User Manual for INCA

Lara J. Jazmin, Veronique Beckers, and Jamey D. Young

Table of Contents

1. Introduction

INCA is a MATLAB-based software package for isotopomer network modeling and metabolic flux analysis (MFA). The software can simulate both steady-state and transient isotope labeling experiments using the elementary metabolite unit (EMU) method [1,2]. It can also estimate pathway fluxes based on extracellular flux measurements, pool size measurements, and/or mass isotopomer measurements supplied to the program. The software can perform a variety of statistical tests to determine goodness-of-fit, to compute parameter confidence intervals (i.e., uncertainties), and to assess model identifiability. It can also perform constraint-based analysis of metabolic networks (e.g., flux balance analysis, flux coupling analysis, etc.) and optimize the design of isotope labeling experiments using computational search algorithms.

The software provides a framework for comprehensive analysis of metabolic networks using mass balances and isotopomer balances. The generation of balance equations and their computational solution is completely automated and can be performed on networks of arbitrary complexity. The graphical user interface (GUI) allows the user to input reaction information and experimental data in a simple text format, while offering a variety of powerful analysis tools to design and interpret isotope labeling experiments. All INCA data objects can be loaded and manipulated from the MATLAB command line, and the driver routines can be called directly from the command line or invoked within custom MATLAB scripts. INCA also offers built-in parallelization capabilities for running certain functions within a distributed computing environment.

2. System Requirements

Computer with MATLAB plus Statistics and Optimization toolboxes installed.

Parallel computations require either of the following:

1) A cluster or multiprocessor environment with
    a. Condor job scheduler
    b. MATLAB and MATLAB Compiler toolbox installed on frontend node (used to run INCA and to compile the `serve` function for distribution to compute nodes)
    c. Compute node access to the MATLAB Compiler Runtime (MCR) library
    d. Compute node access to a shared directory where all input/output files are saved/read by the frontend node (Note: directory name must not contain spaces!)
2) The MATLAB Parallel Computing toolbox and Distributed Computing Server (currently untested).

3. Installation and Configuration

Instructions for downloading INCA are provided at the website http://mfa.vueinnovations.com/licensing. Unpack the INCA program files into a directory that can be readily accessed by MATLAB.

4. Getting Started

Begin a new MATLAB session. Change the MATLAB working directory to the INCA root directory. Type “inca” at the MATLAB command prompt and press enter. This will launch INCA and bring up the ***Edit network*** screen (Figure 1).

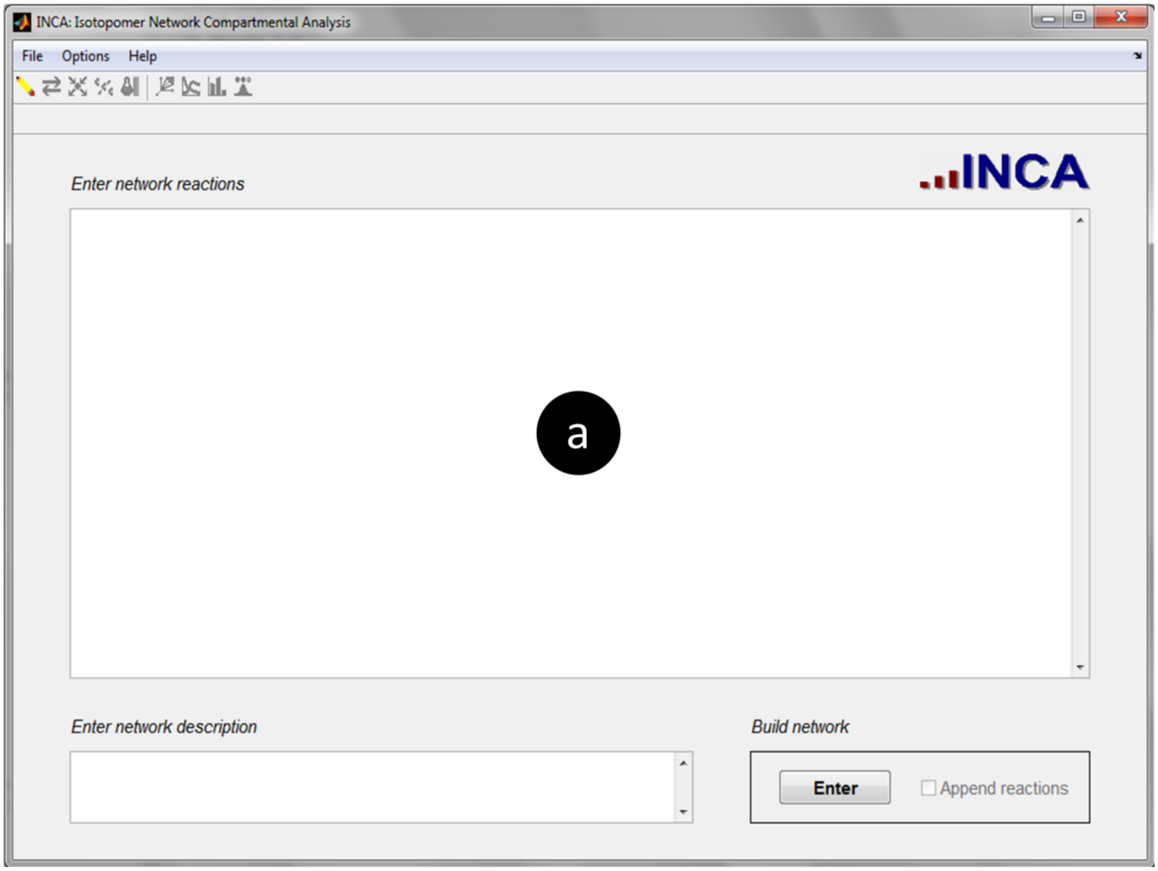


Figure 1. Edit network screen.

Existing networks can be loaded by selecting **File → Open Model** and specifying the location of the saved file. New networks can also be entered manually, which will be discussed in **Section 5. Defining the Model**.

4.1. Menu functions

The main menu at the top of the INCA figure window enables the user to save/load program files, to adjust program options, and to access program documentation. Each menu item is described in detail below.

File →

New: Clears all program inputs/outputs and creates a new INCA model.

Open Model: Loads a previously saved INCA network model.

Open Simulation: Loads a previously saved tracer simulation along with its associated network model. (See Section 8 to learn how to generate a tracer simulation.)

Open Flux Map: Loads a previously saved flux map along with its associated network model and tracer simulation. (See Section 9 to learn how to generate a flux map.)

Import Experiments: Imports experimental datasets from a previously saved model into the currently open model.

Save Model as: Saves the currently open INCA network model.

Save Simulation as: Saves the current tracer simulation and network model.

Save Flux Map as: Saves the current flux map, network model, and tracer simulation.

Export Model as FluxML: Writes the currently open INCA network model to a 13CFLUX2 readable FluxML file. Note: This feature is currently experimental, and some aspects of the conversion need to be manually entered. In particular, reactions with >2 educts are not automatically split into bimolecular reactions as required by 13CFLUX2. Also, natural isotope abundance is not included in the specification of input pools.

Export Model to OpenFLUX: writes the currently open INCA network model to an OpenFLUX readable Excel file. Note: This feature is currently experimental, and some aspects of the conversion need to be manually entered. In particular, reactions with >2 educts are not automatically split into bimolecular reactions. Also, flux measurements are not specified in the Excel file.

Exit INCA: Exits the program

Options →

Simulate natural abundance of labeled atoms: When checked, INCA will simulate the natural isotope abundance of all labeled atoms included in the measured mass isotopomer distributions (See Section 6.3). This box should be checked when the mass isotopomer data has not been previously corrected for natural isotope abundance of labeled atoms, and INCA should simulate the natural isotope contribution when performing flux estimations.

Simulate natural abundance of unlabeled atoms: When checked, INCA will simulate the natural isotope abundance of all unlabeled atoms included in the measured mass isotopomer distrbutions (See Section 6.3). These are typically atoms introduced by derivatization or heteroatoms that cannot become labeled by the tracer. If the mass isotopomer data has not been previously corrected for natural abundance of unlabeled atoms, this box should be checked to simulate the natural isotope contribution of these atoms.

Show corrected MS data: When checked, INCA will correct experimental mass isotopomer data for natural abundance in displayed plots and tables (See Section 6.3). This only affects the data presentation and does not impact the flux estimation procedure.

Simulate sensitivities: When checked, INCA will calculate parameter sensitivities when performing tracer simulations. This may slow down the simulation but is required for subsequent ***Identifiability analysis*** (See Section 8.3).

Simulate steady-state labeling: When checked, INCA will apply steady-state isotopomer balances and will use isotopically stationary MFA for flux estimation. When unchecked, INCA will apply transient isotopomer balances and will use isotopically nonstationary MFA (INST-MFA) to estimate fluxes and pool sizes. ***Note:*** *Some INCA features are only applicable to INST-MFA and will be marked as* ***"INST-MFA only"*** *in the remainder of this document.*

Nonsteady-state simulation parameters: Selecting this menu item will open a dialog box where several options can be set to control the integration of transient isotopomer balance equations (See Section 8.2). (**INST-MFA only**)

Run in parallel using Condor: When checked, INCA will distribute parallel jobs to compute nodes using the Condor job scheduler. This will affect the computation time required for flux estimation, parameter continuation, and Monte Carlo analysis.

Condor settings: Selecting this menu item will open a dialog box where the location of files required for parallelization can be specified to Condor (See Appendix E).

Use random initial guess for flux estimation: When checked, INCA will randomize the initial guess used for flux estimation.

Optimization parameters: Selecting this menu item will open a dialog box where several options can be set to control the search algorithm used for flux estimation (See Section 9.1).

Confidence interval parameters: Selecting this menu item will open a dialog box where several options can be set to control the calculation of confidence intervals using parameter continuation or Monte Carlo analysis (See Section 9.3).

Experiment design parameters: Selecting this menu item will open a dialog box where options can be set to control the search algorithm used for tracer optimization (See Section 10).

Help →

User Manual: Opens this document in a PDF reader.

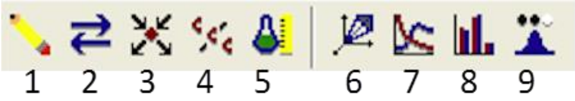
Function Index: Opens an HTML file with links to help documentation on all INCA functions. This is a useful reference guide when calling INCA functions directly from the MATLAB command line or for developing custom scripts that invoke INCA function calls.

About: Displays program version and copyright information.

4.2. Toolbar functions

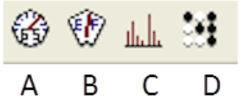
Underneath the main menu is a primary toolbar that enables the user to navigate between different program screens. The icons to the left of the vertical divider link to input screens where the user can set up the network model and supply experimental data to the program. The icons to the right of the divider link to output screens where the user can perform various analyses and view results. When selecting the ***Edit experiments***, ***Tracer simulation***, or ***Flux estimation*** icon using the primary toolbar, a secondary toolbar appears that enables the user to navigate between different subscreens of the selected primary toolbar function. Each toolbar icon is explained further below.

Primary toolbar



1. Edit network (See Section 5.1)
2. Edit reactions (See Section 5.2)
3. Edit nodes (See Section 5.3)
4. Edit metabolites (See Section 5.4)
5. Edit experiments (See Section 6)

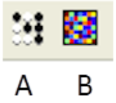
Edit experiments **secondary toolbar**



A. Edit flux measurements (See Section 6.2)
B. Edit pool size measurements (**INST-MFA only**) (See Section 6.2)
C. Edit MS measurements (See Section 6.3)
D. Edit tracers (See Section 6.4)

6. Constraint-based analysis (See Section 7)
7. Tracer simulation (See Section 8)

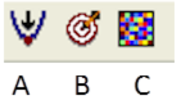
***Tracer simulation* secondary toolbar**



A. Simulate labeling (See Section 8.1)
B. Identifiability analysis (See Section 8.3)

8. Flux estimation

***Flux estimation* secondary toolbar**



A. Optimize parameters (See Sections 9.1-9.3)
B. Assess goodness-of-fit (See Section 9.2)
C. Identifiability analysis (See Section 9.4)

9. Tracer optimization (See Section 10)

5. Defining the Model

5.1. Building a new network

To define a new network, navigate to the ***Edit network*** screen by clicking the ✏ icon on the primary toolbar. Network reactions can be manually entered in the ***Enter network reactions*** panel (Figure 1a) in the following format:

A (abc) -> B (ab) + C (c)

The uppercase letters represent the metabolite nodes, while the lowercase letters in parentheses represent the atom transitions. (Note: It is not necessary to follow this convention of using uppercase for metabolites and lowercase for atom transitions. The parentheses delimit the start and end of each atom transition. Any alphabetic, numeric, or underscore character comprising the regular expression [a-zA-Z_0-9] can be included in the metabolite names and atom transitions.) Irreversible reactions are denoted by a '->' or '=>' arrow and reversible reactions are denoted by a '<->' or '<=>' arrow. Multiple reactions are separated by carriage returns or by placing a semicolon at the end of each reaction equation. A text description of the network model can be typed into the lower-left panel marked ***Enter network description*** (optional). When all reactions have been typed into the text box, click the **Enter** button to build the network.

Metabolites associated with multiple subcellular compartments can be specified by placing a dot separator between the metabolite name and compartment name, as follows:

A.c (abc) -> A.m (abc)

where, for example, '.c' designates the cytosolic compartment and '.m' designates the mitochondrial compartment.

Stoichiometric coefficients other than 1 can be specified by using the '*' symbol to represent multiplication, as follows:

2*A (abc) -> B (cbaabc)

Note that each atom position on the product side of the equation must map to a unique position on the educt side of the equation. Mapping a product atom to more than one educt atom positions, or to a missing position, will result in an error. Furthermore, atom mappings for all reversible reactions must be unique in both directions (i.e., must possess one-to-one correspondence). It is also possible to separate reaction terms with stoichiometric coefficients >1 into multiple terms, as follows:

A (abc) + A (def) -> B (cbadef)

In some cases, this is necessary to avoid ambiguity when specifying the associated atom transitions.

When parsing a reaction with the preceding syntax, the program will name the atoms of each metabolite as ‘1’, ‘2’, ‘3’, etc. in the order they are specified in the metabolite’s atom transitions. It is therefore important that the atom transitions are the same length and that atoms are specified in the same order each time a given metabolite appears in the network. An alternative syntax is available if the user would like to tag each atom with a unique name that can be referenced later. For example, the preceding reaction could be rewritten as follows:

A (C1:a C2:b C3:c) + A (C1:d C2:e C3:f) -> B (C1:c C2:b C3:a C4:d C5:e C6:f)

In this case, the ordering of the atom transition entries is arbitrary because the atoms are referenced by name rather than order. This syntax also allows for multiple characters to be used to designate each atom mapping, which may be necessary if reactions involving >63 atoms (i.e., the number that can be mapped using single characters from the set [a-zA-Z_0-9]) are included in the network.

When writing transitions for atoms of different types (e.g., hydrogen and carbon) it is oftentimes useful to designate uppercase letters for one atom type and lowercase for the other, as follows:

A (AabB) + H (c) -> B (AbBc) + H (a)

Here, we have used uppercase letters for carbon atoms and lowercase for hydrogen atoms bound to those carbons. It is also possible to use the alternative syntax to reference each atom by its name tag:

A (C1:A C2:B H1R:a H1S:b) + H (H1:c) -> B (C1:A C2:B H1:a H2:c) + H (H1:a)

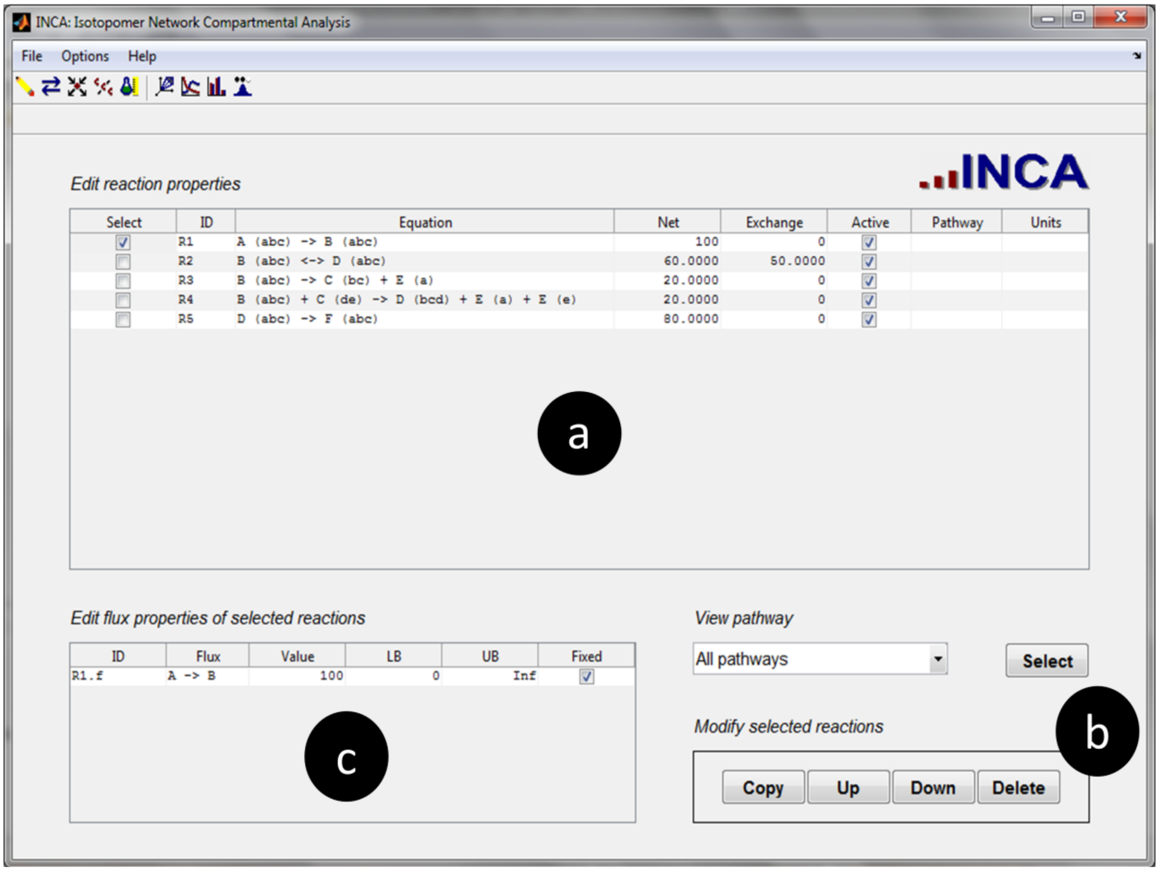
Once a reaction network has been built, additional reactions may be appended to the existing network by checking the **Append reactions** box in the bottom-right corner and entering the additional equations as described above. To avoid duplicating the existing reactions, only the new reactions should be typed into the ***Enter network reactions*** panel prior to clicking **Enter**.

