An Introduction to Transcriptome Overlap Measure (TROM)

Wei Vivian Li and Jingyi Jessica Li

August 29, 2016

Contents

1 Introduction

The Transciptome Overlap Measure (TROM) package aims to find the similarity/correspondence of (i.e., map) transcriptomes of two biological samples within or between different species. This package provides a quantitative measure of transcriptome similarity based on the overlap of "associated genes", which are defined as the genes that capture specific transcriptional activities of a biological sample. Given two sets of associated genes of two samples from the same or different species, a hypergeometric test is carried

out to test the significance of their overlap. The package can automatically select associated genes based on an input gene expression matrix and it also allows users to input their own associated gene lists, which they think can represent the transcriptional characteristics of different samples. A function is provided to calculate TROM scores between different samples based on the associated gene lists. The package also provides a heatmap function to visualize the calculated TROM scores and the resulting mapping patterns between different samples. In addition, the package contains a function to find the enriched Gene Ontology (GO) terms or GO slim terms (cut-down versions of the GO ontologies containing a subset of the terms in the whole GO, giving a broad overview of the ontology content without the detail of the specific fine grained terms) in the associated genes, overlapping associated genes, or any user-specified genes, so users can interpret the observed transcriptome mapping patterns by looking into the corresponding overlapping genes' biological functions.

Figure 1 gives an outline of main functions in package TROM.

2 Setup

The package TROM depends on the following packages.

```
library(lattice)
library(gplots)
library(RColorBrewer)
library(gtools)
library(openxlsx)
library(topGO)
library(GO.db)
```

In case that `TROM` cannot be successfully installed, please check the availability of Bioconductor packages `topGO` and `GO.db`. These two packages can be installed using the following code:

```
source("https://bioconductor.org/biocLite.R")
biocLite(c("topGO", "GO.db"))
```

To run the functions related to finding GO terms (`find.top.GO.terms`) or GO slim terms (`find.top.GO.slim.terms`), users need to have Python pre-installed in their operating systems.

