The AuReMe Workspace

Table des matières

[How to use Docker 3](#_Toc527495981)

[Running a Docker container 3](#_Toc527495982)

[Some tips about Docker 3](#_Toc527495983)

[How to use the AuReMe workspace (default workflow) 4](#_Toc527495984)

[Requirements 4](#_Toc527495985)

[Start a new study 4](#_Toc527495986)

[Define the reference database 4](#_Toc527495987)

[The default workflow 5](#_Toc527495988)

[Data organization 5](#_Toc527495989)

[Bridge structure 5](#_Toc527495990)

[Provide input files 6](#_Toc527495991)

[Check input files validity 7](#_Toc527495992)

[Orthology-based reconstruction 7](#_Toc527495993)

[Method: Pantograph 7](#_Toc527495994)

[Annotation-based reconstruction 8](#_Toc527495995)

[Method: Pathway Tools 8](#_Toc527495996)

[Merge metabolic networks 9](#_Toc527495997)

[Gap-filling 9](#_Toc527495998)

[Method: Meneco 9](#_Toc527495999)

[Manual curation 10](#_Toc527496000)

[Add a reaction from the database or delete a reaction in a network 10](#_Toc527496001)

[Create new reaction(s) to add in a network 11](#_Toc527496002)

[Apply changes 11](#_Toc527496003)

[FAQ 12](#_Toc527496004)

[Can I have a sample of AuReMe? 12](#_Toc527496005)

[How to convert files to different formats? 12](#_Toc527496006)

[How to manage growth medium? 12](#_Toc527496007)

[How to manage metabolic network compartment? 12](#_Toc527496008)

[How to manage the log files? 13](#_Toc527496009)

[How to reproduce studies? 13](#_Toc527496010)

[How to create a new ‘à-la-carte’ workflow? 13](#_Toc527496011)

[How to choose another reference database? 14](#_Toc527496012)

[What is checked in my input files? 14](#_Toc527496013)

[How to regenerate a new database version? 14](#_Toc527496014)

[How to map a metabolic network on another database? 14](#_Toc527496015)

[How to generate reports on results? 15](#_Toc527496016)

[How to generate Wiki? 15](#_Toc527496017)

[How to connect to Pathway-tools? 16](#_Toc527496018)

[The Docker container 16](#_Toc527496019)

[What are “artefacts”? 17](#_Toc527496020)

[How to process Flux Balance Analysis? 17](#_Toc527496021)

[How to set an objective reaction? 17](#_Toc527496022)

# How to use Docker

Requirements: Docker (v 1.10 at least)

To install Docker, please follow the instructions on [docker.com](https://www.docker.com/), considering your operating system\*.

\*On Mac OS: requires at least Yosemite 10.10.4

\*On Windows: requires at least Windows 8

## Running a Docker container

1. Launch the Docker machine (see the instruction on [docker.com](https://www.docker.com/)). For example:

* On Fedora: **sudo systemctl start docker**
* On Mac OS and Windows: run the Docker launcher

1. Download the AuReMe Docker image

**$> docker pull dyliss/aureme-img**

1. To verify that the image has been downloaded correctly, check it in the list of your local images:

**$> docker images -a**

REPOSITORY TAG IMAGE ID CREATED SIZE

**dyliss/aureme-img latest 6cf38ab4edc8 1 hour ago 1.68 GB**

1. Create a folder that will serve as a bridge to share data from/to the Docker container. Let us call it ***bridge*** for instance.
2. Create a Docker container from the following image with this command:

**$> docker run -ti -v** */my/path/to/the/shared/directory/bridge***:/shared --name="***aureme-cont***" dyliss/aureme-img****bash**

The path given for –v is the one to the shared directory on your host machine

**This path has to end on the directory name** (without any **/** at the end)

**The path has to be complete** (from **/users** or from **C:\\** for Windows users)

After the ‘:’ is the name of the mirror directory in the Docker container. Please do not change it.

For Windows users, be careful, you have to indicate your path this way:

**$> docker run -ti –v** *C:\my\path\to\the\directory\bridge***:/shared --name="***aureme-cont***" dyliss/aureme-img****bash**

You can create as many container as you wish, as long as you give them different names.

Your AuReMe container is now running and correctly installed. Congratulations! You are now inside the container ***aureme-cont***.

## Some tips **about** Docker

* To exit the container, tape **exit.**

**$> exit**

* To list all your containers:

**docker ps –a**

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

**fff2e9eca536 aureme-img "bash" 2 hours ago Up 4 days aureme-cont**

Remark that you can see, with this command, the state of your containers in the STATUS column: **up** (running, you can connect to it), **exited** (stopped, need to be started again)

* To start or stop the container (from your host):

**$> docker start** *aureme-cont*

**$> docker stop** *aureme-cont*

* If you want to go inside a started/running container:

**$> docker exec -ti** *aureme-cont* **bash**

* To delete a container: **docker rm *container\_id* (**or ***name*)**
* To delete an image: **docker rmi *image\_id* (**or ***name*)**

This is impossible if any existing container has been created from it. Delete all dependent containers first.

# How to use the AuReMe workspace (default workflow)

## Requirements

* 1. Create your Docker container as explained in the previous step **“Running a Docker container”**, start the container and go inside.

### Start a new study

* 1. Use the following command to start a new study. Choose an identifier for this study (ex: replace ***test*** by your organism name). In order to illustrate this documentation, we will use ***test*** as a run identifier.

**$> aureme –-init=***test*

Now you will find on your own computer (host), in your ***bridge*** directory, a folder ***test*** with many subdirectory and files. This is your work directory, on which AuReMe is going to run.

Notice that from now until the end of the process, every command will be stored as a log in the ***bridge*⏵*test*⏵*log.txt*** file. The whole output of these commands will also be stored in the ***bridge*⏵*test*⏵*full\_log.txt*** file.

If you wish NOT to store such logs, you can use the **quiet** argument in your command(s). This will redirect the output on the terminal

For example:

**$> aureme --run=***test* **–-cmd=″***some\_command***″ -q**

For further details on the log files, please see the **“FAQ ⏵ How to manage the log files”** chapter.

* 1. To get an overview of AuReMe, you can get a sample by using this command.

**$> aureme --sample**

### Define the reference database

* 1. The final step is to define which reference database to use. The available databases are listed in your terminal when you create a new study. If needed, use this command to display them again.

