structure. The `signature_collection_citation` function is run to obtain the source papers, and the `signature_collection` function returns the detailed signature genes of all given signatures. IOBR then uses three computational methods to calculate the signature score, including `PCA`, `z-score` and `ssGSEA`. Note that IOBR collected and used several approaches for variable transition, visualisation, batch survival analysis, feature selection and statistical analysis. Batch analysis and visualisation of results are supported. The details of how IOBR works are described below.

0.2 License

0.3 Previous publication

Zeng D, Ye Z, Shen R, Yu G, Wu J, Xiong Y,..., Liao W (2021) **IOBR**: Multi-Omics Immuno-Oncology Biological Research to Decode Tumor Microenvironment and Signatures. *Frontiers in Immunology*. 12:687975. doi: 10.3389/fimmu.2021.687975

Zeng D, Fang Y, ..., Liao W (2023) **IOBR2**: Multidimensional Decoding of Tumor Microenvironment for Immuno-Oncology Research. *bioRxiv*.

0.4 Major Updates

0.5 Reporting bugs

Please report bugs to the Github issues page

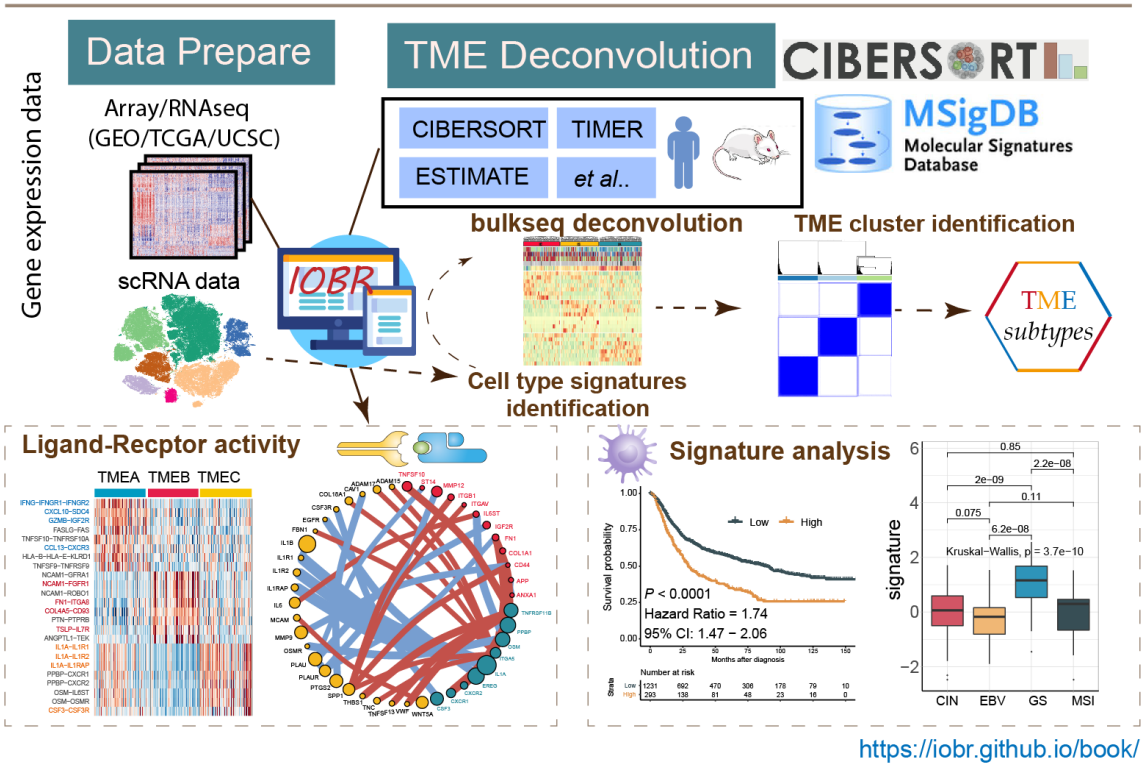


Figure 2: The workflow of IOBR

E-mail any questions to Dr. Fang fyr_nate@163.com or Dr. Zeng interlaken@smu.edu.cn

Chapter 1

How to install IOBR

1.1 Install Dependency Packages

It is essential that you have R 3.6.3 or above already installed on your computer or server. IOBR is a pipeline that utilizes many other R packages that are currently available from CRAN, Bioconductor and GitHub.

```
if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")
depens<-c('tibble', 'survival', 'survminer', 'limma', "DESeq2","devtools", 'limSolve', '
          "devtools", "tidyHeatmap", "caret", "glmnet", "ppcor",  "timeROC", "pracma", "
          "FactoMineR", "WGCNA", "patchwork", 'ggplot2', "biomaRt", 'ggpubr', "PMCMRplus
for(i in 1:length(depens)){
  depen<-depens[i]
  if (!requireNamespace(depen, quietly = TRUE))  BiocManager::install(depen,update = FA
}
```

1.2 Install IOBR package

When the dependent environments are built, users are able to install IOBR from github by typing the following code into your R session:

```
if (!requireNamespace("IOBR", quietly = TRUE))  devtools::install_github("IOBR/IOBR")

library(IOBR)
```

1.3 How to update IOBR

```
detach("package:IOBR")
path<-.libPaths()
remove.packages(c('IOBR'), lib=file.path(path))
devtools::install_github("IOBR/IOBR")
```

Chapter 2

How to use IOBR

2.1 The main pipeline of IOBR

2.2 Main Functions of IOBR

- **Data Preparation: data annotation and transformation**
  - **count2tpm()**: transform gene expression count data into Transcripts Per Million (TPM) values. This function supports gene IDs of type "Ensembl", "Entrez", or "Symbol", and retrieves gene length information using either an online connection to the bioMart database or a local dataset (specified by the source parameter).
  - **anno_eset()**: annotate an ExpressionSet object (eset) with gene symbols using the provided annotation data. It retains only the rows with probes that have matching identifiers in the annotation data. The function handles duplicates according to the specified method. The output is an annotated and cleaned expression set.
  - **remove_duplicate_genes()**: remove duplicate gene symbols from gene expression data. The retention of gene symbols is based either on their mean values (if method is set as "mean") or standard deviation values (if method is set as "sd").
  - **mouse2human_eset()**: convert mouse gene symbols to human gene symbols of expression set.
  - **find_outlier_samples()**: analyze gene expression data and identify potential outlier samples based on connectivity analysis. By utilizing the "WGCNA" package, this function calculates the normalized adjacency and connectivity z-scores for each sample. It also offers multiple parameters to customize analysis and visualization.

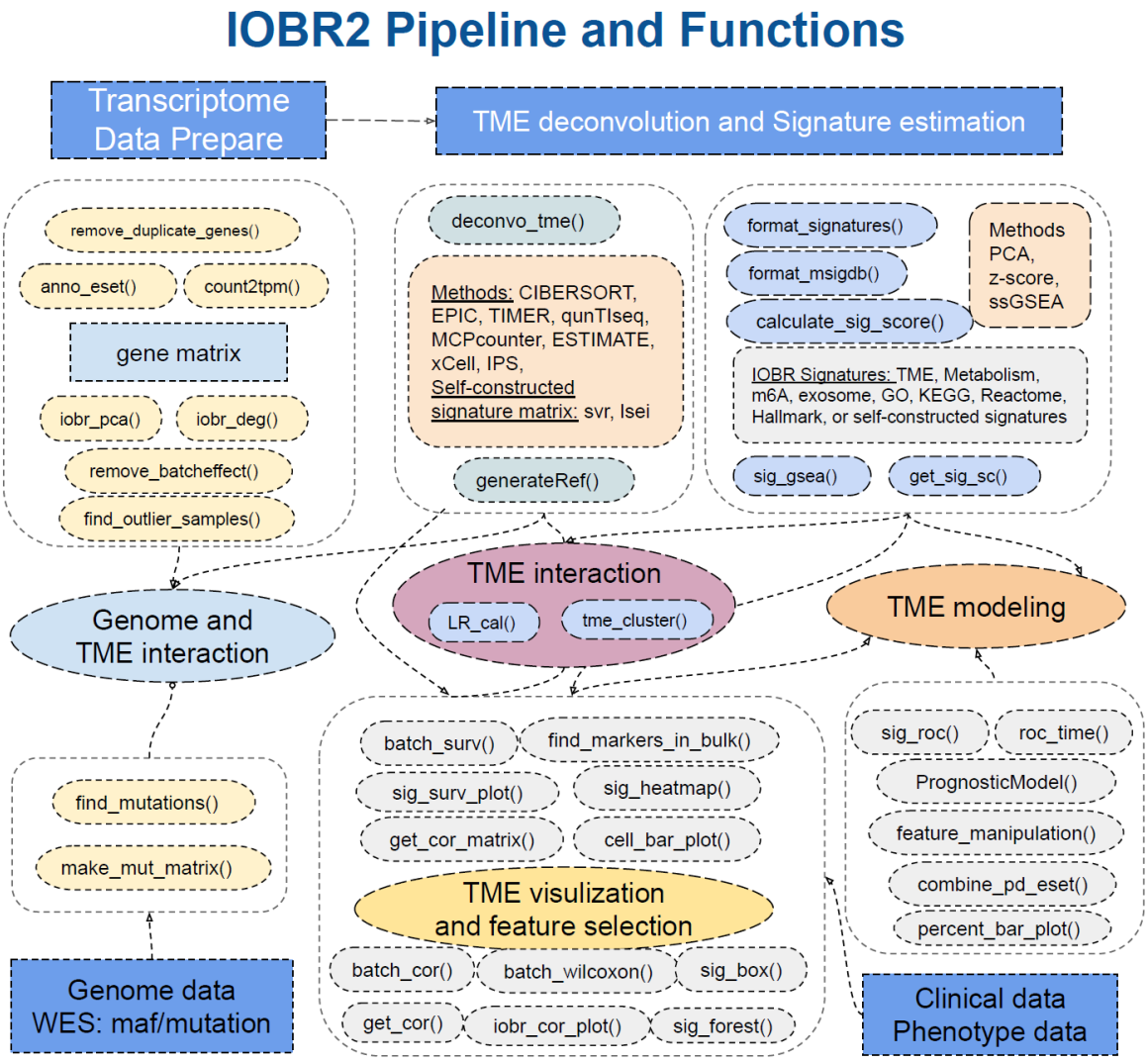


Figure 2.1: The main pipeline of IOBR

  - **remove_batcheffect()**: remove batch effects from given expression datasets and visualize the corrected data using principal component analysis (PCA). It takes three expression datasets as input and performs batch effect correction using the "sva::ComBat" or "sva::ComBat_seq" methods. The function then generates PCA plots to compare the data before and after correction.

- **TME Deconvolution Module: integrate multiple algorithms to decode immune contexture**

  - **deconvo_tme()**: decode the TME infiltration using various deconvolution methodologies, based on bulk RNAseq, microarray or single cell RNAseq data. It currently supports methods include "CIBERSORT", "MCPcounter", "EPIC", "xCell", "IPS", "estimate", "quanTIseq", "TIMER", "SVR" and "lsei".
  - **generateRef()**: generate a novel gene reference data for specific feature deconvolution, such as infiltrating cell, utilizing different methods to identify differentially expressed genes (DEGs) . The function supports both "limma" and "DESeq2" methods.The resulting gene reference data can be used for **deconvo_tme()** with the "svr" and "lsei" algorithms.
  - **generateRef_seurat()**: take a Seurat object "sce" and additional parameters to perform various operations for generating reference gene expression data. It allows for specifying cell types, proportions, assays, preprocessing options, and statistical testing parameters.The resulting gene reference data can be used for **deconvo_tme()** with the "svr" and "lsei" algorithms.

- **Signature Module: calculating signature scores, estimate phenotype related signatures and corresponding genes, and evaluate signatures generated from single-cell RNA sequencing data**

  - **calculate_sig_score()**: estimate the interested signatures enrolled in IOBR R package, which involves TME-associated, tumor-metabolism, and tumor-intrinsic signatures.The supported methods for signature score calculation include "PCA", "ssGSEA", "z-score", and Integration.
  - **feature_manipulation()**: manipulate features including the cell fraction and signatures generated from multi-omics data for latter analysis and model construction. Remove missing values, outliers and variables without significant variance.
  - **format_signatures()**: generate the object for **calculate_sig_score()** function, by inputting a data frame with signatures as column names of corresponding gene sets, and return a list contain the signature information for calculating multiple signature scores.
  - **format_msigdb()**: transform the signature gene sets data with gmt format,

which is not included in the signature collection and might be downloaded in the MSgiDB website, into the object of **calculate_sig_score()** function.

- **sig_gsea()**: conduct Gene Set Enrichment Analysis (GSEA) to identify significant gene sets based on differential gene expression data. This function performs GSEA using the fgsea package and provides visualizations and results in the form of tables and plots. It supports the utilization of user-defined gene sets or the use of predefined gene sets from MSigDB.
- **get_sig_sc()**: get top gene signatures from single-cell differential analysis for **calculate_sig_score()** function. The input is a matrix containing a ranked list of putative markers, and associated statistics (p-values, ROC score, etc.)

- **Batch Analysis and Visualization: batch survival analysis and batch correlation analysis and other batch statistical analyses**

  - **batch_surv()**: perform batch survival analysis. It calculates hazard ratios and confidence intervals for the specified variables based on the given data containing time-related information.
  - **subgroup_survival()**: extract hazard ratio and confidence intervals from a coxph object of subgroup analysis.
  - **batch_cor()**: batch analysis of correlation between two continuous variables using Pearson correlation coefficient or Spearman's rank correlation coefficient.
  - **batch_wilcoxon()**: perform Wilcoxon rank-sum tests on a given data set to compare the distribution of a specified feature between two groups. It computes the p-values and ranks the significant features based on the p-values. It returns a data frame with the feature names, p-values, adjusted p-values, logarithm of p-values, and a star rating based on the p-value ranges.
  - **batch_pcc()**: provide a batch way to calculate the partial correlation coefficient between feature and others when controlling a third variable.
  - **iobr_cor_plot()**: visualization of batch correlation analysis of signatures from "sig_group". Visualize the correlation between signature or phenotype with expression of gene sets in target signature is also supported.
  - **cell_bar_plot()**: batch visualization of TME cell fraction, supporting input of deconvolution results from "CIBERSORT", "EPIC" and "quanTIseq" methodologies to further compare the TME cell distributions within one sample or among different samples.
  - **iobr_pca()**: perform Principal Component Analysis (PCA), which reduces the dimensionality of data while maintaining most of the original variance, and visualizes the PCA results on a scatter plot.

- **iobr_deg()**: perform differential expression analysis on gene expression data using the DESeq2 or limma method. It filters low count data, calculates fold changes and adjusted p-values, and identifies DEGs based on specified cutoffs. It also provides optional visualization tools such as volcano plots and heatmaps.
- **get_cor()**: calculate and visualize the correlation between two variables in a dataset. It provides options to scale the data, handle missing values, and incorporate additional data. The function supports various correlation methods. It generates a correlation plot with optional subtypes or categories, including a regression line.
- **get_cor_matrix()**: calculate and visualize the correlation matrix between two sets of variables in a dataset. It provides flexibility in defining correlation methods, handling missing values, and incorporating additional data. The function supports various correlation methods, such as Pearson correlation, and displays the correlation result in a customizable plot.
- **roc_time()**: generate a Receiver Operating Characteristic (ROC) plot over time to assess the predictive performance of one or more variables in survival analysis. It calculates the Area Under the Curve (AUC) for each specified time point and variable combination, and creates a multi-line ROC plot with corresponding AUC values annotated.
- **sig_box()**: generate a boxplot with optional statistical comparisons. It takes in various parameters such as data, signature, variable, and more to customize the plot. It can be used to visualize and analyze data in a Seurat object or any other data frame.
- **sig_heatmap()**: generate a heatmap plot based on input data, grouping variables, and optional conditions. The function allows customization of various parameters such as palette selection, scaling, color boxes, plot dimensions, and more. It provides flexibility in visualizing relationships between variables and groups in a concise and informative manner.
- **sig_forest()**: create a forest plot for visualizing survival analysis results generated by "batch_surv".
- **sig_roc()**: plot multiple ROC curves in a single graph, facilitating the comparison of different variables in terms of their ability to predict a binary response.
- **sig_surv_plot()**: generat multiple Kaplan-Meier (KM) survival plots for a given signature or gene. It allows for detailed customization and is structured to handle various aspects of survival analysis.
- **find_markers_in_bulk()**: find relevant results from the given gene expression data and meta information. It leverages the "Seurat" package to identify

significant markers across multiple groups within the given data. The supported methods for comparison include "bootstrap", "delong" and "venkatraman".

- **Signature Associated Mutation Module: identify and analyze mutations relevant to targeted signatures**
  - **make_mut_matrix()**: transform the mutation data with MAF format(contain the columns of gene ID and the corresponding gene alterations which including SNP, indel and frameshift) into a mutation matrix in a suitable manner for further investigating signature relevant mutations.
  - **find_mutations()**: identify mutations associated with a distinct phenotype or signature. The function conducts the Cuzick test, Wilcoxon test, or both (when the method is set to "multi"). It generates box plots for the top genes identified through these statistical tests and creates oncoprints to graphically represent the mutation landscape across samples.
- **Model Construction Module: feature selection and fast model construct to predict clinical phenotype**
  - **BinomialModel()**: select features and construct a model to predict a binary phenotype. It accepts a dataset (x and y) as input and performs data processing, splitting into training and testing sets, and model fitting using both Lasso and Ridge regression techniques.
  - **PrognosticMode()**: select features and construct a model to predict clinical survival outcome. It primarily focuses on developing Lasso and Ridge regression models within the Cox proportional hazards framework.
  - **combine_pd_eset()**: combine the expression set (eset) with phenotype data (pdata).
  - **percent_bar_plot()**: create a percent bar plot based on the given data. The input is a data frame, with x and y-axis variables specified.

2.3 Current working environment

```
library(IOBR)
```

```
## Loading required package: tibble
```

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

```
## Loading required package: dplyr
```

```
## Warning: package 'dplyr' was built under R version 4.2.3
##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
## Loading required package: ggplot2
## Warning: package 'ggplot2' was built under R version 4.2.3
## Loading required package: ggpubr
## Warning: package 'ggpubr' was built under R version 4.2.3
## Loading required package: survival
## Loading required package: ComplexHeatmap
## Loading required package: grid
## ========================================
## ComplexHeatmap version 2.15.4
## Bioconductor page: http://bioconductor.org/packages/ComplexHeatmap/
## Github page: https://github.com/jokergoo/ComplexHeatmap
## Documentation: http://jokergoo.github.io/ComplexHeatmap-reference
##
## If you use it in published research, please cite either one:
## - Gu, Z. Complex Heatmap Visualization. iMeta 2022.
## - Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional
##     genomic data. Bioinformatics 2016.
##
##
## The new InteractiveComplexHeatmap package can directly export static
## complex heatmaps into an interactive Shiny app with zero effort. Have a try!
##
## This message can be suppressed by:
##   suppressPackageStartupMessages(library(ComplexHeatmap))
```

```
## ========================================

## Loading required package: tidyHeatmap

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

## ========================================
## tidyHeatmap version 1.8.1
## If you use tidyHeatmap in published research, please cite:
## 1) Mangiola et al. tidyHeatmap: an R package for modular heatmap production
##   based on tidy principles. JOSS 2020.
## 2) Gu, Z. Complex heatmaps reveal patterns and correlations in multidimensional
##   genomic data. Bioinformatics 2016.
## This message can be suppressed by:
##   suppressPackageStartupMessages(library(tidyHeatmap))
## ========================================

##
## Attaching package: 'tidyHeatmap'

## The following object is masked from 'package:stats':
##
##     heatmap

## Loading required package: clusterProfiler

##

## Registered S3 methods overwritten by 'treeio':
##   method              from
##   MRCA.phylo          tidytree
##   MRCA.treedata       tidytree
##   Nnode.treedata      tidytree
##   Ntip.treedata       tidytree
##   ancestor.phylo      tidytree
##   ancestor.treedata   tidytree
##   child.phylo         tidytree
##   child.treedata      tidytree
##   full_join.phylo     tidytree
##   full_join.treedata  tidytree
##   groupClade.phylo    tidytree
##   groupClade.treedata tidytree
```

```
##   groupOTU.phylo      tidytree
##   groupOTU.treedata   tidytree
##   is.rooted.treedata  tidytree
##   nodeid.phylo        tidytree
##   nodeid.treedata     tidytree
##   nodelab.phylo       tidytree
##   nodelab.treedata    tidytree
##   offspring.phylo     tidytree
##   offspring.treedata  tidytree
##   parent.phylo        tidytree
##   parent.treedata     tidytree
##   root.treedata       tidytree
##   rootnode.phylo      tidytree
##   sibling.phylo       tidytree

## clusterProfiler v4.4.4  For help: https://yulab-smu.top/biomedical-knowledge-mining-b
##
## If you use clusterProfiler in published research, please cite:
## T Wu, E Hu, S Xu, M Chen, P Guo, Z Dai, T Feng, L Zhou, W Tang, L Zhan, X Fu, S Liu,

##
## Attaching package: 'clusterProfiler'

## The following object is masked from 'package:stats':
##
##     filter

## Loading required package: patchwork

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

## Loading required package: survminer

##
## Attaching package: 'survminer'

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

## ========================================================================
##   IOBR v0.99.8  Immuno-Oncology Biological Research
```

```
##   For Tutorial: https://iobr.github.io/book/
##   For Help: https://github.com/IOBR/IOBR/issues
##
##  If you use IOBR in published research, please cite:
##  DQ Zeng, ZL Ye, RF Sheng, GC Yu, Y Xiong …, WJ Liao*.
##  IOBR: Multi-omics Immuno-Oncology Biological Research to decode
##   tumor microenvironment and signatures. Frontiers in Immunology. 12:687975,(2021).
##  DOI: 10.3389/fimmu.2021.687975
##  Higly Cited Paper and Hot Paper of WOS
## ========================================================================
```

```
sessionInfo()
```

```
## R version 4.2.0 (2022-04-22 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19045)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=Chinese (Simplified)_China.utf8
## [2] LC_CTYPE=Chinese (Simplified)_China.utf8
## [3] LC_MONETARY=Chinese (Simplified)_China.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=Chinese (Simplified)_China.utf8
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] IOBR_0.99.8           survminer_0.4.9       patchwork_1.1.3
##  [4] clusterProfiler_4.4.4 tidyHeatmap_1.8.1     ComplexHeatmap_2.15.4
##  [7] survival_3.3-1        ggpubr_0.6.0          ggplot2_3.4.4
## [10] dplyr_1.1.2           tibble_3.2.1
##
## loaded via a namespace (and not attached):
##   [1] utf8_1.2.2                  tidyselect_1.2.0
```

```
##   [3] RSQLite_2.3.2               AnnotationDbi_1.58.0
##   [5] BiocParallel_1.30.3         lpSolve_5.6.19
##   [7] scatterpie_0.2.1            ScaledMatrix_1.4.1
##   [9] munsell_0.5.0               codetools_0.2-19
##  [11] preprocessCore_1.58.0       withr_2.5.2
##  [13] colorspace_2.0-3            GOSemSim_2.22.0
##  [15] Biobase_2.56.0              limSolve_1.5.7
##  [17] knitr_1.45                  rstudioapi_0.15.0
##  [19] SingleCellExperiment_1.18.1 stats4_4.2.0
##  [21] ggsignif_0.6.4              DOSE_3.22.1
##  [23] MatrixGenerics_1.8.1        GenomeInfoDbData_1.2.8
##  [25] KMsurv_0.1-5                polyclip_1.10-6
##  [27] bit64_4.0.5                 farver_2.1.1
##  [29] rhdf5_2.40.0                downloader_0.4
##  [31] vctrs_0.6.4                 treeio_1.21.2
##  [33] generics_0.1.3              xfun_0.40
##  [35] R6_2.5.1                    doParallel_1.0.17
##  [37] GenomeInfoDb_1.34.4         clue_0.3-61
##  [39] graphlayouts_1.0.1          rsvd_1.0.5
##  [41] locfit_1.5-9.8              rhdf5filters_1.8.0
##  [43] bitops_1.0-7                cachem_1.0.6
##  [45] fgsea_1.22.0                gridGraphics_0.5-1
##  [47] DelayedArray_0.22.0         scales_1.2.1
##  [49] ggraph_2.1.0                enrichplot_1.16.2
##  [51] gtable_0.3.4                beachmat_2.12.0
##  [53] tidygraph_1.2.3             rlang_1.1.1
##  [55] genefilter_1.78.0           GlobalOptions_0.1.2
##  [57] splines_4.2.0               rstatix_0.7.2
##  [59] lazyeval_0.2.2              broom_1.0.5
##  [61] yaml_2.3.7                  reshape2_1.4.4
##  [63] abind_1.4-5                 backports_1.4.1
##  [65] qvalue_2.28.0               tools_4.2.0
##  [67] bookdown_0.36               ggplotify_0.1.2
##  [69] RColorBrewer_1.1-3          proxy_0.4-27
##  [71] BiocGenerics_0.42.0         Rcpp_1.0.9
##  [73] plyr_1.8.7                  sparseMatrixStats_1.8.0
##  [75] zlibbioc_1.42.0             purrr_1.0.2
```

```
##  [77] RCurl_1.98-1.7              GetoptLong_1.0.5
##  [79] viridis_0.6.4               cowplot_1.1.1
##  [81] S4Vectors_0.34.0            zoo_1.8-10
##  [83] SummarizedExperiment_1.26.1 ggrepel_0.9.1
##  [85] cluster_2.1.3               fs_1.5.2
##  [87] magrittr_2.0.3              data.table_1.14.2
##  [89] DO.db_2.9                   circlize_0.4.15
##  [91] matrixStats_0.62.0          GSVA_1.44.5
##  [93] evaluate_0.22               xtable_1.8-4
##  [95] XML_3.99-0.14               IRanges_2.30.0
##  [97] gridExtra_2.3               shape_1.4.6
##  [99] compiler_4.2.0              crayon_1.5.2
## [101] shadowtext_0.1.2            htmltools_0.5.6.1
## [103] ggfun_0.1.3                 tidyr_1.3.0
## [105] geneplotter_1.74.0          aplot_0.2.2
## [107] DBI_1.1.3                   tweenr_2.0.2
## [109] corrplot_0.92               MASS_7.3-60
## [111] Matrix_1.6-3                car_3.1-2
## [113] cli_3.6.1                   quadprog_1.5-8
## [115] parallel_4.2.0              igraph_1.3.2
## [117] GenomicRanges_1.48.0        pkgconfig_2.0.3
## [119] km.ci_0.5-6                 foreach_1.5.2
## [121] ggtree_3.4.4                annotate_1.74.0
## [123] XVector_0.36.0              yulab.utils_0.1.0
## [125] stringr_1.5.0               digest_0.6.29
## [127] graph_1.74.0                Biostrings_2.64.0
## [129] rmarkdown_2.25              fastmatch_1.1-4
## [131] survMisc_0.5.6              tidytree_0.4.5
## [133] dendextend_1.17.1           DelayedMatrixStats_1.18.2
## [135] GSEABase_1.58.0             rjson_0.2.21
## [137] lifecycle_1.0.3             nlme_3.1-157
## [139] jsonlite_1.8.0              Rhdf5lib_1.18.2
## [141] carData_3.0-5               viridisLite_0.4.2
## [143] limma_3.52.4                fansi_1.0.3
## [145] pillar_1.9.0                lattice_0.20-45
## [147] KEGGREST_1.36.3             fastmap_1.1.1
## [149] httr_1.4.7                  GO.db_3.15.0
```

```
## [151] glue_1.6.2                 png_0.1-7
## [153] iterators_1.0.14           glmnet_4.1-8
## [155] bit_4.0.5                  HDF5Array_1.24.2
## [157] ggforce_0.4.1              class_7.3-22
## [159] stringi_1.7.6              blob_1.2.4
## [161] BiocSingular_1.12.0        DESeq2_1.36.0
## [163] memoise_2.0.1              irlba_2.3.5
## [165] tidyverse_2.0.0            e1071_1.7-13
## [167] ape_5.6-2
```

Chapter 3

RNA Data preprocessing

3.1 Loading packages

Load the IOBR package in your R session after the installation is complete:

```
library(IOBR)
library(tidyverse)
library(clusterProfiler)
```

3.2 Download array data using GEOquery

Obtaining data set from GEO Gastric cancer: GSE62254 using GEOquery R package.

```
if (!requireNamespace("GEOquery", quietly = TRUE))  BiocManager::install("GEOquery")
library("GEOquery")
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo<-getGEO(GEO     = "GSE62254", getGPL  = F, destdir = "./")
eset    <-eset_geo[[1]]
eset    <-exprs(eset)
eset[1:5,1:5]
```

```
##           GSM1523727 GSM1523728 GSM1523729 GSM1523744 GSM1523745
## 1007_s_at  3.2176645  3.0624323  3.0279131   2.921683  2.8456013
## 1053_at    2.4050109  2.4394879  2.2442708   2.345916  2.4328582
## 117_at     1.4933412  1.8067380  1.5959665   1.839822  1.8326058
## 121_at     2.1965561  2.2812181  2.1865556   2.258599  2.1874363
```

```
## 1255_g_at  0.8698382  0.9502466  0.8125414   1.012860  0.9441993
```

3.3 Gene Annotation

Annotation of genes in the expression matrix and removal of duplicate genes.

```
# Load the annotation file `anno_hug133plus2` in IOBR.
head(anno_hug133plus2)
```

```
## # A tibble: 6 x 2
##   probe_id  symbol
##   <fct>     <fct>
## 1 1007_s_at MIR4640
## 2 1053_at   RFC2
## 3 117_at    HSPA6
## 4 121_at    PAX8
## 5 1255_g_at GUCA1A
## 6 1294_at   MIR5193
```

```
# Load the annotation file `anno_grch38` in IOBR.
head(anno_grch38)
```

```
##                id eff_length        gc entrez   symbol chr     start       end
## 1 ENSG00000000003       4536 0.3992504   7105   TSPAN6   X 100627109 100639991
## 2 ENSG00000000005       1476 0.4241192  64102     TNMD   X 100584802 100599885
## 3 ENSG00000000419       9276 0.4252911   8813     DPM1  20  50934867  50958555
## 4 ENSG00000000457       6883 0.4117391  57147    SCYL3   1 169849631 169894267
## 5 ENSG00000000460       5970 0.4298157  55732 C1orf112   1 169662007 169854080
## 6 ENSG00000000938       3382 0.5644589   2268      FGR   1  27612064  27635277
##   strand        biotype
## 1     -1 protein_coding
## 2      1 protein_coding
## 3     -1 protein_coding
## 4     -1 protein_coding
## 5      1 protein_coding
## 6     -1 protein_coding
##
## 1                                                    tetraspanin 6 [Source:HGNC
## 2                                                      tenomodulin [Source:HGNC
```

```
## 3 dolichyl-phosphate mannosyltransferase polypeptide 1, catalytic subunit [Source:HGN
## 4                                                SCY1-like, kinase-like 3 [Source:HGNC
## 5                                     chromosome 1 open reading frame 112 [Source:HGNC
## 6                            FGR proto-oncogene, Src family tyrosine kinase [Source:HGN
```

```
# Load the annotation file `anno_gc_vm32` in IOBR for mouse RNAseq data
head(anno_gc_vm32)
```

```
##                   id eff_length        gc symbol      mgi_id      gene_type
## 1 ENSMUSG00000000001       3262 0.4350092  Gnai3   MGI:95773 protein_coding
## 2 ENSMUSG00000000003        902 0.3481153   Pbsn MGI:1860484 protein_coding
## 3 ENSMUSG00000000028       3506 0.4962921  Cdc45 MGI:1338073 protein_coding
## 4 ENSMUSG00000000031       2625 0.5588571    H19   MGI:95891         lncRNA
## 5 ENSMUSG00000000037       6397 0.4377052  Scml2 MGI:1340042 protein_coding
## 6 ENSMUSG00000000049       1594 0.5050188   Apoh   MGI:88058 protein_coding
##       start       end transcript_id  ont
## 1 108014596 108053462          <NA> <NA>
## 2  76881507  76897229          <NA> <NA>
## 3  18599197  18630737          <NA> <NA>
## 4 142129262 142131886          <NA> <NA>
## 5 159865521 160041209          <NA> <NA>
## 6 108234180 108305222          <NA> <NA>
```

3.3.1 For Array data: HGU133PLUS-2 (Affaymetrix)

```
# Conduct gene annotation using `anno_hug133plus2` file; If identical gene symbols exi

eset<-anno_eset(eset       = eset,
                annotation = anno_hug133plus2,
                symbol     = "symbol",
                probe      = "probe_id",
                method     = "mean")
eset[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
```

```
## LOC101928826   4.219303   4.219670   4.213252
```

3.4 Download RNAseq data using UCSCXenaTools

In this section, we are going to download RNA-seq data from The Cancer Genome Atlas (TCGA) for applying the downstream analysis workflow of **IOBR**. Particularly, we will use the convenient R package **UCSCXenaTools** to query and download the RNA-seq data of TCGA stomach cancer cohort.

Use the following code to check and install **UCSCXenaTools**.

```
if (!requireNamespace("UCSCXenaTools", quietly = TRUE))
  BiocManager::install("UCSCXenaTools")
```

UCSCXenaTools provides an R interface to access public cancer datasets from UCSC Xena data hubs, including multiple pan-cancer studies like TCGA and PCAWG. You can directly access information of all datasets in R.

```
library(UCSCXenaTools)
```

```
## Warning: package 'UCSCXenaTools' was built under R version 4.2.1
```

```
## ================================================================================
## UCSCXenaTools version 1.4.8
## Project URL: https://github.com/ropensci/UCSCXenaTools
## Usages: https://cran.r-project.org/web/packages/UCSCXenaTools/vignettes/USCSXenaTools
##
## If you use it in published research, please cite:
## Wang et al., (2019). The UCSCXenaTools R package: a toolkit for accessing genomics da
##   from UCSC Xena platform, from cancer multi-omics to single-cell RNA-seq.
##   Journal of Open Source Software, 4(40), 1627, https://doi.org/10.21105/joss.01627
## ================================================================================
##                                --Enjoy it--
```

```
##
## Attaching package: 'UCSCXenaTools'
```

```
## The following object is masked from 'package:Biobase':
##
##     samples
```

```
head(XenaData)
```

```
## # A tibble: 6 x 17
##   XenaHosts XenaHostNames XenaCohorts XenaDatasets SampleCount DataSubtype Label
##   <chr>     <chr>         <chr>       <chr>              <int> <chr>       <chr>
## 1 https://~ publicHub     Breast Can~ ucsfNeve_pu~          51 gene expre~ Neve~
## 2 https://~ publicHub     Breast Can~ ucsfNeve_pu~          57 phenotype   Phen~
## 3 https://~ publicHub     Glioma (Ko~ kotliarov20~         194 copy number Kotl~
## 4 https://~ publicHub     Glioma (Ko~ kotliarov20~         194 phenotype   Phen~
## 5 https://~ publicHub     Lung Cance~ weir2007_pu~         383 copy number CGH
## 6 https://~ publicHub     Lung Cance~ weir2007_pu~         383 phenotype   Phen~
## # i 10 more variables: Type <chr>, AnatomicalOrigin <chr>, SampleType <chr>,
## #   Tags <chr>, ProbeMap <chr>, LongTitle <chr>, Citation <chr>, Version <chr>,
## #   Unit <chr>, Platform <chr>
```

```
# You can use view(XenaData) to find your dataset of interest
```

UCSCXenaTools provides workflow functions to generate object, filter, query, download and load the dataset(s) of interest. The following code show a standardized **UCSCXenaTools** data workflow to query the data from UCSC Xena data hub and load it into R.

```
library(UCSCXenaTools)
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_stad<-XenaGenerate(subset = XenaCohorts =="GDC TCGA Stomach Cancer (STAD)") %>%
  XenaFilter(filterDatasets    = "TCGA-STAD.htseq_counts.tsv") %>%
  XenaQuery() %>%
  XenaDownload() %>%
  XenaPrepare()
eset_stad[1:5, 1:3]
```

As the metadata of this dataset have been stored in the `XeneData` data.frame. You can easily recheck the dataset with code.

```
dplyr::filter(XenaData, XenaDatasets == "TCGA-STAD.htseq_counts.tsv") |>
  as.list()
```

3.5 Normalization and Gene annotation

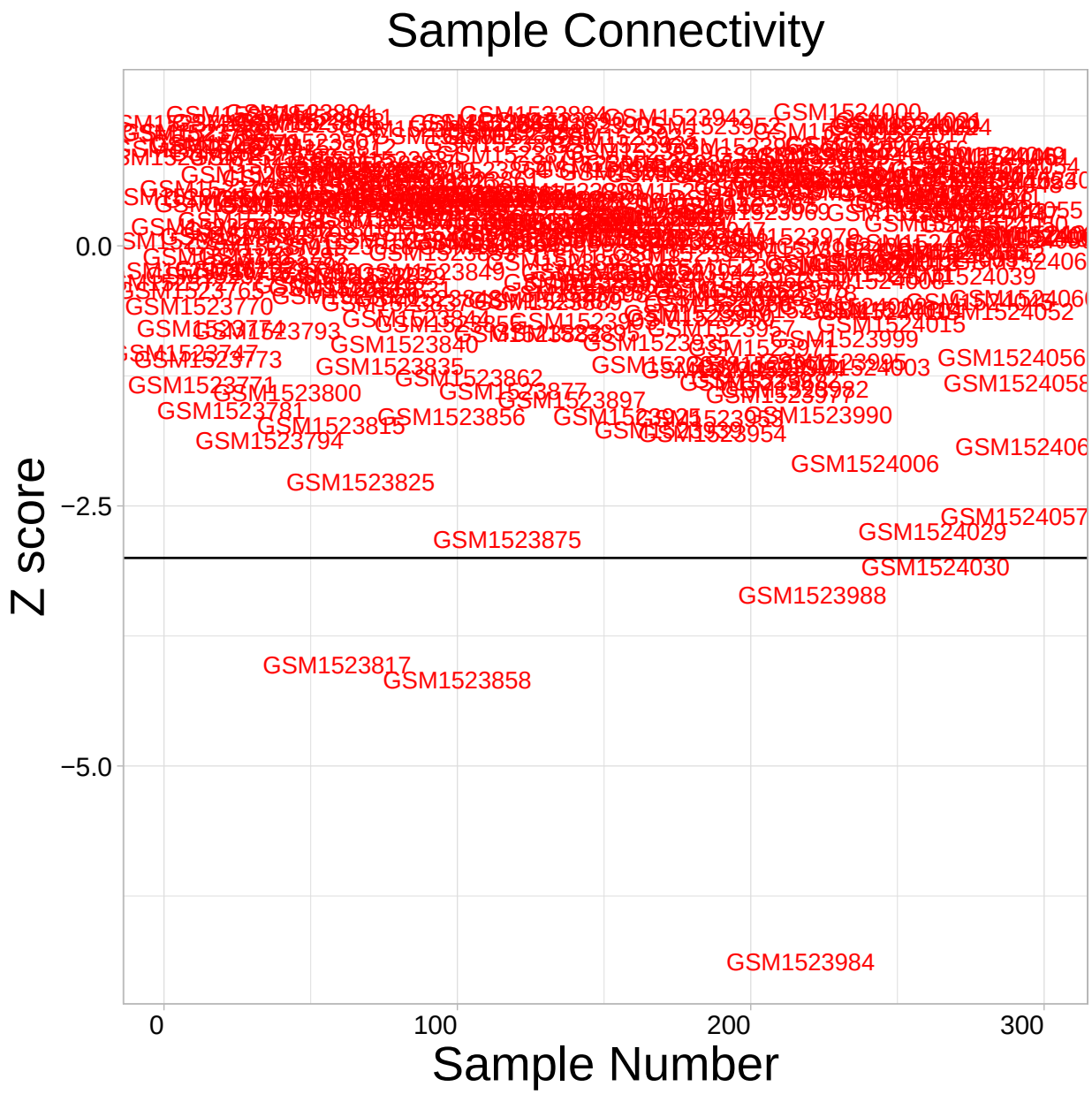
Transform gene expression matrix into TPM format, and conduct subsequent annotation.

```
# Remove the version numbers in Ensembl ID.
eset_stad$Ensembl_ID<-substring(eset_stad$Ensembl_ID, 1, 15)
eset_stad<-column_to_rownames(eset_stad, var = "Ensembl_ID")

# Revert back to original format because the data from UCSC was log2(x+1)transformed.
eset_stad<-(2^eset_stad)+1

eset_stad<-count2tpm(countMat = eset_stad, idType = "Ensembl", org="hsa", source = "loca

eset_stad[1:5,1:5]
```

3.6 Identifying outlier samples

Take ACRG microarray data for example

```
res <- find_outlier_samples(eset = eset, project = "ACRG", show_plot = TRUE)
```




```
## [1] "GSM1523817" "GSM1523858" "GSM1523984" "GSM1523988" "GSM1524030"
```

Removing potential outlier samples

```
eset1 <- eset[, !colnames(eset)%in%res]
```

3.7 PCA analysis of molecular subtypes

```
data("pdata_acrg", package = "IOBR")
res<- iobr_pca(data       = eset1,
```

```
              is.matrix   = TRUE,
              scale       = TRUE,
              is.log      = FALSE,
              pdata       = pdata_acrg,
              id_pdata    = "ID",
              group       = "Subtype",
              geom.ind    = "point",
              cols        = "normal",
              palette     = "jama",
              repel       = FALSE,
              ncp         = 5,
              axes        = c(1, 2),
              addEllipses = TRUE)
```

```
##
##       CIN       EBV       EMT        GS       MSI MSS/TP53- MSS/TP53+
##         0         0        42         0        68       106        79
## [1] ">>== colors for group: "
```

```
res
```

3.8 Batch effect correction

3.8.1 For microarray data

Obtaining another data set from GEO Gastric cancer: GSE57303 using `GEOquery` R package.

```
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo<-getGEO(GEO      = "GSE57303", getGPL  = F, destdir = "./")
eset2    <-eset_geo[[1]]
eset2    <-exprs(eset2)
eset2[1:5,1:5]
```

```
##           GSM1379261 GSM1379262 GSM1379263 GSM1379264 GSM1379265
## 1007_s_at    8.34746    9.67994    8.62643    8.59301    8.63046
## 1053_at      5.07972    4.46377    5.29685    5.78983    4.33359
## 117_at       5.65558    4.48732    4.21615    5.47984    5.20816
## 121_at       5.95123    7.09056    6.19903    5.89872    5.91323
## 1255_g_at    1.66923    1.98758    1.73083    1.56687    1.63332
```

Annotation of genes in the expression matrix and removal of duplicate genes.