Save the model by selecting **File → Save Model As**, which will open a dialog box to save the newly created model as a MATLAB .mat file.

5.2. Editing network reactions

Users can edit the properties of previously entered reactions by navigating to the ***Edit reactions*** screen, which can be accessed by clicking the ⇄ icon. Several reaction parameters can be directly edited within the ***Edit reaction properties*** panel (Figure 2a).

Under the **ID** column, a unique name tag can be specified for each reaction to replace the default 'R1', 'R2', etc. scheme. For example, designating each reaction by its enzyme name is often a useful choice.


INCA: Isotopomer Network Compartmental Analysis
File
Options
Help
INCA
Edit reaction properties
a
Edit flux properties of selected reactions
c
View pathway
All pathways
Select
Modify selected reactions
b
Copy
Up
Down
Delete

Select	ID	Equation	Net	Exchange	Active	Pathway	Units
☑	R1	A (abc) -> B (abc)	100	0	☑		
□	R2	B (abc) <-> D (abc)	60.0000	50.0000	☑		
□	R3	B (abc) -> C (bc) + E (a)	20.0000	0	☑		
□	R4	B (abc) + C (de) -> D (bcd) + E (a) + E (e)	20.0000	0	☑		
□	R5	D (abc) -> F (abc)	80.0000	0	☑		

ID	Flux	Value	LB	UB	Fixed
R1.f	A -> B	100	0	Inf	☑

Figure 2. Edit reactions screen.

The previously entered reaction equations can be directly edited under the **Equation** column. It is possible to edit the stoichiometry, atom transitions, or reversibility of each reaction, and these changes will be immediately applied throughout the rest of the program.

The net and exchange fluxes for each reaction can be edited under the **Net** and **Exchange** columns, respectively. The net flux is defined as the difference between the forward flux and the backward flux of a reversible reaction, or simply as the forward flux of an irreversible reaction. The exchange flux is the minimum of the forward and backward fluxes of a reversible reaction, or is zero for an irreversible reaction. For example, the reaction

$$A \underset{v_B}{\overset{v_F}{\rightleftarrows}} B$$

has net and exchange fluxes given by:

$$v_{net} = v_F - v_B$$
$$v_{exch} = \min(v_F, v_B)$$

Reactions can be activated or inactivated by clicking the checkboxes in the **Active** column. Whenever a new value is entered in the **Net** or **Exchange** column or edits are made to the **Equation** or **Active** column, the software will attempt to reconcile the flux values to ensure network feasibility. This involves solving a least-squares problem using MATLAB's `lsqlin` function to identify a flux vector that is "as close as possible" to the user-specified flux distribution while satisfying all mass balances of the form $\mathbf{S} \cdot \mathbf{v} = \mathbf{0}$, where **S** is the stoichiometric matrix of the balanced species and **v** is the steady-state flux vector of all forward and backward fluxes [3,4]. The flux values entered in this table will be used as the basis and/or initial guess for subsequent ***Constraint-based analysis***, ***Tracer simulation***, ***Flux estimation***, and ***Tracer optimization*** calculations.

Reactions can be assigned to a pathway by entering a text string under the **Pathway** column. This can be used to group reactions into subsets that can be viewed selectively by clicking the desired pathway under the ***View pathway*** drop-down menu. Assigning reactions to pathways is optional and does not impact the subsequent analysis.

Additionally, measurement units for the flux values can be entered under the **Units** column. This text string will be used to annotate figures produced by the ***Flux estimation*** screen. However, specifying units for each reaction is optional and does not impact the underlying calculations. Therefore, it is important to maintain internal consistency when specifying the stoichiometric coefficients of each reaction equation. Metabolite mass balances are obtained by multiplying the stoichiometric coefficient of a given metabolite by the corresponding net flux of each reaction in which it appears, and then summing over all reactions. Each mass balance thus corresponds to a row in the matrix equation $\mathbf{S} \cdot \mathbf{v} = \mathbf{0}$. Different flux units can be specified for different reactions and/or different mass units can be specified for different metabolites within the same network, as long as commensurate units are obtained whenever the mass balances are calculated. In other words, each row of the matrix equation $\mathbf{S} \cdot \mathbf{v} = \mathbf{0}$ must represent a sum of terms with like units.

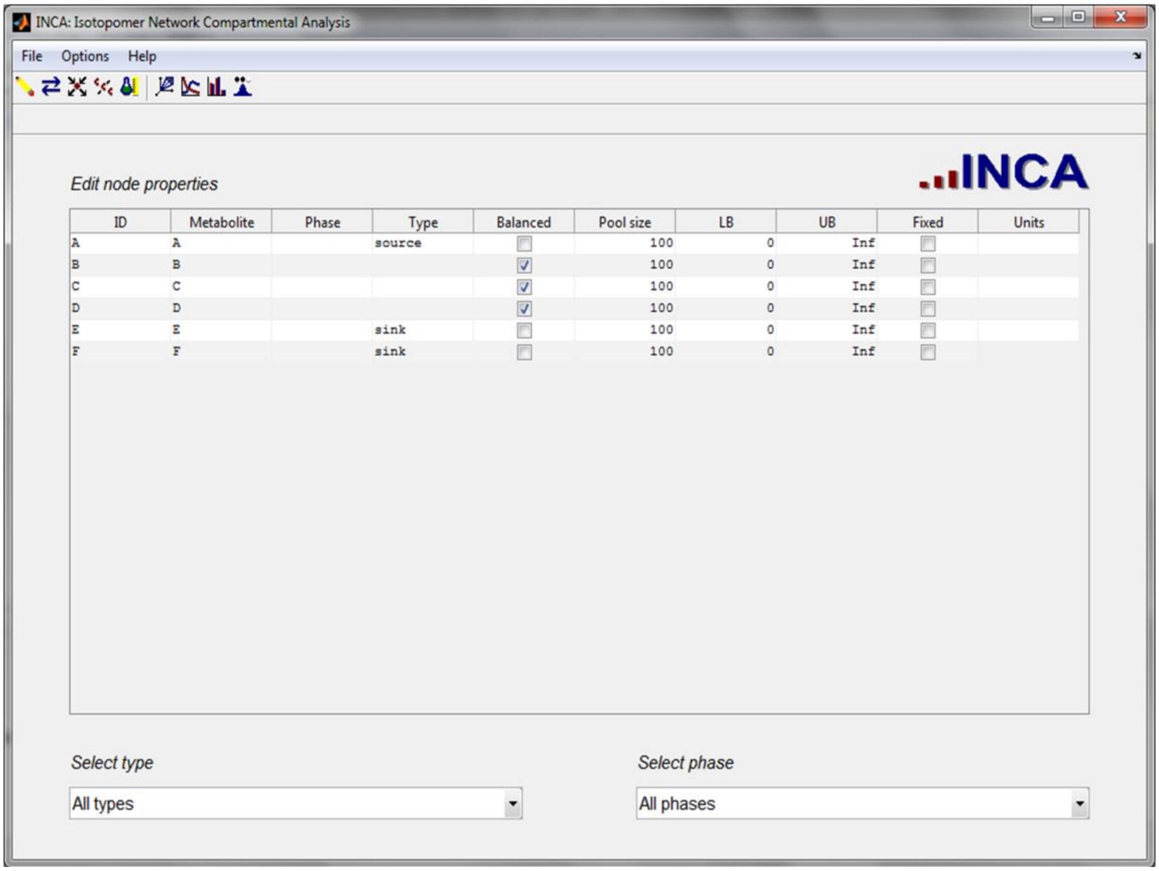
Individual reactions can be selected by checking the box under the **Select** column in the ***Edit reaction properties*** panel (Figure 2a). Alternatively, reactions can be selected as a group by choosing the desired subset using the ***View pathway*** drop-down menu and then clicking the **Select** button that appears just to the right of the menu. The selected reactions can be copied, deleted, or moved up or down in the reaction list by clicking the appropriate button under the ***Modify selected reactions*** panel (Figure 2b).

All selected reactions will also appear as separate forward and backward fluxes under the ***Edit flux properties of selected reactions*** panel (Figure 2c). The flux names appearing in the **ID** column are not editable. They are automatically generated by appending a '.f' (forward flux) or '.b' (backward flux) extension to the corresponding reaction name tag. Also, the **Flux** column is automatically set to the corresponding flux equation. The user can directly edit the unidirectional flux values appearing in this panel, and the **Net** and **Exchange** columns of the corresponding reaction in the ***Edit reaction properties*** panel (Figure 2a) will be appropriately updated. It is also possible to set upper and lower bounds on each flux by adjusting the values in the **LB** and **UB** columns, respectively. These bounds will be applied when performing the analyses on the ***Constraint-based analysis*** and ***Flux estimation*** screens. Finally, any flux may be set to a constant value by checking the corresponding box in the **Fixed** column. This value will be held fixed when performing ***Constraint-based analysis*** or ***Flux Estimation*** calculations and will be excluded from the set of adjustable parameters when performing ***Identifiability analysis***. This feature is useful when performing flux estimations relative to a fixed reference flux. It can also be used to specify a desired initial flux distribution by temporarily fixing several independent flux values and allowing the program to calculate the remaining dependent fluxes using mass balance constraints. (The number of fixed fluxes must be less than or equal to the degrees of freedom of the stoichiometric matrix **S**. Once the network is exactly determined by the fixed fluxes, no additional fluxes can be fixed by the user.) After the desired initial flux distribution has been obtained, the user can reset the fixed fluxes without disturbing the specified flux values.

5.3. Editing network nodes

Properties of the metabolite nodes (i.e., species) involved in the reaction network can be viewed and edited from the ***Edit nodes*** screen, which can be accessed by clicking the icon.

Node properties can be edited in the ***Edit node properties*** panel (Figure 3). The first four columns (**ID**, **Metabolite**, **Phase**, **Type**) are set automatically by the program. The node IDs are the species names collected from the reaction equations. If the model includes compartmentation, the metabolite and compartment names are displayed separately under the **Metabolite** and **Phase** columns, respectively. Nodes are classified as either a 'source', 'sink', or 'internal' species. Source nodes are consumed by one or more reactions but are not produced within the network. Conversely, sink nodes are produced by at least one reaction in the network but are not consumed. Internal nodes are those that appear as both products and educts within the network. They are shown with a blank entry under the **Type** column. The node list can be filtered based on **Phase** or **Type** attributes using the respective drop-down menus at the bottom of the screen.



Edit node properties

ID	Metabolite	Phase	Type	Balanced	Pool size	LB	UB	Fixed	Units
A	A		source	□	100	0	Inf	□	
B	B			☑	100	0	Inf	□	
C	C			☑	100	0	Inf	□	
D	D			☑	100	0	Inf	□	
E	E		sink	□	100	0	Inf	□	
F	F		sink	□	100	0	Inf	□	

Select type: All types

Select phase: All phases

Figure 3. Edit nodes screen. Note that the last five columns (**Pool size**, **LB**, **UB**, **Fixed**, **Units**) are applicable to **INST-MFA only** and are only visible when the ***Simulate steady-state labeling*** option is unchecked.

Mass balances on source and sink nodes are infeasible, by definition, and therefore must be excluded from the model. This is indicated by the fact that all checkboxes in the **Balanced** column are forcibly unchecked by the program. By default, all source nodes are assumed to be unlabeled unless specified as a tracer (see Section 6.4). Because the labeling of source nodes is externally specified, the isotopomer balances on these species are also excluded from the model. On the other hand, isotopomer balances on sink nodes are included in the model, since the labeling of sink nodes is dependent on the network fluxes and tracer inputs.

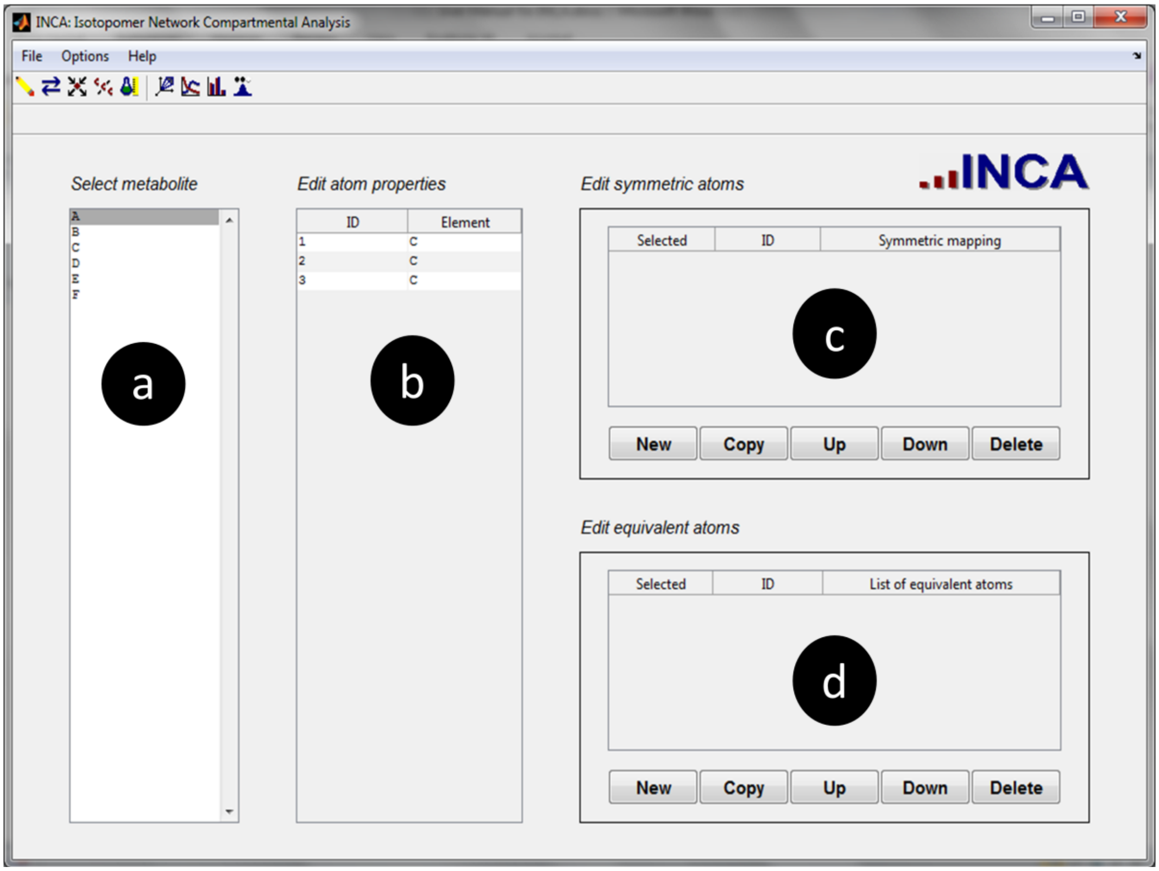
In some cases, it is desirable to specify an internal metabolite as unbalanced because it is not subject to mass balance constraints. For example, metabolites such as CO_2 or O_2 are typically treated as unbalanced because they can freely transfer into or out of the system from the extracellular environment. Also, it is often necessary to treat cofactors such as ATP or NAD(P)H as unbalanced because they are supplied or consumed by processes that have not been explicitly included in the reaction network. By unchecking the corresponding box in the **Balanced** column, the mass and isotopomer balances on an internal node can be excluded from the model. Similar to source nodes, unbalanced internal nodes are assumed to be unlabeled unless the node is specified as a tracer (see Section 6.4). This is tantamount to assuming an infinite pool size for the node, since its labeling is decoupled from the rest of the network.

INST-MFA only: The node pool size can be edited under the **Pool size** column. This represents the total mass of the node, expressed in units that are determined by the metabolite balances and the dynamic time scale of the labeling measurements. For example, if the *i*th row of the mass balance equation $\mathbf{S}\cdot\mathbf{v}=\mathbf{0}$ has units of [C_i] per unit time [t], choosing the time unit [t] of the simulation implicitly determines the mass units [C_i] of the corresponding node. Setting a node's **Pool size** to 0 will apply a quasi-steady state assumption when solving the dynamic isotopomer balances associated with this species. As a result, the labeling of this node will respond instantaneously to changes in the labeling of its precursor metabolites, such that isotopic equilibrium is continuously maintained. Any pool sizes set to 0 thus become fixed parameters that are not adjusted by the ***Flux estimation*** algorithm or included in the ***Identifiability analysis***. It is possible to set upper and lower bounds on each nonzero pool size by editing the values in the **LB** and **UB** columns, respectively. These bounds will be applied when performing ***Flux estimation***. It is also possible to restrict a given pool size to its user-specified value by checking the **Fixed** box. Additionally, **Units** for each pool size can be specified in the final column. However, this text string will only be used to annotate figures produced by the ***Flux estimation*** screen and does not impact the underlying calculations.

5.4. Editing metabolite properties

Further information about each of the metabolites participating in the network can be viewed and updated in the ***Edit metabolites*** screen (Figure 4), accessed by clicking the icon.