**Available database in Aureme:**

**/home/data/database/BIGG/bigg**

**/home/data/database/BIOCYC/METACYC/20.5/metacyc\_20.5\_enhanced**

**/home/data/database/BIOCYC/METACYC/22.0/metacyc\_22.0\_enhanced**

**$> aureme --run=***test* **--cmd=″getdb″**

This reference database is needed to:

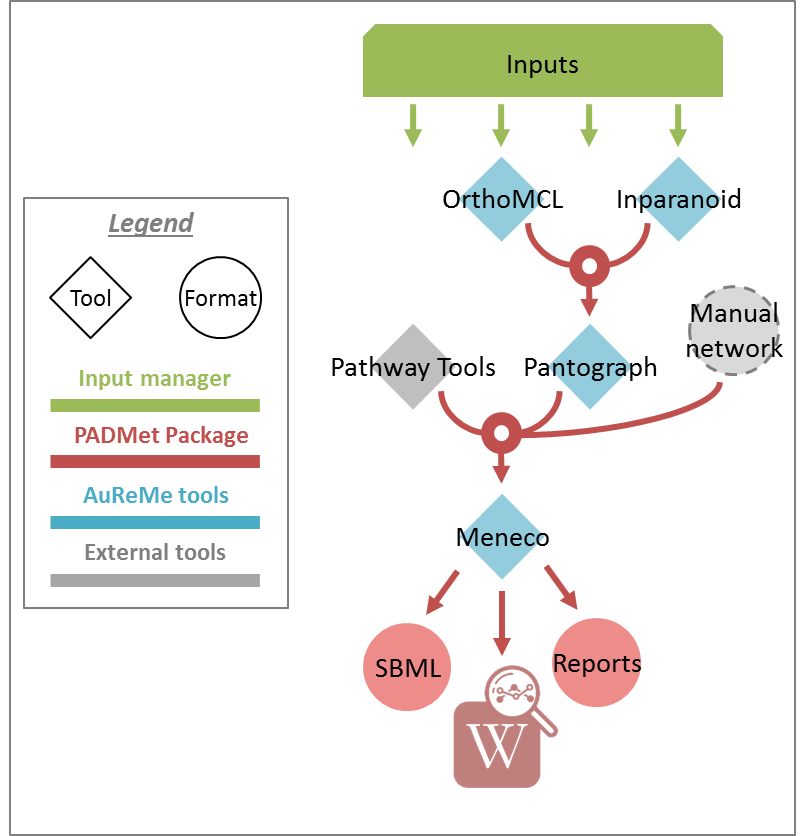
* Be able to match all the identifiers of the entities of metabolic networks
* Gap-fill the metabolic network in the gap-filling step
* Uniforms the data in one unique database

To select one, replace the corresponding path in the configuration file: ***config.txt***, in the ***DATA\_BASE***variable. Or comment the line if you don’t want/can’t use a database.

The ***config.txt*** file is stored at the root of your ***test*** folder.

## The default workflow

By default, the AuReMe workspace includes an automatic workflow for metabolic network reconstruction. This workflow runs several pre-installed tools and generates diverse output files. The process can be either run entirely in a single command, or run step by step to personalize it or do some intermediary analysis.

For instance, if you run the ***draft*** command (see **“Merge metabolic networks”**), it will run all the previous steps automatically as described in the following figure. This figure details the steps of the default workflow.

## Data organization

### Bridge structure

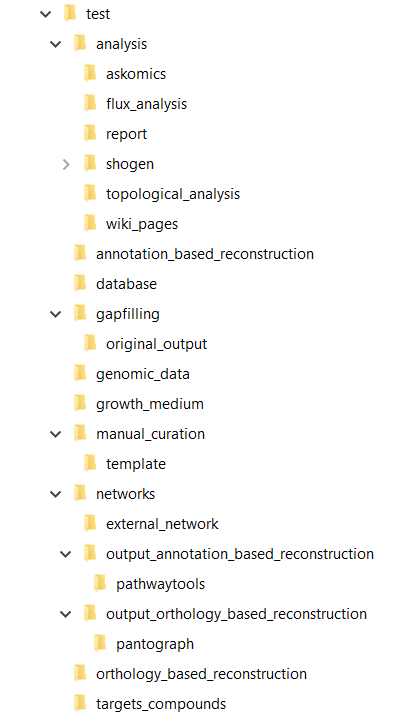
The ***bridge*** directory will store all your input data you will provide, and all the result files the workflow is going to create.

***analysis***: all output files of the analysis processes

***annotation\_based\_reconstruction***: if you want to use annotated genomes (to run the annotation-based reconstruction part of the workflow), put here all the output files of the annotation tool. For instance, with Pathway Tools, copy-paste the whole PGDB directory (see below “Annotation-based reconstruction” for more details).

***database***: if you want to use your own database put in this folder your database in padmet format, if you have a sbml convert this file to padmet (see **“FAQ ⏵ How to convert files to different formats”**)

***gapfilling/original\_output***: if you run the metabolic network reconstruction with gap-filling, will contain all the output files of gap-filling tools before any post-process from AuReMe.

***genomic-data***: the directory in which to put the genomic data on your studied organism, that is to say either a Genbank (.gbk) or a proteome (.faa).

***growth\_medium***: description of the set of metabolites that is available to initiate the metabolism (growth medium), that is to say the seed compounds (.txt) (see **“FAQ ⏵ How to manage growth medium?”**).

***manual\_curation***: all the file to describe the manual curation you want to apply on your metabolic network (either adding, deleting or modifying reactions).

***networks***: all the metabolic networks used or created during the reconstruction process

***networks* ⏵ *external\_network***: put here all existing metabolic networks (.sbml) you want to use. Enables to merge them with the ones created thanks to other methods.

***networks* ⏵ *output\_annotation\_based\_reconstruction***: will contain the processed network from the annotation based reconstruction, after the pre-processing of the data from the ***annotation\_based\_reconstruction*** directory (if you filled this one).

***networks* ⏵ *output\_orthology\_based\_reconstruction***: will contain the processed network from the orthology based reconstruction, after the pre-processing of the data from the ***orthology\_based\_reconstruction*** directory (if you have run this part of the workflow).

***orthology\_based\_reconstruction***: if you want to use model organisms (to run orthology-based reconstruction part of the workflow), put here the proteome (.faa or .gbk) and the metabolic network (.sbml) of your model (see below “Orthology-based reconstruction” for more details).

