Cottrazm: Construct Tumor Transition Zone Microenvironment

Zhenzhen Xun, Xinyu Ding, Youqiong Ye#

2022-1-23

1. Introduction

Cottrazm is an R package to construct tumor spatial microenvironment (TSME) of spatial transcriptomic (ST) data.

Conttrazm adjusted gene expression with morphological information,[1] and based on morphological adjusted gene expression, Cottrazm applied InferCNV[2] to recognize malignant spots (Mal). Further, Cottrazm arranged spatial spots on hexagonal systems, extrapolated layer by layer from core spots of Mal and determined the identity of spots according to UMAP distance to tumor centroid as Mal or boundary spot (Bdy). When all neighbors of Mal spots are not tumor tissue, the extrapolation will be completed, the remained spots are labeled as non-malignant spots (nMal). Next based on corresponding single cell RNA sequencing (scRNAseq) data of tumor, Cottrazm generated signature score matrix from each cell type in scRNAseq, enrichment score matrix of cell types from scRNAseq in each spot by parametric analysis of gene set enrichment (PAGE) analysis, and topic combined the KNN clusters and location information (Mal, Bdy, and nMal). Then, cell types for each topic were determined based on signature score and enrichment score. Cottrazm calculated the possible proportion of each cell type for each spatial spot by dampened weighted least squares (DWLS), a

kind of deconvolution algorithm,[3] to further explore the character of boundary spots as well as malignant and non-malignant spots. Then according to deconvolution result, scRNAseq expression data, and ST expression data, Cottrazm reconstructed the cell type specific gene expression profile (GEP) at sub-spot level, to deeper exploring TSME. Features were weighted in each cell type according to the feature's contribution in each cell type in scRNAseq reference, then feature expression of sub-spot with a certain cell type was calculated by the cell proportion from deconvolution results and feature weight. Cottrazm can be followed by further analysis, including sub-spot GEP analysis, cell-cell interaction, identification of potential targets of tumor boundary.

The full Cottrazm documentation is available in the package. To reach the user's guide, install the Cottrazm package and load it into an R session by library (Cottrazm). And they can get help by help (BoundaryDefine), help (SpatialDecon), and help (SpatialRecon) to see documentation of the main function. Also description of each function is including in docs/Cottrazm_0.1.1.pdf.

2. Installation and requirement

Before installing cottrazm, please properly create conda environment install all dependencies

```
# conda create -n TumorBoundary python=3.8
# conda activate TumorBoundary
# pip install -U stlearn
```

```
library(Seurat)
library(magrittr)
library(dplyr)
library(Matrix)
library(ggplot2)
library(stringr)
```

```
library(RColorBrewer)
library(patchwork)
library(ggtree)
library(BiocGenerics)
library(readr)
library(rtracklayer)
library(infercnv)
library(phylogram)
library(utils)
library(dendextend)
library(assertthat)
library(reticulate)
library(openxlsx)
library(scatterpie)
library(cowplot)
library(stats)
library(quadprog)
library(data.table)
library(Rfast)
library(ggrepel)
library(tibble)
library(clusterProfiler)
library(utils)
library(org.Hs.eg.db)
```

Installation of Cottrazm You can install Cottrazm.tar.gz from local path or directly from github (https://github.com/Yelab2020/Cottrazm).

```
#from local path
devtools::install_github('Yelab2020/Cottrazm')
#from github
install.packages("Cottrazm.tar.gz", repos = NULL, type = "source")
library(Cottrazm)
```

3. functional modules

3.1 Delineation of tumor boundary

3.1.1 Read, preprocessed and quality control of tumor ST data

We firstly read Spaceranger result, then we examined the quality control result with parameters: nCount_Spatial, nFeature_Spatial and percentage of mitochondria genes. The feature plot and .xlsx file of quality control were saved.

```
print('STPreProcess')
InDir = paste(system.file("extdata/outs",package = "Cottrazm"),"/",sep = "")
Sample = "CRC1"
OutDir = paste(getwd(),"/",Sample,"/",sep = "")
TumorST <-
  STPreProcess(
    InDir = InDir,
    OutDir = OutDir,
    Sample = Sample
    )
```

