MicrobioLink pipeline

Codes developed by Balász Bohár, Padhmanand Sudhakar and Tahila Andrighetti.

What do you need for the pipeline

- docker (version = 3.4.0) or
- python (version >= 3.6)

Installing and running the pipeline

Using docker:

1) Intall docker and execute the docker post-installation.
2) Configure your inputs.
3) Execute the script `sh hmipipleine.sh`, this script will build the docker image and start a container, after the script execution you will be in an execution-ready environment.
4) If the installation is successfull, you should get something like this: `root@248342fef37e:/home/hmipipeline#`
5) Execute `sh USER_complete_mode.sh` or `sh USER_stepbysetp_mode.sh`, your outputs will be saved into the `output` folder.

Using python directly:

1) In order to separate your OS python from this project's python, let's create a virtual environment:

   ```
   python3 -m venv .venv
   ```

2) Lets activate our virtual environment:

   ```
   source .venv/bin/activate
   ```

   If it was successfully executed you are going to see the mark `(.venv)` in your terminal.

3) Now we have to install the dependencies:

   ```
   pip install -r requirements.txt
   ```

4) Configure your inputs.

5) Execute `sh USER_complete_mode.sh` or `sh USER_stepbysetp_mode.sh`, your outputs will be saved into the `output` folder.

6) Remember, to exit your virtual environment, type: `deactivate`.

You can choose to run the complete pipeline, passing through all the steps, or running the steps separatedly.

COMPLETE MODE

This section will describe the execution of the default running of all steps automatized. You only need to provide input files containing bacterial proteins, receptors and target genes and, after the run, you will have a complete host-microbiome network.

Find complete information about the steps in the section **Step-by-step mode**.

For running the complete mode, you will open the file: *complete*running.sh_ and complete the variables with your preferences and input files names.

After complete it, run it:

```
sh complete_mode.sh
```

More about needed files and parameters for complete running:

> ***INPUT FILES***:
>
> All your input files must be in the folder "user_input". Complete the variables in *complete*running.sh_ with the name of your inputs. The original file contains the name of the example inputs. Replace the example names with the names of your data. You can also look at the example data how the input files must be.
>
> You will need the following inputs:
>
> – **bacterial_proteins**: This file correspond to bacterial proteins part of your sample. It must include the protein ID it correspondent PFAM. Include the variable *bacterial*column_ID_ and *bacterial*column_PFAM_ to indicate in which column each one is located.
>
> – **human_proteins_DDI**: If you choose to perform the domain-domain interaction prediction, you will have to input this file. This file correspond to the receptor host proteins. It must include the protein ID and its PFAM correspondence. Include the variable *host*column_ID_ and *host*column_PFAM_ to indicate in which column each one is located.

– **human__proteins__DMI**: If you choose to perform the domain-motif interaction prediction, you will have to input this file. This file correspond to the sequence of receptor host proteins. It has the gene ID as a fasta header and the aminoacid sequence above.

– **target__genes__file**: List of your target genes to TieDie execution. It must have the gene name in the 1th column, gene weight in the 2nd column and if it its status is + or - (up or downregulated, for example). The gene weight in the example file is the differential expression between Crohn's disease and healthy condition. You can choose a weight according to your study.

PARAMETERS:

– **DDInDMI**: *DDI* or *DMI* or *ALL* By this parameter you will indicate if you want to run only domain-domain interaction prediction (*DDI*), only domain-motif interaction prediction (*DMI*), or both (*ALL*).

– **id__format**: *genename* or *uniprot* By this parameter you will indicate if the gene names in your data are in format of gene name (gene symbol, use *genename*), or in Uniprot Acession format (use *uniprot*).

OUTPUT FILES:

All the generated output files will be in the folder "output".

Analysis output : *tiedie.sif*: this file contains information about the interactions contained in the generated network. You can open it in Cytoscape to vizualize and to do the filtering steps.

STEP-BY-STEP MODE

If you want to run the steps separatedly, you will open the main file `stepbystep_mode.sh` and complete the variables with the information about files and parameters acording to the step that you will run.

Then, you will run the script:

```
sh stepbystep_mode.sh
```

INPUT INFORMATION

All your input files must be in the folder "user_input". Inside the file `stepbystep_mode.sh` you will find various variables to complete in order to run the script.

- `step=` – complete according to the step that you want to perform. – Options: "DDI", "DMI", "structural_filtering", "tiedie_input", "tiedie" – find above the description for each option
- `id_format=` – your gene or protein identification have to be in Uniprot ID or Gene Symbol/Name format. Complete according with your data. – Options: "genename" or "uniprot"
- `Input variables` – You will complete this variables according to the step you are performing. The options and descriptions are detailed in the steps above. – Variables: input_file1, col_id1, col_pf1, input_file2, col_id2 and col_pf2

The information about each step is found above:

1 Domain-domain interactions (DDI)

Variable in the main file: `step="DDI"`

INPUT FILES :

– **input_file1**: same as *bacterial*proteins_. This file correspond to bacterial proteins part of your sample. It must include the protein ID it correspondent PFAM. Include the variable *col*id1_ and *col*pf1_ to indicate in which column each one is located.