***targets\_compounds***: description of the set target compounds (.txt), that is to say metabolites whose production is supposed to be achieved by the metabolism of the species under study (components of the biomass reactions or other metabolites).

### Provide input files

First of all, you have to provide to AuReMe all the input files needed for the different steps you want to run in the workflow. The steps can be optional or run several times, at different phases of the process. However, you have to store the input data for each steps, observing the architecture described above for the ***bridge*** directory (see **“Data organization ⏵ Bridge structure”**).

Here is the list of input you have to provide to run the pre-set default workflow. If you want to run only part of it, please see the corresponding sections and the chapter **“How to create your ‘à-la-carte’ workflow”**.

* **See “Orthology-based reconstruction ⏵ Inputs”**
* **See “Annotation-based reconstruction ⏵ Inputs”**
* **External source for reconstruction**

If you already have one or several external metabolic networks for your studied species and you want to improve them, just copy-paste them (SBML format) in the ***networks*⏵*external\_network*** folder.

/test

|--networks

|-- external\_network

|-- *first\_manual\_network.sbml*

|-- *second\_manual\_network.sbml*

|-- ...

### Check input files validity

* 1. IMPORTANT: Always check the validity of the inputs before running any workflow task, and after having put every input files needed for the steps of the workflow. This will verify the format and consistency of your data for a better quality result. Moreover, it will generate all the supplementary files needed by the workflow tools and put them into the corresponding folders. For more information about input files validity see **“FAQ ⏵ What is checked in my input files?”**.

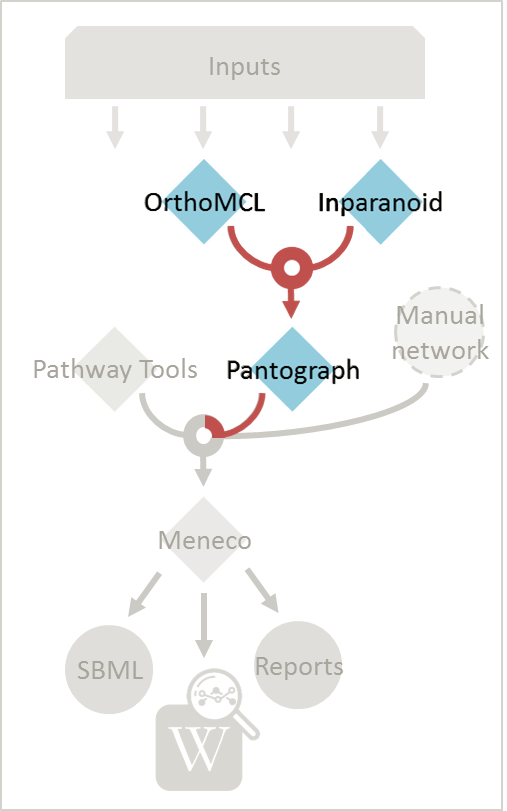
For this purpose, use this command:

**$> aureme --run=***test* **--cmd=″check\_input″**

## Orthology-based reconstruction

### Method: Pantograph

Input files:

- Required for the orthology-based reconstruction (method: Pantograph):

- Genbank or Proteome of your studied organism (.gbk or .faa)

- Genbank or Proteome of your reference organism (.gbk or .faa)

- Metabolic network of your reference organism (.sbml)

- (option) a dictionary file if genes ids used in metabolic network are different with gbk/faa (.txt)

Result file:

/test

|--orthology\_based\_reconstruction

| |-- *model\_a*

| |-- **original\_output\_pantograph\_*model\_a*.sbml**

|-- networks

|-- orthology\_based\_reconstruction

|-- pantograph

|-- **output\_pantograph\_***model\_a***.sbml**

#### Inputs

* 1. Put all the available genomic data of the studied organism in the folder ***genomic\_data***, either a Genbank (.gbk) or a Fasta (.faa) file. WARNING: give them these exact names (respectively): GBK\_study.gbk and FAA\_study.faa.
  2. For each reference organism you want to use, create a folder in the folder ***orthology\_based\_reconstruction***. Give it the name of your model organism (e.g. ***model\_a***).

On a Linux operating system, here is the command to create a new folder named ***model\_a***:

**$> mkdir orthology\_based\_reconstruction/***model\_a*

* 1. In each folder, put:
     + the Genbank file of your model organism, with the exact name GBK\_model.gbk

OR the proteome of your model organism, with the exact name FAA\_model.faa

* + - the metabolic network of your model organism, with the exact name metabolic\_model.sbml

/test

|--orthology\_based\_reconstruction

|-- *model\_a* (you can change the name of the folder)

|-- GBK\_model.gbk or FAA\_model.faa

|-- metabolic\_model.sbml

|-- *dict\_genes.txt (option)*

* 1. The genome (or proteome) and the metabolic network of your model organism have to contain the same kind genes (or proteins) identifiers to be comparable. If not enough genes (or proteins) are in common between the two files, the process will stop to avoid poor quality data production.  
     If you want to pursue on the process, please provide a dictionary file between the gene (or protein) identifiers of these two files. Name this dictionary ***dict\_genes.txt***. Here is the dictionary file format asked (tabulation separated values):

**gene\_id\_from\_sbml1 gene\_id\_from\_faaA**

**gene\_id\_from\_sbml2 gene\_id\_from\_faaB**

**gene\_id\_from\_sbml3 gene\_id\_from\_faaC**

#### Run

* 1. Important: Remember to check the validity of the inputs before running any workflow task. If you want to run only the orthology-based reconstruction, use now this command:

**$> aureme --run=***test* **--cmd=″check\_input″**

* 1. To run only the orthology-based reconstruction, use this command:

**$> aureme --run=***test* **--cmd=″orthology\_based″**

* 1. IMPORTANT: Because the metabolic network from the reference organism could came from different databases, it’s critical to check the database of each network and if needed convert the network to your reference database selected (see **“How to use the AuReMe workspace (default workflow)⏵ Define the reference database”**).