3.1.2 Morphological adjusted cluster determination

We firstly use morphological information to adjust spatial feature expression, then the adjusted expression data was used to cluster spatial spots. A list of genes represent lymphocytes was used to score spatial spots to recognize normal spatial cluster, the cluster with the highest score of lymphocytes' features was defined as normal cluster. Then we used the barcode and the result of morphological adjusted cluster to generate cell annotation file for future use in InferCNV.

```
print('STModiCluster')
res = 1.5
TumorST <-
  STModiCluster(
    InDir = InDir,
    Sample = Sample,
    OutDir = OutDir,
    TumorST = TumorST,
    res = res
  )
```

3.1.3 Run InferCNV on ST data

We firstly use clusters with the highest score of lymphocytes' features as reference, because the CNV copy number of lymphocytes were relatively stable. Then we run InferCNV on ST data. You can select non-adjusted gene expression assay when tissue was properly permeabilized otherwise we recommend use morphological adjusted gene expression assay. When running InferCNV, the parameter "analysis_mode" was set as random_trees, the observation spots will be divided to 8 clusters, it will take longer time, but malignant and non-malignant spots will be better separated.

```
print('STCNV' )
STInferCNV <-
  STCNV(
    TumorST = TumorST,
    OutDir  = OutDir,
    assay = "Spatial"
    )
```

Then we score ST data based on result of InferCNV, visualize CNV scores of each CNV labels defined by random_tree model. Next according to CNV scores, CNV labels, and morphology information, you need to assign malignant labels.

```
print('STCNVScore')

TumorST <-
  STCNVScore(
    TumorST = TumorST,
    assay = "Spatial",
    Sample = Sample,
    OutDir = OutDir
    )
```

3.1.4 Define of tumor boundary

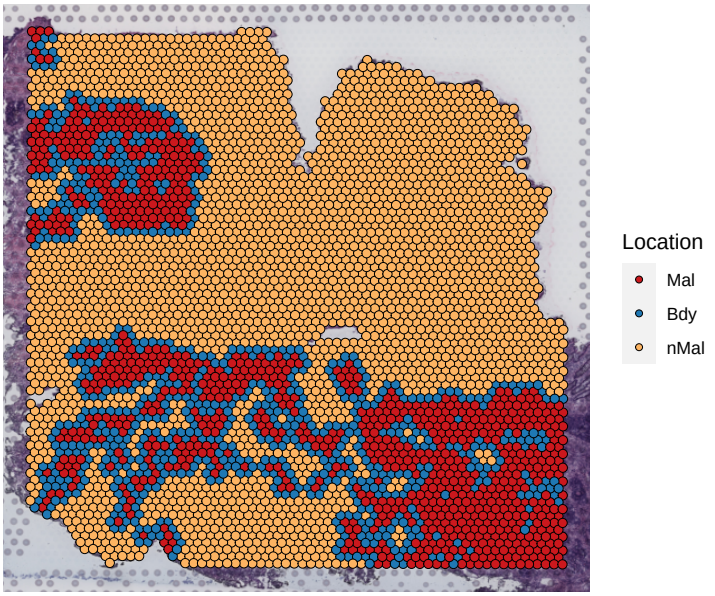
According to plots above, CNV labels 1 and 2 had the highest CNV scores and share similar morphological state, so we recognize these labels as putative malignant labels. Next, we start define the Boundary of putative malignant labels, the processed plots and data were saved in OutDir:

```
TumorSTn <-
  BoundaryDefine(
    TumorST = TumorST,
    MalLabel = c(7,8),
    OutDir = OutDir,
    Sample = Sample
  )
```

