# Computation and analysis of microbe-microbe metabolic interactions


## Author: Almut Heinken, Molecular Systems Physiology Group, National University of Ireland Galway.


This tutorial demonstrates how to join a given list of microbial COBRA models in all possible combinations and compute the metabolic

interactions between the microbes depending on the implemented diet. Moreover, the tradeoff between the growth of different joined microbes is computed. The tutorial can be adapted to any number of AGORA models and dietary conditions analyzed.

## Requirements

This tutorial requires the Parallel Computing Toolbox, Bioinformatics Toolbox, and Statistics and Machine Learning Toolbox add-ons in MATLAB.

## Initialize the COBRA Toolbox

```
initCobraToolbox
```

```
                                                |
 _____   _____   _____   _____     _____        |
/  ___| /  _  \ |  _  \ |  _  \   / ___ \       |   COnstraint-Based Reconstruction and Analysis
| |     | | | | | |_| | | |_| |  | |___| |      |   The COBRA Toolbox - 2021
| |     | | | | |  _  { |  _  /  |  ___  |     |
| |___  | |_| | | |_| | | | \ \  | |   | |     |   Documentation:
\_____| \_____/ |_____/ |_|  \_\ |_|   |_|     |   http://opencobra.github.io/cobratoolbox
                                                |

 > Checking if git is installed ...  Done (version: 2.24.3).
 > Checking if the repository is tracked using git ...  Done.
 > Checking if curl is installed ...  Done.
 > Checking if remote can be reached ...
```

## Prepare input data and models

We will use the AGORA resource (Magnusdottir et al., Nat Biotechnol. 2017 Jan;35(1):81-89) in this tutorial. AGORA version 1.03 is available at https://github.com/VirtualMetabolicHuman/AGORA. Download AGORA and place the models into a folder.

```
websave('AGORA-master.zip','https://github.com/VirtualMetabolicHuman/AGORA/archive/mast
try
    unzip('AGORA-master')
end
modPath = [pwd filesep 'AGORA-master' filesep 'CurrentVersion' filesep 'AGORA_1_03' fil
```

Import a file with information on the AGORA organisms including reconstruction names and taxonomy.

```
[~,infoFile,~]=xlsread('AGORA_infoFile.xlsx');
```

Note: if you get a 'file not found' error for AGORA_infoFile.xlsx, please run initCobraToolbox and ensure that the cobratoolbox/papers folder is in the MATLAB path.

## Creation of pairwise models

For the sake of this tutorial, we will use ten random AGORA reconstructions from the info file.

```
modelList = infoFile(randi([2 length(infoFile)],1,10),1);
```

Uncomment the following line to join all AGORA reconstructions in all combinations. NOTE: this is very time-consuming due to the large number of model combinations analyzed.

modelList=infoFile(2:end,1);

You may also enter a custom selection of AGORA reconstructions as a cell array named modelList.

Let us define some parameters for joining the models. Set the coupling factor c, which defined how the flux through all reactions in a model is coupled to the flux through its biomass reaction. Allowed flux span through each reaction= -(c * flux(biomass)) to +(c * flux(biomass)).

```
c = 400;
```

Set the threshold u, which defines the flux through each reaction that is allowed if flux through the biomass reaction is zero.

```
u = 0;
```

Define whether or not genes from the models are merged and kept in the joined models. If set to true the joining is more time-consuming.

```
mergeGenes = false;
```

Define the number workers for parallel pool to allow parallel computing.Recommended if a large number of microbe models is computed. Set to zero if parallel computing is not available.

```
numWorkers = 4;
```

path where to save results

```
mkdir('Results');
resPath = [pwd filesep 'Results'];
```

Join the models in all possible combinations.

```
joinModelsPairwiseFromList(modelList,modPath,'pairwiseModelFolder', resPath,'c',c,'u',u
```

## Computation of pairwise interactions

The interactions between all microbes joined in the first step will be simulated on given dietary conditions. Here, we will use four dietary conditions used in Magnusdottir et al., Nat Biotechnol. 2017.: Western Diet without oxygen, Western Diet with oxygen, High fiber diet without oxygen, and High fiber diet with oxygen. Let us define the input parameters for the simulation of pairwise interactions.

