

Chapter 2

Analyzing and Designing Cell Factories with OptFlux

Paulo Vilaça, Paulo Maia, Hugo Giesteira, Isabel Rocha, and Miguel Rocha

Abstract

OptFlux was launched in 2010 as the first open-source and user-friendly platform containing all the major methods for performing metabolic engineering tasks in silico. Main features included the possibility of performing microbial strain simulations with widely used methods such as Flux Balance Analysis and strain design using Evolutionary Algorithms. Since then, OptFlux suffered a major re-factoring to improve its efficiency and reliability, while many features were added in the form of novel plug-ins, such as the BioVisualizer and the over/under expression plug-ins. The current chapter described the main mathematical formulations of the major methods implemented within OptFlux, also providing a detailed guide on the usage of those functionalities.

Key words Computational biology and bioinformatics, Systems biology, Metabolic engineering, Flux-balance analysis

1 Introduction

In industrial biotechnology processes, microorganisms are usually used as microbial cell factories to produce chemical compounds of economic interest, from chemical building blocks, to food ingredients and bioplastics. In order to increase the yield, productivity, and specificity of these industrial processes, strain improvement methods were developed mostly based on random mutagenesis and on screening strains with naturally enhanced production levels [1, 2]. These techniques were labor intensive, but they were the best alternative for strain improvement before advanced genetic engineering procedures became available [3].

The buildup of knowledge on microbial metabolism, combined with the development of more sophisticated genome engineering techniques, gave rise to the rational modifications of microorganisms aiming to improve their phenotypical properties toward a certain goal. This discipline is usually referred to as Metabolic

Isabel Rocha and Miguel Rocha contributed equally to this work.

Engineering (ME) [3–6]. ME has been applied of a vast number of tasks over the past 20 years, due to the extraordinary growth in adoption of industrial biotechnological processes for the production of bulk chemicals, pharmaceuticals, food ingredients, enzymes, among other products [7, 8].

Many different approaches have been used to aid in ME efforts, from which Systems Biology (SB) deserves to be highlighted [9, 10]. SB addresses the computational and mathematical modeling of complex biological systems with the objective of examining the structure and dynamics of cells or organisms and understanding the properties of these systems. In other words, SB is focused on building and validating *in silico* models of biological systems that can be applied to generate novel, testable, and often quantitative predictions of cellular behavior, thus being able to support the rational development of optimized cell factories.

More recently, with the remarkable advances on genome sequencing technologies, culminated by the surge of the next-generation sequencing technologies [11], as well as semi-automated annotation techniques, an increasingly large number of fully annotated microbial genomes are being made available.

The advent of complete genomic sequences allowed the reconstruction of genome-scale networks that can be employed to generate models of diverse cellular processes, such as signaling transduction [12, 13], transcriptional regulation [14], and metabolism [15]. A reconstructed network is defined as a list of biochemical reactions occurring in a particular cellular system (such as metabolism) and the associations between these reactions and relevant proteins, transcripts and genes. A reconstruction can be converted to a mathematical model by including the assumptions necessary for the computational simulations, such as maximum reaction rates and nutrient uptake and production rates. An extensive collection of methods for analyzing metabolic genome-scale models (GSM) have been developed and applied to study a growing number of biological questions [16, 17].

The collection of the stoichiometry and reversibility of all the metabolic reactions from an annotated genome is the starting point for the construction of a GSM. Furthermore, many additional curation steps are required until models are complete. The full process of GSM reconstruction has been described in detail in several publications [18–20] and software tools that can help in the reconstruction process are also available (*see* Chapter 1 for more details) [21–24]. These software tools can be a great help in the reduction of the total time required to reconstruct a GSM and have proven extremely valuable for annotating genomes of less studied organisms.

Besides the stoichiometry and reversibility of all chemical reactions that can occur in a certain organism, GSMs can include additional details, such as the kinetic parameters for each enzymatic

reaction. However, the availability of kinetic information is very scarce, which makes it very challenging to gather these data at the genome scale [25, 26]. Since the biochemical information included in most GSMs is limited to the stoichiometry and reversibility of all reactions, the application of these models is restricted to the steady-state modeling of intracellular fluxes [27, 28].

Since the advent of GSMs in 1999, when the *Haemophilus influenza* GSM was first published [29], the number of available GSMs grew to more than 100 models [30]. Some repositories facilitate the access to some of these models in standard formats, such as www.optflux.org/models or <http://systemsbiology.ucsd.edu/InSilicoOrganisms/OtherOrganisms>.

Many different approaches have been used to identify bottlenecks or targets for genetic engineering that take into account models together with mathematical tools and/or experimental data. Some of these techniques, like Metabolic Control Analysis (MCA), use dynamical representations of the metabolism, while others like Metabolic Flux Analysis (MFA) or Flux Balance Analysis (FBA) consider only the stoichiometry of the system to build a Constraint-Based Model, allowing the study of the phenotype of microorganisms, under different environmental and genetic conditions.

While the need for mathematical and computational tools to aid in ME efforts was already identified by James Bailey in 2001 [31], very few user-friendly software tools were available then and in the following years. *OptFlux* was introduced in 2010 [32] as a proposal to tackle this problem. OptFlux is a user-friendly software tool that aims to be a reference platform for the ME community, and it was developed with the objective of collecting several tools and algorithms that use GSMs in an integrated, extensible, and easy-to-use platform. Some of the main features of this tool are the following:

- Open-source—it allows all users to use the tool freely and invites the contribution of other researchers;
- User-friendly—facilitates its use by users with no/little background in modeling/informatics;
- Modular—facilitates the addition of specific features by computer scientists, given its plug-in based architecture;
- Compatible with standards—compatibility with the System Biology Markup Language (SBML) and layout standards as SBGN.

Optflux accommodates several tools and algorithms that have been developed to help in the analysis of GSMs. In the next sections, it will be explained how to apply these resources using Optflux.

2 Materials

OptFlux version 3 can be downloaded in www.optflux.org/downloads. The software is fully implemented in the Java Language. However, to perform the optimization of constraint-based model (CBM) problems, external software is required. Optflux has embedded two different open source external applications to solve these problems, CLP¹ and GLPK². By default CLP is configured to solve Linear Programming (LP) and Quadratic Programming (QP) problems, while GLPK is configured to solve Mixed-Integer Linear Programming (MILP) problems. An interface for the commercial solver CPLEX³ was implemented. CPLEX provides native support for several classes of constraint-based programming problems, such as LP, QP, and MILP.

Optflux was implemented in such a way that new features are easily plugged in. It is entirely implemented on the top of AIBench [33], a development framework that enforces the Model-View-Controller (MVC) design pattern, incorporating three types of well-defined artifacts: operations (controller), data types (model), and views. This leads to units of work with high coherence that can be easily combined and reused.

Furthermore, it is plug-in based: applications are developed by incorporating components, called plug-ins, each containing a set of functionalities, allowing the reuse and integration of functionalities from past and future developments. The management of OptFlux's plug-ins is easily achieved by a repository manager, which provides an intuitive graphical user interface (GUI), where users can select which plug-ins are installed or updated at each time.

All the OptFlux's source code is available in git repositories located in SourceForge⁴, where it is also possible to submit support requests and bug tickets, keeping the developers aware of the problems reported by the users and helping in the planning of code development timelines.

Figure 1 depicts how OptFlux looks like and highlights the global layout of the platform. Most of OptFlux's main features and operations are accessible to the user either through the *Menu* or the *Toolbar*. Users can also access many of the operations by right-clicking in the *Clipboard* area. All the data types, i.e., projects, metabolic models, environmental conditions, simulation/optimization results, layouts for visualization, etc., are always placed in the *Clipboard* area, data type names can be changed by right clicking the object and selecting the "rename" option. The *Visualization*

¹ <https://projects.coin-or.org/Clp>

² <https://www.gnu.org/software/glpk/>

³ <https://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/>

⁴ <https://sourceforge.net/projects/optflux/>

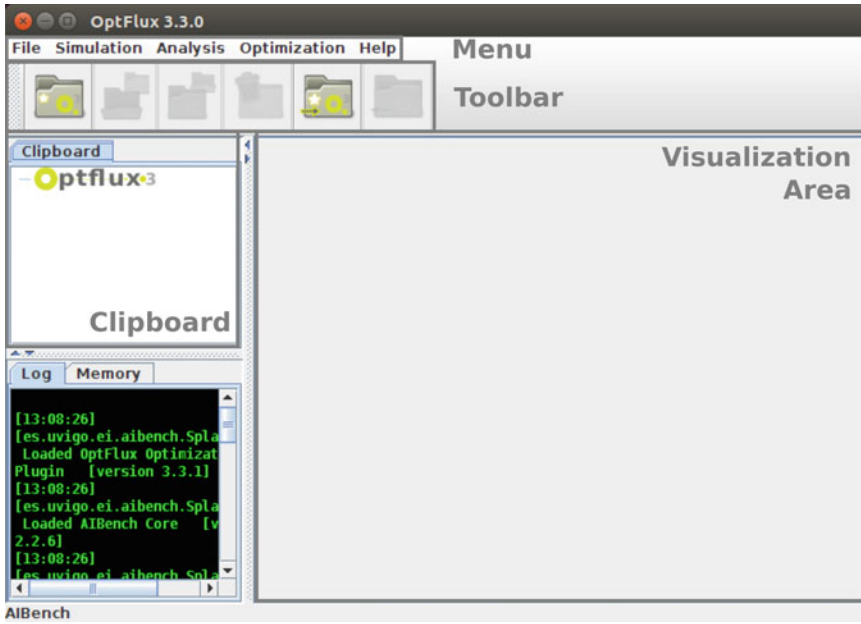


Fig. 1 The OptFlux application interface is split into four main parts identified in the figure: (a) Menu—where the user has access to all the operations existent in OptFlux application; (b) Toolbar—where the user has access to some important short-cuts for operations related to projects such as create, open, close, delete, import, and export project. (c) Clipboard—where the data types will be placed after being created by the operations. (d) Visualization area—where the user will be able to see the data types' contents when clicked on the top of those

Area is where the user can examine those data types in greater detail. When the user clicks in a data type, different views for that object are displayed in this area.

3 Methods

3.1 Constraint-Based Modeling

The formulation of Constraint-based models requires two levels of metabolic information (Fig. 2a). First, metabolic stoichiometry is required to write down all chemical reactions that take place in a metabolic network of interest. The second item is the information of the demands that are placed on the metabolic system. These demands include non-growth-associated maintenance requirements, biomass synthesis, and nutrients inputs. The non-growth-associated maintenance requirements can be obtained from strain-specific experiments [34]. In terms of the biomass synthesis, it is necessary the inclusion of an artificial reaction in the network that represents the cellular growth. Such a reaction is built taking into account the contribution, in millimoles per gram of cell dry weight, of all macromolecules or building blocks and essential cofactors, based on the biomass composition. The next Equation represents

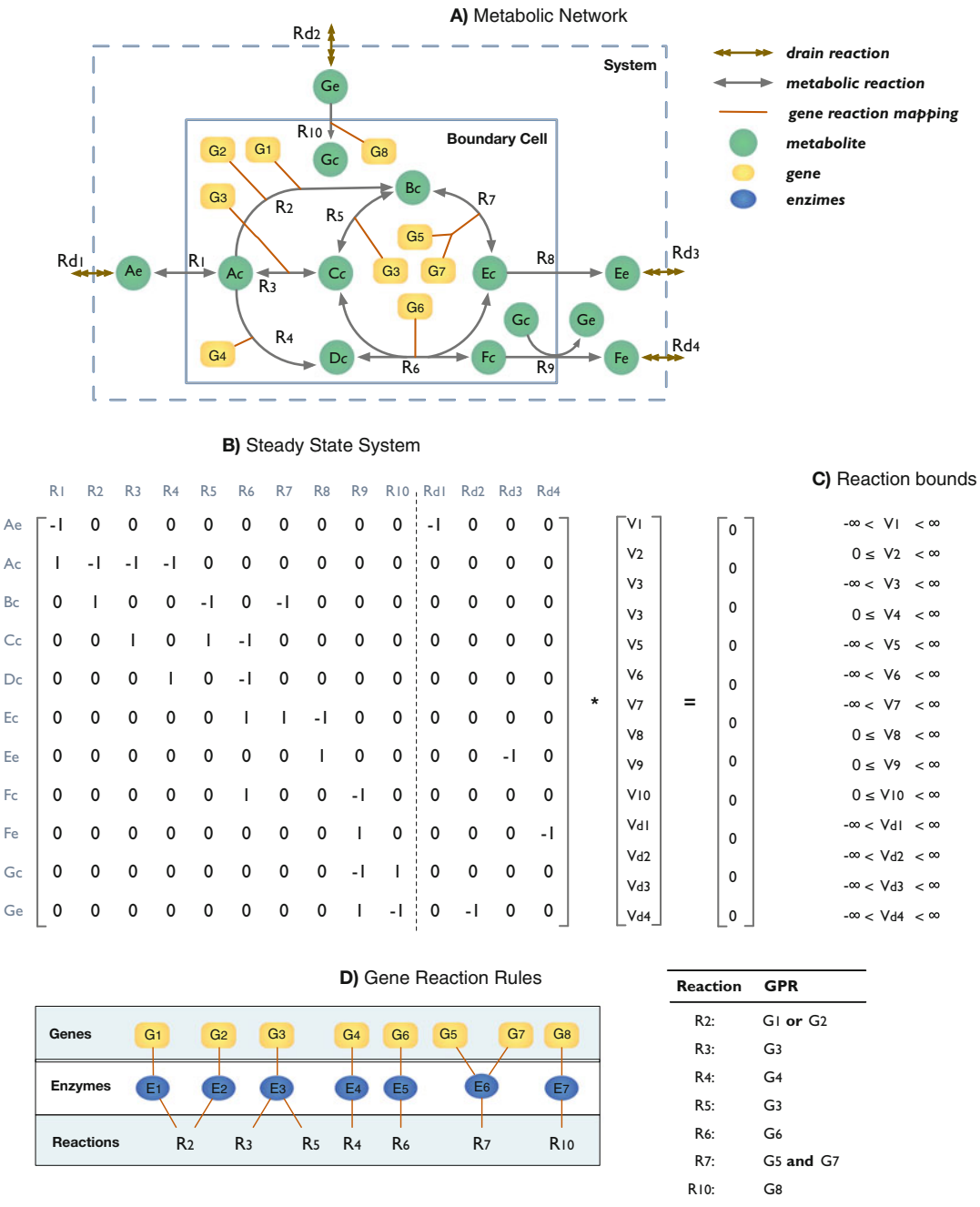


Fig. 2 In the figure a small network is represented (a) composed by various components commonly used in constraint-based models such as metabolites, genes, reactions, and drains. The corresponding steady-state system (b), the bounds of the reactions (c), and gene protein rules (d) are shown

the biomass equation, where M represents the biomass components and the ∂m represents the contribution of these components for the biomass in mmol/(g of biomass).