Each network metabolite is listed in the ***Select metabolite*** panel (Figure 4a). After selecting a metabolite from the list, the name tag of each atom comprising the metabolite and its element type are displayed in the ***Edit atom properties*** panel (Figure 4b). The entries in the **ID** column cannot be edited, because they are determined by parsing the atom transitions in the reaction network (See Section 5.1). However, the **Element** type can be edited according to the composition of the metabolite. It should be noted that only atoms that can be potentially labeled need to be included in the atom transitions. Therefore, in most cases only a single element type will be listed in this table. (The default is 'C', since ^{13}C tracers are most commonly used for MFA studies.) However, if isotopes other than carbon are administered, these atoms will need to be included in the atom transitions and also updated in the ***Edit atom properties*** panel (Figure 4b).



ID	Element
1	C
2	C
3	C

Figure 4. Edit metabolites screen.

Rotationally symmetric molecules (e.g., fumarate or succinate) cause scrambling of isotopic labeling [1]. This can be handled directly by accounting for each symmetric orientation as separate terms in every reaction equation where a symmetric metabolite appears. Alternatively, INCA provides a simplified approach that involves accounting for only one orientation when entering the reaction equations and then specifying the symmetric mapping under the ***Edit symmetric atoms*** panel (Figure 4c). For example, the alpha-ketoglutarate (AKG) dehydrogenase reaction could be written as follows (without directly accounting for the symmetry of succinate):

AKG (abcde) -> Succinate (bcde) + CO2 (a)

Then, selecting 'Succinate' from the list in Figure 4a will allow the user to enter its symmetry mapping by clicking **New** in Figure 4c. This will create a new **Symmetric mapping** with the default orientation '4321'. This is similar to an atom transition in that it maps each atom to its new position following a symmetry operation: ‘1’ maps to ‘4’, ‘2’ maps to ‘3’, ‘3’ maps to ‘2’, and ‘4’ maps to ‘1’. This also assumes that each atom is named in order according to the default convention ‘1’, ‘2’, ‘3’, ‘4’. If custom atom names were used when defining the reaction network, the default entries in the **Symmetric mapping** column will need to be updated. For example, if the AKG dehydrogenase reaction had been written

AKG (abcde) -> Succinate (C1:b C2:c C3:d C4:e) + CO2 (a),

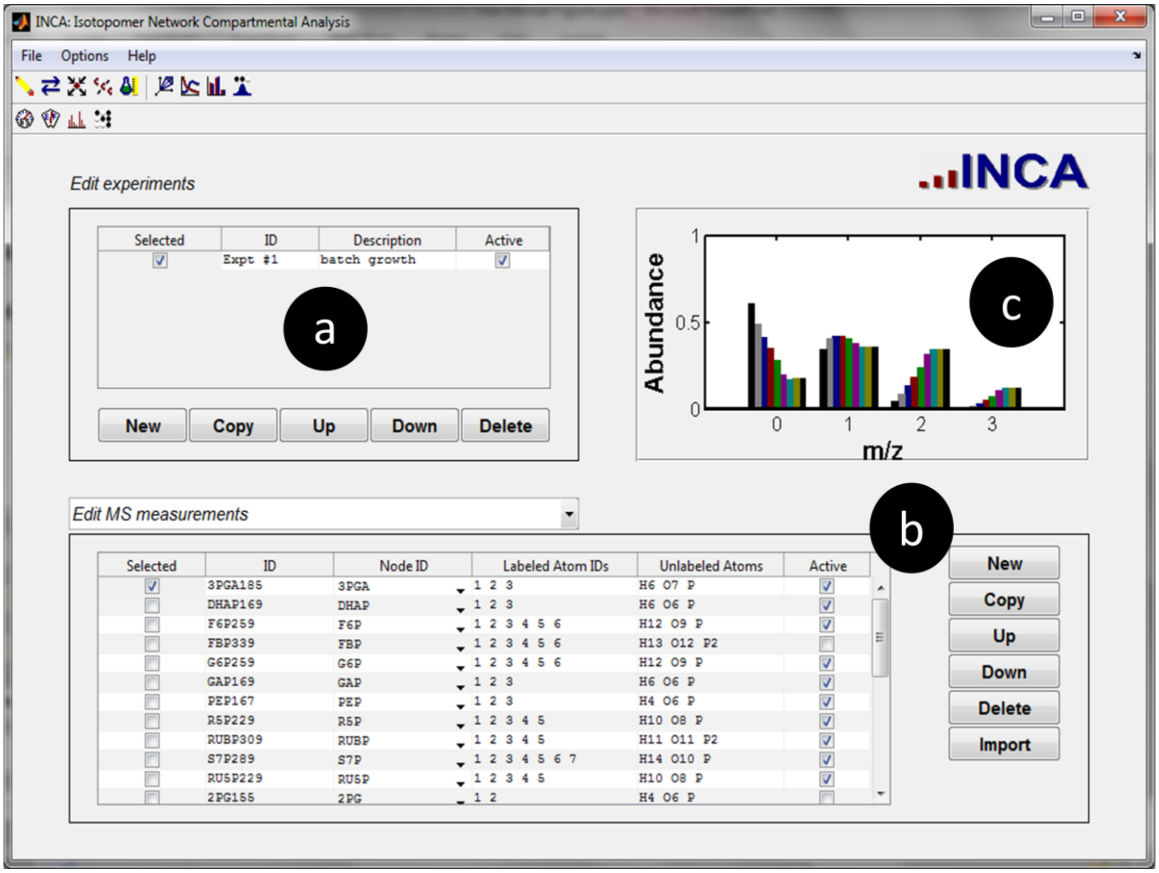
the atoms of Succinate would be assigned the name tags ‘C1’,’C2’,’C3’, and ‘C4’ when INCA parses the network. The **Symmetric mapping** for Succinate would then need to be entered as ‘C1:C4 C2:C3 C3:C2 C4:C1’.

A different type of scrambling occurs when metabolites contain groups of equivalent atoms that are biochemically indistinguishable [1]. Labeling of any atom within the group is rapidly equilibrated across all equivalent positions. For example, biochemical equivalency is commonly assumed among the three hydrogen atoms bound to a methyl carbon or the two oxygen atoms bound to a carboxylate carbon. This can be an important consideration when analyzing hydrogen (^{2}H) or oxygen (^{18}O) tracer experiments. However, atom equivalency does not typically impact carbon (^{13}C) tracer experiments. In INCA, equivalent atom groups can be entered by selecting the appropriate metabolite from the list in Figure 4a and then clicking the **New** button in the ***Edit equivalent atoms*** panel (Figure 4d). This will create a default entry with ‘1 2 3’ under **List of equivalent atoms**, which identifies the atoms ‘1’, ‘2’, and ‘3’ as equivalent. If a different group of equivalent atoms is desired, list the atom name tags separated by spaces as shown in the default example.

It is possible, although not typical, that a metabolite may have more than one symmetry operation or more than one group of equivalent atoms. In either case, the user can create multiple entries in Figure 4c or 4d by clicking **New** more than once. A unique **ID** tag can be created for each entry. You can also **Copy**, **Delete**, or move entries **Up** or **Down** in the list by clicking the appropriate buttons within each panel.

6. Entering Experimental Datasets

When performing MFA, fluxes are estimated by minimizing the difference between simulated and experimental measurements. This requires specification of (i) flux measurements, (ii) isotope labeling measurements, and (iii) isotope tracers to the program. When performing INST-MFA, the user can also specify pool size measurements. All of these experimental parameters can be entered in the ***Edit experiments*** screen (Figure 5), which can be accessed by clicking the icon. When selecting this icon on the primary toolbar, the ***Edit MS measurements*** panel is initially shown at the bottom of the screen (Figure 5b).



Selected	ID	Description	Active
☑	Expt #1	batch growth	☑

Selected	ID	Node ID	Labeled Atom IDs	Unlabeled Atoms	Active
☑	3PGA185	3PGA	1 2 3	H6 O7 P	☑
☐	DHAP169	DHAP	1 2 3	H6 O6 P	☑
☐	F6P259	F6P	1 2 3 4 5 6	H12 O9 P	☑
☐	FBP339	FBP	1 2 3 4 5 6	H13 O12 P2	☐
☐	G6P259	G6P	1 2 3 4 5 6	H12 O9 P	☑
☐	GAP169	GAP	1 2 3	H6 O6 P	☑
☐	PEP167	PEP	1 2 3	H4 O6 P	☑
☐	R5P229	R5P	1 2 3 4 5	H10 O8 P	☑
☐	RUBP309	RUBP	1 2 3 4 5	H11 O11 P2	☑
☐	S7P289	S7P	1 2 3 4 5 6 7	H14 O10 P	☑
☐	RU5P229	RU5P	1 2 3 4 5	H10 O8 P	☑
☐	2PG155	2PG	1 2	H4 O6 P	☐

Figure 5. Edit experiments screen with *Edit MS measurements* lower panel shown. Panels for editing the flux measurements, pool size measurements (**INST-MFA only**), and tracers can also be accessed using the secondary toolbar.

6.1. Adding a new experimental dataset

To create a new experimental dataset, click **New** in the upper ***Edit experiments*** panel (Figure 5a) and enter a name for the experiment under the **ID** column. An optional experiment **Description** can also be entered as a text string. **Selected** experiments can be copied, deleted, or moved up or down in the experiment list by clicking the appropriate button within the ***Edit experiments*** panel. One powerful feature of INCA is that it allows multiple experiments to be **Active** simultaneously. When performing MFA, the program will search for a single flux map that minimizes the total lack-of-fit with respect to all active experiments. This can be of interest when fitting several replicate experiments or parallel labeling experiments with different tracers.

6.2. Entering flux and/or pool size measurements

Directly measured fluxes (e.g., growth rate, substrate uptake, product formation) can be entered by clicking the icon on the secondary toolbar to display the ***Edit net flux measurements*** panel at the bottom of the screen.

To add flux measurements to a specific experimental dataset, select that experiment in the upper-left ***Edit experiments*** panel (Figure 5a). Click **New** in the ***Edit net flux measurements*** panel. Select a **Reaction ID** from the drop-down menu that corresponds to the measured flux. Enter the measured flux value and its standard error (i.e., uncertainty) in the columns marked **Data** and **Error**, respectively. Individual flux measurements can be activated or deactivated using the checkboxes in the **Active** column. **Selected** reactions can be copied, deleted, or moved up or down in the list by clicking the appropriate button within the ***Edit net flux measurements*** panel.

One common mistake when entering experimental measurements is to set the standard error to zero. Because each of the terms in the least-squares objective function is weighted by the inverse of its standard error, this will lead to an infinite objective function value. When performing MFA calculations, this will cause the flux estimation to terminate early and return an infinite (Inf) sum-of-squared residuals (SSR) value. The program will display a warning message to the MATLAB command line indicating which measurement was at fault.

In the same way that flux measurements are entered, pool size measurements can also be entered when performing INST-MFA calculations. This can be done by clicking the icon on the secondary toolbar to display the ***Edit pool size measurements*** panel at the bottom of the screen (**INST-MFA only**). Rather than selecting a **Reaction ID**, however, the user will select a **Node ID** that corresponds to the measured species. Then, the measured pool size value (**Data**) and standard error (**Error**) can be entered and activated/deactivated (**Active**) by editing the appropriate column entries.

6.3. Entering MS measurements

To enter and edit mass isotopomer measurements, click the icon on the secondary toolbar to access the ***Edit MS measurements*** panel (Figure 5b) at the bottom of the screen.

To add MS measurements to a specific experimental dataset, select that experiment in the upper-left ***Edit experiments*** panel (Figure 5a). Click **New** in the ***Edit MS measurements*** panel (Figure 5b) to create a new MS measurement. Enter an **ID** for the MS measurement (e.g., 'Ala232') and select the measured species from the drop-down menu under the **Node ID** column. Enter a list of atom IDs (separated by spaces) included in the MS fragment ion under the heading **Labeled Atom IDs**. These must be chosen from the list of atom IDs shown for the corresponding metabolite on the ***Edit metabolites*** screen. Under the **Unlabeled atoms** heading, enter the molecular formula of any unlabeled atoms that are included in the measured fragment ion but have been excluded from the atom transitions in the reaction network (e.g., 'H6O7P'). These are typically atoms introduced by chemical derivatization or heteroatoms that cannot become labeled by the tracer. Individual MS measurements can be activated or deactivated using the checkboxes in the **Active** column. Also, **Selected** measurements can be copied, deleted, or moved up or down in the list by clicking the appropriate buttons at the right of the table.

Mass isotopomer data must be entered for each active MS measurement listed in the ***Edit MS measurements*** panel. There are two methods to add mass isotopomer data to a specific MS measurement. The first method is to select the corresponding measurement ID from the ***Edit MS measurements*** drop-down menu. This will bring up a new lower panel with information specific to the selected MS measurement. (You can return to the list of all MS measurements in the lower panel at any time by selecting the top-most entry 'Edit MS measurements' from the drop-down menu.)

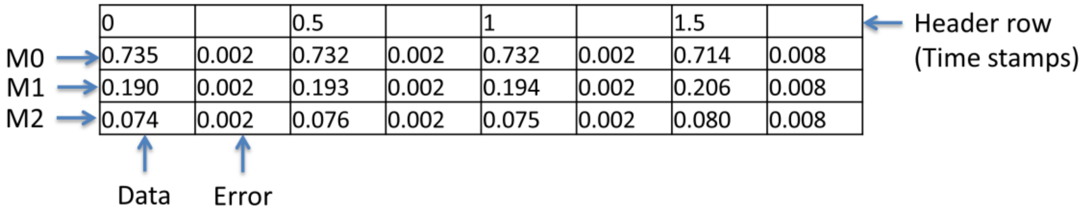
Clicking the **New** button will add a new data point to the table with the column headings **ID**, **Time** (**INST-MFA only**), **Data**, **Error**, and **Active**. **ID** is a unique text identifier used to designate each replicate measurement or measurement time point. The **Time** column is only visible when the ***Simulate steady-state labeling*** option is unchecked. It contains numeric entries that represent the time when each data point was collected. The mass isotopomer distribution vector (MDV) obtained at each point is entered into the **Data** column. This is a vector that contains the fractional abundance of each mass isotopomer ordered from lowest mass (M0) to highest mass, with each element of the vector separated by a space. For example, an MDV with M0 abundance of 0.5, M1 abundance of 0.3, and M2 abundance of 0.2 would be entered as '0.5 0.3 0.2'. 'NaN' can also be entered for any elements of the MDV that are missing or unmeasurable. When performing MFA calculations, the program introduces MS scaling factors that renormalize the measured MDVs. By optimizing the values of these scaling factors, the program removes the effects of missing (i.e., 'NaN') elements. In addition to the measured MDV, the standard error associated with each mass isotopomer abundance should be entered in the **Error** column. This can be entered as a vector of the same length as the **Data** vector, where each element represents the standard error of the corresponding MDV element. It is also possible to enter a single value that applies to all elements of the **Data** vector. If no value is entered in the **Error** column, a default standard error vector will be assumed based on typical values: 0.3 mol% for mass isotopomer abundances ≤0.5 mol% with linear scaling up to 1 mol% for mass isotopomer abundances ≥25 mol%. Note that each element of the **Error** vector must be larger than zero; otherwise, infinite SSR values will be returned by the MFA calculation. One or more data points can be activated/deactivated by checking the appropriate boxes under the **Active** column heading. When performing stationary MFA, the program will treat each active point as a replicate MS measurement and will attempt to fit all active data points simultaneously. When performing INST-MFA, on the other hand, each active MS measurement represents a separate time point, and the program will optimize flux and pool size parameters to match the experimental labeling trajectories.

Once the measured MDVs have been entered, they can be **Selected** for plotting in the accompanying figure panel (Figure 5c). **Selected** data points can also be copied, deleted, or moved up or down in the list by clicking the appropriate buttons to the right of the table. Additionally, INCA is capable of correcting MDVs for natural isotope abundance of labeled and unlabeled atoms using the approach of Fernandez et al. [5]. Selecting **Options→Show corrected MS data** will display the corrected data in the figure panel as well as the data table. The table entries under the **Data** and **Error** columns will become uneditable when the corrected data are displayed. The user can edit the raw data again only after unchecking **Show corrected MS data.** If the MS data were previously corrected for natural isotope abundance prior to being entered, selecting **Show corrected MS data** will produce erroneous values because further correction is unnecessary. In this case, the user should also de-select **Simulate natural abundance of labeled atoms** and **Simulate natural abundance of unlabeled atoms** under the **Options** menu prior to performing tracer simulations or flux estimations.

The second method for MS data entry is to import data from an Excel spreadsheet. After selecting ***Edit MS measurements*** from the drop-down menu to reveal the full list of MS measurements, a button will appear in the list at the far-right labeled **Import**. Checking the desired measurement as **Selected** followed by clicking **Import** will bring up a dialog box that will guide the user through the import process. The method of import is platform-dependent, as described below.

<u>Importing MS data from Excel: Mac vs. Windows</u>

For both Mac and Windows operating systems, separate MDV data points associated with the **Selected** MS measurement should be arranged into columns within the Excel spreadsheet from lowest (M0) to highest mass isotopomer (Figure 6). These values will be imported into the **Data** column of the associated MS measurement panel in INCA. Each MDV column should have a time stamp along the header row that will be imported with the MS data. (If only steady-state labeling data are imported, the time values will not be used by the program and arbitrary time stamps can be placed in the header row.) The adjacent column to the right of each MDV column should contain the standard errors associated with the MDV. These values will be imported into the **Error** column. For Macs, each MS measurement should be placed on a separate worksheet within the Excel file. After clicking the **Import** button, a dialog box will prompt the user to enter the name of the Excel file and, on Macs, the worksheet to be imported. For Windows, the data for separate MS measurements (i.e., fragment ions) can be placed on the same worksheet, and the area to be imported can be interactively selected after specifying the Excel file.



