MOFA: Microbial Optimization without Forced Altruism

Authors: Soraya Mirzaei and Mojtaba Tefagh

This tutorial demonstrates the use of the MOFA algorithm to compute the maximum community growth rate in a microbial community. (MOFA.py)

Requirements

To run this tutorial, the COBRA Toolbox, IPOPT solver and Pyomo must be installed in the Python environment.

- pip install cobra   (or python -m pip install cobra)
- pip install pyomo   (or python -m pip install pyomo)
- pip install numpy
- pip install pandas
- pip install ipopt

Prepare Input data and models

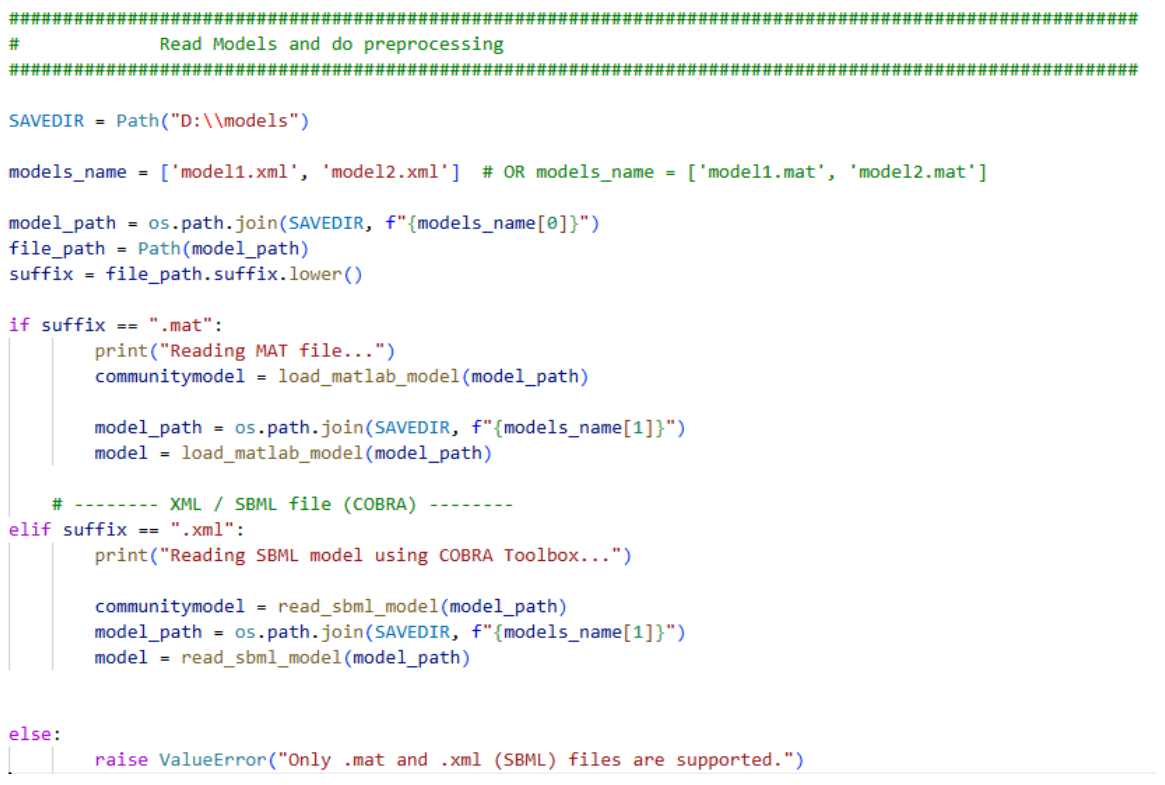
In this tutorial, MOFA is used. Please define the path to the directory containing the species' metabolic network files and specify the biomass reaction name of each species.

- SAVEDIR = Path("D:\\ models")
- models_name = ["model1.xml", "model2.xml"]
- biomass_id=['biomass536_species1','biomass525_species2']

After calling the add_model_to_community() function, the species models are merged to build the community model. You can then set the diet as demonstrated in MOFA_GutMicrobiome.py. Finally, run MOFA.py to compute the maximum community growth rate.