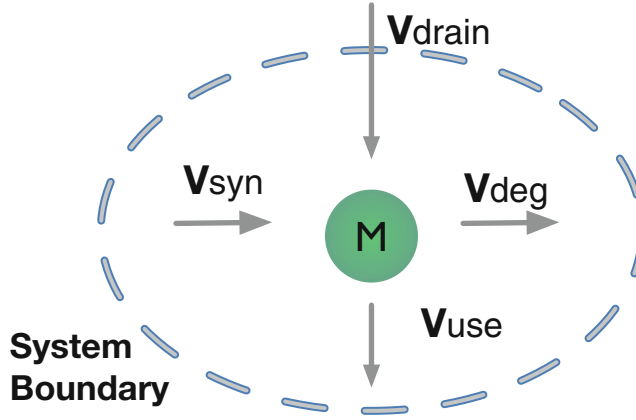


Fig. 3 Constraint-based models use the balance around each metabolite to represent a metabolic network. In the figure, all types of transformations that could be present in each CBM metabolite are represented. The V_{drain} represents the fluxes carrying in and out a metabolite to and from the system. These fluxes deal with the nutrients, as well as the outputs including the by-products of the system. V_{use} represents the fluxes required for the metabolic requirements, such as growth and maintenance. V_{syn} are the fluxes that synthesize each metabolite and V_{deg} are the fluxes that represent the consumption of the metabolite

$$\partial m \cdot M \rightarrow \text{biomass}$$

Such an equation is used to calculate the growth rate of the organism in h^{-1} , and thus the other fluxes on the model are in mmol/gDW/h units.

In order to define the availability of nutrients and excretion of products in/to the medium, some special reactions are built and included in the system, denominated by drains. Drains define which are the metabolites that could be inputs and/or outputs of the modeled organism. The next equation illustrates how those special reactions look like; the reaction is only composed by one metabolite M which is usually defined as a reactant. Thus, when its flux is positive, metabolite M is an output of the system, while, if the flux is negative it is an input of the system.



The mass balance of a specific metabolite M is illustrated in Fig. 3. In the figure, four different rates that can perturb the concentration of M are represented. V_{drain} represents the rate responsible for the translocation of metabolite M . Those rates will be dealing with requirements of the system, i.e., nutrients, as well as the by-products. V_{syn} are the fluxes that synthesize the metabolite and V_{deg} are the fluxes that consume metabolite M . V_{use} represents the rate of utilization of metabolite M for metabolic requirements, such as growth and maintenance. Thus, the conservation law [35] can be applied allowing the representation of the accumulation of metabolite M as

$$\frac{dM}{dt} = V_{\text{syn}} - V_{\text{deg}} - V_{\text{use}} - V_{\text{drain}}$$

CBMs are based on the assumption that metabolic transients are more rapid than both cellular growth rates and the dynamic changes in the organism's environment. Metabolism typically has transients that are shorter than a few minutes, and thus it is reasonable to place the metabolic system in a steady state when investigating aspects of metabolism related to growth. At the steady state, the concentrations of metabolites in a network are constant and the fluxes that generate a metabolite must be equivalent to the fluxes that consume that metabolite. So, the accumulation of an internal metabolite M is zero, as stated in

$$0 = V_{\text{syn}} - V_{\text{deg}} - V_{\text{use}} - V_{\text{drain}}$$

By applying the same principle to all metabolites, a homogeneous system of linear equations can be defined as

$$S^* v = 0$$

In the system, S is an $m \times n$ stoichiometric matrix, for a set of m metabolites and a set of n reactions, and v is the vector of n reaction fluxes. In this representation, it is assumed that all the reactions are mass balanced (Fig. 2b). For each reaction of a metabolic network, bounds for their minimum and maximum values can be configured to reflect thermodynamic feasibility, i.e., reaction directionality, and flux capacity of the reactions (Fig. 2c):

$$lb \leq v_i \leq ub$$

Further details, such as translational/transcriptional representation in the form of Gene-Protein-Reaction (GPR) associations, are also typically included in the models [36]. The representation of GPR associations usually resorts to Boolean logic, where the relationships between reactions and their encoding genes are modeled as logical AND/OR operations. Typically, the AND operator represents the formation of protein complexes, and the OR operator usually represents genes encoding isoenzymes. This allows the identification of which genes are responsible for coding a reaction in the model (Fig. 2d).

3.1.1 Importing Constraint-Based Model in Optflux

Optflux allows several ways of loading a constraint-based model. Due to the absence of clear standards to collect this type of information prior to 2007, each author or CBM platform used its own specific format. With the release of the Systems Biology Markup Language (SBML) in 2003 [37] and subsequent availability of easy-to-use libraries for exchanging models in this format (libSBML) in 2008 [38], this format started to be adopted and became the most used standard for collecting CBM models.

Nonetheless, there are other legacy formats that the community still uses. Because of that, OptFlux is shipped with readers that provide support for loading models from several formats, namely:

1. SBML levels 2 and 3, with the option of using the flux balance constraints plug-in (FBC). OptFlux is able to import models using the SBML specification from level 2 version 1 onward;
2. Metatool reader specification⁵ [39];
3. CellDesigner (SBML specification extended with layout information from CellDesigner);
4. Table format (useful to ease the burden of importing models from Excel datasheets—a format commonly used by authors to publish their new reconstructions).

Furthermore, OptFlux includes access to a model repository—an internal repository of different models and organisms, which makes the access to commonly used models much less cumbersome to OptFlux users. Currently, the repository provides access to nearly 50 models.

All of these readers are available via the same operation in OptFlux *menu*: *File/new Project*. This operation has four main steps, also depicted in Fig. 4:

1. Specify the project name and choose one of the available readers;
2. Define reader parameters; this step is different depending on the chosen reader and can have more than one sub-step;
3. Identify/define drains in the loaded model;
4. Identify biomass reaction.

One of the most important steps in the process of correctly importing a model is the definition of drains. Until this day, there are no standards to unambiguously define the drains on CBM models. Some authors opted for adding boundary metabolites in the model that should be removed during the import process. Other models have been created with no drains, being necessary to create them for all the external metabolites. Finally, some authors include all the necessary drains in the model, effectively eliminating the need for the user to do anything other than correctly importing the model matrix.

The user should understand how the drain reactions are dealt with in the model source file and select the applicable method in OptFlux:

- Do not create drains—if drains are already present in the model;
- Remove external metabolites—if it is necessary to delete boundary metabolites to create the drains;

⁵ http://penguin.biologie.uni-jena.de/bioinformatik/networks/metatool/metatool5.0/ecoli_networks.html

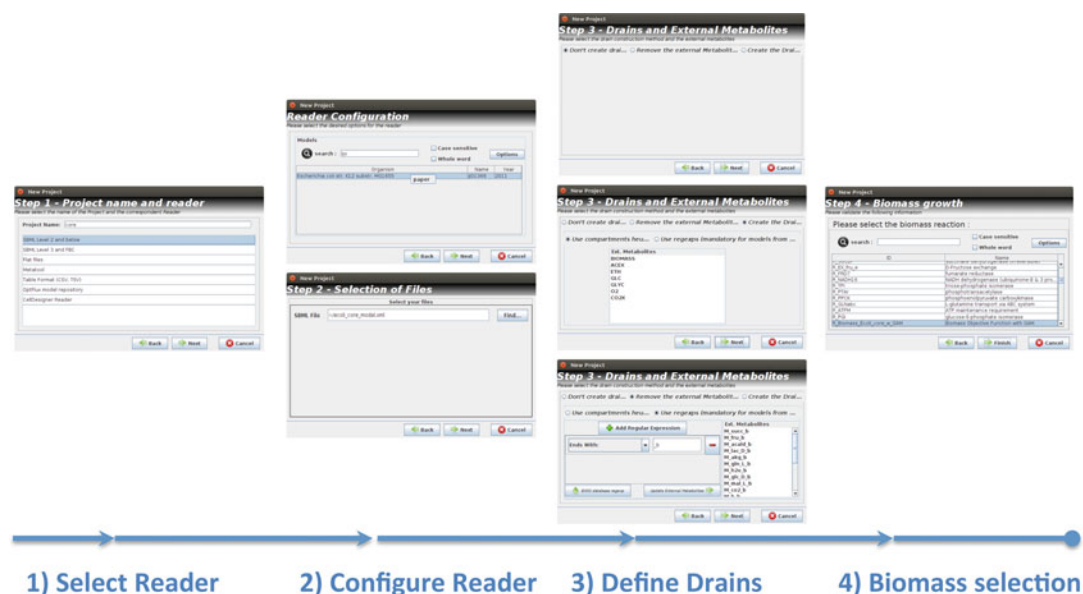


Fig. 4 The process of a project creation in OptFlux includes four main steps: (1) The Reader Selection where the user selects a specific reader to construct a project. OptFlux comes with readers for several formats such as SBML, CellDesigner, Table Format, and Model repository. (2) The reader configurations. Different readers can have different inputs and in this step the user is able to configure those inputs. In the figure, two different reader configurations are shown: the first one is the configuration for the model repository where the user selects one of the models present on the database (she/he can inspect the respective paper by clicking in the second button of the mouse), and the second is the SBML reader where the user only has to select a valid SBML file. (3) Define drains.—Different formats and different authors of models have different forms to define the drains on the models so it is necessary to select the method in this step. (4) Biomass selection, where the user is able to select a reaction to be the biomass artificial reaction; this will help on the usage of several operations on OptFlux

- Create drains—if it is necessary to create new drain reactions for the external metabolites.

When selecting either the “*Remove external Metabolites*” or the “*Create Drains*” options, the user must also select the method by which the external metabolites are detected. OptFlux provides two different ways of doing this:

- Use compartment heuristic—OptFlux begins by identifying the external compartment and subsequently mapping all the metabolites associated with it.
- Use regular expressions—the user is prompted to define a regular expression that OptFlux uses to compute the matching metabolite identifiers.

In Fig. 5, three examples are shown. On the first (line 1) the network already contains drain reactions so the option that should be chosen is “Don’t create drains.” On the second example (line 2), the network does not contain drain reactions, but there are

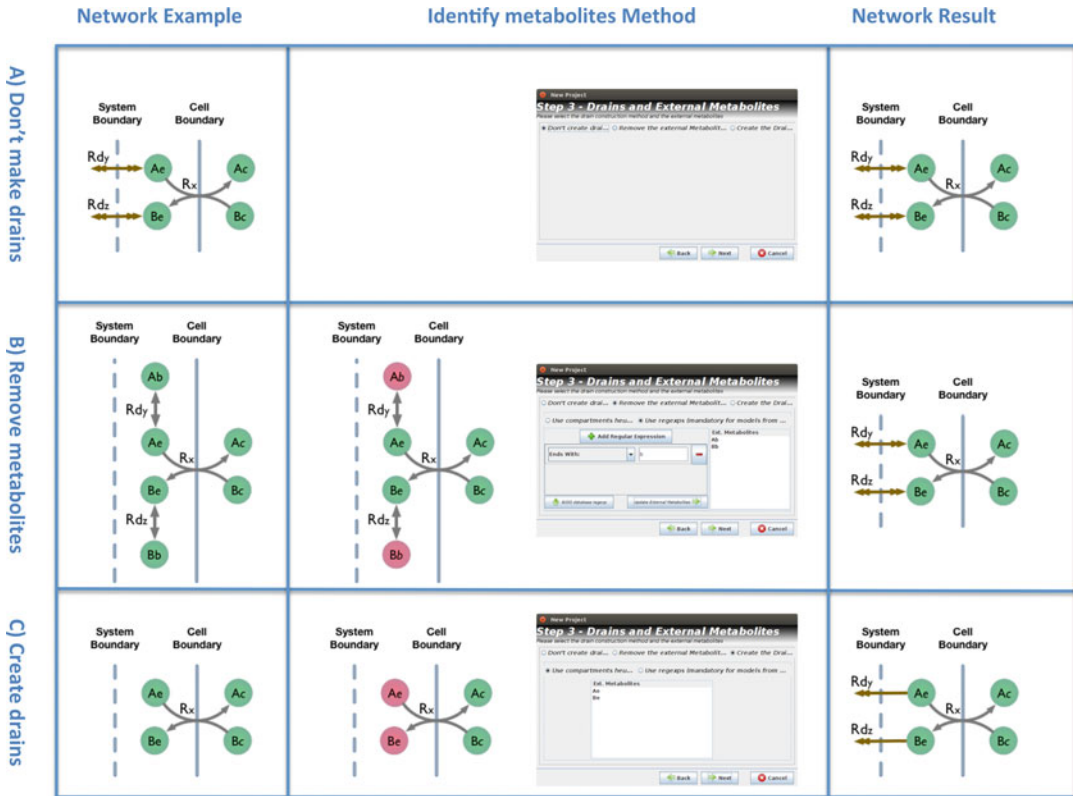


Fig. 5 In the figure, three standard examples of how drains could be stored in the models are schematized (first column), the best option to use in OptFlux to define drains (column 2) and the resulting network after OptFlux defines the drains (column 3)

reactions in the model that contain artificial metabolites that when removed will transform these reactions into drains. So, the option that should be selected in this case is “Remove external metabolites.” Those artificial metabolites are called “boundary metabolites” and for OptFlux to remove them, the user should identify them first. On those examples, the boundary metabolites are in the same compartment as external metabolites so the user can use a regular expression in OptFlux to detect them (in this example “Ends with” “b”).

In the third example (line 3), the network does not contain drains and no artificial metabolites are available, thus the drains need to be created. Since the metabolites where the drains should be added are the external metabolites, the user can use one of the two strategies to identify the metabolites: “External compartment,” where all the metabolites that should have a drain are in the same external compartment, or use a regular expression (“Ends with” “e” on this example).

OptFlux can also use a heuristic method to identify the correct drain configuration, and select the best option by default. Being a

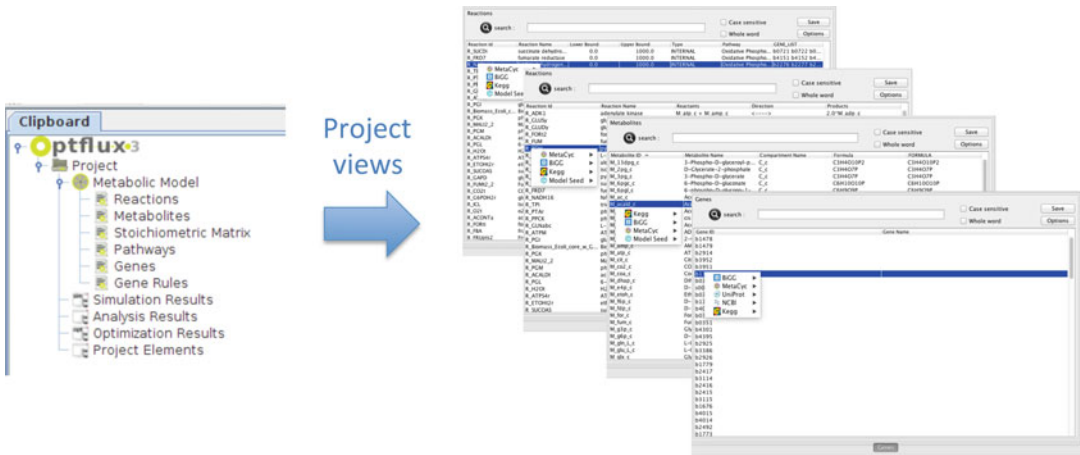


Fig. 6 The Project data type is composed by several sub-data types. The data type that collects the information regarding the loaded model is the “Metabolic Model” (the first data type inside the project). Inside, the user has detailed information about the components present in the model. Each component has a view that opens in the visual area of OptFlux (by clicking on the right button of the mouse over a specific entity in the table, OptFlux will search the entity in several databases)

heuristic, this method is not 100% guaranteed to provide the best result.