0		0.5		1		1.5	
0.735	0.002	0.732	0.002	0.732	0.002	0.714	0.008
0.190	0.002	0.193	0.002	0.194	0.002	0.206	0.008
0.074	0.002	0.076	0.002	0.075	0.002	0.080	0.008

M0
M1
M2
Data
Error
Header row
(Time stamps)

Figure 6. Example of MS data formatted for import from Excel.

6.4. Editing tracer information

Information about the tracer(s) used in each experiment is required by INCA to correctly model the isotope labeling data. This information can be edited by selecting the icon on the secondary toolbar to access the ***Edit tracers*** panel at the bottom of the screen. To add a tracer to a particular dataset, select that experiment in the ***Edit experiments*** panel (Figure 5a) and click **New** in the ***Edit tracers*** panel. Provide a unique name tag for the tracer in the **ID** column and select the labeled species in the **Node ID** drop-down list. The fractional abundance of the tracer should be entered in the **Enrichment** column as a numerical value between 0 and 1. Multiple tracers can be entered, corresponding to the same or different labeled species, by clicking **New** more than once. If the total enrichment of a particular labeled species sums to less than 1, the balance is assumed to be unlabeled (or naturally labeled, if **Options→Simulate natural abundance of labeled atoms** is selected). Any source nodes or unbalanced nodes that are not specified in the ***Edit tracers*** panel are assumed to be 100% unlabeled (or naturally labeled). **Selected** tracers can be copied, deleted, or moved up or down in the list by clicking the appropriate buttons to the right of the table.

Each tracer can be further edited by selecting its ID from the ***Edit tracers*** drop-down menu. Clicking the **New** button to the right of the table will create a group of labeled atoms that are assumed to have the same isotopic purity (i.e., fractional enrichment). For each table row created, a space-delimited list of **Atom IDs** can be entered to designate the metabolite atoms that are included in the group. The labeling of each atom group is specified by entering its **Atom MDV**, which represents the MDV of each individual atom in the group. For example, entering the vector '0 1' corresponds to a group of atoms that have 0% probability of being unlabeled (M0) and 100% probability of being M1 labeled. This syntax is easily generalizable to isotopes with mass shifts >1 (e.g., a 100% ^{18}O tracer would have an **Atom MDV** of '0 0 1' because it is shifted two mass units away from the lowest mass ^{16}O isotope). It can also be used to designate tracers with <100% isotopic purity by entering the appropriate mass distribution (e.g., a ^{13}C tracer with 99% isotopic purity would have an **Atom MDV** of '0.01 0.99'). The specified **Atom MDV** applies to all atoms within the group and assumes the same isotopic purity for all atoms. For example, a glucose tracer that is labeled with 95% isotopic purity on atom 1 and 99% purity on atom 6 should be specified with two groups of labeled atoms: one with **Atom ID** '1' and **Atom MDV** '0.05 0.95' and the other with **Atom ID** '6' and **Atom MDV** '0.01 0.99'. Checking the **Selected** box for one or more atom groups will plot the combined MDV of these groups in the upper-right panel (Figure 5c). **Selected** atom groups can also be copied, deleted, or moved up or down in the list by clicking the appropriate buttons at the right of the table.

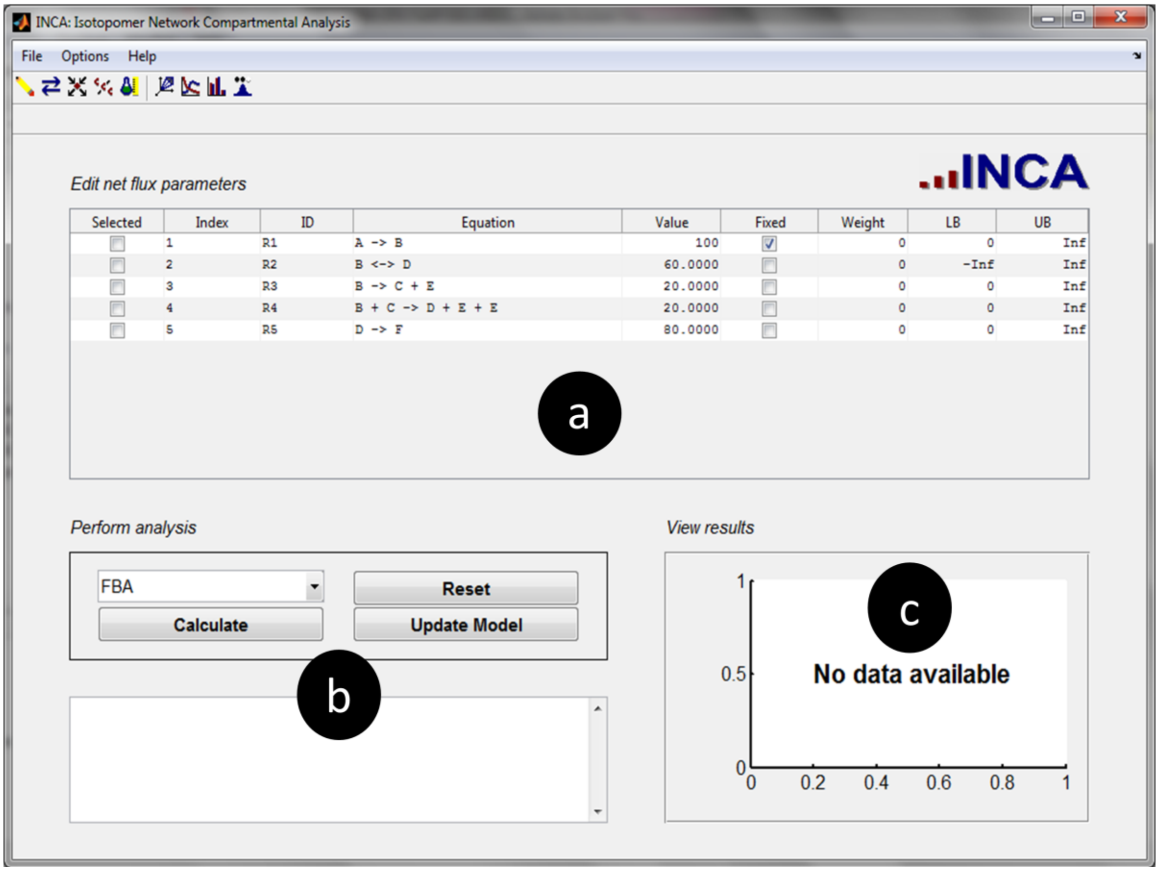
7. Constraint-based Analysis

There are several types of constraint-based analysis that can provide insight into metabolic network capabilities, even in the absence of experimental measurements [3,6]. These methods are based on analyzing the solution space, or “flux cone”, associated with the undetermined system of mass balance equations $\mathbf{N}\cdot\mathbf{v}_{\mathbf{net}} = \mathbf{0}$, where **N** is the stoichiometric matrix associated with the net fluxes contained in the vector $\mathbf{v}_{\mathbf{net}}$. Once a reaction network has been entered into INCA, selecting the icon from the primary toolbar will display the ***Constraint-based analysis*** screen (Figure 7), which provides a variety of tools to assess network function and to predict metabolic phenotypes. Many of these tools are also useful when developing new models for MFA, as they can help users identify potential network inconsistencies such as “blocked” pathways (where net flux is forced to zero by mass balance constraints), or locations where material can enter/leave the network via unbalanced nodes.

The main ***Edit net flux parameters*** panel (Figure 7a) displays all active reactions in the network. The **Index**, **ID**, **Equation, Value**, and **Fixed** columns are not editable. They are set according to the values entered on the ***Edit reactions*** screen, and the user must navigate back to that screen in order to edit these values directly. On the other hand, the values in the **Weight, LB**, and **UB** columns can be edited to specify objective function weights, flux lower bounds, and flux upper bounds, respectively. The default values in the **LB** and **UB** columns are taken from the flux bounds entered previously on the ***Edit reactions*** screen. However, changes made to these columns only apply to ***Constraint-based analysis*** calculations and will not propagate to other screens. The entries in the **Weight**, **LB**, and **UB** columns can be reset to their default values at any time by clicking the **Reset** button under the ***Perform analysis*** panel (Figure 7b).

When performing certain calculations (e.g., FBA and MOMA), the resulting solution vector will be returned in the **Value** column. In order to copy this result over to the ***Edit reactions*** screen so that it will become the new flux basis of the network model, the user must click the **Update Model** button under the ***Perform analysis*** panel (Figure 7b). Otherwise, the result will be lost when navigating away from the ***Constraint-based analysis*** screen.

The drop-down menu in the ***Perform analysis*** panel (Figure 7b) is used to select the type of analysis requested, and the **Calculate** button is used to generate the results. After calculations have been completed, a status message will appear in the text box at the lower-left of the screen with a description of program outputs and any warnings returned during the analysis. A graphical representation of the analysis results will also be displayed in the ***View results*** panel (Figure 7c).



Edit net flux parameters

Selected	Index	ID	Equation	Value	Fixed	Weight	LB	UB
□	1	R1	A -> B	100	☑	0	0	Inf
□	2	R2	B <-> D	60.0000	□	0	-Inf	Inf
□	3	R3	B -> C + E	20.0000	□	0	0	Inf
□	4	R4	B + C -> D + E + E	20.0000	□	0	0	Inf
□	5	R5	D -> F	80.0000	□	0	0	Inf

Figure 7. Constraint-based analysis screen.

Further details about each of the ***Constraint-based analysis*** techniques available within INCA are provided in the following sections. Note that all of these features were added to the program recently, and they should be treated as "experimental" until more extensive testing has been completed.

7.1. FBA

Flux balance analysis (FBA) applies linear programming to search for a flux vector $\mathbf{v}_{\mathbf{net}}$ that maximizes a scalar objective function $z = \mathbf{w} \cdot \mathbf{v}_{\mathbf{net}}$ subject to the mass balance constraints $\mathbf{N} \cdot \mathbf{v}_{\mathbf{net}} = \mathbf{0}$ and upper/lower bounds on all fluxes. Here, **w** is a "weighting vector" that specifies the contribution, or weight, of each net flux to the overall objective function value. The elements of this vector can be entered directly under the **Weight** heading. Selecting 'FBA' from the drop-down menu in Figure 7b and clicking **Calculate** applies MATLAB's `linprog` function to calculate the optimal flux vector determined by the entries in the **Weight** column and the lower and upper bounds in the **LB** and **UB** columns, respectively. The solution vector is displayed in the **Value** column, and the result is plotted in the figure panel at lower-right (Figure 7c). Further information about FBA can be found in references [3] and [7].

7.2. MOMA

Minimization of metabolic adjustment (MOMA) is a method to predict the phenotype of recombinant “knockout” strains. It assumes that the metabolic network of the host cell is regulated to minimize the impact of genetic perturbations, and that the flux phenotype of a knockout strain will be “as close as possible” (in the least-squares sense) to that of its parent strain, subject to the additional flux constraints imposed by gene deletions. The MOMA prediction for the knockout flux vector $\mathbf{v_{KO}}$ is obtained by minimizing the quadratic objective function $z = \|\mathbf{v_{KO}} - \mathbf{v_{WT}}\|^2$ subject to the mass balance constraints $\mathbf{N} \cdot \mathbf{v_{KO}} = \mathbf{0}$ and upper/lower bounds on all net fluxes in $\mathbf{v_{KO}}$. Here, $\mathbf{v_{WT}}$ is the net flux vector of the “wild-type” parent strain. This is a linear least-squares problem that can be solved using MATLAB’s `lsqlin` function to determine the net flux vector $\mathbf{v_{KO}}$ predicted for the knockout strain. Typically, the flux constraints on $\mathbf{v_{KO}}$ are more restrictive than those on $\mathbf{v_{WT}}$, thus forcing the knockout solution to deviate from that of its wild-type parent.

Performing MOMA predictions in INCA involves first setting the initial flux basis shown in the **Value** column to that of the wild-type strain. These values may be entered directly on the ***Edit reactions*** screen (see Section 5.2) or may be the output of a previous MFA or FBA calculation. (*Tip*: when entering fluxes on the ***Edit reactions*** screen, it helps to temporarily ‘fix’ fluxes as they are specified so that they won’t be further adjusted by the program when subsequent fluxes are entered. The fixed fluxes can be freed up once again after all values have been specified, before returning to the ***Constraint-based analysis*** screen.) After the wild-type flux basis has been set, the next step is to adjust the flux bounds in the **LB** and/or **UB** columns to reflect the gene deletions introduced to the knockout strain (e.g., set the upper and lower bounds of all knocked-out reactions to zero). Finally, select ‘MOMA’ from the drop-down menu in Figure 7b and click **Calculate** to generate the MOMA flux prediction. The knockout solution vector is displayed in the **Value** column, and a comparison of the wild-type and mutant strains is plotted in the figure panel at lower-right (Figure 7c). Further information about MOMA can be found in references [3] and [8].

7.3. FVA

Flux variability analysis (FVA) determines the range of flux values that can be attained while holding the objective function at its maximum level $z_{\max}$. The calculation applies MATLAB’s `linprog` function to first maximize and then minimize each flux, one at a time, subject to the constraint that $z = z_{\max}$ (in addition to the mass balance constraints $\mathbf{N} \cdot \mathbf{v_{net}} = \mathbf{0}$ and user-specified flux bounds). This defines the upper and lower limits of the allowable range for each flux under optimal conditions. Selecting ‘FVA’ from the drop-down menu in Figure 7b and clicking **Calculate** replaces the user-specified values shown in the **LB** and **UB** columns with the flux limits determined by the FVA algorithm, using the objective function weights specified in the **Weight** column. The flux variability ranges are plotted graphically in the figure panel at lower-right (Figure 7c), and the median of each range is displayed in the **Value** column of the data table.

If all upper and lower flux limits are equal to each other, it implies that the FBA solution for the specified network and objective function is unique. In this case, the median values returned in the **Value** column should match the FBA solution. If some upper and lower flux limits are not equal to each other, this implies that degenerate FBA solutions exist. In this case, the variability range for each flux represents the uncertainty of its FBA solution. Another useful application of the FVA feature is to determine the entire feasible range of each flux, determined solely by the mass balance constraints and user-specified flux bounds. This can be accomplished by setting all entries in the **Weight** column to zero prior to performing FVA so that all feasible flux distributions will have the same objective function value (=0). Further information about FVA can be found in references [3] and [9].

7.4. FCF

The flux coupling finder (FCF) algorithm can be used to identify structural network dependencies that link fluxes together. The algorithm is similar to FVA and is described in detail by Burgard et al. [10]. Pairs of reaction fluxes are classified as either (1) directionally coupled, if the activity of one flux implies the activity of the other without the converse necessarily holding true, (2) partially coupled, if the activity of one flux implies the activity of the other and vice versa, or (3) fully coupled, if activity of one flux fixes the activity of the other to a proportional value. The FCF also identifies "blocked" reactions, which are incapable of carrying flux under steady-state conditions.

Selecting 'FCF' from the drop-down menu in Figure 7b and clicking **Calculate** plots the flux coupling results as a matrix diagram in the figure panel at lower-right (Figure 7c). Each colored element of the matrix (e.g., with row index *i* and column index *j*) represents the coupling relationship between two fluxes (e.g., v_i and v_j). A white entry indicates no coupling between v_i and v_j. A dark red entry indicates full coupling between v_i and v_j ($v_i \Leftrightarrow v_j$). A light red entry indicates directional coupling whereby a nonzero v_i implies a nonzero v_j ($v_i \rightarrow v_j$). If the symmetric element at position (*j*, *i*) is also light red, this indicates partial coupling between v_i and v_j ($v_i \leftrightarrow v_j$). Finally, a black entry along the matrix diagonal indicates that the corresponding flux is blocked.

7.5. Robustness analysis

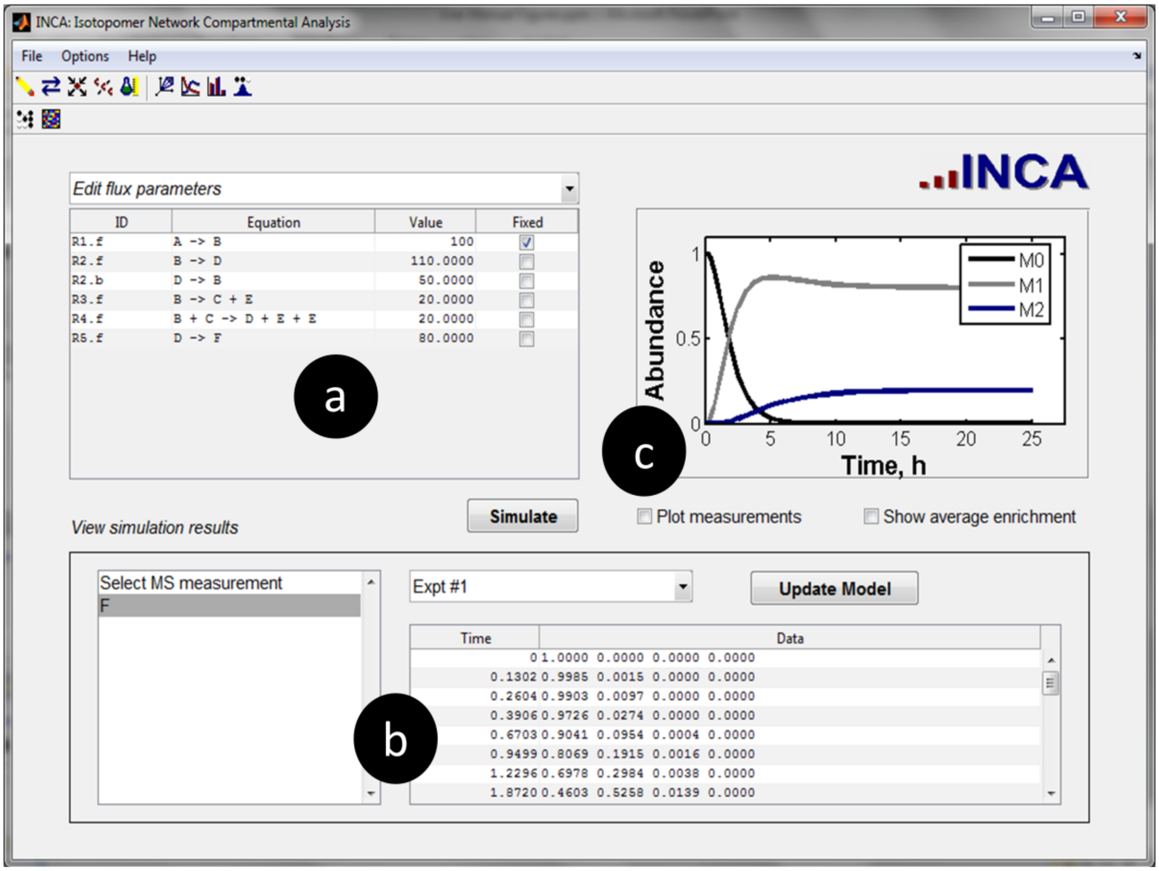
Robustness analysis is an approach to assess the sensitivity of a metabolic objective function to changes in a single flux v_j. This involves solving the FBA subproblem repeatedly while varying v_j between its lower and upper limits in a step-wise manner. To perform robustness analysis in INCA, the user must first specify the objective function weights in the **Weight** column and then select the flux to vary by checking the corresponding box in the **Selected** column. Finally, selecting 'Robustness analysis' from the drop-down menu in Figure 7b and clicking **Calculate** plots the robustness analysis results in the figure panel at lower-right (Figure 7c). The plot depicts the maximum objective function value obtainable as a function of the varied flux. The slope of this plot represents the shadow price of the varied flux. Further information about robustness analysis can be found in reference [3].