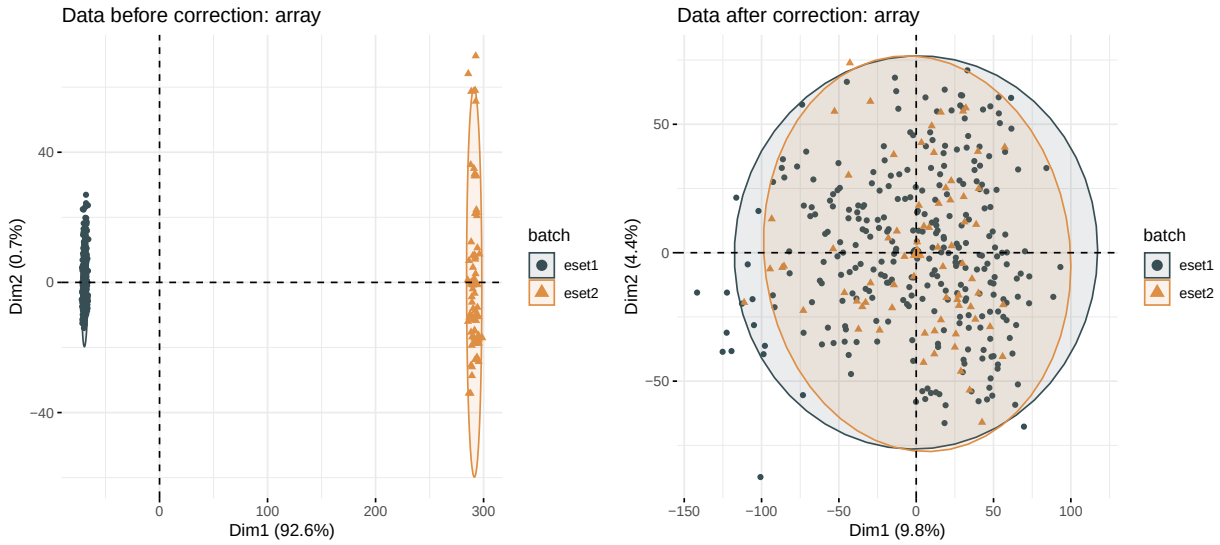
```
eset2<-anno_eset(eset       = eset2,
                 annotation = anno_hug133plus2,
                 symbol     = "symbol",
                 probe      = "probe_id",
                 method     = "mean")
eset2[1:5, 1:5]
```

```
##         GSM1379261 GSM1379262 GSM1379263 GSM1379264 GSM1379265
## ND4        13.1695    13.1804    13.0600    12.4544    13.0457
## ATP6       13.1433    13.0814    13.0502    12.4831    13.1168
## SH3KBP1    12.9390    13.1620    12.9773    12.8745    13.1169
## COX2       13.0184    13.0489    12.8621    12.7489    12.9732
## RPL41      13.0201    12.6034    12.7929    13.0153    12.9404
```

```
eset_com <- remove_batcheffect( eset1       = eset1,
                                eset2       = eset2,
                                eset3       = NULL,
                                id_type     = "symbol",
                                data_type   = "array",
                                cols        = "normal",
                                palette     = "jama",
                                log2        = TRUE,
                                check_eset  = TRUE,
                                adjust_eset = TRUE,
                                repel       = FALSE,
                                path        = "result")
```

```
##
## eset1 eset2
##   295    70
## [1] ">>== colors for group: "
```

```
##
## eset1 eset2
##   295    70
## [1] ">>== colors for group: "
```




```
dim(eset_com)
```

```
## [1] 21752   365
```

3.8.2 For RNAseq count data

```
data("eset_stad", package = "IOBR")
head(eset_stad)
```

```
##                 TCGA-BR-6455 TCGA-BR-7196 TCGA-BR-8371 TCGA-BR-8380
## ENSG00000000003         8006         2114          767         1556
## ENSG00000000005            1            0            5            5
## ENSG00000000419         3831         2600         1729         1760
## ENSG00000000457         1126          745         1040         1260
## ENSG00000000460          857          463          231          432
## ENSG00000000938          758         1126          557          557
##                 TCGA-BR-8592 TCGA-BR-8686 TCGA-BR-A4IV TCGA-BR-A4J4
## ENSG00000000003         2806         2923         1524         7208
## ENSG00000000005           60            1           22            2
## ENSG00000000419         2273         1934         2838         4418
```

```
## ENSG00000000457         1814          707         1683         1335
## ENSG00000000460          635          323          270          423
## ENSG00000000938          828          666          760          597
##                 TCGA-BR-A4J9 TCGA-FP-7916
## ENSG00000000003          711         2747
## ENSG00000000005            0            3
## ENSG00000000419         2426         2824
## ENSG00000000457         1590         1672
## ENSG00000000460          276          773
## ENSG00000000938          370          688
```

```
data("eset_blca", package = "IOBR")
head(eset_blca)
```

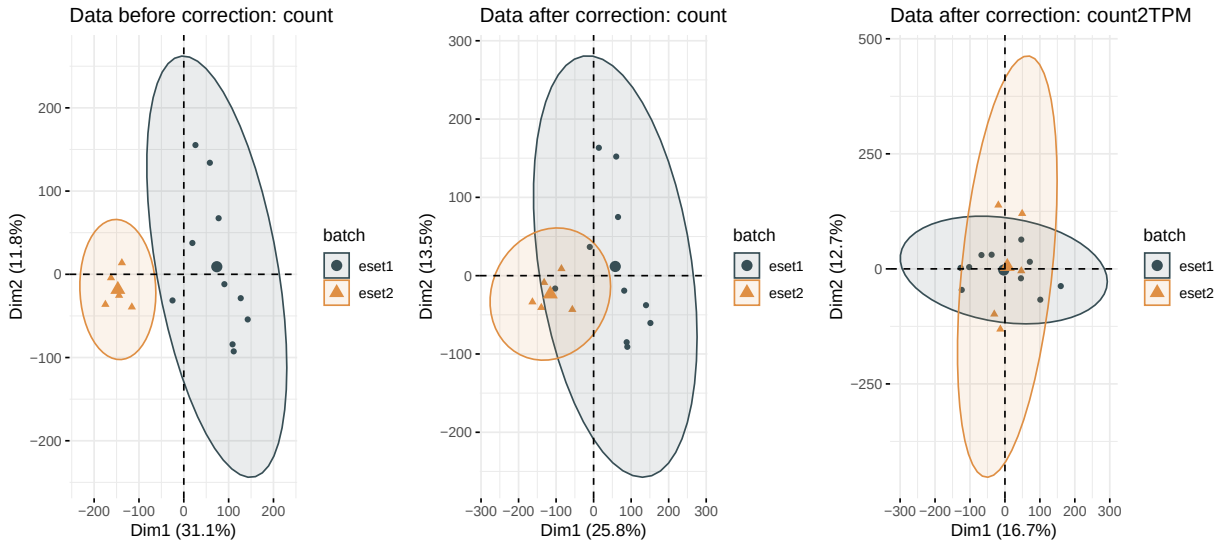
```
##                 TCGA-2F-A9KO TCGA-2F-A9KP TCGA-2F-A9KQ TCGA-2F-A9KR
## ENSG00000000003         6092        11652         5426         4383
## ENSG00000000005            0            4            1            1
## ENSG00000000419         3072         2656         1983         2061
## ENSG00000000457         1302          984         1134         1092
## ENSG00000000460          779          924          421          386
## ENSG00000000938          436          116          312          590
##                 TCGA-2F-A9KT
## ENSG00000000003         3334
## ENSG00000000005            0
## ENSG00000000419         2930
## ENSG00000000457          496
## ENSG00000000460          318
## ENSG00000000938          362
```

```
eset_com <- remove_batcheffect(eset_stad, eset_blca, id_type = "ensembl", data_type = "c
```

```
## Found 2 batches
## Using null model in ComBat-seq.
## Adjusting for 0 covariate(s) or covariate level(s)
## Estimating dispersions
## Fitting the GLM model
## Shrinkage off - using GLM estimates for parameters
## Adjusting the data
```

```
## Warning in count2tpm(countMat = combined.expr.combat, idType = id_type, :
```

```
## >>>--- Omit 1263 genes of which length is not available !

##
## eset1 eset2
##    10     5
## [1] ">>== colors for group: "
##
## eset1 eset2
##    10     5
## [1] ">>== colors for group: "
##
## eset1 eset2
##    10     5
## [1] ">>== colors for group: "
```




```
# The returned matrix is the count matrix after removing the batches.
head(eset_com)
```

```
##                 TCGA-BR-6455 TCGA-BR-7196 TCGA-BR-8371 TCGA-BR-8380
## ENSG00000000003        10264         3536         1710         2964
## ENSG00000000005            1            0            4            5
## ENSG00000000419         4500         3099         2111         2167
## ENSG00000000457         1203          707         1106         1353
## ENSG00000000460         1059          590          310          560
## ENSG00000000938          731         1202          507          485
```

```
##                 TCGA-BR-8592 TCGA-BR-8686 TCGA-BR-A4IV TCGA-BR-A4J4
## ENSG00000000003         4761         3964         3115         9565
## ENSG00000000005           33            1           14            3
## ENSG00000000419         2782         2270         3444         5176
## ENSG00000000457         2089          817         1845         1469
## ENSG00000000460          810          405          368          548
## ENSG00000000938          769          723          677          532
##                 TCGA-BR-A4J9 TCGA-FP-7916 TCGA-2F-A9KO TCGA-2F-A9KP
## ENSG00000000003         1739         4371         2812         6796
## ENSG00000000005            0            3            0           10
## ENSG00000000419         2943         3362         2189         1849
## ENSG00000000457         1804         2044          994          817
## ENSG00000000460          371          959          495          584
## ENSG00000000938          281          654          456          156
##                 TCGA-2F-A9KQ TCGA-2F-A9KR TCGA-2F-A9KT
## ENSG00000000003         1971         1429         1057
## ENSG00000000005            1            1            0
## ENSG00000000419         1355         1420         2094
## ENSG00000000457          916          876          438
## ENSG00000000460          251          230          190
## ENSG00000000938          353          604          383
```

3.9 References

Wang et al., (2019). The UCSCXenaTools R package: a toolkit for accessing genomics data from UCSC Xena platform, from cancer multi-omics to single-cell RNA-seq. Journal of Open Source Software, 4(40), 1627, https://doi.org/10.21105/joss.01627

Zhang et al., ComBat-seq: batch effect adjustment for RNA-seq count data, NAR Genomics and Bioinformatics, Volume 2, Issue 3, September 2020, lqaa078, https://doi.org/10.1093/nargab/lqaa078

Leek, J. T., et al., (2012). The sva package for removing batch effects and other unwanted variation in high-throughput experiments. Bioinformatics, 28(6), 882-883.

Chapter 4

Signature Score Calculation

4.1 Loading packages

Load the IOBR package in your R session after the installation is complete:

```
library(IOBR)
library(survminer)
library(tidyverse)
```

4.2 Downloading data for example

Obtaining data set from GEO Gastric cancer: GSE62254 using `GEOquery` R package.

```
if (!requireNamespace("GEOquery", quietly = TRUE))  BiocManager::install("GEOquery")
library("GEOquery")
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo <- getGEO(GEO = "GSE62254", getGPL  = F, destdir = "./")
eset    <-eset_geo[[1]]
eset    <-exprs(eset)
eset[1:5,1:5]
```

```
##           GSM1523727 GSM1523728 GSM1523729 GSM1523744 GSM1523745
## 1007_s_at  3.2176645  3.0624323  3.0279131   2.921683  2.8456013
## 1053_at    2.4050109  2.4394879  2.2442708   2.345916  2.4328582
## 117_at     1.4933412  1.8067380  1.5959665   1.839822  1.8326058
## 121_at     2.1965561  2.2812181  2.1865556   2.258599  2.1874363
```

```
## 1255_g_at  0.8698382  0.9502466  0.8125414   1.012860  0.9441993
```

Annotation of genes in the expression matrix and removal of duplicate genes.

```
# Load the annotation file `anno_hug133plus2` in IOBR.
head(anno_hug133plus2)
```

```
## # A tibble: 6 x 2
##   probe_id  symbol
##   <fct>     <fct>
## 1 1007_s_at MIR4640
## 2 1053_at   RFC2
## 3 117_at    HSPA6
## 4 121_at    PAX8
## 5 1255_g_at GUCA1A
## 6 1294_at   MIR5193
```

```
# Conduct gene annotation using `anno_hug133plus2` file; If identical gene symbols exi

eset<-anno_eset(eset       = eset,
                annotation = anno_hug133plus2,
                symbol     = "symbol",
                probe      = "probe_id",
                method     = "mean")
eset[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
## LOC101928826   4.219303   4.219670   4.213252
```

4.3 Signature score estimation

4.3.1 Signature collection of IOBR

```
# Return available parameter options of signature estimation.
signature_score_calculation_methods
```

```
##           PCA        ssGSEA       z-score   Integration
##         "pca"      "ssgsea"      "zscore" "integration"
```

```
#TME associated signatures
names(signature_tme)[1:20]
```

```
##  [1] "CD_8_T_effector"            "DDR"
##  [3] "APM"                        "Immune_Checkpoint"
##  [5] "CellCycle_Reg"              "Pan_F_TBRs"
##  [7] "Histones"                   "EMT1"
##  [9] "EMT2"                       "EMT3"
## [11] "WNT_target"                 "FGFR3_related"
## [13] "Cell_cycle"                 "Mismatch_Repair"
## [15] "Homologous_recombination"   "Nucleotide_excision_repair"
## [17] "DNA_replication"            "Base_excision_repair"
## [19] "TMEscoreA_CIR"              "TMEscoreB_CIR"
```

```
#Metabolism related signatures
names(signature_metabolism)[1:20]
```

```
##  [1] "Cardiolipin_Metabolism"
##  [2] "Cardiolipin_Biosynthesis"
##  [3] "Cholesterol_Biosynthesis"
##  [4] "Citric_Acid_Cycle"
##  [5] "Cyclooxygenase_Arachidonic_Acid_Metabolism"
##  [6] "Prostaglandin_Biosynthesis"
##  [7] "Purine_Biosynthesis"
##  [8] "Pyrimidine_Biosynthesis"
##  [9] "Dopamine_Biosynthesis"
## [10] "Epinephrine_Biosynthesis"
## [11] "Norepinephrine_Biosynthesis"
## [12] "Fatty_Acid_Degradation"
## [13] "Fatty_Acid_Elongation"
## [14] "Fatty_Acid_Biosynthesis"
## [15] "Folate_One_Carbon_Metabolism"
## [16] "Folate_biosynthesis"
## [17] "Gluconeogenesis"
## [18] "Glycolysis"
## [19] "Glycogen_Biosynthesis"
```

```
## [20] "Glycogen_Degradation"
```

Signatures associated with basic biomedical research, such as m6A, TLS, ferroptosis and exosomes.

```
names(signature_tumor)
```

```
##  [1] "Nature_metabolism_Hypoxia"
##  [2] "Winter_hypoxia_signature"
##  [3] "Hu_hypoxia_signature"
##  [4] "Molecular_Cancer_m6A"
##  [5] "MT_exosome"
##  [6] "SR_exosome"
##  [7] "Positive_regulation_of_exosomal_secretion"
##  [8] "Negative_regulation_of_exosomal_secretion"
##  [9] "Exosomal_secretion"
## [10] "Exosome_assembly"
## [11] "Extracellular_vesicle_biogenesis"
## [12] "MC_Review_Exosome1"
## [13] "MC_Review_Exosome2"
## [14] "CMLS_Review_Exosome"
## [15] "Ferroptosis"
## [16] "EV_Cell_2020"
```

`signature_collection` including all aforementioned signatures

```
names(signature_collection)[1:20]
```

```
##  [1] "CD_8_T_effector"            "DDR"
##  [3] "APM"                        "Immune_Checkpoint"
##  [5] "CellCycle_Reg"              "Pan_F_TBRs"
##  [7] "Histones"                   "EMT1"
##  [9] "EMT2"                       "EMT3"
## [11] "WNT_target"                 "FGFR3_related"
## [13] "Cell_cycle"                 "Mismatch_Repair"
## [15] "Homologous_recombination"   "Nucleotide_excision_repair"
## [17] "DNA_replication"            "Base_excision_repair"
## [19] "TMEscoreA_CIR"              "TMEscoreB_CIR"
```

```
#citation of signatures
signature_collection_citation[1:20, ]
```

```
## # A tibble: 20 x 6
##    Signatures                `Published year` Journal         Title PMID  DOI
##    <chr>                                <dbl> <chr>           <chr> <chr> <chr>
##  1 CD_8_T_effector                       2018 Nature          TGF ~ 2944~ 10.1~
##  2 DDR                                   2018 Nature          TGF ~ 2944~ 10.1~
##  3 APM                                   2018 Nature          TGF ~ 2944~ 10.1~
##  4 Immune_Checkpoint                     2018 Nature          TGF ~ 2944~ 10.1~
##  5 CellCycle_Reg                         2018 Nature          TGF ~ 2944~ 10.1~
##  6 Pan_F_TBRs                            2018 Nature          TGF ~ 2944~ 10.1~
##  7 Histones                              2018 Nature          TGF ~ 2944~ 10.1~
##  8 EMT1                                  2018 Nature          TGF ~ 2944~ 10.1~
##  9 EMT2                                  2018 Nature          TGF ~ 2944~ 10.1~
## 10 EMT3                                  2018 Nature          TGF ~ 2944~ 10.1~
## 11 WNT_target                            2018 Nature          TGF ~ 2944~ 10.1~
## 12 FGFR3_related                         2018 Nature          TGF ~ 2944~ 10.1~
## 13 Cell_cycle                            2018 Nature          TGF ~ 2944~ 10.1~
## 14 Mismatch_Repair                       2018 Nature          TGF ~ 2944~ 10.1~
## 15 Homologous_recombination              2018 Nature          TGF ~ 2944~ 10.1~
## 16 Nucleotide_excision_repair            2018 Nature          TGF ~ 2944~ 10.1~
## 17 DNA_replication                       2018 Nature          TGF ~ 2944~ 10.1~
## 18 Base_excision_repair                  2018 Nature          TGF ~ 2944~ 10.1~
## 19 TMEscoreA_CIR                         2019 Cancer Immunol~ Tumo~ 3084~ 10.1~
## 20 TMEscoreB_CIR                         2019 Cancer Immunol~ Tumo~ 3084~ 10.1~
```

The evaluation of signature scores involved three methodologies: Single-sample Gene Set Enrichment Analysis (ssGSEA), Principal Component Analysis (PCA), and Z-score.

4.4 Estimation of signature using PCA method

The PCA method is ideal for gene sets with co-expression. Heatmaps and correlation matrices can be used to determine if co-expression is present in the applicable gene set.

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = signature_collection,
                             method          = "pca",
                             mini_gene_count = 2)
```

```
sig_tme <- t(column_to_rownames(sig_tme, var = "ID"))
sig_tme[1:5, 1:3]
```

```
##                    GSM1523727 GSM1523728 GSM1523729
## CD_8_T_effector    -2.5513794  0.7789141 -2.1770675
## DDR                -0.8747614  0.7425162 -1.3272054
## APM                 1.1098368  2.1988688 -0.9516419
## Immune_Checkpoint -2.3701787  0.9455120 -1.4844104
## CellCycle_Reg      0.1063358  0.7583302 -0.3649795
```

4.5 Estimated using the ssGSEA methodology

This method is appropriate for gene sets that contain a large number of genes (> 30 genes), such as those of GO, KEGG, REACTOME gene sets.

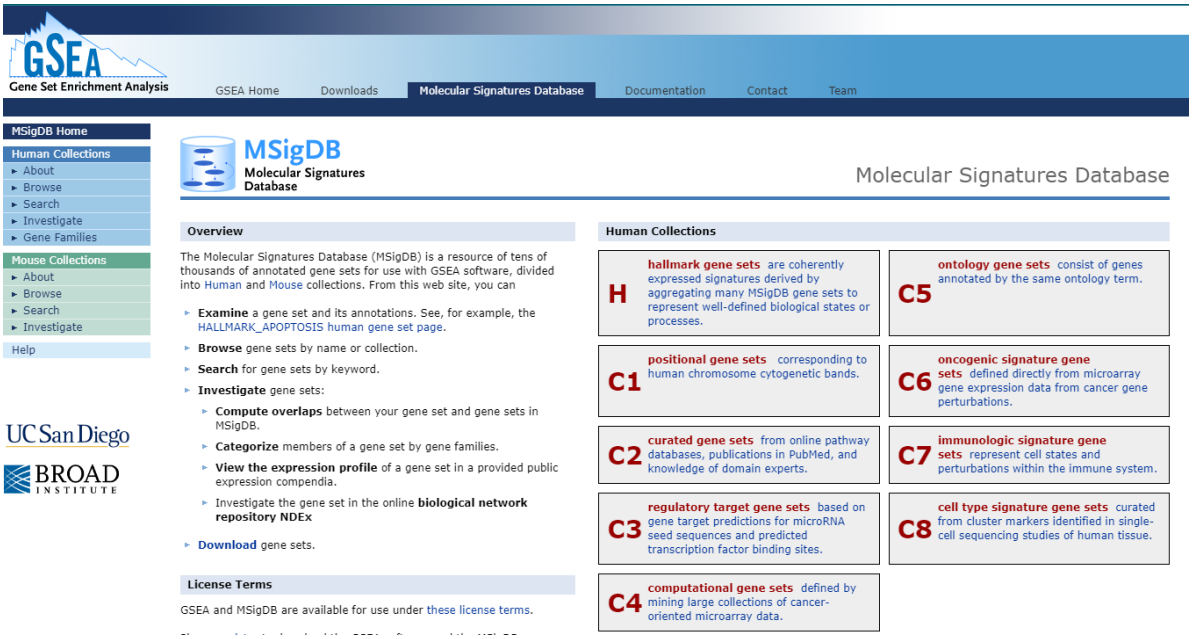


Figure 4.1: Gene sets of MSigDb

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = go_bp,
                             method          = "ssgsea",
                             mini_gene_count = 2)
```

4.6 Calculated using the z-score function.

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = signature_collection,
                             method          = "zscore",
                             mini_gene_count = 2)
```

4.7 Calculated using all three methods at the same time

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = signature_collection,
                             method          = "integration",
                             mini_gene_count = 2)
```

The same signature in this case will be scored using all three methods simultaneously.

```
colnames(sig_tme)[grep(colnames(sig_tme), pattern = "CD_8_T_effector")]
```

The `select_method()` function allows the user to extract data using various methods.

```
sig_tme_pca <- select_method(data = sig_tme, method = "pca")
colnames(sig_tme_pca)[grep(colnames(sig_tme_pca), pattern = "CD_8_T_effector")]
```

4.8 Classification of signatures

As more signatures related to the tumour microenvironment were collected in IOBR, and we may continue to add gene signatures related to the tumour microenvironment in the future, we have made a basic classification of these signatures by combining them with our analysis experience. Users can compare the signatures in the same group during the analysis process to improve the reliability and consistency of the conclusions.

```
sig_group[1:8]
```

4.9 How to customise the signature gene list for `calculate_signature_score`

4.9.1 Method-1: Use excel for storage and construction

Users can collect gene signatures using either an `Excel` or `CSV` file. The format should have the name of the signature in the first row, followed by the genes contained in each signature from the second row onwards. Once imported, the function `format_signature` can be used to transform the data into a gene list of signatures required for `calculate_signature_score`. To import the file into R, users can use the functions read.csv or read_excel. It is important to note here that the user needs to use the longest signature as a criterion and then replace all the vacant grids using NA, otherwise an error may be reported when reading into R.

Here we provide a sample data `sig_excel`, please refer to this format to construct the required csv or excel files.

```
data("sig_excel", package = "IOBR")
sig <- format_signatures(sig_excel)
print(sig[1:5])
```

```
## $Tcell_co_inhibitors
##  [1] "ADORA2A"  "BTLA"     "BTN2A2"   "BTN3A1"   "BTN3A2"   "BTNL2"
##  [7] "C10orf54" "CSF1R"    "HAVCR2"   "IDO1"     "IL10"     "IL10RB"
## [13] "KDR"      "KIR2DL1"  "SLAMF7"   "TGFB1"    "TIGIT"    "VRCN1"
## [19] "VTCN1"    "CD247"    "CTLA4"    "CD160"    "CD244"    "CD274"
## [25] "CD276"    "CD48"     "CD96"     "KIR2DL2"  "KIR2DL3"  "LAG3"
## [31] "LAIR1"    "LGALS9"   "PVRL2"    "PDCD1"    "PDCD1LG2"
##
## $Tcell_co_stimuiations
##  [1] "BTNL8"     "CD226"     "CD27"      "CD28"      "CD40"      "CD58"
##  [7] "CD70"      "SLAMF1"    "TMIGD2"    "TNFRSF13B" "TNFRSF13C" "TNFRSF14"
## [13] "TNFRSF4"   "TNFRSF8"   "TNFSF8"    "TNFSF9"    "ENTPD1"    "NT5E"
## [19] "ICOS"      "TNFSF4"    "TNFSF15"   "CD80"      "CD86"      "EGFR"
## [25] "HAVCR1"    "TNFSF18"   "ICOSLG"    "TNFSF13B"  "TNFRSF9"   "TNFSF13"
##
## $Tcell_function
```

```
## [1] "CD3E"  "CD4"   "CD8B"  "FOXP3" "GZMB"  "PRF1"  "TBX21" "IL2RA" "IKZF2"
##
## $Tcell_checkpoint
##  [1] "CD274"    "CTLA4"    "LAG3"     "TIM3"     "TNFRSF9"  "TIGIT"
##  [7] "CD226"    "CD7"      "GZMB"     "PRF1"     "TNFRSF18" "TNFRSF4"
## [13] "HAVCR2"   "NLG1"     "CD4"      "CD8A"     "CD8B"     "FOXP3"
## [19] "IL2"      "CXCL8"    "PDCD1"    "IFNG"
##
## $Teffctore_score
## [1] "CD8A"   "CXCL10" "CXCL9"  "GZMA"   "GZMB"   "IFNG"   "PRF1"   "TBX21"
```

For simple structures or when the number of signatures to be added is relatively small, the following two methods can also be used.

4.9.2 Method-2: Build the list structure directly

```
sig <- list("CD8" = c("CD8A",  "CXCL10", "CXCL9",  "GZMA",   "GZMB",   "IFNG",   "PRF1",
            "ICB" = c("CD274",   "PDCD1LG2", "CTLA4",    "PDCD1",    "LAG3",     "HAVCR2
sig
```

```
## $CD8
## [1] "CD8A"   "CXCL10" "CXCL9"  "GZMA"   "GZMB"   "IFNG"   "PRF1"   "TBX21"
##
## $ICB
## [1] "CD274"    "PDCD1LG2" "CTLA4"    "PDCD1"    "LAG3"     "HAVCR2"   "TIGIT"
```

4.9.3 Method3: Add the new signature to the existing gene list

```
sig<- signature_tumor
sig$CD8 <- c("CD8A",  "CXCL10", "CXCL9",  "GZMA",   "GZMB",   "IFNG",   "PRF1",   "TBX21
sig
```

```
## $Nature_metabolism_Hypoxia
##  [1] "ACOT7"  "SLC2A1" "ALDOA"  "CDKN3"  "ENO1"   "LDHA"   "MIF"    "MRPS17"
##  [9] "NDRG1"  "P4HA1"  "PGAM1"  "TPI1"   "TUBB6"  "VEGFA"  "ADM"
##
## $Winter_hypoxia_signature
## [1] "VEGF"  "GLUT1" "PDK-1" "ENO1"  "HK2"   "CA9"   "AK3"   "CCNG2" "PFKB3"
##
```

```
## $Hu_hypoxia_signature
##  [1] "FABP5"     "UCHL1"     "GAL"       "PLODDDIT4" "VEGF"      "ADM"
##  [7] "ANGPTL4"   "NDRG1"     "NP"        "SLC16A3"   "C14ORF58"  "RRAGD"
##
## $Molecular_Cancer_m6A
##  [1] "METTL3"    "METTL14"   "RBM15"     "RBM15B"    "WTAP"      "KIAA1429"
##  [7] "CBLL1"     "ZC3H13"    "ALKBH5"    "FTO"       "YTHDC1"    "YTHDC2"
## [13] "YTHDF1"    "YTHDF2"    "YTHDF3"    "IGF2BP1"   "HNRNPA2B1" "HNRNPC"
## [19] "FMR1"      "LRPPRC"    "ELAVL1"
##
## $MT_exosome
##  [1] "YWHAG"  "YWHAQ"  "CLTC"   "NCKAP1" "CFL1"   "ACTB"   "CCT4"   "RDX"
##  [9] "GNA13"  "CTNNB1"
##
## $SR_exosome
## [1] "HSP70" "HSP90" "CD9"   "CD63"  "CD81"  "CD82"
##
## $Positive_regulation_of_exosomal_secretion
##  [1] "ATP13A2" "CHMP2A"  "HGS"     "MYO5B"   "PDCD6IP" "RAB7"    "SDC1"
##  [8] "SDC4"    "SDCBP"   "SMPD3"   "SNF8"    "STAM"    "TSG101"  "VPS4A"
##
## $Negative_regulation_of_exosomal_secretion
## [1] "VPS4B" "PRKN"  "RAB7"
##
## $Exosomal_secretion
## [1] "STEAP3" "TSG101" "RAB11A" "RAB27A" "COPS5"
##
## $Exosome_assembly
## [1] "CD34"    "PDCD6IP" "SDC1"    "SDC4"    "SDCBP"   "STAM"    "TSG101"
##
## $Extracellular_vesicle_biogenesis
##  [1] "ARRDC1"  "ARRDC4"  "ATP13A2" "CD34"    "CHMP2A"  "COPS5"   "HGS"
##  [8] "MYO5B"   "PDCD6IP" "PRKN"    "RAB7"    "RAB11A"  "RAB27A"  "SDC1"
## [15] "SDC4"    "SDCBP"   "SMPD3"   "SNF8"    "STAM"    "STEAP3"  "TSG101"
## [22] "VPS4B"
##
## $MC_Review_Exosome1
```

```
##  [1] "TSG101"  "CD9"     "CD81"    "CD63"    "FLOT1"   "ITGB1"   "ITGA1"
##  [8] "HSP70"   "AIP1"    "ALIX"    "PDCD6IP"
##
## $MC_Review_Exosome2
##  [1] "RAB27A"  "RAB27B"  "PIKFYVE" "HRS"     "SYT7"    "CTTN"    "STAT3"
##  [8] "PKM2"    "UNC13D"  "miR-155" "EGFR"    "RAS"     "EIF3C"   "LKB1"
## [15] "STK11"
##
## $CMLS_Review_Exosome
##  [1] "HRS"      "STAM1"    "TSG101"   "CHMP4C"   "ALIX"     "VAT1"
##  [7] "VPS4"     "CD9"      "CD82"     "CD63"     "LMP1"     "TSPAN8"
## [13] "VAMP7"    "YKT6"     "PKM2"     "SNAP-23"  "RALA"     "RALB"
## [19] "RAB2B"    "RAB5A"    "RAB9A"    "RAB7"     "RAB11"    "RAB27A"
## [25] "RAB27B"   "RAB35"    "DGKA"     "PLD2"     "ARF6"     "ATG12"
## [31] "ATG7"     "PIKFYVE"  "BST2"     "ATP6V0A4"
##
## $Ferroptosis
##  [1] "ACSL4"      "AKR1C1-3"   "ALOXs"      "ATP5G3"     "CARS"
##  [6] "CBS"        "CD44v"      "CHAC1"      "CISD1"      "CS"
## [11] "DPP4"       "FANCD2"     "GCLC/GCLM"  "GLS2"       "GPX4"
## [16] "GSS"        "HMGCR"      "HSPB1/5"    "KOD"        "LPCAT3"
## [21] "MT1G"       "NCOA4"      "NFE2L2"     "PTGS2"      "RPL8"
## [26] "SAT1"       "SLC7A11"    "SQS"        "TFRC"       "TP53"
## [31] "TTC35/EMC2" "MESH1"
##
## $EV_Cell_2020
##  [1] "HSP90AB1" "HSP90AA1" "CD9"      "ALIX"     "FLOT1"    "FLOT2"
##  [7] "TSG101"   "HSPA8"    "CD81"     "CD63"     "HBB"      "JCHAIN"
## [13] "A2M"      "B2M"      "FN1"      "RAP1B"    "LGALS3BP" "GSN"
## [19] "MSN"      "FLNA"     "ACTB"     "STOM"     "PRDX2"
##
## $CD8
## [1] "CD8A"   "CXCL10" "CXCL9"  "GZMA"   "GZMB"   "IFNG"   "PRF1"   "TBX21"
```

4.10 How to export gene signature

Using the `output_sig` function, user can export the signatures of the list structure to a csv file for other purposes. This step is exactly the reverse of `format_signatures`.

```
sig <- output_sig(signatures = signature_sc, format = "csv", file.name = "sc_signature")
sig[1:8, 1:5]
```

```
##   CD4_c0_Tcm CD4_c1_Treg CD4_c10_Tn_LEF1_ANKRD55 CD4_c11_Tisg CD4_c2_Tn
## 1      ANXA1       FOXP3                ANKRD55        ISG15    NBEAL1
## 2       LMNA       IL2RA                   LEF1         IFI6      CCR7
## 3        VIM     TNFRSF4                   TCF7       IFI44L   GLTSCR2
## 4      KLRB1       TIGIT                  NOSIP          MX1      TCF7
## 5       IL7R      CARD16                   SELL        IFIT3    GNB2L1
## 6      ZFP36    TNFRSF18                  IL6ST        IFIT1      SELL
## 7    ZFP36L2        BATF                LDLRAP1        RSAD2   C6orf48
## 8     GPR183       CTLA4                 RIPOR2        STAT1    TMEM66
```

4.11 References

ssgsea: Barbie, D.A. et al (2009). Systematic RNA interference reveals that oncogenic KRAS-driven cancers require TBK1. Nature, 462(5):108-112.

gsva: Hänzelmann, S., Castelo, R. and Guinney, J. (2013). GSVA: Gene set variation analysis for microarray and RNA-Seq data. BMC Bioinformatics, 14(1):7.

zscore: Lee, E. et al (2008). Inferring pathway activity toward precise disease classification. PLoS Comp Biol, 4(11):e1000217.

PCA method: Mariathasan S, Turley SJ, Nickles D, et al. TGF attenuates tumour response to PD-L1 blockade by contributing to exclusion of T cells. Nature. 2018 Feb 22;554(7693):544-548.

MSigDB:Dolgalev I (2022). msigdbr: MSigDB Gene Sets for Multiple Organisms in a Tidy Data Format. R package version 7.5.1. (https://www.gsea-msigdb.org/gsea/msigdb/)

Chapter 5

TME deconvolution

This section demonstrates various algorithms for parsing the tumour microenvironment using data from the bulk transcriptome. We also describe how to construct the reference signature matrix for the popular SVR algorithm (CIBERSORT) from single-cell data.

