FlexFlux
a flexible Java framework to perform flux balance analyses with regulatory constraints

Lucas MARMIESSE[1], Ludovic COTTRET[1]
[1]Laboratoire des Interactions Plantes-Microorganismes (LIPM), UMR441 INRA, UMR2594 CNRS, Castanet-Tolosan, F-31326, France.
lucas.marmiesse@toulouse.inra.fr, ludovic.cottret@toulouse.inra.fr

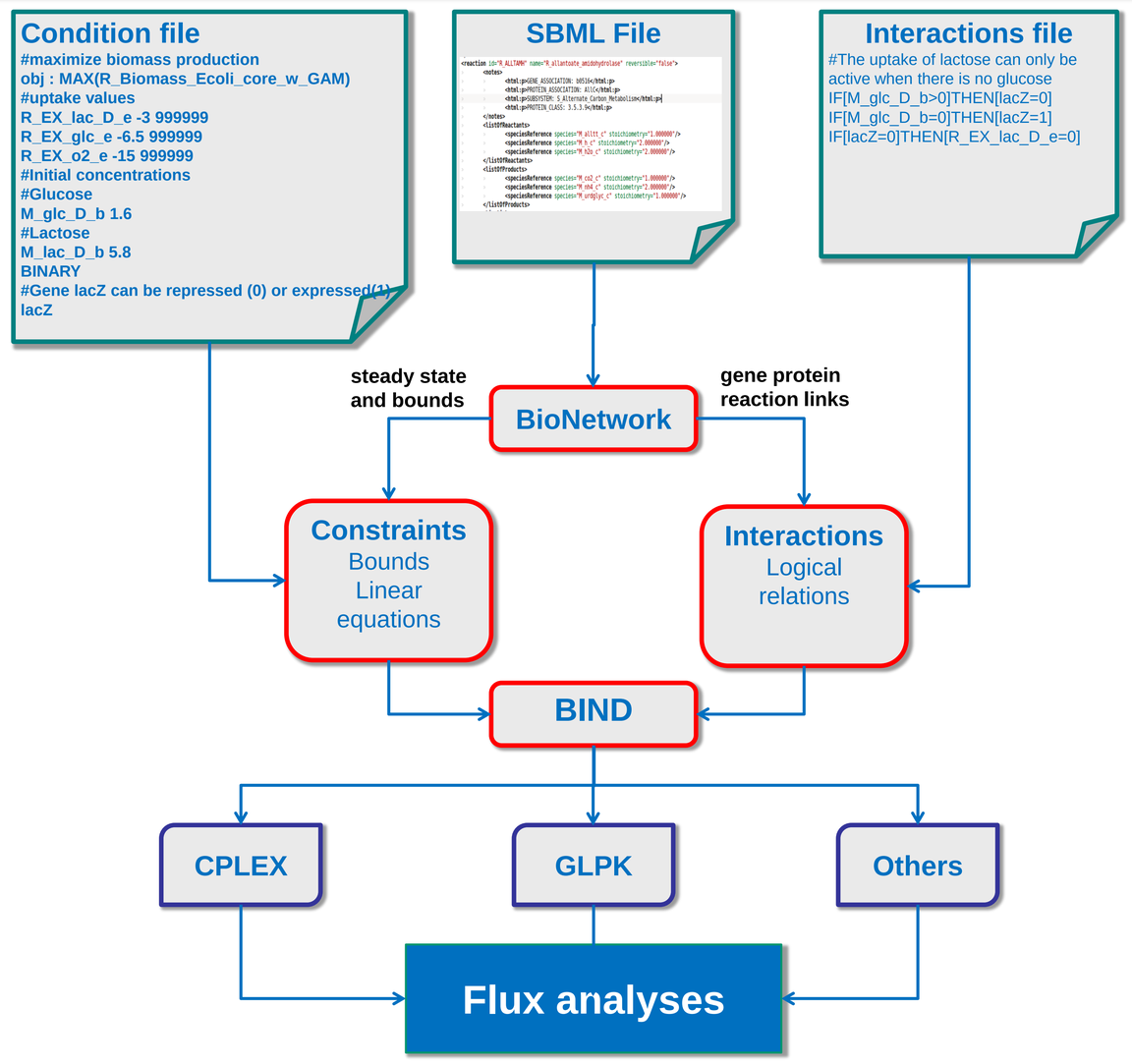
Constraint-based methods allow to analyze the flux distribution of genome-scale metabolic networks under various environmental or genetic conditions[1]. Several software and libraries were developed to perform flux analyses[2-4]. However, it often remains obscur for a modeler not used to informatics to add new constraints into the model, and thus to explore their own biological questions. Furthermore, the existing libraries have rarely been developed in a modular way, which makes difficult the implementation of new flux analysis methods.
We developed FlexFlux, a flux analysis JAVA library. In FlexFlux, the modeler can easily add new constraints or new logical relations to model regulatory links without any informatics knowledge.
Classical high-level functions have been implemented in the FlexFlux framework, such as flux variability analysis, knock out analysis and time dependant flux balance analysis.
Two methods have been also implemented for the first time: comparative flux variability analysis and the best objective fitting functions.
The modularity of FlexFlux allows the developers to easily add new functionalities. At last, the use of parallelization and solver tricks makes FlexFlux one of the fastest flux analysis frameworks.
Availability: FlexFlux sources, Windows and Linux executables, developer and user documentation are available at **http://lipm-bioinfo.toulouse.inra.fr/flexflux**