7.6. PhPP analysis

Phenotype phase plane (PhPP) analysis is similar to robustness analysis, except that two fluxes are varied simultaneously to produce a two-dimensional plot. To perform PhPP analysis in INCA, the user must first specify the objective function weights in the **Weight** column and then select two fluxes to vary by checking the corresponding boxes in the **Selected** column. Finally, selecting 'PhPP analysis' from the drop-down menu in Figure 7b and clicking **Calculate** plots the phenotype phase plane in the figure panel at lower-right (Figure 7c). The plot depicts the maximum objective function value obtainable as a function of the two varied fluxes. The plot is typically separated into several distinct regions where the relative slopes (i.e., shadow prices) of the two varied fluxes remain constant. These regions represent distinct metabolic phases with differing flux phenotypes. Further information about PhPP analysis can be found in references [3] and [11].

8. Tracer Simulation

MFA involves solving an "inverse problem" where metabolic fluxes and pool sizes are estimated by iterative least-squares regression of measured isotope labeling patterns. At each iteration, a "forward problem" is solved in order to simulate measured isotopomer distributions resulting from a given metabolic network and a given set of parameter estimates. This involves numerical solution of isotopomer balance equations for all measurable metabolites, either at steady state [1] or non-steady state [2]. In INCA, clicking the icon will display the ***Tracer simulation*** screen (Figure 8), which allows the user to directly simulate mass isotopomer distributions (i.e., solve the forward problem) based on an initial set of flux and pool size parameters. This can be useful for determining which measurements are sensitive to certain fluxes. It can also aid in the selection of tracers or tracer combinations that are capable of precise flux quantification. Therefore, simulation of isotope labeling experiments based on best-guess parameter estimates is an important tool for identifying optimal combinations of tracers, measurements, and sampling time points. This can be done before any actual experiments have taken place, thus saving time and effort through careful *a priori* experiment design.



Edit flux parameters

ID	Equation	Value	Fixed
R1.f	A -> B	100	☑
R2.f	B -> D	110.0000	□
R2.b	D -> B	50.0000	□
R3.f	B -> C + E	20.0000	□
R4.f	B + C -> D + E + E	20.0000	□
R5.f	D -> F	80.0000	□

Expt #1

Time	Data
0	1.0000 0.0000 0.0000 0.0000
0.1302	0.9985 0.0015 0.0000 0.0000
0.2604	0.9903 0.0097 0.0000 0.0000
0.3906	0.9726 0.0274 0.0000 0.0000
0.6703	0.9041 0.0954 0.0004 0.0000
0.9499	0.8069 0.1915 0.0016 0.0000
1.2296	0.6978 0.2984 0.0038 0.0000
1.8720	0.4603 0.5258 0.0139 0.0000

Figure 8. Tracer simulation screen (example shown is based on a transient isotopomer model).

8.1. Simulating isotope labeling

In order to simulate isotope labeling, initial flux estimates must be entered into the program. This can be done either at the ***Edit reactions*** screen (Section 5.2) or within the ***Edit flux parameters*** panel (Figure 8a) on the ***Simulate labeling*** screen that appears when first selecting the icon on the primary toolbar. (The ***Simulate labeling*** screen can also be reached from other ***Tracer simulation*** screens by clicking the icon on the secondary toolbar.) The entries in both the **Value** and **Fixed** columns of the ***Edit flux parameters*** panel (Figure 8a) can be edited directly. Whenever edits are made to these columns, the software will attempt to reconcile the flux values to ensure network feasibility. (*Tip:* it is sometimes helpful to temporarily 'fix' fluxes as they are specified so that they won't be adjusted by the program when subsequent fluxes are entered. The fixed fluxes can be freed up once again after all flux values have been specified.) Similarly, node properties can be entered by selecting ***Edit node parameters*** from the drop-down menu at the top-left of Figure 8. The panel that appears in Figure 8a can be used to specify which nodes are **Balanced**. It can also be used to enter initial **Pool size** estimates when performing transient isotopomer simulations (**INST-MFA only**).

Once the network parameters have been specified, clicking the **Simulate** button will launch the calculation. A progress bar will appear that can be used to cancel the simulation at any time, if desired. When complete, the progress bar will disappear and control will return to the INCA window. The listbox in the ***View simulation results*** panel (Figure 8b) will be immediately populated with the names of all simulated isotopomer measurements available for inspection. The results can be viewed by clicking the desired MS measurement in the list and selecting the experiment of interest from the **Select experiment** drop-down menu. The simulated mass isotopomer distributions will be displayed in the data table at the bottom-right and will be plotted in the figure panel above (Figure 8c). When performing steady-state simulations, a single mass isotopomer distribution 'M0 M1 M2 …' will be displayed for each MS measurement in the **Data** column of the table and will be shown as a bar graph in the figure window. When performing transient simulations, multiple measurement time points will be shown in the table, and a **Time** column will appear to indicate the corresponding sample times (**INST-MFA only**). In this case, the entire labeling trajectory for the selected MS measurement will be displayed as a line plot in the figure window.

Whenever simulated data are displayed in the figure window, it is possible to overlay the plot with the actual data entered on the ***Edit experiments*** screen (see Section 6.3) by checking the **Plot experiment data** box. This will enable a direct comparison between simulated and experimental mass isotopomer data. Additionally, checking the **Show average enrichment** box will plot the average isotopic enrichment of the selected MS measurement, as defined by the formula $(1/N)\sum_{i=1}^{N} Mi \times i$ where N is the number of labeled atoms in the MS ion fragment and Mi is the fractional abundance of the *i*th mass isotopomer.

When performing simulation studies of isotope labeling experiments, it is often useful to perform flux estimation and confidence interval calculations using simulated measurements, in order to assess the flux resolution that is achievable using various experimental designs. This can be readily accomplished by clicking the **Update model** button, which will copy all of the simulated MS measurements into the ***Edit experiments*** screen so that they can be treated as actual experimental data. Then, one can perform flux estimations and confidence interval calculations from the ***Flux estimation*** screen (see Section 9) based on the simulated measurements, even if no actual measurements were initially provided. This is the most rigorous approach to assess the precision of flux estimates obtainable from a particular experiment design. However, it is very important to note that clicking the **Update model** button will overwrite any of the previously entered MS data, so make sure that you have saved the model file before attempting this.

8.2. Adjusting integration parameters (INST-MFA only)

Steady-state isotopomer measurements can be simulated to a high degree of accuracy, because the solution is calculated by inverting a system of linear algebraic equations [1]. Transient isotopomer measurements, on the other hand, contain significant error because they result from numerical integration of ordinary differential equations (ODEs) [2]. Therefore, it is sometimes necessary to adjust the error tolerances and other integration parameters used to solve the transient isotopomer balances by selecting **Options→Nonsteady-state simulation parameters**. (This item is only active when the **Simulate steady-state labeling** option is unchecked.) A dialog box will appear that allows the user to set (i) the simulation time units, (ii) the simulation time span, (iii) the maximum time step, relative integration tolerances for both (iv) the simulated mass isotopomer abundances and (v) the simulated sensitivities, and (vi) the integration timeout. The **simulation time units** entry should contain a text string that will be used to annotate plots produced on the ***Tracer simulation*** and ***Flux estimation*** screens, but otherwise does not impact the underlying calculations. The **simulation time span** is a space-delimited list of simulation time points to be evaluated during the integration. If only two points are entered, these will be treated as the initial and final time points of the simulation, and the intermediate time points will be determined by the ODE solver routine. If more than two points are entered, simulation results will only be returned at the specified times and the outputs at other time points will be suppressed. For example, entering ‘0 5 10 15 20 25’ will only return values at the listed time points. Setting the **maximum time step** to a finite number will restrict the integration time step to be no larger than the value specified. This is useful if the results appear too granular when performing simulations using the default step size determined by the ODE solver. The **relative integration tolerance on simulated data** controls the maximum allowable local error that is introduced at each time step of the integration. Similarly, the **relative integration tolerance on sensitivities** controls the local accuracy of sensitivity equation integration. (The default values of 0.001 and 0.01 for the former and latter, respectively, have been found to provide an acceptable tradeoff between accuracy and performance when simulating typical MS data.) Finally, the **integration timeout** can be set to terminate long-running integrations that exceed the specified limit. This can be quite important when performing INST-MFA, as some parameter sets will cause the isotopomer balances to become overly stiff or will lead to integration time steps that are too small to provide a complete solution within a reasonable time. When the timeout threshold is encountered, the integration will terminate and the INST-MFA algorithm will attempt to find a new parameter set that is closer to the previous point of successful integration.

8.3. Identifiability analysis

There are several local identifiability metrics that can be calculated based on the simulated MS data, if the **Simulate sensitivities** option was checked at the time when the tracer simulation was performed. These can be viewed by clicking the ▦ icon on the secondary toolbar. Doing so will display a heat map of the selected identifiability matrix. There are several possible choices available under the ***Select matrix*** drop-down menu, which are described in further detail below. Each matrix is arranged with adjustable model parameters along the rows. Some matrices have experimental measurements arranged along the columns (contribution and sensitivity matrices) and others have the adjustable parameters arranged along both the rows and columns (covariance and correlation matrices). A single parameter or subset of parameters can be selected for display using the ***Select parameter*** drop-down menu. The experiment of interest can be selected using the ***Select experiment*** drop-down menu, and a single measurement or subset of measurements can be selected using the ***Select measurement*** drop-down menu. (If the covariance or correlation matrix is displayed, these two latter menus will be inactive, and the matrix will be calculated using all available measurements.)

The five possible matrices that can be viewed on the ***Identifiability analysis*** screen are as follows (see Antoniewicz et al. [12] for further details):

- *Contribution matrix.* Each element of this matrix (e.g., at row *i* and column *j*) represents the fractional contribution of the *j*th measurement to the local variance of the *i*th parameter. Therefore, all elements are positive and each row of the matrix sums to 1. When more than one MS measurements are displayed, the contributions of all mass isotopomers and time points are summed to give an overall contribution for each MS measurement. When a single transient MS measurement is displayed, the contributions of all mass isotopomers are summed to give an overall contribution for each time point (**INST-MFA only**). Finally, if a single steady-state measurement is displayed, the contribution of each individual mass isotopomer measurement is shown separately.

- *Scaled covariance matrix.* Each element of this matrix (e.g., at row i and column j) is scaled by dividing the raw covariance matrix entries Σ_{ij} by the sum of $|k_i k_j| + |\Sigma_{ij}|$, where k_i (k_j) is the value of the ith (jth) adjustable parameter. This will scale the matrix elements to the range [-1,1]. Large relative covariances will be indicated by matrix elements that approach 1 or -1, while small relative covariances will be indicated by matrix elements near 0.
- *Correlation matrix.* Each element of this matrix (e.g., at row i and column j) is calculated by dividing the raw covariance matrix entries Σ_{ij} by the square-root of the corresponding variances $\sqrt{\Sigma_{ii}\Sigma_{jj}}$. The diagonal elements of the correlation matrix are 1, by definition. Each off-diagonal element represents the correlation coefficient between parameters k_i and k_j. Coefficients near 1 indicate positive correlation, coefficients near -1 indicate negative correlation, and coefficients near 0 indicate that k_i and k_j are uncorrelated.
- *Scaled sensitivity matrix.* Each element of this matrix (e.g., at row i and column j) represents the partial derivative of the ith parameter with respect to the jth measurement. The absolute value of each element is calculated and scaled by the maximum absolute value within its row, so that all entries are in the range [0,1]. When more than one MS measurements are displayed, the root-mean-square (RMS) value over all mass isotopomers and time points is calculated to give an overall sensitivity for each MS measurement. When a single transient MS measurement is displayed, the RMS value over all mass isotopomers is calculated to give an overall sensitivity for each time point (**INST-MFA only**). Finally, if a single steady-state measurement is displayed, the scaled sensitivity of each individual mass isotopomer measurement is shown separately.
- *Error sensitivity matrix.* This matrix is computed in the same way as described above, except that the derivative is calculated relative to the measurement error rather than the measurement value.

9. Flux Estimation

Solving the inverse problem to estimate intracellular fluxes is at the heart of MFA. This involves an optimization search to identify flux parameters (and pool sizes, in the case of **INST-MFA**) that minimize the sum-of-squared residuals (SSR) between computationally simulated and experimentally determined measurements [13,14]. The inverse problem can be expressed mathematically as

$$\begin{aligned} &\min_{\mathbf{u},\mathbf{c},\mathbf{h}} \Phi = \mathbf{r}^{\mathbf{T}} \cdot \mathbf{\Sigma}^{\text{-1}} \cdot \mathbf{r} \\ &\text{s.t. } \mathbf{K} \cdot \mathbf{u} \geq \mathbf{0},\ \ \mathbf{c} \geq \mathbf{0},\ \ \mathbf{h} \geq \mathbf{0} \\ &\text{where } \mathbf{r} = \mathbf{m}(\mathbf{u},\mathbf{c},\mathbf{h},t) - \hat{\mathbf{m}}(t) \end{aligned}$$

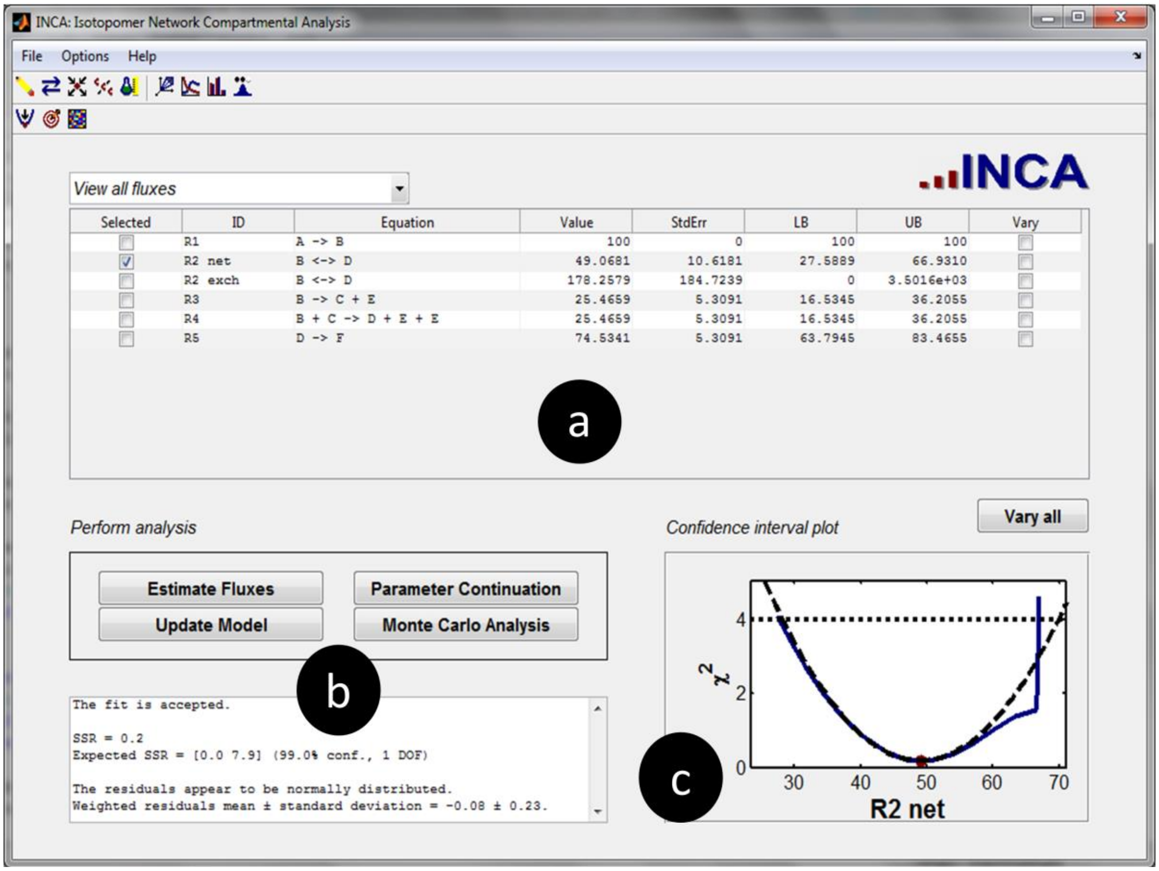
The SSR objective function Φ is minimized by adjusting the vector of free fluxes **u**, the vector of pool sizes **c**, and the vector of MS scaling factors **h**. A scaling factor is used to renormalize each MDV, since only the relative abundances of mass isotopomers are important for flux estimation, not absolute abundances. This enables the program to readily account for missing mass isotopomer measurements, so that complete MDVs are not required by INCA (i.e., some elements can be 'NaN' values). See Mollney et al. [15] for more information on the use of measurement scaling factors in isotopomer models.