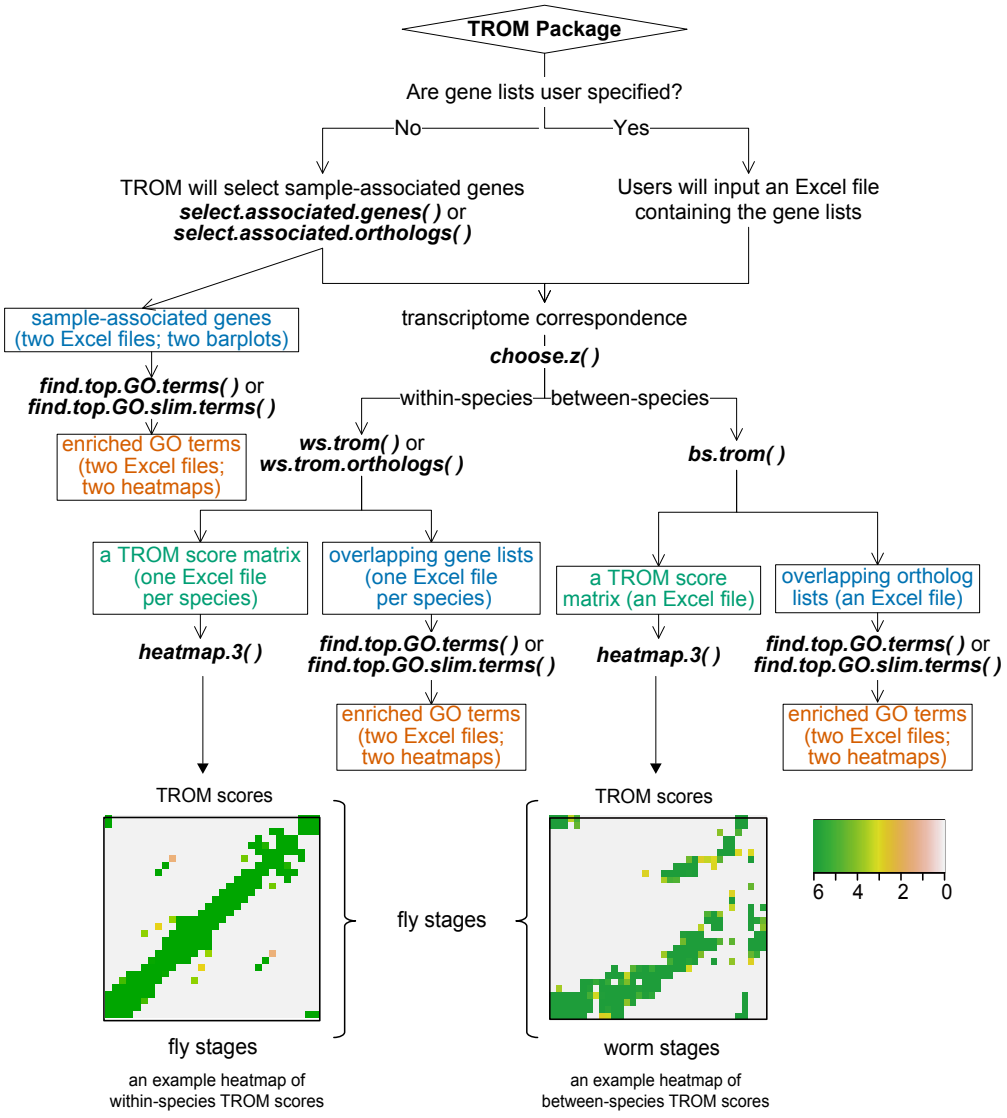

Figure 1: Outline of package TROM

3 Within-species TROM scores

We illustrate the calculation of within-species TROM in the example of comparing developmental stages of *D. melanogaster*. The gene expression data (in FPKM units) of *D.melanogaster* can be downloaded and unzipped from `http://www.stat.ucla.edu/~jingyi.li/software-and-data/trom.html`. After users move the file into `R`'s working directory, the data can be imported into `R` through

```
load("dm_gene_expr.rda")
```

`dm_gene_expr` is a data frame with 15095 observations on 31 variables. The first column is gene IDs, and the rest columns represent stages with column names as corresponding stage names. Other gene expression datasets, for which the package is to be used, should have the same format as `dm_gene_expr`.

3.1 Select genes for transcriptome mapping

The first step is to select genes for transcriptome mapping.

If users want to use their own gene lists that they think can represent the characteristics of different stages instead of using the stage-associated genes automatically selected by TROM, they need to prepare a one-sheet Excel file. In the Excel file, rows represent gene IDs and columns represent biological samples. Each column of the file stores the user-provided genes corresponding to the sample of that column. Please note that different columns may have different numbers of rows, i.e., different samples may have different numbers of genes.

Otherwise, the TROM package can select associated genes for the users based on the input gene expression data by default. Associated genes of a sample are defined by the following criterion: the genes that have expression units (e.g. FPKM, RPKM) $\geq a$ and Z-score (normalized expression units) $\geq b$ in the sample. Stage-associated genes are relatively highly expressed at that stage compared to other stages throughout development. Users can set their own `z_thre` (i.e., b: a threshold on the Z-score) according to their needs.

```
dm_associated_genes <- select.associated.genes(
                       sp_gene_expr = dm_gene_expr,
                       z_thre = 1.5, save = TRUE,
                       plot_distribution = TRUE)
```

In the example above, we set `z_thre=1.5`. The function `select.associated.genes()` returns IDs of the selected stage-associated genes of *D. melanogaster*. If `save = TRUE`, this function also saves the results to an Excel file named "associated genes.xlsx" in R's working directory (Users can find R's working directory by `getwd()`). We can also see the distribution of the number of associated genes of different samples by setting `plot_distribution = TRUE` (defaults to FALSE). Then this function outputs a barplot in a pdf file named "number of sample associated genes.pdf". In the barplot, the height of each bar represents the number of associated genes in a biological sample; each bar is divided into sub bars according to the number of samples that subsets of the genes are associated with. Please note that if users are not interested in which genes are specifically stage-associated but just want to directly calculate the TROM scores, they can skip this step and directly move on to the next step.