FlexFlux principles

We mainly focused the development at three levels:
- **Flexibility for the developer:** the library is highly modular and new FBA methods are thus easy to develop. Moreover, several optimization solvers can be bound to FlexFlux without many efforts. For the moment, two solvers are bound: CPLEX and GLPK but the FlexFlux documentation indicates how to implement new solver links. Moreover, JAVA is a free language and does not need any license to use it on the contrary to commercial languages, such as Matlab.
- **Flexibility for the user:** the high-level functions available in FlexFlux allow to address complex biological questions. Moreover, new regulatory constraints and conditions are easy to create. New biological questions are thus easy to address in FlexFlux.
- **Speed:** the use of parallelization and solver tricks makes FlexFlux **one of the fastest flux analysis frameworks.**

FlexFlux architecture

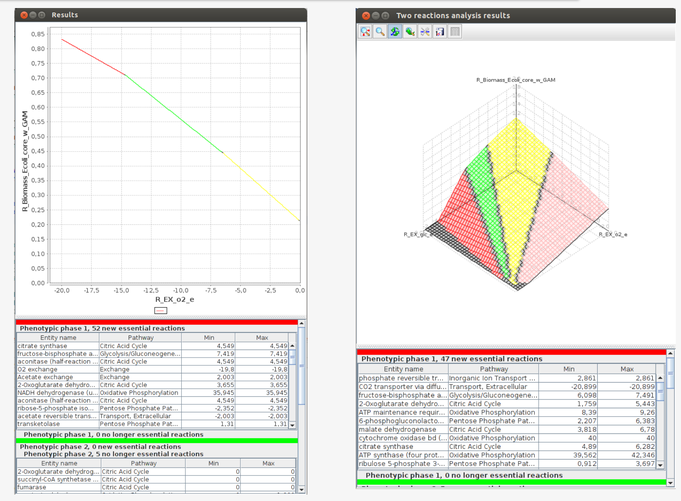
The starting point of FlexFlux is the **BioNetwork** instance which contains all the reactions, metabolites, genes and proteins that are involved in a metabolic network. Especially, it contains all the links between genes proteins and reactions (GPR) and the lower and upper bounds for each reaction flux. A BioNetwork instance is created from a **SBML file** (exchange format for metabolic networks) .
The **condition file** contains objective functions, genetic or environmental constraints on the metabolic network, variables as well as new defined variables defined by the user. Multiple objective functions can be written. In this case, all objective functions are sequentially considered. The optimal value of an objective function becomes a new constraint (with a percentage of liberty than can be specified) for the following optimization.
The **interaction file** contains relation links between variables . An interaction works with two parts: a condition that is checked and a consequence that will be done if the condition is true.
The problem is **translated into linear or mixed integer linear programming statements**. Then, an external optimization library solves the problem. For the moment, only CPLEX and GLPK are bound but it's easy to bind other optimization libraries.



FlexFlux high-level functions

- **Identification of dead reactions,** i.e reactions that are not able to carry any flux and thus clearly represent missing information in the model
- **Flux Balance Analysis (FBA):** returns the value of the optimized objective function as well as all variable values, considering genetic, environmental and regulatory constraints
- **Flux Variability Analysis (FVA):** returns the objective value, and the minimum and maximum value for each entity desired [5].
- **Flux Variability Analysis Comparison**: compares the results of two FVA given a metabolic network, an objective function and two different set of constraints. This analysis returns the objective value, the minimum and maximum value for each entity desired, for both conditions.
- **Knock out analysis:** consists in setting network entities (genes, reactions or user defined) values to 0, and observe the effect on the objective function [1].
- **Time dependant FBA:** is based on external metabolic concentrations and cell density. Given initial metabolite concentrations, cell density, a time step and a number of iterations, this analysis returns the value of each metabolite and cell density for each time [6].
- **Phenotypic phase analysis**: computes the objective value for a range of values of one or two fluxes. This allows to identify flux range sets where the metabolic network has the same behavior [7].
- **Pareto analysis:** determines which objective (or set of objectives) is optimized under some experimental conditions [8].