The objective function is calculated by first simulating the measurement vector **m**, which is in general a function of **u**, **c**, **h**, and the time coordinate *t*. (For steady-state MFA, **m** is only a function of **u** and **h**.) The vector **m** consists of all simulated flux, pool size, and mass isotopomer distributions for which experimental measurements are available. For any realistic network model (i.e., containing no "traps"), **m** can be explicitly and uniquely calculated by solving the forward problem (i.e., solving all mass and isotopomer balance equations) for a given set of model parameters **u**, **c**, and **h** [16,17]. Once **m** has been simulated, the residual vector **r** can be constructed by differencing each element of **m** with the corresponding element of the experimental measurement vector $\hat{\mathbf{m}}$. To form the objective function Φ, each residual r_i is weighted by the inverse of its standard measurement error σ_i, and the weighted residuals are each squared and then summed together. This is written as a weighted inner product of **r** with itself, using the diagonal weighting matrix $\mathbf{\Sigma}^{\text{-1}} = \mathbf{diag}(\boldsymbol{\sigma})^{\text{-1}}$.

The optimization search is subject to non-negativity constraints on the forward/backward fluxes **v**, the pool sizes **c**, and the scaling factors **h**. The elements of **v** can be obtained by multiplying the free flux vector **u** by the kernel matrix **K**, whose columns span the null space of the stoichiometric matrix **S** (i.e., the columns of **K** form a basis set for generating all solutions of the mass balance equations $\mathbf{S} \cdot \mathbf{v} = \mathbf{0}$) [12]. The search algorithm iteratively adjusts the parameters **u**, **c**, and **h** and recalculates Φ until no further improvement is achievable.

9.1. Parameter optimization

Once a network model has been constructed and experimental datasets have been entered into INCA, MFA calculations can be performed by clicking the ***Flux estimation*** icon . Initially, this will reveal the ***Optimize parameters*** screen (Figure 9), which can also be reached from other ***Flux estimation*** screens by clicking the icon on the secondary toolbar. Clicking the **Estimate Fluxes** button within the ***Perform analysis*** panel (Figure 9b) will launch the iterative MFA calculation procedure.



Selected	ID	Equation	Value	StdErr	LB	UB	Vary
☐	R1	A -> B	100	0	100	100	☐
☑	R2 net	B <-> D	49.0681	10.6181	27.5889	66.9310	☐
☐	R2 exch	B <-> D	178.2579	184.7239	0	3.5016e+03	☐
☐	R3	B -> C + E	25.4659	5.3091	16.5345	36.2055	☐
☐	R4	B + C -> D + E + E	25.4659	5.3091	16.5345	36.2055	☐
☐	R5	D -> F	74.5341	5.3091	63.7945	83.4655	☐

```
The fit is accepted.

SSR = 0.2
Expected SSR = [0.0 7.9] (99.0% conf., 1 DOF)

The residuals appear to be normally distributed.
Weighted residuals mean ± standard deviation = -0.08 ± 0.23.
```

Figure 9. Flux estimation screen.

INCA applies a Levenberg-Marquardt (local search) algorithm to minimize the SSR objective function Φ described above [18,19]. A reduced gradient method is applied to handle the linear inequality constraints on the parameters **u**, **c**, and **h** [20]. At each iteration, the change in the parameter vector $\mathbf{\Delta p = [\Delta u\ \Delta c\ \Delta h]^T}$ is computed by solving a quadratic programming (QP) subproblem of the form

$$\min_{\mathbf{\Delta p}} \tfrac{1}{2}\cdot\mathbf{\Delta p}^{\mathbf{T}}\cdot\mathbf{Z}\cdot\underbrace{\mathbf{Z}^{\mathbf{T}}\cdot\left(\mathbf{H}+\lambda\mathbf{I}\right)\cdot\mathbf{Z}}_{\mathbf{G}}\cdot\mathbf{Z}^{\mathbf{T}}\cdot\mathbf{\Delta p}+\mathbf{\Delta p}^{\mathbf{T}}\cdot\mathbf{Z}\cdot\underbrace{\mathbf{Z}^{\mathbf{T}}\cdot\mathbf{g}}_{\mathbf{f}}$$

$$\text{where } \mathbf{H}=\mathbf{J}^{\mathbf{T}}\mathbf{\Sigma}^{\mathbf{-1}}\mathbf{J},\;\; \mathbf{g}=\mathbf{J}^{\mathbf{T}}\mathbf{\Sigma}^{\mathbf{-1}}\mathbf{r},\;\; \text{and } \mathbf{J}=\frac{\partial\mathbf{m}}{\partial\mathbf{p}}$$

The Jacobian matrix **J** is computed by solving the sensitivity equations that are obtained by differentiating the isotopomer balances with respect to the model parameters [1,2]. The sensitivity equations are solved in tandem with the balance equations at each iteration of the search. The Jacobian matrix can be used to estimate the Hessian matrix **H** = ∇∇Φ and gradient vector **g** = ∇Φ, as shown above. The columns of the matrix **Z** span the null space of the active inequality constraints, and therefore **Z** is used to project the step direction **Δp** onto a subspace that is orthogonal to the active constraints (i.e., so that $\mathbf{A}\cdot\mathbf{\Delta p}=\mathbf{0}$ where **A** is the matrix of active constraint coefficients) [20].

The solution to the QP subproblem is obtained by inverting the matrix **G** to obtain the solution $\mathbf{\Delta p}=-\mathbf{Z}\cdot\mathbf{G}^{-1}\cdot\mathbf{f}$. The Levenberg-Marquardt algorithm ensures that the matrix **G** is positive-definite (and therefore invertible) by adding a diagonal matrix λ**I** to **H** prior to projection, where **I** is the identity matrix and λ is the so-called "damping parameter" that is used to penalize large steps [18,19]. (Note: The damping parameter is referred to as μ in reference [18].) As a result, the inverse of **G** can be stably and efficiently computed using its Cholesky factorization. Once the optimal step **Δp** has been calculated, the algorithm will attempt to compute the value of Φ at the corresponding point in parameter space (or at an intermediate point, if an inactive inequality constraint is encountered that forces the step to be truncated). This move is accepted if it results in an improved value of Φ. Otherwise, the value of λ will be increased and a new step will be computed.

Each time a step is accepted, the next iteration will commence from this new point in parameter space. If a full step was taken without hitting a constraint, the Lagrange multipliers of all active constraints will be computed to determine whether any non-binding constraints can be dropped from the active set. Otherwise, the blocking constraint will be added to the active set.

Several options can be adjusted to control the convergence of the optimization search algorithm. Selecting **Options→Optimization parameters** will display a dialog box where these parameters can be directly edited. The 'relative convergence tolerance' should be set to the maximum relative uncertainty that is acceptable in each element of the parameter vector (default value is 0.01, or 1%). This will determine the termination threshold based on size of **Δp** relative to the current parameter vector **p**. The user can also set the 'tau' parameter described in reference [18], which controls the initial size of the damping parameter (default value is 1×10^{-6}). If this value is set too large, the search algorithm will terminate prematurely because $\mathbf{\Delta p}$ will be over-damped during the first few iterations, and the algorithm will not have enough time to properly adapt the value of λ prior to achieving convergence. It is also possible to enhance the probability of obtaining a global optimum solution by performing a "multistart". Setting **Options→Use random initial guess for flux estimation** to be checked will randomize the initial flux and pool size values stored in the network model prior to optimization. The amount of random noise introduced to each parameter is determined by the option 'number of logs to perturb initial values' within the **Options→Optimization parameters** dialog box. This option specifies the maximum number of logarithms (i.e., powers of 10) that each parameter can be varied in either direction, up or down. The actual starting guesses will be uniformly sampled from this interval. Finally, setting the 'number of restarts' option to a value greater than 1 will repeat the flux estimation calculation the specified number of times. If the initial guesses are randomized, this is tantamount to a multistart global search algorithm with Levenberg-Marquardt local searches. (If the initial guesses are NOT randomized, this is a BIG waste of time because all local searches will arrive at the same result!) If INCA has been configured to run within a distributed computing environment, each restart will be performed as a parallel job (see Section 2 and Appendix E for details).

After clicking the **Flux estimation** button, the progress of the optimization search can be tracked from the MATLAB command window. After each iteration, the current SSR value ('Residual') is printed to the screen, along with the relative 'Step-size', the 'Directional derivative' $\mathbf{g}\cdot\mathbf{\Delta p}$, and the damping parameter value 'Lambda' from the previous step. If no blocking constraint was encountered, the 'Step-size' will be reported as '1'. Otherwise, the reported value represents the fraction of the optimal step taken prior to hitting a constraint.

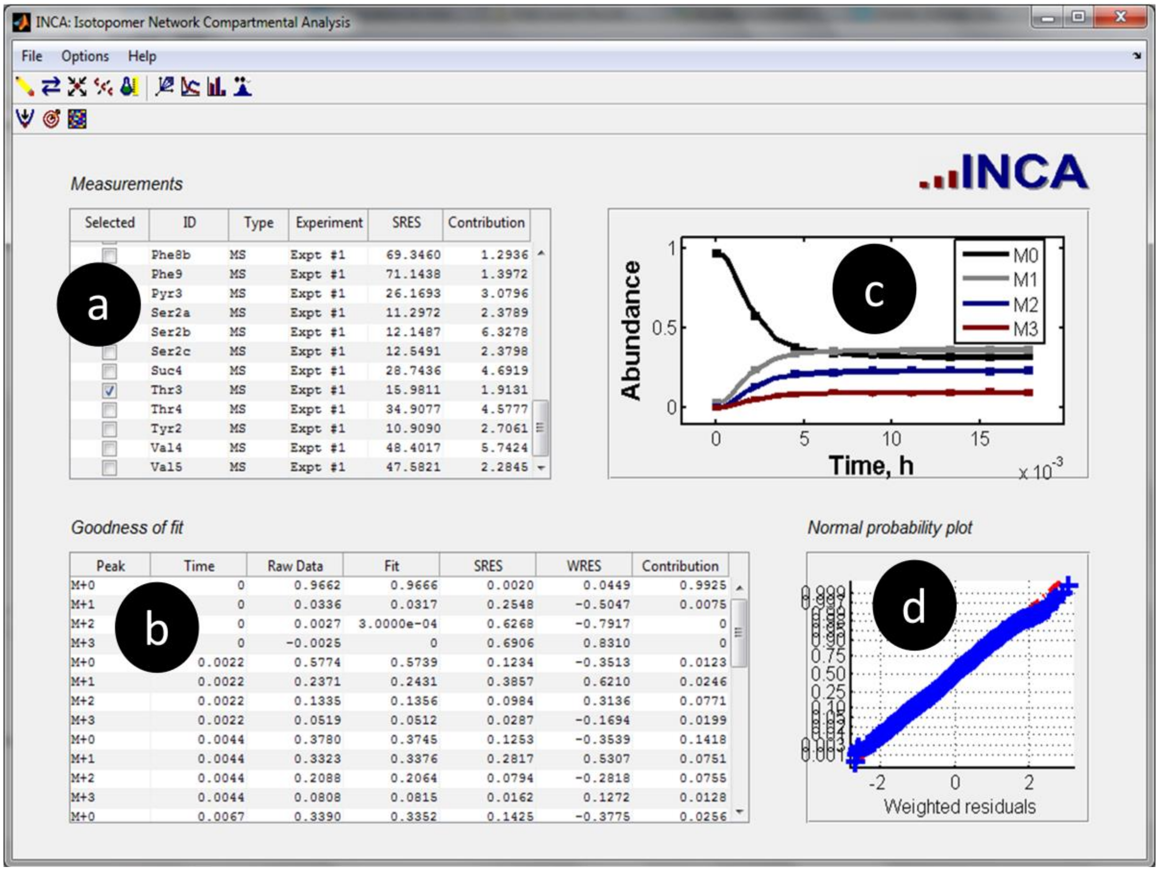
Upon termination of the search, control will return to the INCA GUI window and the flux estimation results will appear in the upper-left panel (Figure 9a) of the ***Optimize parameters*** screen. The drop-down menu above the table can be used to select a subset of parameters for viewing: either 'all fluxes', 'net fluxes', 'exchange fluxes', 'pool sizes', or 'MS norms' (i.e., scaling factors). The optimal 'Value' of each parameter will be displayed along with its standard error ('StdErr'), which is a local estimate of uncertainty obtained from the diagonal elements of the parameter covariance matrix evaluated at the optimal solution [12]. The parameter values shown in the table WILL NOT automatically replace the base values stored in the network model. However, clicking **Update Model** in Figure 9b will copy these values into the network model and overwrite the previously stored values. If flux estimation is subsequently repeated, the newly stored values will be used when initializing the optimization search.

9.2. Assessing goodness-of-fit

The flux estimation results are only meaningful if the model is able to achieve an acceptable fit to the experimental data. Otherwise, some inconsistency exists in the model or data that needs to be corrected before further analysis can be performed. When flux estimation has completed, INCA automatically provides several statistical metrics that can be used to assess goodness-of-fit. The text box at the bottom-left of the ***Optimize parameters*** screen (Figure 9b) provides a status message indicating whether the fit is accepted based on the SSR value obtained. This test assumes that the minimized variance-weighted SSR follows a chi-square distribution with *n-p* degrees of freedom (DOF), where *n* is the number of independent measurements and *p* is the number of fitted parameters. The expected SSR range is calculated as $\left[\chi^2_{\alpha/2}\left(n-p\right), \chi^2_{1-\alpha/2}\left(n-p\right)\right]$, where α is a threshold p-value at which the fit is rejected. (The program default is $\alpha = 0.001$.) The fit is 'accepted' if the actual SSR falls within the limits of the expected SSR and is 'rejected' if it does not.

In addition to performing a chi-square test on the total SSR, the distribution of the residuals should also be analyzed at convergence. The error-weighted residuals are expected to be normally distributed with zero mean. INCA applies the Lilliefors test to evaluate the normality assumption using MATLAB's `lillietest` function and the previously described α significance level. The outcome of the Lilliefors test, as well as the estimated mean and standard deviation of the residuals, is reported in the text box at bottom-left of the ***Optimize parameters*** screen (Figure 9b). It should be noted, however, that the Lilliefors test is fairly strict, and will frequently assess the residuals to be non-normally distributed because of an overabundance of small (i.e., near-zero) values. In this event, it is helpful to visualize the distribution of residuals by selecting the ***Assess goodness-of-fit*** icon 🎯 on the secondary toolbar. The bottom-right panel in this screen (Figure 10d) shows a ***Normal probability plot*** produced using the MATLAB `normplot` command. If the residuals are normally distributed, the plotted points should lie approximately on a straight line. An overabundance of small residuals will produce a sigmoid shape centered about zero. The near-zero residuals can be re-weighted by adjusting their associated measurement errors to improve the normality of the distribution. However, one should become especially concerned if the distribution is skewed to one side or if there are extreme outliers. This can indicate a more severe problem with the data regression.

The ***Assess goodness-of-fit screen*** also shows a table of all fitted measurements (Figure 10a), which includes the squared residual 'SRES' and 'Contribution' associated with each measurement. The 'Contribution' column contains the sum of all elements of the contribution matrix associated with a given measurement. The higher the 'Contribution' value, the more important the measurement is for determining the fitted parameters (see Antoniewicz et al. [12] for further details). Selecting any of the entries in the ***Measurements*** panel will produce a plot (Figure 10c) comparing the simulated measurements to the experimentally determined values. This provides a convenient way to visually inspect the lack-of-fit associated with each measurement. The ***Goodness-of-fit*** panel (Figure 10b) displays detailed information about the residuals associated with the currently selected measurement. This table can be used to further pinpoint individual measurements that are responsible for a poorly fitted model. The experimentally determined 'Raw Data' are listed along with the model-fitted values for each data point. The squared residual 'SRES', the weighted residual 'WRES', and the total 'Contribution' associated with each data point are also shown.



Measurements

Selected	ID	Type	Experiment	SRES	Contribution
□	Phe8b	MS	Expt #1	69.3460	1.2936
	Phe9	MS	Expt #1	71.1438	1.3972
	Pyr3	MS	Expt #1	26.1693	3.0796
	Ser2a	MS	Expt #1	11.2972	2.3789
	Ser2b	MS	Expt #1	12.1487	6.3278
□	Ser2c	MS	Expt #1	12.5491	2.3798
□	Suc4	MS	Expt #1	28.7436	4.6919
☑	Thr3	MS	Expt #1	15.9811	1.9131
□	Thr4	MS	Expt #1	34.9077	4.5777
□	Tyr2	MS	Expt #1	10.9090	2.7061
□	Val4	MS	Expt #1	48.4017	5.7424
□	Val5	MS	Expt #1	47.5821	2.2845

Goodness of fit

Peak	Time	Raw Data	Fit	SRES	WRES	Contribution
M+0	0	0.9662	0.9666	0.0020	0.0449	0.9925
M+1	0	0.0336	0.0317	0.2548	-0.5047	0.0075
M+2	0	0.0027	3.0000e-04	0.6268	-0.7917	0
M+3	0	-0.0025	0	0.6906	0.8310	0
M+0	0.0022	0.5774	0.5739	0.1234	-0.3513	0.0123
M+1	0.0022	0.2371	0.2431	0.3857	0.6210	0.0246
M+2	0.0022	0.1335	0.1356	0.0984	0.3136	0.0771
M+3	0.0022	0.0519	0.0512	0.0287	-0.1694	0.0199
M+0	0.0044	0.3780	0.3745	0.1253	-0.3539	0.1418
M+1	0.0044	0.3323	0.3376	0.2817	0.5307	0.0751
M+2	0.0044	0.2088	0.2064	0.0794	-0.2818	0.0755
M+3	0.0044	0.0808	0.0815	0.0162	0.1272	0.0128
M+0	0.0067	0.3390	0.3352	0.1425	-0.3775	0.0256

Figure 10. Assess goodness-of-fit screen (example shown is based on a transient isotopomer model).