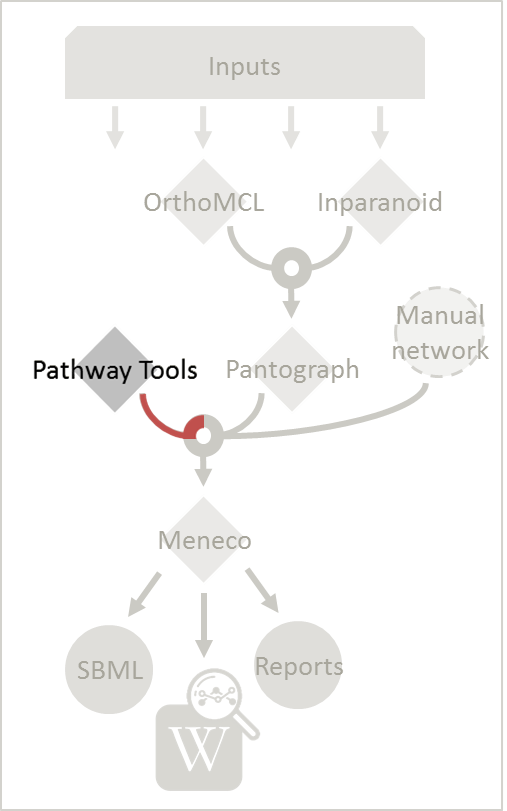
**$> aureme --run=***test* **--cmd=″which\_db SBML=** **output\_pantograph\_model\_a.sbml″**

The previous command will check the database of the file output\_pantograph\_mode\_a.sbml, if the database is different for the reference, use the next command to create a mapping file to metacyc database. For more information about sbml mapping see **“FAQ ⏵ How to map a sbml to another database?”**.

**$> aureme --run=***test* **--cmd=″sbml\_mapping SBML=** **output\_pantograph\_model\_a.sbml DB=METACYC″**

## Annotation-based reconstruction

### Method: Pathway Tools



Input files:

- Required for the annotation-based reconstruction (method: Pathway Tools):

The output of Pathway tools (PGDB folder)

Result file:

/test

|-- networks

|-- annotation\_based\_reconstruction

|-- pathwaytools

|-- **output\_pathwaytools\_***genome\_a***.padmet**

|-- **output\_pathwaytools\_***genome\_b***.padmet**

#### Inputs

* 1. Put the output of Pathway Tools (the whole PGDB directory) in the folder ***annotation\_based\_reconstruction***

/test

|--annotation\_based\_reconstruction

|-- *genome\_a* (you can change the name of the folder)

|-- proteins.dat

|-- reactions.dat

|-- genes.dat

|-- enzrxns.dat

|-- pathways.dat

|-- compounds.dat

* 1. If you have run several times Pathway Tools and want to use all of these annotations, just copy-paste the other PGDB folders in the ***annotation\_based\_reconstruction*** directory.

#### Run

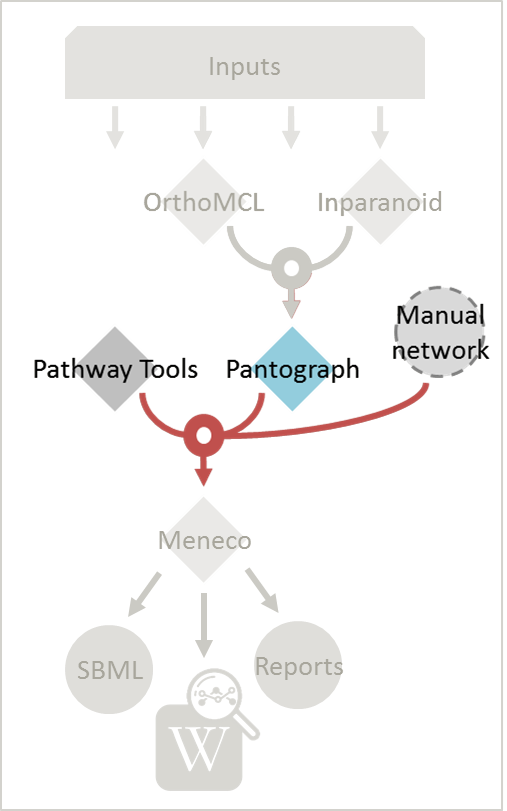
* 1. Important: Remember to check the validity of the inputs before running any workflow task. If you want to run only the annotation-based reconstruction, use now this command:

**$> aureme --run=***test* **--cmd=″check\_input″**

* 1. To run only the annotation-based reconstruction, use this command.

**$> aureme --run=***test* **--cmd=″annotation\_based″**

## Merge metabolic networks



Input files:

- metabolic networks in the ***networks*** directory

Result files:

/test

|-- netowrks

|-- **draft.padmet**

To merge all available networks from the ***networks*** directory into one metabolic network, merging all data on the studied species, run this command:

**$> aureme --run=***test* **--cmd=″draft″**

Note that you can also add external metabolic network to create the draft (see **“Data organization ⏵ Provide input files ⏵ External source for reconstruction”**).

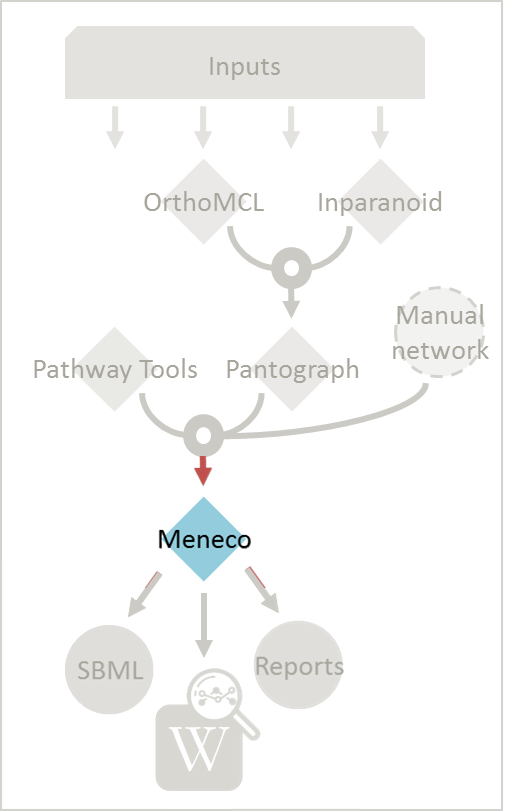
IMPORTANT: Before merging your networks, check if not already done if all the sbml are using the reference database. Also check the compartment ids used in each of them, delete and change compartment if need.

For example: if a sbml is using KEGG database but your reference database is metacyc, you will have to map this sbml to create a mapping file which will be used automatically in the merging process.