Phenotypic phase analysis

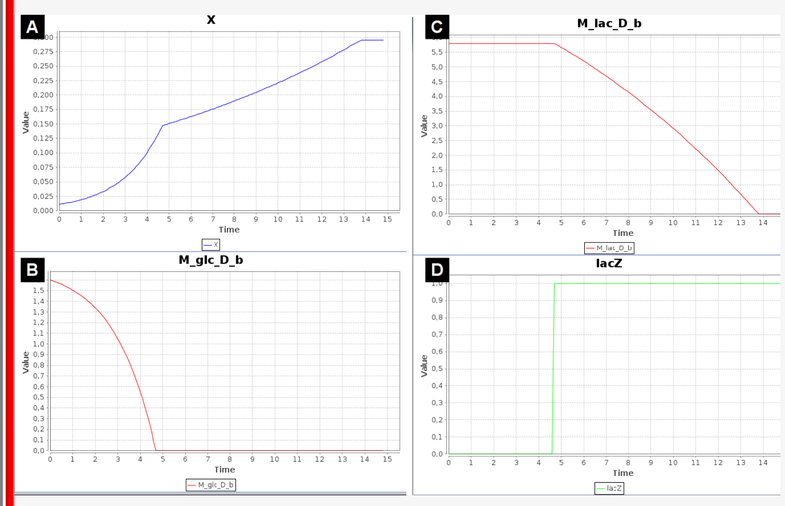


Effect of varying one flux on the optimal growth rate

Effect of varying two fluxes on the optimal growth rate

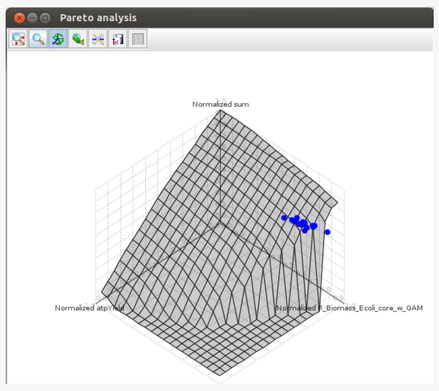
Phenotypic phases are calculated by optimizing the objective function with one or two changing reaction fluxes. Phenotypic phases (represented in different colors) corresponds to a specific metabolic network behavior. To give a hint on what reactions are active in each phase, a FVA is applied on each and the result is graphically shown : essential reactions are displayed.

Time dependant FBA



For each plot, the X axis represents the time.
A. Cell density variation.
B. External glucose concentration.
C. External lactose concentration.
D. LacZ gene activation.
While the glucose concentration is greater than 0, the gene lacZ is repressed. When all the glucose is used, lacZ is not repressed and the cell can use the lactose to grow. To go further in the analysis, interaction activation and persistence times can be specified.

Best cellular objective fitting



Each axis represents one objective function. The surface, called Pareto surface, is composed by points that are "Pareto optimal" that are defined by the fact that the value of one objective can be increased only at the cost of another. The blue points represent experimental values. The position of the blue points from the Pareto surface enables to identify which objectives are optimized in the experimental conditions.

[1]Orth *et al.*. (2010). What is flux balance analysis? *Nature biotechnology*, **28**(3), 245–8.
[2] Gevorgyan *et al.* (2011). SurreyFBA: a command line tool and graphics user interface for constraint-based modeling of genome-scale metabolic reaction networks. *Bioinformatics* (Oxford, England), **27**(3), 433–4.
[3] Hoppe *et al.* (2011). FASIMU: flexible software for flux-balance computation series in large metabolic networks. *BMC Bioinformatics*, **12**(1), 28.
[4] Schellenberger *et al.* (2011). Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox v2.0. Nat Protoc, 6(9), 1290–1307.
[5]]Gudmundsson *et al.* (2010). Computationally efficient flux variability analysis. *BMC bioinformatics*, **11**(1), 489.
[6] Covert *et al.* (2002). Transcriptional regulation in constraints-based metabolic models of Escherichia coli. *JBC*, **277**(31), 28058–64.
[7] Edwards *et al.* (2002). Characterizing the metabolic phenotype: a phenotype phase plane analysis. *Biotechnology and Bioengineering*, **77**(1), 27-36. John Wiley & Sons.
[8]Schuetz *et al.* (2012). Multidimensional optimality of microbial metabolism. Science, 336(6081), 601–604.