3.2 Calculate the within-species TROM scores

The second step is to calculate the within-species TROM scores. TROM score = -$\log_{10}$(Bonferroni-corrected p-value from a hypergeometric test).

Below is an example to calculate the TROM scores among different developmental stages of *D. melanogaster*. In this case, we set `provide = FALSE`, assuming the users have skipped the gene selection step described in Section 3.1 and letting the function automatically select associated genes based on the criterion: Z-scores $\geq$ 1.5.

```
dm_trom <- ws.trom(sp_gene_expr = dm_gene_expr,
                   z_thre = 1.5, provide = FALSE,
                   save_overlap_genes = FALSE)
```

If users have already completed the first step, they will have the file "associated genes.xlsx", which stores the gene lists for every sample of *D. melanogaster*. Or the users may have an Excel file containing their own gene lists, then users can calculate the TROM scores by providing this Excel file to `ws.trom()`.

```
dm_trom2 <- ws.trom(provide = TRUE,
                    gene_lists = "associated genes.xlsx",
                    save_overlap_genes = FALSE)
```

Similarly, if users want to calculate within-species TROM scores using only orthologous genes, they can follow the procedures below. Here we calculate the TROM scores within the species *D. melanogaster* using its

genes that have orthologous genes in the other species *C. elegans* (see also: Section 4.1).

```
load("dm_ce_orthologs.rda")
dm_trom_orth <- ws.trom.orthologs(
                sp1_sp2_orthologs = dm_ce_orthologs,
                sp_gene_expr = dm_gene_expr,
                z_thre = 1.5, i = 1,
                save_overlap_genes = FALSE)
```

The TROM package also supports functionality to compare the biological samples of the same species between two different datasets (eg. from two experiments or two laboratories). In this case, users should specify `single = FALSE` in `ws.trom()` or `ws.trom.orthologs()`. Meanwhile, users should either provide another gene expression dataset as `sp_gene_expr2` or provide `gene_lists`. If provided, `gene_lists` should be a two-sheet Excel file with the first sheet for one dataset and the second sheet for the other dataset. In each sheet, rows represent gene ids and columns represent biological samples. Each column of the file stores the user-provided genes corresponding to the sample of that column. Below is an example to compare two gene expression datasets within *D. melanogaster*.

```
dm_gene_expr2 <- dm_gene_expr[ ,1:13]
dm_trom3 <- ws.trom(sp_gene_expr = dm_gene_expr, single = FALSE,
                        sp_gene_expr2 = dm_gene_expr2, z_thre = 1.5,
                        provide = FALSE, save_overlap_genes = FALSE)
```

In the examples above, we set `save_overlap_genes = FALSE`. However, if users are interested in the overlap genes between two biological samples in the hypergeometric test, please specify `save_overlap_genes = TRUE`. Then `ws.trom()` saves the overlap genes to an Excel file named "within-species overlapping genes between sample pairs.xlsx"; `ws.trom.orthologs()` saves the overlap orthologs to an Excel file named "within-species overlapping genes (within ortholog genes) between sample pairs.xlsx". Please note that the default setting is `FALSE`.

3.3 Choose a threshold of Z-scores

If users do not have any preference on the threshold of Z-scores (`z_thre`) for selecting associated genes, they can use `choose.z` to calculate the suggested `z_thre` for the species and then tune this value based on the TROM scores and the resulting mapping pattern. `choose.z` calculates the TROM scores

for the within-species comparison using different `z_thre` ranging from -2 to 3 and selects the one that gives the most sparse but still stable correspondance map of the transcriptomes. For instance, we can find the suggested `z_thre` for *D. melanogaster* through the following code:

```
z_thre <- choose.z(dm_gene_expr)
```

3.4 Plot the within-species TROM scores

The package TROM contains a function `heatmap.3()` specifically designed to plot within-species or between-species TROM scores. Larger TROM scores are shown in darker colors, corresponding to a scale allowing saturation of the scores, with a default saturation level at 6 (any TROM scores larger than 6 are shown as 6).