After defined the Boundary of malignant spots, we need to project the malignant spots (Mal), Boundary spots (Bdy) and non-malignant spots (nMal) to spatial HE/IF staining plot. Finally, we generated and saved a boundary defined Seurat object for future analysis.

```
TumorST <-
  BoundaryPlot(
    TumorSTn = TumorSTn,
    TumorST = TumorST,
```

```
    OutDir = OutDir,
    Sample = Sample
)
```



3.2 Deconvolution for each spot

To better understand the cell composition of tumor boundary as well as malignant and other non-malignant spots, we combined tumor ST data and tumor scRNAseq data and used deconvolution method to estimate the proportion of cell types in scRNAseq data infiltrated in each spatial spot.

3.2.1 Preporcess of scRNAseq data

Firstly, we need to generate significant gene expression file (sig_exp) and a list of markers for each scRNAseq cell type (clustermarkers_list).

```
#clustermarkers_list
sc_obj <- readr::read_rds("YourPath/scRNAseqRef.rds.gz")
clustermarkers <-
  Seurat::FindAllMarkers(object = sc_obj,
                         logfc.threshold = 0.25,
                         only.pos = TRUE)

clustermarkers_list <- split(clustermarkers, clustermarkers$cluster)
clustermarkers_list <-
  lapply(names(clustermarkers_list), function(cluster) {
    sub_markers <- clustermarkers_list[[cluster]]$gene
  })
names(clustermarkers_list) <-
  names(split(clustermarkers, clustermarkers$cluster))

#sig_exp
sig_exp <-
  get_sig_exp(
    se.obj = sc_obj,
    DefineTypes = "MajorTypes",
    sig_scran = unique(unlist(clustermarkers_list))
  )

#In this vignette we read processed clustermarkers_list and sig_exp
clustermarkers_list <- readr::read_rds(system.file("inst/extdata/clustermarkers_list.rd
sig_exp <- readr::read_rds(system.file("inst/extdata/sig_exp.rds.gz",package = "Cottraz
```

3.2.2 Enrichment analysis of ST data

Then, we preprocessed ST data to get signature score matrix and applied PAGE enrichment analysis on ST data to get the enrichment score matirx of cell types in scRNAseq data of each spatial spot.

```r
#Processed ST data
TumorST <- NormalizeData(TumorST, assay = "Spatial")
TumorST@meta.data$Decon_topics <-
  paste(TumorST@meta.data$Location,
        TumorST@meta.data$seurat_clusters,
        sep = "_")
expr_values = as.matrix(TumorST@assays$Spatial@data) #ST expr log
nolog_expr = 2 ^ (expr_values) - 1 #ST expr nolog
meta_data <-
  TumorST@meta.data[, c("nCount_Spatial", "Decon_topics", "Location")]

#Signature score
for (cluster in names(clustermarkers_list)) {
  cluster_markers = clustermarkers_list[[cluster]][1:25]
  cluster_score <-
    apply(TumorST@assays$Spatial@data[rownames(TumorST@assays$Spatial@data) %in% cluste
  meta_data <- cbind(meta_data, cluster_score)
}
colnames(meta_data) <-
  c("nCount_Spatial",
    "Decon_topics",
    "Location",
    names(clustermarkers_list))

#filter st and sc feature
intersect_gene = intersect(rownames(sig_exp), rownames(nolog_expr))
filter_sig = sig_exp[intersect_gene, ]
filter_expr = nolog_expr[intersect_gene, ]
filter_log_expr = expr_values[intersect_gene, ]

#enrichment analysis
enrich_matrix <-
  get_enrich_matrix(filter_sig = filter_sig,
```

```
                    clustermarkers_list = clustermarkers_list)
enrich_result <-
  enrich_analysis(filter_log_expr = filter_log_expr,
                  enrich_matrix = enrich_matrix)
```