Name the four dietary conditions that will be simulated.

```
conditions = {'WesternDiet_NoOxygen','WesternDiet_WithOxygen','HighFiberDiet_NoOxygen',
```

Define the corresponding constraints to implement for each diet. The input file needs to be a string array.

```
dietConstraints{1}={'EX_fru[u]','-0.14899','1000';'EX_glc_D[u]','-0.14899','1000';'EX_g
dietConstraints{2}={'EX_fru[u]','-0.14899','1000';'EX_glc_D[u]','-0.14899','1000';'EX_g
dietConstraints{3}={'EX_fru[u]','-0.03947','1000';'EX_glc_D[u]','-0.03947','1000';'EX_g
dietConstraints{4}={'EX_fru[u]','-0.03947','1000';'EX_glc_D[u]','-0.03947','1000';'EX_g
```

NOTE: if you design your own diet, make sure that exchange reaction abbreviations correspond to the lumen exchanges in the joint models ('EX_compound[u]').

Define what counts as significant difference between single growth of the microbes and growth when joined with another microbe-here we choose 10%.

```
sigD = 0.1;
```

Simulate the pairwise interactions on the four dietary conditions.

```
for i = 1:length(conditions)
    % assign dietary constraints
    [pairwiseInteractions]=simulatePairwiseInteractions(resPath,'inputDiet',dietConstra
Interactions.(conditions{i})=pairwiseInteractions;
end
```

## Analysis of computed pairwise interactions

The computed microbe-microbe interactions will be plotted by type and analyzed in the context of the taxonomy of the joined strains. There are six possible types of interactions total that can result in increased growth (+), no change in growth (=) or decreased growth (-) compared with the single condition for each joined microbe.

- Competition (-/-)
- Parasitism (+/-)
- Amensalism (=/-)
- Neutralism (=/=)
- Commensalism (+/=)
- Mutualism (+/+)

This results in nine different outcomes total from the perspective of each joined microbe.

Plot the percentage of interactions computed.

```
figure('rend','painters','pos',[10 10 900 600])
typesIA=unique(pairwiseInteractions(2:end,10));
for i = 1:length(conditions)
    pairwiseInteractions=Interactions.(conditions{i});
    listIA=pairwiseInteractions(2:end,10);
    for j=1:length(typesIA)
        dat(j)=sum(strcmp(listIA(:),typesIA{j}));
```

```
    end
    subplot(2,2,i)
    pie(dat)
    set(gca,'FontSize',10)
    h=title(conditions{i});
    set(h,'interpreter','none')
    title(conditions{i})
end
legend1=legend(typesIA);
set(legend1,'Position',[0.42 0.45 0.2 0.2],'FontSize',12)
suptitle('Percentage of computed pairwise interactions')
```

Next, the percentage of interactions will be calculated on different taxon levels (genus, family, order, class, phylum) using the taxon information contained in AGORA_infoFile.xlsx. Here, the interactions will be considered from the perspective of each joined microbe resulting in nine possible interactions total.

Calculate the percentage of interactions predicted for each taxon included in the list of microbes analyzed.

```
for i = 1:length(conditions)
    pairwiseInteractions=Interactions.(conditions{i});
    [InteractionsByTaxon]=calculateInteractionsByTaxon(pairwiseInteractions,infoFile);
    TaxonSummaries.(conditions{i})=InteractionsByTaxon;
end
```