Finally, the user is also asked to select a default biomass (growth) equation from the model. This step is required because many operations in OptFlux, such as phenotype predictions and strain optimization, among others, require the biomass equation to be specified in the objective function. By setting the default biomass equation, the repeated use of many of these operations becomes greatly simplified. OptFlux also uses a heuristic to select a reaction that it believes to be the correct one, but the user should always confirm this and change it if necessary.

In the end of the “*New Project*” operation, a new project will appear in the clipboard section with the name chosen by the user in **step 1**. The core data type of the project is named “Metabolic Model” (Fig. 6). Inside, the user can access detailed information about reactions, metabolites, and also the stoichiometry of the system in a human-readable way. Furthermore, genes, gene rules, and pathway information are shown if available.

3.2 Visualization of Data Using CBMs

Within the field of Systems Biology, the analysis of different types of biological networks is an important task in understanding the underlying biological processes. Data can be associated with the biological components facilitating its visualization and interpretation. Visualizing data in this way contextualizes and enriches the dataset. Data-rich visualizations have been extremely valuable for viewing, interpreting, and communicating data. Two-dimensional pathway maps have long been a popular visual representation of metabolic pathways and other biological pathways.

CBM models are not easy to understand in their mathematical form, which turns their analysis and interpretation into a very hard and time-demanding task. To aid in this task, visualization tools have been developed to complement 2-dimensional pathways maps with CBM outputs. CellDesigner [40, 41] is one of the most popular tools for creating, editing, and visualizing biochemical networks, but it lacks specific methods for CBMs. Alternatively, Cytoscape [42] became a standard tool for the integrated analysis and visualization of biological networks, but faces the same problem of the CellDesigner platform, missing a native CBM specification. Regarding these problems, some platforms that integrate visualization technologies with the CBM methodology have been developed. The BioVisualizer platform [43], available as a plug-in for OptFlux, is one of them. BioVisualizer provides a visualization framework based on a well-defined abstract representation of metabolic pathways and CBMs. This plug-in provides a linkage between the constraint-based models and pathway layouts, natively understanding how to interact with each other and allowing the highlight of properties or results based on the use of the CBM.

3.2.1 *Optflux* *Visualization Capabilities*

OptFlux allows the creation of pathway layouts using the BioVisualizer plug-in [43]. In the current version of Optflux, this plugin comes already pre-installed.

The visualization plug-in provides all the functionalities related to the visualization and editing of the metabolic layout. One of these features is the specification of the default colors and shapes of the nodes. The graphical user interface (GUI) is composed of two major elements (Fig. 7): the network view, where it is possible to edit the network and click/drag the nodes (Fig. 7a), and the side panel where filters, overlaps, node information, zoom, and export functionalities are available (Fig. 7b). In this way, it is possible for the user to easily explore the network, using all the features the interface has to offer. The pathway layout used by the visualizer is the Force Directed Layout (FDL) [44] with adaptations to support fixed nodes. This was coupled with the possibility to fix/unfix nodes, allowing the user to fix a node to the specific position it is in, or drag it to a desired new position; unfixing a node will remove the position information of the node, making it susceptible to the FDL algorithm to adjust its position according to its surroundings. It is also possible to unfix and fix nodes by type, allowing a user to fix/unfix all reaction or metabolite nodes at the same time.

BioVisualizer is also able to customize loaded layouts, by clicking in the right mouse button over nodes on network view, where several procedures are available, namely:

1. Changing the node type. It allows the user to mark a metabolite node as a “currency metabolite,” a special type of node. Often, highly connected metabolite nodes, the so-called

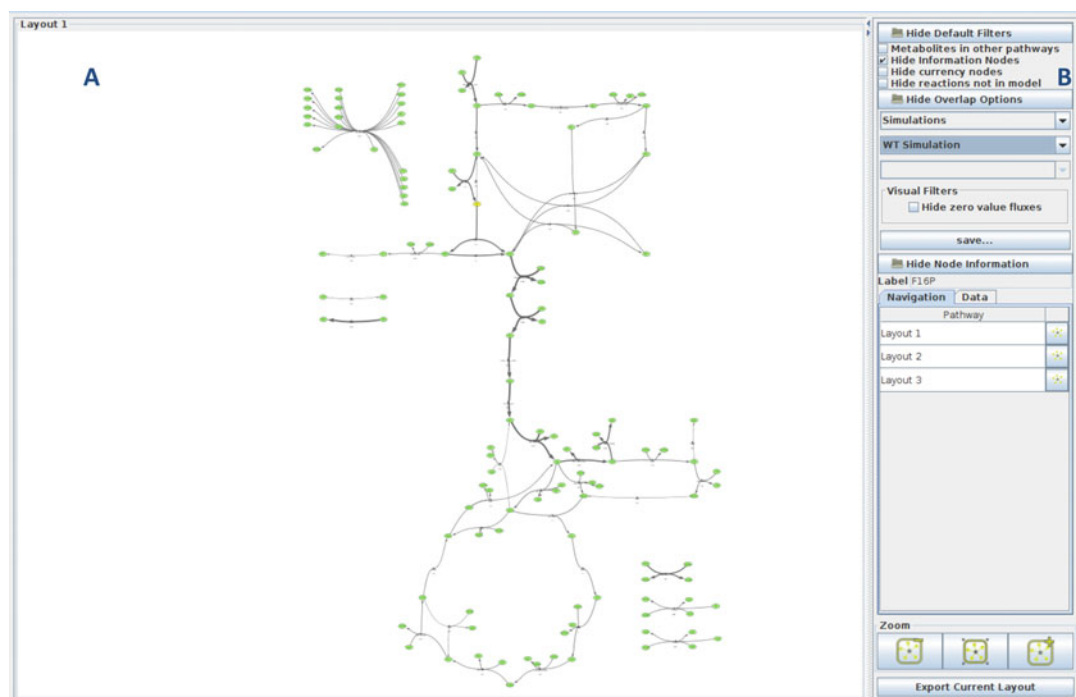


Fig. 7 BioVisualizer view. The view is divided into two different areas. The network viewer (a) where the users can access the nodes of the network and apply some functionality by clicking on the right button of the mouse. The side panel (b) with functionalities like node/edge filters, overlap selection, zooming options, navigation, and exporting the layout in several formats

“currency metabolites,” like energy-carrying molecules (ATP, NAD(H), NADP(H), etc.) can be marked to clearly identify this type of node and possibly hide them to unclutter the pathway visualization.

2. Replicating a node. Highly connected nodes can be replicated to reduce the number of overlapping connections in the layout.
3. Highlight replicated nodes. Replicated nodes are highlighted when selected.
4. Merging equivalent nodes. Replicated nodes can be merged (one to one).

All these features, when combined with the import and export capabilities, allow the user to create and edit their own layouts, being able to export them for later use in this or other compatible software tools.

Filtering and overlaying capabilities are also provided. It is possible to filter the network, by hiding parts of it, based in the node type (e.g., hide all currency metabolites) or by reaction identifier.

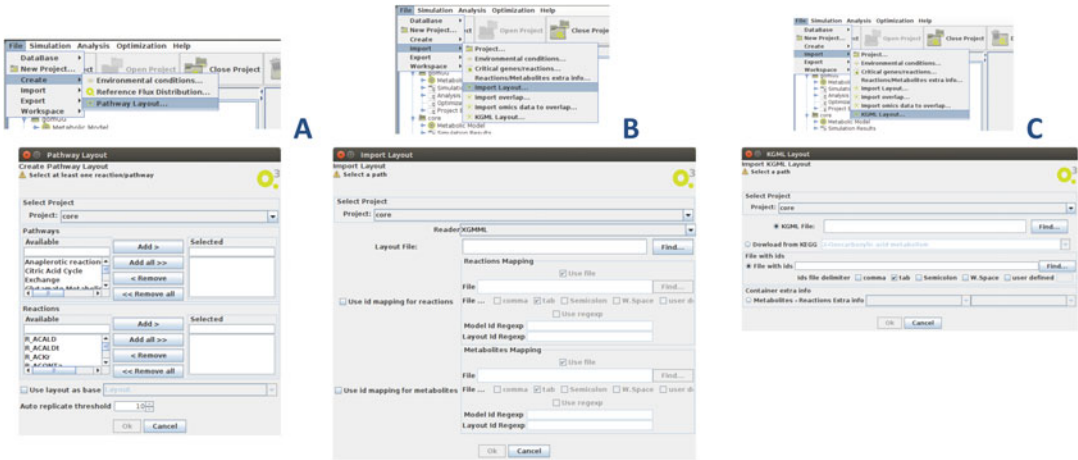


Fig. 8 Three different ways to create a layout for a model in OptFlux: creating a layout using model information (a), loading a layout file (b) or importing a KEGG layout (c)

It is also possible to overlay information over the network. The visualizer allows altering the visual aspect of the network components, supporting the change of the direction, thickness, and colors of the edges, while for nodes it is possible to change the color and shape. This feature allows, for instance, overlaying flux distributions in the metabolic pathway layouts.

3.2.2 Loading/Creating Layouts

The Biovisualizer plugin supports the creation of layouts using information existent in a model or the import of layouts in different formats. There are three different operations to import layouts in OptFlux (Fig. 8): the “Pathway Layout” operation in *menu:File/Create/Pathway Layout*, the “Import Layout” operation in *menu:File/Import/Import Layout*, and the “KGML Layouts” in *menu:File/Import/KGML Layout*.

Using the “Pathway Layout” operation (Fig. 8a) the user can build a layout using reaction stoichiometry information present in the model. This can be done by following two possible strategies: (1) choosing a list of reactions to be represented in the layout, or (2) in the case the model provides pathway information, builds the layout directly with the reactions from a selected set of pathways.

In the “Import Layout” operation (Fig. 8b) it is possible to select one of the following formats:

1. eXtensible Graph Markup and Modeling Language (XGMML): a format based on the Graph Modeling Language (GML), used for graph description using XML tags to describe nodes and edges. It is used in different tools such as Cytoscape [42].
2. CellDesigner SBML (CD-SBML): graphical notation system proposed by Funahashi and coworkers [40, 41], where layouts are stored using a specific extension of SBML.

3. Systems Biology Graphical Notation (SBGN): graphical notation system to represent networks of biochemical interactions in a standard and unambiguous way [45].
4. COBRA version 1 layouts: set of maps specifically created for the COBRA Toolbox. Several maps were constructed using this format for many of the models hosted in the BiGG-v1⁶ [46] knowledge-base. These layouts have the advantage of being valid for use with multiple models containing similar pathways, since they provide a correct mapping of the identifiers between the pathway layout and BiGG models.
5. Escher⁷: Escher is a web-based tool for building, viewing, and sharing visualizations of biological pathways [47]. It uses a format based on the JavaScript Object Notation (JSON) to represent the layouts. Those files had been created for some of the models hosted in the BiGG-v2⁸ [48].

The “KGML Layout” operation (Fig. 8c) allows the user to import a KEGG pathway⁹ to OptFlux. To do so, the user can choose a KGML file that can be downloaded from the KEGG database or choose directly the pathway name to be retrieved from KEGG. This interface also provides the tools to map a metabolic model with a pathway layout. It is possible to load a 2-column file providing the mapping between the metabolic identifiers in the model and KEGG identifiers. If the model loaded in OptFlux has already that information, then it is possible to map it automatically.

3.3 Simulation Using CBM

3.3.1 Simulation Methods

On the previous chapter, it was presented how a system of homogeneous linear equations can be derived, assuming that there is no accumulation of metabolites, and thus all fluxes leading to their formation and degradation are mass-balanced [49, 50]. This system is representative of an entire metabolic phenotypic space of a specific organism strain. Such systems are typically underdetermined since the number of reactions (variables of the system) normally exceeds the number of metabolites (constraints of the system), so, a plurality of solutions exists, expressed in an infinite number of possibilities of distribution of metabolic fluxes through available reactions. These possibilities are constrained by the stoichiometric matrix and the flux limits, forming a domain of stoichiometrically feasible behaviors.

The feasible domain of a constraint-based model can be conceptualized as the “metabolic genotype,” i.e., all reactions that can be catalyzed with enzymes codified by the genes in the genome of

⁶ <http://bigg1.ucsd.edu>

⁷ <https://escher.github.io/>

⁸ <http://bigg.ucsd.edu/>

⁹ <http://www.genome.jp/kegg/pathway.html>

the modeled strain. A particular point in this feasible domain is a specific flux distribution that can be conceptualized as a “metabolic phenotype” since this flux distribution is a characteristic of how the strain responds to an input, as a medium or stress. Therefore, constraint-based models are able to identify how a strain adjusts its metabolic distribution to an environmental perturbation, such as an alternative carbon source, or even to genetic modifications, such as gene deletions.

A particular solution for the distribution of metabolic fluxes may be found using constraint-based optimization by stating an objective and seeking its maximum/minimum value, within the stoichiometrically defined domain. Such objective encodes a biological rationale, mimicking how a cell chooses the distribution of metabolic fluxes on the domain of feasible behavior. However, the mechanisms by which metabolic flux distributions are chosen by the cell are a complex interplay of enzymes, genetic regulatory, and signaling events and not all of those events are known in detail. Nonetheless, in terms of the evolutionary selection process it is expected that, for a given environment, a given cell chooses to express and regulate a specific set of metabolic enzymes, which act in concert to produce an “optimal metabolic flux distribution,” which may be thought of as the “metabolic phenotype” of that strain under those conditions. Such “optimal flux distribution” depends on the objective used by the cell, such that in past years several phenotype prediction methods have been proposed on the top of the same steady-state assumptions, but differing on the objective function and/or additional constraints.

Flux Balance Analysis (FBA) was the first optimization-based phenotype prediction method developed for predicting the phenotypes using CBM models of metabolism. Briefly, FBA finds a distribution of fluxes that meets a certain objective formulated with Linear Programming (LP). The most common objective function used with FBA is the maximization of the specific growth rate, encoded in a biomass pseudo-equation, assuming that the metabolic phenotype of a wild-type strain is defined by a tendency to optimize its growth rate. In a practical way, FBA joins the under-determined system of linear equations with the objective of maximizing the growth reaction flux. The growth reaction (explained in detail above) describes the consumption of biosynthetic precursors and energy requirements for synthesizing a specific amount of cellular material [49, 51–53].