3.2.3 Spot deconvolution

Next, we applied DWLS deconvolution analysis to ST data to explore the proportion of scRNAseq cell type infiltrated in each spot.

```
DeconData <- SpatialDecon(
  enrich_matrix = enrich_matrix,
  enrich_result = enrich_result,
  filter_expr = filter_expr,
  filter_sig = filter_sig,
  clustermarkers_list = clustermarkers_list,
  meta_data = meta_data,
  malignant_cluster = "Malignant epithelial cells",
  tissue_cluster = "Epithelial cells",
  stromal_cluster = "Fibroblast cells"
  )
```

This step takes long time, in this vignette we read in our deconvoluted result.

```
DeconData <- openxlsx::read.xlsx(system.file("inst/extdata/DeconData.xlsx",package = "C
```

```
#>              cell_ID          T   Myeloid   B/Plasma Fibroblast Endothelial
#> 1 AAACAAGTATCTCCCA-1 0.21696559 0.0000000 0.00000000  0.4575330   0.3255014
#> 2 AAACAATCTACTAGCA-1 0.10427466 0.0000000 0.00000000  0.7143582   0.1813672
#> 3 AAACACCAATAACTGC-1 0.00000000 0.3863709 0.00000000  0.6136291   0.0000000
#> 4 AAACAGAGCGACTCCT-1 0.00000000 0.0000000 0.08444942  0.3858252   0.0000000
#> 5 AAACAGCTTTCAGAAG-1 0.00000000 0.3665742 0.00000000  0.6334258   0.0000000
```

```
#> 6 AAACAGGGTCTATATT-1 0.02722014 0.0000000 0.00000000  0.0000000   0.0000000
#>       Tumor Epithelial
#> 1 0.0000000  0.0000000
#> 2 0.0000000  0.0000000
#> 3 0.0000000  0.0000000
#> 4 0.0000000  0.5297254
#> 5 0.0000000  0.0000000
#> 6 0.9727799  0.0000000
```