The major input argument of `heatmap.3()` is the output of `ws.trom()`, `ws.trom.orthologs()` or `bs.trom()`. In section 3.2, we obtain `dm_trom`, a square matrix of within-species TROM scores, where rows and columns correspond to the samples of *D. melanogaster* respectively. Now we can illustrate the TROM scores using `heatmap.3` and the output heatmap is shown in Figure 2.

```
dm_trom <- ws.trom(sp_gene_expr = dm_gene_expr,
                   z_thre = 1.5, provide = FALSE,
                   save_overlap_genes = FALSE)
heatmap.3(   dm_trom,
             Rowv = NULL,
             Colv = NULL,
             dendrogram = c("none"),
             distfun = dist,
             hclustfun = hclust,
             xlab = "",
             ylab = "",
             main = "D. melanogaster Stage Mapping",
             key = TRUE,
             keysize = 1,
             trace = "none",
             density.info = c("none"),
             col = terrain.colors(120),
             max_score = 6
  )
```

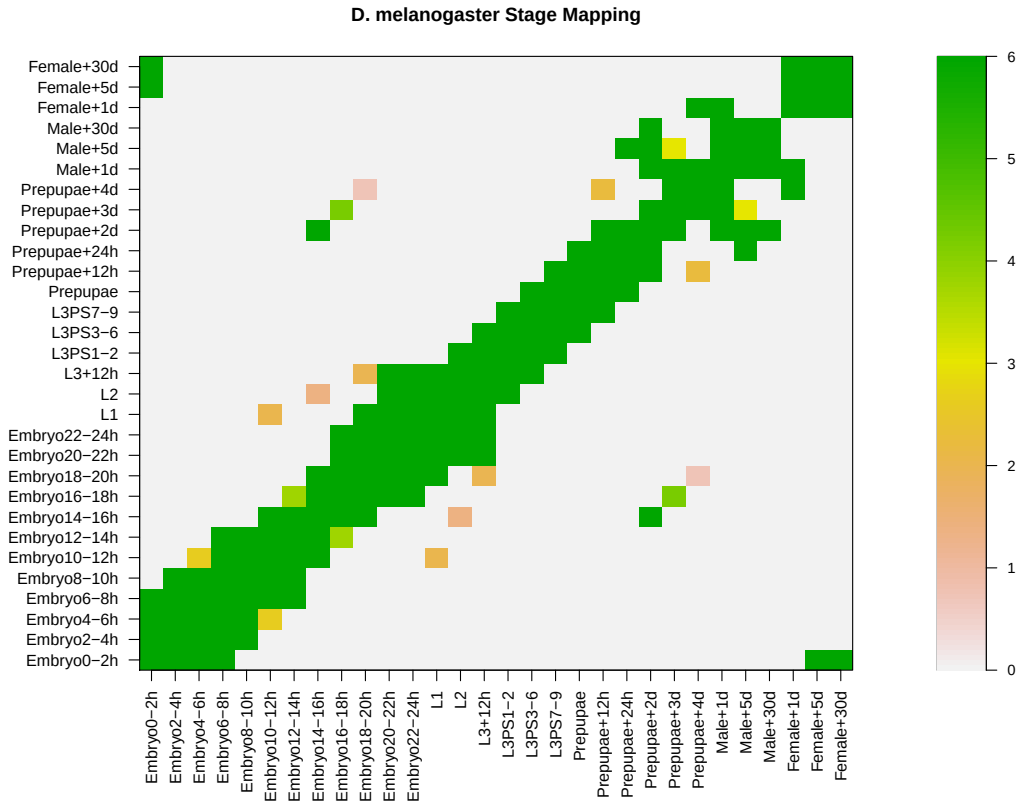

Figure 2: Transcriptome mapping of developmental stages within D. melanogaster

The users can also draw a heatmap with dendrograms using `heatmap.3()`. We present this functionality through the transcriptome mapping of tissues/cell lines within *D. melanogaster*. Users can download the data set for *D. melanogaster* gene expression estimates in 29 fly tissues and 19 fly cell lines from `http://www.stat.ucla.edu/~jingyi.li/data/fly-worm/dm_tissuecell_FPKMs.txt.bz2` and import the data into R using

```
dm_tissue_gene_expr <- read.table("dm_tissuecell_FPKMs.txt", header=TRUE)
```