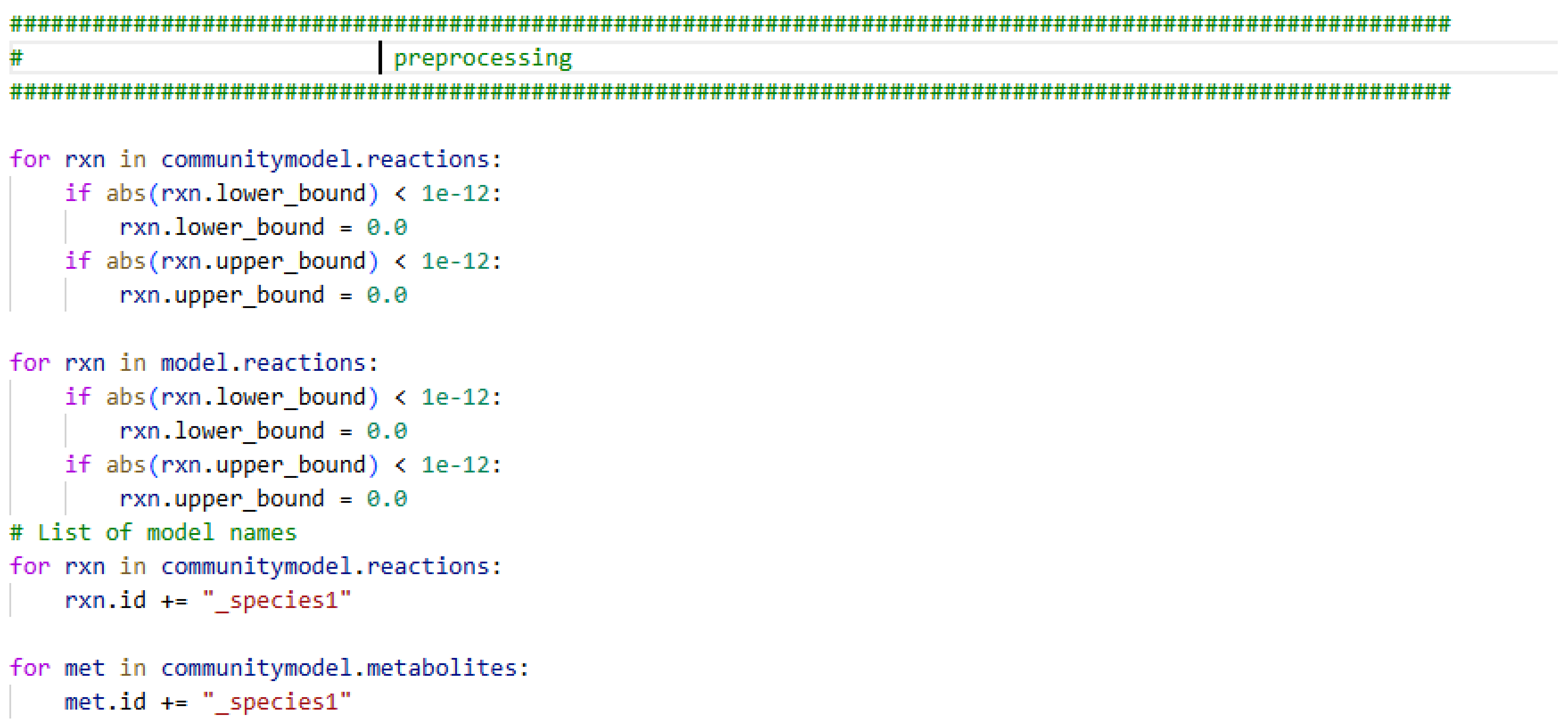
The MOFA.py script consists of five main parts.