One handicap of the flux distributions obtained with FBA is that they are not unique. The values of the reaction fluxes in the metabolic network can vary for the same optimal value of the objective function (i.e., multiple flux distributions can lead to the same maximum growth rate).

Parsimonious enzyme usage FBA (pFBA) [54, 55] is a two-step LP optimization problem, which tries to reach the simplest flux

distribution under the maximum growth rate. In the first step, a normal FBA is issued to compute the optimal growth rate, while on the second step a minimization of the sum of the absolute values of all the reaction fluxes is performed, while the optimal growth rate calculated in the first level is maintained as a constraint. pFBA removes some artifacts in the flux distribution that do not contribute for the maximum growth and/or are physiologically unsound, such as fluxes related to futile cycles in the network. The assumption of this method is based on the efficiency of metabolic networks, which expects the cell to perform a certain task with the minimal amount of resources (amount of proteins and genes), thereby predicting the most resource-wise efficient flux distribution for the maximum growth rate.

While the maximum growth rate assumption has been well accepted by the scientific community as a good approximation to mimic the phenotypic behavior of a wild-type strain, the same cannot be said when the system is subjected to a perturbation, such as a gene deletion or a stress condition in the environment. To more accurately predict how a metabolic network reacts to a genomic perturbation, distinct phenotype prediction methods have been proposed.

The Minimization Of Metabolic Adjustment (MOMA) [56] was developed in 2002 and it was the first proposed formulation with the purpose of simulating the effect of genetic perturbations in a metabolic network. This methodology assumes that a mutant cell (i.e., a cell with a perturbation in its genome like a gene deletion) will try to minimize the adjustments of the flux values in comparison with its behavior on the wild-type strain. MOMA is formulated as a Quadratic Programming problem, assuming its objective function as the minimization of the Euclidean distance between the mutant set of fluxes and the reference wild-type fluxes. The growth predictions of gene knockouts simulated with MOMA are more conservative than the results obtained with FBA and it has been shown that MOMA can predict more accurately gene essentiality in some cases [56, 57].

Shlomi et al. pursued the same concept of minimal metabolic adjustment, but in a different perspective [58]. Instead of minimizing the flux differences between the mutant and the wild-type strains, the methodology entitled Regulatory On/Off Minimization of metabolic flux changes after genetic perturbations (ROOM) minimizes the number of reactions that are activated or deactivated in a mutant in comparison with a reference flux distribution. This approach requires the introduction of binary variables in the objective function, thus converting the problem into a Mixed-Integer Linear Programming problem, increasing its complexity. The assumption behind ROOM is that, when faced with a set of knockouts, a cell will adjust its internal fluxes by making the minimum amount of regulatory changes, i.e., the magnitude of the fluxes can change, but the set of active enzymes should be similar to the wild-

type organism. The predictions obtained for ROOM were closer to FBA than MOMA and revealed that MOMA is better at estimating transient metabolic adaptations, while FBA and ROOM can better predict the phenotype of an evolved knockout mutant [58].

In addition to the regular MOMA formulation, two additional variations are available in the literature: linear MOMA (LMOMA) [59] and PSEUDO [60]. One common issue usually encountered in the flux distributions computed with MOMA is that it favors a large number of small flux changes in detriment of a few large changes in the metabolic network. This is caused by the quadratic formulation used to calculate the flux distance in MOMA and can be solved by using LMOMA, which uses the Manhattan distance between the reference and perturbed networks to find the flux phenotype of the knockout mutant. Another issue that might arise from using MOMA/LMOMA is the importance given to a reference flux distribution. Usually, the reference set of fluxes is calculated using FBA or pFBA [61–63] and any error in this flux distribution will be propagated to all the predictions. The methodology developed in PSEUDO can tackle this issue by not using a single flux distribution as a reference but a region of the flux space delimited by a minimum threshold imposed on the biomass yield [60]. The authors of this methodology reported some improvements over MOMA and FBA flux predictions [60].

Another issue encountered in the formulation of MOMA and LMOMA was the dependence of the mutant phenotype on the scale of the stoichiometry of the metabolic reactions [64]. By using different stoichiometric representations of a metabolic network that are biochemically equivalent, Brochado et al. showed that the simulation outcome of MOMA/LMOMA was sensitive to the stoichiometric representation chosen for the network [64]. Since biochemically equivalent networks should produce the same results, the authors proposed a new methodology entitled Minimization of Metabolites Balance (MiMBI). The formulation underlying MiMBI solves the stoichiometry dependence of other algorithms by using the metabolite turnovers as the variables in the objective function. Instead of minimizing the changes in the fluxes in comparison with a reference network, MiMBI minimizes the changes in the turnovers of all metabolites in the network. As a consequence, MiMBI provides more robust results, which are not dependent on the numerical stoichiometric representation chosen to describe a metabolic network.

3.3.2 Flux Variability Analysis

A key issue that may arise with the use of constraint-based phenotype prediction methods is the existence of alternative optimal solutions within the same maximal objective (e.g., growth rate), which can be achieved through different flux distributions. Flux Variability Analysis (FVA) is an efficient LP-based strategy used to calculate the full range of values of each flux that can be present, to achieve optimal or suboptimal objective states. This technique is

one of the most used to investigate the limits of the feasible domain of a constraint-based model.

FVA can be a useful tool, not only for determining the ranges of the fluxes that could be achieved using a specific CBM model, but also to determine how a flux can vary in an optimal or suboptimal solution determined by a phenotype prediction method. In this latter case, the value of the objective function used by that specific phenotype prediction method is computed in a first step, and afterward its value is set as restriction, followed by two new optimizations to calculate the maximum and the minimum flux values for a specific reaction given those constraints.

Through the use of FVA it is possible to determine the domain of all the possible solutions (usually represented by an n -dimensional polyhedron called the flux cone) where each dimension refers to a flux) and to analyze how two fluxes influence each other.

To do so, it is necessary to select a target flux and a pivot flux. The operation begins by splitting the range of all possible values of the pivot flux in multiple steps. Afterward, for each step, the value of the pivot flux is fixed and the maximum and minimum values for the target flux are computed. These values can be plotted into a two-dimensional surface that can be thought of as the projection of the flux cone into the dimensions of the pivot and the target fluxes.

This technique can be very useful to understand how an input/output of the system influences the biomass flux (phase plane analysis), or to analyze the robustness of a production strain (production envelope analysis).

To help illustrate the first point (phase plane analysis), Fig. 9 is put in place, where the relationship between a pivot and a target

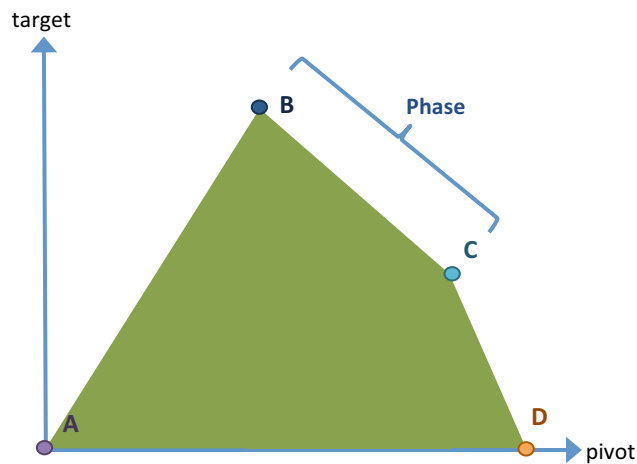


Fig. 9 Surface plot. The range of values of a pivot flux is split and the minimum and maximum values of the target flux are calculated generating in this way a polygon. In such polygon it is easy to identify the minimum and the maximum that the pivot flux could achieve (a and d) the maximum value of the target flux (b) and where there are changes in the phase (c)

flux are represented. In a closer inspection of the plot, it is possible to identify four different points—A, B, C, and D. Points A and D represent the minimum and maximum values of the pivot flux, point B represents the maximum achievable value of the target flux for the corresponding and optimal pivot flux value (b'). Finally, point C represents a phase change and c' the corresponding pivot flux threshold value (prompting the phase change). Phase changes can be detected by analyzing the first derivatives of the points of the plot (i.e., the slope of the line) and correspond to the reduced cost of the pivot flux. The reduced costs are associated with the LP variables and represent the change in the objective function of the LP problem, promoted by a small change in the pivot flux. A positive reduced cost translates into an increase in the value of the objective function, while a negative one represents a decrease. These values are important, since they allow us to identify phenotypic phase changes. While no change is observed in the reduced cost, it can be assumed that the system does not need to change its behavior to comply with its restrictions. Alternatively, when a change in the reduced cost is observed, it can be assumed that a phase limit has been reached where the system needs to change its behavior to keep complying with its set of restrictions. These system behavior changes are translated to phenotypic changes. As an example, a phase change can be detected when simulating *E. coli* with oxygen limitations. By gradually reducing the oxygen uptake, it is possible to detect a phase where *E. coli* starts producing acetate and another where *E. coli* produces ethanol, formate, and acetate (via mixed-acid fermentation).

Another informative analysis that can be achieved via FVA is the robustness analysis of a production strain (production envelope analysis). In this case, FVA is also used to generate a plot similar to the one previously explained, usually called the production envelope. The production envelope is used to analyze the phenotype of a strain that is already producing a target product. The procedure to calculate this plot is the same as before, but in this case the pivot is usually the biomass flux and the target is the product flux. In a typical production envelope, there are three important points to keep in mind; A—the theoretical maximum production rate of the target metabolite, B—the maximum production rate of the target when the biomass is in its maximum value, and C—the threshold point of the pivot flux, where the minimum value of the target flux reaches 0. As exemplified in Fig. 10, analyzing the position of these three points in the plot allows the characterization of the solution robustness as:

1. Non-robust—when points B and C are aligned in the maximum of biomass. This means that there is a range of possible values for the target product, including 0 (no production).

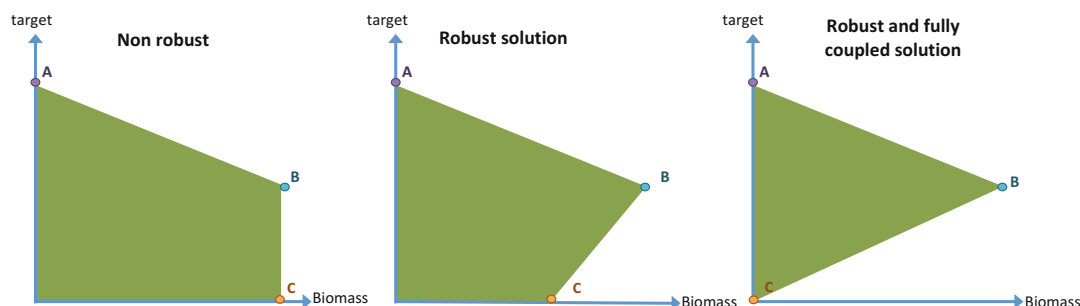


Fig. 10 Surface plots used to characterize productive phenotypes

2. Robust solution—when point C is not aligned with B, meaning that there is a range of biomass values between C and B, where there is a guaranteed minimum production of the target compound.
3. Robust and fully (biomass) coupled solution—when point C is overlapped with 0 in the biomass axis and not aligned with point B, meaning that the production of the target compound is guaranteed for the whole range of possible pivot (biomass) values.

The analysis of the results can reveal the best tradeoff between biomass formation and target product secretion.

In conclusion, FVA can be used to study the entire range of achievable metabolic functions, as well as the redundancy in optimal phenotypes [65].

3.3.3 OptFlux Operations

Environmental Conditions

When a project is loaded into OptFlux, the metabolic model is created with default lower and upper flux bounds that are defined in source information (e.g., SBML file). Those values can be analyzed in the reactions data type within the metabolic model. To override these default values, an environmental condition (EC) can be created. The EC is typically used to define the medium where the organism is growing, by defining lower/upper bounds in the drains available in the system. An EC could be also used to block reactions, change reversibilities or impose maximum capacities for some reactions. There are two different ways to create an EC in OptFlux (Fig. 11): importing a pre-existent environmental condition from file or creating it from scratch.

Importing Environmental conditions using text files can be performed using the operation on *menu:File/Import/Environmental conditions* (Fig. 11a). In this operation the user needs to select the project where the EC will be imported using a tabular data file. A tabular data file is a plain text file, where each line is a data record consisting of one or more fields separated by a delimiter character such as commas or tabs. In EC tabular files, each record is expected to have three fields, the first one is the reaction identifier (id), the

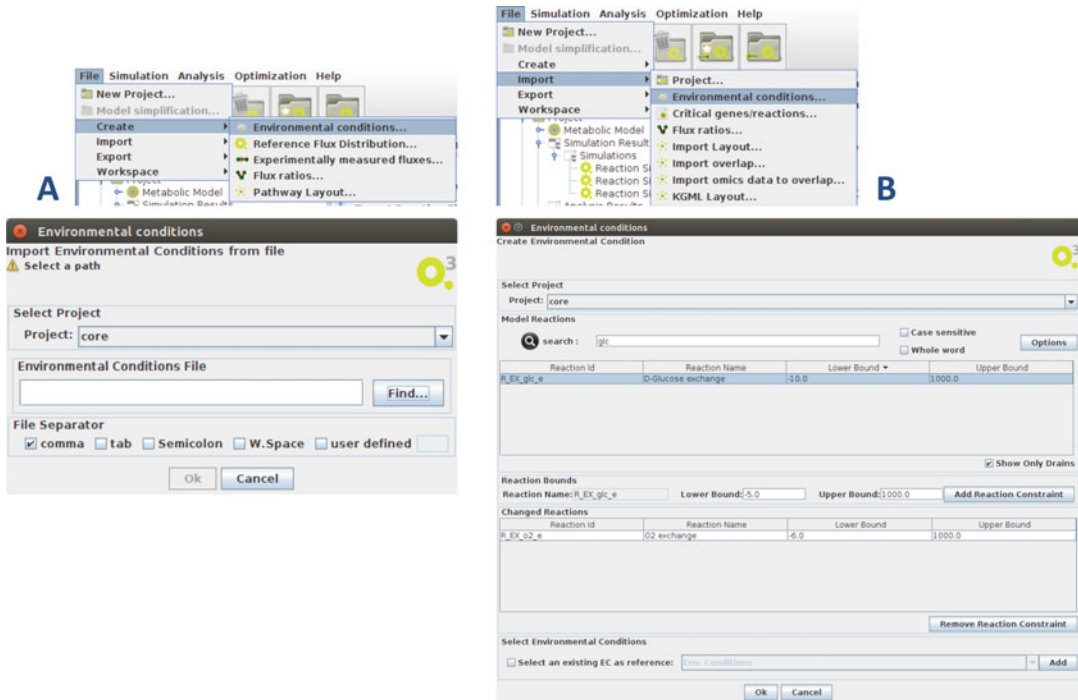


Fig. 11 Two different ways to create an environmental condition in OptFlux. Loading environmental conditions from a file (a), or creating an environmental condition by setting reaction bounds individually (b)

second the lower bound and the third the upper bound. The file separator can also be defined in the operation.