5.1 Loading packages

Load the IOBR package in your R session after the installation is complete:

```
library(IOBR)
library(survminer)
library(tidyverse)
```

5.2 Downloading data for example

Obtaining data set from GEO Gastric cancer: GSE62254 using `GEOquery` R package.

```
if (!requireNamespace("GEOquery", quietly = TRUE))  BiocManager::install("GEOquery")
library("GEOquery")
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo <- getGEO(GEO = "GSE62254", getGPL  = F, destdir = "./")
eset    <- eset_geo[[1]]
eset    <- exprs(eset)
eset[1:5, 1:5]
```

```
##           GSM1523727 GSM1523728 GSM1523729 GSM1523744 GSM1523745
```

```
## 1007_s_at  3.2176645  3.0624323  3.0279131   2.921683  2.8456013
## 1053_at    2.4050109  2.4394879  2.2442708   2.345916  2.4328582
## 117_at     1.4933412  1.8067380  1.5959665   1.839822  1.8326058
## 121_at     2.1965561  2.2812181  2.1865556   2.258599  2.1874363
## 1255_g_at  0.8698382  0.9502466  0.8125414   1.012860  0.9441993
```

Annotation of genes in the expression matrix and removal of duplicate genes.

```
library(IOBR)

# Load the annotation file `anno_hug133plus2` in IOBR.
head(anno_hug133plus2)
```

```
## # A tibble: 6 x 2
##   probe_id  symbol
##   <fct>     <fct>
## 1 1007_s_at MIR4640
## 2 1053_at   RFC2
## 3 117_at    HSPA6
## 4 121_at    PAX8
## 5 1255_g_at GUCA1A
## 6 1294_at   MIR5193
```

```
# Conduct gene annotation using `anno_hug133plus2` file; If identical gene symbols exi

eset<-anno_eset(eset       = eset,
                annotation = anno_hug133plus2,
                symbol     = "symbol",
                probe      = "probe_id",
                method     = "mean")
eset[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
## LOC101928826   4.219303   4.219670   4.213252
```

5.3 Available Methods to Decode TME Contexture

```
tme_deconvolution_methods
```

```
##         MCPcounter               EPIC             xCell          CIBERSORT
##       "mcpcounter"             "epic"            "xcell"        "cibersort"
## CIBERSORT Absolute                IPS          ESTIMATE                SVR
##    "cibersort_abs"              "ips"         "estimate"              "svr"
##               lsei              TIMER          quanTIseq
##             "lsei"            "timer"        "quantiseq"
```

```
# Return available parameter options of deconvolution methods
```

The input data is a matrix subseted from ESET of ACRG cohort, with genes in rows and samples in columns. The row name must be HGNC symbols and the column name must be sample names.

```
eset_acrg <- eset[, 1:50]
eset_acrg[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
## LOC101928826   4.219303   4.219670   4.213252
```

Check detail parameters of the function

```
# help(deconvo_tme)
```

5.4 Method 1: CIBERSORT

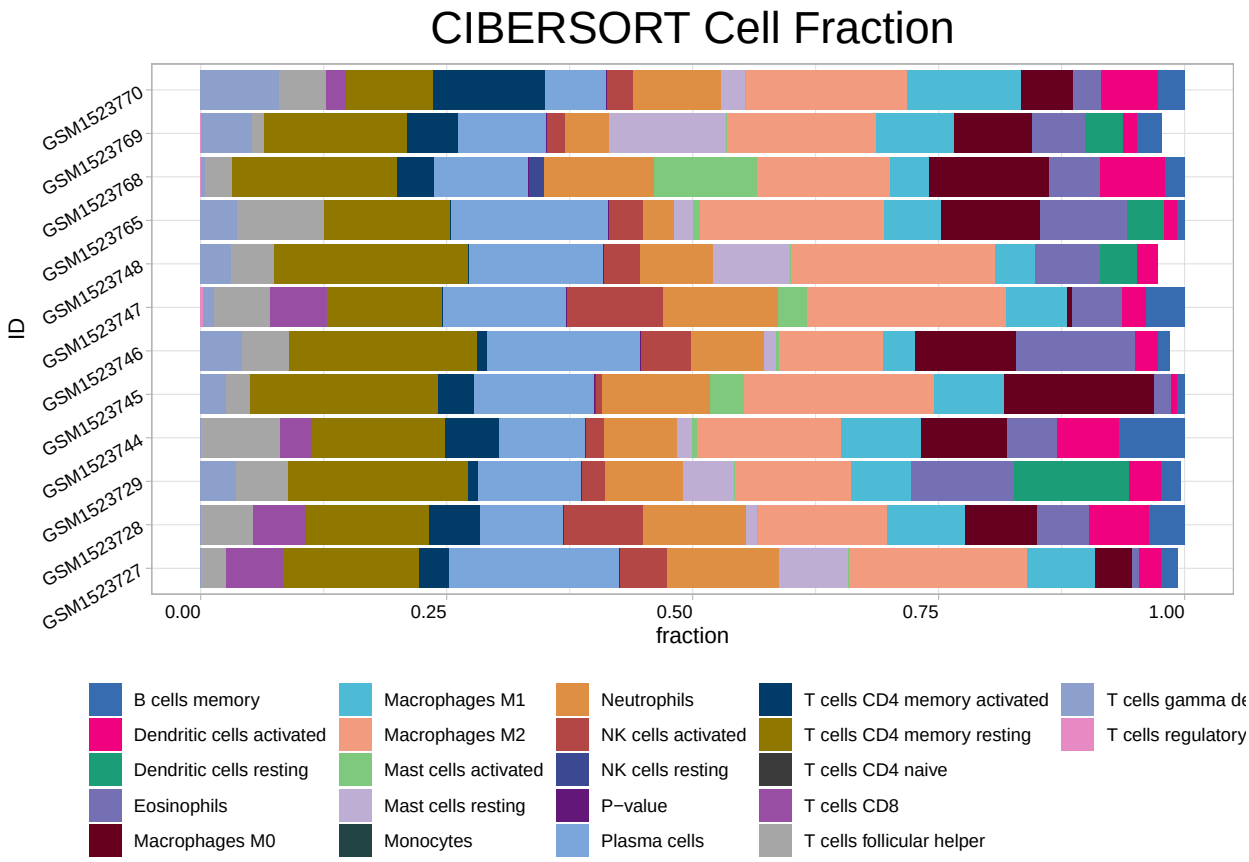
```
cibersort<-deconvo_tme(eset = eset_acrg, method = "cibersort", arrays = TRUE, perm = 100
```

```
##
## >>> Running CIBERSORT
```

```
# head(cibersort)
res<-cell_bar_plot(input = cibersort[1:12,], features = colnames(cibersort)[3:24], title
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## >>>>=== Palette option for random: 1: palette1; 2: palette2; 3: palette3;  4: palette
```



5.5 Method 2: EPIC

```
# help(deconvo_epic)
epic<-deconvo_tme(eset = eset_acrg, method = "epic", arrays = TRUE)
```

```
##
## >>> Running EPIC
```

```
## Warning in IOBR::EPIC(bulk = eset, reference = ref, mRNA_cell = NULL, scaleExprs = TR
## GSM1523744; GSM1523746; GSM1523781; GSM1523786
##  - check fit.gof for the convergeCode and convergeMessage
```

```
## Warning in IOBR::EPIC(bulk = eset, reference = ref, mRNA_cell = NULL,
## scaleExprs = TRUE): mRNA_cell value unknown for some cell types: CAFs,
## Endothelial - using the default value of 0.4 for these but this might bias the
```

```
## true cell proportions from all cell types.
```

```
head(epic)
```

```
## # A tibble: 6 x 9
##   ID      Bcells_EPIC CAFs_EPIC CD4_Tcells_EPIC CD8_Tcells_EPIC Endothelial_EPIC
##   <chr>         <dbl>     <dbl>           <dbl>           <dbl>            <dbl>
## 1 GSM152~      0.0292   0.00888           0.145          0.0756           0.0876
## 2 GSM152~      0.0293   0.0109            0.159          0.0745           0.0954
## 3 GSM152~      0.0308   0.0106            0.149          0.0732           0.0941
## 4 GSM152~      0.0273   0.0108            0.145          0.0704           0.0860
## 5 GSM152~      0.0280   0.0111            0.151          0.0707           0.0928
## 6 GSM152~      0.0320   0.00958           0.148          0.0716           0.0907
## # i 3 more variables: Macrophages_EPIC <dbl>, NKcells_EPIC <dbl>,
## #   otherCells_EPIC <dbl>
```

5.6 Method 3: MCPcounter

```
mcp<-deconvo_tme(eset = eset_acrg, method = "mcpcounter")
```

```
##
## >>> Running MCP-counter
```

```
head(mcp)
```

```
## # A tibble: 6 x 11
##   ID         T_cells_MCPcounter CD8_T_cells_MCPcounter Cytotoxic_lymphocytes_M~1
##   <chr>                   <dbl>                  <dbl>                      <dbl>
## 1 GSM1523727               1.47                   1.11                       1.33
## 2 GSM1523728               1.53                   1.05                       1.60
## 3 GSM1523729               1.47                   1.07                       1.37
## 4 GSM1523744               1.46                   1.02                       1.44
## 5 GSM1523745               1.51                   1.10                       1.49
## 6 GSM1523746               1.51                   0.992                      1.40
## # i abbreviated name: 1: Cytotoxic_lymphocytes_MCPcounter
## # i 7 more variables: B_lineage_MCPcounter <dbl>, NK_cells_MCPcounter <dbl>,
## #   Monocytic_lineage_MCPcounter <dbl>,
## #   Myeloid_dendritic_cells_MCPcounter <dbl>, Neutrophils_MCPcounter <dbl>,
## #   Endothelial_cells_MCPcounter <dbl>, Fibroblasts_MCPcounter <dbl>
```

5.7 Method 4: xCELL

```
xcell<-deconvo_tme(eset = eset_acrg, method = "xcell", arrays = TRUE)
```

```
head(xcell)
```

```
## # A tibble: 6 x 68
##   ID         aDC_xCell Adipocytes_xCell Astrocytes_xCell `B-cells_xCell`
##   <chr>          <dbl>            <dbl>            <dbl>           <dbl>
## 1 GSM1523727  4.78e-19          0.0250          0                0
## 2 GSM1523728  9.41e- 2          0.00433         7.70e- 3         0
## 3 GSM1523729  1.02e- 1          0.0789          2.04e- 2         0
## 4 GSM1523744  7.88e- 2          0.0538          4.82e-18         0.0126
## 5 GSM1523745  9.02e- 2          0.0136          1.93e- 2         0
## 6 GSM1523746  3.40e- 2          0.0331          9.22e- 2         0
## # i 63 more variables: Basophils_xCell <dbl>,
## #   `CD4+_memory_T-cells_xCell` <dbl>, `CD4+_naive_T-cells_xCell` <dbl>,
## #   `CD4+_T-cells_xCell` <dbl>, `CD4+_Tcm_xCell` <dbl>, `CD4+_Tem_xCell` <dbl>,
## #   `CD8+_naive_T-cells_xCell` <dbl>, `CD8+_T-cells_xCell` <dbl>,
## #   `CD8+_Tcm_xCell` <dbl>, `CD8+_Tem_xCell` <dbl>, cDC_xCell <dbl>,
## #   Chondrocytes_xCell <dbl>, `Class-switched_memory_B-cells_xCell` <dbl>,
## #   CLP_xCell <dbl>, CMP_xCell <dbl>, DC_xCell <dbl>, ...
```

5.8 Method 5: ESTIMATE

```
estimate<-deconvo_tme(eset = eset_acrg, method = "estimate")
```

```
## [1] "Merged dataset includes 9940 genes (472 mismatched)."
## [1] "1 gene set: StromalSignature  overlap= 136"
## [1] "2 gene set: ImmuneSignature  overlap= 138"
```

```
head(estimate)
```

```
## # A tibble: 6 x 5
##   ID         StromalScore_estimate ImmuneScore_estimate ESTIMATEScore_estimate
##   <chr>                      <dbl>                <dbl>                   <dbl>
## 1 GSM1523727                -1250.                 268.                  -982.
## 2 GSM1523728                  197.                1334.                  1531.
## 3 GSM1523729                 -111.                 822.                   711.
```

```
## 4 GSM1523744                -119.                662.                 544.
## 5 GSM1523745                 324.               1015.                1339.
## 6 GSM1523746                -594.                621.                  27.0
## # i 1 more variable: TumorPurity_estimate <dbl>
```

5.9 Method 6: TIMER

```
timer<-deconvo_tme(eset = eset_acrg, method = "timer", group_list = rep("stad",dim(eset_
```

```
## [1] "Outlier genes: AGR2 B2M COL1A2 COL3A1 COX2 CYAT1 EEF1A1 EIF1 FTH1 GKN1 HUWE1 IGK
```

```
head(timer)
```

```
## # A tibble: 6 x 7
##   ID         B_cell_TIMER T_cell_CD4_TIMER T_cell_CD8_TIMER Neutrophil_TIMER
##   <chr>             <dbl>            <dbl>            <dbl>             <dbl>
## 1 GSM1523727        0.104            0.128            0.183             0.108
## 2 GSM1523728        0.103            0.130            0.192             0.118
## 3 GSM1523729        0.106            0.130            0.190             0.110
## 4 GSM1523744        0.101            0.126            0.187             0.111
## 5 GSM1523745        0.104            0.127            0.191             0.116
## 6 GSM1523746        0.105            0.129            0.192             0.111
## # i 2 more variables: Macrophage_TIMER <dbl>, DC_TIMER <dbl>
```

5.10 Method 7: quanTIseq

```
quantiseq<-deconvo_tme(eset = eset_acrg, tumor = TRUE, arrays = TRUE, scale_mrna = TRUE,
```

```
##
## Running quanTIseq deconvolution module

## Gene expression normalization and re-annotation (arrays: TRUE)

## Removing 17 genes with high expression in tumors

## Signature genes found in data set: 152/153 (99.35%)

## Mixture deconvolution (method: lsei)

## Deconvolution sucessful!
```

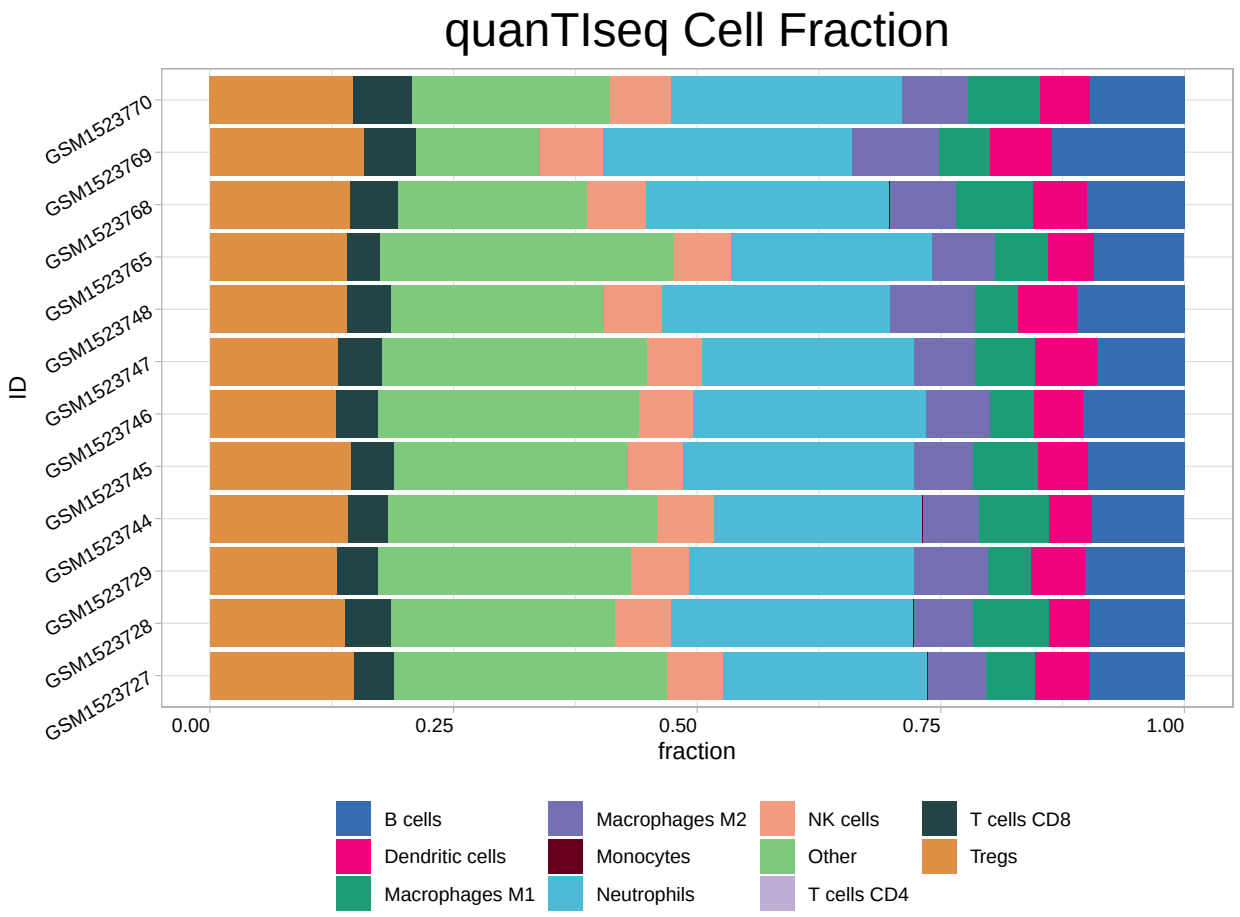
```
head(quantiseq)
```

```
## # A tibble: 6 x 12
##   ID         B_cells_quantiseq Macrophages_M1_quantiseq Macrophages_M2_quantiseq
##   <chr>                  <dbl>                    <dbl>                    <dbl>
## 1 GSM1523727            0.0983                   0.0510                   0.0598
## 2 GSM1523728            0.0967                   0.0795                   0.0607
## 3 GSM1523729            0.102                    0.0450                   0.0758
## 4 GSM1523744            0.0954                   0.0725                   0.0579
## 5 GSM1523745            0.0991                   0.0669                   0.0613
## 6 GSM1523746            0.105                    0.0453                   0.0662
## # i 8 more variables: Monocytes_quantiseq <dbl>, Neutrophils_quantiseq <dbl>,
## #   NK_cells_quantiseq <dbl>, T_cells_CD4_quantiseq <dbl>,
## #   T_cells_CD8_quantiseq <dbl>, Tregs_quantiseq <dbl>,
## #   Dendritic_cells_quantiseq <dbl>, Other_quantiseq <dbl>
```

```
res<-cell_bar_plot(input = quantiseq[1:12, ], id = "ID", features = colnames(quantiseq)
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## >>>>=== Palette option for random: 1: palette1; 2: palette2; 3: palette3;  4: palette
```



5.11 Method 8: IPS

```
ips<-deconvo_tme(eset = eset_acrg, method = "ips", plot= FALSE)
head(ips)
```

```
## # A tibble: 6 x 7
##   ID         MHC_IPS EC_IPS SC_IPS  CP_IPS AZ_IPS IPS_IPS
##   <chr>        <dbl>  <dbl>  <dbl>   <dbl>  <dbl>   <dbl>
## 1 GSM1523727    2.25  0.404 -0.192  0.220    2.68       9
## 2 GSM1523728    2.37  0.608 -0.578 -0.234    2.17       7
## 3 GSM1523729    2.10  0.480 -0.322  0.0993   2.36       8
## 4 GSM1523744    2.12  0.535 -0.333  0.0132   2.34       8
## 5 GSM1523745    1.91  0.559 -0.479  0.0880   2.08       7
## 6 GSM1523746    1.94  0.458 -0.346  0.261    2.31       8
```

5.12 Combination of above deconvolution results

```
tme_combine<-cibersort %>%
  inner_join(.,mcp,by       = "ID") %>%
  inner_join(.,xcell,by     = "ID") %>%
  inner_join(.,epic,by      = "ID") %>%
  inner_join(.,estimate,by  = "ID") %>%
  inner_join(.,timer,by     = "ID") %>%
  inner_join(.,quantiseq,by = "ID") %>%
  inner_join(.,ips,by       = "ID")
dim(tme_combine)
```

```
## [1]  50 138
```

5.13 How to customise the signature matrix for SVR and lesi algorithm

The recent surge in single-cell RNA sequencing has enabled us to identify novel microenvironmental cells, tumour microenvironmental characteristics, and tumour clonal signatures with high resolution. It is necessary to scrutinize, confirm and depict these features attained from high-dimensional single-cell information in bulk-seq with extended specimen sizes for clinical phenotyping. This is a demonstration using the results of 10X single-cell sequencing data of PBMC to construct gene signature matrix for `deconvo_tme` function and estimate the abundance of these cell types in bulk transcriptome data.

Download PBMC dataset through: https://cf.10xgenomics.com/samples/cell/pbmc3k/pbmc3k_filtered_gene_bc_matrices.tar.gz

Initialize the Seurat object with the raw (non-normalized data).

```
library(Seurat)
pbmc.data <- Read10X(data.dir = "./pbmc3k_filtered_gene_bc_matrices/filtered_gene_bc_mat
pbmc <- CreateSeuratObject(counts = pbmc.data, project = "pbmc3k", min.cells = 3, min.fe
```

Data prepare using Seurat's standard pipeline.

```
pbmc <- FindVariableFeatures(pbmc, selection.method = "vst", nfeatures = 2000, verbose =
pbmc <- NormalizeData(pbmc, normalization.method = "LogNormalize", scale.factor = 10000,
pbmc <- ScaleData(pbmc, features =  rownames(pbmc), verbose = FALSE)
```

```
pbmc <- RunPCA(pbmc, features = VariableFeatures(object = pbmc), verbose = FALSE)
pbmc <- FindNeighbors(pbmc, dims = 1:10, verbose = FALSE)
pbmc <- FindClusters(pbmc, resolution = 0.5, verbose = FALSE)
# Annotate cells according to seurat's tutorials
# https://satijalab.org/seurat/articles/pbmc3k_tutorial
new.cluster.ids <- c("Naive_CD4_T", "CD14_Mono", "Memory_CD4_T", "Bcells", "CD8_Tcell",
names(new.cluster.ids) <- levels(pbmc$seurat_clusters)
pbmc <- RenameIdents(pbmc, new.cluster.ids)
pbmc$celltype <- Idents(pbmc)
```

Generate reference matrix using generateRef_seurat function.

```
sm<- generateRef_seurat(sce = pbmc, celltype = "celltype", slot_out = "data")
```

```
## >>>---Assay used to find markers:
## [1] ">>>>> RNA"
##
##       Bcells    CD14_Mono    CD8_Tcell           DC  FCGR3A_Mono Memory_CD4_T
##          349          491          339           36          159          467
##  Naive_CD4_T           NK     Platelet
##          696          148           15
## >>> Find markers of each celltype...
## # A tibble: 450 x 7
## # Groups:   cluster [9]
##        p_val avg_log2FC pct.1 pct.2 p_val_adj cluster     gene
##        <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>       <chr>
##  1 5.43e-142      0.681 0.999 0.994 7.45e-138 Naive_CD4_T RPS6
##  2 5.65e-138      0.626 0.999 0.995 7.74e-134 Naive_CD4_T RPL32
##  3 5.62e-137      0.716 1     0.99  7.70e-133 Naive_CD4_T RPS12
##  4 1.90e-131      0.695 0.999 0.992 2.61e-127 Naive_CD4_T RPS27
##  5 2.36e-127      0.765 0.997 0.973 3.23e-123 Naive_CD4_T RPS25
##  6 3.96e-121      0.751 0.996 0.963 5.43e-117 Naive_CD4_T RPL31
##  7 2.91e-120      0.605 0.999 0.995 3.99e-116 Naive_CD4_T RPS14
##  8 1.74e-113      0.727 0.996 0.969 2.38e-109 Naive_CD4_T RPL9
##  9 4.38e-110      0.590 0.999 0.993 6.01e-106 Naive_CD4_T RPS3
## 10 6.80e-108      0.665 0.997 0.979 9.33e-104 Naive_CD4_T RPL30
## # i 440 more rows
## >>>-- Aggreating scRNAseq data...
```

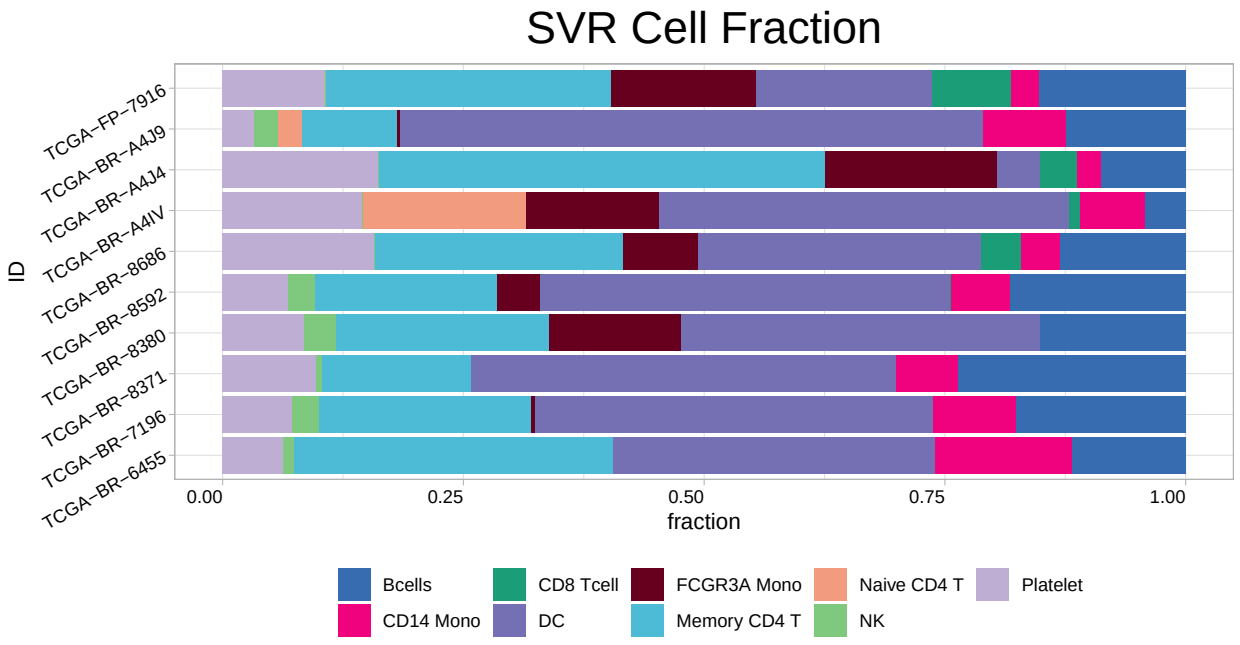
```
## >>>-- `orig.ident` was set as group. User can define through parameter `celltype` ...
```

Load the bulk RNA-seq data

```
data(eset_stad, package = "IOBR")
eset <- count2tpm(countMat = eset_stad, source = "local", idType = "ensembl")
svr<- deconvo_tme(eset = eset, reference  = sm,  method = "svr", arrays  = FALSE, absolu
head(svr)
```

```
## # A tibble: 6 x 13
##   ID           Naive_CD4_T_CIBERSORT CD14_Mono_CIBERSORT Memory_CD4_T_CIBERSORT
##   <chr>                        <dbl>               <dbl>                  <dbl>
## 1 TCGA-BR-6455                     0             0.143                   0.332
## 2 TCGA-BR-7196                     0             0.0862                  0.221
## 3 TCGA-BR-8371                     0             0.0642                  0.156
## 4 TCGA-BR-8380                     0             0.00125                 0.221
## 5 TCGA-BR-8592                     0             0.0621                  0.189
## 6 TCGA-BR-8686                     0             0.0411                  0.259
## # i 9 more variables: Bcells_CIBERSORT <dbl>, CD8_Tcell_CIBERSORT <dbl>,
## #   FCGR3A_Mono_CIBERSORT <dbl>, NK_CIBERSORT <dbl>, DC_CIBERSORT <dbl>,
## #   Platelet_CIBERSORT <dbl>, `P-value_CIBERSORT` <dbl>,
## #   Correlation_CIBERSORT <dbl>, RMSE_CIBERSORT <dbl>
```

```
res<-cell_bar_plot(input = svr, features = colnames(svr)[2:10], title = "SVR Cell Fracti
```



5.14 References

If you use this package in your work, please cite both our package and the method(s) you are using.

Citation and licenses of these deconvolution methods

CIBERSORT; free for non-commerical use only; Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., … Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles. Nature Methods, 12(5), 453–457. https://doi.org/10.1038/nmeth.3337;

ESTIMATE; free (GPL2.0); Vegesna R, Kim H, Torres-Garcia W, …, Verhaak R. (2013). Inferring tumour purity and stromal and immune cell admixture from expression data. Nature Communications 4, 2612. http://doi.org/10.1038/ncomms3612;

quanTIseq; free (BSD); Finotello, F., Mayer, C., Plattner, C., Laschober, G., Rieder, D., Hackl, H., …, Sopper, S. (2019). Molecular and pharmacological modulators of the tumor immune contexture revealed by deconvolution of RNA-seq data. Genome medicine, 11(1), 34. https://doi.org/10.1186/s13073-019-0638-6;

TIMER; free (GPL 2.0); Li, B., Severson, E., Pignon, J.-C., Zhao, H., Li, T., Novak, J., … Liu, X. S. (2016). Comprehensive analyses of tumor immunity: implications for cancer immunotherapy. Genome Biology, 17(1), 174. https://doi.org/10.1186/s13059-016-1028-7;

IPS; free (BSD); P. Charoentong et al., Pan-cancer Immunogenomic Analyses Reveal Genotype-Immunophenotype Relationships and Predictors of Response to Checkpoint Blockade. Cell Reports 18, 248-262 (2017). https://doi.org/10.1016/j.celrep.2016.12.019;

MCPCounter; free (GPL 3.0); Becht, E., Giraldo, N. A., Lacroix, L., Buttard, B., Elarouci, N., Petitprez, F., … de Reyniès, A. (2016). Estimating the population abundance of tissue-infiltrating immune and stromal cell populations using gene expression. Genome Biology, 17(1), 218. https://doi.org/10.1186/s13059-016-1070-5;

xCell; free (GPL 3.0); Aran, D., Hu, Z., & Butte, A. J. (2017). xCell: digitally portraying the tissue cellular heterogeneity landscape. Genome Biology, 18(1), 220. https://doi.org/10.1186/s13059-017-1349-1;

EPIC; free for non-commercial use only (Academic License); Racle, J., de Jonge, K., Baumgaertner, P., Speiser, D. E., & Gfeller, D. (2017). Simultaneous enumeration of cancer and immune cell types from bulk tumor gene expression data. ELife, 6, e26476. https://doi.org/10.7554/eLife.26476;

GSVA free (GPL (>= 2)) Hänzelmann S, Castelo R, Guinney J (2013). "GSVA: gene set variation analysis for microarray and RNA-Seq data." BMC Bioinformatics, 14, 7. doi: 10.1186/1471-2105-14-7, http://www.biomedcentral.com/1471-2105/14/7

Chapter 6

Signature Score and Relevant phenotypes

6.1 Loading packages

Load the IOBR package in your R session after the installation is complete:

```
library(IOBR)
library(survminer)
library(tidyverse)
```

6.2 Downloading data for example

Obtaining data set from GEO Gastric cancer: GSE62254 using `GEOquery` R package.

```
if (!requireNamespace("GEOquery", quietly = TRUE))  BiocManager::install("GEOquery")
library("GEOquery")
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo <- getGEO(GEO = "GSE62254", getGPL  = F, destdir = "./")
eset    <- eset_geo[[1]]
eset    <- exprs(eset)
eset[1:5,1:5]
```

```
##           GSM1523727 GSM1523728 GSM1523729 GSM1523744 GSM1523745
## 1007_s_at  3.2176645  3.0624323  3.0279131   2.921683  2.8456013
## 1053_at    2.4050109  2.4394879  2.2442708   2.345916  2.4328582
```

```
## 117_at     1.4933412  1.8067380  1.5959665   1.839822  1.8326058
## 121_at     2.1965561  2.2812181  2.1865556   2.258599  2.1874363
## 1255_g_at  0.8698382  0.9502466  0.8125414   1.012860  0.9441993
```

6.3 Gene Annotation

Annotation of genes in the expression matrix and removal of duplicate genes.

```
# Load the annotation file `anno_hug133plus2` in IOBR.
head(anno_hug133plus2)
```

```
## # A tibble: 6 x 2
##   probe_id  symbol
##   <fct>     <fct>
## 1 1007_s_at MIR4640
## 2 1053_at   RFC2
## 3 117_at    HSPA6
## 4 121_at    PAX8
## 5 1255_g_at GUCA1A
## 6 1294_at   MIR5193
```

```
# Conduct gene annotation using `anno_hug133plus2` file; If identical gene symbols exi

eset<-anno_eset(eset       = eset,
                annotation = anno_hug133plus2,
                symbol     = "symbol",
                probe      = "probe_id",
                method     = "mean")
eset[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
## LOC101928826   4.219303   4.219670   4.213252
```

6.4 Estimation of signatures

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = signature_collection,
                             method          = "pca",
                             mini_gene_count = 2)

sig_tme <- t(column_to_rownames(sig_tme, var = "ID"))
sig_tme[1:5, 1:3]
```

```
##                   GSM1523727 GSM1523728 GSM1523729
## CD_8_T_effector   -2.5513794  0.7789141 -2.1770675
## DDR               -0.8747614  0.7425162 -1.3272054
## APM                1.1098368  2.1988688 -0.9516419
## Immune_Checkpoint -2.3701787  0.9455120 -1.4844104
## CellCycle_Reg      0.1063358  0.7583302 -0.3649795
```

6.5 Combining score data and phenotype data

```
data("pdata_acrg", package = "IOBR")
head(pdata_acrg)
```

```
##            ID ProjectID  Technology       platform Gender Age RFS_time
## 71 GSM1523727  GSE62254 Affymetrix  HG-U133_Plus_2      M  67     3.97
## 72 GSM1523728  GSE62254 Affymetrix  HG-U133_Plus_2      F  68     4.03
## 73 GSM1523729  GSE62254 Affymetrix  HG-U133_Plus_2      F  42    74.97
## 74 GSM1523744  GSE62254 Affymetrix  HG-U133_Plus_2      M  69    89.77
## 75 GSM1523745  GSE62254 Affymetrix  HG-U133_Plus_2      M  68    84.60
## 76 GSM1523746  GSE62254 Affymetrix  HG-U133_Plus_2      M  56     5.77
##    RFS_status OS_time OS_status     Lauren Differtiation AJCC_Stage_confuse
## 71         NA   88.73         0 Intestinal            MD                 2
## 72         NA   88.23         0 Intestinal            PD                 2
## 73          0   88.23         0    Diffuse            PD                 2
## 74          0  105.70         0    Diffuse            PD                 2
## 75          0  105.53         0    Diffuse            PD                 3
## 76          1   25.50         1      Mixed            PD                 2
```

```
##    T_stage N_stage M_stage Lymph_node_examined Positive_lymph_nodes
## 71       2       1       0                  20                    3
## 72       2       1       0                  40                    1
## 73       2       1       0                  21                    1
## 74       2       1       0                  24                    3
## 75       3       2       0                  52                   11
## 76       2       1       0                  22                    5
##    Revisedlocation MSI EBV Hpylori   Subtype TP53mutated B.cells.naive
## 71            Body   1   0      NA       MSI           0   0.006611704
## 72            Body   1  NA      NA       MSI           0   0.000000000
## 73          Antrum   0   0       0 MSS/TP53+           1   0.003306927
## 74          Antrum   1   0       1       MSI           0   0.000000000
## 75          Antrum   0   0      NA MSS/TP53-           0   0.000000000
## 76          Antrum   0   0       0 MSS/TP53-           0   0.013619480
##    B.cells.memory Plasma.cells T.cells.CD8 T.cells.CD4.naive
## 71   0.014570868   0.17555729  0.05712737                 0
## 72   0.036202099   0.08523233  0.05336971                 0
## 73   0.020935673   0.10489546  0.00000000                 0
## 74   0.072648177   0.08755997  0.03465107                 0
## 75   0.009798381   0.12251030  0.00000000                 0
## 76   0.012784581   0.15602714  0.00000000                 0
##    T.cells.CD4.memory.resting T.cells.CD4.memory.activated
## 71                  0.1439895                   0.025159835
## 72                  0.1250515                   0.049617381
## 73                  0.1849220                   0.008407981
## 74                  0.1396439                   0.055268600
## 75                  0.1916398                   0.036578672
## 76                  0.1905921                   0.008992440
##    T.cells.follicular.helper T.cells.regulatory..Tregs. T.cells.gamma.delta
## 71                0.02453957                         0          0.00000000
## 72                0.05318251                         0          0.00000000
## 73                0.05098080                         0          0.03714459
## 74                0.07825130                         0          0.00000000
## 75                0.02223859                         0          0.02657259
## 76                0.04740728                         0          0.04283296
##    NK.cells.resting NK.cells.activated Monocytes Macrophages.M0 Macrophages.M1
## 71      0.000000000        0.049325657         0     0.03865693     0.06910287
```

```
## 72      0.000000000        0.081481924         0     0.07370723     0.08016443
## 73      0.000000000        0.025252673         0     0.00000000     0.06161940
## 74      0.000000000        0.016121853         0     0.08866391     0.08173804
## 75      0.001738259        0.006267907         0     0.15255902     0.07161270
## 76      0.000000000        0.052117471         0     0.10298038     0.03246627
##    Macrophages.M2 Dendritic.cells.resting Dendritic.cells.activated
## 71      0.1829208               0.0000000              0.022904531
## 72      0.1320919               0.0000000              0.060491149
## 73      0.1170839               0.1171129              0.032385282
## 74      0.1441202               0.0000000              0.060937005
## 75      0.1919279               0.0000000              0.006087801
## 76      0.1093805               0.0000000              0.023914527
##    Mast.cells.resting Mast.cells.activated Eosinophils Neutrophils.x P.value
## 71        0.069286038          0.000000000 0.006315889    0.11393115       0
## 72        0.003322764          0.005197745 0.056141443    0.10474585       0
## 73        0.052571970          0.000000000 0.104493538    0.07888690       0
## 74        0.012494201          0.006833953 0.050435095    0.07063272       0
## 75        0.000000000          0.033928747 0.017164438    0.10937487       0
## 76        0.014373257          0.002764802 0.115772442    0.07397439       0
##    Pearson.Correlation      RMSE    T.cells CD8.T.cells Cytotoxic.lymphocytes
## 71           0.3359926 0.9415173 -0.9275804   0.8492914            -1.1005262
## 72           0.4793134 0.8827802 -0.5306279  -0.2017907             0.1858499
## 73           0.3638005 0.9308186 -0.9566316   0.2411951            -0.8800338
## 74           0.3569989 0.9332100 -1.0464552  -0.5771205            -0.5619472
## 75           0.4226987 0.9062522 -0.6796120   0.6670229            -0.3361456
## 76           0.4113346 0.9112588 -0.6978480  -1.1110102            -0.7631710
##        NK.cells   B.lineage Monocytic.lineage Myeloid.dendritic.cells
## 71 -0.083623737 -0.54974243       -1.40389061              -0.7589211
## 72  0.156167025 -0.33750363       -0.03696397              -0.6393975
## 73  0.003538847  0.01597566       -0.67105808               0.7452174
## 74 -0.010774923 -0.56740438        0.06877240              -0.2511140
## 75 -0.028429092 -0.73180429        0.21574792              -0.1165082
## 76  0.466964699  0.15583392       -0.97524359              -0.7448360
##    Neutrophils.y Endothelial.cells Fibroblasts StromalScore ImmuneScore
## 71    -0.9527759       -1.42753593 -1.22754105   -1.8047694  -1.3347047
## 72     0.5640500       -0.17320689  0.41586717    0.1825225   0.1950604
## 73    -0.3415288       -0.25784297  0.04110246   -0.1863425  -0.4960305
```

```
## 74    -1.2984378       -1.05394707  0.00743277   -0.2731398  -0.7950682
## 75     0.4227674        0.03025664  0.32245183    0.3165798  -0.2416774
## 76    -0.4411653       -0.29582293 -0.68833740   -0.9119449  -0.8475150
##    ESTIMATEScore TumorPurity ProjectID2   TMEscoreA    TMEscoreB    TMEscore
## 71   -1.70632719   1.1687573   GSE62254 -1.06110812 -1.270222413  0.60585688
## 72    0.20292720          NA   GSE62254  1.14698153 -0.333585646  0.73717229
## 73   -0.35721073  -1.3859061   GSE62254 -0.89026369 -0.007906066 -0.35452887
## 74   -0.55795758  -0.9855180   GSE62254 -0.01116022 -0.984841623  0.79880007
## 75    0.05885805          NA   GSE62254 -0.27102383 -0.017592784 -0.09554256
## 76   -0.94967710  -0.2162267   GSE62254 -0.94526260  0.161818627 -0.51527214
##    TMEscore_binary
## 71             Low
## 72            High
## 73             Low
## 74            High
## 75             Low
## 76             Low
```

```
input <- combine_pd_eset(eset = sig_tme, pdata = pdata_acrg, scale = T)
```

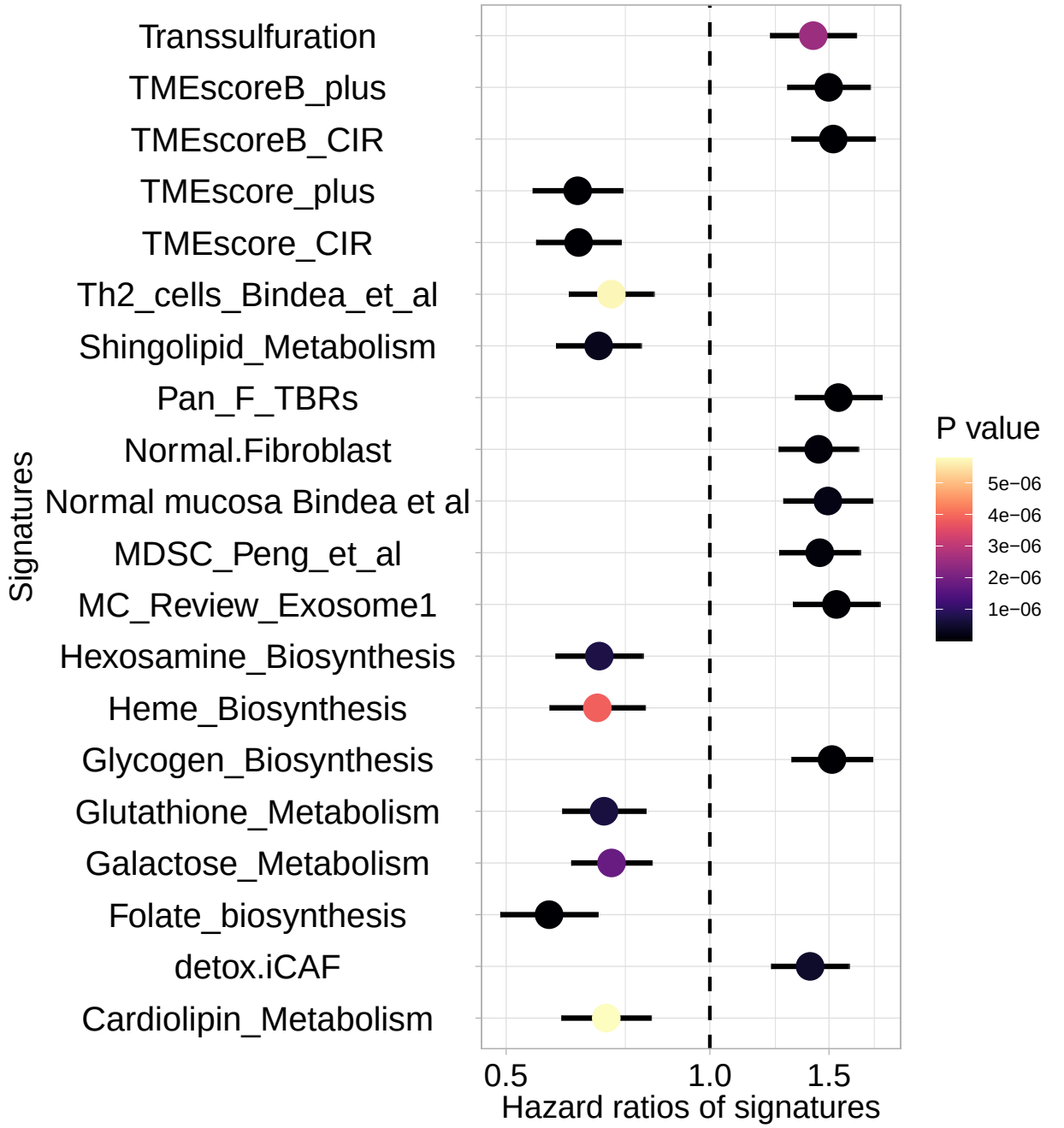
6.6 Identifying features associated with survival