Now we have the object `dm_tissue_gene_expr` in R's working directory. It is a data frame with 15054 observations on 49 variables, with rows representing genes, the first column as gene IDs and the rest columns as tissues/cell lines. If the users want to draw a heatmap of the TROM scores for tissues/cell lines mapping and to find possible clustering of the tissues/cell lines (using hierarchical clustering on the heatmap), they can specify the arguments as below. The output heatmap is shown in Figure 3.

```
dm_tissue_trom <- ws.trom(
                  sp_gene_expr = dm_tissue_gene_expr,
                  z_thre = 1.5, provide = FALSE,
                  save_overlap_genes = FALSE)
heatmap.3(   dm_tissue_trom,
             Rowv = TRUE,
             Colv = TRUE,
             dendrogram = c("row"),
             distfun = dist,
             hclustfun = hclust,
             xlab = "",
             ylab = "",
             main = "D. melanogaster Tissues/cell lines Mapping",
             key = TRUE,
             keysize = 1,
             trace = "none",
             density.info = c("none"),
             col = terrain.colors(120)
  )
```

4 Between-species TROM scores

We illustrate the calculation of between-species TROM scores through the comparison of the developmental stages between *D. melanogaster* and *C.*

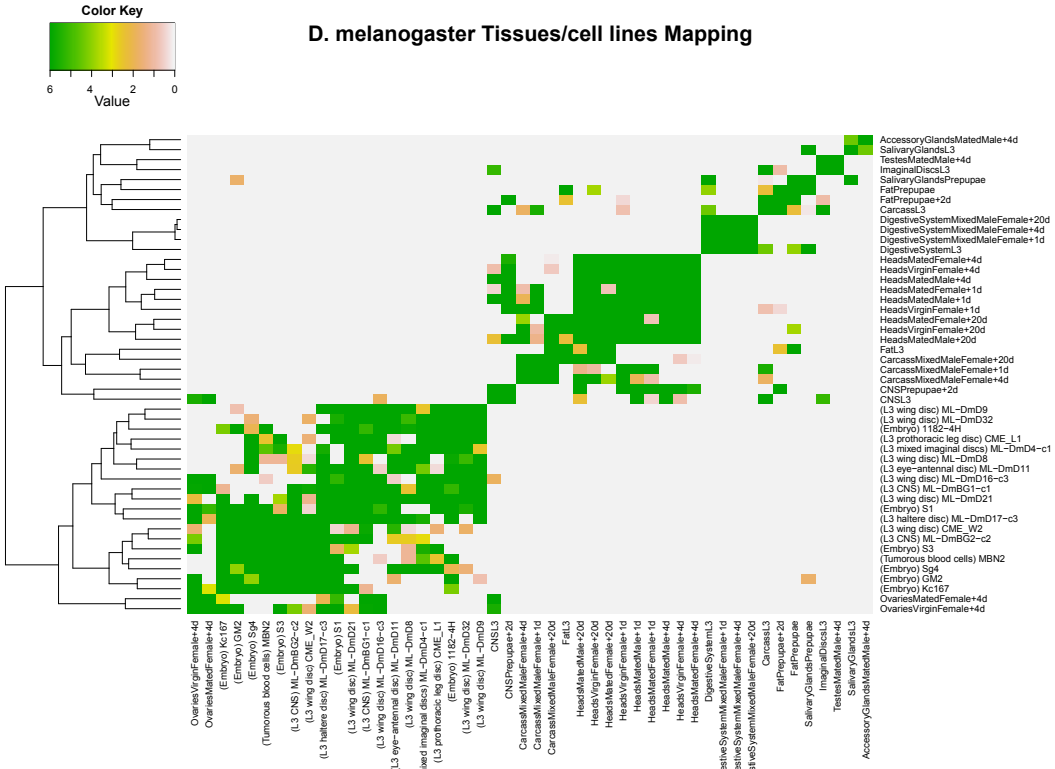

Figure 3: Transcriptome mapping of tissues/cell lines within D. melanogaster (with dendrogram)

elegans on the basis of shared orthologs in their stage-associated genes.

