

Supporting data and code for "Comparative analysis of cell type-specific gene regulation in early branching metazoans".

The code has been only tested on CentOS Linux release 7.3.1611.

1. Content

Scripts/ Includes the version of the MetaCell package we used.

UMI_tables/ Contains the single cell gene molecule counts per MARS-seq batch.

Annotation_and_config_files/ For each species (sp), includes:

- sp_MARS_Batches: MARSseq batches table.
- sp_cell_metadata: cell barcodes, batch, and plate coordinates.
- sp_gene_ages: Gene age definition, based on OrthoMCL clustering and parsimony (as described in Materials and Methods).
- sp_gene_annotation: Different custom functional annotations (including Pfam (pfam.xfam.org) domain architectures and Blast Best Hits against different species)
- sp_TF_annotation: Annotation of transcription factors, including name, TF class and color-code used in the paper.
- sp_black_list: List of blacklisted genes for clustering

Anlalysis/ - include R object with the results of our MetaCell and clustering analysis

- sp_cluster_object.Rdata - the S4 class object for the species (sp). See below on how to use it.

2. Using the analysis files

#Prerequisites.

R packages: `install.packages('<package_name>')`

devtools #IMPORTANT NOTE: to install/use devtools, you may need to install (with admin privileges) the libraries: `sudo yum install <library_name>`

- `openssl-devel`
- `libcurl-devel`
- `libxml2-devel`

roxygen2
dplyr
data.table
pdist
pheatmap
plotrix
zoo
matrixStats
glus
doMC
reshape2
xlsx

Bioconductor packages: `source("https://bioconductor.org/biocLite.R")`
`biocLite('package_name')`

graph
Rgraphviz

To load the needed libraries for interpreting MetaCell object (tested on R version 3.3.2 and 3.4.1):

```
library("devtools")
load_all("../Scripts/MetaCell/")
library("methods")
library("Matrix")
library("matrixStats")
library("plyr")
```

To load the S4 class objects for species "sp" (e.g. mnemiopsis):

```
load("Analysis/Mnemiopsis_cluster_object.Rdata")
```

This gives you two objects. The first is the clustering object (e.g. mnemiopsis_clust) and, among others, it has the following important slots:

mnemiopsis_clust@scmat@mat	#Gene (rows) cell (columns) UMI counts (After filtering small cells and bad meta-cells, as described in Materials and Methods)
mnemiopsis_clust@scmat@cell_metadata	#Cell metadata
mnemiopsis_clust@clusts	#Cluster affiliation of each single cell.
mnemiopsis_clust@feat_mat	#UMI counts for the selected marker genes
mnemiopsis_clust@clust_fp	#Regularized fold change enrichment for each gene (row) on each cell cluster (column)

You can explore the umi table distribution, or the enrichment of specific genes in the clust_fp data frame.

For example, to generate a bargraph showing enrichment of a gene of interest over cell clusters, use:

```
barplot(mnemiopsis_clust@clust_fp["ML02275a",], las=2, col="gray30")
```

The second is the 2d object (e.g. mnemiopsis_2d) and, among others, it has the following important slots:

mnemiopsis_2d@x_cl	#X coordinates of the clusters 2d projection
mnemiopsis_2d@y_cl	#Y coordinates of the clusters 2d projection
mnemiopsis_2d@x	#X coordinates of the cells 2d projection
mnemiopsis_2d@y	#Y coordinates of the cells 2d projection
mnemiopsis_2d@cl_ord	#cluster ordering (as used in the paper)

To visualize a gene projection in 2D, use:

```
scp_plot_gene_2d(mnemiopsis_2d, "ML02275a", w=800, h=800, base_dir=".", reg_factor=4, cont_levels=2)
```