```
res<- batch_surv(pdata    = input,
                 time     = "OS_time",
                 status   = "OS_status",
                 variable = colnames(input)[69:ncol(input)])
head(res)
```

```
## # A tibble: 6 x 5
##   ID                          P    HR CI_low_0.95 CI_up_0.95
##   <chr>                   <dbl> <dbl>       <dbl>      <dbl>
## 1 Folate_biosynthesis   1.00e-10 0.579       0.490      0.683
## 2 TMEscore_CIR          1.32e- 9 0.640       0.554      0.739
## 3 Glycogen_Biosynthesis 3.24e- 9 1.52        1.32       1.74
## 4 Pan_F_TBRs            6.33e- 9 1.55        1.34       1.80
## 5 TMEscoreB_CIR         7.17e- 9 1.52        1.32       1.75
## 6 TMEscore_plus         8.08e- 9 0.638       0.547      0.743
```

Use forest plots `sig_forest` to show the most relevant variables to overall survival

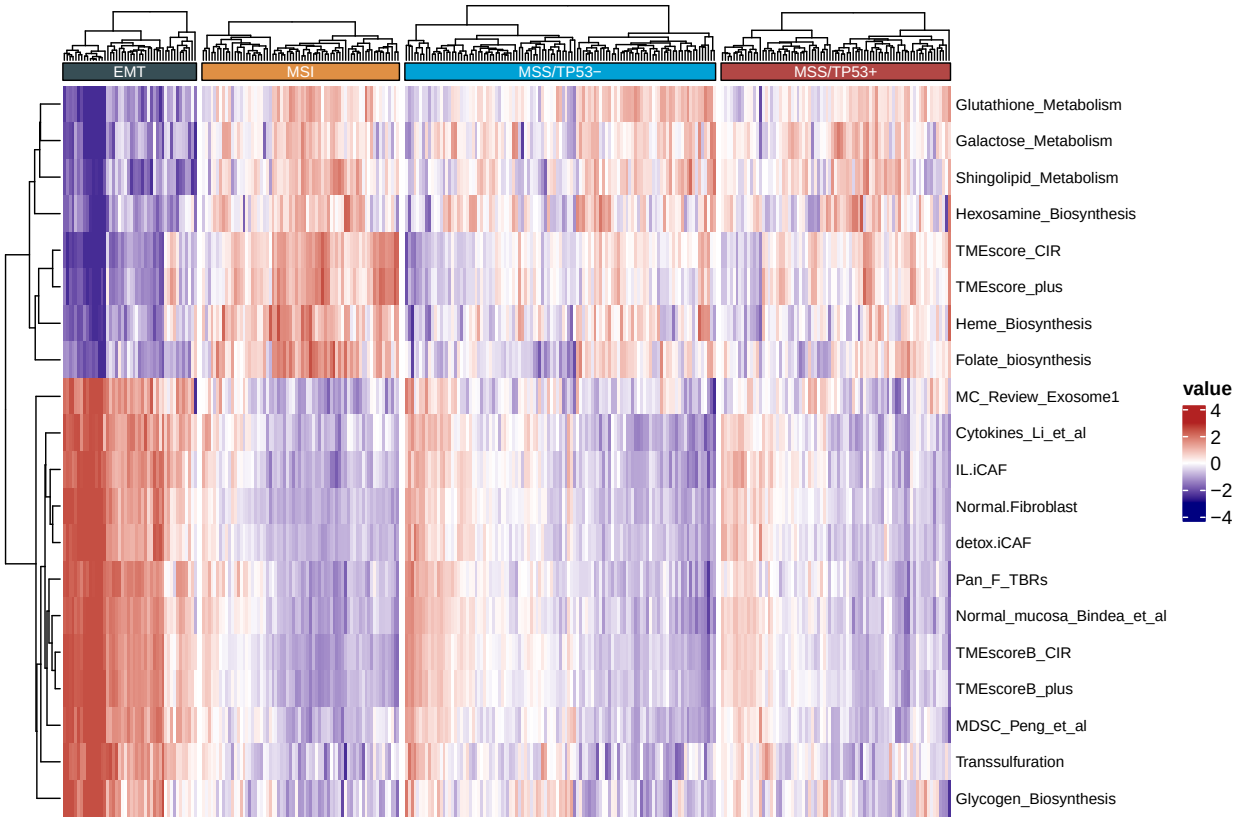
```
res<- res[nchar(res$ID)<=28, ]
p1<- sig_forest(res, signature = "ID", n = 20)
```



6.7 Visulization using heatmap

Relationship between Signatures and molecular typing. Heatmap visualisation using `IOBR`'s `sig_heatmap`

```
p2 <- sig_heatmap(input         = input,
                  features      = res$ID[1:20],
                  group         = "Subtype",
                  palette_group = "jama",
                  palette       = 6,
                  path          = "result" )
```



6.8 Focus on target signatures

```
p1 <- sig_box(data          = input,
              signature     = "Glycogen_Biosynthesis",
              variable      = "Subtype",
              jitter        = FALSE,
```

```
              cols           =  NULL,
              palette        = "jama",
              show_pvalue    = TRUE,
              size_of_pvalue = 5,
              hjust          = 1,
              angle_x_text   = 60,
              size_of_font   = 8)
```

```
## # A tibble: 6 x 8
##   .y.       group1    group2          p    p.adj p.format p.signif method
##   <chr>     <chr>     <chr>       <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature EMT       MSI       5.39e-15 3.20e-14 5.4e-15  ****     Wilcoxon
## 2 signature EMT       MSS/TP53- 5.53e-13 2.8 e-12 5.5e-13  ****     Wilcoxon
## 3 signature EMT       MSS/TP53+ 1.90e-12 7.6 e-12 1.9e-12  ****     Wilcoxon
## 4 signature MSI       MSS/TP53- 1.14e- 3 3.4 e- 3 0.0011   **       Wilcoxon
## 5 signature MSI       MSS/TP53+ 7.05e- 3 1.4 e- 2 0.0071   **       Wilcoxon
## 6 signature MSS/TP53- MSS/TP53+ 7.16e- 1 7.2 e- 1 0.7161   ns       Wilcoxon
```

```
p2 <- sig_box(data           = input,
              signature      = "Pan_F_TBRs",
              variable       = "Subtype",
              jitter         = FALSE,
              cols           = NULL,
              palette        = "jama",
              show_pvalue    = TRUE,
              angle_x_text   = 60,
              hjust          = 1,
              size_of_pvalue = 5,
              size_of_font   = 8)
```

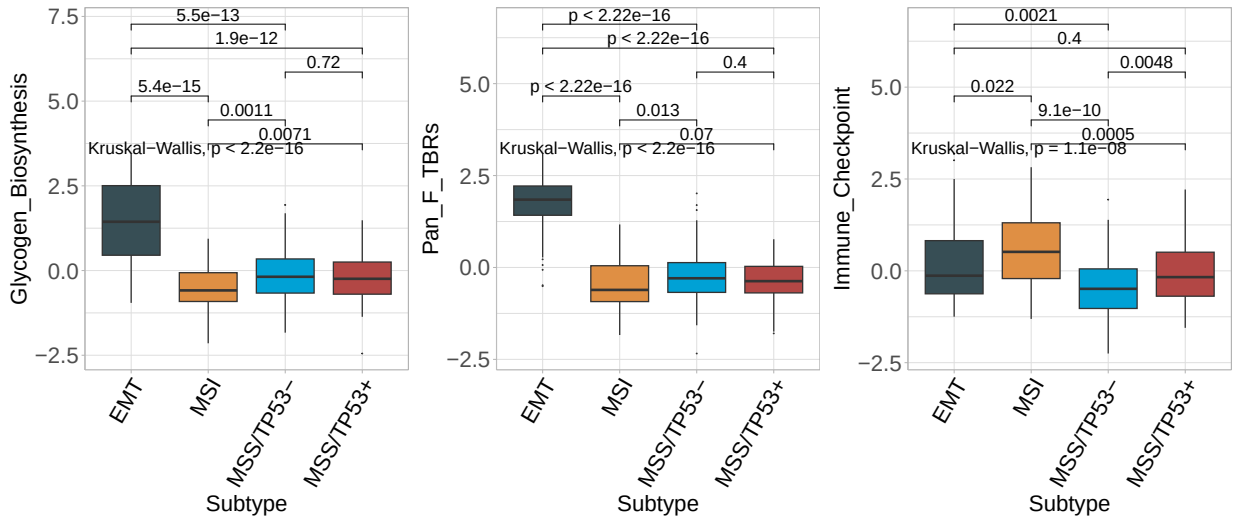
```
## # A tibble: 6 x 8
##   .y.       group1    group2          p    p.adj p.format p.signif method
##   <chr>     <chr>     <chr>       <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature EMT       MSI       7.98e-17 3.20e-16 <2e-16   ****     Wilcoxon
## 2 signature EMT       MSS/TP53- 1.70e-17 1   e-16 <2e-16   ****     Wilcoxon
## 3 signature EMT       MSS/TP53+ 2.57e-17 1.3 e-16 <2e-16   ****     Wilcoxon
## 4 signature MSI       MSS/TP53- 1.32e- 2 4   e- 2 0.013    *        Wilcoxon
## 5 signature MSI       MSS/TP53+ 6.99e- 2 1.4 e- 1 0.070    ns       Wilcoxon
```

```
## 6 signature MSS/TP53- MSS/TP53+ 4.02e- 1 4   e- 1 0.402    ns       Wilcoxon
```

```
p3 <- sig_box(data           = input,
              signature      = "Immune_Checkpoint",
              variable       = "Subtype",
              jitter         = FALSE,
              cols           = NULL,
              palette        = "jama",
              show_pvalue    = TRUE,
              angle_x_text   = 60,
              hjust          = 1,
              size_of_pvalue = 5,
              size_of_font   = 8)
```

```
## # A tibble: 6 x 8
##   .y.       group1    group2          p        p.adj p.format p.signif method
##   <chr>     <chr>     <chr>       <dbl>        <dbl> <chr>    <chr>    <chr>
## 1 signature EMT       MSI       2.20e- 2 0.044        0.0220   *        Wilcoxon
## 2 signature EMT       MSS/TP53- 2.11e- 3 0.0085       0.0021   **       Wilcoxon
## 3 signature EMT       MSS/TP53+ 4.03e- 1 0.4          0.4026   ns       Wilcoxon
## 4 signature MSI       MSS/TP53- 9.13e-10 0.0000000055 9.1e-10  ****     Wilcoxon
## 5 signature MSI       MSS/TP53+ 5.03e- 4 0.0025       0.0005   ***      Wilcoxon
## 6 signature MSS/TP53- MSS/TP53+ 4.82e- 3 0.014        0.0048   **       Wilcoxon
```

```
p1|p2|p3
```



6.9 Survival analysis and visulization

6.9.1 Kaplan-Meier plot

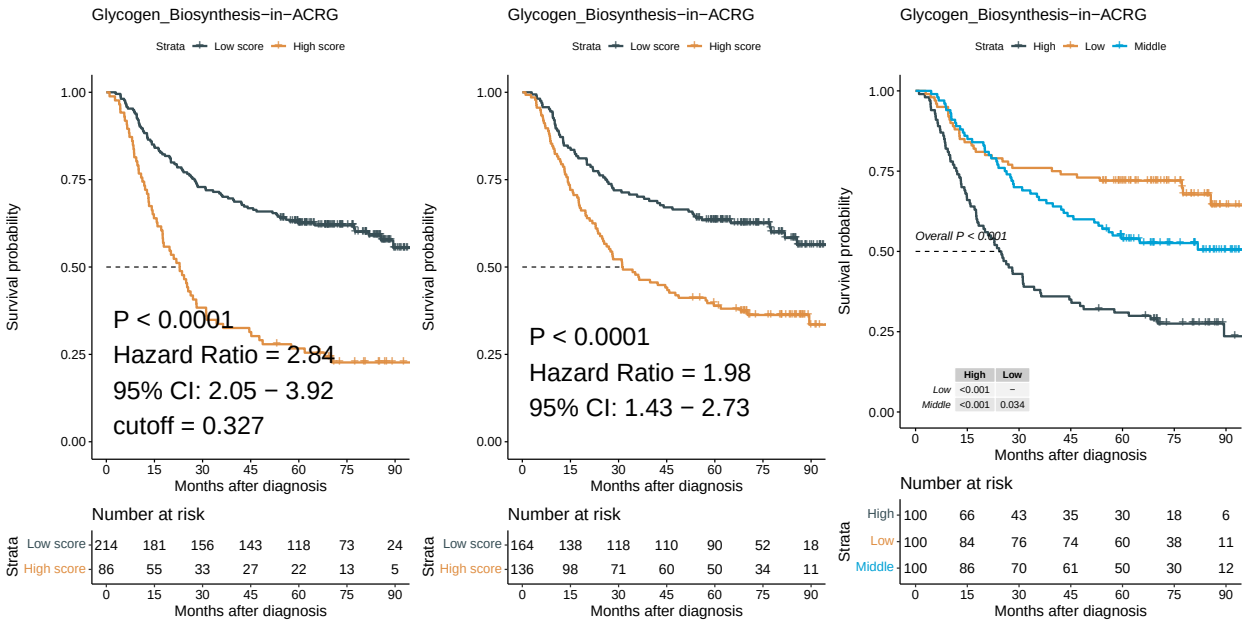
Displaying the outcomes of survival analyses using Kaplan-Meier plot. Multiple stratifications of the signature were used to judge the efficacy of this metric in predicting patient survival.

```
res <-       sig_surv_plot(input_pdata       = input,
                           signature         = "Glycogen_Biosynthesis",
                           cols              = NULL,
                           palette           = "jama",
                           project           = "ACRG",
                           time              = "OS_time",
                           status            = "OS_status",
                           time_type         = "month",
                           save_path         = "result")
```

```
##           ID   time status Glycogen_Biosynthesis group3 group2 bestcutoff
## 1 GSM1523727  88.73      0            -0.3612213 Middle    Low        Low
## 2 GSM1523728  88.23      0            -0.6926726    Low    Low        Low
## 3 GSM1523729  88.23      0            -0.9388531    Low    Low        Low
## 4 GSM1523744 105.70      0            -1.1825136    Low    Low        Low
## 5 GSM1523745 105.53      0            -0.3034304 Middle    Low        Low
## 6 GSM1523746  25.50      1             0.7517934   High   High       High
```

```
## [1] ">>>>>>>>>>"
```

```
res$plots
```



6.9.2 Time-Dependent ROC curve

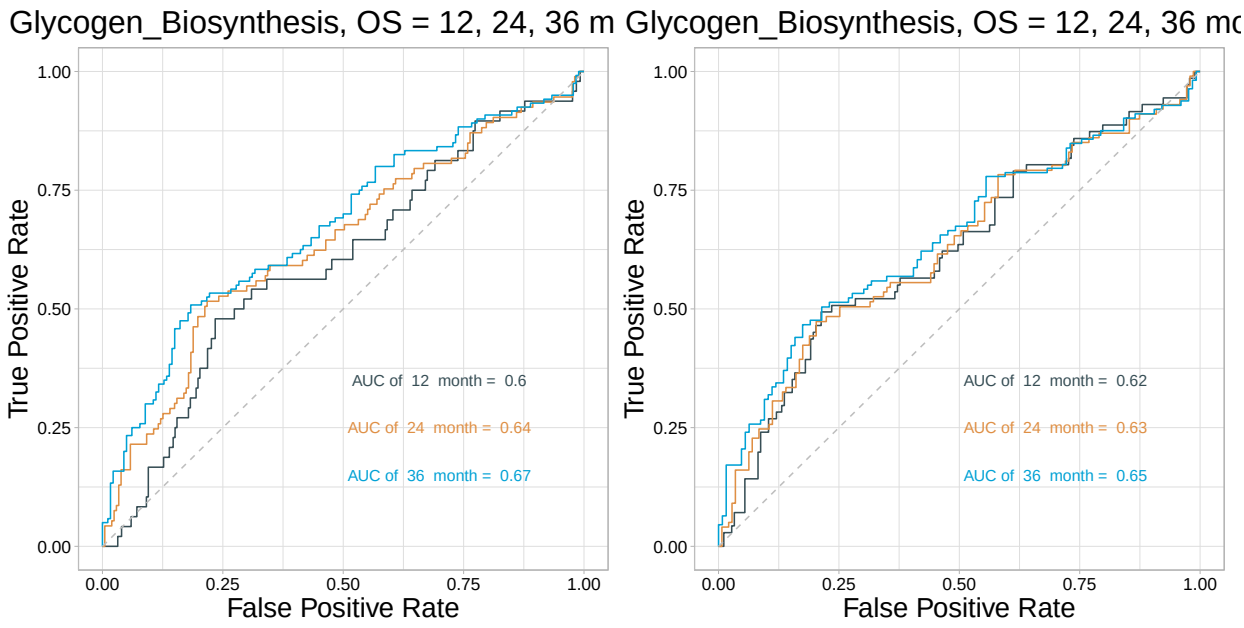
```
p1<- roc_time(input      = input,
              vars       = "Glycogen_Biosynthesis",
              time       = "OS_time",
              status     = "OS_status",
              time_point = c(12, 24, 36),
              time_type  = "month",
              palette    = "jama",
              cols       = "normal",
              seed       = 1234,
              show_col   = FALSE,
              path       = "result",
              main       = "OS",
              index      = 1,
              fig.type   = "pdf",
              width      = 5,
              height     = 5.2)
```

```
## [1] ">>>-- Range of Time: "
## [1]   1.0 105.7
```

```
p2<- roc_time(input      = input,
              vars       = "Glycogen_Biosynthesis",
              time       = "RFS_time",
              status     = "RFS_status",
              time_point = c(12, 24, 36),
              time_type  = "month",
              palette    = "jama",
              cols       = "normal",
              seed       = 1234,
              show_col   = FALSE,
              path       = "result",
              main       = "OS",
              index      = 1,
              fig.type   = "pdf",
              width      = 5,
              height     = 5.2)
```

```
## [1] ">>>-- Range of Time: "
## [1]   0.10 100.87
```

```
p1|p2
```



6.10 Batch correlation analysis

6.10.1 Finding continuity variables associated with signatures

Identifying genes or signatures related to the target signatures

6.10.1.1 Correlation between two variables

```
res <- batch_cor(data = input, target = "Glycogen_Biosynthesis", feature = colnames(inpu
```

```
## # A tibble: 6 x 6
##   sig_names                          p.value statistic    p.adj log10pvalue stars
##   <chr>                                <dbl>     <dbl>    <dbl>       <dbl> <fct>
## 1 TMEscoreB_CIR                     8.89e-42     0.678 2.27e-39        41.1 ****
## 2 Glycine__Serine_and_Threonine_M~ 7.49e-40    -0.666 9.54e-38        39.1 ****
## 3 Ether_Lipid_Metabolism           3.84e-39     0.662 3.27e-37        38.4 ****
## 4 MDSC_Peng_et_al                  1.13e-38     0.659 7.21e-37        37.9 ****
## 5 Glycerophospholipid_Metabolism   8.72e-38    -0.653 4.44e-36        37.1 ****
## 6 TIP_Release_of_cancer_cell_anti~ 2.32e-37    -0.650 9.86e-36        36.6 ****
```

```
head(res)
```

```
## # A tibble: 6 x 6
##   sig_names                          p.value statistic    p.adj log10pvalue stars
##   <chr>                                <dbl>     <dbl>    <dbl>       <dbl> <fct>
## 1 TMEscoreB_CIR                     8.89e-42     0.678 2.27e-39        41.1 ****
## 2 Glycine__Serine_and_Threonine_M~ 7.49e-40    -0.666 9.54e-38        39.1 ****
## 3 Ether_Lipid_Metabolism           3.84e-39     0.662 3.27e-37        38.4 ****
## 4 MDSC_Peng_et_al                  1.13e-38     0.659 7.21e-37        37.9 ****
## 5 Glycerophospholipid_Metabolism   8.72e-38    -0.653 4.44e-36        37.1 ****
## 6 TIP_Release_of_cancer_cell_anti~ 2.32e-37    -0.650 9.86e-36        36.6 ****
```

```
p1<- get_cor(eset = sig_tme, pdata = pdata_acrg, is.matrix = TRUE, var1 = "Glycogen_Bios
             var2 = "TMEscore_CIR", subtype = "Subtype", palette = "aaas", path = "resul
```

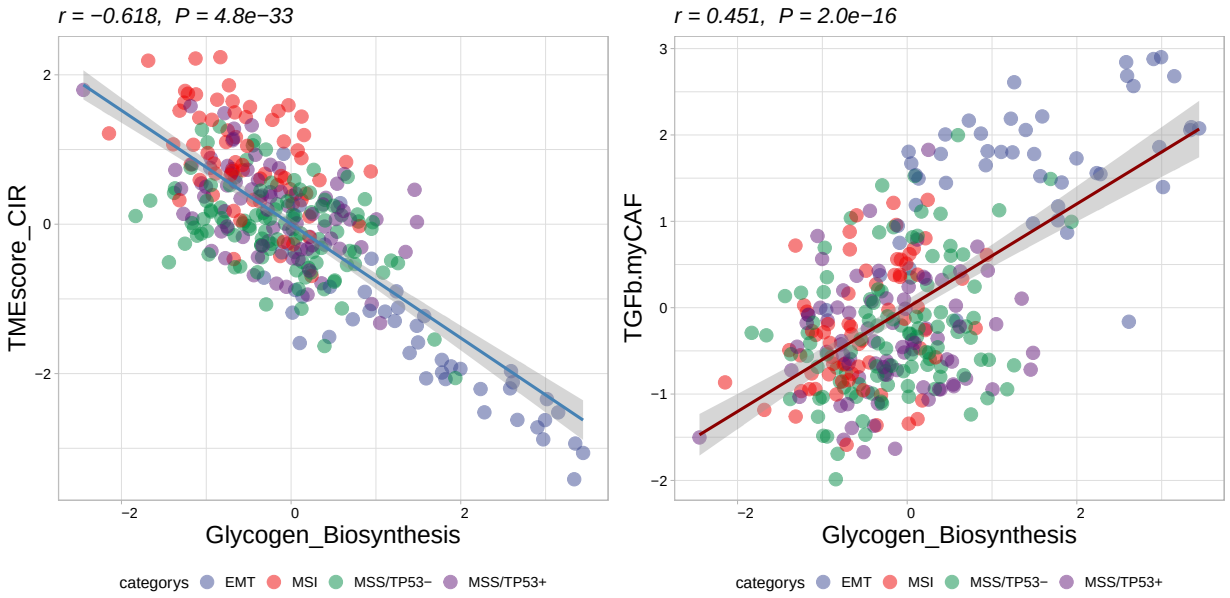
```
##
##  Spearman's rank correlation rho
##
## data:  data[, var1] and data[, var2]
## S = 7282858, p-value < 2.2e-16
```

```
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##        rho
## -0.6184309
##
## [1] ">>>--- The exact p value is: 4.78971420439895e-33"
##       EMT       MSI MSS/TP53- MSS/TP53+
##        46        68       107        79
```

```
p2<- get_cor(eset = sig_tme, pdata = pdata_acrg, is.matrix = TRUE, var1 = "Glycogen_Bios
             var2 = "TGFb.myCAF", subtype = "Subtype", palette = "aaas", path = "result"
```

```
##
##  Spearman's rank correlation rho
##
## data:  data[, var1] and data[, var2]
## S = 2471758, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.4507143
##
## [1] ">>>--- The exact p value is: 2.04505761057615e-16"
##       EMT       MSI MSS/TP53- MSS/TP53+
##        46        68       107        79
```

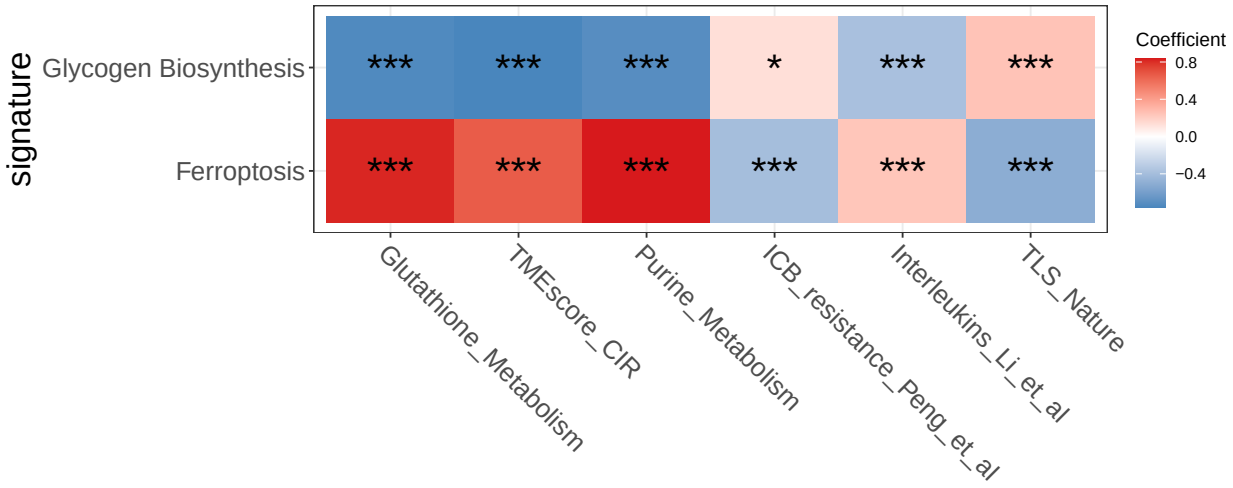
```
p1|p2
```



6.10.1.2 Demonstrate correlation between multiple variables

Visualisation via correlation matrix

```
feas1 <- c("Glycogen_Biosynthesis", "Ferroptosis")
feas2 <- c("Glutathione_Metabolism", "TMEscore_CIR", "Purine_Metabolism", "ICB_resistanc
p <- get_cor_matrix(data          = input,
                    feas1         = feas2,
                    feas2         = feas1,
                    method        = "pearson",
                    font.size.star = 8,
                    font.size     = 15,
                    fill_by_cor   = FALSE,
                    round.num     = 1,
                    path          = "result")
```

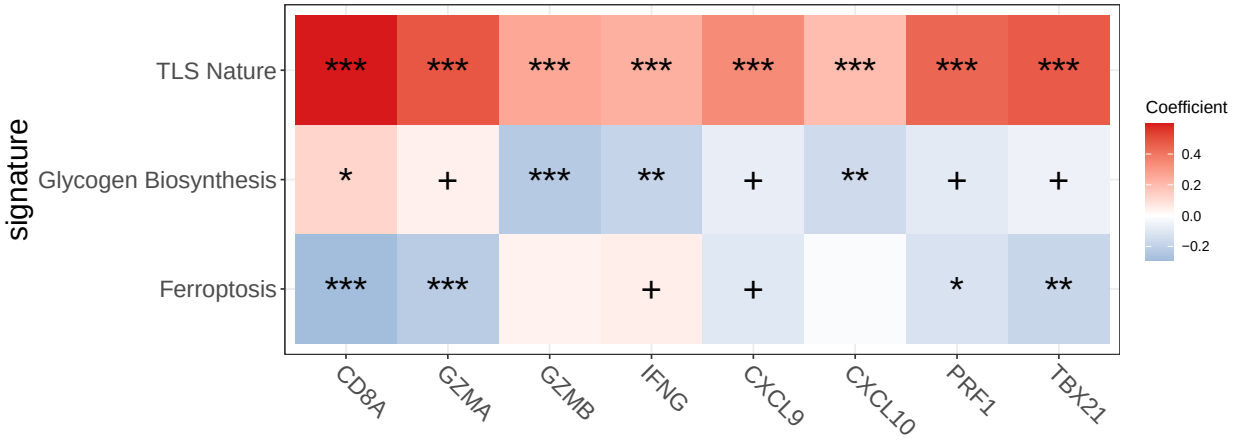


Demonstrate the correlation between signatures and genes

```
input2 <- combine_pd_eset(eset = eset, pdata =  input[, c("ID", "Glycogen_Biosynthesis",
feas1 <- c("Glycogen_Biosynthesis","TLS_Nature", "Ferroptosis")
feas2 <- signature_collection$CD_8_T_effector
feas2
```

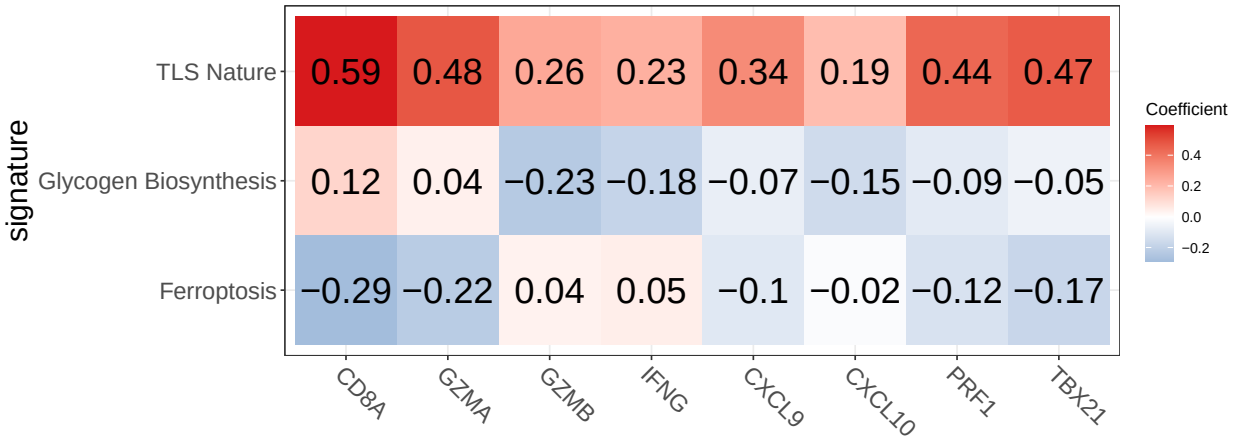
```
## [1] "CD8A"   "GZMA"   "GZMB"   "IFNG"   "CXCL9"  "CXCL10" "PRF1"   "TBX21"
```

```
p <- get_cor_matrix(data           = input2,
                    feas1          = feas2,
                    feas2          = feas1,
                    method         = "pearson",
                    scale          = T,
                    font.size.star = 8,
                    font.size      = 15,
                    fill_by_cor    = FALSE,
                    round.num      = 1,
                    path           = "result")
```



Users can customize the image using parameters.

```
p <- get_cor_matrix(data           = input2,
                    feas1          = feas2,
                    feas2          = feas1,
                    method         = "pearson",
                    scale          = T,
                    font.size.star = 8,
                    font.size      = 15,
                    fill_by_cor    = TRUE,
                    round.num      = 2,
                    path           = "result")
```



6.10.2 Identifying Category Variables Linked to Signatures

6.10.2.1 For binary variable

```
res <- batch_wilcoxon(data = input, target = "TMEscore_binary", feature = colnames(input
```

```
##
## High  Low
##   71  228
```

```
head(res)
```

```
## # A tibble: 6 x 8
##   sig_names         p.value   High    Low statistic    p.adj log10pvalue stars
##   <chr>               <dbl>  <dbl>  <dbl>     <dbl>    <dbl>       <dbl> <fct>
## 1 TMEscore_CIR     4.44e-37  1.17  -0.365      1.54 1.14e-34        36.4 ****
## 2 TMEscore_plus    3.97e-34  1.23  -0.380      1.61 5.08e-32        33.4 ****
## 3 TMEscoreA_plus   1.68e-25  1.18  -0.359      1.54 1.44e-23        24.8 ****
## 4 TMEscoreB_CIR    5.59e-24 -0.881  0.279     -1.16 3.36e-22        23.3 ****
## 5 ADP_Ribosylation 6.56e-24  1.06  -0.329      1.39 3.36e-22        23.2 ****
## 6 TMEscoreA_CIR    1.02e-22  1.11  -0.337      1.45 3.80e-21        22.0 ****
```

```
p1 <- sig_box(data           = input,
              signature      = res$sig_names[1],
              variable       = "TMEscore_binary",
              jitter         = FALSE,
              cols           =  NULL,
              palette        = "jco",
              show_pvalue    = TRUE,
              size_of_pvalue = 5,
              hjust          = 1,
              angle_x_text   = 60,
              size_of_font   = 8)
```

```
## # A tibble: 1 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature High   Low    4.44e-37 4.40e-37 <2e-16   ****     Wilcoxon
```

```
p2 <- sig_box(data           = input,
              signature      = res$sig_names[2],
```

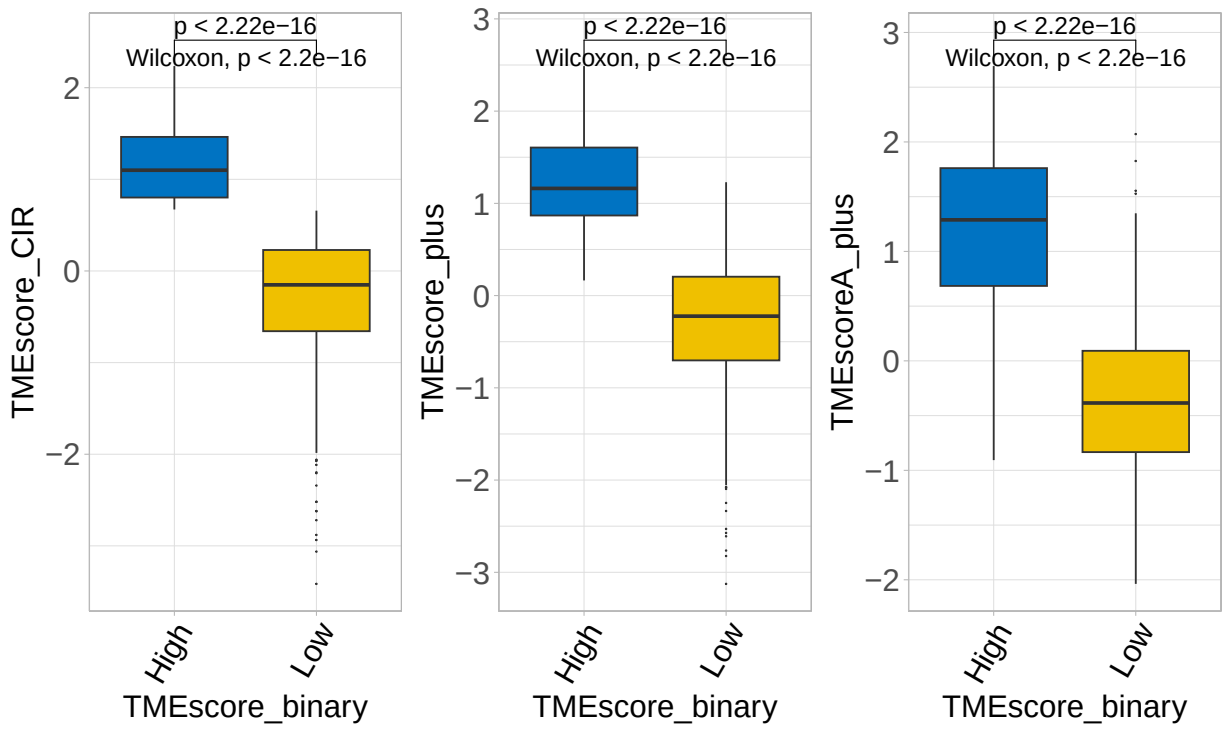
```
              variable       = "TMEscore_binary",
              jitter         = FALSE,
              cols           = NULL,
              palette        = "jco",
              show_pvalue    = TRUE,
              angle_x_text   = 60,
              hjust          = 1,
              size_of_pvalue = 5,
              size_of_font   = 8)
```

```
## # A tibble: 1 x 8
##   .y.       group1 group2        p p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl> <dbl> <chr>    <chr>    <chr>
## 1 signature High   Low    3.97e-34 4e-34 <2e-16   ****     Wilcoxon
```

```
p3 <- sig_box(data           = input,
              signature      = res$sig_names[3],
              variable       = "TMEscore_binary",
              jitter         = FALSE,
              cols           = NULL,
              palette        = "jco",
              show_pvalue    = TRUE,
              angle_x_text   = 60,
              hjust          = 1,
              size_of_pvalue = 5,
              size_of_font   = 8)
```

```
## # A tibble: 1 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature High   Low    1.68e-25 1.70e-25 <2e-16   ****     Wilcoxon
```

```
p1|p2|p3
```



6.10.3 For multicategorical variables (>2 subgroups)

```
res <- batch_kruskal(data = input, group = "Subtype", feature = colnames(input)[69:ncol
```

```
##
##       EMT       MSI MSS/TP53- MSS/TP53+
##        46        68       107        79
```

```
head(res)
```

```
## # A tibble: 6 x 10
##   sig_names        p.value   EMT    MSI `MSS/TP53-` `MSS/TP53+`    mean    p.adj
##   <chr>              <dbl> <dbl>  <dbl>       <dbl>       <dbl>   <dbl>    <dbl>
## 1 TMEscore_CIR    1.35e-28 -1.36  1.00        0.305      0.0577 -0.119  3.46e-26
## 2 Ether_Lipid_Me~ 4.37e-27  1.46 -0.830      -0.253     -0.375   0.165  4.64e-25
## 3 TMEscoreB_CIR   5.88e-27  1.55 -0.829      -0.420     -0.303   0.169  4.64e-25
## 4 Inositol_Phosp~ 7.25e-27  1.53 -0.808      -0.315     -0.408   0.177  4.64e-25
## 5 Selenocompound~ 1.17e-26 -1.48  0.824       0.328      0.326  -0.163  5.99e-25
## 6 Folate_biosynt~ 1.63e-26 -1.12  1.05        0.127     -0.0573 -0.0792 6.15e-25
## # i 2 more variables: log10pvalue <dbl>, stars <fct>
```

```
p1 <- sig_box(data           = input,
              signature      = res$sig_names[1],
              variable       = "Subtype",
              jitter         = FALSE,
              cols           =  NULL,
              palette        = "jco",
              show_pvalue    = TRUE,
              size_of_pvalue = 5,
              hjust          = 1,
              angle_x_text   = 60,
              size_of_font   = 8)
```

```
## # A tibble: 6 x 8
##   .y.       group1    group2          p    p.adj p.format p.signif method
##   <chr>     <chr>     <chr>       <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature EMT       MSI       3.64e-17 2.20e-16 < 2e-16  ****     Wilcoxon
## 2 signature EMT       MSS/TP53- 1.08e-13 3.20e-13 1.1e-13  ****     Wilcoxon
## 3 signature EMT       MSS/TP53+ 2.64e-14 1.10e-13 2.6e-14  ****     Wilcoxon
## 4 signature MSI       MSS/TP53- 1.27e-15 6.40e-15 1.3e-15  ****     Wilcoxon
## 5 signature MSI       MSS/TP53+ 5.96e- 9 1.20e- 8 6.0e-09  ****     Wilcoxon
## 6 signature MSS/TP53- MSS/TP53+ 7.71e- 3 7.7 e- 3 0.0077   **       Wilcoxon
```

```
p2 <- sig_box(data           = input,
              signature      = res$sig_names[2],
              variable       = "Subtype",
              jitter         = FALSE,
              cols           = NULL,
              palette        = "jco",
              show_pvalue    = TRUE,
              angle_x_text   = 60,
              hjust          = 1,
              size_of_pvalue = 5,
              size_of_font   = 8)
```

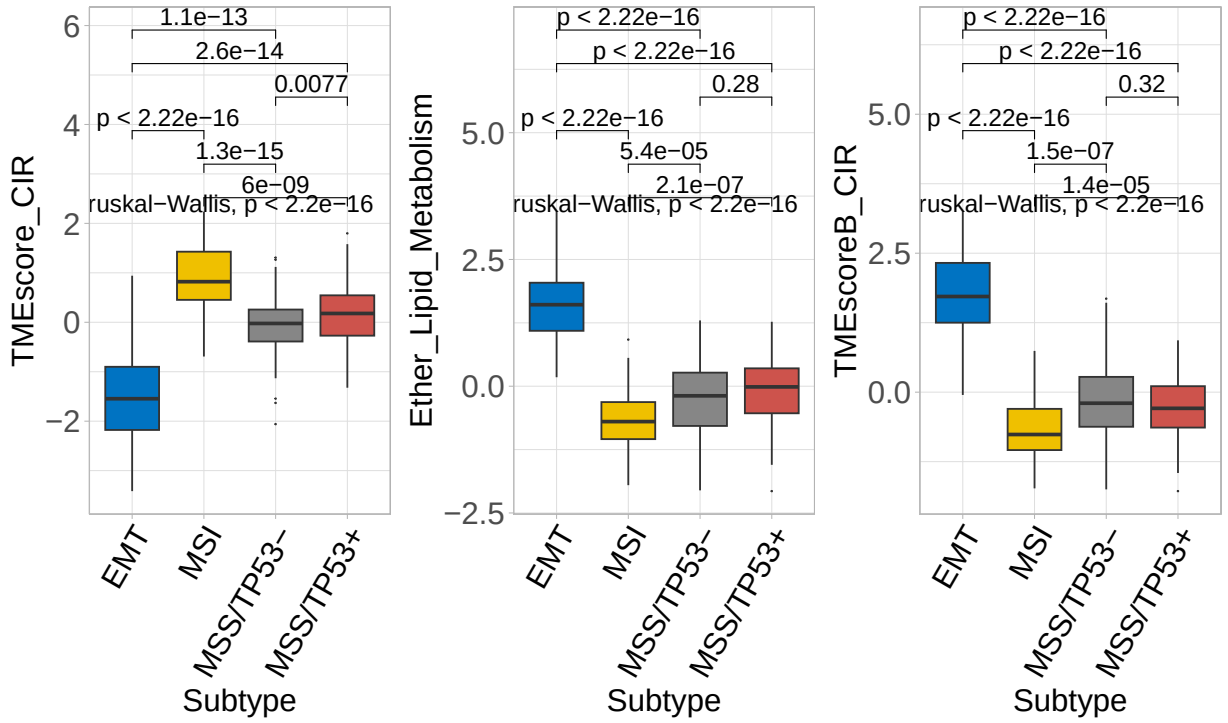
```
## # A tibble: 6 x 8
##   .y.       group1    group2          p    p.adj p.format p.signif method
##   <chr>     <chr>     <chr>       <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature EMT       MSI       3.76e-19 1.9 e-18 < 2e-16  ****     Wilcoxon
```

```
## 2 signature EMT       MSS/TP53- 4.26e-20 2.6 e-19 < 2e-16  ****     Wilcoxon
## 3 signature EMT       MSS/TP53+ 5.19e-18 2.10e-17 < 2e-16  ****     Wilcoxon
## 4 signature MSI       MSS/TP53- 5.43e- 5 1.1 e- 4 5.4e-05  ****     Wilcoxon
## 5 signature MSI       MSS/TP53+ 2.12e- 7 6.40e- 7 2.1e-07  ****     Wilcoxon
## 6 signature MSS/TP53- MSS/TP53+ 2.84e- 1 2.8 e- 1 0.28     ns       Wilcoxon
```

```
p3 <- sig_box(data           = input,
              signature      = res$sig_names[3],
              variable       = "Subtype",
              jitter         = FALSE,
              cols           = NULL,
              palette        = "jco",
              show_pvalue    = TRUE,
              angle_x_text   = 60,
              hjust          = 1,
              size_of_pvalue = 5,
              size_of_font   = 8)
```

```
## # A tibble: 6 x 8
##   .y.       group1    group2           p    p.adj p.format p.signif method
##   <chr>     <chr>     <chr>        <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature EMT       MSI       9.59e-19 4.80e-18 < 2e-16  ****     Wilcoxon
## 2 signature EMT       MSS/TP53- 6.07e-19 3.60e-18 < 2e-16  ****     Wilcoxon
## 3 signature EMT       MSS/TP53+ 2.89e-18 1.20e-17 < 2e-16  ****     Wilcoxon
## 4 signature MSI       MSS/TP53- 1.48e- 7 4.50e- 7 1.5e-07  ****     Wilcoxon
## 5 signature MSI       MSS/TP53+ 1.44e- 5 2.90e- 5 1.4e-05  ****     Wilcoxon
## 6 signature MSS/TP53- MSS/TP53+ 3.17e- 1 3.2 e- 1 0.32     ns       Wilcoxon
```

```
p1|p2|p3
```



6.11 Reference

Cristescu, R., Lee, J., Nebozhyn, M. et al. Molecular analysis of gastric cancer identifies subtypes associated with distinct clinical outcomes. Nat Med 21, 449–456 (2015). https://doi.org/10.1038/nm.3850

Chapter 7

TME Interaction analysis

7.1 Loading packages

```
library(IOBR)
```

7.2 Downloading data for example

Obtaining data set from GEO Gastric cancer: GSE62254 using `GEOquery` R package.