If the flux estimation provides a poor fit, further investigation is needed to discover the root cause of the discrepancy. The ***Assess goodness-of-fit*** screen is designed to help the user quickly locate measurements that are responsible for the overall lack of fit and to diagnose the source of any mismatch between simulated and experimental measurements. In general, there are three possible causes for a poor fit that should be evaluated: (1) presence of gross measurement errors, (2) inappropriate weighting of the residuals, or (3) an error or omission in the metabolic reaction network. Process of elimination should be used to determine the root cause of a poor fit, and then corrective steps should be taken to obtain an acceptable fit before further analysis is performed.

9.3. Assessing parameter uncertainties

Not all flux parameters can be estimated with equal precision when solving the inverse problem by least-squares regression. Therefore, it is imperative to calculate confidence intervals or standard errors that convey the uncertainties associated with all estimated parameters. The standard errors displayed in the 'StdErr' column of the ***Optimize parameters*** screen (Figure 9a) are merely local estimates of uncertainty obtained from the diagonal elements of the parameter covariance matrix [12]. These values do not account for the presence of nonlinearities or inequality constraints in the parameter regression. To obtain a more accurate picture of parameter uncertainties, it is necessary to perform either **Parameter Continuation** or **Monte Carlo Analysis** using the appropriate buttons in Figure 9b. Once either of these calculations is completed, the entries in the 'LB' and 'UB' columns will be updated to show the computed lower and upper bounds of the 95% confidence interval for each fitted parameter. (The confidence level can be adjusted from the default α=0.05 to any other value by selecting **Options→Confidence interval parameters** and editing the 'Confidence level' entry in the dialog box that appears.) Selecting any parameter in the table will show a plot of the confidence region in the panel at the lower-right of the screen (Figure 9c).

Parameter Continuation is more efficient that **Monte Carlo Analysis** for typical models. This calculation involves varying each adjustable parameter, one-at-a-time, to determine the sensitivity of the minimized SSR value Φ to the parameter in question [12]. In many cases, it is also desirable to calculate confidence intervals with respect to a "derived" parameter $k_i = \mathbf{L}_i \cdot \mathbf{p}$ (e.g., a net or exchange flux) that can be expressed as a linear combination of the "free" parameters contained in **p** using the transformation matrix $\mathbf{L}_i$. Each parameter is varied upward or downward until a threshold $\Delta\Phi$ value is reached that corresponds to a chi-square distribution with one DOF, evaluated at a cumulative probability of 1–α (e.g., the $\Delta\Phi$ threshold is given by $\chi^2_{0.95}(1) = 3.84$ when α=0.05). The interval over which the parameter can be varied while staying below this $\Delta\Phi$ threshold defines the desired confidence region.

Parameter continuation is performed using a predictor-corrector method that is loosely based on the path following algorithm described by Seydel [21] (Chapter 4). The predictor step involves moving along a vector $\Delta\mathbf{p}$ in parameter space that will produce a desired change in Φ (let's call it $\Delta\Phi_{\text{pred}}$) by varying the parameter of interest k_i. However, the gradient of Φ with respect to all other parameters (or combinations of parameters orthogonal to the row vector $\mathbf{L}_i$) should be held at zero to maintain optimality with respect to those other dimensions. Therefore, the step $\mathbf{\Delta p}$ should be chosen such that the change in the objective function gradient $\mathbf{\Delta g} = \mathbf{H} \cdot \mathbf{\Delta p}$ is parallel to $\mathbf{L}_i^{\mathbf{T}}$. If there are no active inequality constraints and **H** is invertible, the predictor step direction can be calculated as $\mathbf{\Delta p} = \mathbf{H}^{\mathbf{-1}}\mathbf{L}_i^{\mathbf{T}}$. In the more general case that allows for active constraints and near-singular **H** matrices , the procedure of Section 9.1 can be followed to compute $\mathbf{\Delta p} = -\mathbf{Z} \cdot \mathbf{G}^{\mathbf{-1}} \cdot \mathbf{f}$, where **Z** is a matrix whose columns span the null space of the active inequality constraints, $\mathbf{G} = \mathbf{Z}^{\mathbf{T}} \cdot (\mathbf{H} + \lambda \mathbf{I}) \cdot \mathbf{Z}$ is the projected "damped" Hessian matrix, and **f** is the projected coefficient vector $\mathbf{Z}^{\mathbf{T}} \cdot \mathbf{L}_i^{\mathbf{T}}$. Once the step direction has been computed, the step length is adjusted to achieve the desired value of $\Delta\Phi_{\text{pred}}$. This is accomplished using an approach similar to Antoniewicz et al. [12], which involves solving a quadratic equation for the step length *h* such that

$$\Delta\Phi_{\text{pred}} = a_1 \cdot h + a_2 \cdot h^2$$

$$\text{where } a_1 = 2 \cdot \mathbf{\Delta p}^{\mathbf{T}} \cdot \mathbf{g} \text{ and } a_2 = \mathbf{\Delta p}^{\mathbf{T}} \cdot \mathbf{H} \cdot \mathbf{\Delta p}$$

The value of $\Delta\Phi_{\text{pred}}$ is controlled by specifying the 'Desired number of continuation steps' within the dialog box that appears after selecting **Options→Confidence interval parameters.** With a $\Delta\Phi$ threshold value of $\chi^2_{1-\alpha}(1)$, the $\Delta\Phi_{\text{pred}}$ on each step should be approximately $\chi^2_{1-\alpha}(1)/n$ to complete the continuation in *n* steps. The default value of *n* is 5, but this can be adjusted up or down to achieve smaller or larger steps, respectively.

Once the predictor step direction and step size have been computed, the continuation algorithm will calculate the actual objective function change $\Delta\Phi_{\text{actual}}$ after adding $h \cdot \mathbf{\Delta p}$ to the previous parameter vector. The predictor error is computed as the difference $E_{\text{pred}} = \Delta\Phi_{\text{actual}} - \Delta\Phi_{\text{pred}}$. If the relative predictor error $\varepsilon_{\text{pred}} = E_{\text{pred}} / \Delta\Phi_{\text{pred}}$ is larger than a specified tolerance ε_{max}, the step is rejected and a new step will be computed by increasing the damping parameter λ and recalculating $\mathbf{\Delta p}$ and *h*. The relative error tolerance ε_{max} has a default value of one, which can be adjusted by selecting **Options→Confidence interval parameters** and specifying a new value for 'Predictor step relative error tolerance' in the dialog box that appears.

If the value of $|E_{\text{pred}}|$ is very small (e.g., $\leq 10^{-5}$), it is likely that the predictor step gave an accurate estimate of the optimal $\Delta\Phi$ and no further correction is necessary. However, if E_{pred} is nontrivial but still less than $\varepsilon_{\text{max}} \cdot \Delta\Phi_{\text{pred}}$, a corrector step will be used to minimize Φ while holding $k_i = \mathbf{L}_i \cdot \mathbf{p}$ constant at its current value. This is accomplished using the same Levenberg-Marquardt algorithm described in Section 9.1, but with an additional equality constraint to hold k_i fixed. Inclusion of the corrector step allows for larger steps to be taken at each iteration of the algorithm, because errors introduced by the predictor step can be subsequently removed. This is in contrast with the algorithm of Antoniewicz et al. [12], which does not have a corrector step and therefore must adjust the step size at each iteration to keep $|E_{\text{pred}}|$ at a low value of approximately 10^{-4}. Once the corrector step has completed, the predictor-corrector iterations will continue until the $\Delta\Phi$ threshold of $\chi^2_{1-\alpha}(1)$ is achieved. In this way, the continuation algorithm will systematically trace out the envelope of minimized $\Delta\Phi$ values as a function of the parameter k_i, in order to determine its confidence interval bounds.

When the **Parameter Continuation** button is clicked, confidence intervals will be calculated for all parameters that have a check in their 'Vary' column of Figure 9a. Confidence intervals can be computed for a subset of parameters by unchecking some of these boxes. (All visible parameters can be checked or unchecked as a group by toggling the **Vary all** button.) The algorithm first searches for fully coupled parameters to avoid unnecessary calculations. Then, a single parameter continuation job will be performed for each fully coupled set. If INCA has been configured to run within a distributed computing environment, each continuation will be performed as a parallel job (see Section 2 and Appendix E for details). A progress bar will appear to show the fraction of jobs that have completed, and its **Cancel** button can be used to terminate the continuation procedure manually.

The progress of each parameter continuation is printed to the MATLAB command window (unless redirected to a file). The displayed variables are: the iteration counter 'Iteration', the total change in SSR value 'Delta residual', the total change in parameter value 'Delta parameter', the change in parameter value on the current step 'Step-size', the 'Predictor error' E_{pred}, the improvement in SSR obtained during the corrector step 'Corrector adjustment', the number of Levenberg-Marquardt 'Corrector iterations' performed during each iteration, the number of failed predictor steps 'Failed steps', the ratio of the actual step taken to the optimal h value obtained from the predictor calculation 'h/hopt' (Note: this will be <1 whenever the predictor step hits a constraint), and the damping parameter value 'Lambda' used in the predictor step.

Once all the parameter continuation jobs have completed, control will return to the INCA GUI window and the 'LB' and 'UB' columns will be updated with the confidence interval bounds. If an improved optimal solution was encountered during the parameter continuation, a dialog box will appear that will prompt the user to save the alternative flux map. This can occur if the prior **Flux Estimation** procedure did not achieve a global best-fit solution.

Selecting any parameter in Figure 9a will show a sensitivity plot in the lower-right panel marked ***Confidence interval plot*** (Figure 9c). The dashed line represents the local approximation of $\Delta\Phi$ based on the parameter covariance matrix evaluated at the optimal solution. The blue line represents the accurate $\Delta\Phi$ values determined by parameter continuation. Small sensitivities (i.e., small changes in the minimized SSR in response to large changes in a parameter value) indicate that the parameter cannot be estimated precisely. On the other hand, large sensitivities indicate that the flux is well determined, as indicated by a steep curve. Fixed parameters show a straight vertical line in the confidence interval plot, since these parameters cannot vary. A horizontal dotted line shows the $\Delta\Phi$ threshold that was used to determine the confidence interval bounds (i.e., where the blue line crosses the dotted line).

Any parameters whose confidence interval calculation did not complete normally will have an upper bound or lower bound of 'NaN'. Selecting these parameters for plotting in Figure 9c will reveal the progress of the continuation prior to termination. In some cases, it will be clear that the parameter in question is practically unidentifiable based on the confidence interval plot. In other cases, it may be necessary to repeat the confidence interval calculation after adjusting the settings in **Options→Confidence interval parameters** to improve convergence. In particular, increasing the 'Predictor step relative error tolerance' is sometimes necessary if the command window outputs indicate that slow progress was caused by too many failed predictor steps. In preparation for this, the program will automatically uncheck the 'Vary' box for any parameter whose confidence interval calculation completed successfully, but will leave this box checked for all parameters that still contain 'NaN' values in their 'LB' or 'UB' columns.

INCA is also capable of computing confidence intervals using Monte Carlo analysis, as an alternative to parameter continuation. Pressing the **Monte Carlo Analysis** button in Figure 9b will launch a series of Monte Carlo simulations using the approach described by Press et al. [22] (Chapter 15.6). The optimal parameter vector is used to generate many synthetic data sets, which are then used to re-estimate the model parameters. This involves simulating values for all measurements assuming that the optimal fitted parameters are "true". Then, by adding Gaussian noise to each simulated measurement based on their experimental error, the program will generate many replicate data sets. Re-fitting fluxes to each data set provides a probability distribution for each parameter that can be used to compute confidence intervals based on the spread of the distribution. This can be a computationally demanding approach, since the program must continue generating new synthetic data sets and re-fitting until the chosen uncertainty measure stabilizes. Therefore, many rounds of Monte Carlo analysis are required before the "tail" probabilities of interest finally converge.

Convergence is monitored by performing Monte Carlo runs in incremental batches of 100 synthetic data sets and recalculating the confidence intervals after each batch. The calculation continues until the maximum relative change in any confidence interval bound is less than 5×10^{-4} for three consecutive batches. This implies that the confidence interval bounds are correct to three significant figures. The 'Error Norm' at each iteration can be tracked from the MATLAB command window. Also, a progress bar will show the fraction of the next batch that has already completed. Pressing the **Cancel** button on this progress bar will terminate the Monte Carlo analysis early. If INCA has been configured to run within a distributed computing environment, each data set within a given batch will be regressed as a parallel job (see Section 2 and Appendix E for details).

Once convergence is achieved, control will return to the INCA GUI window and the values in 'LB' and 'UB' will be updated with the Monte Carlo results. Note that any prior results in these columns will be overwritten. Because Monte Carlo is able to compute confidence intervals on all parameters simultaneously, every parameter will be updated regardless of whether its 'Vary' box is checked. If Monte Carlo results are available, selecting a given parameter will show a frequency histogram of the values obtained during the analysis, with vertical dotted lines denoting the estimated lower and upper confidence interval bounds.

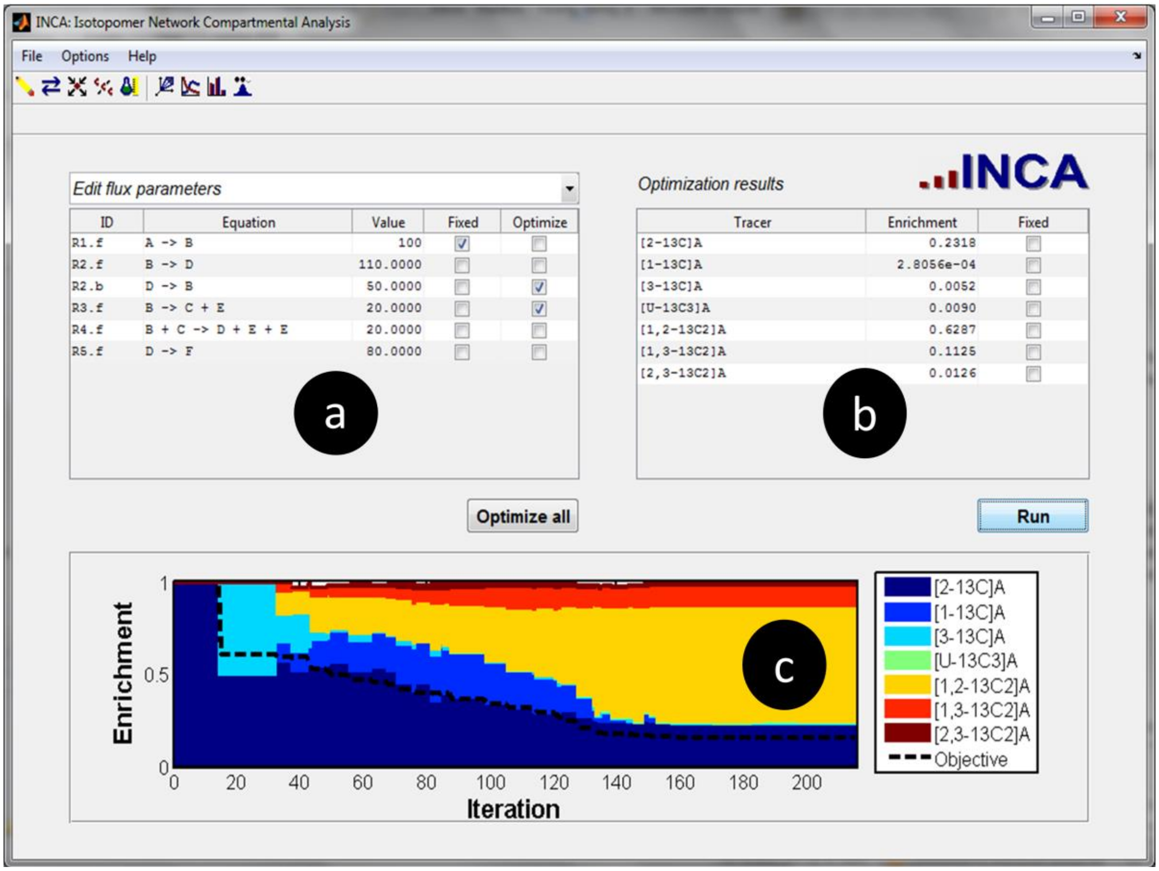
9.4. Identifiability analysis

After the flux estimation completes, the local sensitivity information (e.g., Hessian, Jacobian, etc.) available at the best-fit solution is used by INCA to generate the identifiability matrices described in Section 8.3. These can be viewed interactively by clicking the ▦ icon on the secondary toolbar (see Section 8.3 for details). Inspection of the identifiability results can provide useful insights into the flux estimation procedure, as discussed by Antoniewicz et al. [12].

10. Tracer Optimization

The ***Tracer optimization*** feature applies the method of Mollney et al. [15] to search for tracer combinations that minimize either the A- or D-optimality criterion. Note that this feature was added to the program recently, and it should be treated as "experimental" until more extensive testing has been completed.

Selecting the icon from the primary toolbar will display the ***Tracer optimization*** screen (Figure 11). In the upper-left panel (Figure 11a), the drop-down menu can be used to switch between tables where flux and node properties are displayed for editing. As described in Section 8.1, the **Value** and **Fixed** columns in the ***Edit flux parameters*** table and the **Balanced** column in the ***Edit node parameters*** table can be edited to set the base values of these network properties. The ***Edit node parameters*** table can also be used to enter initial **Pool size** estimates when performing transient isotopomer simulations (**INST-MFA only**).