3. Rerunning the analysis from raw data.

Use the following pipeline to reproduce our analysis, note that minor differences in the derived clustering are expected due to the randomized nature of the algorithms.

```

#Source scripts
library("devtools")
load_all("./Scripts/MetaCell/")
library("methods")
library("Matrix")
library("matrixStats")
library("plyr")

#1. Load UMI tables, filter small cells and, optionally, background noise.
sc_mat = sc_pipe_clean(index_fn =
"Annotation_and_config_files/Trichoplax/Trichoplax_MARS_Batches.txt",
  base_dir = "./UMI_tables/Trichoplax/",
  mark_blacklist_terms=scan("./Annotation_and_config_files/Trichoplax/
Trichoplax_black_list",what=""),
  batch_meta_attr = "MARS_BATCH",
  mark.sz_cor_norm_max= -0.05,
  mark.niche_T = 0.05,
  mark.min_tot=100,
  mark.min_var_mean = 5,
  clust_knn=150,
  min_umi_n = 100,
  max_umi_n=5000,
  amb_epsilon=0.03,
  min_clust_size = 30,
  sample_n_batches = NA,
  filt_amb_on_clusts=T,
  filt_outliers_on_clusts=F)

#2. Marker gene selection and meta-cell analysis
sc_cl = sc_pipe_cluster(sc_mat,
  mark.sz_cor_norm_max= -0.05,
  mark.niche_T = 0.05,
  mark.min_var_mean = 5,
  mark.min_tot=100,
  clust_knn=150,
  mark_blacklist_terms=scan("./Annotation_and_config_files/Trichoplax/
Trichoplax_black_list",what=""),
  min_clust_size = 30,
  filt_outliers_on_clusts=F,
  clust_fp_metadata_fields = NA)

#3. Plotting
sc_pipe_plots(sc_cl,
  focus_tfs_fn=NA,
  mark_blacklist_terms=scan("./Annotation_and_config_files/Trichoplax/
Trichoplax_black_list",what=""),
  force_max_deg_2dproj = 8,
  store_rda_fn="Plotting_data.Rda",
  T_edge_asym = F,
  K_2dproj = 60,
  K_cells_2dproj = 30,
  T_edge_2dproj = 0.05,
  restrict_edges_by_fpcor=T)

```

#4. Bootstrapping.

Note: modify number of cpu according to your system (n_cpu). There is a non-parallelized version of this function: `tg_piecewise_knn_graph_cover_bootstrap`

```
boot1000=tg_piecewise_knn_graph_cover_bootstrap_parallel(t(as.matrix(sc_cl@feat_mat)),k_knn=115,min_clust_size=22,boot_ratio=0.75,N_boot=1000,n_cpu=20)
```

Returns a list with:

```
boot1000$coclust      #For each cell pair, number of times they coclustered.
boot1000$num_trials   #For each cell pair, number of times they have been
                      included in a test (max boot_ratio*N_boot)
```

4. Downstream analysis examples (based on `sc_cl` object)

```
source("Scripts/Downstream_analysis.R")
```

```
load("Plotting_data.Rda")
```

```
##Automatic cluster ordering, can be reordered manually.
```

```
cluster_order=as.character(sc_2d@cl_ord)
```

```
##Plot single-cell profiles for top cluster markers
```

```
scr_plot_cmod_markers(sc_cl,output_file="sc_marker_expression_map.png",gene_annotation_file="Annotation_and_config_files/Trichoplax/Trichoplax_gene_annotation",black_list=scan("./Annotation_and_config_files/Trichoplax/Trichoplax_black_list",what=""),gene_min_fold=3,per_clust_genes=35,h=4000,clust_ord=cluster_order,transversality_N=17)
```

```
##TF maps
```

```
scr_tf_tf_cor_footprint(footprint=sc_cl@clust_fp,sc_object=sc_cl,tf_file="Annotation_and_config_files/Trichoplax/Trichoplax_TF_annotation",min_FC=1.7,nmodules=30,fuse_modules2=T,T_totumi=15,h=2500,w=2000,pmin=2.5,cor_zlim=0.75)
```

```
##Gene age distribution analysis
```

```
scr_phylostratigraphy_cell_modules(sc_cl,footprint=sc_cl@clust_fp,PS_order=c("Eukaryota", "Holozoa", "Metazoa", "Parahoxozoa", "Trichoplax"),PS_table_file="Annotation_and_config_files/Trichoplax/Trichoplax_gene_ages",fc_threshold=1.8,T_totumis_seen=20)
```

```
##Export cluster annotation table:
```

```
scr_export_cluster_annotation(sc_cl,excel_fn="Trichoplax_cluster_annotation.xlsx",gene_annot_file="Annotation_and_config_files/Trichoplax/Trichoplax_gene_annotation")
```

Note that code for motif finding and cross-validation is computationally heavier and is therefore not provided. See methods for description of the procedure, which is based on standard tools.