4.1 Select genes for transcriptome mapping

The first step is to select genes for between-species transcriptome mapping.

If users want to use their own gene lists that they think can represent the characteristics of different stages instead of using the stage-associated genes automatically selected by TROM, they need to prepare a two-sheet Excel file with the first sheet for species 1 and the second sheet for species 2. In each sheet, rows represent gene IDs and columns represent biological samples. Each column of the sheets stores the user-provided genes corresponding to the sample of that column. Please note that different columns may have different numbers of rows, i.e., different samples may have different numbers of genes.

Otherwise, the TROM package can select associated ortholog genes for each species respectively. The procedure to select associated ortholog genes is similar to that we use to select associated genes. The difference is that users need to first obtain ortholog gene pairs between the species of interest. As introduced in Section 3.2, we use `dm_ce_orthologs`, which stores ortholog gene pairs between species *D. melanogaster* and *C. elegans* (This dataset can also be downloaded and unzipped from `http://www.stat.ucla.edu/~jingyi.li/packages/TROM/TROM_Rdata.zip`). Then users can select the associated orthologs using the same criterion: Z-score $\geq a$. Here we used $a = 1.5$ and $b = 1.0$ in the example below.

```
load("dm_gene_expr.rda")
load("ce_gene_expr.rda")
load("dm_ce_orthologs.rda")
dm_associated_orthologs <- select.associated.orthologs(
                           sp_gene_expr = dm_gene_expr,
                           sp1_sp2_orthologs = dm_ce_orthologs,
                           z_thre = 1.5, i = 1, save = TRUE)
ce_associated_orthologs <- select.associated.orthologs(
                           sp_gene_expr = ce_gene_expr,
                           sp1_sp2_orthologs = dm_ce_orthologs,
                           z_thre = 1.5, i = 2, save = FALSE)
```

Among the arguments, `sp1_sp2_orthologs` takes a data frame containing ortholog gene pairs between species 1 and 2. Taking `dm_ce_orthologs` as an example, it is a data frame with 31622 observations on 2 variables,

one for Flybase gene IDs and the other for Wormbase gene IDs. `i` is an integer specifying which column of `sp1_sp2_orthologs` is the species the users want to select associated genes for: 1 for the first column and 2 for the second column. The function `select.associated.orthologs()` returns the IDs of the selected associated orthologs. If `save = TRUE`, this function also saves the results to an excel file named "associated genes within ortholog genes.xlsx" in R's working directory. As in Section 3.1, users can specify `plot_distribution = TRUE` if they want to see distribution of the number of associated orthologs of different samples. The output barplot is in a pdf file named "number of sample associated orthologous genes.pdf". Please note that if the users are not interested in which ortholog genes are specifically stage-associated and just need the between-species TROM scores, they can skip this step and directly move on to the next step.

4.2 Calculate the between-species TROM scores

The second step is to calculate the between-species TROM scores. TROM score = -$\log_{10}$(Bonferroni-corrected p-value from a hypergeometric test).