If a sbml contains a compartment id like ‘C\_c’ and another contains ‘c’, although they correspond to the same compartment ‘cytosol’ because of different ids, a compound in ‘C\_c’ is not the same as a compound in ‘c’, therefore there will be a loss of connectivity in the network. see **“FAQ ⏵ How to map a sbml to another database?” and “FAQ ⏵ How to manage compartment?”**

## Gap-filling

### Method: Meneco

Input files:

- Required for the gap-filling (method: Meneco):

- A metabolic network reference database (.padmet or .sbml)

(metacyc 20.5, 22.0, BIGG and ModelSeed are available by default)

- Seed and target metabolites (.txt)

- A metabolic network to fill (typically created during the previous steps)

Result files:

/test

|-- netowrks

|-- *network\_name***.sbml**

|-- *network\_name***.padmet**

|-- gapfilling

|-- original\_output

| |-- **meneco\_output\_** *network\_name***.txt**

|-- **gapfilling\_solution\_** *network\_name***.csv**

#### Input

* 1. You must have selected a reference database to fill-in the potential gaps in the metabolic network. If it is not done yet, please see **“Requirements ⏵ Define the reference database ”**
  2. Put the seeds file (named seeds.txt) in the ***growth\_medium*** folder. The seed compounds are the description of the set of metabolites that is available to initiate the metabolism (growth medium).

**Seed\_compound\_id1**

**Seed\_compound\_id2**

**Seed\_compound\_id3**

Here is as example of the seed file format:

* 1. Set the growth medium using this command:

**$> aureme --run=***test* **--cmd=″set\_medium NETWORK=***network\_name* **NEW\_NETWORK=***new\_network\_name***″**

For more details on the medium settings, see **“FAQ ⏵ How to manage growth medium?”**.

WARNING: If you don’t precise any **NEW\_NETWORK** name, the current network will be overwritten.

* 1. Put the target file (named targets.txt) in the ***targets\_compounds*** folder. The targets are metabolites whose production is supposed to be achieved by the metabolism of the species under study (components of the biomass reactions or other metabolites).

Here is as example of the seed file format:

**target\_compound\_id1**

**target\_Compound\_id2**

**target\_Compound\_id3**

* 1. You will have to indicate which metabolic network you want to gap-fill with the Meneco software. If you want to gap-fill a network created in the previous steps, there is nothing to do. Otherwise, put the network you want to gap-fill (PADMET format) in the ***networks*** directory.

/test

|-- networks

| |-- *network\_name*.padmet

|-- growth\_medium

| |-- seeds.txt

|-- targets\_compunds

|-- targets.txt

#### Run

* 1. (optional step) To generate the gap-filling solution run this command:

**$> aureme --run=***test* **--cmd=″gap\_filling\_solution NETWORK=***network\_name***″**

Note: Do not forget the quotation marks.

It will calculate the gap-filling solution on the *network\_name* network (in the ***networks*** directory) and put it into the ***gapfilling*** directory as gapfilling\_solution\_network\_name.csv

* 1. To generate the gap-filled network (and run step 6), run this command:

**$> aureme --run=***test* **--cmd=″gap\_filling NETWORK=***network\_name* **NEW\_NETWORK=***new\_network\_name***″**

Note: Do not forget the quotation marks.

It will calculate the gap-filling solution (if it is not yet done) on the *network\_name* network (in the ***networks*** directory) and put it into the ***gapfilling*** directory. Then it will generate the metabolic network (*new\_network\_name*), completed with the gap-filling solution, in the ***networks*** directory.

Note that you can first generate the solution, modify it, then generate the gap-filled network.

WARNING: If you don’t precise any **NEW\_NETWORK** name, the current network will be overwritten.

## **Manual curation**

This step can be done several times and at any moment of the workflow.

* 1. Describe the manual curation(s) you want to apply by filling the corresponding form(s) as explained below.

Important note: It is highly recommanded to create a new form file (.csv) each time you want to apply other changes, in order to keep tracks of them.

### Add a reaction from the database or delete a reaction in a network

* + 1. Copy from the folder **manual\_curation⏵data⏵template**  the file **reaction\_to\_add\_delete.csv** and paste it into the **manual\_curation⏵data** directory (this way on Linux operating systems):

**$> cp manual\_curation/data/template/reaction\_to\_add\_delete.csv manual\_curation/data/***my\_change\_form***.csv**

* + 1. Fill this file (follow the exemple in the template).

**idRef Comment Action**

**my\_rxn Reaction deleted for of x reason delete**

**+-BORNEOL-DEHYDROGENASE-RXN Reaction added for of x reason add**

### Create new reaction(s) to add in a network

* + 1. Copy from the folder **manual\_curation⏵data⏵template** the file ***reaction\_creator.csv*** and paste it into the **manual\_curation⏵data** directory (this way on Linux operating systems):

**$> cp manual\_curation/data/template/reaction\_creator.csv manual\_curation/data/***my\_create\_form***.csv**

* + 1. Fill this file (follow the exemple in the template).

**reaction\_id my\_rxn**

**comment reaction added for X reason**

**reversible false**

**linked\_gene (gene\_a or gene\_b) and gene\_c**

**#reactant/product #stoichio:compound\_id:compart**

**reactant 1.0:compound\_a:c**

**reactant 2.0:compound\_b:c**

**product 1.0:compound\_c:c**

**reaction\_id my\_rxn\_2**

**comment reaction added for X reason**

**reversible true**

**linked\_gene**

**#reactant/product #stoichio:compound\_id:compart**

**reactant 1.0:compound\_a:c**

**reactant 2.0:compound\_d:c**

**product 1.0:compound\_c:c**

### Apply changes

* 1. To apply the changes described in the *my\_form\_file.csv* form file, run this command:

**$> aureme --run=***test* **--cmd=″curation NETWORK=***network\_name* **NEW\_NETWORK=***new\_network\_name* **DATA=***my\_form\_file.csv***″**

WARNING: If you don’t precise any **NEW\_NETWORK** name, the current network will be overwritten.

Note that all the manual curations made are stored in history files in the **manual\_curation⏵history** directory. You can use them to do the same corrections on other networks for example.

# FAQ

## Can I have a sample of AuReMe?

To get an overview of AuReMe, you can get a sample by using this command:

**$> aureme --sample**

You will get a folder named ‘aureme\_sample’ in your bridge directory. This folder contains all input and output files as if you had run the entire metabolic network reconstruction workflow for the example files about *Tisochrysis lutea (brown algae)*.