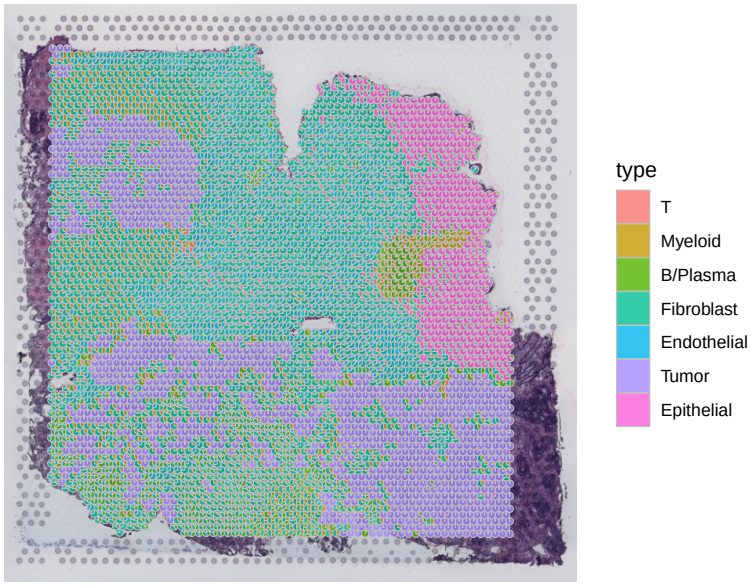
3.2.4 Visualization

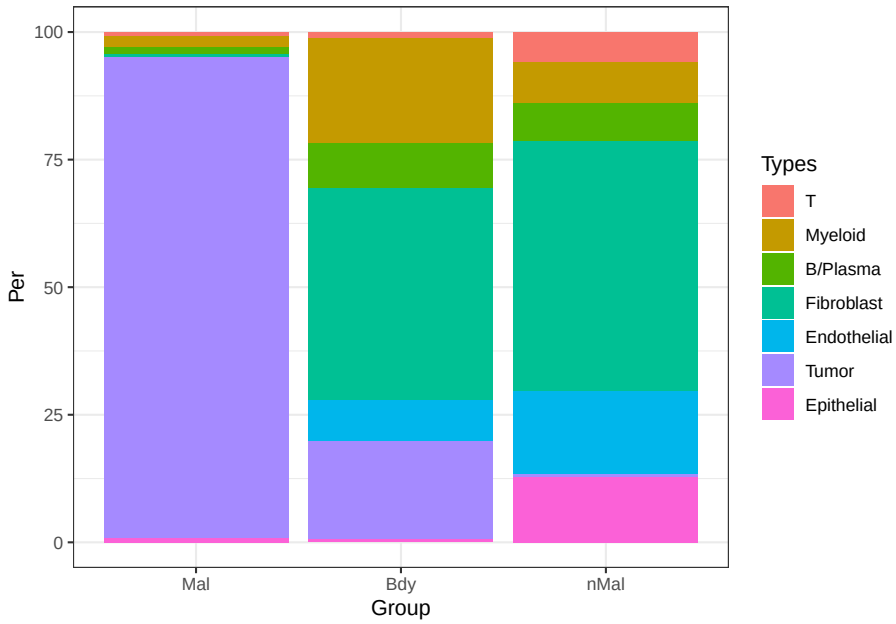
After getting the deconvolution result, we need to visualize the result with pie plot and bar plot. Pie plot was used to project the cell type infiltrated in each spot to spatial. Bar plot was used to compare cell type infiltrated in different locations including: malignant spots (Mal), boundary spots (Bdy), and non-malignant spots (nMal).

```
plot_col <- colnames(DeconData)[2:ncol(DeconData)]
img_path = system.file("inst/extdata/outs/spatial/tissue_lowres_image.png",package = "C

DeconPieplot(
  DeconData = DeconData,
  TumorST = TumorST,
  plot_col = plot_col,
  img_path = img_path,
  pie_scale = 0.4,
  scatterpie_alpha = 0.8,
  border_color = "grey"
  )

DeconBarplot(
  DeconData = DeconData,
  TumorST = TumorST,
  plot_col = plot_col
```

```
  )
```

```
#> Scale for 'y' is already present. Adding another scale for 'y', which will
#> replace the existing scale.
```





3.3 Reconstruct of tumor spatial microencironment

Based on tumor boundary define result and spatial deconvolution result, we eager to reconstruct the tumor spatial microenrironment (TSME) at sub-spot level, which made up for the problem of insufficient resolution of ST data.

3.3.1 Reconstruction of spatial gene expression profile of scRNAseq cell types for sub-spots

Finally, we reconstructed the ST gene expression profile of scRNAseq cell types at sub-spot level based on ST data, scRNAseq data and spot deconvolution result, which could be used for further analysis. In this vignette, we only reconstruct the TSME of boundary spots, which was the region most likely associated with tumor progression, drug resistance and tumor metastasis. If you want to reconstruct TSME of whole ST data, you can change parameter Location = c("Mal","Byd","nMal"), but it will take longer time.

```
TumorSTRecon <-
  SpatialRecon(
    TumorST = TumorST,
    sig_exp = sig_exp,
    clustermarkers_list = clustermarkers_list,
    DeconData = DeconData,
    Location = "Bdy"
  )
```