To create an EC from scratch the user must access the operation via *menu:File/create/Environmental conditions* (Fig. 11b). The operation allows the user to select reactions, one by one and define their limits. By default the checkbox for “Show Only Drains” is selected making only the drains visible, assuming that the user only has to define a growth medium. However, the user is able to define the bounds of any reaction in the model by unchecking that box. The upper part of the operation allows the selection of the reactions (easing its search), while the bottom part allows adding these constraints to the EC, specifying the lower and upper bounds. The user can also add all reaction bounds contained in a previously defined EC.

Phenotype Simulation

Optflux has implemented five operations to perform phenotype simulations each one with different assumptions to perform genetic modifications:

1. Wild-Type Simulations accessible in the *menu:simulation/wild type* (Fig. 12a).
2. Reaction Knockout simulations accessible in the *menu:simulation/knockouts/reaction* (Fig. 12b).

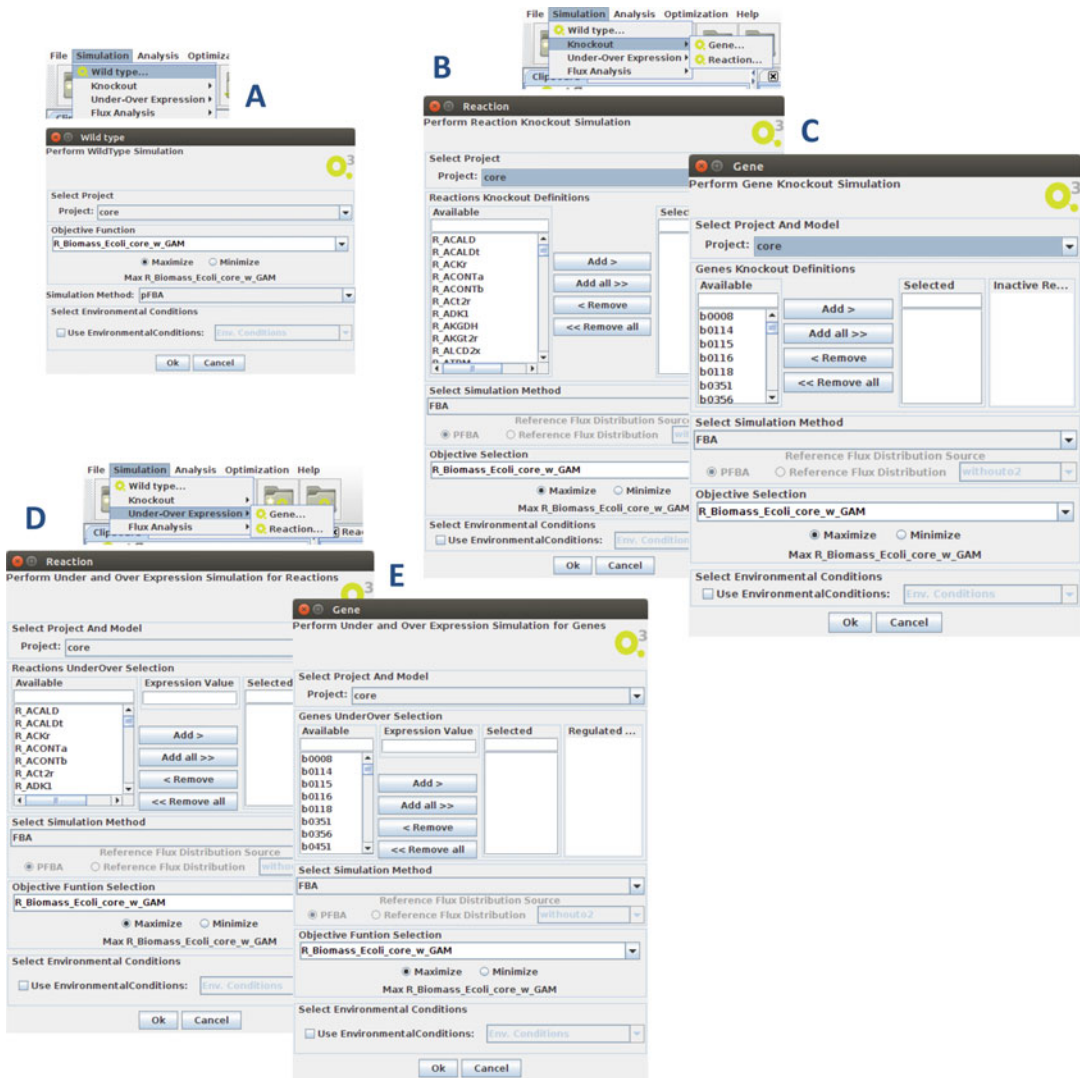


Fig. 12 Five different types of phenotype simulations to simulate in Optflux. Wild type (a), reaction knockouts (b), gene knockouts (c), reaction under/over-expression (d), and gene under/over-expression (e). In all the operations, the user is able to also select a simulation method, the biomass reaction, and the environmental condition

3. Reaction Under/Over-expression simulations accessible in the *menu:simulation/Under-Over Expression/reaction* (Fig. 12d).
4. Gene Knockout simulations accessible in the *menu:simulation/knockouts/gene* (Fig. 12c).
5. Genes Over/Under-expression simulations accessible in the *menu:simulation/Under-Over Expression/gene* (Fig. 12e).

For all of these operations it is possible to select the OptFlux project that will be used, an environmental condition (if it is

necessary to override the default bounds existing in the model), the phenotype simulation method and some parameters for the objective function (method specific).

In terms of wild-type simulations, two simulation methods are available, FBA and pFBA; for the objective function it is required to specify a target reaction for optimization (by default the biomass equation is selected) and an objective function sense, i.e., maximizing or minimizing the selected flux (maximization is selected by default).

For other types of phenotype simulations (such as reaction/gene knockout/over-under expression—*see* **Note 1**), in addition to the FBA and pFBA methods, the user can also select MOMA, LMOMA, ROOM, and MiMBI. These additional methods share the common characteristic of requiring a reference flux distribution to run, and the way this reference is selected depends on the user preference. There are two possibilities; (1) the default option, where OptFlux performs a pFBA simulation and uses the resulting flux distribution as the reference; and, (2) select a previously computed phenotype prediction as the reference flux distribution.

The major difference between these operations is the type of genetic changes that will be simulated. When performing the wild-type simulation operation, no genetic changes are imposed upon the model, while for all the other operations perturbations at the reaction or at the gene level are allowed. For knockouts, two operations are available, allowing the selection of either a set of reactions or a set of genes to knockout. Finally, for over/under expression operations, a pFBA flux distribution is assumed as the reference and a set of reactions or genes is selected. For each entity (reaction or gene) the user is asked to define an expression value (over/under) relative to that of the reference flux distribution. For additional information consult the sections above and **Note 1**.

The result of any phenotype simulation operation is an object created in the clipboard, placed within the Simulation Results list, under the Simulations data type. The phenotype simulation result data type collects multiple information regarding the performed operation, which is displayed in several tabs (Fig. 13):

1. Simulation solution: In this tab, OptFlux presents some of the selected inputs for the simulation, such as the method name and the selected ECs, as well as the value for the objective function obtained with the selected phenotype prediction method, the biomass value, and the net conversion of the system. The net conversion represents the net rates of the metabolites that are consumed and produced, providing an overall perspective of the predicted consumptions and secretions for the current simulation.

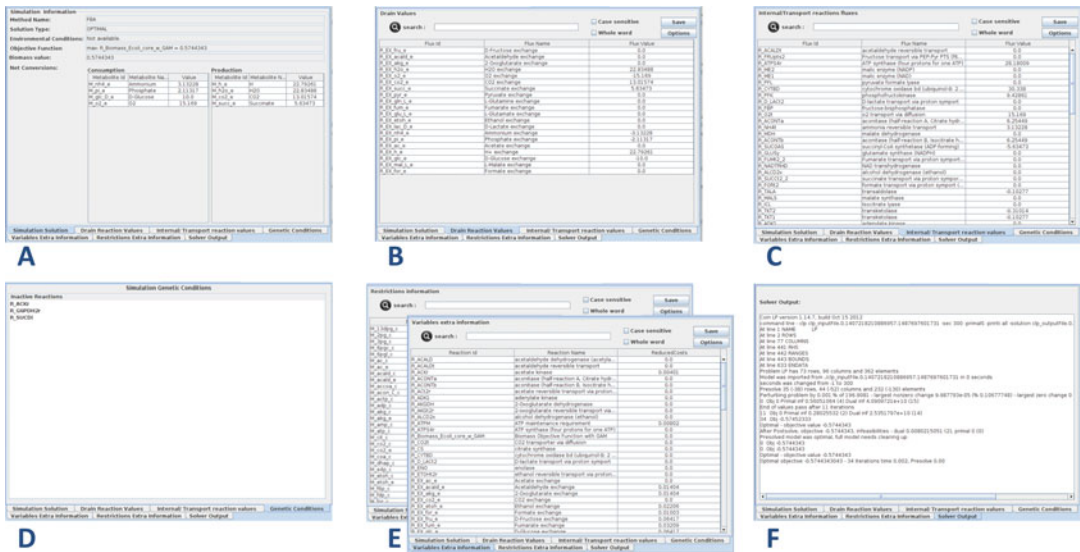


Fig. 13 Results for phenotype simulation in an example where three reactions were knocked out (R_ACKr, R_SUCDi, and R_G6PDH2r) using an *E. coli* core model [51]. (a) Phenotype simulation method and the net conversion showing the inputs and outputs of the system; (b) Drain reactions' fluxes; (c) Internal reactions' fluxes; (d) Genetic alterations; (e) Extra information regarding the variables and restrictions of the problem like shadow prices and reduced costs; (f) Raw solver output

2. The values of the fluxes are split between two views: one displays the drain reaction values and the other displays the Internal/Transport reaction values. In these views, the information is presented in a tabular format, allowing the user to search for specific information (via an intelligent search bar), hide the rows containing zero values, as well as exporting the information to a table format file.
3. Genetic conditions: allows inspecting the genetic conditions (including knockouts, over-under expression) applied. In the wild-type operations, no genetic conditions are shown.
4. Variables and restrictions extra information: extra information associated with the variables and the restrictions, such as shadow prices and reduced costs.
5. Solver output: textual output of the solver used in the constraint-based optimization problem.

Flux Variability Analysis

As previously mentioned to investigate the limits of the feasible domain of a CBM model or the alternative optimal solutions of the phenotype prediction methods, an FVA operation can be performed.

OptFlux provides two different operations concerning FVA analysis (Fig. 14):

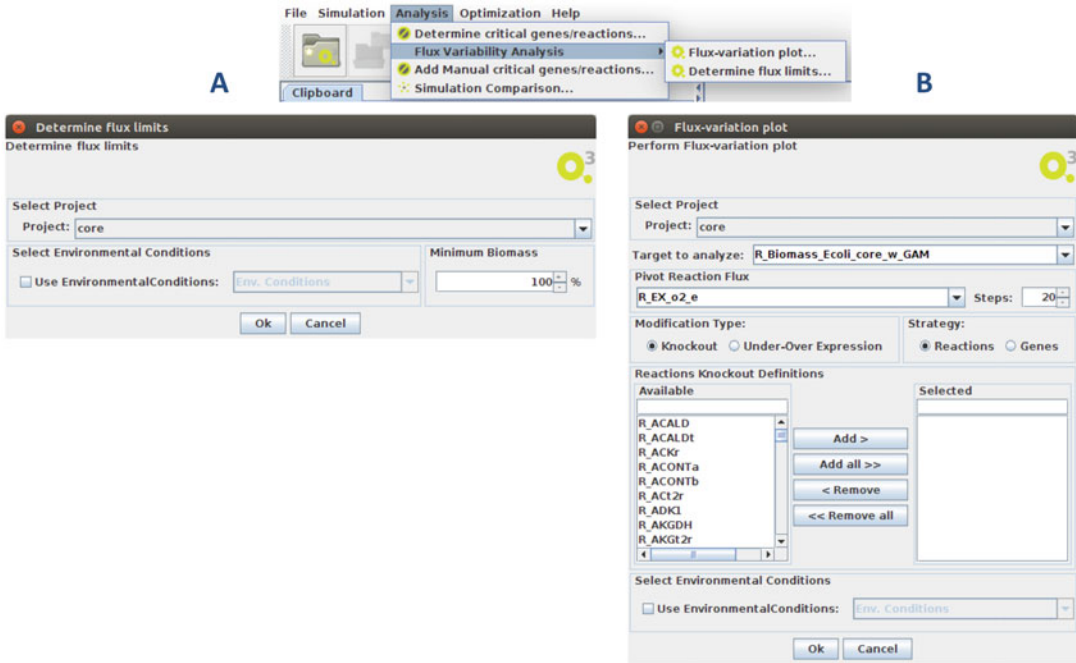


Fig. 14 OptFlux GUIs for accessing the two different ways of analyzing flux variability: determine all fluxes' ranges (a); determine a surface selecting a pivot and a target flux (b)

1. Determine flux limits in *menu:Analysis/Flux Variability Analysis/Determine flux limits* (Fig. 14).
2. Flux-variation plot in the *menu:Analysis/Flux Variability Analysis/Flux-variation plot* (Fig. 14).

The “Determine flux limits” operation is used to determine the domain of all fluxes present in the CBM model, by restricting the system to a defined level of the biomass flux. Such a level is a percentage of the wild-type value (maximum biomass value). In the configuration of the operation, the user selects the project, the biomass level used as a restriction, and can also select an EC. The result of the operation is a new object in the clipboard, placed within the “Analysis Results” list, under the “Flux Limits” data type. The result contains the minimum and maximum values possible for all the fluxes (Fig. 15).