If `provide = FALSE`, which means users do not provide their own gene lists, the function `bs.trom()` can automatically select associated orthologs based on the criterion: Z-scores $\geq$ `z_thre`. The code below calculates the TROM scores of the comparison between the developmental stages of *D. melanogaster* and *C. elegans*.

```
dm_ce_trom <- bs.trom(sp1_gene_expr = dm_gene_expr,
                      sp2_gene_expr = ce_gene_expr,
                      sp1_sp2_orthologs = dm_ce_orthologs,
                      z_thre = 1.5, provide = FALSE,
                      save_overlap_genes = FALSE)
```

If users want to use their own gene lists instead of letting `bs.trom()` choose associated genes, please specify `provide = TRUE` and specify `gene_lists` as a path to the excel file prepared in Section 4.1. Suppose we have the gene lists stored in an Excel file named "bs_genes.xlsx" and the Excel file is in R's working directory. Then we can calculate between-species TROM scores in the following way:

```
dm_ce_trom_2 <- bs.trom(provide = TRUE,
                      sp1_sp2_orthologs = dm_ce_orthologs,
                      gene_lists="bs_genes.xlsx",
                      save_overlap_genes = FALSE)
```

If users want to obtain overlap orthologs between every two samples from different species, please specify `save_overlap_genes = TRUE`. Then `bs.trom()` saves the overlap orthologs of species 1 to an Excel file named "between-species overlapping genes (of species 1) between sample pairs.xlsx" and the overlap orthologs of species 2 to "between-species overlapping genes (of species 2) between sample pairs.xlsx".

4.3 Plot the between-species TROM scores

As described in Section 3.3, we can plot the between-species TROM scores using `heatmap.3()`. As an example, the code below plots the between-species overlapping result `dm_ce_trom`, which is obtained in Section 4.2. The resulting heatmap is shown in Figure 4 with x-axis representing developmental stages of *C. elegans* and y-axis developmental stages of *D. melanogaster*.

```
heatmap.3( dm_ce_trom,
           max_score = 6,
           Rowv = NULL,
           Colv = NULL,
           dendrogram = c("none"),
           distfun = dist,
           hclustfun = hclust,
           xlab = "worm stages",
           ylab = "fly stages",
           main = "D. melanogaster vs. C. elegans Stage Mapping",
           key = TRUE,
           keysize = 1,
           trace = "none",
           density.info = c("none"),
           col = terrain.colors(120)
  )
```

5 Find enriched gene ontology (GO) terms

As shown in Figure 1, we can obtain the following Excel files containing gene lists in within-species or between-species transcriptome mapping:

- For each species, we have "associated genes.xlsx" from `select.associated.genes()` or "associated genes within ortholog genes.xlsx" from `select.associated.orthologs()`.