– **input_file2**: same as *human*proteins_DDI_. If you choose to perform the domain-domain interaction prediction, you will have to input this file. This file correspond to the receptor host proteins. It must include the protein ID and its PFAM correspondence.

Include the variable *col*id2_ and *col*pf2_ to indicate in which column each one is located.

input_file1= bacterial proteins file

col_id1= number of bacterial protein ID column col_pf1= number of bacterial protein PFAM column

input_file2= host protein file

col_id2= number of host protein ID column

col_pf2= number of host protein PFAM column

USED DATABASE

- **pfam_interactions.txt**: this file contains the complete database to use as reference to the predictions by default. It was obtained in PFAM database. If you want to change it, find it at */deploy/pipeline/1.DDI*

OUTPUT: *DDIpreds.txt* and in *DDIpreds*with_info.txt_.

2 Domain-motif interactions (DMI)

Variable in the main file: `step="DMI"`

INPUT FILES :

- **input_file1**: same as *bacterial*proteins_. This file correspond to bacterial proteins part of your sample. It must include the protein ID it correspondent PFAM. Include the variable *col*id1_ and *col*pf1_ to indicate in which column each one is located.

- **input_file2**: If you choose to perform the domain-motif interaction prediction, you will have to input this file. This file correspond to the sequence of receptor host proteins. It has the gene ID as a fasta header and the aminoacid sequence above. The variables *col*id2_ and *col*pf2_ are not needed. Keep it as 0.

  input_file1= bacterial proteins file

  col_id1= number of bacterial protein ID column

  col_pf1= number of bacterial protein PFAM column

  input_file2= host protein sequences fasta file col_id2=0

col_pf2=0

USED DATABASE **- elm__motif.tsv** and **elm__interaction__domains.tsv**: this file contains the complete database to use as reference to the predictions. It was obtained in ELM database. If you want to change it, find it at */deploy/pipeline/2.DMI*

OUTPUT: *MPDMIresult.tsv*

3 Structural filtering

Variable in the main file:
`step="structural_filtering"`

In this step, you will perform the *structural filtering of the interactions obtained in the former step.* Structural filtering is applied only to DMI, not to DDI.

INPUT FILES :

- **input__file1**: This file contains the final domain-motif interactions between the bacterial and host receptor proteins. See the file *MPDMIresult.tsv* to more information about how the input must be.

The variables *input*file2__ and col_* are not needed and you can leave it empty (“”).

input_file1= DMI file

col_id1=0

col_pf1=0

input_file2=" "

col_id2=0

col_pf2=0

OUTPUT : *DMI*filtered.cvs_: This file contains the filtered domain-motif interactions between the bacterial and host receptor proteins. Contains columns with bacterial ID, domains and motifs information.

4 TIEDIE

In this step, you will generate the intermediary network between the receptor proteins (upstream proteins, the proteins that interacts with the bacterial proteins) and the target genes (downstream). TieDie tool uses weights in the genes in order to retain the most relevant pathways.

4.1 TieDie INPUT GENERATOR :

Variable in the main file:
`step="tiedie_input"`

By default, MicrobioLink uses the number of bacterial proteins that each host receptor protein is connected to. Then, this step is used to count these interactions for each receptor, generating an input that can be used in TieDie.

- **input_file1**: This file contains interactions between bacterial and host proteins, with its id. Indicate in the variables *col*id1_ the number of column with bacterial proteins and *col*id2_ the number of column with host proteins, as showed above:

  input_file1= file

  col_id1= number of bacterial protein ID column

  col_pf1=0

  input_file2=" "

  col_id2= number of host protein ID column

  col_pf2=0

OUTPUT FILE: contains the information of the **upstream** proteins, the host proteins connected with bacterial proteins. It is the default input file for TieDie. This file has 3 columns:1) protein ID (it has tp be in Gene Symbol or Uniprot ID); 2) weight of the protein (with how much bacterial protein it is connected, by default); 3) if the protein is positive (+) or negative (-) according to the study context (they are all + for MicrobioLink default).

4.2. TieDie EXECUTION : > Variable in the main file:
> `step="tiedie_input"` > > ***INPUT FILES*** :

– **input_file1**: contains the information of the **upstream** proteins, the host proteins connected with bacterial proteins. This file must have 3 columns: 1) protein ID (it has tp be in Gene Symbol or Uniprot ID); 2) weight of the protein (with how much bacterial protein it is connected, by default); 3) if the protein is positive (+) or negative (-) according to the study context (they are all + for MicrobioLink default).