The “Flux variation plot” operation is used to build a 2-dimentional projection of the solution space. To perform this operation the user must select a pivot and a target flux, and (optionally) can also impose a genetic perturbation (gene/reaction knockout or gene/reaction under/over expression). Using this operation the user is able to compute a phase plane analysis or an envelope production analysis depending on the chosen fluxes to pivot and the target (see above for more details). The result of the operation is

Flux Limits 100.0%

search :

☐ Case sensitive

☐ Whole word

Save Options

Reaction Id	Reaction Name	MIN	MAX
R_ACALDt	acetaldehyde reversible tra...	-0.00025	0.0
R_FRUpts2	Fructose transport via PEP:...	0.0	0.0
R_ATPS4r	ATP synthase (four protons...	45.51041	45.5159
R_EX_acald_e	Acetaldehyde exchange	0.0	0.00025
R_ME2	malic enzyme (NADP)	0.0	0.00229
R_EX_akg_e	2-Oxoglutarate exchange	0.0	0.00014
R_ME1	malic enzyme (NAD)	0.0	0.00172
R_PFL	pyruvate formate lyase	0.0	0.00114
R_CYTBD	cytochrome oxidase bd (ubi...	43.59822	43.59975
R_PFK	phosphofructokinase	7.47607	7.47916
R_D_LAct2	D-lactate transport via prot...	-0.00021	0.0
R_FBP	fructose-bisphosphatase	0.0	0.00172
R_O2t	o2 transport via diffusion	21.79911	21.79987
R_ACONta	aconitase (half-reaction A, ...	6.00606	6.0088
R_NH4t	ammonia reversible transport	4.76527	4.7654
R_MDH	malate dehydrogenase	5.06273	5.07006
R_ACONtb	aconitase (half-reaction B, l...	6.00606	6.0088
R_SUCOAS	succinyl-CoA synthetase (A...	-5.06594	-5.05633
R_GLUsy	glutamate synthase (NADPH)	0.0	0.00172
R_FUMt2_2	Fumarate transport via pro...	0.0	0.0
R_NADTRHD	NAD transhydrogenase	0.0	0.00686
R_EX_fum_e	Fumarate exchange	0.0	0.0
R_EX_glu_L_e	L-Glutamate exchange	0.0	0.00013
R_ALCD2x	alcohol dehydrogenase (et...	-0.00022	0.0
R_EX_lac_D_e	D-Lactate exchange	0.0	0.00021
R_SUCct2_2	succinate transport via pro...	0.0	0.00229
R_EX_pi_e	Phosphate exchange	-3.2149	-3.21486
R_EX_h_e	H ⁺ exchange	17.53069	17.53183

FL Solution

Fig. 15 Results of performing a “Determine flux limits” operation using E. coli core model using 100% of wild-type simulation as biomass restriction. On the result the user can inspect the minimum and maximum limits of all fluxes

a new object in the clipboard, placed within the “Analysis Results” list, under the “FVA Simulations” data type. The information of the new data type is displayed in several tabs:

1. “FVA Chart,” where the user can visualize a plot that represents the projection of the two selected fluxes (pivot and target).
2. “FVA values,” where the user can see the maximum and minimum values of the target flux restricted to a value of pivot flux, as well as the reduced costs of the pivot variable for the minimum and maximum simulations.
3. “FVA information,” here the user can inspect the pre-configured parameters used to perform the operation.

In Figs. 16 and 17, an example of a phase plane analysis and an example of an envelope production analysis are represented, respectively.

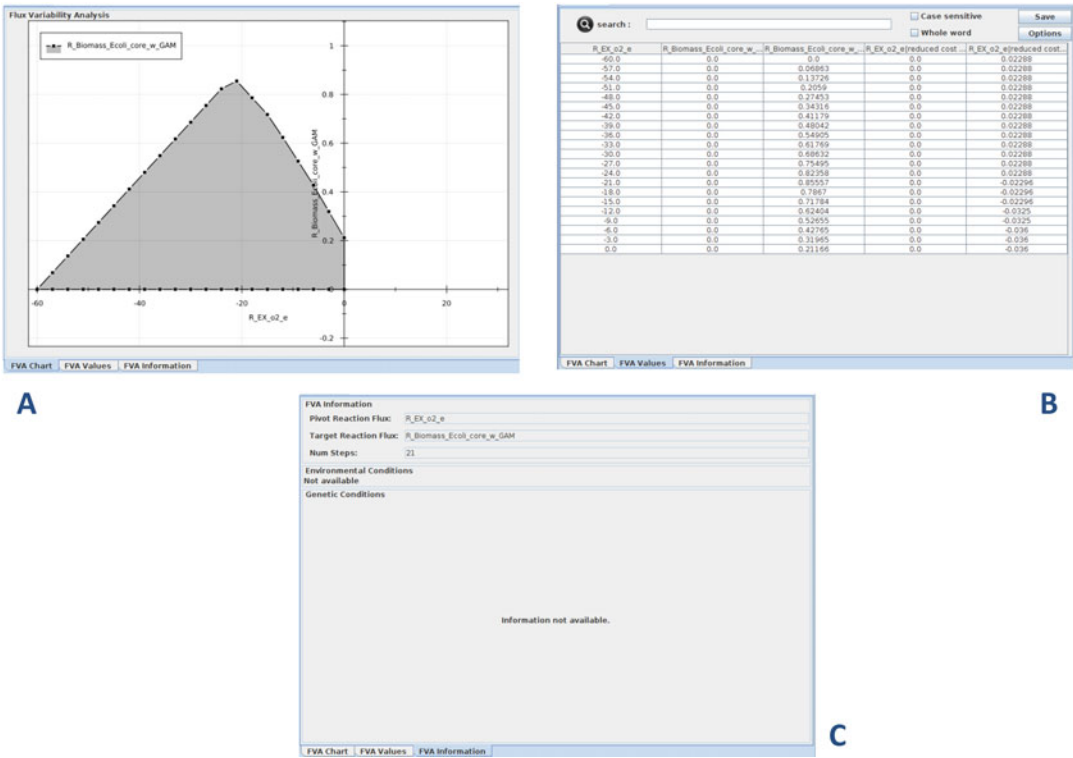


Fig. 16 Results of performing a flux-variation plot operation using E. coli core model, with the oxygen drain as pivot flux and the biomass as target flux. In the FVA chart view (**a**), the user is able to see the shape of the phase plan generated by the operation. In the FVA values view (**b**), the user is able to see all the steps calculated for the pivot flux, the minimum and maximum values of the target and also the reduced costs for the pivot target. In this example, by analyzing the reduced costs (last column) it is possible to detect four different phases: Phase 1 [−60, −21] mmol/gDWh of the oxygen input flux, where oxygen is decreasingly harmful for growth; Phase 2 [−21, −12] mmol/gDWh of the oxygen input flux, where E. coli produces Acetate as a co-product; Phase 3 [−12, −9] mmol/gDWh of the oxygen input flux, where E. coli produces Acetate and Formate as by-products and Phase 4 [−6, 0] mmol/gDWh of the oxygen input flux, where E. coli produces Acetate, Formate and Ethanol as by-products of the mixed-acid fermentation process. In the FVA information view (**c**), the user is able to see a summary of the configured parameters to reach the result.

Critical Genes/Reactions

A reaction or gene is considered critical (or essential) when its knockout leads to an unviable phenotype strain. Critical information is usually used not only to validate the models, but also to restrict the search when the objective is to construct a mutant strain to overproduce a compound of interest. In OptFlux, there are two operations that are able to build a critical reactions/genes set (Fig. 18):

1. *menu:Analysis/Determine Critical Genes/Reactions*—where OptFlux will perform a single knockout to genes or reactions assuming as critical all those where their knockout leads to a value of biomass less than 5% of the referenced value calculated

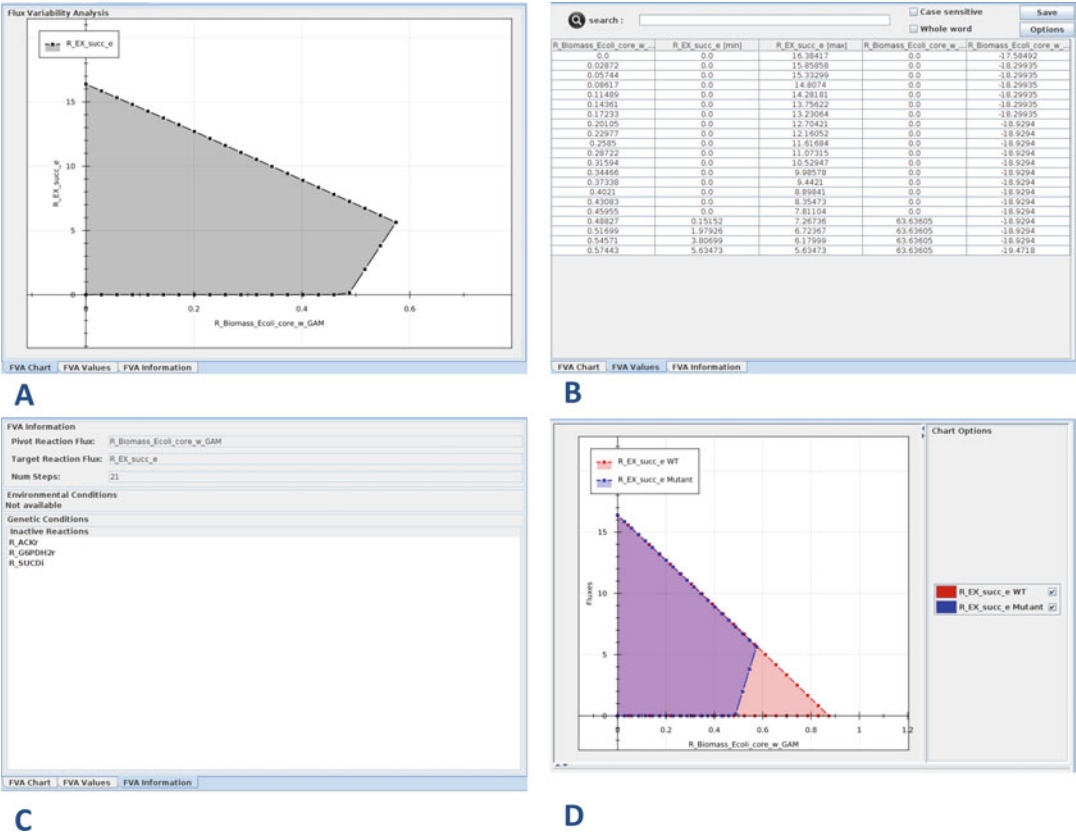


Fig. 17 Production Envelope for succinate using E. coli core model with reactions R_ACKr (Acetate kinase), R_SUCDi (succinate dehydrogenase), and R_G6PDH2r (G6P-dehydrogenase) suppressed. Production envelope plot (a); values for the biomass steps, minimum/maximum values of succinate and reduced costs to biomass (b); summary of the parameters used in the FVA operation (c); view overlapping the envelope production of wild type and mutant (d)

using wild-type simulation. On this operation, the user can also select an environmental condition.

2. *menu:Analysis/add manual critical genes/reactions*—where the user has the freedom to create a specific set of critical information. This operation is useful to the users that know that a given reaction or gene is essential or when the user wants to exclude some entities from the potential genetic alteration targets in strain optimization processes (i.e., ATPm is not a critical reaction but should not be considered as target of strain optimization process).

The result of both the operations is a new data type in the clipboard containing all critical genes or reactions.

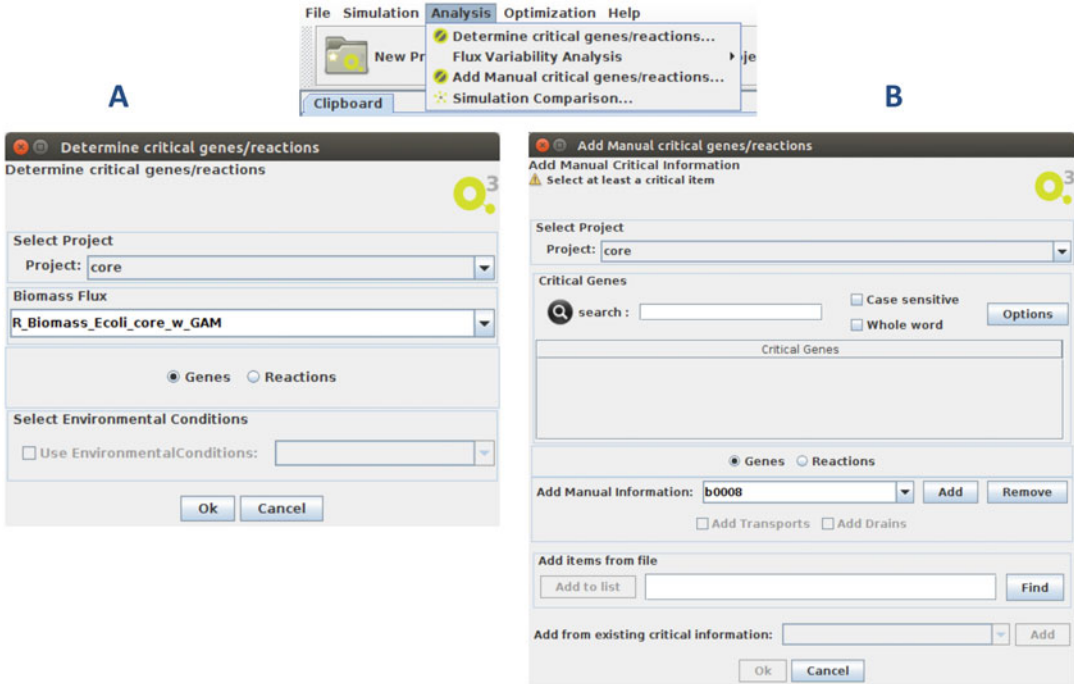


Fig. 18 Two different ways to create critical information on Optflux. (a) Determining by phenotype simulation approach, (b) Manual creation

3.4 Constraint-Based Strain Optimization Methods

While the in silico simulation of phenotypes can be of great help for analyzing the phenotype/genotype of a few rationally designed strains, it would take too long to manually discover combinations of genetic modifications that meet a certain metabolic engineering objective. In order to search for interesting genetic targets, several Computational Strain Optimization Methods (CSOM) have been developed to search among the large number of different strain engineering strategies for the ones that result in the desired phenotype [66].

CSOM links the abstraction of the constraint-based metabolic models and the assumptions of the constraint-based simulation methods to reveal insights into the best genetic design strategies to rationally address a ME objective.

In 2016, Maia and coworkers proposed a new taxonomy for the existent CSOMs, organizing them in three big clusters:

1. Bilevel mixed-integer programming methods;
2. Metaheuristics methods;
3. Elementary Mode Analysis based methods.

The bi-level mixed-integer programming methods are problems where two objective functions are simultaneously accounted for, usually the maximization of metabolite production and the maximum cellular growth. Such methods are formulated taking into account two distinct constraint-based optimization problems,

the inner problem and the outer problem. The inner problem is typically concerned with the cellular objective, while the outer problem is focused on the engineering goal, i.e., the search for the best combination of genetic alterations that favor the overproduction of a desired compound. This mathematical formulation is based on the strong duality property, which states that if the primal and the dual optimal solutions are bounded, then, at optimality, the gap between the objective function values must be zero [67]. This property allows the bi-level formulation to be transformed into a single-level MILP by setting the primal and dual objectives equal to one another and accumulating their respective constraints.

OptKnock [68] was the first method to use such methodology representing a breakthrough in the field and establishing the framework used by many of the developed CSOMs until today. OptKnock uses FBA as the inner problem to calculate the phenotype of a certain combination of knockouts by assuming maximum biomass formation. The result returned by OptKnock is the best combination of knockouts that maximize the overproduction of a desired compound, while taking into account a maximum number of knockouts and a minimum biomass formation rate.

One important drawback of the target discovery methods that search for a global optimum solution like OptKnock is that, as the number of allowed genetic modifications increases, the total searchable space of strain designs grows exponentially, which makes the computation time required to solve the problem impractical.

This severely limits the maximum number of genetic modifications that can be included in the strain designs computed with OptKnock and similar methods. One of the possibilities to solve this limitation is to use metaheuristic methods, such as evolutionary algorithms or other nature-inspired heuristics to find strain designs with desired phenotypes.