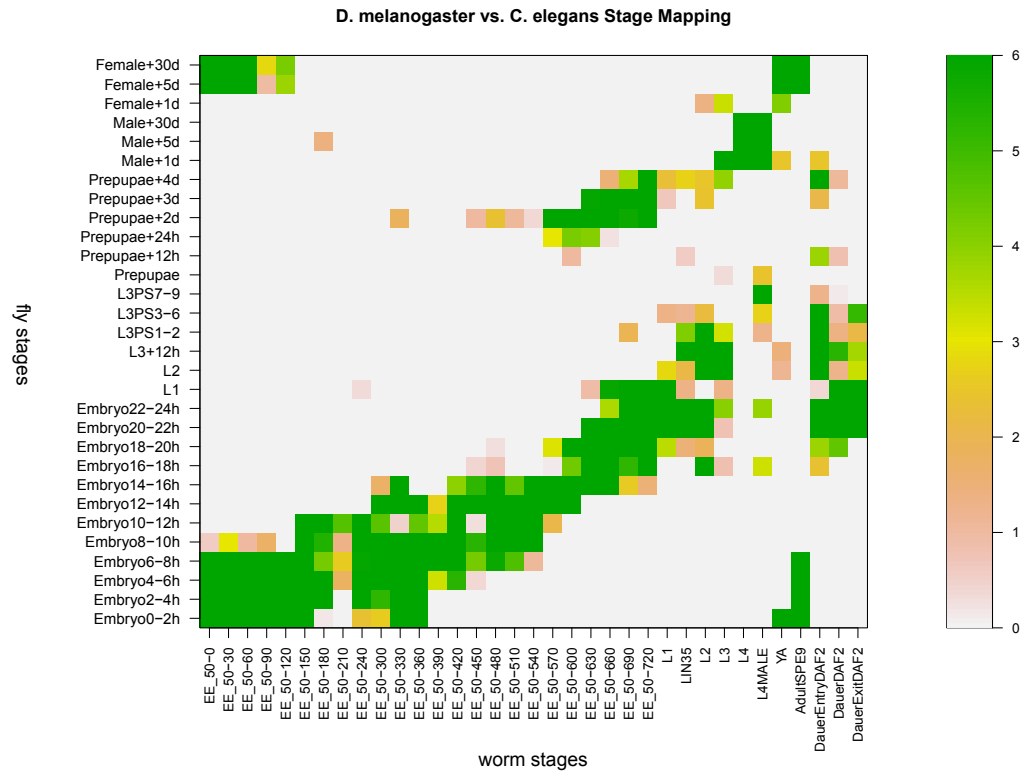

Figure 4: Transcriptome mapping of stages between D. melanogaster and C.elegans

- For each species, we have "within-species overlap genes between sample pairs.xlsx" from `ws.trom()` or "within-species overlap genes (within ortholog genes) between sample pairs.xlsx" from `ws.trom.orthologs()`.
- We have "between-species overlap genes (of species 1) between sample pairs.xlsx" and "between-species overlap genes (of species 2) between sample pairs.xlsx" from `bs.trom()`.

Users can use the function `find.top.GO.terms()` or `find.top.GO.slim.terms()` to find top enriched GO terms or GO slim terms in the above gene lists. They can either directly use the above Excel files or extract columns of interest and save them to a new Excel file.

As an example, we find top enriched GO terms in the developmental stages of *D. melanogaster*. Firstly, we need the GO mapping file (containing GO annotations of genes) of the species and it can be downloaded from `http://geneontology.org/page/download-annotations` and unzipped from the .gz files. We name the mapping file of *D. melanogaster* as "gene_association.fb" and save it to R's working directory. `dm_genes_all` gives the population of *D. melanogaster*'s genes and we find the top enriched GO terms in stage-associated genes (specified by `gene_lists`). Suppose we are interested in the top 20 enriched GO terms. Then we set `topNum = 20` and we can find the enriched GO terms in each biological stages with the following codes. The results are saved to a file named "top 20 enriched GO terms in fly stage-associated genes.txt".

```
dm_genes_all <- as.character(dm_gene_expr[,1])
# generate the associated gene lists: "associated genes.xlsx"
dm_associated_genes <- select.associated.genes(dm_gene_expr,
                                               z_thre=1.5)
dm_stage_GO <- find.top.GO.terms(
 gene_lists = "associated genes.xlsx",
 all_genes = dm_genes_all,
 GOmappingfile = "gene_association.fb",
 output_file = "top 20 enriched terms in fly stage-associated genes.txt",
 topNum = 20,
 heatmap = TRUE)
```

In the example above, we set heatmap = TRUE so that a heatmap for the enrichment analysis of top enriched GO terms will be saved to the working directory as "Top enriched GO terms across samples.pdf". For those GO terms that are at least top enriched (number specified by topNum) in one

sample, the heatmap gives their enrichment results across all the biological samples.

We can also find the top enriched GO slim terms of *D. melanogaster* using the same mapping file "gene_association.fb". `GO_slim_id` is a character vector which stores the IDs of all the GO slim terms. This data can be downloaded from `http://www.geneontology.org/ontology/subsets/goslim_generic.obo`. By setting the arguments in `find.top.GO.slim.terms` in a way similar to the example above, we can find top 20 enriched GO slim terms and save the results to a file named "top 20 enriched GO slim terms in fly stage-associated genes.txt".

```
data(GO_slim_id)
dm_stage_GO_slim <- find.top.GO.slim.terms(
 gene_lists = "associated genes.xlsx",
 all_genes = dm_genes_all,
 GOmappingfile = "gene_association.fb",
 output_file = "top 20 enriched GO slim terms in fly stage-associated genes.txt",
 GO_slim_id = GO_slim_id,
 topNum = 20,
 heatmap = TRUE)
```