– **input_file2**:contains the information of the **downstream** proteins, the target genes. This file must have 3 columns: 1) protein ID (it has tp be in Gene Symbol or Uniprot ID); 2) weight of the protein (here, it can be choosen by the user according with the study); 3) if the protein is positive (+) or negative (-) according to the study context (upregulated or downregulated, for example).

The variables *col* . . . are not needed. Keep it as 0.

input_file1= upstream file

col_id1= 0

col_pf1=0

input_file2= downstream file

col_id2= 0

col_pf2=0

Obs.: it is normal to return the following message: "*Warning: input heat node Q5MNZ9 not in the network and will be ignored. . .*"

OUTPUT : *tiedie.sif*: this file contains information about the interactions contained in the generated network. You can open it in Cytoscape to vizualize and to do the filtering steps.

PERFORMING AN EXAMPLE

In this section, you will se how to do a testing simmulation with the provided example files. It is a reduced simmulation, only to perform MicrobioLink in a shorter time and to show how the input and output files must be. It doesn't have biological meaning.

The example files are found in the folder *example*files_. For performing the example, paste the file from the example folder into the folder *user*inputs_ according with the simulation you want to perform. Follow the steps above:

Complete simulation example

INPUT FILES: Paste all the input files mentioned above in the folder "user_input", or run the following script:

```
cp example_files/Metaproteome.txt example_files/humanDDI.txt
example_files/humanDMI.fasta example_files/Tie_Die_downstream.input
user_inputs/
```

In the file `USER_complete_mode.sh` complete the variables as following:

bacterial_proteins="Metaproteome.txt"

bacterial_column_id=1

bacterial_column_pf=2

human_proteins_DDI="humanDDI.txt"

human_column_id=1

human_column_pf=2

human_proteins_DMI="humanDMI.fasta"

target_genes_file="Tie_Die_downstream.input"

DDInDMI="ALL"

id_format="uniprot"

save

run `sh USER_complete_mode.sh`

OUTPUT:

You must see in the terminal:

—-STARTING DDI PREDICTION

```
— DDI FINISHED!

— # PREDICTIONS: 316

– Output file: DDIpreds.txt and DDIpreds_with_info.txt

—-STARTING DMI PREDICTION

— DMI FINISHED!

– PREDICTIONS: 1116

– Output file: MPDMIresult.tsv

—-STARTING STRUCTURAL FILTERING

528

1116

O14763

(. . . a list of proteins)

— STRUCTURAL FILTERING FINISHED!

Output file: DMI_filtered.csv
— GENERATING TieDie INPUT FROM PREDICTED INTERACTIONS
— TieDie INPUT FINISHED!
— TieDie starting. . .
Warning: No kernel file supplied, will use SCIPY to compute the matrix expo-
nential, t=0.1. . .
Parsing Network File..
Warning: input heat node O60883 not in the network and will be ignored. . .
Warning: input heat node P47775 not in the network and will be ignored. . .
Warning: input heat node O43424 not in the network and will be ignored. . .
(. . . more warning)
```

STEP BY STEP SIMULATION EXAMPLE

INPUT FILES:

Paste the input files in the folder “user_input” according to the step you want to perform.

In the file `USER_stepbystep_mode.sh` complete the variables as following: `id_format= "uniprot"`

1 Domain-domain interactions (DDI)

Variable in the main file: `step="DDI"`

input_file1=“Metaproteome.txt”

col_id1=1

col_pf1=2

input_file2=“humanDDI.txt”

col_id2=1

col_pf2=2

2 Domain-motif interactions (DMI)

Variable in the main file: `step="DMI"`

input_file1=“Metaproteome.txt”

col_id1=1

col_pf1=2

input_file2=“humanDMI.fasta”

col_id2=0

col_pf2=0

3 Structural filtering

Variable in the main file:
`step="structural_filtering"`

input_file1="MPDMIresult.tsv"

col_id1=0

col_pf1=0

input_file2=" "

col_id2=0

col_pf2=0

4 TIEDIE

4.1 TieDie INPUT GENERATOR :

Variable in the main file:
`step="tiedie_input"`

input_file1="DDIpreds.txt"

col_id1=1

col_pf1=0

input_file2=""

col_id2=2

col_pf2=0

4.2. TieDie EXECUTION :

Variable in the main file:
`step="tiedie"`

input_file1="Tie_Die_upstream.input"

col_id1=0

col_pf1=0

input_file2="Tie_Die_downstream.input"

col_id2=0

col_pf2=0

After complete the variables:

save
run `sh USER_stepbystep_mode.sh`