```
if (!requireNamespace("GEOquery", quietly = TRUE))  BiocManager::install("GEOquery")
library("GEOquery")
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo<- getGEO(GEO     = "GSE62254", getGPL  = F, destdir = "./")
eset    <- eset_geo[[1]]
eset    <- exprs(eset)
eset[1:5,1:5]
```

```
##           GSM1523727 GSM1523728 GSM1523729 GSM1523744 GSM1523745
## 1007_s_at  3.2176645  3.0624323  3.0279131   2.921683  2.8456013
## 1053_at    2.4050109  2.4394879  2.2442708   2.345916  2.4328582
## 117_at     1.4933412  1.8067380  1.5959665   1.839822  1.8326058
## 121_at     2.1965561  2.2812181  2.1865556   2.258599  2.1874363
## 1255_g_at  0.8698382  0.9502466  0.8125414   1.012860  0.9441993
```

7.3 Gene Annotation: HGU133PLUS-2 (Affaymetrix)

```
# Conduct gene annotation using `anno_hug133plus2` file; If identical gene symbols exi

eset<-anno_eset(eset       = eset,
                annotation = anno_hug133plus2,
                symbol     = "symbol",
                probe      = "probe_id",
                method     = "mean")
eset[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
## LOC101928826   4.219303   4.219670   4.213252
```

7.4 TME deconvolution using CIBERSORT algorithm

```
cell <- deconvo_tme(eset = eset, method = "cibersort", arrays = TRUE, perm = 1000, absol
head(cell)
```

```
## # A tibble: 6 x 27
##   ID        B_cells_naive_CIBERS~1 B_cells_memory_CIBER~2 Plasma_cells_CIBERSORT
##   <chr>                      <dbl>                  <dbl>                   <dbl>
## 1 GSM15237~                0.00610                 0.0136                   0.149
## 2 GSM15237~                0                       0.0339                   0.0765
## 3 GSM15237~                0.00335                 0.0183                   0.0939
## 4 GSM15237~                0                       0.0594                   0.0773
## 5 GSM15237~                0                       0.00738                  0.109
## 6 GSM15237~                0.0118                  0.0115                   0.138
## # i abbreviated names: 1: B_cells_naive_CIBERSORT, 2: B_cells_memory_CIBERSORT
## # i 23 more variables: T_cells_CD8_CIBERSORT <dbl>,
## #   T_cells_CD4_naive_CIBERSORT <dbl>,
## #   T_cells_CD4_memory_resting_CIBERSORT <dbl>,
## #   T_cells_CD4_memory_activated_CIBERSORT <dbl>,
```

```
## #   T_cells_follicular_helper_CIBERSORT <dbl>,
## #   `T_cells_regulatory_(Tregs)_CIBERSORT` <dbl>, ...
```

7.5 Identifying TME patterns

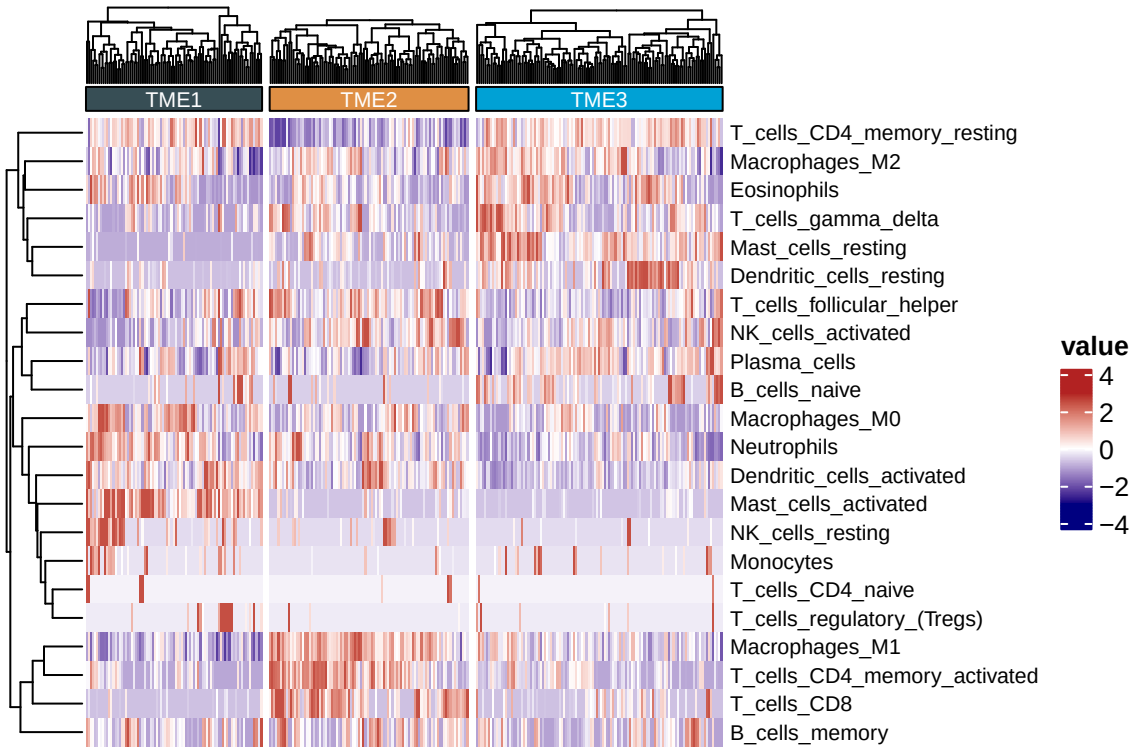
Identification of optimal clustering based on cellular infiltration patterns in the microenvironment.

```
tme <- tme_cluster(input = cell, features = colnames(cell)[2:23], id = "ID", scale = TRU
```

```
## [1] ">>>== Best number of TME clusters is: "
## Number_clusters     Value_Index
##          3.0000          2.7266
## [1] ">>>== Cluster of samples: "
## TME1 TME2 TME3
##   85   96  119
```

Use of heatmaps to reflect cellular differences between TME subtypes

```
colnames(tme) <- gsub(colnames(tme), pattern = "_CIBERSORT", replacement = "")
res <- sig_heatmap(input = tme, features = colnames(tme)[3:ncol(tme)], group = "cluster
```



7.6 Cell abundance of each cluster

```
cols <- c('#2692a4','#fc0d3a','#ffbe0b')
p1 <- sig_box(tme, variable = "cluster", signature = "Macrophages_M1", jitter = TRUE,
              cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   2.25e-17 4.50e-17 < 2e-16  ****     Wilcoxon
## 2 signature TME3   TME1   3.48e- 6 3.5 e- 6 3.5e-06  ****     Wilcoxon
## 3 signature TME2   TME1   6.50e-24 2   e-23 < 2e-16  ****     Wilcoxon
```

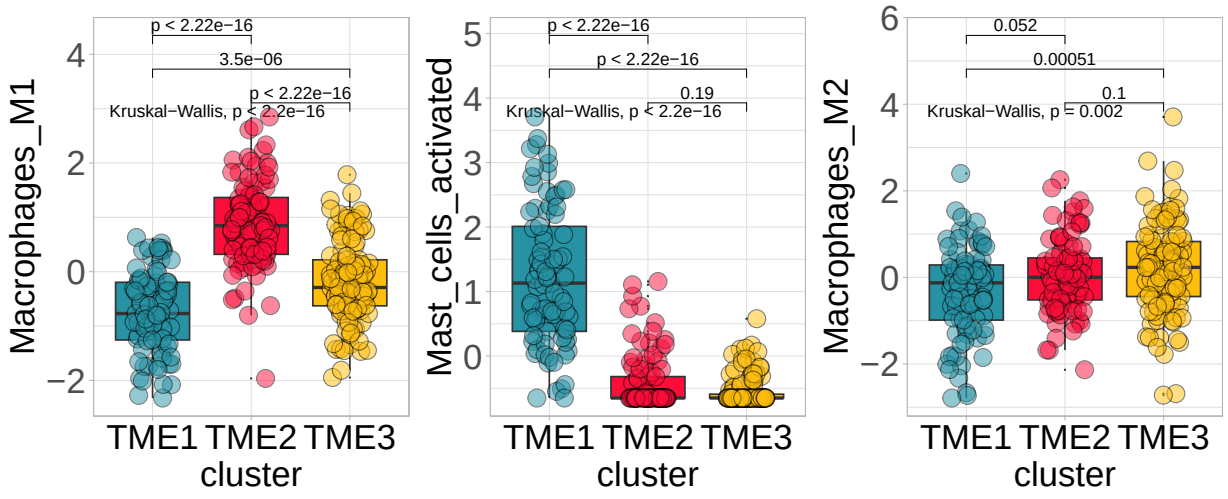
```
p2 <- sig_box(tme, variable = "cluster", signature = "Mast_cells_activated",
              jitter = TRUE, cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   1.89e- 1 1.9 e- 1 0.19     ns       Wilcoxon
## 2 signature TME3   TME1   6.89e-33 2.10e-32 <2e-16   ****     Wilcoxon
## 3 signature TME2   TME1   1.12e-25 2.20e-25 <2e-16   ****     Wilcoxon
```

```
p3 <- sig_box(tme, variable = "cluster", signature = "Macrophages_M2",
              jitter = TRUE, cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p  p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>  <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   0.101    0.1    0.10063  ns       Wilcoxon
## 2 signature TME3   TME1   0.000513 0.0015 0.00051  ***      Wilcoxon
## 3 signature TME2   TME1   0.0520   0.1    0.05203  ns       Wilcoxon
```

```
p1|p2|p3
```



7.7 DEG analysis between TME subtypes

Identifing TME subtype-related differential genes using `find_markers_in_bulk`.

We have developed a reliable classifier for the tumour microenvironment in gastric cancer using the same analysis pipelineTMEclassifier. The classifier was constructed by identifying the most robust gastric cancer TME classification through parsing the tumour microenvironment using the `tme_cluster` method. Next, genes specifically expressed by each microenvironmental subtype are obtained using the `find_markers_in_bulk method`. Finally, a machine learning approach was used to construct the classifier model.

```
library(Seurat)
res <- find_markers_in_bulk(pdata      = tme,
                            eset       = eset,
                            group      = "cluster",
                            nfeatures  = 2000,
                            top_n      = 50,
                            thresh.use = 0.15,
                            only.pos   = TRUE,
                            min.pct    = 0.10)
```

```
##
## TME3 TME2 TME1
##  119   96   85
## # A tibble: 150 x 7
## # Groups:   cluster [3]
```

```
##       p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##       <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 3.05e-22      0.896     1     1  6.63e-18 TME3    TMEM100
##  2 7.92e-22      1.13      1     1  1.72e-17 TME3    ADH1B
##  3 1.61e-20      0.691     1     1  3.51e-16 TME3    HHIP
##  4 1.93e-20      0.985     1     1  4.19e-16 TME3    ABCA8
##  5 5.73e-20      0.701     1     1  1.25e-15 TME3    FCER1A
##  6 9.42e-19      0.927     1     1  2.05e-14 TME3    MAMDC2
##  7 1.61e-18      0.773     1     1  3.49e-14 TME3    C1QTNF7
##  8 1.77e-18      0.718     1     1  3.85e-14 TME3    C16orf89
##  9 3.91e-18      0.729     1     1  8.51e-14 TME3    FHL1
## 10 5.87e-18      0.684     1     1  1.28e-13 TME3    ITGA8
## # i 140 more rows
```

```
top15 <-  res$top_markers %>% dplyr:: group_by(cluster) %>%  dplyr::top_n(15, avg_log2F
top15$gene
```

```
##  [1] "TMEM100"        "ADH1B"          "ABCA8"          "MAMDC2"
##  [5] "SCN7A"          "LIPF"           "C7"             "C2orf40"
##  [9] "PGA4"           "OGN"            "GKN2"           "GHRL"
## [13] "C6orf58"        "SCRG1"          "GIF"            "IFNG"
## [17] "WARS"           "CXCL10"         "IDO1"           "GZMB"
## [21] "CXCL11"         "GBP4"           "CXCL9"          "GNLY"
## [25] "GBP5"           "AIM2"           "RTEL1-TNFRSF6B" "COL11A1"
## [29] "S100A2"         "SLCO1B3"        "IL1A"           "IL1B"
## [33] "PPBP"           "IL11"           "CXCL6"          "CCL3L3"
## [37] "TREM1"          "PROK2"          "IL24"           "PI15"
## [41] "HCAR3"          "CLEC5A"         "MAGEA6"         "MAGEA12"
## [45] "REG1B"
```

Heatmap visualisation using `Seurat`'s `DoHeatmap`

```
#
cols <- c('#2692a4','#fc0d3a','#ffbe0b')
p1 <- DoHeatmap(res$sce, top15$gene, group.colors = cols )+
  scale_fill_gradientn(colours = rev(colorRampPalette(RColorBrewer::brewer.pal(11,"RdBu
```

Extracting variables from the expression matrix to merge with TME subtype

```
input <- combine_pd_eset(eset = eset, pdata = tme, feas = top15$gene, scale = T)
p2 <- sig_box(input, variable = "cluster", signature = "IFNG", jitter = TRUE,
              cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

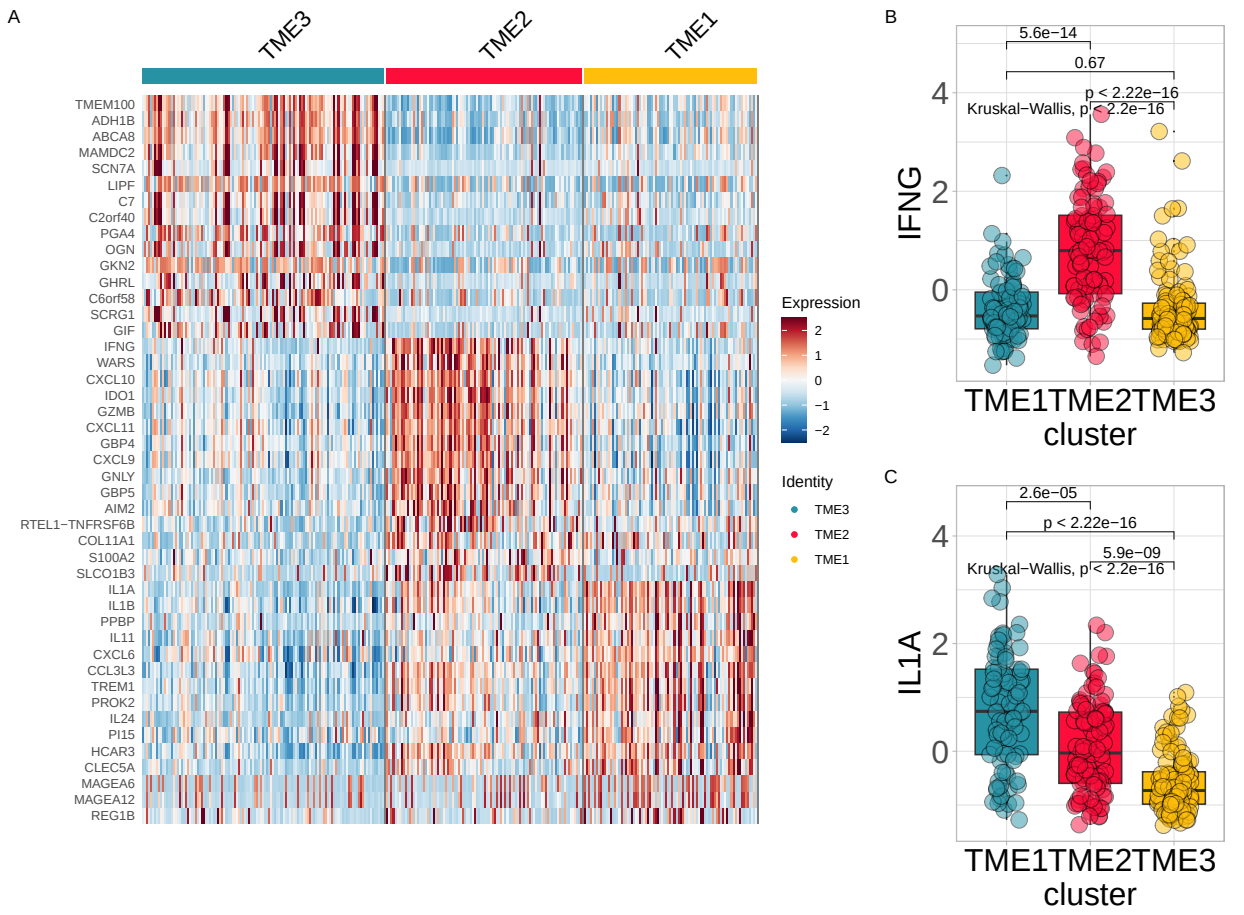
```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   1.11e-16 3.30e-16 < 2e-16  ****     Wilcoxon
## 2 signature TME3   TME1   6.70e- 1 6.7 e- 1 0.67     ns       Wilcoxon
## 3 signature TME2   TME1   5.60e-14 1.10e-13 5.6e-14  ****     Wilcoxon
```

```
p3 <- sig_box(input, variable = "cluster", signature = "IL1A",
              jitter = TRUE, cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   5.94e- 9 1.20e- 8 5.9e-09  ****     Wilcoxon
## 2 signature TME3   TME1   7.96e-18 2.40e-17 < 2e-16  ****     Wilcoxon
## 3 signature TME2   TME1   2.60e- 5 2.6 e- 5 2.6e-05  ****     Wilcoxon
```

Combining the results obtained above

```
# if (!requireNamespace("patchwork", quietly = TRUE))   install.packages("patchwork")
library(patchwork)
p <- (p1|p2/p3) + plot_layout(widths = c(2.3,1))
p + plot_annotation(tag_levels = 'A')
```



7.8 Identifying signatures associated with TME clusters

Calculate TME associated signatures-(through PCA method).

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = signature_collection,
                             method          = "pca",
                             mini_gene_count = 2)
sig_tme <- t(column_to_rownames(sig_tme, var = "ID"))
sig_tme[1:5, 1:3]
```

```
##                  GSM1523727 GSM1523728 GSM1523729
## CD_8_T_effector  -2.5513794  0.7789141 -2.1770675
## DDR              -0.8747614  0.7425162 -1.3272054
```

```
## APM                1.1098368  2.1988688 -0.9516419
## Immune_Checkpoint -2.3701787  0.9455120 -1.4844104
## CellCycle_Reg      0.1063358  0.7583302 -0.3649795
```

Finding characteristic variables associated with TME clusters

```
res <- find_markers_in_bulk(pdata = tme, eset = sig_tme, group = "cluster", nfeatures =
```

```
##
## TME3 TME2 TME1
##  119   96   85
## # A tibble: 59 x 7
## # Groups:   cluster [3]
##       p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##       <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 1.05e-25       5.03 0.832 0.287  2.70e-23 TME3    Glycolysis
##  2 1.15e-23       3.76 0.79  0.238  2.93e-21 TME3    Tyrosine-Metabolism
##  3 8.38e-18       4.07 0.756 0.32   2.15e-15 TME3    Drug-Metabolism-by-Cytochr~
##  4 8.59e-14       4.10 0.689 0.359  2.20e-11 TME3    Retinol-Metabolism
##  5 2.59e-13       3.55 0.723 0.348  6.64e-11 TME3    Metabolism-of-Xenobiotics-~
##  6 5.99e-11      10.0  0.546 0.227  1.53e- 8 TME3    detox.iCAF
##  7 7.25e-11      10.6  0.571 0.26   1.86e- 8 TME3    Normal.Fibroblast
##  8 2.32e-10       3.71 0.664 0.343  5.94e- 8 TME3    Ether-Lipid-Metabolism
##  9 1.99e- 9       5.12 0.555 0.276  5.10e- 7 TME3    TMEscoreB-CIR
## 10 2.23e- 8       3.43 0.664 0.387  5.71e- 6 TME3    Drug-Metabolism-by-other-e~
## # i 49 more rows
```

```
top15 <-  res$top_markers %>% dplyr:: group_by(cluster) %>%  dplyr::top_n(15, avg_log2F

p1 <- DoHeatmap(res$sce, top15$gene, group.colors = cols)+
  scale_fill_gradientn(colours = rev(colorRampPalette(RColorBrewer::brewer.pal(11,"RdBu
```

```
top15$gene  <- gsub(top15$gene, pattern = "-", replacement = "\\_")
input <- combine_pd_eset(eset = sig_tme, pdata = tme, feas = top15$gene, scale = T)

p2 <- sig_box(input, variable = "cluster", signature = "CD_8_T_effector", jitter = TRUE,
              cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4, size_of_font = 6)
```

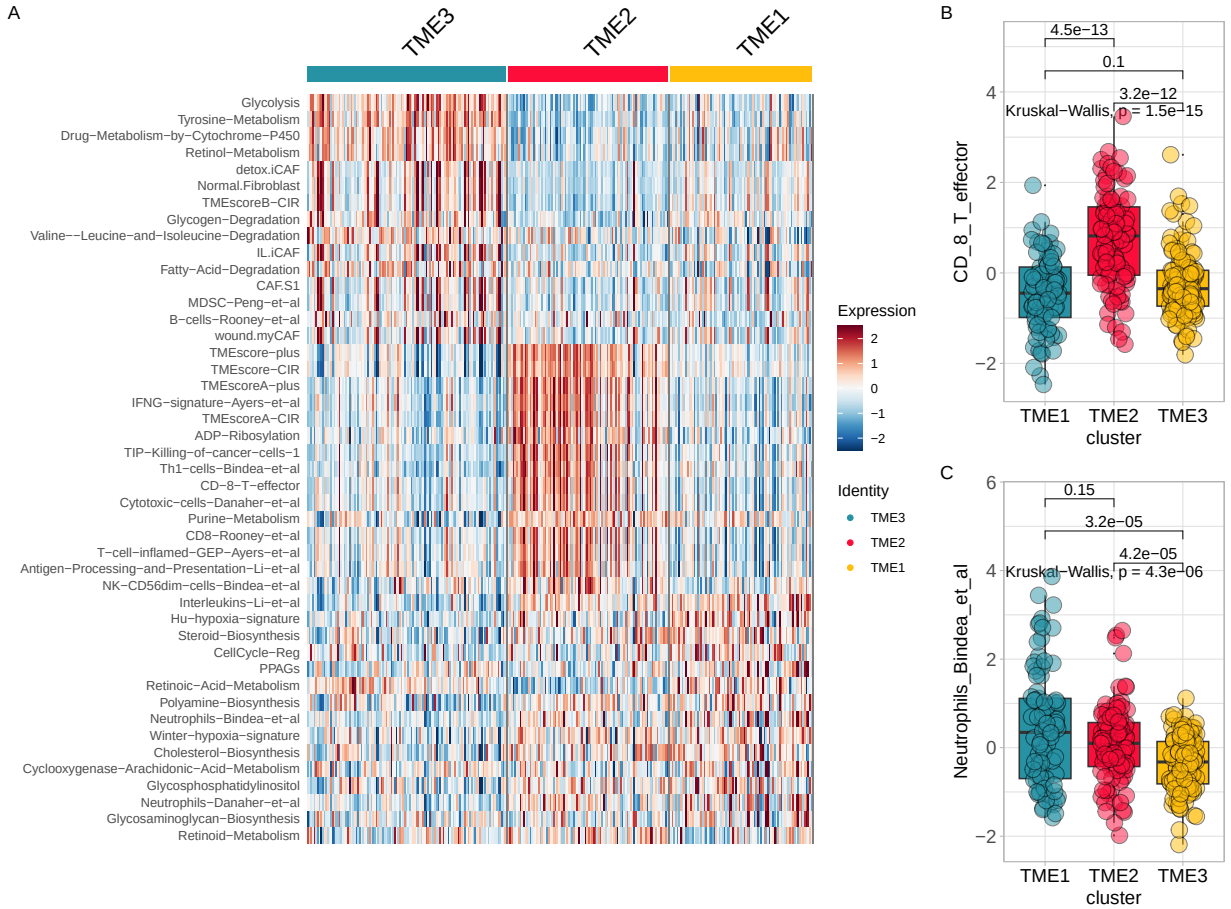
```
## # A tibble: 3 x 8
```

```
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   3.18e-12 6.40e-12 3.2e-12  ****     Wilcoxon
## 2 signature TME3   TME1   1.01e- 1 1    e- 1 0.1      ns       Wilcoxon
## 3 signature TME2   TME1   4.53e-13 1.4 e-12 4.5e-13  ****     Wilcoxon
```

```
p3 <- sig_box(input, variable = "cluster", signature = "Neutrophils_Bindea_et_al",
              jitter = TRUE, cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4, size_
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature TME3   TME2   0.0000416 0.000097 4.2e-05  ****     Wilcoxon
## 2 signature TME3   TME1   0.0000323 0.000097 3.2e-05  ****     Wilcoxon
## 3 signature TME2   TME1   0.149     0.15     0.15     ns       Wilcoxon
```

```
p <- (p1|p2/p3) + plot_layout(widths = c(2.3,1))
p + plot_annotation(tag_levels = 'A')
```



Survival differences between tumour microenvironment subtypes

```
library(survminer)
data(pdata_acrg, package = "IOBR")
input <- merge(pdata_acrg, input, by = "ID")
p1<-surv_group(input_pdata       = input,
               target_group      = "cluster",
               ID                = "ID",
               reference_group   = "High",
               project           = "ACRG",
               cols              = cols,
               time              = "OS_time",
               status            = "OS_status",
               time_type         = "month",
               save_path         = "result")
```

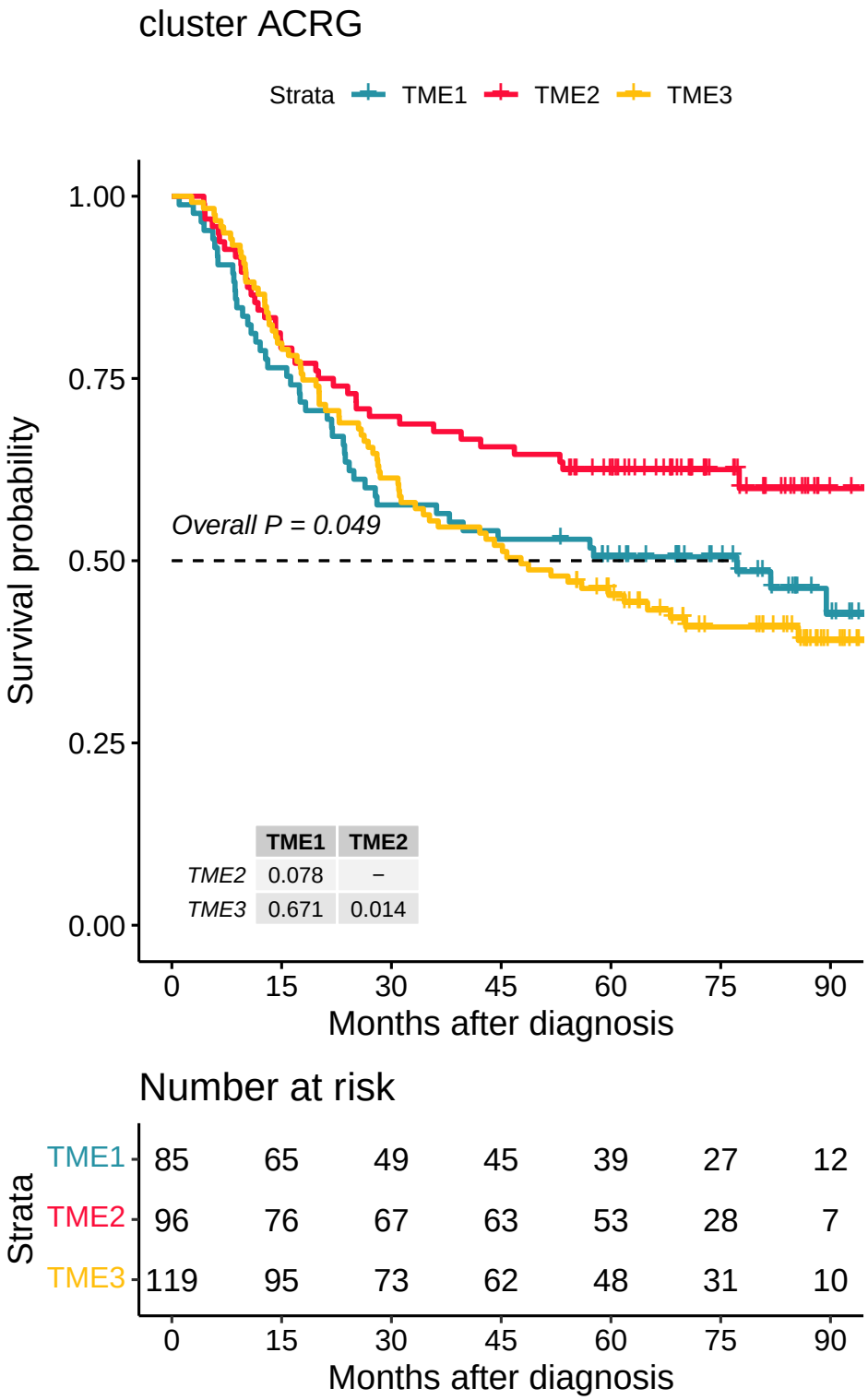
```
## >>> Dataset's survival follow up time is range between 1 to 105.7 months
```

```
## TME1 TME2 TME3
##   85   96  119
```

```
## 8596119
```

```
##   Maximum of follow up time is 105.7 months; and will be divided into 6 sections;
```

```
p1
```



Relationship between tumour microenvironmental subtypes and other subtypes

```
p1<- percent_bar_plot(input, x = "cluster" , y = "Subtype", palette = "jama", axis_angle
```

```
## # A tibble: 12 x 5
## # Groups:   cluster [3]
##    cluster Subtype    Freq  Prop count
##    <chr>   <fct>     <dbl> <dbl> <dbl>
##  1 TME1    EMT          14  0.16    85
##  2 TME1    MSI          12  0.14    85
##  3 TME1    MSS/TP53-    34  0.4     85
##  4 TME1    MSS/TP53+    25  0.29    85
##  5 TME2    EMT           6  0.06    96
##  6 TME2    MSI          47  0.49    96
##  7 TME2    MSS/TP53-    22  0.23    96
##  8 TME2    MSS/TP53+    21  0.22    96
##  9 TME3    EMT          26  0.22   119
## 10 TME3    MSI           9  0.08   119
## 11 TME3    MSS/TP53-    51  0.43   119
## 12 TME3    MSS/TP53+    33  0.28   119
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```

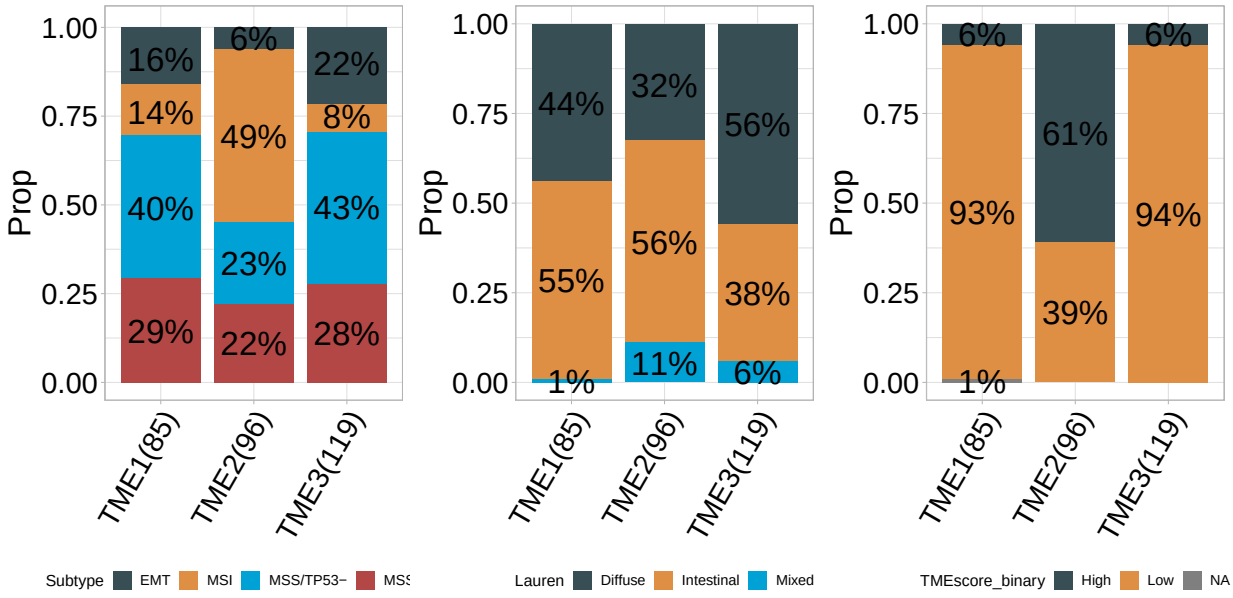
```
p2<- percent_bar_plot(input, x = "cluster" , y = "Lauren", palette = "jama", axis_angle
```

```
## # A tibble: 9 x 5
## # Groups:   cluster [3]
##   cluster Lauren      Freq  Prop count
##   <chr>   <fct>      <dbl> <dbl> <dbl>
## 1 TME1    Diffuse       37  0.44    85
## 2 TME1    Intestinal    47  0.55    85
## 3 TME1    Mixed          1  0.01    85
## 4 TME2    Diffuse       31  0.32    96
## 5 TME2    Intestinal    54  0.56    96
## 6 TME2    Mixed         11  0.11    96
## 7 TME3    Diffuse       67  0.56   119
## 8 TME3    Intestinal    45  0.38   119
## 9 TME3    Mixed          7  0.06   119
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```

```
p3<- percent_bar_plot(input, x = "cluster" , y = "TMEscore_binary", palette = "jama", ax
```

```
## # A tibble: 7 x 5
## # Groups:   cluster [3]
##   cluster TMEscore_binary  Freq  Prop count
##   <chr>   <fct>           <dbl> <dbl> <dbl>
## 1 TME1    High                5  0.06    85
## 2 TME1    Low                79  0.93    85
## 3 TME1    <NA>               1  0.01    85
## 4 TME2    High               59  0.61    96
## 5 TME2    Low                37  0.39    96
## 6 TME3    High                7  0.06   119
## 7 TME3    Low               112  0.94   119
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```

```
p1|p2|p3
```



7.9 References

Cristescu, R., Lee, J., Nebozhyn, M. et al. Molecular analysis of gastric cancer identifies subtypes associated with distinct clinical outcomes. Nat Med 21, 449–456 (2015). https://doi.org/10.1038/nm.3850

CIBERSORT; Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., … Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles. Nature Methods, 12(5), 453–457. https://doi.org/10.1038/nmeth.3337;

Seurat: Hao and Hao et al. Integrated analysis of multimodal single-cell data. Cell (2021)