Combine the four conditions into one structure.

```
InteractionsByTaxonCombined=struct;
for i = 1:length(conditions)
    InteractionsByTaxon=TaxonSummaries.(conditions{i});
    taxLevels=fieldnames(InteractionsByTaxon);
    if i==1
        for j=1:length(taxLevels)
            InteractionsByTaxonCombined.(taxLevels{j})=InteractionsByTaxon.(taxLevels{j
            InteractionsByTaxonCombined.(taxLevels{j})(2:end,1)=strcat(InteractionsByTa
        end
    else
        for j=1:length(taxLevels)
            rowLength=size(InteractionsByTaxonCombined.(taxLevels{j}),1);
            InteractionsByTaxonCombined.(taxLevels{j})=[InteractionsByTaxonCombined.(ta
            InteractionsByTaxonCombined.(taxLevels{j})(rowLength+1:end,1)=strcat(Intera
        end
    end
end
```

Let us plot the distributions of interactions for all dietary conditions combined on the level of genera as an example. Note: The xticklabels/yticklabels function is only available in MATLAB R2016b or newer. Older versions of MATLAB will be unable to display the labels.

```
for i=5
    xlabels=InteractionsByTaxonCombined.(taxLevels{i})(1,2:end);
    ylabels=InteractionsByTaxonCombined.(taxLevels{i})(2:end,1);
    data=string(InteractionsByTaxonCombined.(taxLevels{i})(2:end,2:end));
    data=str2double(data);
```

```
    figure;
    imagesc(data)
    colormap('hot')
    colorbar
    set(gca,'xtick',1:length(xlabels));
    xticklabels(xlabels);
    set(gca,'ytick',1:length(ylabels));
    yticklabels(ylabels);
    xtickangle(90)
    set(gca,'TickLabelInterpreter', 'none');
    title(taxLevels{i})
end
```

## Pareto optimality analysis

Another way to analyze the metabolic interactions between two microbes in Pareto optimality analysis. In this method, the tradeoff between two competing objectives (e.g., the biomasses of two joined microbes) is calculated. The resulting Pareto frontier depicts all possible outcomes of co-growth between the two microbes under the given constraints.

Let us compute the Pareto frontier for five randomly chosen pairs from the list of AGORA models.

```
modelInd = randi([2 length(infoFile)],2,5);
```

The Pareto frontier will be computed on the Western diet without oxygen.

```
dietConstraints{1}={'EX_fru[u]','-0.14899','1000';'EX_glc_D[u]','-0.14899','1000';'EX_g
```

By default, the points of the frontier will be generated at steps of 0.001.

```
dinc=0.001;
```

Perform the Pareto optimality analysis for the five pairs. The shape of the computed Pareto frontier, which represents all possible optimal solutions of simultaneously optimized growth, depends on the metabolic networks of the two joined microbes.

```
for i=1:size(modelInd,2)
    models={};
    model=readCbModel([modPath filesep infoFile{modelInd(1,i),1} '.mat']);
    models{1,1}=model;
    bioID{1,1}=model.rxns(find(strncmp(model.rxns,'biomass',7)));
    nameTagsModels{1,1}=strcat(infoFile{modelInd(1,i),1},'_');
    model=readCbModel([modPath filesep infoFile{modelInd(2,i),1} '.mat']);
    models{2,1}=model;
    nameTagsModels{2,1}=strcat(infoFile{modelInd(2,i),1},'_');
    bioID{2,1}=model.rxns(find(strncmp(model.rxns,'biomass',7)));
    [pairedModel] = createMultipleSpeciesModel(models,'nameTagsModels',nameTagsModels);
    [pairedModel]=coupleRxnList2Rxn(pairedModel,pairedModel.rxns(strmatch(nameTagsModel
    [pairedModel]=coupleRxnList2Rxn(pairedModel,pairedModel.rxns(strmatch(nameTagsModel
    pairedModel=useDiet(pairedModel,dietConstraints{1});
    [ParetoFrontier] = computeParetoOptimality(pairedModel,strcat(infoFile{modelInd(1,i
end
```

Can you interpret the shapes of the five Pareto frontiers that were computed? Are there microbe pairs that are always competing with each other? Are there pairs in which one microbe can benefit the other at certain points in the curve and vice versa?