Look at the logs file to understand the different commands used in the reconstruction process.

Note: if you do not want to pollute your log files when testing things in your sample run, do not forget to use the **quiet** argument in your command(s) if you wish NOT to store any log, this way:

**$> aureme –run=***aureme\_sample* **–-cmd=”***some\_command some\_arguments***” -q**

## How to convert files to different formats?

The AuReMe workspace natively provides several functions for formats conversion, through the PADMet Python package. The available convertors are:

* From sbml to padmet format:

**$> aureme --run=***test* **–-cmd=”***draft***”**

This command will convert all sbml in networks folder of ‘*test*’ to one padmet. If you want to convert one sbml to padmet format, simply put this file in networks folder of your run and make sure there is no other sbml file, then run the command. If you want to merge many sbml to one padmet, add all of them in networks folder then run the command. Ensure that there is no other files in network folder before running the command, in the case of sbml they could be added and in other case a reading error could occur.

* From padmet to sbml format:

**$> aureme --run=***test* **–-cmd=”***padmet\_to\_sbml NETWORK=my\_network [LVL=3]***”**

This command will convert the padmet file *my\_network.padmet* from networks folder of ‘*test*’ to create a sbml file *my\_network.sbml*. By default the sbml level is set to ‘*3*’, you can change the default value in the config.txt file or with the argment LVL (3 or 2)

* From txt to sbml format:

**$> aureme --run=***test* **–-cmd=”***compounds\_to\_sbml CPD=/path/to/txt\_file.txt***”**

This command will convert a txt file containing compounds ids to a sbml file */path/to/txt\_file.sbml*. The txt file must contain one compound id by line and optionally the compartment of the id which by default is ‘c’. Example of file:

**WATER**

**ATP\tC-BOUNDARY**

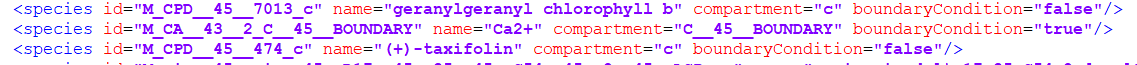
* From GFF/GBK to FAA format:

**$> aureme --run=***test* **–-cmd=”***gbk\_to\_faa GBK=/poth/to/gbk\_file OUTPUT=/path/to/output\_file***”**

**NOTE**: AuReMe integrate some scripts from padmet-utils tools. for example, gbk\_to\_faa command use the script /programs/padmet-utils/connection/gbk\_to\_faa.py. Not all functions are encapsulated in AuReMe, there is a lot of scripts that could be helpful. For more information, see https://gitlab.inria.fr/maite/padmet-utils.

## How to manage growth medium?

In AuReMe, a compound is defined as a part of the growth medium (or ‘seeds’ for gap-filling tools) if this compound is in the compartment ‘C-BOUNDARY’.

The growth medium is linked to the metabolic network by two reactions, a non-reversible reaction named ‘TransportSeed-*compound-id*’ which transport a compound of the growth medium from the compartment ‘C-BOUNDARY’ to the ‘e’ (extra-cellular) and a reversible reaction named ‘ExchangeSeed-*compound-id’* which exchange the same compound from ‘e’ to the ‘c’ (cytosol). When creating a sbml file, the compounds in the ‘C-BOUNDARY’ compartment will be set as ‘BOUNDARY-CONDITION=TRUE’ to allow flux (see <http://sbml.org/Documents/FAQ#What_is_this_.22boundary_condition.22_business.3F>).

Note: Some metabolic networks manage the growth medium with a reversible reaction which consume nothing and produce a compound in the ‘c’ compartment. We chose not to do the same for clarity and because it made some dedicated tools for metabolic network crash.

* Get the list of compounds corresponding to the growth medium of a network in padmet format:

**$> aureme --run=***test* **–-cmd=”***get\_medium NETWORK=network\_name***”**

Return a list of compounds or an empty list

* Set the growth medium of a network in padmet format:

**$> aureme --run=***test* **–-cmd=”***del\_medium NETWORK=network\_name [NEW\_NETWORK=new\_network\_name]***”**

This command will remove the current growth medium if existing, then create the new growth medium by adding the required reactions as described before.

* Delete the growth medium of a network in padmet format:

**$> aureme --run=***test* **–-cmd=”***set\_medium NETWORK=network\_name [NEW\_NETWORK=new\_network\_name]***”**

This function will remove all reactions consuming/producing a compound in ‘C-BOUNDARY’ compartment.

WARNING: If you don’t precise any **NEW\_NETWORK** name, the current network will be overwritten.

## How to manage metabolic network compartment?

In a metabolic network a compound can occur in different compartment. Given a reaction transporting CA2+ from ‘e’ (extra-cellular compartment) to ‘c’ (cytosol compartment), the compartments involved are ‘e’ and ‘c’. It is important to properly manage the compartments defined in a network to ensure a correct connection of the reactions. In some case metabolic networks can use different id to define a same compartment like ‘C\_c’, ‘C’, ‘c’ for cytosol, merging those networks could leak to a loss of network connectivity. A reaction producing CA2+ in ‘c’ and a reaction consuming CA2+ in ‘C\_c’ are actually not connected, hence the interest of the metabolic network compartment management commands of AuReMe.

* Get the complete list of compartment from a network in padmet format:

**$> aureme --run=***test* **–-cmd=”***get\_compart NETWORK=network\_name***”**

Return a list of compartment or an empty list

* Change the id of a compartment from a network in padmet format:

**$> aureme --run=***test* **–-cmd=”***change\_compart NETWORK=network\_name OLD=old\_id NEW=new\_id [NEW\_NETWORK=new\_network\_name]***”**

This command will change the id of the compartment ‘*old\_id*’ to ‘*new\_id*’. This command is required if different ids are used to define a same compartment, example changing ‘C\_c’ to ‘c’, or ‘C-c’ to ‘c’…

* Delete the growth medium of a network in padmet format:

**$> aureme --run=***test* **–-cmd=”***del\_compart NETWORK=network\_name compart=compart\_id [NEW\_NETWORK=new\_network\_name]***”**

This function will remove all reactions consuming/producing a compound in ‘*compart\_id*’ compartment.