Chapter 8

Tumor ecosystem analysis

8.1 Loading packages

```
library(IOBR)
```

8.2 Downloading data for example

Obtaining data set from GEO Gastric cancer: GSE62254 using `GEOquery` R package.

```
if (!requireNamespace("GEOquery", quietly = TRUE))  BiocManager::install("GEOquery")
library("GEOquery")
# NOTE: This process may take a few minutes which depends on the internet connection s
eset_geo<-getGEO(GEO     = "GSE62254", getGPL  = F, destdir = "./")
eset    <-eset_geo[[1]]
eset    <-exprs(eset)
eset[1:5,1:5]
```

```
##           GSM1523727 GSM1523728 GSM1523729 GSM1523744 GSM1523745
## 1007_s_at  3.2176645  3.0624323  3.0279131   2.921683  2.8456013
## 1053_at    2.4050109  2.4394879  2.2442708   2.345916  2.4328582
## 117_at     1.4933412  1.8067380  1.5959665   1.839822  1.8326058
## 121_at     2.1965561  2.2812181  2.1865556   2.258599  2.1874363
## 1255_g_at  0.8698382  0.9502466  0.8125414   1.012860  0.9441993
```

8.3 Gene Annotation: HGU133PLUS-2 (Affaymetrix)

```
# Conduct gene annotation using `anno_hug133plus2` file; If identical gene symbols exi

eset<-anno_eset(eset       = eset,
                annotation = anno_hug133plus2,
                symbol     = "symbol",
                probe      = "probe_id",
                method     = "mean")
eset[1:5, 1:3]
```

```
##              GSM1523727 GSM1523728 GSM1523729
## SH3KBP1        4.327974   4.316195   4.351425
## RPL41          4.246149   4.246808   4.257940
## EEF1A1         4.293762   4.291038   4.262199
## COX2           4.250288   4.283714   4.270508
## LOC101928826   4.219303   4.219670   4.213252
```

8.4 Determine TME subtype of gastric cancer using TMEclassifier R package

```
if (!requireNamespace("TMEclassifier", quietly = TRUE)) devtools::install_github("LiaoW
library(TMEclassifier)
tme <- tme_classifier(eset = eset, scale = TRUE)
```

```
## Step-1: Expression data preprocessing...
## Step-2: TME deconvolution...
## Step-3: Predicting TME phenotypes...
## [21:37:33] WARNING: src/learner.cc:1203:
##   If you are loading a serialized model (like pickle in Python, RDS in R) generated b
##   older XGBoost, please export the model by calling `Booster.save_model` from that ve
##   first, then load it back in current version. See:
##
##     https://xgboost.readthedocs.io/en/latest/tutorials/saving_model.html
##
##   for more details about differences between saving model and serializing.
##
```

```
## [21:37:33] WARNING: src/learner.cc:888: Found JSON model saved before XGBoost 1.6, pl
## [21:37:33] WARNING: src/learner.cc:553:
##   If you are loading a serialized model (like pickle in Python, RDS in R) generated b
##   older XGBoost, please export the model by calling `Booster.save_model` from that ve
##   first, then load it back in current version. See:
##
##     https://xgboost.readthedocs.io/en/latest/tutorials/saving_model.html
##
##   for more details about differences between saving model and serializing.
##
## >>>--- DONE!
```

```
table(tme$TMEcluster)
```

```
##
##  IA  IE  IS
## 107  96  97
```

```
head(tme)
```

```
##           ID          IE         IS         IA TMEcluster
## 1 GSM1523727 0.204623557 0.11212681 0.68324962         IA
## 2 GSM1523728 0.009599504 0.11179146 0.87860903         IA
## 3 GSM1523729 0.852615046 0.11369089 0.03369407         IE
## 4 GSM1523744 0.053842233 0.06994632 0.87621145         IA
## 5 GSM1523745 0.055973019 0.80839488 0.13563209         IS
## 6 GSM1523746 0.545343299 0.37437568 0.08028102         IE
```

```
table(tme$TMEcluster)
```

```
##
##  IA  IE  IS
## 107  96  97
```

```
head(tme)
```

```
##           ID          IE         IS         IA TMEcluster
## 1 GSM1523727 0.204623557 0.11212681 0.68324962         IA
## 2 GSM1523728 0.009599504 0.11179146 0.87860903         IA
## 3 GSM1523729 0.852615046 0.11369089 0.03369407         IE
## 4 GSM1523744 0.053842233 0.06994632 0.87621145         IA
```

```
## 5 GSM1523745 0.055973019 0.80839488 0.13563209         IS
## 6 GSM1523746 0.545343299 0.37437568 0.08028102         IE
```

8.5 DEG analysis: method1

Differential analysis of selected immune-activated and immune-expelled gastric cancers

```
pdata <- tme[!tme$TMEcluster=="IS", ]
deg  <-   iobr_deg(eset          = eset,
                   annotation    = NULL,
                   pdata         = pdata,
                   group_id      = "TMEcluster",
                   pdata_id      = "ID",
                   array         = TRUE,
                   method        = "limma",
                   contrast      = c("IA","IE"),
                   path          = "result",
                   padj_cutoff   = 0.01,
                   logfc_cutoff = 0.5)
```

```
## >>>== Matching grouping information and expression matrix

## >>>== limma was selected for differential gene analysis of Array data

## Warning: package 'limma' was built under R version 4.2.1

##
## Attaching package: 'limma'

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

## group1 = IE

## group2 = NA

## # A tibble: 6 x 11
##   symbol  log2FoldChange AveExpr     t   pvalue     padj     B sigORnot    label
##   <chr>            <dbl>   <dbl> <dbl>    <dbl>    <dbl> <dbl> <chr>       <chr>
## 1 TMEM100          0.774    1.84  13.9 2.47e-31 5.37e-27  60.4 Up_regulat~ Both
## 2 ABCA8            0.933    1.90  12.9 3.11e-28 3.38e-24  53.4 Up_regulat~ Both
```

```
## 3 HHIP             0.613    1.73  12.1 7.62e-26 4.46e-22  48.0 Up_regulat~ Both
## 4 LMNB2           -0.287    2.25 -12.1 9.28e-26 4.46e-22  47.8 NOT         Sign~
## 5 MCM6            -0.211    3.02 -12.1 1.02e-25 4.46e-22  47.7 NOT         Sign~
## 6 ADH1B            0.907    1.86  12.0 2.27e-25 7.04e-22  47.0 Up_regulat~ Both
## # i 2 more variables: IE <dbl>, `` <dbl>
```

8.6 GSEA analysis based on differential express gene analysis results

Select the gene set list in IOBR's signature collection.

```
head(deg)
```

```
## # A tibble: 6 x 11
##   symbol  log2FoldChange AveExpr     t   pvalue     padj     B sigORnot    label
##   <chr>            <dbl>   <dbl> <dbl>    <dbl>    <dbl> <dbl> <chr>       <chr>
## 1 TMEM100          0.774    1.84  13.9 2.47e-31 5.37e-27  60.4 Up_regulat~ Both
## 2 ABCA8            0.933    1.90  12.9 3.11e-28 3.38e-24  53.4 Up_regulat~ Both
## 3 HHIP             0.613    1.73  12.1 7.62e-26 4.46e-22  48.0 Up_regulat~ Both
## 4 LMNB2           -0.287    2.25 -12.1 9.28e-26 4.46e-22  47.8 NOT         Sign~
## 5 MCM6            -0.211    3.02 -12.1 1.02e-25 4.46e-22  47.7 NOT         Sign~
## 6 ADH1B            0.907    1.86  12.0 2.27e-25 7.04e-22  47.0 Up_regulat~ Both
## # i 2 more variables: IE <dbl>, `` <dbl>
```

```
sig_list <- signature_collection[c("TMEscoreB_CIR", "TMEscoreA_CIR", "DNA_replication",
                                   "Pan_F_TBRs", "TGFb.myCAF", "Ferroptosis", "TLS_Natur
sig_list
```

```
## $TMEscoreB_CIR
##   [1] "DCN"          "SEPP1"        "ACTA2"        "SPARCL1"      "BEX3"
##   [6] "MYLK"         "AKR1C1"       "TIMP2"        "MXRA7"        "C11orf96"
##  [11] "CAV1"         "PDGFRA"       "FHL1"         "MGP"          "EID1"
##  [16] "LOC101930400" "DST"          "GREM1"        "FERMT2"       "TNC"
##  [21] "CYBRD1"       "LTBP1"        "ACTG2"        "TMEM47"       "SERPINE2"
##  [26] "ANTXR2"       "GNG11"        "TAGLN"        "GSTA4"        "PKIG"
##  [31] "MAOA"         "PTRF"         "FAM3B"        "PBX1"         "WLS"
##  [36] "SELM"         "SVIL"         "MYH11"        "AGT"          "SPON1"
##  [41] "TGFB1I1"      "PDLIM3"       "PDK4"         "SYNPO2"       "MSRB3"
```

```
##  [46] "PROS1"         "EDNRA"         "AKAP12"       "PSD3"          "TNS1"
##  [51] "JAM3"          "PDZRN3"        "DDR2"         "HMGCS2"        "SGCE"
##  [56] "MRVI1"         "WFDC1"         "FBLN1"        "FMO5"          "MAOB"
##  [61] "AMOTL1"        "AKT3"          "CNRIP1"       "CPE"           "MAP1B"
##  [66] "RBP1"          "GNAI1"         "FOXF2"        "SORBS2"        "ZCCHC24"
##  [71] "ZNF704"        "ARMCX1"        "DIXDC1"       "SSTR1"         "THRB"
##  [76] "C3orf70"       "PKIB"          "CNN1"         "SYTL5"         "DACT1"
##  [81] "SYNPO"         "GAS1"          "DPYSL3"       "CCDC80"        "TSPYL5"
##  [86] "DCHS1"         "SOBP"          "AOC3"         "NDN"           "FGF7P3"
##  [91] "SMAD9"         "MCC"           "CLMP"         "MYL9"          "RBP4"
##  [96] "PLN"           "SPOCK1"        "COL14A1"      "CRYAB"         "SRPX"
## [101] "EML1"          "RERG"          "PPP1R3C"      "LOC100506718" "CH25H"
## [106] "HSPB8"         "PID1"          "TTC28"        "STON1"         "ABCG2"
## [111] "ZSCAN18"       "SCIN"          "C14orf132"    "TMEM55A"       "WASF3"
## [116] "PAPLN"         "COLEC12"       "ACKR1"        "TMEM150C"      "RAI2"
## [121] "TSPAN7"        "MRGPRF"        "ABCA8"        "CHIC1"         "NBEA"
## [126] "FAM13C"        "SETBP1"        "LDOC1"        "TMEM100"       "LOC101930349"
## [131] "PRICKLE2"      "TSPAN18"       "FABP4"        "ARHGEF26"      "ERICH5"
## [136] "MYOCD"         "BEX2"          "PPP1R14A"     "FGF13"         "RUNX1T1"
## [141] "MAGI2-AS3"     "LINC01279"     "REEP1"        "PLAC9"         "MYEF2"
## [146] "PRKD1"         "RGN"           "CLDN11"       "ANK2"          "ESRRG"
## [151] "SYNC"          "ZNF667-AS1"    "FGF7"         "SFRP1"         "HMCN1"
## [156] "TCEAL7"        "OGN"           "MAGI2"        "MIR100HG"      "FILIP1"
## [161] "LOC100507334" "ANKRD6"        "PLEKHH2"      "ZNF542P"       "ARMCX4"
## [166] "NOV"           "DCLK1"         "ARHGAP28"     "C2orf40"       "TRHDE"
## [171] "EPHA7"         "SCRG1"         "ZNF677"       "ZFPM2"         "PEG3"
## [176] "SERP2"         "ZNF415"        "MAMDC2"       "RBM24"         "MEOX2"
##
## $TMEscoreA_CIR
##  [1] "HLA-DPB1"       "UBD"           "LOC100509457"   "WARS"
##  [5] "TAP1"           "HLA-DMA"       "TRIM22"         "PSAT1"
##  [9] "CXCL10"         "SOCS3"         "CXCL9"          "PBK"
## [13] "CCL4"           "CCL5"          "BCL2A1"         "TRBC1"
## [17] "IDO1"           "NFE2L3"        "CCL3L3"         "DTL"
## [21] "MMP9"           "SLC2A3"        "ZNF367"         "RCC1"
## [25] "STIL"           "TRAC"          "HELLS"          "GZMB"
## [29] "RTEL1-TNFRSF6B" "CXCL11"        "GBP5"           "CD2"
```

```
## [33] "CDCA2"          "CDT1"           "TNFAIP2"        "TYMP"
## [37] "MICB"           "SLC2A14"        "GZMK"           "CD8A"
## [41] "CENPH"          "MND1"           "BATF2"          "BRIP1"
## [45] "E2F7"           "KIF18A"         "AIM2"           "ETV7"
## [49] "ITK"            "GNLY"           "GPR171"         "WDHD1"
## [53] "GBP4"           "MB21D1"         "NLRP3"          "MCEMP1"
## [57] "POLR3G"         "NLRC3"          "KLRC2"          "CLEC5A"
## [61] "ARHGAP11A"      "GPR84"          "IFNG"           "ZBED2"
##
## $DNA_replication
##  [1] "RNASEH2A" "POLD3"    "DNA2"     "FEN1"     "POLA2"    "RNASEH1"
##  [7] "RPA4"     "LIG1"     "MCM2"     "MCM3"     "MCM4"     "MCM5"
## [13] "MCM6"     "MCM7"     "PCNA"     "POLE3"    "POLA1"    "POLD1"
## [19] "POLD2"    "POLE"     "POLE2"    "PRIM1"    "PRIM2"    "POLE4"
## [25] "POLD4"    "RFC1"     "RFC2"     "RFC3"     "RFC4"     "RFC5"
## [31] "RPA1"     "RPA2"     "RPA3"     "SSBP1"    "RNASEH2B" "RNASEH2C"
##
## $Base_excision_repair
##  [1] "PARP2" "PARP3" "POLD3" "PARP1" "PARP4" "FEN1"  "SMUG1" "NEIL2" "APEX2"
## [10] "POLL"  "HMGB1" "APEX1" "LIG1"  "LIG3"  "MPG"   "MUTYH" "NTHL1" "OGG1"
## [19] "PCNA"  "POLE3" "POLB"  "POLD1" "POLD2" "POLE"  "POLE2" "NEIL3" "POLE4"
## [28] "POLD4" "UNG"   "XRCC1" "NEIL1" "MBD4"
##
## $Pan_F_TBRs
##  [1] "ACTA2"    "ACTG2"    "ADAM12"   "ADAM19"   "CNN1"     "COL4A1"
##  [7] "CTGF"     "CTPS1"    "FAM101B"  "FSTL3"    "HSPB1"    "IGFBP3"
## [13] "PXDC1"    "SEMA7A"   "SH3PXD2A" "TAGLN"    "TGFBI"    "TNS1"
## [19] "TPM1"
##
## $TGFb.myCAF
##  [1] "CST1"    "LAMP5"   "LOXL1"   "EDNRA"   "TGFB1"   "TGFB3"   "TNN"
##  [8] "CST2"    "HES4"    "COL10A1" "ELN"     "THBS4"   "NKD2"    "OLFM2"
## [15] "COL6A3"  "LRRC17"  "COL3A1"  "THY1"    "HTRA3"   "TMEM204" "11-Sep"
## [22] "COMP"    "TNFAIP6" "ID4"     "GGT5"    "INAFM1"  "CILP"    "OLFML2B"
##
## $Ferroptosis
##  [1] "ACSL4"      "AKR1C1-3"   "ALOXs"      "ATP5G3"     "CARS"
```

```
##  [6] "CBS"         "CD44v"       "CHAC1"       "CISD1"       "CS"
## [11] "DPP4"        "FANCD2"      "GCLC/GCLM"   "GLS2"        "GPX4"
## [16] "GSS"         "HMGCR"       "HSPB1/5"     "KOD"         "LPCAT3"
## [21] "MT1G"        "NCOA4"       "NFE2L2"      "PTGS2"       "RPL8"
## [26] "SAT1"        "SLC7A11"     "SQS"         "TFRC"        "TP53"
## [31] "TTC35/EMC2" "MESH1"
##
## $TLS_Nature
## [1] "CD79B"  "CD1D"   "CCR6"   "LAT"    "SKAP1"  "CETP"   "EIF1AY" "RBP5"
## [9] "PTGDS"
##
## $Glycolysis
##  [1] "ACSS1"   "ACSS2"   "ADH1A"   "ADH1B"   "ADH1C"   "ADH4"    "ADH5"
##  [8] "ADH6"    "ADH7"    "ADPGK"   "AKR1A1"  "ALDH1A3" "ALDH1B1" "ALDH2"
## [15] "ALDH3A1" "ALDH3A2" "ALDH3B1" "ALDH3B2" "ALDH7A1" "ALDH9A1" "ALDOA"
## [22] "ALDOB"   "ALDOC"   "BPGM"    "DLAT"    "DLD"     "ENO1"    "ENO2"
## [29] "ENO3"    "FBP1"    "FBP2"    "G6PC"    "G6PC2"   "GALM"    "GAPDH"
## [36] "GAPDHS"  "GCK"     "GPI"     "HK1"     "HK2"     "HK3"     "HKDC1"
## [43] "LDHA"    "LDHAL6A" "LDHAL6B" "LDHB"    "LDHC"    "PANK1"   "PCK1"
## [50] "PCK2"    "PDHA1"   "PDHA2"   "PDHB"    "PFKFB1"  "PFKFB2"  "PFKFB3"
## [57] "PFKFB4"  "PFKL"    "PFKM"    "PFKP"    "PGAM1"   "PGAM2"   "PGAM4"
## [64] "PGK1"    "PGK2"    "PGM1"    "PGM2"    "PKLR"    "PKM"     "SLC2A2"
## [71] "TPI1"
```

```
gsea<-     sig_gsea(deg,
                    genesets          = sig_list,
                    path              = "result",
                    gene_symbol       = "symbol",
                    logfc             = "log2FoldChange",
                    org               = "hsa",
                    show_plot         = FALSE,
                    msigdb            = TRUE,
                    category          = "H",
                    subcategory       = NULL,
                    palette_bar       = "set2")
```

Hallmark gene signatures

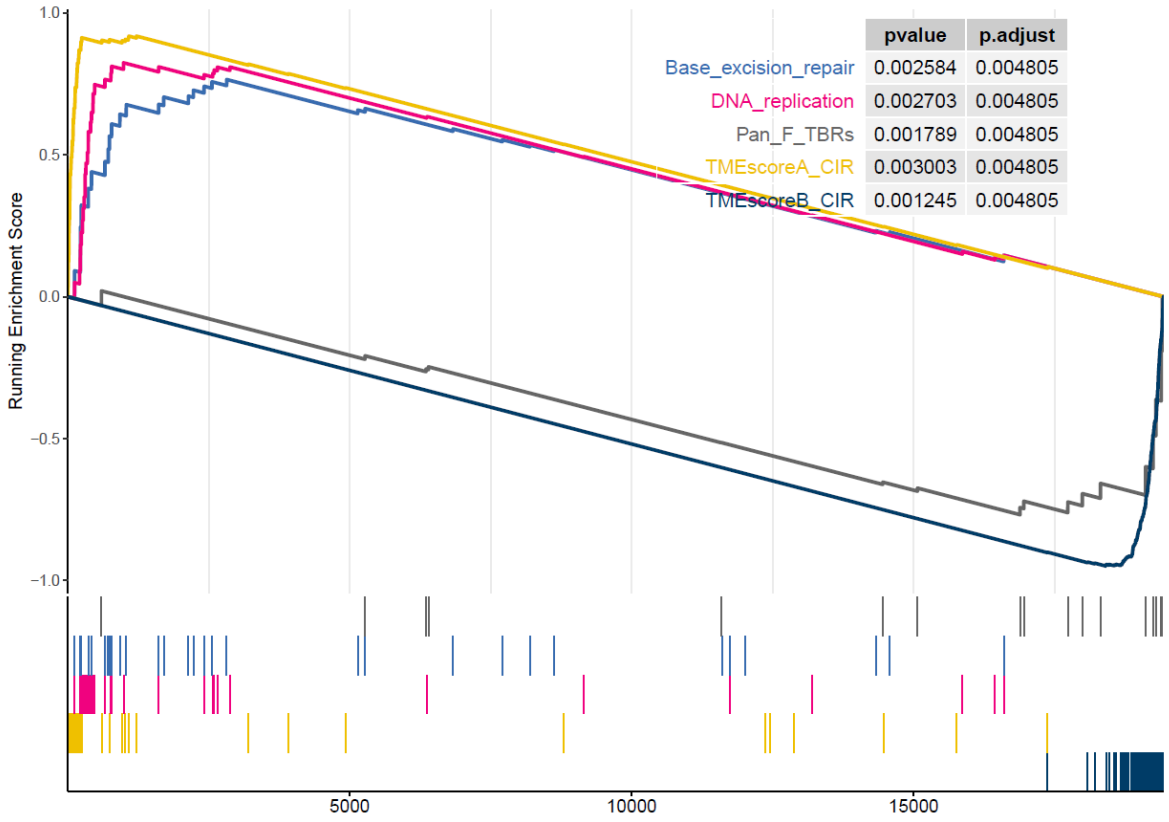


Figure 8.1: GSEA of TME gent sets

```
gsea<-     sig_gsea(deg,
                    genesets          = NULL,
                    path              = "GSEA",
                    gene_symbol       = "symbol",
                    logfc             = "log2FoldChange",
                    org               = "hsa",
                    show_plot         = FALSE,
                    msigdb            = TRUE,
                    category          = "H",
                    subcategory       = NULL,
                    palette_bar       = "aaas",
                    show_bar          = 5,
                    show_gsea         = 6)
```

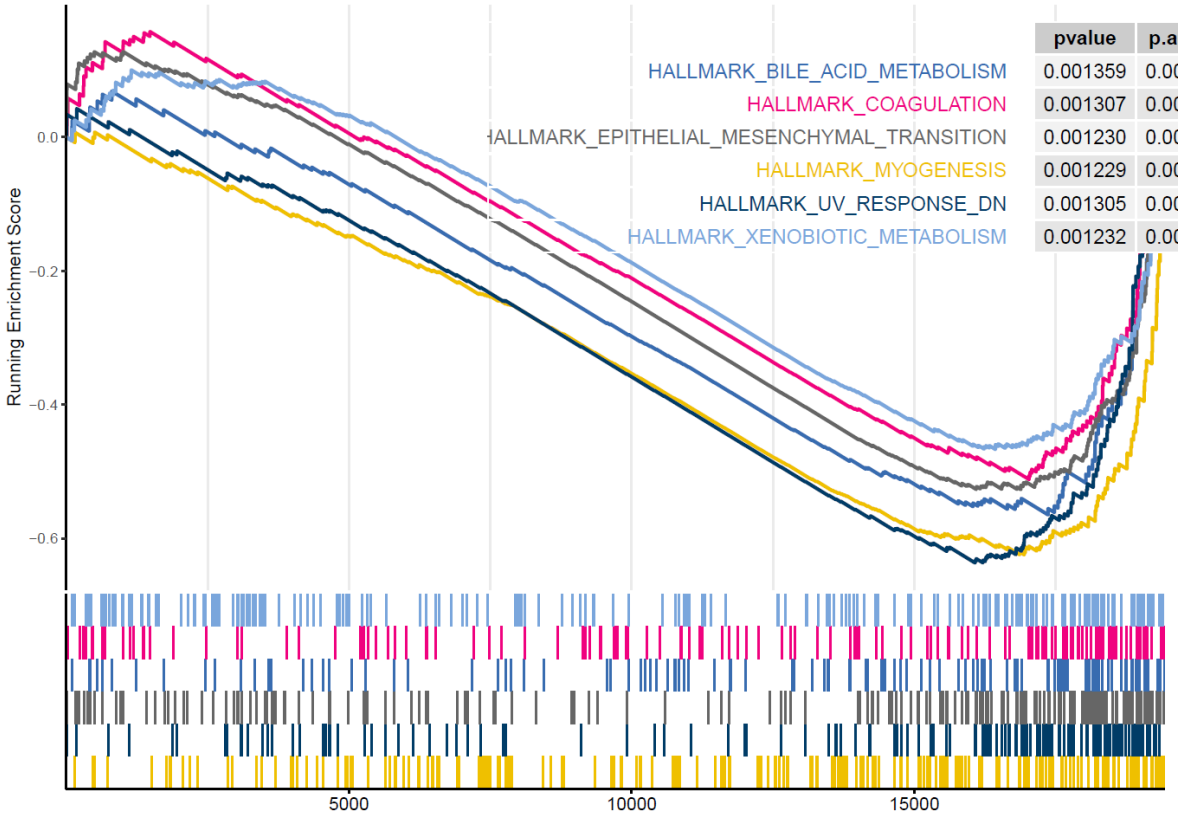


Figure 8.2: GSEA of Hallmark gent sets

8.7 DEG analysis: method2

Identifing TME subtype-related differential genes using `find_markers_in_bulk`

```
library(Seurat)
res <- find_markers_in_bulk(pdata      = tme,
                            eset       = eset,
                            group      = "TMEcluster",
                            nfeatures  = 2000,
                            top_n      = 50,
                            thresh.use = 0.15,
                            only.pos   = TRUE,
                            min.pct    = 0.10)
```

```
##
##  IA  IE  IS
## 107  96  97
## # A tibble: 150 x 7
## # Groups:   cluster [3]
##       p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##       <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 3.37e-22      0.410     1     1  7.34e-18 IA      TAP1
##  2 3.29e-20      0.632     1     1  7.15e-16 IA      IFNG
##  3 2.58e-19      0.380     1     1  5.61e-15 IA      ETV7
##  4 3.86e-19      0.403     1     1  8.39e-15 IA      MB21D1
##  5 1.81e-18      0.671     1     1  3.93e-14 IA      CXCL10
##  6 1.93e-17      0.421     1     1  4.20e-13 IA      MND1
##  7 3.23e-17      0.369     1     1  7.02e-13 IA      PSMB9
##  8 7.47e-17      0.378     1     1  1.62e-12 IA      CDT1
##  9 1.01e-16      0.655     1     1  2.20e-12 IA      GZMB
## 10 2.82e-16      0.817     1     1  6.12e-12 IA      CXCL11
## # i 140 more rows
```

```
top15 <-  res$top_markers %>% dplyr:: group_by(cluster) %>%  dplyr::top_n(15, avg_log2F
top15$gene
```

```
##  [1] "IFNG"          "CXCL10"        "GZMB"          "CXCL11"
##  [5] "CXCL9"         "WARS"          "IDO1"          "UBD"
##  [9] "GBP4"          "GNLY"          "KLRC2"         "GZMH"
```

```
## [13] "VSNL1"          "AIM2"           "SLCO1B3"        "ADH1B"
## [17] "ABCA8"          "MAMDC2"         "SCN7A"          "MYH11"
## [21] "C7"             "C2orf40"        "LIPF"           "PGA4"
## [25] "SCRG1"          "GHRL"           "CNN1"           "OGN"
## [29] "GIF"            "ATP4A"          "IL1A"           "EREG"
## [33] "PPBP"           "IL11"           "PI15"           "IL24"
## [37] "PROK2"          "HCAR3"          "RBP4"           "MAGEA10-MAGEA5"
## [41] "MAGEA4"         "MAGEA12"        "MAGEA6"         "MAGEA2B"
## [45] "REG1B"
```

Heatmap visualisation using Seurat's DoHeatmap

```
#
cols <- c('#2692a4','#fc0d3a','#ffbe0b')
p1 <- DoHeatmap(res$sce, top15$gene, group.colors = cols )+
  scale_fill_gradientn(colours = rev(colorRampPalette(RColorBrewer::brewer.pal(11,"RdBu
```

Extracting variables from the expression matrix to merge with TME subtypes

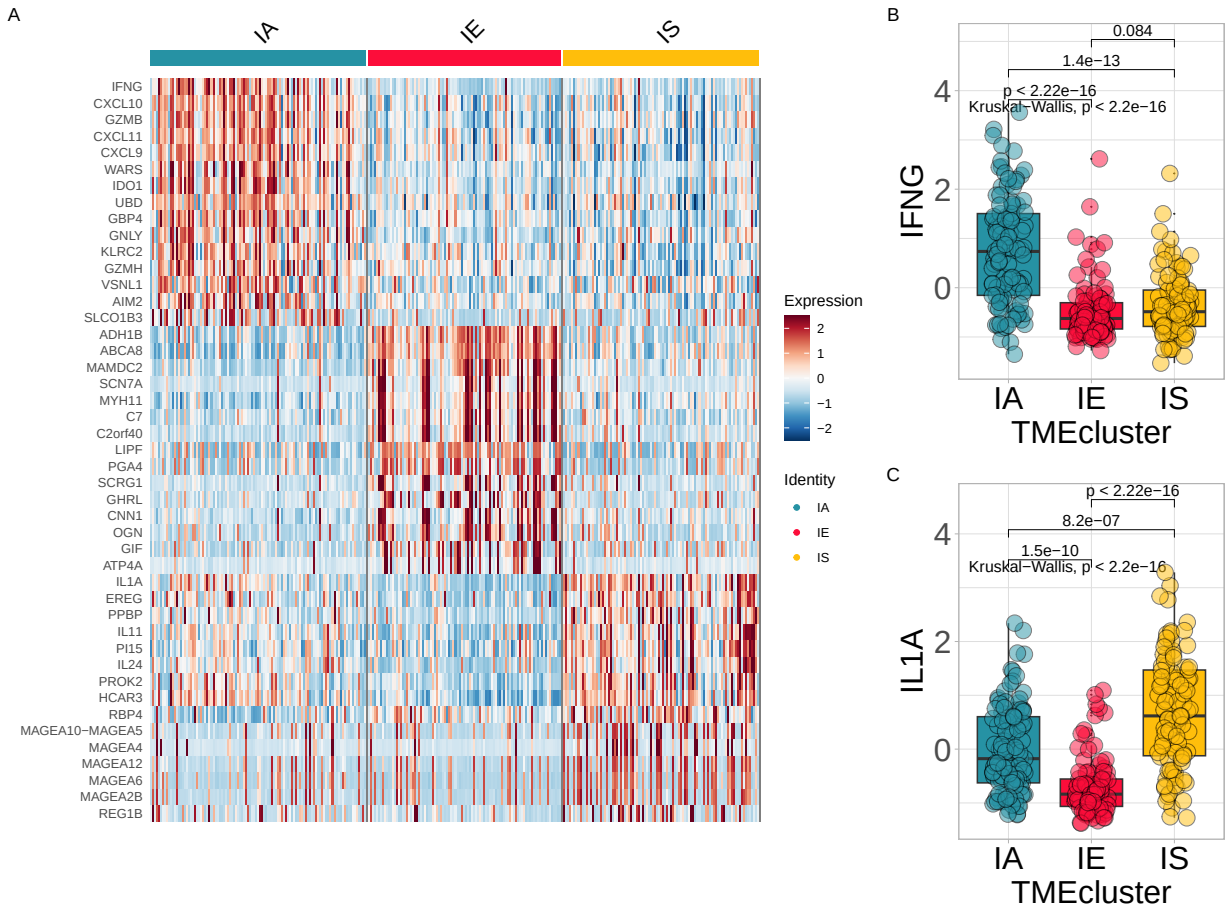
```
input <- combine_pd_eset(eset = eset, pdata = tme, feas = top15$gene, scale = T)
p2 <- sig_box(input, variable = "TMEcluster", signature = "IFNG", jitter = TRUE,
              cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature IA     IE     4.09e-17 1.20e-16 < 2e-16  ****     Wilcoxon
## 2 signature IA     IS     1.44e-13 2.90e-13 1.4e-13  ****     Wilcoxon
## 3 signature IE     IS     8.35e- 2 8.4 e- 2 0.084    ns       Wilcoxon
```

```
p3 <- sig_box(input, variable = "TMEcluster", signature = "IL1A",
              jitter = TRUE, cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature IA     IE     1.46e-10 2.90e-10 1.5e-10  ****     Wilcoxon
## 2 signature IA     IS     8.22e- 7 8.2 e- 7 8.2e-07  ****     Wilcoxon
## 3 signature IE     IS     4.90e-20 1.5 e-19 < 2e-16  ****     Wilcoxon
```

```
if (!requireNamespace("patchwork", quietly = TRUE))   install.packages("patchwork")
library(patchwork)
p <- (p1|p2/p3) + plot_layout(widths = c(2.3,1))
p + plot_annotation(tag_levels = 'A')
```



8.8 Identifying signatures associated with TME clusters

Calculate TME associated signatures-(through PCA method).

```
sig_tme<-calculate_sig_score(pdata           = NULL,
                             eset            = eset,
                             signature       = signature_collection,
                             method          = "pca",
                             mini_gene_count = 2)
sig_tme <- t(column_to_rownames(sig_tme, var = "ID"))
```

```
sig_tme[1:5, 1:3]
```

```
##                   GSM1523727 GSM1523728 GSM1523729
## CD_8_T_effector   -2.5513794  0.7789141 -2.1770675
## DDR               -0.8747614  0.7425162 -1.3272054
## APM                1.1098368  2.1988688 -0.9516419
## Immune_Checkpoint -2.3701787  0.9455120 -1.4844104
## CellCycle_Reg      0.1063358  0.7583302 -0.3649795
```

Finding signatures or cell types associated with TMEcluster

```
res <- find_markers_in_bulk(pdata = tme, eset = sig_tme, group = "TMEcluster", nfeatures
```

```
##
##  IA  IE  IS
## 107  96  97
## # A tibble: 60 x 7
## # Groups:   cluster [3]
##       p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
##       <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
##  1 4.10e-31      11.6  0.907 0.29   1.05e-28 IA      TMEscore-plus
##  2 1.05e-27      21.4  0.907 0.368  2.69e-25 IA      TMEscore-CIR
##  3 2.83e-23       7.30 0.757 0.254  7.24e-21 IA      TMEscoreA-plus
##  4 1.98e-17       8.88 0.701 0.316  5.07e-15 IA      TMEscoreA-CIR
##  5 4.95e-15       5.30 0.673 0.275  1.27e-12 IA      CD-8-T-effector
##  6 7.70e-15       3.67 0.71  0.332  1.97e-12 IA      Th1-cells-Bindea-et-al
##  7 9.76e-11       5.39 0.673 0.342  2.50e- 8 IA      Cytotoxic-cells-Danaher-et~
##  8 8.78e-10       4.17 0.682 0.394  2.25e- 7 IA      NK-CD56dim-cells-Bindea-et~
##  9 3.27e- 9       8.11 0.673 0.415  8.36e- 7 IA      Antigen-Processing-and-Pre~
## 10 5.40e- 9       6.37 0.645 0.409  1.38e- 6 IA      T-cell-inflamed-GEP-Ayers-~
## # i 50 more rows
```

```
top15 <-  res$top_markers %>% dplyr:: group_by(cluster) %>%  dplyr::top_n(15, avg_log2FC

p1 <- DoHeatmap(res$sce, top15$gene, group.colors = cols)+
  scale_fill_gradientn(colours = rev(colorRampPalette(RColorBrewer::brewer.pal(11,"RdBu
```

```
top15$gene  <- gsub(top15$gene, pattern = "-", replacement = "\\_")
input <- combine_pd_eset(eset = sig_tme, pdata = tme, feas = top15$gene, scale = T)
```

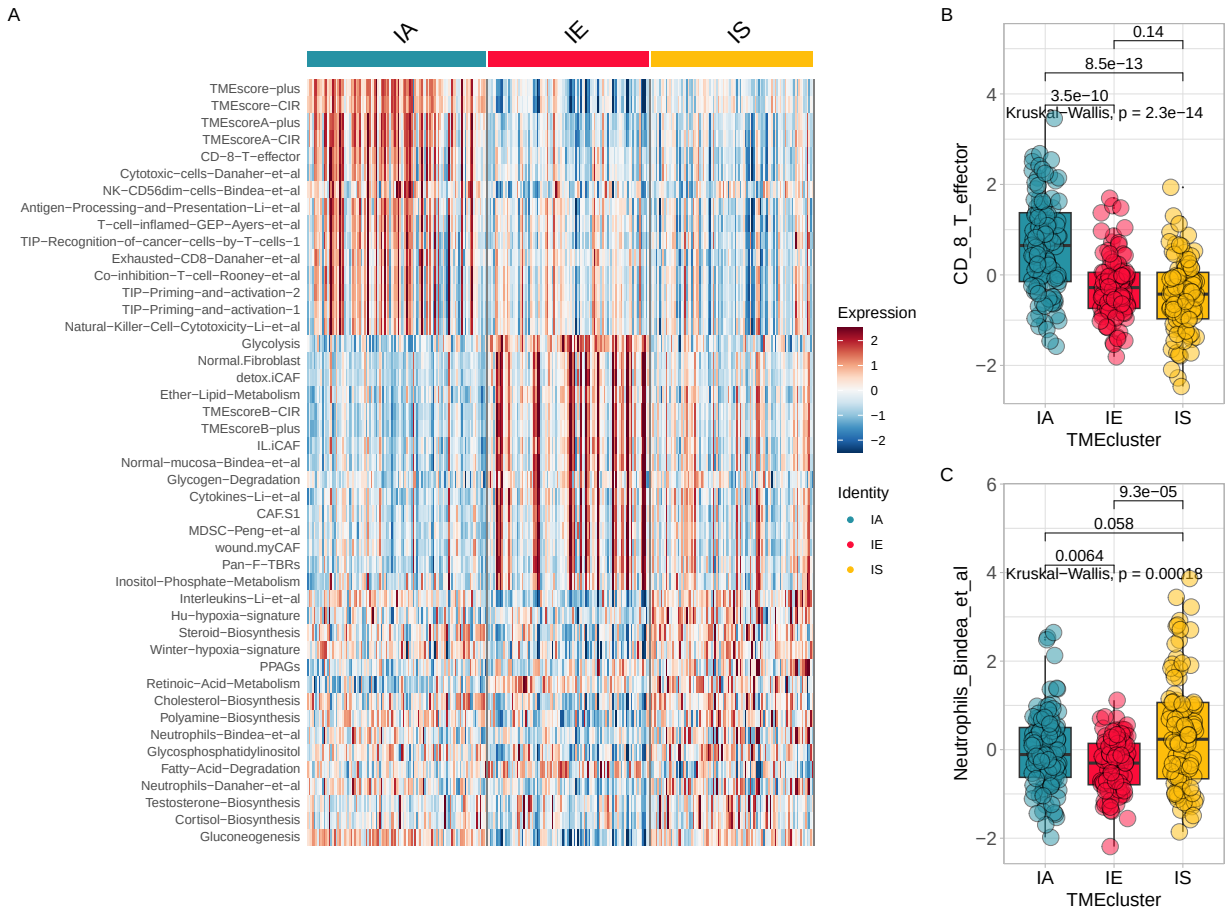
```
p2 <- sig_box(input, variable = "TMEcluster", signature = "CD_8_T_effector", jitter = TR
              cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4, size_of_font = 6)
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2        p    p.adj p.format p.signif method
##   <chr>     <chr>  <chr>     <dbl>    <dbl> <chr>    <chr>    <chr>
## 1 signature IA     IE     3.53e-10 7.10e-10 3.5e-10  ****     Wilcoxon
## 2 signature IA     IS     8.49e-13 2.5 e-12 8.5e-13  ****     Wilcoxon
## 3 signature IE     IS     1.41e- 1 1.4 e- 1 0.14     ns       Wilcoxon
```

```
p3 <- sig_box(input, variable = "TMEcluster", signature = "Neutrophils_Bindea_et_al",
              jitter = TRUE, cols =  cols, show_pvalue = TRUE, size_of_pvalue = 4, size_
```

```
## # A tibble: 3 x 8
##   .y.       group1 group2         p   p.adj p.format p.signif method
##   <chr>     <chr>  <chr>      <dbl>   <dbl> <chr>    <chr>    <chr>
## 1 signature IA     IE     0.00639   0.013   0.0064   **       Wilcoxon
## 2 signature IA     IS     0.0584    0.058   0.0584   ns       Wilcoxon
## 3 signature IE     IS     0.0000929 0.00028 9.3e-05  ****     Wilcoxon
```

```
p <- (p1|p2/p3) + plot_layout(widths = c(2.3,1))
p + plot_annotation(tag_levels = 'A')
```




```
library(survminer)
data(pdata_acrg, package = "IOBR")
input <- merge(pdata_acrg, input, by = "ID")
p1<-surv_group(input_pdata       = input,
               target_group      = "TMEcluster",
               ID                = "ID",
               reference_group   = "High",
               project           = "ACRG",
               cols              = cols,
               time              = "OS_time",
               status            = "OS_status",
               time_type         = "month",
               save_path         = "result")
```