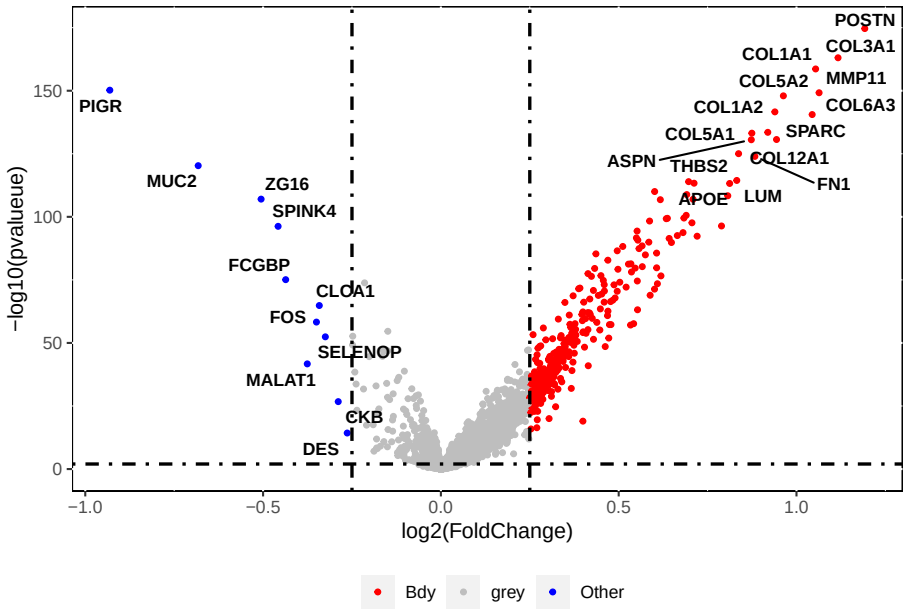
3.3.2 Differentially expressed genes of tumor Boundary

To reconstruct of tumor spatial microenvironment, we then find differential expressed genes (DEG) in boundary spots:

```
DiffGenes <- FindDiffGenes(TumorST = TumorST, assay = "Spatial")
```

Nex, we visualized boundary DEG by volcano plot. If you want to visualize other types of ST data spots, you can change parameter Location to "Mal" or "nMal".

```
DiffVolcanoplot(
  DiffGenes = DiffGenes,
  Location = "Bdy",
  cut_off_pvalue = 2,
  cut_off_logFC = 0.25,
  n = 15
)
```



After find the DEG of boundary spots, we explored the KEGG and GO pathways theses features enriched in and speculated the possibly function of tumor boundary.

```
boundary_enrich <-
  FeatureEnrichment(
    DiffGenes = DiffGenes,
    cut_off_logFC = 0.25,
    cut_off_pvalue = 0.05,
    Location = "Bdy"
  )
```

4. Summary

Cottrazm includes three main function modules (BoundaryDefine, SpaitalDecon, and SpatialRecon) which allow users to easily analys ST tumor data. Here may still be gaps in the toolkit, but we'll work on more features later. If you have any suggestions or questions, please feel free to

contact us (vera-morland@sjtu.edu.cn).