WARNING: If you don’t precise any **NEW\_NETWORK** name, the current network will be overwritten.

## How to manage the log files?

By default, the system registers all the executed commands as a log in the ***bridge*⏵*test*⏵*log.txt*** file. The whole output of these commands will also be stored in another file: the ***bridge*⏵*test*⏵*full\_log.txt*** file.

If you DO NOT wish to store such logs, you can use the **quiet** argument in your command(s). For example:

**$> aureme --run=***test* **–-cmd=”***some\_command***” -q**

It is possible to re-run a previous command by copying the corresponding command line in the ***bridge*⏵*test*⏵*log.txt*** file, and pasting it in the Docker container terminal.

To be able to reproduce the whole workflow applied in a previous study, please see the **“FAQ ⏵ Ho to reproduce studies ”** section.

## How to reproduce studies?

If you want to re-run the complete workflow of a pre-run study, built with AuReMe:

* first of all please create a new study (as described in the **“Requirements** ⏵ **Define the reference database ”** section) by running the init command:

**$> aureme –-init=***my\_run2*

(You can choose any run name you want, except pre-existing runs. Please, avoid other special character than ‘\_’ and numbers)

It generates a new folder named *my\_run2* in the *bridge* directory.

* Now, copy all the input data from the previous study in this new folder (please, follow the folder architecture described in the **“Data organization”** section).
* Copy also the ***log.txt*** file in the ***bridge*⏵*my\_run2*** directory. In this log file, change every occurrence of the previous run name by ***my\_run2***.
* Execute this log file.

**$> ./shared/my\_run2/log.txt**

## How to create a new ‘à-la-carte’ workflow?

If you want to add a new step in the workflow or add a new method, it is possible to customize AuReMe. For that it is necessary to update the Makefile in your run. Here is an example of how to do it.

* Add a new method:

First, install your tool by following the documentation associated. For the example we will add a new tool for orthology-based reconstruction ‘new\_tool’ which use the same input as Pantograph (a metabolic network in sbml format, a gbk of the reference species and the gbk of the study species) and generate the same output (a metabolic network in sbml format).

Secondly we will update the Makefile by adding these lines:

**new\_tool:**

**@for dir in $(ORTHOLOGY\_MODEL\_FOLDER)/\*;\**

**do if [ -d "$${dir}" ] && ! [ -e $(new\_tool\_output)\_$$(basename $$dir).sbml ];\**

**then echo "------>RUNNING STEP : Running new\_tool ";\**

**‘INSERT HERE THE COMMAND REQUIRED TO RUN NEW\_TOOL’; fi; done**

Basically this command says that for each folder in orthology\_based\_reconstruction (variable declared in config.txt), if the expected output is not already created, run new\_tool.

Finally, to select this method in your new workflow, change in the file config.txt the variable ORTHOLOGY\_METHOD=pantograph by ORTHOLOGY\_METHOD=new\_tool

* Add a new step or function:

Just update the Makefile by adding a new step and use it with this command

**$> aureme --run=***my\_run* **--cmd=”my\_new\_function some\_argments”**

## How to choose another reference database?

It is possible to select a reference database among several. You can display the list of all available databases by using this command:

* *The BIGG database*
* *The Metacyc database, version 20.5 with added non-generic reactions*
* *The Metacyc database, version 22.5*
* *The ModelSeed database*

**Available database in Aureme:**

**/home/data/database/BIGG/bigg**

**/home/data/database/BIOCYC/METACYC/20.5 /metacyc\_20.5\_enhanced**

**/home/data/database/BIOCYC/Metacyc/22.0/metacyc\_22.0\_enhanced**

**/home/data/database/MODELSEED/modelSeed**

**$> aureme --run=***aureme\_sample* **--cmd=”getdb” -q**

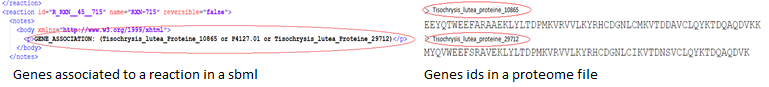
The reference database is needed to:

* be able to match all the identifiers of the entities of metabolic networks
* gap-fill the metabolic network in the gap-filling step

To select one, replace the corresponding path in the configuration file: ***config.txt***, in the ***DATA\_BASE***variable. Or you can comment the line if you don’t want/can’t use a database. The ***config.txt*** file is stored at the root of your ***bridge*** folder (see **“Running a Docker container ⏵ 4.”**).

## What is checked in my input files?

Before running any command in AuReMe, it is highlight recommended to use the command ‘check\_input’. This command checks the validity of the input files and can also create required files. Concretely this command:

* Checks database: If database was specified in the config.txt file (see the **“FAQ ⏵ How to choose another reference database studies ”** section). If so, checks if a sbml version exist and create it on the other hand.
* Checks studied organism data: Search if there is a genbank (gbk/gff) ‘GBK\_study.gbk’ and proteome (faa) ‘FAA\_study.faa’ in genomic\_data folder. If there is only a genbank, create the proteome (command ‘gbk\_to\_faa). If there is only the proteome or any of them, just continue the checking process. Note that the proteome is only required for the orthology-based reconstruction, method: Pantograph.
* Checks orthology-based reconstruction data: for each folder found in ‘orthology\_based\_reconstruction’ folder checks in each of them if there is proteome ‘FAA\_model.faa’ and a metabolic network ‘metabolic\_model.sbml’, if there is no proteome but a genbank file ‘GBK\_study.faa’, create the proteome (command ‘gbk\_to\_faa). Finally, the command compares the ids of genes/proteins between the proteome and the metabolic network.

If cutoff… important because… dict file to create a new proteome file …

* Checks annotation-based reconstruction data: for each folder found in annotation\_based\_reconstruction’ folder checks in each of them if it’s a PGBD from pathway then create (if not already done) a padmet file ‘output\_pathwaytools\_’folder\_name’.padmet in networks/output\_annotation\_based\_reconstruction folder.
* Checks gap-filling data: In order to gap-fill a metabolic network, Pantograph required as input, a file ‘seeds.sbml’ describing the seeds (the compounds available for the network), another describing the targets (the compounds that the network have to be able to reach), the metabolic network to fill and the database from where to draw the reactions all in sbml format. It’s possible to start from txt files for seeds ‘seeds.txt’ and targets ‘targets.txt’, each file containing the ids of the compounds, one by line. The command will then convert them to sbml (command ‘compounds\_to\_sbml’).