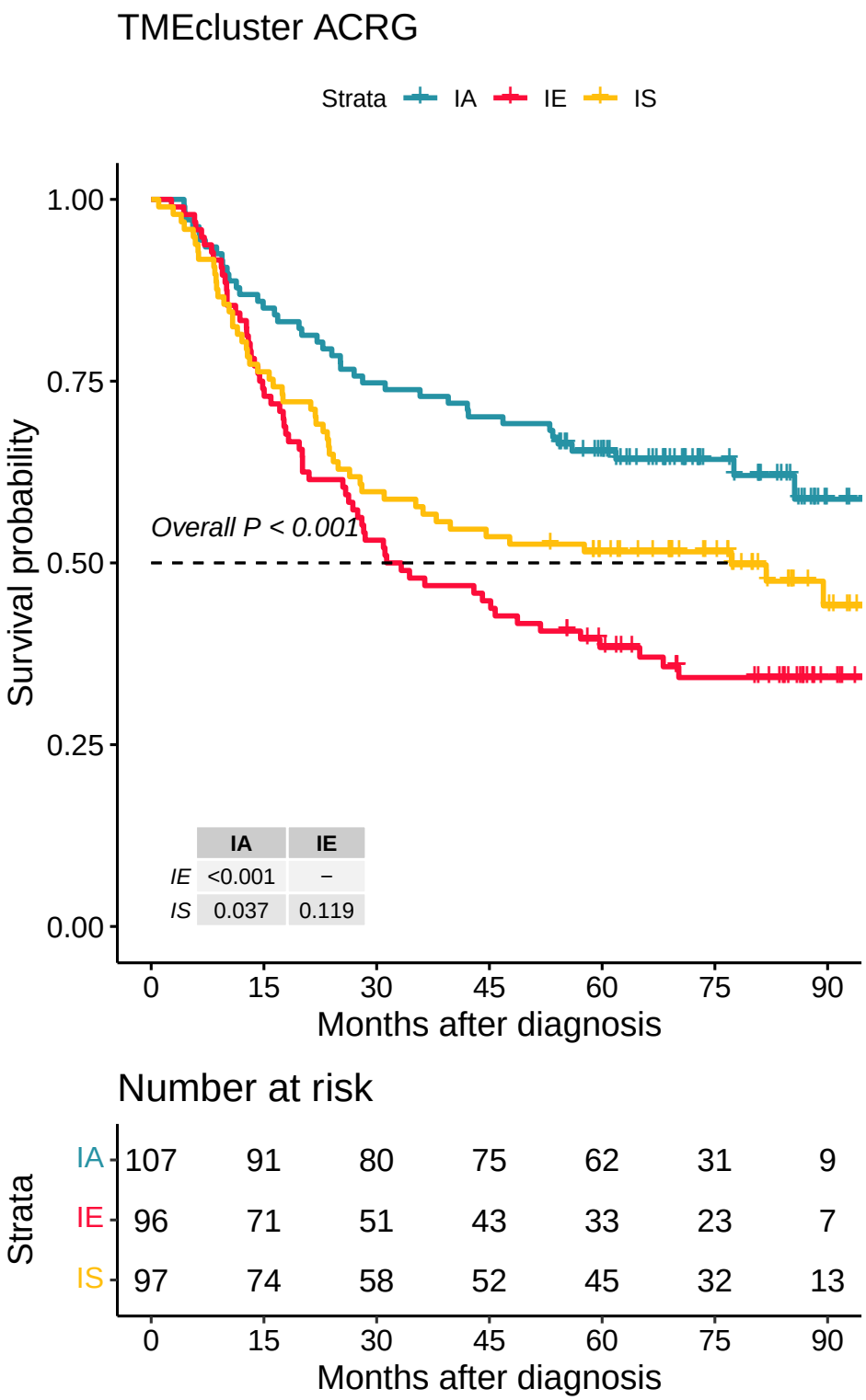
```
## >>> Dataset's survival follow up time is range between 1 to 105.7 months

##  IA  IE  IS
```

```
## 107  96  97
```

```
## 1079697
```

```
##   Maximum of follow up time is 105.7 months; and will be divided into 6 sections;
```

```
p1
```




```
p1<- percent_bar_plot(input, x = "TMEcluster" , y = "Subtype", palette = "jama")
```

```
## # A tibble: 12 x 5
## # Groups:   TMEcluster [3]
```

```
##    TMEcluster Subtype    Freq  Prop count
##    <chr>      <fct>     <dbl> <dbl> <dbl>
##  1 IA         EMT           7  0.07   107
##  2 IA         MSI          49  0.46   107
##  3 IA         MSS/TP53-    27  0.25   107
##  4 IA         MSS/TP53+    24  0.22   107
##  5 IE         EMT          24  0.25    96
##  6 IE         MSI           3  0.03    96
##  7 IE         MSS/TP53-    40  0.42    96
##  8 IE         MSS/TP53+    29  0.3     96
##  9 IS         EMT          15  0.15    97
## 10 IS         MSI          16  0.16    97
## 11 IS         MSS/TP53-    40  0.41    97
## 12 IS         MSS/TP53+    26  0.27    97
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```

```
p2<- percent_bar_plot(input, x = "TMEcluster" , y = "Lauren", palette = "jama")
```

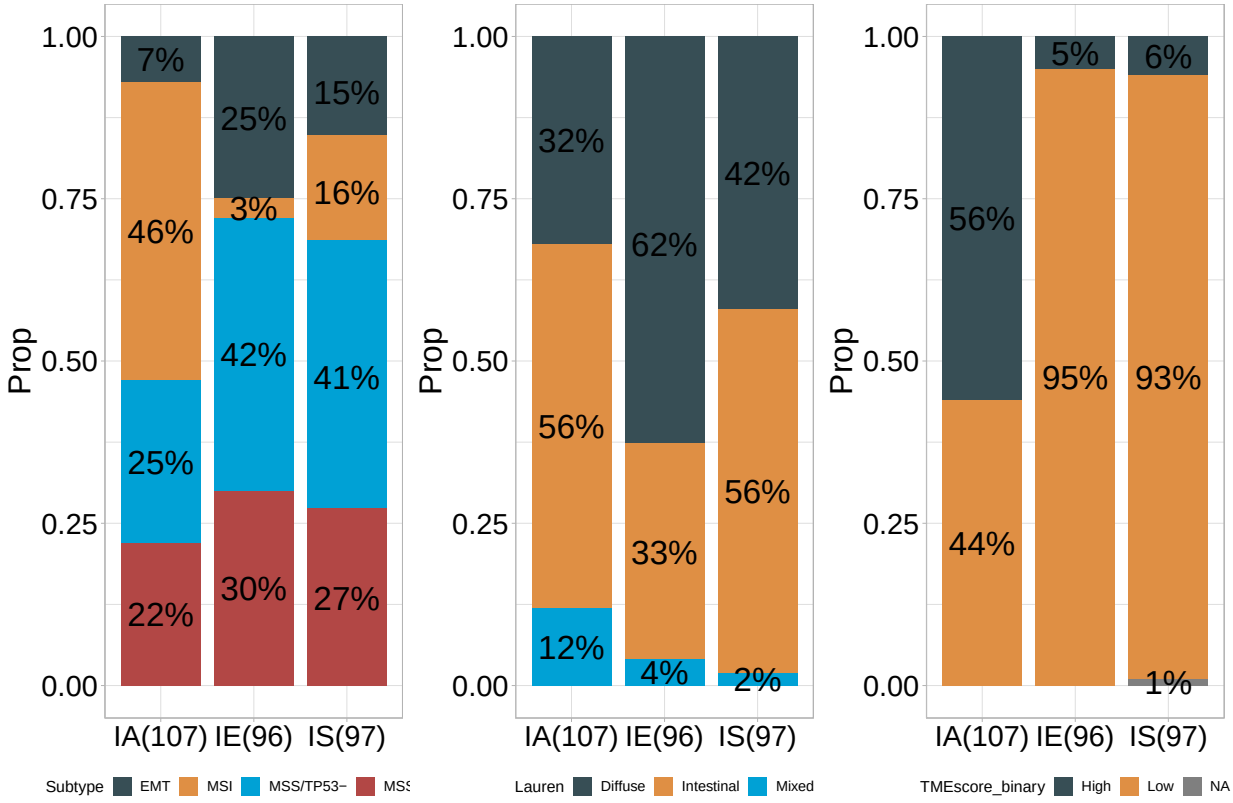
```
## # A tibble: 9 x 5
## # Groups:   TMEcluster [3]
##   TMEcluster Lauren      Freq  Prop count
##   <chr>      <fct>      <dbl> <dbl> <dbl>
## 1 IA         Diffuse       34  0.32   107
## 2 IA         Intestinal    60  0.56   107
## 3 IA         Mixed         13  0.12   107
## 4 IE         Diffuse       60  0.62    96
## 5 IE         Intestinal    32  0.33    96
## 6 IE         Mixed          4  0.04    96
## 7 IS         Diffuse       41  0.42    97
## 8 IS         Intestinal    54  0.56    97
## 9 IS         Mixed          2  0.02    97
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```

```
p3<- percent_bar_plot(input, x = "TMEcluster" , y = "TMEscore_binary", palette = "jama")
```

```
## # A tibble: 7 x 5
## # Groups:   TMEcluster [3]
##   TMEcluster TMEscore_binary  Freq  Prop count
##   <chr>      <fct>           <dbl> <dbl> <dbl>
```

```
## 1 IA          High                60  0.56   107
## 2 IA          Low                 47  0.44   107
## 3 IE          High                 5  0.05    96
## 4 IE          Low                 91  0.95    96
## 5 IS          High                 6  0.06    97
## 6 IS          Low                 90  0.93    97
## 7 IS          <NA>                 1  0.01    97
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```

```
p1|p2|p3
```



8.9 References

Cristescu, R., Lee, J., Nebozhyn, M. et al. Molecular analysis of gastric cancer identifies subtypes associated with distinct clinical outcomes. Nat Med 21, 449–456 (2015). https://doi.org/10.1038/nm.3850

CIBERSORT; Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., … Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles.

Nature Methods, 12(5), 453–457. https://doi.org/10.1038/nmeth.3337;

Seurat: Hao and Hao et al. Integrated analysis of multimodal single-cell data. Cell (2021)

Zeng D, Yu Y, Qiu W, Mao Q, …, Zhang K, Liao W; Tumor microenvironment immunotyping heterogeneity reveals distinct molecular mechanisms to clinical immunotherapy applications in gastric cancer. (2023) Under Review.

Chapter 9

TME and genomic interaction

9.1 Loading packages

```
library(IOBR)
```

9.2 Genomic data prepare

In this section, we are going to use the MAF data of TCGA-STAD cohort as an example dataset. This dataset could be found in multiple places, here we show two ways to get it.

9.2.1 Using TCGAbiolinks to download genomic data

R Bioconductor package **TCGAbiolinks** provides an R interface of GDC data portal, which stores updating TCGA data. You can check and install this package with the following code.

```
if (!requireNamespace("TCGAbiolinks", quietly = TRUE))
  BiocManager::install("TCGAbiolinks")
```

Then you can query, download and prepare the required dataset.

```
library(TCGAbiolinks)
query <- GDCquery(
  project = "TCGA-STAD",
  data.category = "Simple Nucleotide Variation",
  access = "open",
  data.type = "Masked Somatic Mutation",
```

```
  workflow.type = "Aliquot Ensemble Somatic Variant Merging and Masking"
)
GDCdownload(query)
maf <- GDCprepare(query)
```

This `maf` object is a `data.frame`, you can use `read.maf()` from R package **Maftools** to convert it to a MAF object.

In this example, we used the maf file of TCGA-STAD to extract the SNPs in it,and then transformed it into a non-negative matrix.

```
(load("TCGA-STAD.maf.RData"))
```

```
## [1] "maf"
```

```
mut_list <- make_mut_matrix(maf = maf, isTCGA    = T, category = "multi")
```

```
## -Validating
## -Silent variants: 45460
## -Summarizing
## --Possible FLAGS among top ten genes:
##   TTN
##   MUC16
##   SYNE1
##   FLG
## -Processing clinical data
## --Missing clinical data
## -Finished in 11.8s elapsed (11.0s cpu)
##        Frame_Shift_Del        Frame_Shift_Ins           In_Frame_Del
##                  21547                   4526                   1196
##           In_Frame_Ins      Missense_Mutation      Nonsense_Mutation
##                    106                 102137                   5669
##       Nonstop_Mutation           Splice_Site Translation_Start_Site
##                    117                   2242                    107
##    DEL    INS    ONP    SNP
##  22997   4675     10 109965
```

```
mut <- mut_list$snp
```

9.3 Identifying Mutations Associated with TME

The microenvironmental data from the TCGA-STAD expression matrix was merged. The Cuzick or Wilcoxon test was used to identify genetic variants associated with microenvironmental factors. CD_8_T_effector was used as the target variable in this example.

```
data("tcga_stad_sig", package = "IOBR")
res<-find_mutations(mutation_matrix     = mut,
                    signature_matrix    = tcga_stad_sig,
                    id_signature_matrix = "ID",
                    signature           = "CD_8_T_effector",
                    palette             = "nrc",
                    min_mut_freq        = 0.01,
                    plot                = TRUE,
                    jitter              = TRUE,
                    point.alpha         = 0.25)
```

```
## [1] ">>>> Result of Cuzick Test"
##             p.value  names statistic adjust_pvalue
## PIK3CA 6.279087e-10 PIK3CA  6.183261  3.139543e-07
## PLXNA4 8.196539e-05 PLXNA4  3.938580  2.049135e-02
## DMD    3.514677e-04    DMD  3.574075  3.882570e-02
## SPEG   3.665757e-04   SPEG  3.563047  3.882570e-02
## AHNAK2 3.882570e-04 AHNAK2  3.547940  3.882570e-02
## TCHH   6.571311e-04   TCHH  3.406867  5.476093e-02
## ABCC9  1.043516e-03  ABCC9  3.278524  7.453684e-02
## ANK3   1.470480e-03   ANK3  3.180447  7.713409e-02
## WDFY3  1.690248e-03  WDFY3  3.139867  7.713409e-02
## LRP1   1.762002e-03   LRP1  3.127666  7.713409e-02
```

```
## [1] ">>> Result of Wilcoxon test (top 10)"
##             p.value  names statistic adjust_pvalue
## PIK3CA 3.045532e-11 PIK3CA      4006  1.522766e-08
## TCHH   2.585594e-05   TCHH      2929  6.463985e-03
## PLXNA4 5.257043e-05 PLXNA4      2904  7.754311e-03
## LRP1   6.203449e-05   LRP1      2330  7.754311e-03
```

```
## RNF213 1.066555e-04 RNF213      3827  1.066555e-02
## SPEG   1.809850e-04   SPEG      1854  1.439761e-02
## WDFY3  2.420595e-04  WDFY3      2626  1.439761e-02
## DMD    2.430706e-04    DMD      4995  1.439761e-02
## ANK3   2.591569e-04   ANK3      3845  1.439761e-02
## AHNAK2 3.456706e-04 AHNAK2      4828  1.728353e-02

## All mutation types: mut.

## Warning: You defined `cell_fun` for a heatmap with more than 100 rows or
## columns, which might be very slow to draw. Consider to use the
## vectorized version `layer_fun`.

## All mutation types: mut.

## Warning: You defined `cell_fun` for a heatmap with more than 100 rows or
## columns, which might be very slow to draw. Consider to use the
## vectorized version `layer_fun`.
```

9.4 OncoPrint of result

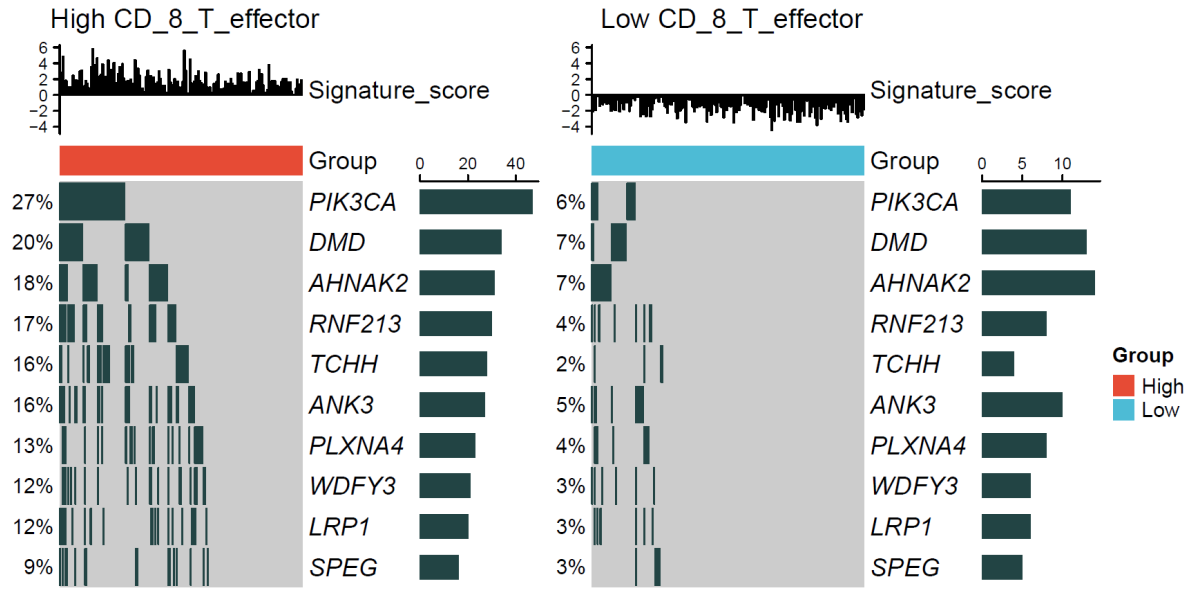


Figure 9.1: OncoPrint

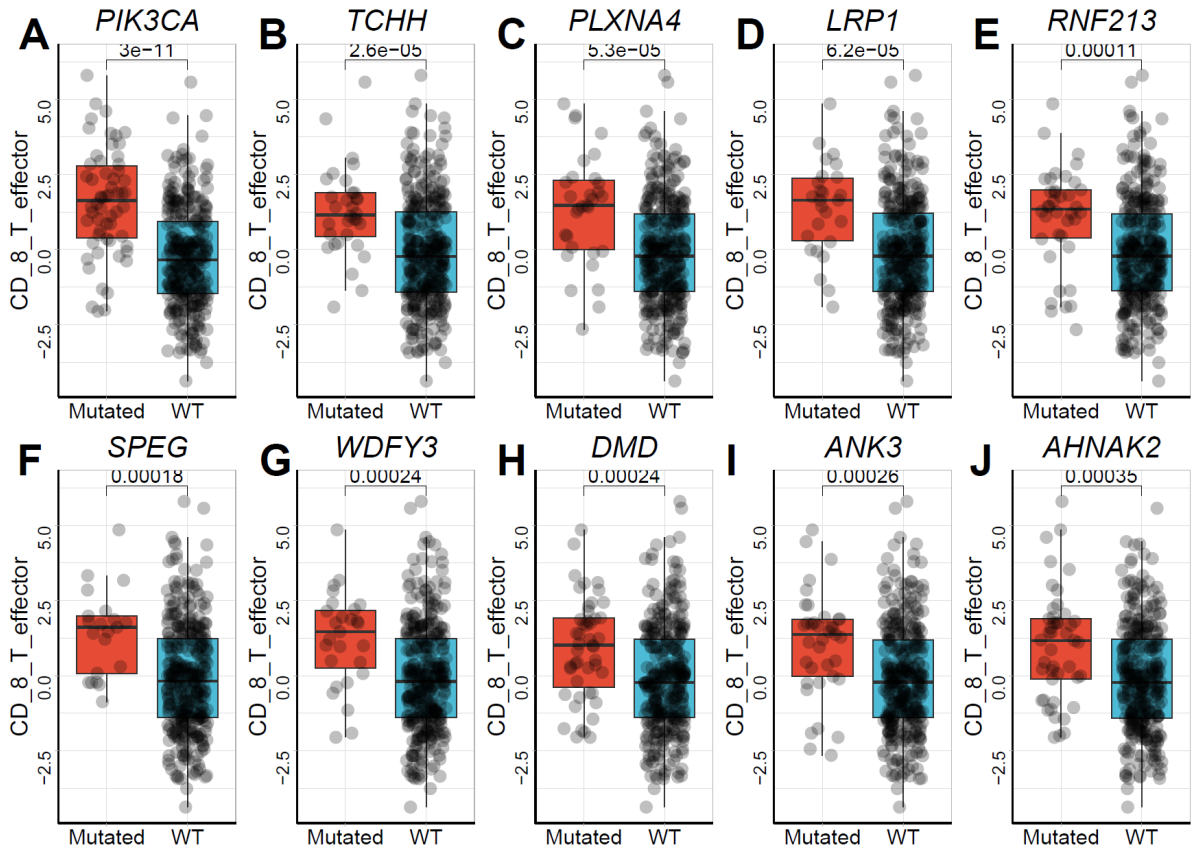


Figure 9.2: Top 10 mutated genes

9.5 Boxplot of top 10 mutated genes

9.6 Other methods to obtaind genomic data

9.6.1 Using TCGAmutations

As its name, the R package **TCGAmutations** provides pre-compiled, curated somatic mutations from 33 TCGA cohorts along with relevant clinical information for all sequenced samples. You can install it similar to the **TCGAbiolinks**.

```
if (!requireNamespace("TCGAmutations", quietly = TRUE))
  BiocManager::install("PoisonAlien/TCGAmutations")
```

It's quite simple to use:

```
maf = TCGAmutations::tcga_load(study = "STAD")
# Change `source` argument to Firehose for MAF files from Broad Firehose
# maf = TCGAmutations::tcga_load(study = "STAD", source = "Firehose")
```

9.6.2 Using maftools

If you are a user of R package **Maftools**, you can access and load the data in a similar way (because the author of **TCGAmutations** and **Maftools** is the same person).

```
# The following github can be changed to gitee
# it maybe fast in China mainland
maftools::tcgaAvailable(repo = "github")
maftools::tcgaLoad("STAD", repo = "github")
```

MAF data transformation To prepare the data for the downstream analysis. We need to extract the SNV data in it and transform it into a non-negative matrix.

```
mut_list <- make_mut_matrix(maf = maf, isTCGA = TRUE, category = "multi")
```

9.7 References

Wang et al., (2019). The UCSCXenaTools R package: a toolkit for accessing genomics data from UCSC Xena platform, from cancer multi-omics to single-cell RNA-seq. Journal of Open Source Software, 4(40), 1627, https://doi.org/10.21105/joss.01627

Gu, Z. (2022) Complex Heatmap Visualization. iMeta.

Anand Mayakonda et al., (2018) Maftools: efficient and comprehensive analysis of somatic variants in cancer. Genome Research

Chapter 10

TME Modeling

Previous studies have shown that the tumour microenvironment is a complex ecosystem. No single cell or gene is sufficient to influence the phenotype. Therefore, machine learning models of the tumour microenvironment or models of tumour microenvironment typing are used to predict tumour phenotypes and treatment response. In the last section, we present common considerations and scenarios for constructing tumour microenvironment models.

10.1 Loading packages

```
library(IOBR)
```

10.2 Data prepare

Using data from IMvigor210, we demonstrate two common scenarios for building models of the tumour microenvironment: predicting survival and predicting treatment response (BOR, RECIEST 1.1).

```
data("imvigor210_sig", package = "IOBR")
data("imvigor210_pdata", package = "IOBR")
```

10.3 Input data (overall survival) prepare

```
pdata_prog <- imvigor210_pdata %>%
  dplyr::select(ID, OS_days, OS_status) %>%
```

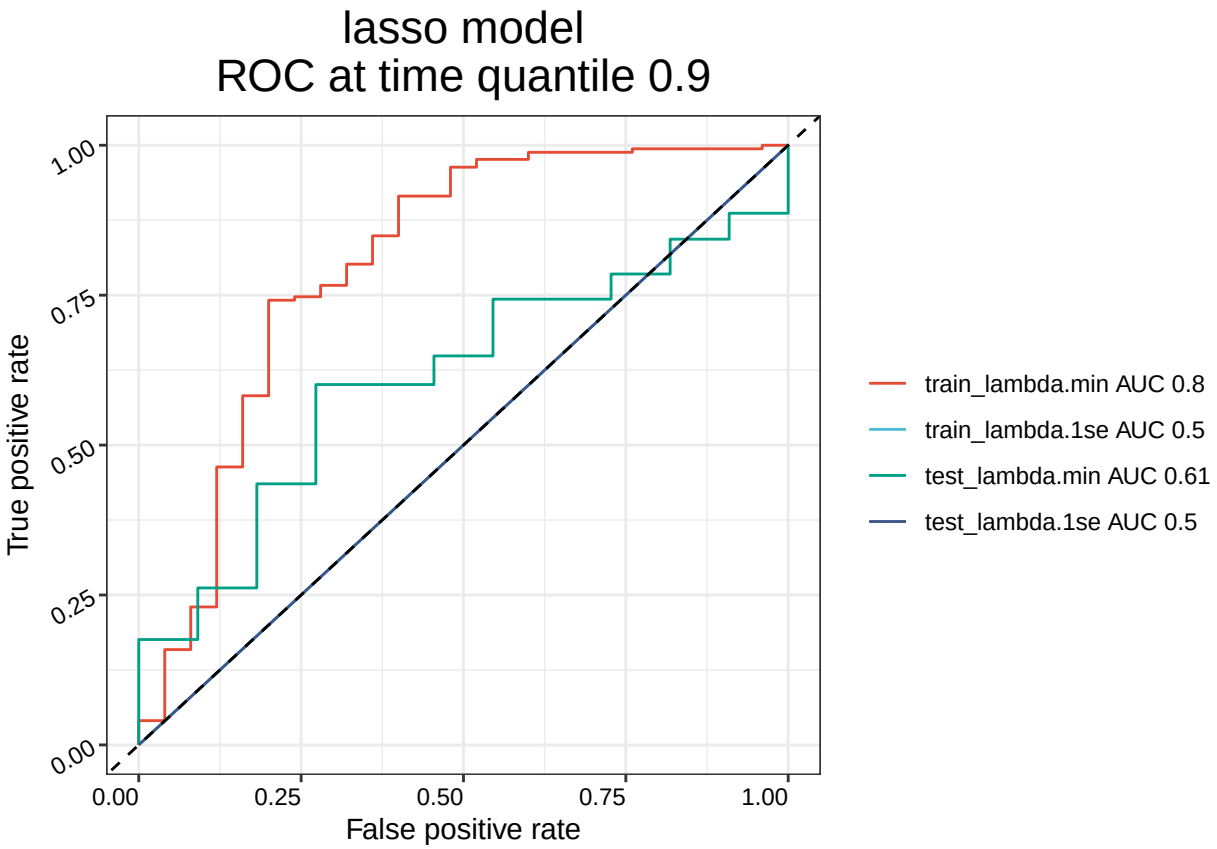
```
  mutate(OS_days = as.numeric(.$OS_days)) %>%
  mutate(OS_status = as.numeric(.$OS_status))

head(pdata_prog)
```

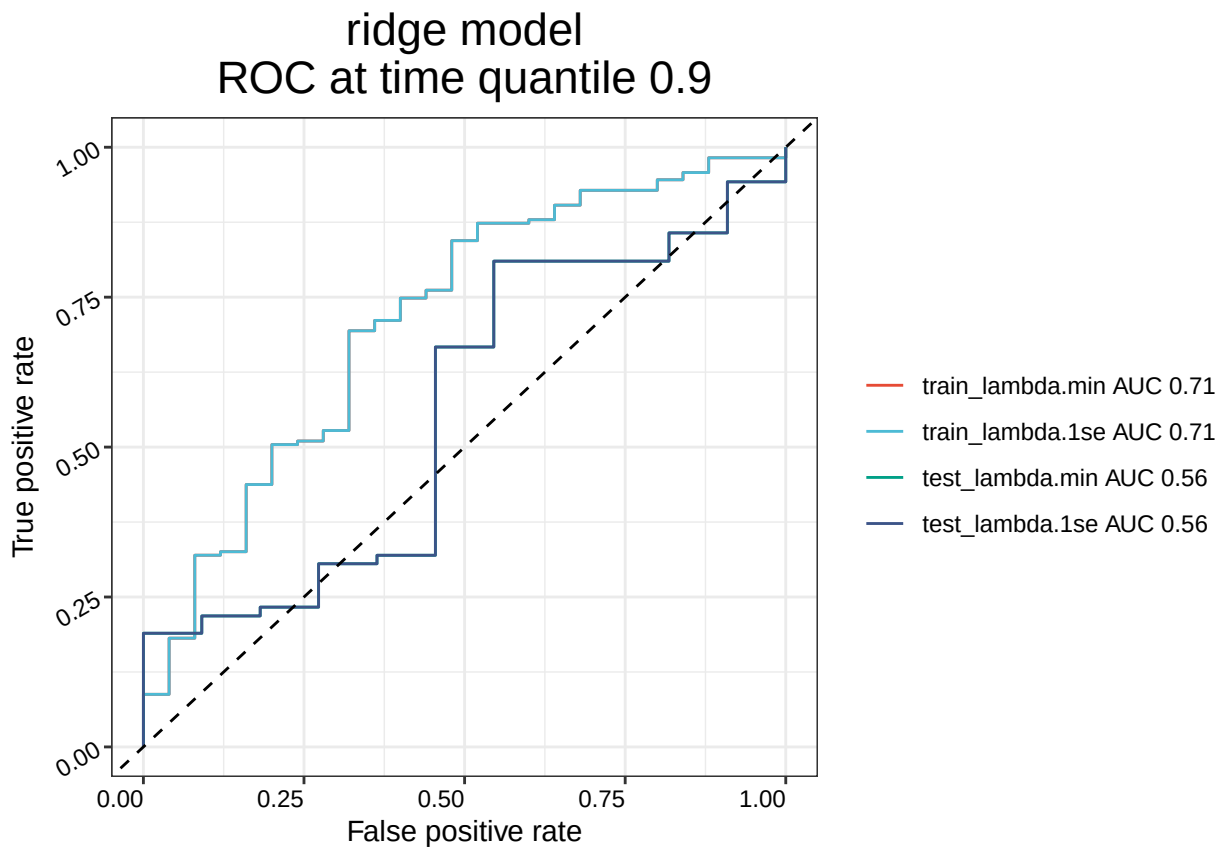
```
## # A tibble: 6 x 3
##   ID              OS_days OS_status
##   <chr>             <dbl>     <dbl>
## 1 SAM00b9e5c52da9    57.2         1
## 2 SAM0257bbbbd388   469.          1
## 3 SAM025b45c27e05   263.          1
## 4 SAM032c642382a7    74.9         1
## 5 SAM04c589eb3fb3    20.7         0
## 6 SAM0571f17f4045   136.          1
```

10.4 Constructing survival prediction models

```
prognostic_result <- PrognosticModel(x           = imvigor210_sig,
                                     y           = pdata_prog,
                                     scale       = T,
                                     seed        = 123456,
                                     train_ratio = 0.7,
                                     nfold       = 8,
                                     plot        = TRUE)
```

```
## NULL
## NULL
## NULL
```


lasso model
ROC at time quantile 0.9
True positive rate
False positive rate
0.00
0.25
0.50
0.75
1.00
train_lambda.min AUC 0.8
train_lambda.1se AUC 0.5
test_lambda.min AUC 0.61
test_lambda.1se AUC 0.5



10.5 Input data (Response) prepare

```
pdata_group <- imvigor210_pdata[!imvigor210_pdata$BOR_binary=="NA",c("ID","BOR_binary")]
pdata_group$BOR_binary <- ifelse(pdata_group$BOR_binary == "R", 1, 0)
head(pdata_group)
```

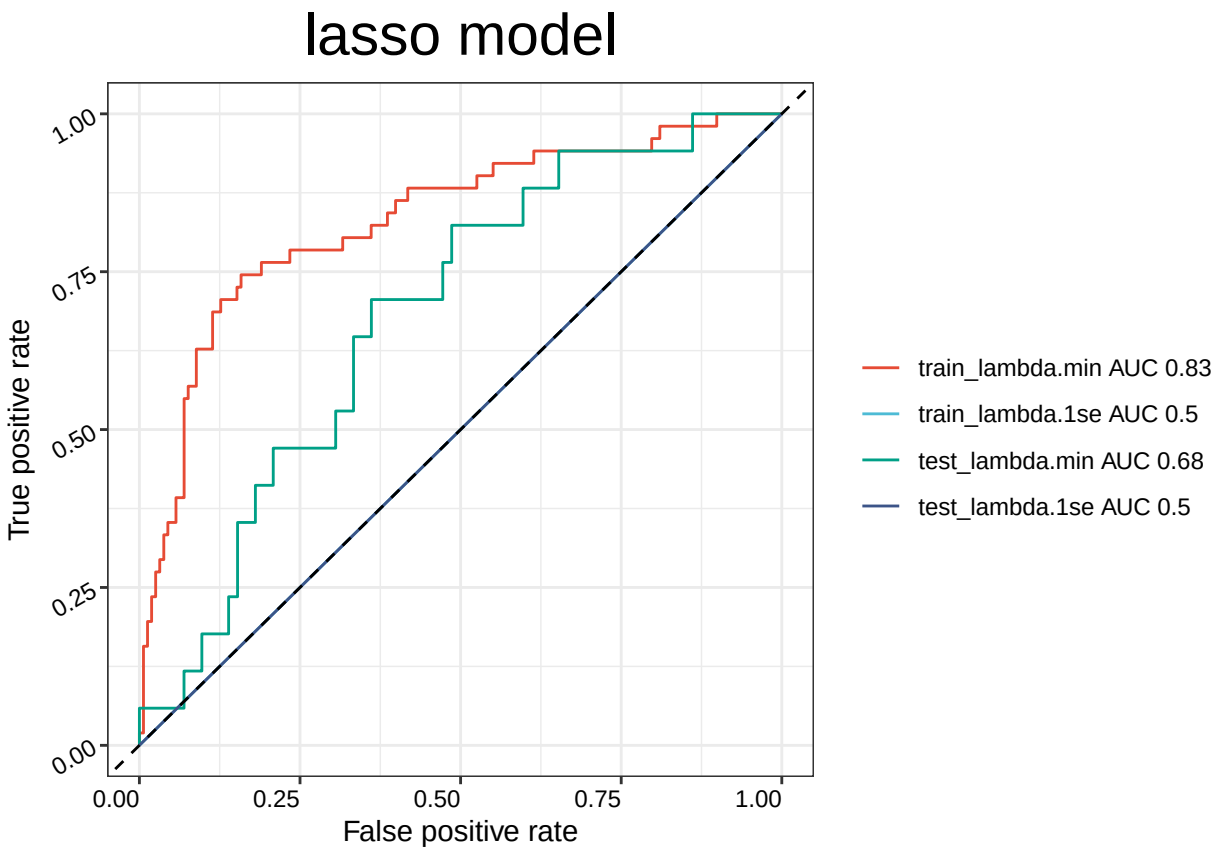
```
## # A tibble: 6 x 2
##   ID              BOR_binary
##   <chr>                <dbl>
## 1 SAM0257bbbbd388          0
## 2 SAM025b45c27e05          0
## 3 SAM032c642382a7          0
## 4 SAM0571f17f4045          0
## 5 SAM065890737112          1
## 6 SAM0684af734db1          1
```

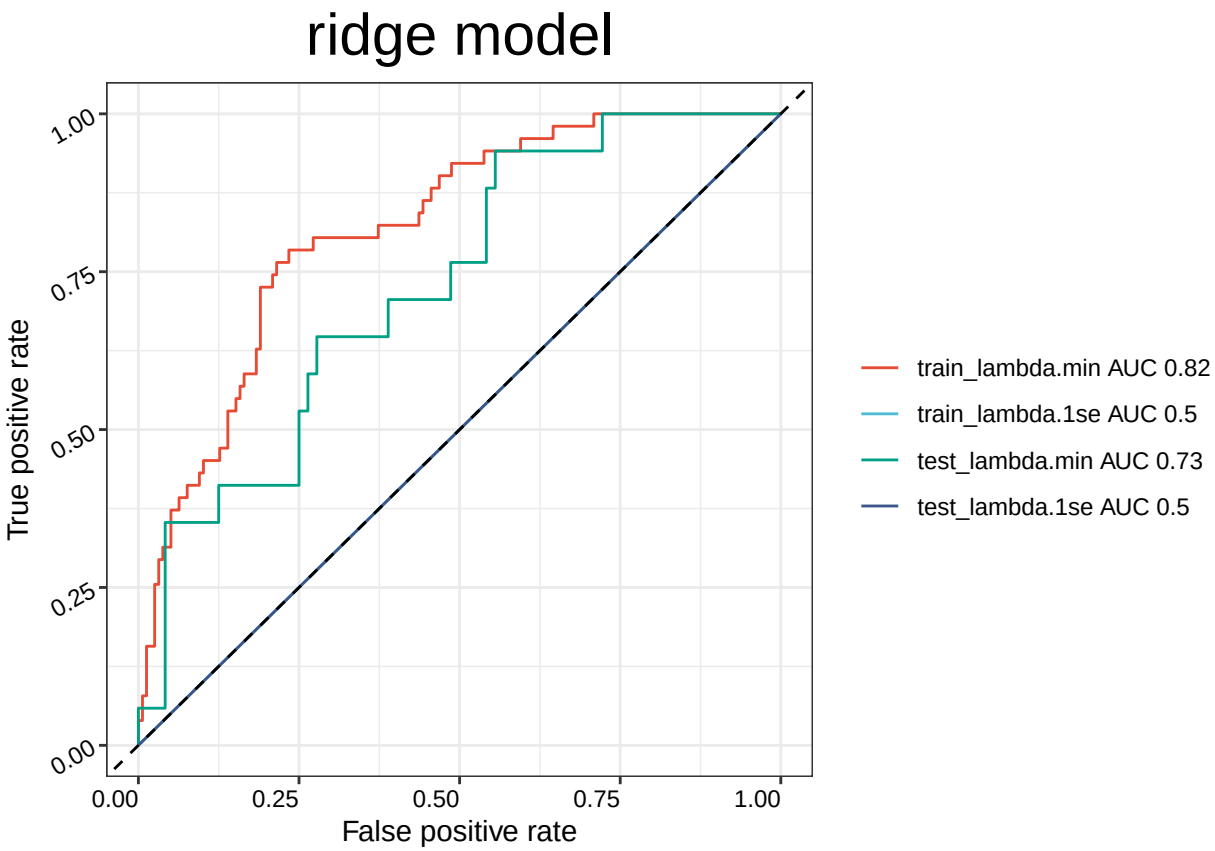
10.6 Constructing prediction models for response

```
binom_res <- BinomialModel(x           = imvigor210_sig,
                           y           = pdata_group,
                           seed        = 123456,
                           scale       = TRUE,
                           train_ratio = 0.7,
                           nfold       = 8,
                           plot        = T)
```

```
## NULL
## NULL
## NULL
```

```
## NULL
```






```
## NULL
```

10.7 References

Cristescu, R., Lee, J., Nebozhyn, M. et al. Molecular analysis of gastric cancer identifies subtypes associated with distinct clinical outcomes. Nat Med 21, 449–456 (2015). https://doi.org/10.1038/nm.3850

CIBERSORT; Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., … Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles. Nature Methods, 12(5), 453–457. https://doi.org/10.1038/nmeth.3337;

Seurat: Hao and Hao et al. Integrated analysis of multimodal single-cell data. Cell (2021)

Chapter 11

Some tips about IOBR

In this section, we'll cover some of the tips inside the IOBR package in terms of data processing and visualisation.

11.1 Loading packages

```
library(IOBR)
```

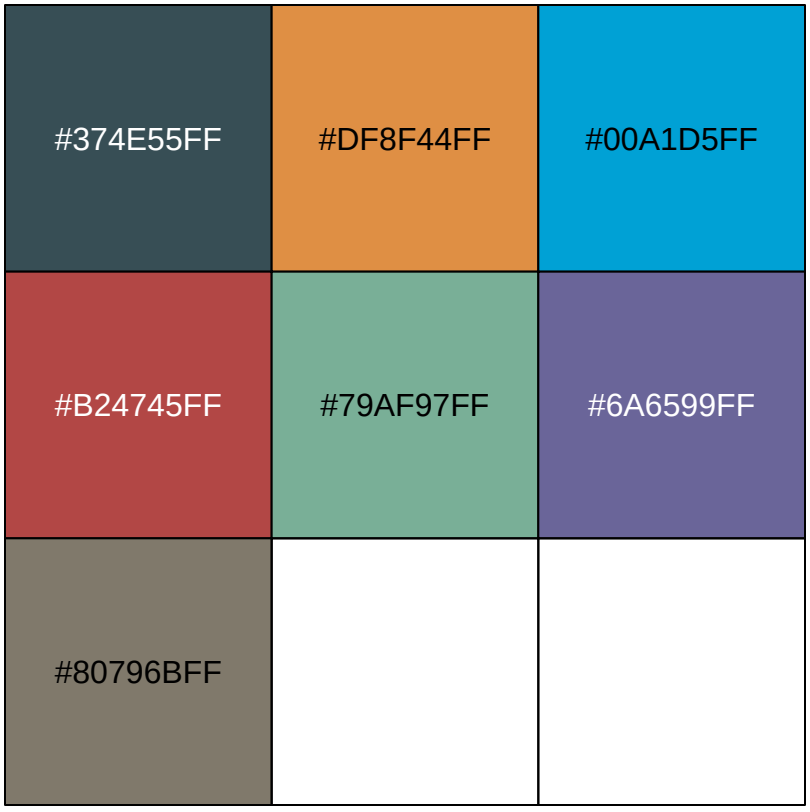
11.2 Colour Configuration

11.2.1 categorical variable

In IOBR, we created a function for get_col. The user can get the colours with some parameters in it. Please refer to the following example. In order to better suit the requirements of the journal, we have provided some of the more commonly used colour schemes. These include `nrc`, `jama`, `aaas`, `jco`, `paired1`, `paired2`, `paired3`, `paired4`, `accent`, `set2`.