5. Session information

```
sessionInfo()
#> R version 4.1.0 (2021-05-18)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: CentOS Linux 7 (Core)
#>
#> Matrix products: default
#> BLAS/LAPACK: /usr/lib64/libopenblas-r0.3.3.so
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
#>  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
#>
#> attached base packages:
#> [1] stats4    parallel  stats     graphics  grDevices utils     datasets
#> [8] methods   base
#>
#> other attached packages:
#>  [1] org.Hs.eg.db_3.13.0   AnnotationDbi_1.54.1  Biobase_2.52.0
#>  [4] clusterProfiler_4.0.5 tibble_3.1.7          ggrepel_0.9.1
#>  [7] Rfast_2.0.6           RcppZiggurat_0.1.6    Rcpp_1.0.8.3
#> [10] data.table_1.14.2     quadprog_1.5-8        cowplot_1.1.1
#> [13] scatterpie_0.1.7      openxlsx_4.2.4        reticulate_1.22
#> [16] assertthat_0.2.1      dendextend_1.15.1     phylogram_2.1.0
#> [19] infercnv_1.8.1        rtracklayer_1.52.1    GenomicRanges_1.44.0
#> [22] GenomeInfoDb_1.30.0   IRanges_2.26.0        S4Vectors_0.30.2
```

```
#> [25] readr_2.0.0           BiocGenerics_0.38.0   ggtree_3.0.4
#> [28] patchwork_1.1.1       RColorBrewer_1.1-3    stringr_1.4.0
#> [31] ggplot2_3.3.5         Matrix_1.4-0          dplyr_1.0.9
#> [34] magrittr_2.0.3        SeuratObject_4.0.4    Seurat_4.1.0
#>
#> loaded via a namespace (and not attached):
#>   [1] scattermore_0.8              coda_0.19-4
#>   [3] tidyr_1.2.0                  bit64_4.0.5
#>   [5] knitr_1.39                   irlba_2.3.5
#>   [7] multcomp_1.4-17              DelayedArray_0.18.0
#>   [9] rpart_4.1.16                 KEGGREST_1.32.0
#>  [11] RCurl_1.98-1.6               doParallel_1.0.17
#>  [13] generics_0.1.2               callr_3.7.2
#>  [15] lambda.r_1.2.4               TH.data_1.0-10
#>  [17] RSQLite_2.2.14               usethis_2.1.6
#>  [19] shadowtext_0.0.9             RANN_2.6.1
#>  [21] future_1.26.1                enrichplot_1.12.3
#>  [23] bit_4.0.4                    tzdb_0.1.2
#>  [25] spatstat.data_2.2-0          httpuv_1.6.5
#>  [27] SummarizedExperiment_1.22.0  viridis_0.6.2
#>  [29] xfun_0.30                    hms_1.1.0
#>  [31] evaluate_0.15                promises_1.2.0.1
#>  [33] argparse_2.1.1               fansi_1.0.3
#>  [35] restfulr_0.0.13              caTools_1.18.2
#>  [37] igraph_1.2.11                DBI_1.1.2
#>  [39] htmlwidgets_1.5.4            futile.logger_1.4.3
#>  [41] reshape_0.8.8                spatstat.geom_2.4-0
#>  [43] purrr_0.3.4                  ellipsis_0.3.2
#>  [45] libcoin_1.0-9                deldir_1.0-6
#>  [47] MatrixGenerics_1.4.3         vctrs_0.4.1
#>  [49] SingleCellExperiment_1.14.1  remotes_2.4.2
#>  [51] ROCR_1.0-11                  abind_1.4-5
#>  [53] cachem_1.0.6                 withr_2.5.0
```

```
#>  [55] ggforce_0.3.3                sctransform_0.3.3
#>  [57] GenomicAlignments_1.28.0     treeio_1.16.2
#>  [59] prettyunits_1.1.1            goftest_1.2-3
#>  [61] DOSE_3.18.3                  cluster_2.1.3
#>  [63] ape_5.6-2                    lazyeval_0.2.2
#>  [65] crayon_1.5.1                 labeling_0.4.2
#>  [67] edgeR_3.34.1                 pkgconfig_2.0.3
#>  [69] tweenr_1.0.2                 nlme_3.1-157
#>  [71] pkgload_1.2.4                devtools_2.4.3
#>  [73] rlang_1.0.2                  globals_0.15.0
#>  [75] lifecycle_1.0.1              miniUI_0.1.1.1
#>  [77] downloader_0.4               sandwich_3.0-1
#>  [79] rprojroot_2.0.3              polyclip_1.10-0