Note that by default, AuReMe will integrate the artefacts ‘default\_artefacts\_metacyc\_20.0.txt’ to the seeds to create a file ‘seeds\_artefacts.txt’ and ‘seeds\_artefacts.sbml’. For more information about the artefacts see **“FAQ ⏵ What are ‘artefacts’ ”** section

Example:

------>RUNNING STEP : creating mapping file output\_pantograph\_athaliana\_dict.csv for output\_pantograph\_athaliana.sbml

nb reactions: 892

More than one mapping for reaction R\_R00494\_c: ['RXN-12618', 'RXN-15856']

More than one mapping for compound S\_Starch\_p: ['CPD-8556', 'Starch']

#######

Mapped reactions: 458/892

Reactions with more than one mapping: 4

Mapped species: 53/1318

Species with more than one mapping: 1

Mapped reactions from species: 5

R\_TCM3

R\_TCP26

R\_TCV3

R\_TCX8

R\_TCX10

Total reactions mapped: 463/892

#######

**[output]**

INSERT SCREEN FROM check\_input log

## What is the Makefile?

## What is the config.txt file?

## How to regenerate a new database version?

Voir les notes de Jeanne sur le problème de Sebastian

## How to map a metabolic network on another database?

Metabolic networks can be products of varied databases. If you want to merge efficiently information about metabolic networks coming from different databases, you will need to map the metabolic network(s) to a common database. To do so, a solution is provided be AuReMe.

Note: to use this method, the metabolic network to map needs to be in the SBML format and stored in the ***networks*** folder.

* First of all, you need to know the origin database of the data. To recognize the database used in an SBML file, use the ***which\_db*** command:  
  Example:

Check from which database is this sbml:

Database ref:kegg

{'Unknown': 61, 'total\_rxn': 892, 'kegg': 831}

**$> aureme --run=***aureme\_sample* **--cmd=″which\_db SBML=***output\_pantograph\_athaliana.sbml***″ -q**

**[output]**

* When you know the origin database of the data, you have to generate the mapping dictionary from this database to the new one:

**$> aureme --run=***aureme\_sample* **--cmd=″sbml\_mapping SBML=***output\_pantograph\_athaliana.sbml* **DB=***metacyc***″ -q**

Example:

------>RUNNING STEP : creating mapping file output\_pantograph\_athaliana\_dict.csv for output\_pantograph\_athaliana.sbml

nb reactions: 892

More than one mapping for reaction R\_R00494\_c: ['RXN-12618', 'RXN-15856']

More than one mapping for compound S\_Starch\_p: ['CPD-8556', 'Starch']

#######

Mapped reactions: 458/892

Reactions with more than one mapping: 4

Mapped species: 53/1318

Species with more than one mapping: 1

Mapped reactions from species: 5

R\_TCM3

R\_TCP26

R\_TCV3

R\_TCX8

R\_TCX10

Total reactions mapped: 463/892

#######

**[output]**

In this example, the system has found more than just one mapping for the *R\_R00494\_c* reaction and the *S\_Starch\_p* compound. It did not manage to choose between the propositions: the mapping will not be added to the output mapping. If you want to force the mapping, you have to modify the mapping file manually.

* Once you have created a mapping dictionary file, it will be automatically applied across the workflow to translate the data.

## How to generate reports on results?

Create reports on the *network\_name* network (in the ***networks*** directory). The reports is created in the ***analysis⏵reports*** directory.

**$> aureme --run=***test* **--cmd=″report NETWORK=***network\_name***″**

Crée 4 fichiers bridge/test/analysis/report/network\_name:

* All\_genes :

Id common name linked reactions (;)

* All\_metabolites

dbRef\_id common name Produced (p), Consumed (c), Both (cp)

* All\_pathways

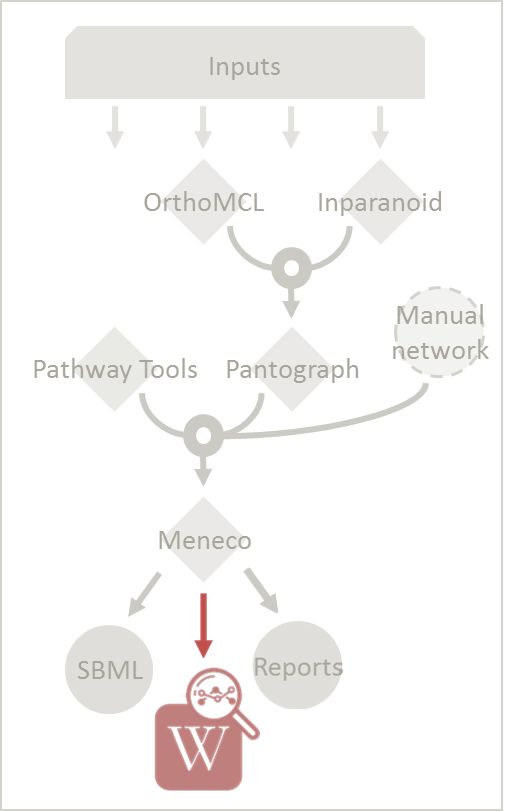
dbRef\_id common name Number of reaction found Total number of reaction Ratio

* All\_reactions

nbRef\_id common name formula (with ID) formula (with common name) in pathway associated genes categories

## How to generate Wiki?

Voir la formation de Méziane



* 1. Create a wiki
     1. Create the wiki pages. The pages will be in analysis/wiki\_pages/network\_name

**$> aureme --run=***test* **--cmd=″wiki\_pages NETWORK=network\_name″**

Wiki\_Docker is an image that allows to automatize the creation of wiki in containers.

* Run the next commands from your machine and not from the AuReMe container.
  + 1. Download the wiki docker image.

**$> docker pull dyliss/wiki-img**

* + 1. Run and setup a container with wiki docker. Follow the instructions to setup correctly the wiki.

**$> make run name=***network\_name*

* + 1. Send the pages and the configuration to the wiki

**$> make init name=***network\_name* **pages\_dir=../analysis/wiki\_pages/***network\_name*

## How to connect to Pathway-tools?

* Create PGDB from output of AuReMe

## What are “artefacts”?

## How to process Flux Balance Analysis?

Notes Mez

To set the objective reaction, please see the following FAQ section.

## How to set an objective reaction?

Notes Mez