First: the first part handles reading the metabolic models, which can be in SBML (.xml) or MATLAB (.mat) format. Please make sure to update the file paths in the code according to your local or working directory.



```
#########################################################################################################
#             Read Models and do preprocessing
#########################################################################################################

SAVEDIR = Path("D:\\models")

models_name = ['model1.xml', 'model2.xml']  # OR models_name = ['model1.mat', 'model2.mat']

model_path = os.path.join(SAVEDIR, f"{models_name[0]}")
file_path = Path(model_path)
suffix = file_path.suffix.lower()

if suffix == ".mat":
        print("Reading MAT file...")
        communitymodel = load_matlab_model(model_path)

        model_path = os.path.join(SAVEDIR, f"{models_name[1]}")
        model = load_matlab_model(model_path)

    # -------- XML / SBML file (COBRA) --------
elif suffix == ".xml":
        print("Reading SBML model using COBRA Toolbox...")

        communitymodel = read_sbml_model(model_path)
        model_path = os.path.join(SAVEDIR, f"{models_name[1]}")
        model = read_sbml_model(model_path)


else:
        raise ValueError("Only .mat and .xml (SBML) files are supported.")
```

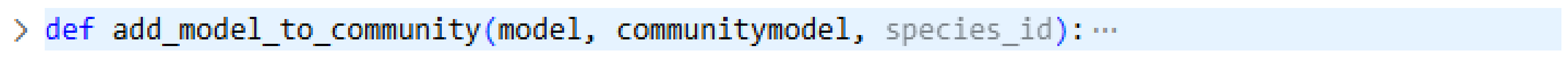
Second, this part of the code may require some preprocessing depending on the user's metabolic network, such as setting exchange reactions or standardizing reaction names. Here, we looped through all reactions in the community model and set any lower or upper bounds that were extremely close to zero (less than 1e-12) exactly to zero. This step eliminates numerical noise and ensures clean flux bounds for subsequent optimization.



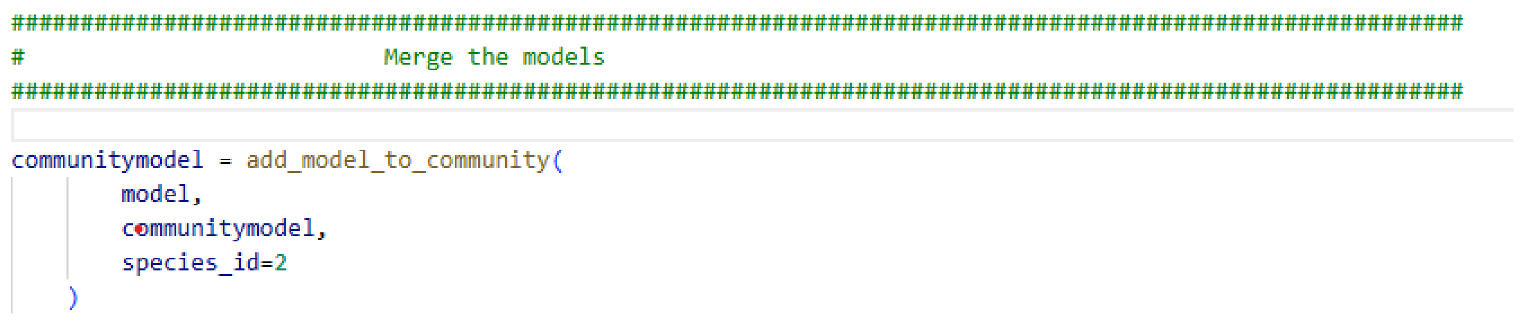
```
###########################################################################################################
#                           preprocessing
###########################################################################################################

for rxn in communitymodel.reactions:
    if abs(rxn.lower_bound) < 1e-12:
        rxn.lower_bound = 0.0
    if abs(rxn.upper_bound) < 1e-12:
        rxn.upper_bound = 0.0

for rxn in model.reactions:
    if abs(rxn.lower_bound) < 1e-12:
        rxn.lower_bound = 0.0
    if abs(rxn.upper_bound) < 1e-12:
        rxn.upper_bound = 0.0
# List of model names
for rxn in communitymodel.reactions:
    rxn.id += "_species1"

for met in communitymodel.metabolites:
    met.id += "_species1"
```

Third: Call the merging method

The add_model_to_community() function combines the metabolic networks of individual species to construct a community model.



```
> def add_model_to_community(model, communitymodel, species_id):···
```



```
###########################################################################################################
#                           Merge the models
###########################################################################################################

communitymodel = add_model_to_community(
        model,
        communitymodel,
        species_id=2
    )
```

Fourth: Set the diet for the community metabolic network, as we did for the gut microbiome.

```
###########################################################################################################
#
#                           Set diet
#
###########################################################################################################
```

The final step involves solving the MOFA nonlinear optimization problem.