Heuristic methods are usually computationally less expensive approaches for a myriad of optimization problems. Although, due to their nature, they do not guarantee that the overall optimal solutions are found, they allow the definition of optimization frameworks with an enriched set of objective functions, fostering a clear separation of the strain optimization from the phenotype simulation layers, while allowing optimization over larger search spaces (e.g., a higher number of genetic modifications). In Fig. 19, a generic workflow for a typical metaheuristic CSOM is presented.

The first effort to move in this direction was OptGene [69], presented by Patil and coworkers, which appeared shortly after the publication of OptKnock. Inspired by the Darwinian natural evolution theory, OptGene formulates a bi-level decoupled approach, supported by the use of a genetic algorithm [70]. The idea is to encode solutions as individuals in an evolving population. Here, each solution is represented as binary variables encoding the metabolic genome or a set of integer values encoding reaction deletions.

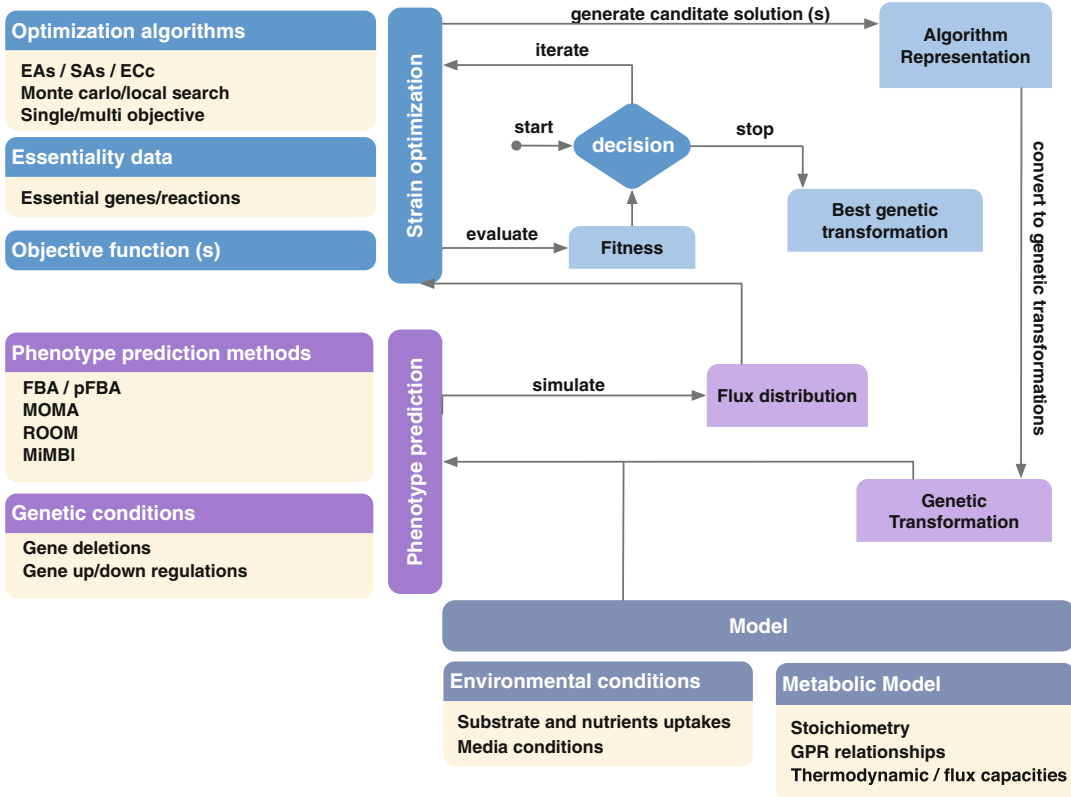


Fig. 19 Representation of the interactions between the meta-heuristic algorithms (blue), phenotype prediction (purple), and CBM (gray) on CSOM approaches

The algorithm starts by the random generation of an initial set of candidate solutions (the initial population), and each is decoded into a set of reaction deletions, which are translated into constraints. Each of these candidate solutions is simulated by means of one of the phenotype simulation methods described above, followed by the assignment of a fitness value by a user-defined objective function. Subsequently, the algorithm enters an iterative stage, starting with a selection step that chooses solutions as primary candidates for the reproduction in a stochastic way that depends on their assigned fitness (fitter individuals have a higher probability of generating offspring solutions). Finally, by combining these individuals via crossover or mutation operators, a new population is attained and re-evaluated. This cycle is repeated until a desired phenotype is achieved or another user-defined termination criterion is met (typically, a defined maximum number of generations or solutions evaluated).

The original implementation of OptGene was extended by the authors to support an optimized representation, which represents solutions as variable sized sets of deletions. Also, another nature inspired meta-heuristic—Simulated Annealing (SA)—was

proposed, which evolves a single solution instead of a population, by mimicking the process of thermal annealing usually found in metallurgy and affine areas [71]. Later, an approach based on multi-objective evolutionary algorithms, in particular the Strength Pareto Evolutionary Algorithm 2 (SPEA2), allowing for the optimization of multiple criteria, possibly including conflicting objectives, was developed and applied to various case studies [72]. Finally, an integrated framework that homogenized these three heuristics (EAs, SAs, and SPEA2) was proposed, providing the current optimization back-end available in OptFlux [73].

Although these methods still follow a bi-level design, in this case, the bioengineering and the biological optimization tasks are clearly decoupled and are performed independently. This decoupling of the outer and inner optimization problems results in some very powerful properties. For example, the inner phenotype evaluation method can easily be swapped to any phenotype simulation method. Another important advantage is the flexibility in the definition of the objective function in the outer problem, which is here not bounded by linearity. Nonlinear objective functions (even discontinuous) can easily be included, as is the case with the biomass-product coupled yield (BPCY), which resembles productivity [69], allowing the definition of more meaningful and powerful functions. The flexibility gained by the decoupling of the two layers also allows the easier switch of the optimization heuristic used to search for metabolic engineering strategies and allows the different optimization tasks to be addressed with a similar framework.

3.4.1 Strain Optimization Using Optflux

OptFlux allows performing strain optimization operations using metaheuristic methods. This operation can be accessed on *menu: optimization/evolutionary* (Fig. 20). To perform this operation a user needs to define the following inputs:

1. Select the project that will be used to perform the optimization.
2. Select the outer-layer optimization method that will perform the ME optimization process. In this selection, the user is able to choose among one of the three available optimization algorithm: SPEA2, EA or SA, as well as the genetic modifications targets (genes or reactions) and strategy (knockouts or over/under expressions).
3. Select the objective functions. The objective functions will encode the objectives of the desired ME process. There are several objective functions available in OptFlux, including biomass-product coupled yield (BPCY), product yield with minimum biomass (YIELD), maximization/minimization of a flux value or maximization/minimization of the number of genetic modifications. Each objective function requires its own configuration, selecting which fluxes are used in the

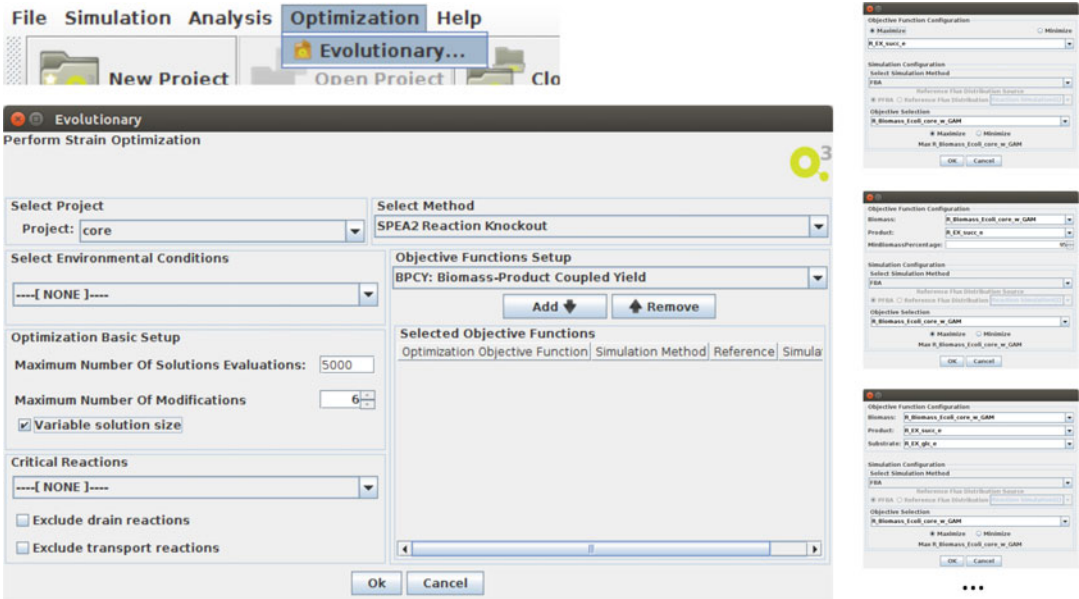


Fig. 20 Optflux Evolutionary strain optimization operation and some different objective functions setups

computation of the function value. A constraint-based strain simulation method must also be selected and associated with each objective function. This simulation configuration is similar to the simulation operation and allows selecting one of the implemented methods (FBA, pFBA, ROOM, MOMA, LMOMA, and MiMBI) and an objective function reaction, usually the biomass equation. This means that an optimization process can have multiple objective functions, each of them evaluated using different phenotype simulation configurations.

4. Select the environmental condition to be used.
5. It is also possible to select the essential information (genes or reactions), which excludes them from the optimization process, effectively reducing the search space.
6. The optimization basic setup allows the user to define the maximum number of solution evaluations (which is the number of phenotype simulations that will be performed—a termination condition), as well as the maximum number of genetic modifications. The user can also choose whether the solution sizes are fixed or if the algorithm is allowed to find solutions with different sizes. When the user decides for variable size solutions, this number will be used as the maximum possible size of the solutions.

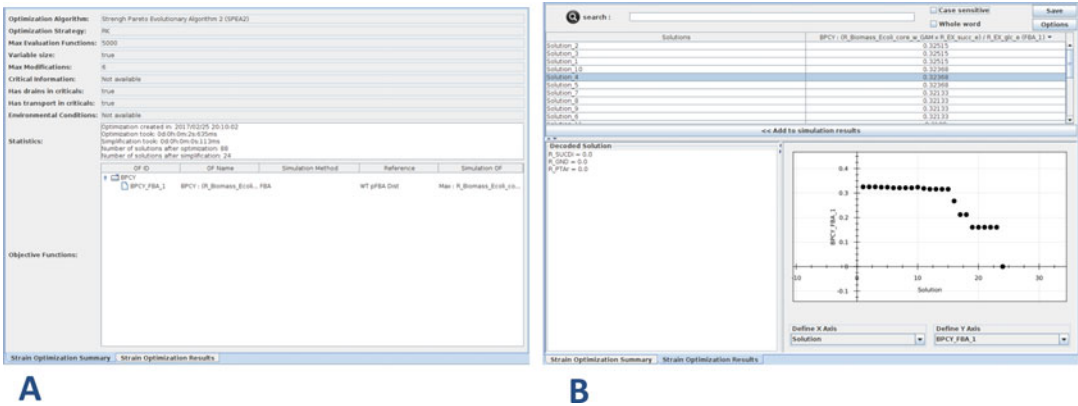


Fig. 21 Views of an evolutionary operation result performed with *E. coli* core model using BPCY objective function for the production of succinate using glucose as carbon source. **(a)** resume of all the parameters configured to run the operation. **(b)** view showing all the mutant strains computed by the method

By default the optimization is configured to find designs of up to six modifications, within a maximum of 5000 solution evaluations. These configurations are conservative and must be adapted for different models/conditions (e.g., by increasing the number of function evaluations, number of modifications, among others).

The result of the Evolutionary Algorithms is a new object in the clipboard, placed within the Optimization Results list, under the Evolutionary data type. The Optimization Result data type has two views associated (Fig. 21):

1. Strain optimization summary: where all the inputs used to produce the result are provided.
2. Strain optimization results, where the computed solutions are described. On this view, the user is also able to extract a specific simulation result to the clipboard for further analysis.

4 Notes

1. Representation of Genetic modifications in CBMs

The suppression of a given metabolic function can be accomplished in vivo by disrupting the functioning of specific genes by targeted modifications through homologous recombination [74] or intron introduction [75]. With in silico CBMs, this task is commonly accomplished by imposing constraints that force the flux of the disabled reactions to zero, deterring the occurrence of flux over those reactions, followed by the evaluation of the effect of that perturbation.

Constraint-based phenotype prediction methods can take advantage of gene-reaction rules information contained within

the models to perform gene deletions instead of reactions deactivations. To attain that, it is necessary to use the Boolean rules to derive the off genes and understand which reactions must be deactivated. Such deactivation is performed by setting the reaction bounds of those reactions to 0. This type of gene deletions provides a closer resemblance to the in vivo scenario, since they inherently account for the occurrence of multifunctional and multimeric proteins, as well as isoenzymes.

Gonçalves and coworkers [76] suggested an alternative to use constraint-based phenotype prediction methods to represent over or under regulation of certain fluxes in the network. This methodology uses a wild-type flux distribution as a reference and a set of “mutated” reactions, where each reaction is associated with an expression level (p). If the expression value is smaller than 1, the reaction is under-expressed constraining its flux to be lower than the reference value, which will be multiplied by p . If the expression value is >1 the reaction will be overexpressed, and its flux is constrained to be higher than the reference value. The knockouts are formulated with an expression level of zero.