```
cols <- get_cols(palette = "jama", show_col = T)
```

```
## [1] "'#374E55FF', '#DF8F44FF', '#00A1D5FF', '#B24745FF', '#79AF97FF', '#6A6599FF', '#
```




```
cols
```

```
## [1] "#374E55FF" "#DF8F44FF" "#00A1D5FF" "#B24745FF" "#79AF97FF" "#6A6599FF"
## [7] "#80796BFF"
```

```
cols <- get_cols( palette = "jco", show_col = T)
```

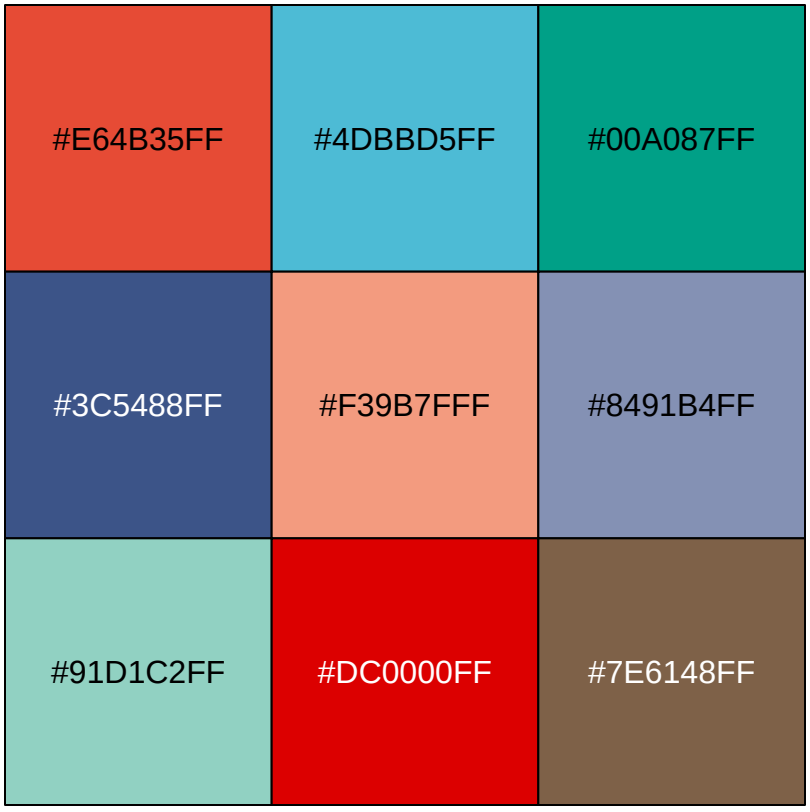
```
## [1] "'#0073C2FF', '#EFC000FF', '#868686FF', '#CD534CFF', '#7AA6DCFF', '#003C67FF', '#
```


```
cols
```

```
## [1] "#0073C2FF" "#EFC000FF" "#868686FF" "#CD534CFF" "#7AA6DCFF" "#003C67FF"
## [7] "#8F7700FF" "#3B3B3BFF" "#A73030FF"
```

```
cols <- get_cols(palette = "nrc", show_col = T)
```

```
## [1] "'#E64B35FF', '#4DBBD5FF', '#00A087FF', '#3C5488FF', '#F39B7FFF', '#8491B4FF', '#
```




```
cols
```

```
## [1] "#E64B35FF" "#4DBBD5FF" "#00A087FF" "#3C5488FF" "#F39B7FFF" "#8491B4FF"
## [7] "#91D1C2FF" "#DC0000FF" "#7E6148FF"
```

```
cols <- get_cols(palette = "aaas", show_col = T)
```

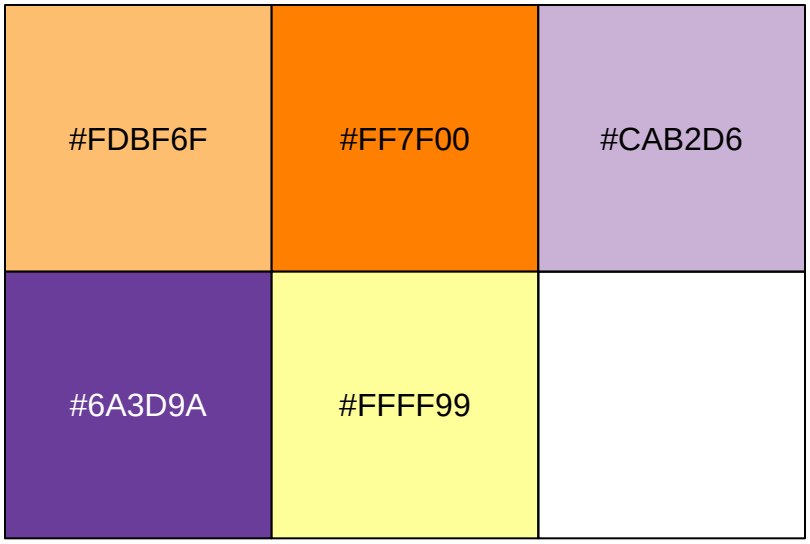
```
## [1] "'#3B4992FF', '#EE0000FF', '#008B45FF', '#631879FF', '#008280FF', '#BB0021FF', '#
```


```
cols
```

```
## [1] "#3B4992FF" "#EE0000FF" "#008B45FF" "#631879FF" "#008280FF" "#BB0021FF"
## [7] "#5F559BFF" "#A20056FF" "#808180FF"
```

```
cols <- get_cols(palette = "paired4", show_col = T)
```

```
## [1] "'#FDBF6F', '#FF7F00', '#CAB2D6', '#6A3D9A', '#FFFF99'"
```

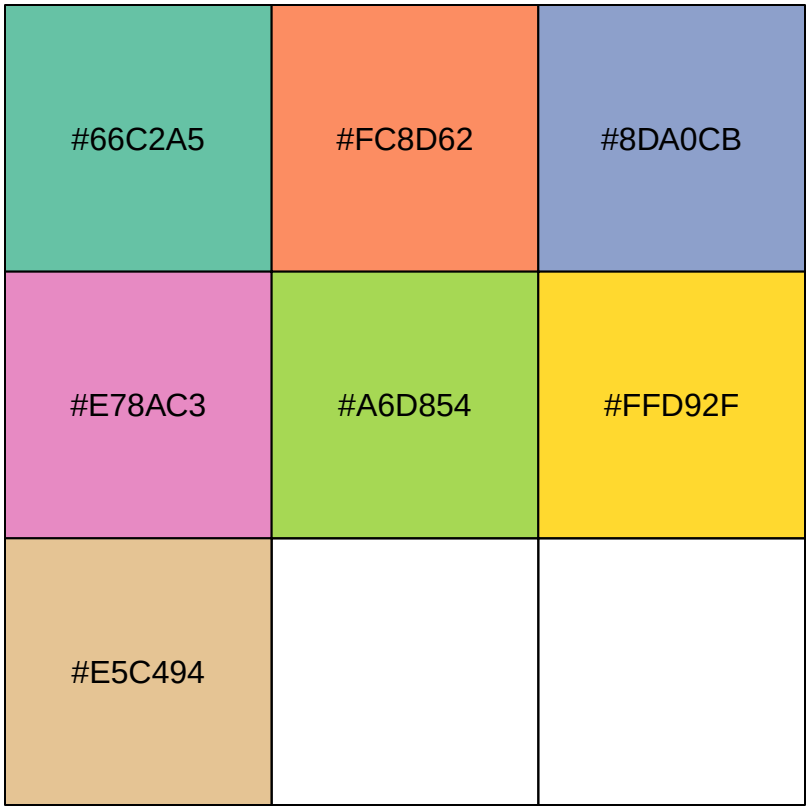



```
cols
```

```
## [1] "#FDBF6F" "#FF7F00" "#CAB2D6" "#6A3D9A" "#FFFF99"
```

```
cols <- get_cols(palette = "set2", show_col = T)
```

```
## [1] "'#66C2A5', '#FC8D62', '#8DA0CB', '#E78AC3', '#A6D854', '#FFD92F', '#E5C494'"
```




```
cols
```

```
## [1] "#66C2A5" "#FC8D62" "#8DA0CB" "#E78AC3" "#A6D854" "#FFD92F" "#E5C494"
```

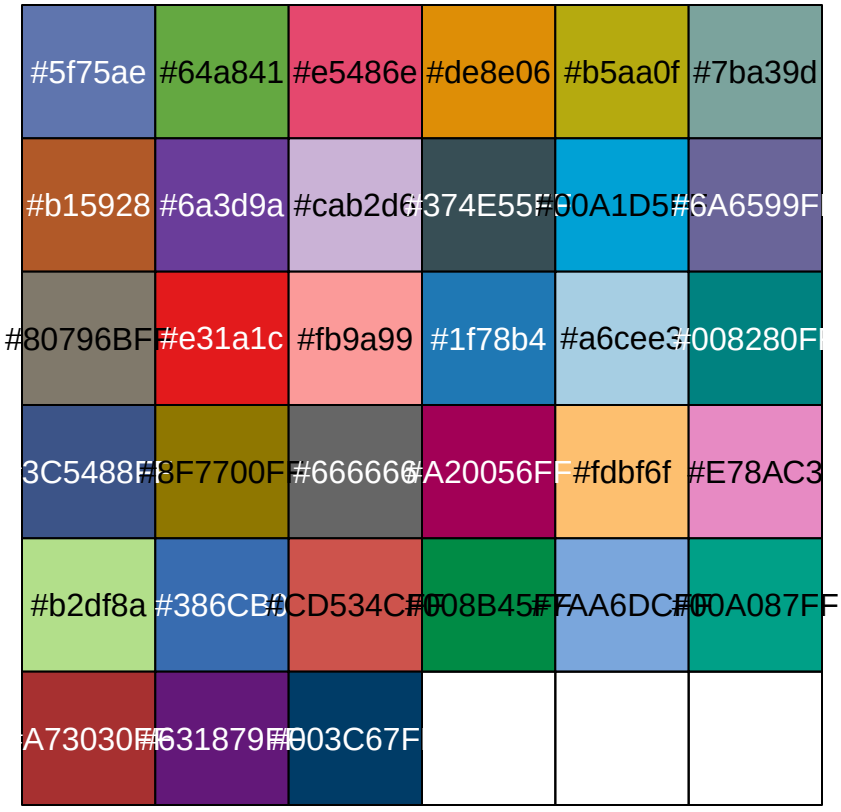
11.2.2 Higher number of subgroups

In order to cope with multiple groupings, we have selected some colours that are more identifiable with each other and stored these colourways in the objects palette1-4. The user can obtain them by setting the parameter `palette`.

```
cols <- get_cols(palette = 1, show_col = T)
```

```
## >>>>=== Palette option for random: 1: palette1; 2: palette2; 3: palette3;  4: palette
```

```
## [1] "'#5f75ae', '#64a841', '#e5486e', '#de8e06', '#b5aa0f', '#7ba39d', '#b15928', '#6
```




```
cols
```

```
##  [1] "#5f75ae"   "#64a841"   "#e5486e"   "#de8e06"   "#b5aa0f"   "#7ba39d"
##  [7] "#b15928"   "#6a3d9a"   "#cab2d6"   "#374E55FF" "#00A1D5FF" "#6A6599FF"
## [13] "#80796BFF" "#e31a1c"   "#fb9a99"   "#1f78b4"   "#a6cee3"   "#008280FF"
## [19] "#3C5488FF" "#8F7700FF" "#666666"   "#A20056FF" "#fdbf6f"   "#E78AC3"
## [25] "#b2df8a"   "#386CB0"   "#CD534CFF" "#008B45FF" "#7AA6DCFF" "#00A087FF"
## [31] "#A73030FF" "#631879FF" "#003C67FF"
```

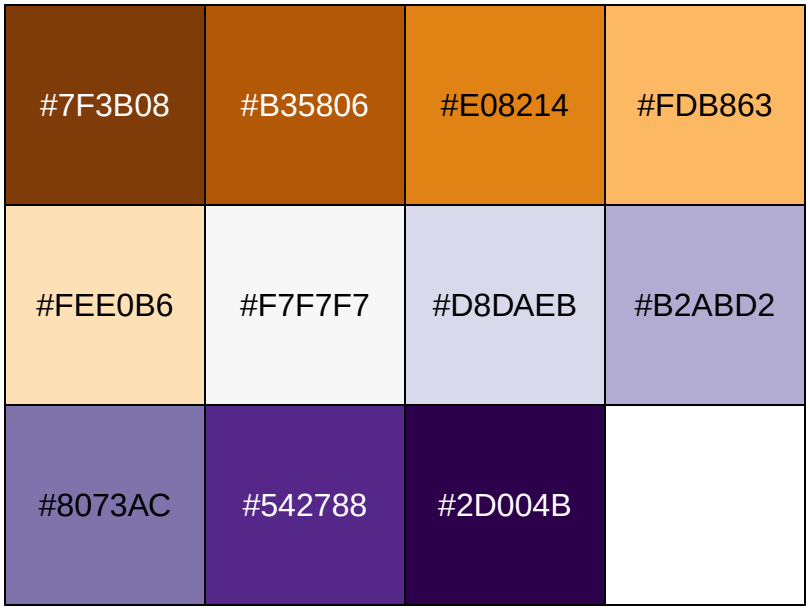
11.2.3 Gradient colour or heatmap colour scheme

```
palettes(category = "continue", palette = "puor", show_col = TRUE, show_message = TRUE)
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are four palettes you can choose: rdbu, puor, blues, reds
```

```
## [1] "'#7F3B08', '#B35806', '#E08214', '#FDB863', '#FEE0B6', '#F7F7F7', '#D8DAEB', '#B
```



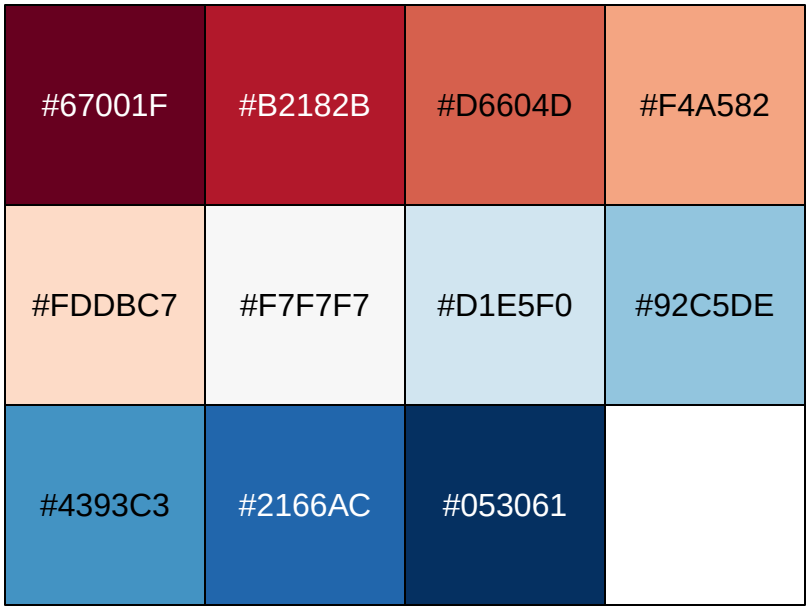

```
##  [1] "#7F3B08" "#B35806" "#E08214" "#FDB863" "#FEE0B6" "#F7F7F7" "#D8DAEB"
##  [8] "#B2ABD2" "#8073AC" "#542788" "#2D004B"
```

```
palettes(category = "continue", palette = "rdbu", show_col = TRUE, show_message = TRUE)
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are four palettes you can choose: rdbu, puor, blues, reds
```

```
## [1] "'#67001F', '#B2182B', '#D6604D', '#F4A582', '#FDDBC7', '#F7F7F7', '#D1E5F0', '#9
```




```
##  [1] "#67001F" "#B2182B" "#D6604D" "#F4A582" "#FDDBC7" "#F7F7F7" "#D1E5F0"
##  [8] "#92C5DE" "#4393C3" "#2166AC" "#053061"
```

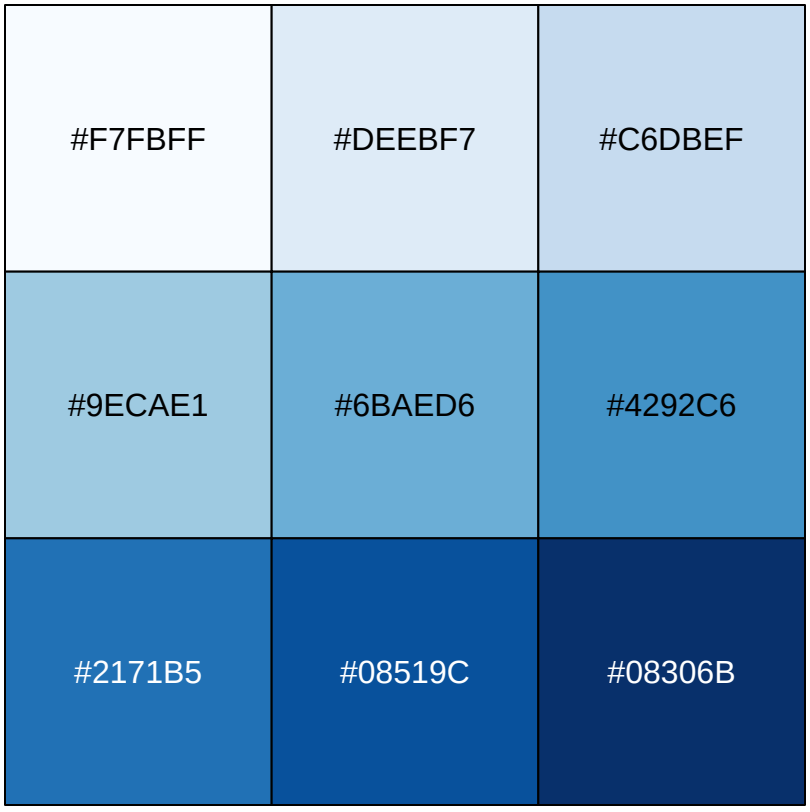
```
palettes(category = "continue", palette = "blues", show_col = TRUE, show_message = TRUE)
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are four palettes you can choose: rdbu, puor, blues, reds
```

```
## Warning in RColorBrewer::brewer.pal(11, "Blues"): n too large, allowed maximum for pa
## Returning the palette you asked for with that many colors
```

```
## [1] "'#F7FBFF', '#DEEBF7', '#C6DBEF', '#9ECAE1', '#6BAED6', '#4292C6', '#2171B5', '#0
```




```
## [1] "#F7FBFF" "#DEEBF7" "#C6DBEF" "#9ECAE1" "#6BAED6" "#4292C6" "#2171B5"
## [8] "#08519C" "#08306B"
```

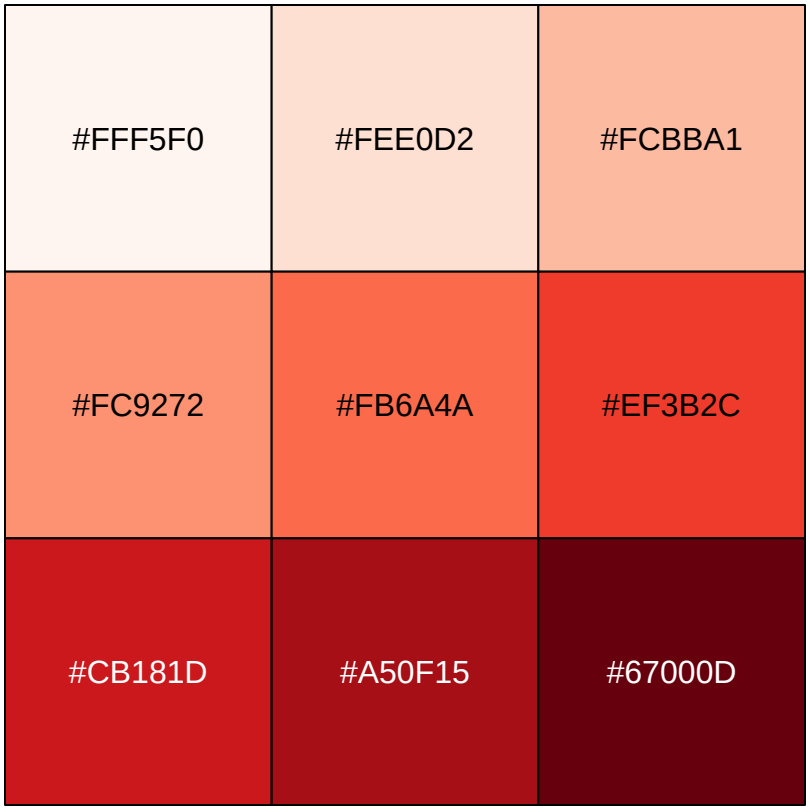
```
palettes(category = "continue", palette = "reds", show_col = TRUE, show_message = TRUE)
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are four palettes you can choose: rdbu, puor, blues, reds
```

```
## Warning in RColorBrewer::brewer.pal(11, "Reds"): n too large, allowed maximum for pal
## Returning the palette you asked for with that many colors
```

```
## [1] "'#FFF5F0', '#FEE0D2', '#FCBBA1', '#FC9272', '#FB6A4A', '#EF3B2C', '#CB181D', '#A
```




```
## [1] "#FFF5F0" "#FEE0D2" "#FCBBA1" "#FC9272" "#FB6A4A" "#EF3B2C" "#CB181D"
## [8] "#A50F15" "#67000D"
```

For heatmap colour configuration, usually more colours are needed. Users can adjust the number of colours to be returned by setting the parameter `category` to heatmap and by adjusting `count`. In IOBR, we offer a total of 7 colour schemes. Users can choose the colour scheme by setting the palette.

```
palettes(category = "heatmap", palette = "1", counts = 20, show_col = TRUE, show_message
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are five palettes you can choose: 1 = pheatmap,  2 = peach,  3 = blues, 4 = vir
```

```
## [1] "'#4575B4', '#5D8CC0', '#75A3CC', '#8CBBD8', '#A5CCE2', '#BEDDEB', '#D7EDF4', '#E
```

#4575B4	#5D8CC0	#75A3CC	#8CBBD8	#A5CCE2
#BEDDEB	#D7EDF4	#E6F5EC	#F0F9DA	#FAFDC8
#FEFAB7	#FEF0A8	#FEE699	#FDD78A	#FDBD78
#FCA267	#FA8856	#EE6A46	#E24D36	#D73027

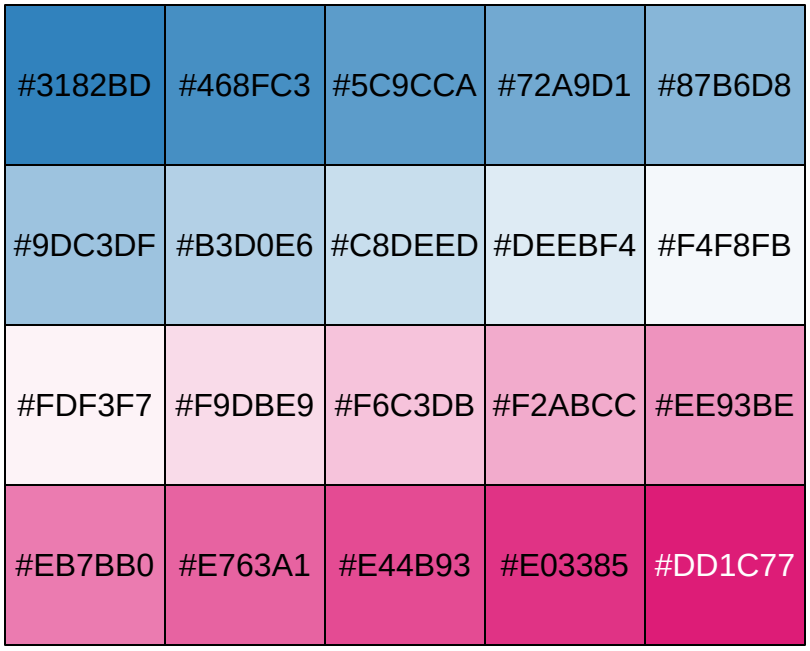
```
##  [1] "#4575B4" "#5D8CC0" "#75A3CC" "#8CBBD8" "#A5CCE2" "#BEDDEB" "#D7EDF4"
##  [8] "#E6F5EC" "#F0F9DA" "#FAFDC8" "#FEFAB7" "#FEF0A8" "#FEE699" "#FDD78A"
## [15] "#FDBD78" "#FCA267" "#FA8856" "#EE6A46" "#E24D36" "#D73027"
```

```
palettes(category = "heatmap", palette = "2", counts = 20, show_col = TRUE, show_message
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are five palettes you can choose: 1 = pheatmap,  2 = peach,  3 = blues, 4 = vir
```

```
## [1] "'#3182BD', '#468FC3', '#5C9CCA', '#72A9D1', '#87B6D8', '#9DC3DF', '#B3D0E6', '#C
```




```
##  [1] "#3182BD" "#468FC3" "#5C9CCA" "#72A9D1" "#87B6D8" "#9DC3DF" "#B3D0E6"
##  [8] "#C8DEED" "#DEEBF4" "#F4F8FB" "#FDF3F7" "#F9DBE9" "#F6C3DB" "#F2ABCC"
## [15] "#EE93BE" "#EB7BB0" "#E763A1" "#E44B93" "#E03385" "#DD1C77"
```

```
palettes(category = "heatmap", palette = "3", counts = 20, show_col = TRUE, show_message
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are five palettes you can choose: 1 = pheatmap,  2 = peach,  3 = blues, 4 = vir
```

```
## [1] "'#084594', '#1155A0', '#1A65AC', '#2474B6', '#3080BD', '#3C8CC3', '#4A97C9', '#5
```

#084594	#1155A0	#1A65AC	#2474B6	#3080BD
#3C8CC3	#4A97C9	#59A2CF	#68ACD5	#7BB6D9
#8DC1DD	#A0CAE1	#AED1E6	#BDD7EC	#C9DDF0
#D2E3F3	#DBE9F6	#E4EFF9	#EDF5FC	#F7FBFF

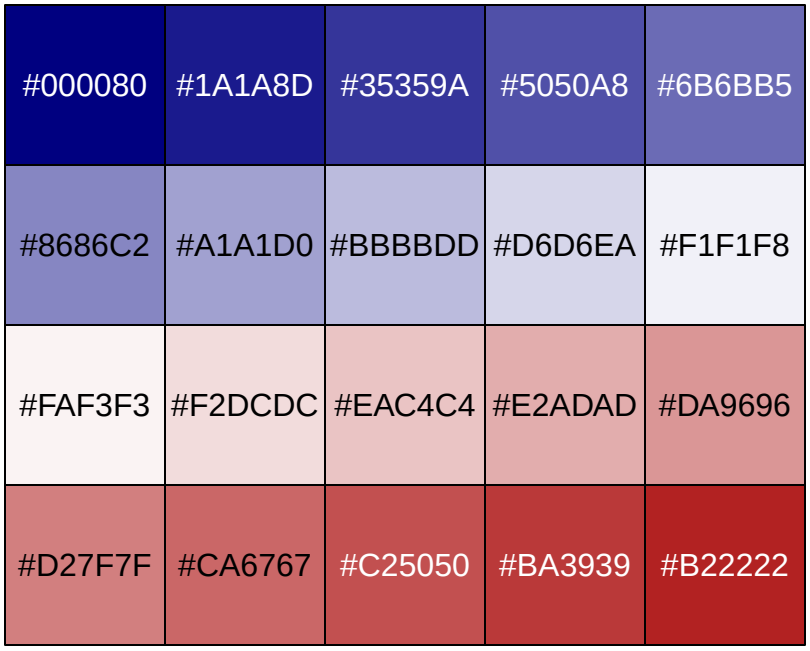
```
##  [1] "#084594" "#1155A0" "#1A65AC" "#2474B6" "#3080BD" "#3C8CC3" "#4A97C9"
##  [8] "#59A2CF" "#68ACD5" "#7BB6D9" "#8DC1DD" "#A0CAE1" "#AED1E6" "#BDD7EC"
## [15] "#C9DDF0" "#D2E3F3" "#DBE9F6" "#E4EFF9" "#EDF5FC" "#F7FBFF"
```

```
palettes(category = "heatmap", palette = "7", counts = 20, show_col = TRUE, show_message
```

```
## There are seven categories you can choose: box, continue2, continue, random, heatmap,
```

```
## There are five palettes you can choose: 1 = pheatmap,  2 = peach,  3 = blues, 4 = vir
```

```
## [1] "'#000080', '#1A1A8D', '#35359A', '#5050A8', '#6B6BB5', '#8686C2', '#A1A1D0', '#B
```




```
##  [1] "#000080" "#1A1A8D" "#35359A" "#5050A8" "#6B6BB5" "#8686C2" "#A1A1D0"
##  [8] "#BBBBDD" "#D6D6EA" "#F1F1F8" "#FAF3F3" "#F2DCDC" "#EAC4C4" "#E2ADAD"
## [15] "#DA9696" "#D27F7F" "#CA6767" "#C25050" "#BA3939" "#B22222"
```

Chapter 12

References

If IOBR R package is utilized in your published research, please cite:

Zeng D, Ye Z, Shen R, Yu G, Wu J, Xiong Y,..., Liao W (2021) **IOBR**: Multi-Omics Immuno-Oncology Biological Research to Decode Tumor Microenvironment and Signatures. *Frontiers in Immunology*. 12:687975. doi: 10.3389/fimmu.2021.687975

12.1 TME deconvolution

Please cite the following papers appropriately for TME deconvolution algorithm if used:

CIBERSORT: Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., … Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles. Nature Methods, 12(5), 453–457. https://doi.org/10.1038/nmeth.3337

ESTIMATE: Vegesna R, Kim H, Torres-Garcia W, ..., Verhaak R.*(2013). Inferring tumour purity and stromal and immune cell admixture from expression data. Nature Communications 4, 2612. http://doi.org/10.1038/ncomms3612

quanTIseq: Finotello, F., Mayer, C., Plattner, C., Laschober, G., Rieder, D., Hackl, H., ..., Sopper, S.* (2019). Molecular and pharmacological modulators of the tumor immune contexture revealed by deconvolution of RNA-seq data. Genome medicine, 11(1), 34. https://doi.org/10.1186/s13073-019-0638-6

TIMER: Li, B., Severson, E., Pignon, J.-C., Zhao, H., Li, T., Novak, J., … Liu, X. S.* (2016). Comprehensive analyses of tumor immunity: implications for cancer immunotherapy. Genome Biology, 17(1), 174.

IPS: P. Charoentong et al.*, Pan-cancer Immunogenomic Analyses Reveal Genotype-Immunophenotype Relationships and Predictors of Response to Checkpoint Blockade. Cell Reports 18, 248-262 (2017). https://doi.org/10.1016/j.celrep.2016.12.019

MCPCounter: Becht, E., Giraldo, N. A., Lacroix, L., Buttard, B., Elarouci, N., Petitprez, F., … de Reyniès, A*. (2016). Estimating the population abundance of tissue-infiltrating immune and stromal cell populations using gene expression. Genome Biology, 17(1), 218. https://doi.org/10.1186/s13059-016-1070-5

xCell: Aran, D., Hu, Z., & Butte, A. J.* (2017). xCell: digitally portraying the tissue cellular heterogeneity landscape. Genome Biology, 18(1), 220. https://doi.org/10.1186/s13059-017-1349-1

EPIC: Racle, J., de Jonge, K., Baumgaertner, P., Speiser, D. E., & Gfeller, D*. (2017). Simultaneous enumeration of cancer and immune cell types from bulk tumor gene expression data. ELife, 6, e26476. https://doi.org/10.7554/eLife.26476

12.2 TME Signatures

For signature score estimation, please cite corresponding literature below:

ssgsea: Barbie, D.A. et al (2009). Systematic RNA interference reveals that oncogenic KRAS-driven cancers require TBK1. Nature, 462(5):108-112.

gsva: Hänzelmann, S., Castelo, R. and Guinney, J. (2013). GSVA: Gene set variation analysis for microarray and RNA-Seq data. BMC Bioinformatics, 14(1):7.

zscore: Lee, E. et al (2008). Inferring pathway activity toward precise disease classification. PLoS Comp Biol, 4(11):e1000217.

12.3 Data sets

For the datasets enrolled in IOBR, please cite the data sources:

UCSCXena: Wang et al.,et al (2019). The UCSCXenaTools R package: a toolkit for accessing genomics data from UCSC Xena platform, from cancer multi-omics to single-cell RNA-seq. Journal of Open Source Software, 4(40), 1627

TLSscore: Helmink BA, Reddy SM, Gao J, et al. B cells and tertiary lymphoid structures promote immunotherapy response. Nature. 2020 Jan;577(7791):549-555.

IMvigor210 immuntherapy cohort: Mariathasan S, Turley SJ, Nickles D, et al. TGF attenuates tumour response to PD-L1 blockade by contributing to exclusion of T cells. Nature. 2018 Feb 22;554(7693):544-548. **HCP5**: Kulski, J.K. Long Noncoding RNA HCP5, a Hybrid HLA Class I Endogenous Retroviral Gene: Structure, Expression, and Disease Associations. Cells 2019, 8, 480.

HCP5: Li, Y., Jiang, T., Zhou, W. et al. Pan-cancer characterization of immune-related lncRNAs identifies potential oncogenic biomarkers. Nat Commun 11, 1000 (2020). HCP5: Sun J, Zhang Z, Bao S, et alIdentification of tumor immune infiltration-associated lncRNAs for improving prognosis and immunotherapy response of patients with non-small cell lung cancerJournal for ImmunoTherapy of Cancer 2020;8:e000110.

LINC00657: Feng Q, Zhang H, Yao D, Chen WD, Wang YD. Emerging Role of Non-Coding RNAs in Esophageal Squamous Cell Carcinoma. Int J Mol Sci. 2019 Dec 30;21(1):258. doi: 10.3390/ijms21010258.

LINC00657: Qin X, Zhou M, Lv H, Mao X, Li X, Guo H, Li L, Xing H. Long noncoding RNA LINC00657 inhibits cervical cancer development by sponging miR-20a-5p and targeting RUNX3. Cancer Lett. 2020 Oct 28:S0304-3835(20)30578-4. doi: 10.1016/j.canlet.2020.10.044. **LINC00657**: Zhang XM, Wang J, Liu ZL, Liu H, Cheng YF, Wang T. LINC00657/miR-26a-5p/CKS2 ceRNA network promotes the growth of esophageal cancer cells via the MDM2/p53/Bcl2/Bax pathway. Biosci Rep. 2020;40(6):BSR20200525.

TCGA-STAD: Cancer Genome Atlas Research Network. Comprehensive molecular characterization of gastric adenocarcinoma. Nature. 2014 Sep 11;513(7517):202-9. doi: 10.1038/nature13480. TCGA.STAD MAF data: https://api.gdc.cancer.gov/data/c06465a3-50e7-46f7-b2dd-7bd654ca206b

12.4 Others

1. Newman, A. M., Liu, C. L., Green, M. R., Gentles, A. J., Feng, W., Xu, Y., … Alizadeh, A. A. (2015). Robust enumeration of cell subsets from tissue expression profiles. Nature Methods, 12(5), 453–457.

2. Vegesna R, Kim H, Torres-Garcia W, …, Verhaak R.*(2013). Inferring tumour purity and stromal and immune cell admixture from expression data. Nature Communications 4, 2612.

3. Rieder, D., Hackl, H., …, Sopper, S.* (2019). Molecular and pharmacological modulators of the tumor immune contexture revealed by deconvolution of RNA-seq data. Genome medicine, 11(1), 34.

4. Li, B., Severson, E., Pignon, J.-C., Zhao, H., Li, T., Novak, J., … Liu, X. S.* (2016). Comprehensive analyses of tumor immunity: implications for cancer immunotherapy. Genome Biology, 17(1), 174.

5. P. Charoentong et al.*, Pan-cancer Immunogenomic Analyses Reveal Genotype-Immunophenotype Relationships and Predictors of Response to Checkpoint Blockade. Cell Reports 18, 248-262 (2017).

6. Becht, E., Giraldo, N. A., Lacroix, L., Buttard, B., Elarouci, N., Petitprez, F., … de Reyniès, A*. (2016). Estimating the population abundance of tissue-infiltrating immune and stromal cell populations using gene expression. Genome Biology, 17(1), 218.

7. Aran, D., Hu, Z., & Butte, A. J.* (2017). xCell: digitally portraying the tissue cellular heterogeneity landscape. Genome Biology, 18(1), 220.

8. Racle, J., de Jonge, K., Baumgaertner, P., Speiser, D. E., & Gfeller, D*. (2017). Simultaneous enumeration of cancer and immune cell types from bulk tumor gene expression data. ELife, 6, e26476.

9. Barbie, D.A. et al (2009). Systematic RNA interference reveals that oncogenic KRAS-driven cancers require TBK1. Nature, 462(5):108-112.

10. Hänzelmann, S., Castelo, R. and Guinney, J. (2013). GSVA: Gene set variation analysis for microarray and RNA-Seq data. BMC Bioinformatics, 14(1):7.

11. Lee, E. et al (2008). Inferring pathway activity toward precise disease classification. PLoS Comp Biol, 4(11):e1000217.

12. Wang et al.,et al (2019). The UCSCXenaTools R package: a toolkit for accessing genomics data from UCSC Xena platform, from cancer multi-omics to single-cell RNA-seq. Journal of Open Source Software, 4(40), 1627

13. Helmink BA, Reddy SM, Gao J, et al. B cells and tertiary lymphoid structures promote immunotherapy response. Nature. 2020 Jan;577(7791):549-555.

14. Mariathasan S, Turley SJ, Nickles D, et al. TGF attenuates tumour response to PD-L1 blockade by contributing to exclusion of T cells. Nature. 2018 Feb 22;554(7693):544-548.

15. Kulski, J.K. Long Noncoding RNA HCP5, a Hybrid HLA Class I Endogenous Retroviral Gene: Structure, Expression, and Disease Associations. Cells 2019, 8, 480.

16. Li, Y., Jiang, T., Zhou, W. et al. Pan-cancer characterization of immune-related lncR-

NAs identifies potential oncogenic biomarkers. Nat Commun 11, 1000 (2020).

17. Sun J, Zhang Z, Bao S, et alIdentification of tumor immune infiltration-associated lncRNAs for improving prognosis and immunotherapy response of patients with non-small cell lung cancerJournal for ImmunoTherapy of Cancer 2020;8:e000110.

18. Feng Q, Zhang H, Yao D, Chen WD, Wang YD. Emerging Role of Non-Coding RNAs in Esophageal Squamous Cell Carcinoma. Int J Mol Sci. 2019 Dec 30;21(1):258. doi: 10.3390/ijms21010258.

19. Qin X, Zhou M, Lv H, Mao X, Li X, Guo H, Li L, Xing H. Long noncoding RNA LINC00657 inhibits cervical cancer development by sponging miR-20a-5p and targeting RUNX3. Cancer Lett. 2020 Oct

20. Zhang XM, Wang J, Liu ZL, Liu H, Cheng YF, Wang T. LINC00657/miR-26a-5p/CKS2 ceRNA network promotes the growth of esophageal cancer cells via the MDM2/p53/Bcl2/Bax pathway. Biosci Rep. 2020;40(6):BSR20200525.

21. Cancer Genome Atlas Research Network. Comprehensive molecular characterization of gastric adenocarcinoma. Nature. 2014 Sep 11;513(7517):202-9. doi: 10.1038/nature13480.