#>  [81] matrixStats_0.62.0           lmtest_0.9-40
#>  [83] aplot_0.1.1                  zoo_1.8-10
#>  [85] ggridges_0.5.3               processx_3.7.0
#>  [87] png_0.1-7                    viridisLite_0.4.0
#>  [89] rjson_0.2.21                 bitops_1.0-7
#>  [91] KernSmooth_2.23-20           Biostrings_2.60.2
#>  [93] blob_1.2.3                   qvalue_2.24.0
#>  [95] coin_1.4-2                   parallelly_1.31.1
#>  [97] spatstat.random_2.2-0        gridGraphics_0.5-1
#>  [99] scales_1.2.0                 memoise_2.0.1
#> [101] plyr_1.8.7                   ica_1.0-2
#> [103] gplots_3.1.3                 zlibbioc_1.38.0
#> [105] compiler_4.1.0               BiocIO_1.2.0
#> [107] fitdistrplus_1.1-8           Rsamtools_2.8.0
#> [109] cli_3.3.0                    XVector_0.32.0
#> [111] listenv_0.8.0                pbapply_1.5-0
#> [113] ps_1.7.0                     formatR_1.12
#> [115] MASS_7.3-57                  mgcv_1.8-40
#> [117] tidyselect_1.1.2             stringi_1.7.6
#> [119] highr_0.9                    GOSemSim_2.18.1
```

```
#> [121] yaml_2.3.5                  locfit_1.5-9.5
#> [123] grid_4.1.0                  fastmatch_1.1-3
#> [125] tools_4.1.0                 future.apply_1.9.0
#> [127] rstudioapi_0.13             foreach_1.5.2
#> [129] rjags_4-10                  gridExtra_2.3
#> [131] farver_2.1.0                Rtsne_0.15
#> [133] ggraph_2.0.5                digest_0.6.29
#> [135] shiny_1.7.1                 later_1.3.0
#> [137] RcppAnnoy_0.0.19            httr_1.4.3
#> [139] colorspace_2.0-3            brio_1.1.3
#> [141] XML_3.99-0.9                fs_1.5.2
#> [143] tensor_1.5                  splines_4.1.0
#> [145] uwot_0.1.11                 yulab.utils_0.0.4
#> [147] tidytree_0.3.5              spatstat.utils_2.3-1
#> [149] graphlayouts_0.7.1          phyclust_0.1-30
#> [151] ggplotify_0.1.0             plotly_4.10.0
#> [153] sessioninfo_1.2.2           xtable_1.8-4
#> [155] jsonlite_1.8.0              futile.options_1.0.1
#> [157] tidygraph_1.2.0             modeltools_0.2-23
#> [159] ggfun_0.0.4                 testthat_3.1.4
#> [161] R6_2.5.1                    pillar_1.7.0
#> [163] htmltools_0.5.2             mime_0.12
#> [165] glue_1.6.2                  fastmap_1.1.0
#> [167] BiocParallel_1.26.2         codetools_0.2-18
#> [169] fgsea_1.18.0                pkgbuild_1.3.1
#> [171] mvtnorm_1.1-2               utf8_1.2.2
#> [173] lattice_0.20-45             spatstat.sparse_2.1-1
#> [175] leiden_0.4.2                gtools_3.9.2.1
#> [177] GO.db_3.13.0                zip_2.2.0
#> [179] survival_3.3-1              limma_3.48.3
#> [181] rmarkdown_2.16              desc_1.4.1
#> [183] munsell_0.5.0               DO.db_2.9
#> [185] fastcluster_1.2.3           GenomeInfoDbData_1.2.6
```

```
#> [187] iterators_1.0.14            reshape2_1.4.4
#> [189] gtable_0.3.0                spatstat.core_2.4-4
```

6. Reference

1. Pham, D. *et al.* stLearn: integrating spatial location, tissue morphology and gene expression to find cell types, cell-cell interactions and spatial trajectories within undissociated tissues. *undefined* (2020) doi:10.1101/2020.05.31.125658.

2. Tirosh, I. *et al.* Dissecting the multicellular ecosystem of metastatic melanoma by single-cell RNA-seq. *Science (New York, N.Y.)* **352**, 189–196 (2016).

3. Dong, R. & Yuan, G. C. SpatialDWLS: accurate deconvolution of spatial transcriptomic data. *Genome Biology* **22**, 1–10 (2021).