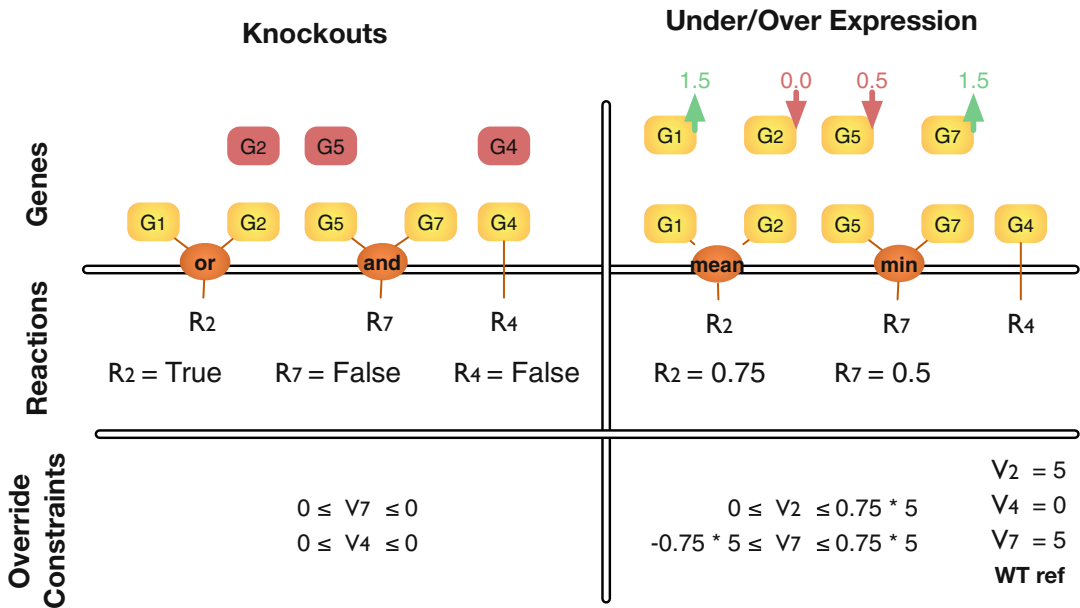


Fig. 22 Schema explaining how to represent genetic modifications within CBMs. In the left side, knockouts are shown (G2, G4, and G5), then the GPRs were applied to verify which reactions should be knocked out, and finally the bounds associated to deactivated reactions were set to 0. In the right side, under/over expressions are shown, where G1 and G7 were overexpressed, G5 was under-expressed and G2 was knocked out. GPRs were changed to support real values (Or operator changed to MEAN; AND operator changed to MIN), and these functions were applied to calculate the expression values of the reactions. Then, the bounds of the reactions were changed taking into account the wild-type flux distribution

To apply this methodology on the gene level, it was necessary to convert Boolean operators to functions able to handle numerical expression values by using the available GPR information. Typical examples of AND situations are the formation of protein complexes, where the expression of all the genes is necessary to produce the enzyme. With this in mind, the authors decided to translate the AND operator to the minimum value between the two expression values, implying that the expression of all the genes is necessary to produce the enzyme, and the gene for which the minimum number of copies are expressed will act as a bottleneck for the activity of the corresponding enzyme. On the other hand, typical examples of OR situations are genes encoding isoenzymes, where the genes involved work independently, so OR operators were transformed to the mean of all expression values. Thus, by using the converted GPR, it is possible to convert a set of genes associated with an expression value to a set of reactions associated with an expression value, allowing the flux bounds transformation explained before.

Figure 22 shows an example of how a genetic transformation is translated to the reactions bounds.

References

1. Jackson DA, Symons RH, Berg P (1972) Biochemical method for inserting new genetic information into {DNA} of Simian Virus 40: circular {SV40} {DNA} molecules containing lambda phage genes and the galactose operon of *Escherichia coli*. *Proc Natl Acad Sci U S A* 69:2904–2909
2. Lobban PE, Kaiser AD (1973) Enzymatic end-to-end joining of {DNA} molecules. *J Mol Biol* 78:453–471
3. Stephanopoulos G, Aristidou AA, Nielsen J (1998) *Metabolic engineering: principles and methodologies*. Academic press, New York, NY
4. Bailey J (1991) Toward a science of metabolic engineering. *Science* 252:1668–1675
5. Woolston BM, Edgar S, Stephanopoulos G (2013) *Metabolic engineering: past and future*. *Annu Rev Chem Biomol Eng* 4:259–288
6. Bailey JE, Birnbaum S, Galazzo JL et al (1990) Strategies and challenges in metabolic engineering. *Ann N Y Acad Sci* 589:1–15
7. Gavrilescu M, Maria G, Yusuf C (2005) Biotechnology—a sustainable alternative for chemical industry. *Biotechnol Adv* 23:471–499
8. Hatti-Kaul R, Rajni H-K, Ulrika T et al (2007) Industrial biotechnology for the production of bio-based chemicals – a cradle-to-grave perspective. *Trends Biotechnol* 25:119–124
9. Kitano H (2002) Systems biology: a brief overview. *Science* 295:1662–1664
10. Bork P (2005) Is there biological research beyond Systems Biology? A comparative analysis of terms. *Mol Syst Biol* 1:E1–E2
11. Schuster SC (2007) Next-generation sequencing transforms today's biology. *Nat Methods* 5:16–18
12. Papin JA, Tony H, Palsson BO et al (2005) Reconstruction of cellular signalling networks and analysis of their properties. *Nat Rev Mol Cell Biol* 6:99–111
13. Samaga R, Regina S, Steffen K (2013) Modeling approaches for qualitative and semi-quantitative analysis of cellular signaling networks. *Cell Commun Signal* 11:43
14. Covert MW, Knight EM, Reed JL et al (2004) Integrating high-throughput and computational data elucidates bacterial networks. *Nature* 429:92–96
15. Reed JL, Iman F, Ines T et al (2006) Towards multidimensional genome annotation. *Nat Rev Genet* 7:130–141
16. Fong SS, Palsson BØ (2004) Metabolic gene-deletion strains of *Escherichia coli* evolve to computationally predicted growth phenotypes. *Nat Genet* 36:1056–1058
17. Papin JA, Joerg S, Price ND et al (2004) Comparison of network-based pathway analysis methods. *Trends Biotechnol* 22:400–405

18. Rocha I, Förster J, Nielsen J (2008) Design and application of genome-scale reconstructed metabolic models. *Methods Mol Biol* 416:409–431
19. Thiele I, Palsson BØ (2010) A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nat Protoc* 5:93–121
20. Hamilton JJ, Reed JL (2014) Software platforms to facilitate reconstructing genome-scale metabolic networks. *Environ Microbiol* 16:49–59
21. Notebaart RA, van Enckevort FHJ, Francke C et al (2006) Accelerating the reconstruction of genome-scale metabolic networks. *BMC Bioinformatics* 7:296
22. Agren R, Liu L, Shoaie S et al (2013) The {RAVEN} toolbox and its use for generating a genome-scale metabolic model for *Penicillium chrysogenum*. *PLoS Comput Biol* 9:e1002980
23. Henry CS, DeJongh M, Best AA et al (2010) High-throughput generation, optimization and analysis of genome-scale metabolic models. *Nat Biotechnol* 28:977–982
24. Dias O, Rocha M, Ferreira EC et al (2015) Reconstructing genome-scale metabolic models with merlin. *Nucleic Acids Res* 43:3899–3910
25. Smallbone K, Kieran S, Messiha HL et al (2013) A model of yeast glycolysis based on a consistent kinetic characterisation of all its enzymes. *FEBS Lett* 587:2832–2841
26. Smallbone K, Kieran S, Evangelos S et al (2010) Towards a genome-scale kinetic model of cellular metabolism. *BMC Syst Biol* 4:6
27. Bordbar A, Monk JM, King ZA et al (2014) Constraint-based models predict metabolic and associated cellular functions. *Nat Rev Genet* 15:107–120
28. Gombert AK, Jens N (2000) Mathematical modelling of metabolism. *Curr Opin Biotechnol* 11:180–186
29. Edwards JS, Palsson BO (1999) Systems properties of the *haemophilus influenzae* Rd metabolic genotype. *J Biol Chem* 274:17410–17416
30. Monk J, Nogales J, Palsson BO (2014) Optimizing genome-scale network reconstructions. *Nat Biotechnol* 32:447–452
31. Bailey JE (2001) Reflections on the scope and the future of metabolic engineering and its connections to functional genomics and drug discovery. *Metab Eng* 3:111–114
32. Rocha I, Maia P, Evangelista P et al (2010) {OptFlux}: an open-source software platform for in silico metabolic engineering. *BMC Syst Biol* 4:45
33. Glez-Peña D, Reboiro-Jato M, Maia P et al (2010) {AIBench}: a rapid application development framework for translational research in biomedicine. *Comput Methods Programs Biomed* 98:191–203
34. Schulze KL, Lipe RS (1964) Relationship between substrate concentration, growth rate, and respiration rate of *Escherichia coli* in continuous culture. *Arch Mikrobiol* 48:1–20
35. Bowen JH (1968) Basic principles and calculations in chemical engineering. *Chem Eng Sci* 23:191
36. Reed JL, Vo TD, Schilling CH, Palsson BO (2013) An expanded genome-scale model of *Escherichia coli* K-12 (i JR904 GSM/GPR). *Genome Biol* 4(9):R54
37. Hucka M, Finney A, Sauro HM et al (2003) The systems biology markup language ({SBML}): a medium for representation and exchange of biochemical network models. *Bioinformatics* 19:524–531
38. Bornstein BJ, Keating SM, Jouraku A et al (2008) {LibSBML}: an {API} library for {SBML}. *Bioinformatics* 24:880–881
39. von Kamp A, Schuster S (2006) Metatool 5.0: fast and flexible elementary modes analysis. *Bioinformatics* 22:1930–1931
40. Funahashi A, Morohashi M, Kitano H et al (2003) {CellDesigner}: a process diagram editor for gene-regulatory and biochemical networks. *BioSilico* 1:159–162
41. Funahashi A, Matsuoka Y, Jouraku A et al (2008) {CellDesigner} 3.5: a versatile modeling tool for biochemical networks. *Proc IEEE* 96:1254–1265
42. Cline MS, Smoot M, Cerami E et al (2007) Integration of biological networks and gene expression data using Cytoscape. *Nat Protoc* 2:2366–2382
43. Noronha A, Vilaça P, Rocha M (2014) An integrated network visualization framework towards metabolic engineering applications. *BMC Bioinformatics* 15:420
44. Fruchterman TMJ, Reingold EM (1991) Graph drawing by force-directed placement. *Softw Pract Exp* 21:1129–1164
45. Le Novère N, Hucka M, Mi H et al (2009) The systems biology graphical notation. *Nat Biotechnol* 27(8):735–741
46. Schellenberger J, Park JO, Conrad TM et al (2010) {BiGG}: a Biochemical Genetic and Genomic knowledgebase of large scale metabolic reconstructions. *BMC Bioinformatics* 11:213
47. King ZA, Dräger A, Ebrahim A et al (2015) Escher: a web application for building, sharing, and embedding data-rich visualizations of biological pathways. *PLoS Comput Biol* 11: e1004321
48. King ZA, Lu J, Dräger A et al (2016) BiGG models: a platform for integrating, standardizing and sharing genome-scale models. *Nucleic Acids Res* 44:D515–D522

49. Varma A, Palsson BO (1994) Metabolic flux balancing: basic concepts, scientific and practical use. *Nat Biotechnol* 12:994
50. Papoutsakis ET (1984) Equations and calculations for fermentations of butyric acid bacteria. *Biotechnol Bioeng* 26:174–187
51. Orth JD, Thiele I, Palsson BØ (2010) What is flux balance analysis? *Nat Biotechnol* 28:245–248
52. Feist AM, Palsson BO (2010) The biomass objective function. *Curr Opin Microbiol* 13:344–349
53. Schuetz R, Kuepfer L, Sauer U (2007) Systematic evaluation of objective functions for predicting intracellular fluxes in *Escherichia coli*. *Mol Syst Biol* 3:119
54. Ponce de León M, Cancela H, Acerenza L (2008) A strategy to calculate the patterns of nutrient consumption by microorganisms applying a two-level optimisation principle to reconstructed metabolic networks. *J Biol Phys* 34:73–90
55. Lewis NE, Hixson KK, Conrad TM et al (2010) Omic data from evolved *E. coli* are consistent with computed optimal growth from genome-scale models. *Mol Syst Biol* 6:390
56. Segrè D, Vitkup D, Church GM (2002) Analysis of optimality in natural and perturbed metabolic networks. *Proc Natl Acad Sci U S A* 99:15112–15117
57. Snitkin ES, Dudley AM, Janse DM et al (2008) Model-driven analysis of experimentally determined growth phenotypes for 465 yeast gene deletion mutants under 16 different conditions. *Genome Biol* 9:R140
58. Shlomi T, Berkman O, Ruppin E (2005) Regulatory on/off minimization of metabolic flux changes after genetic perturbations. *Proc Natl Acad Sci U S A* 102:7695–7700
59. Becker SA, Feist AM, Mo ML et al (2007) Quantitative prediction of cellular metabolism with constraint-based models: the {COBRA} Toolbox. *Nat Protoc* 2:727–738
60. Wintermute EH, Lieberman TD, Silver PA (2013) An objective function exploiting sub-optimal solutions in metabolic networks. *BMC Syst Biol* 7:98
61. Park JH, Lee KH, Kim TY et al (2007) Metabolic engineering of *Escherichia coli* for the production of L-valine based on transcriptome analysis and in silico gene knockout simulation. *Proc Natl Acad Sci U S A* 104:7797–7802
62. Alper H, Jin Y-S, Moxley JF et al (2005) Identifying gene targets for the metabolic engineering of lycopene biosynthesis in *Escherichia coli*. *Metab Eng* 7:155–164
63. Jung YK, Kim TY, Park SJ et al (2010) Metabolic engineering of *Escherichia coli* for the production of polylactic acid and its copolymers. *Biotechnol Bioeng* 105:161–171
64. Brochado AR, Andrejev S, Maranas CD et al (2012) Impact of stoichiometry representation on simulation of genotype-phenotype relationships in metabolic networks. *PLoS Comput Biol* 8:e1002758
65. Mahadevan R, Schilling CH (2003) The effects of alternate optimal solutions in constraint-based genome-scale metabolic models. *Metab Eng* 5:264–276
66. Maia P, Rocha M, Rocha I (2016) In silico {constraint-based} strain optimization methods: the quest for optimal cell factories. *Microbiol Mol Biol Rev* 80:45–67
67. Ignizio JP (1994) Linear programming. Pearson College Division, London
68. Burgard AP, Pharkya P, Maranas CD (2003) Optknock: a bilevel programming framework for identifying gene knockout strategies for microbial strain optimization. *Biotechnol Bioeng* 84:647–657
69. Patil KR, Rocha I, Förster J et al (2005) Evolutionary programming as a platform for in silico metabolic engineering. *BMC Bioinformatics* 6:308
70. Holland JH (1992) Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control and artificial intelligence. MIT Press, Cambridge, MA
71. Rocha M, Maia P, Mendes R et al (2008) Natural computation meta-heuristics for the in silico optimization of microbial strains. *BMC Bioinformatics* 9:499
72. Maia P, Rocha I, Ferreira EC et al (2008) Evaluating evolutionary multiobjective algorithms for the in silico optimization of mutant strains. In: 8th IEEE International Conference on Bio-Informatics and BioEngineering, BIBE 2008
73. Maia P, Rocha I, Rocha M (2013) An integrated framework for strain optimization. In: 2013 IEEE Congress on Evolutionary Computation, CEC 2013, pp 198–205
74. Datsenko KA, Wanner BL (2000) One-step inactivation of chromosomal genes in *Escherichia coli* {K-12} using {PCR} products. *Proc Natl Acad Sci U S A* 97:6640–6645
75. Frazier CL, San Filippo J, Lambowitz AM et al (2003) Genetic manipulation of *Lactococcus lactis* by using targeted group {II} introns: generation of stable insertions without selection. *Appl Environ Microbiol* 69:1121–1128
76. Gonçalves E, Pereira R, Rocha I et al (2012) Optimization approaches for the in silico discovery of optimal targets for gene over/under-expression. *J Comput Biol* 19:102–114

Metabolic Network Reconstruction and Modeling
Methods and Protocols

Fondi, M. (Ed.)

2018, XI, 410 p. 103 illus., 92 illus. in color. With online
files/update., Hardcover

ISBN: 978-1-4939-7527-3

A product of Humana Press