ID	Equation	Value	Fixed	Optimize
R1.f	A -> B	100	☑	□
R2.f	B -> D	110.0000	□	□
R2.b	D -> B	50.0000	□	☑
R3.f	B -> C + E	20.0000	□	☑
R4.f	B + C -> D + E + E	20.0000	□	□
R5.f	D -> F	80.0000	□	□

Tracer	Enrichment	Fixed
[2-13C]A	0.2318	□
[1-13C]A	2.8056e-04	□
[3-13C]A	0.0052	□
[U-13C3]A	0.0090	□
[1,2-13C2]A	0.6287	□
[1,3-13C2]A	0.1125	□
[2,3-13C2]A	0.0126	□

Figure 11. Tracer optimization screen.

All previously entered tracers within the **Active** experiments on the ***Edit experiments*** screen will be listed in the ***Optimization results*** panel (Figure 11b) at the upper-right of the ***Tracer optimization*** screen, along with the initial

Enrichment values entered previously (see Section 6.4). If multiple experiments are activated, the tracers associated with each parallel experiment will populate the table as separate row entries. The initial values in the **Enrichment** column can be edited directly on the ***Tracer optimization*** screen, but note that any changes will also propagate to the ***Edit experiments*** screen. Checking the **Fixed** box for any listed tracer will hold its enrichment constant during the optimization, so that its enrichment will not be adjusted away from the initial value shown in the table.

Once the initial values in Figures 11a and 11b have been set, clicking the **Run** button will launch the optimization search. The program applies a custom Nelder-Mead simplex algorithm called `amoeba`, which is based on MATLAB's built-in `fminsearch` function, to minimize the specified A- or D-optimality criterion. The progress of the computation is displayed to the MATLAB command window: the iteration counter 'Iteration', the number of objective function evaluations 'Func-count', the minimum objective function value in the current simplex 'min f(x)', and the 'Procedure' used to update the simplex are output to the screen at the end of each iteration. A progress bar is also shown that indicates the fractional convergence of the algorithm based on its default termination tolerances: all enrichments must converge to within 1% (absolute) and the objective function value must converge to within 0.1% (relative). Pressing the **Cancel** button on the progress bar at any time will terminate the search immediately.

Several options can be adjusted to control the ***Tracer optimization*** search. Selecting **Options→Experiment design parameters** will display a dialog box where the user can specify the optimality criterion (A or D) to be used in the search. The A-optimality criterion will minimize the trace of the covariance matrix, whereas the D-optimality criterion will minimize its determinant. (The objective function values are computed by $\sqrt{\frac{1}{n}\operatorname{trace}(C)}$ in the case of A-optimality and $\sqrt[2n]{\det(C)}$ in the case of D-optimality, where *C* is the parameter covariance matrix and *n* is the number of parameters included in *C*.) The parameters included in the covariance matrix *C* can be chosen interactively using the checkboxes in the **Optimize** column of Figure 11a. This is equivalent to applying an A_L- or D_L-criterion with the transformation matrix **L** chosen to exclusively select the checked parameters. In the case of a steady-state isotopomer model, only flux parameters can be optimized, whereas both fluxes and pool sizes can be optimized in the case of a transient model. When the ***Tracer optimization*** screen is first displayed, only the "free" model parameters are initially selected in the **Optimize** column (i.e., independent parameters that are not fixed by the user or determined by other parameters due to mass balance constraints). For convenience, there is an **Optimize all** button that enables the user to cycle between three default states: (i) all parameters selected, (ii) no parameters selected, or (iii) only free parameters selected.

The dialog box that appears after selecting **Options→Experiment design parameters** also allows the user to specify whether to reinitialize the starting guess. If 'Y' is entered, the program will initialize the starting simplex such that each vertex corresponds to a pure tracer at its maximum enrichment, plus one additional vertex at the origin with all zero enrichments. This will begin the search using the largest possible simplex. On the other hand, entering 'N' will initialize the starting simplex to include the combination shown in Figure 11b, plus other nearby points (10% maximum deviation in each component). This will perform a local search starting from the previously entered enrichment values.

When the search terminates, a status message will be printed to the MATLAB command line and control will return to the INCA GUI window. The optimal enrichment values identified during the search will overwrite the previous values listed in the **Enrichment** column of the ***Optimization results*** panel (Figure 11b). Also, the progress of the search will be displayed to the figure window at the bottom of the screen (Figure 11c). The plot will show the optimal tracer combination (colored areas) as well as the relative objective function value (dotted line) encountered at each iteration. It is very important to note that all previously entered tracer enrichments for the active experiments will be overwritten when the search terminates, so make sure that you have saved the model file before attempting to perform ***Tracer optimization***. Once a new tracer combination has been identified, it can be subjected to more rigorous testing by performing ***Tracer simulation*** to generate simulated labeling data, followed by parameter estimation, confidence interval calculation, and identifiability analysis of the simulated dataset within the ***Flux estimation*** screen (see Sections 8 and 9).

Bibliography

1. Antoniewicz MR, Kelleher JK, Stephanopoulos G (2007) Elementary metabolite units (EMU): a novel framework for modeling isotopic distributions. Metab Eng 9: 68-86.
2. Young JD, Walther JL, Antoniewicz MR, Yoo H, Stephanopoulos G (2008) An elementary metabolite unit (EMU) based method of isotopically nonstationary flux analysis. Biotechnol Bioeng 99: 686-699.
3. Palsson B (2006) Systems biology : properties of reconstructed networks. Cambridge ; New York: Cambridge University Press. xii, 322 p. p.
4. Stephanopoulos G, Aristidou AA, Nielsen JH (1998) Metabolic engineering : principles and methodologies. San Diego: Academic Press. xxi, 725 p. p.
5. Fernandez CA, Des Rosiers C, Previs SF, David F, Brunengraber H (1996) Correction of 13C mass isotopomer distributions for natural stable isotope abundance. J Mass Spectrom 31: 255-262.
6. Price ND, Reed JL, Palsson BO (2004) Genome-scale models of microbial cells: evaluating the consequences of constraints. Nat Rev Microbiol 2: 886-897.
7. Kauffman KJ, Prakash P, Edwards JS (2003) Advances in flux balance analysis. Curr Opin Biotechnol 14: 491-496.
8. Segre 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.
9. Mahadevan R, Schilling CH (2003) The effects of alternate optimal solutions in constraint-based genome-scale metabolic models. Metab Eng 5: 264-276.
10. Burgard AP, Nikolaev EV, Schilling CH, Maranas CD (2004) Flux coupling analysis of genome-scale metabolic network reconstructions. Genome Res 14: 301-312.
11. Edwards JS, Ibarra RU, Palsson BO (2001) In silico predictions of Escherichia coli metabolic capabilities are consistent with experimental data. Nat Biotechnol 19: 125-130.
12. Antoniewicz MR, Kelleher JK, Stephanopoulos G (2006) Determination of confidence intervals of metabolic fluxes estimated from stable isotope measurements. Metab Eng 8: 324-337.
13. Wiechert W (2001) ^{13}C Metabolic Flux Analysis. Metab Eng 3: 195-206.
14. Sauer U (2006) Metabolic networks in motion: ^{13}C-based flux analysis. Mol Syst Biol 2: 62.
15. Mollney M, Wiechert W, Kownatzki D, de Graaf AA (1999) Bidirectional reaction steps in metabolic networks: IV. Optimal design of isotopomer labeling experiments. Biotechnol Bioeng 66: 86-103.
16. Wiechert W, Wurzel M (2001) Metabolic isotopomer labeling systems - Part I: global dynamic behavior. Mathematical Biosciences 169: 173-205.
17. Wiechert W, Mollney M, Isermann N, Wurzel M, de Graaf AA (1999) Bidirectional reaction steps in metabolic networks: III. Explicit solution and analysis of isotopomer labeling systems. Biotechnol Bioeng 66: 69-85.
18. Madsen K, Nielsen HB, Tingleff O (2004) Methods for Non-linear Least Squares Problems, 2nd Edition. Technical University of Denmark.

19. Dennis JE, Schnabel RB (1983) Numerical methods for unconstrained optimization and nonlinear equations. Englewood Cliffs, N.J.: Prentice-Hall. xiii, 378 p. p.
20. Gill PE, Murray W, Wright MH (1981) Practical Optimization. New York: Academic Press.
21. Seydel R (1988) From equilibrium to chaos : practical bifurcation and stability analysis. New York: Elsevier. xv, 367 p. p.
22. Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical Recipes in C: The Art of Scientific Computing. Cambridge, UK: Cambridge University Press.
23. Kelleher JK, Masterson TM (1992) Model equations for condensation biosynthesis using stable isotopes and radioisotopes. Am J Physiol 262: E118-125.
24. Antoniewicz MR, Kraynie DF, Laffend LA, Gonzalez-Lergier J, Kelleher JK, et al. (2007) Metabolic flux analysis in a nonstationary system: fed-batch fermentation of a high yielding strain of E. coli producing 1,3-propanediol. Metab Eng 9: 277-292.
25. Murphy TA, Dang CV, Young JD (2013) Isotopically nonstationary 13C flux analysis of Myc-induced metabolic reprogramming in B-cells. Metab Eng 15: 206-217.

Appendix

Appendix A: Directory tree

The INCA program files are arranged as shown in the attached tree diagram. The startup files are located in the INCA root directory, and the remaining files are grouped into subdirectories as follows:

- cast: Functions for converting one object into another or transferring data between objects.
- class: Class definitions and methods.
- core: Core computational routines used to solve steady-state and transient EMU balance equations.
- driver: High-level driver functions for tracer simulation, flux estimation, parameter continuation, Monte Carlo analysis, and optimal experiment design.
- fluxtools: Helper functions for analyzing steady-state flux balances.
- gui: Functions that control the graphical user interface (GUI).
- idtools: Helper functions for parameter optimization, continuation, and identifiability analysis.
- mstools: Helper functions for processing, correcting, and simulating mass isotopomer distributions.
- parallel: Functions that enable parallel processing features.
- util: Low-level utility functions.
- vis: Plotting and data visualization functions.

In addition to these program files, INCA program documentation (including this user manual) is located in the ‘doc’ folder. Several example files can be found in the ‘demo’ folder (see Appendix F for details).

Appendix B: Class diagrams

INCA data objects can be created from the command line using the class constructor methods in the ‘class’ subdirectory. Alternatively, INCA data objects can be loaded from a previously saved ‘.mat’ file generated by the GUI. Each INCA object contains several property fields, comprised of either built-in MATLAB data types or other INCA data objects defined within the ‘class’ subdirectory. There are three top-level objects used by INCA to store program data, which are described further below. The attached diagrams show the tree structure of each top-level class. (The name of each property field is given with its class name in brackets. INCA objects are shown in boxes, and built-in MATLAB data types are unboxed.)

‘model’ objects contain all information needed to define a single network model, as well as its associated experimental datasets.

‘simdata’ objects contain the results of a tracer simulation.

’fitdata’ objects contain the results of a flux estimation (i.e., a flux map).

Appendix C: Function index

All INCA functions can be called from the MATLAB command line, using the previously described data objects and/or built-in MATLAB data types as inputs and outputs. Typing ‘help’ followed by the function name at the command prompt will return detailed function documentation. Furthermore, an HTML index is provided with function descriptions and cross-references. It can be accessed through the link above or by locating index.html within the ‘doc’ folder and opening it within a web browser. (Hint: Start by inspecting the functions contained in the ‘driver’ folder, as they operate directly on the data objects described in Appendix B and can be used to perform several high-level INCA computations directly from the MATLAB command line.)

Appendix D: Handling compartmentalized pools and dilution fluxes

A common issue that arises when modeling isotope tracer experiments is that a cell or tissue sample can contain multiple pools of the same metabolite that differ in their isotopic labeling. When the pools are extracted from the sample for measurement, they will become aggregated. As a result, the isotopomer measurements will reflect the labeling of the mixed pool, rather than the separate contributions from individual compartments. In order to correctly describe these measurements, we need to introduce “pseudofluxes” into our network model to account for pool mixing. For example,

Let A.c = cytoplasmic pool
Let A.m = mitochondrial pool
Let A.s = sampled pool (mixture of A.c and A.m)
Let Sink = an arbitrary network sink node

We can include the following pseudofluxes to represent mixing of the two compartmentalized pools:

0*A.c (abc…) -> A.s (abc…)
0*A.p (abc…) -> A.s (abc…)
A.s -> Sink

The zero coefficients on ‘A.c’ and ‘A.p’ ensure that the pseudofluxes do not impact the mass balances and therefore will not alter the “true” fluxes in the biochemical network. This is important because the pseudofluxes do not exist *in vivo* but only arise due to sampling of the system. In essence, we are creating the sampled pool

'A.s' out of "thin air" in each of the first two reactions, but with a labeling pattern that matches 'A.c' in the first reaction or 'A.m' in the second reaction. By adjusting the relative contributions of these two reactions, the program will attempt to match the labeling measurements on 'A.s'. Therefore, when entering MS measurements for metabolite 'A' on the ***Edit Experiments*** screen, we will specify 'A.s' as the **Node ID**.

When including pseudofluxes into our model to account for pool mixing, we are only interested in the relative contributions from each compartment. The absolute rates are meaningless. Therefore, it is convenient to "fix" the total flux 'A.s -> Sink' to a value of '100' by selecting the reaction on the ***Edit reactions*** screen and editing the appropriate entries in the ***Edit flux properties of selected reactions*** panel (see Section 5.2). In doing so, the fluxes estimated for the first and second reactions will represent the percentage contribution of each compartment to the sampled pool.

It is also possible to introduce *G* dilution parameters into isotopomer models using pseudofluxes. These parameters were originally introduced as part of the Isotopomer Spectral Analysis (ISA) framework to account for lack of isotopic steady-state in biosynthetic products derived from condensation reactions [23]. (The so-called *D* parameters used by ISA, on the other hand, do not require pseudofluxes because they reflect true dilution of the tracer before it enters the system. This type of dilution can be modeled simply by including two reactions that produce the labeled substrate: one that derives from the tracer and another that derives from an unlabeled source.) There have been several examples where *G* parameters have been introduced to account for slowly labeled (i.e., "cold") pools in MFA models [24,25]. This can be accomplished in a manner similar to that described above for the case of compartmentalization.

Let A = true pool (labeled node involved in other network reactions)
Let A.u = cold pool (unlabeled source)
Let A.s = sampled pool (mixture of A and A.u)
Let Sink = an arbitrary network sink node

The pseudofluxes represent mixing of the labeled and unlabeled pools of 'A':

0*A (abc…) -> A.s (abc…)
0*A.u (abc…) -> A.s (abc…)
A.s -> Sink

In this case, it is not necessary to use a zero coefficient for 'A.u' because it is not a balanced node. However, including it does no harm. Selecting 'A.s' as the **Node ID** of all MS measurements involving 'A' and fixing the flux of 'A.s -> Sink' to '1' will enable the model to estimate the fractional contributions from the labeled and unlabeled pools of 'A'. The contribution from the true (labeled) pool is equivalent to the *G* parameter, while the contribution from the cold (unlabeled) pool is 1-*G*.

There is one additional constraint that must be set when mixing or dilutional pseudofluxes are introduced to transient isotopomer models. We must set the **Pool size** of 'A.s' to '0' on the ***Edit nodes*** screen to ensure that the sampled pool is at quasi-steady state relative to the compartmentalized pools (**INST-MFA only).** This will force the residence time of the sampled pool to be zero, so that its labeling will track its inputs instantaneously. Without this additional constraint, there will be an artificial time constant associated with pool mixing that does not have physical meaning.

One final type of dilution that can be introduced to the model, which does not involve pseudofluxes, can be used to represent cases where there is reversible exchange of a labeled metabolite with a large, unlabeled pool but without net flux in either direction. For example, this might be used to account for dilution of a free amino acid pool due to protein turnover in cases where there is no net synthesis or degradation of protein. Unlike the previous case, this represents a true dilution of the labeled pool that will propagate through the network to other downstream metabolites. The *G* parameter, on the other hand, will only dilute the isotopomer measurements of a single metabolite (e.g., 'A' in the previous case) without directly impacting the labeling of other metabolites in the network.

We can represent the previous example of amino acid dilution due to protein turnover as follows:

Let Ala = free alanine pool (balanced)
Let Ala.p = protein-bound alanine (unbalanced)
Let dummy = dummy metabolite not involved in other reactions (balanced)

The dilution flux allows for reversible exchange between Ala and the unlabeled Ala.p pool:

Ala (abc…) <-> Ala.p (abc…) + dummy

We will mark Ala.p as unbalanced by unchecking the appropriate box in the **Balanced** column on the ***Edit nodes*** screen. This will force Ala.p to remain unlabeled but will also result in the net flux of the reaction becoming unconstrained. Therefore, we include a balanced 'dummy' metabolite to force the net flux to zero. Since this reaction is the only one that involves 'dummy', its net flux must be zero to preserve mass balance. In the more general case where we want to allow both reversible exchange AND net flux to/from the unlabeled pool, we can remove the dummy variable from the reaction equation. This will provide an additional point where material can enter or leave the network, which should typically be associated with a net flux measurement. (Otherwise, large uncertainties can result due to unconstrained flux to/from the unbalanced node.)

Appendix E: Parallelization

INCA has been natively configured to take advantage of distributed computing environments by parallelizing certain long-running processes: (i) flux estimation (`estimate` function), (ii) parameter continuation (`continuate` function), and (iii) Monte Carlo analysis (`montecarlo` function). These functions can be launched from the INCA GUI (running on the frontend node) using the **Estimate Fluxes**, **Parameter Continuation**, and **Monte Carlo Analysis** buttons found on the ***Optimize parameters*** screen that first appears when selecting the ***Flux estimation*** icon 📊. As described in Section 2, there are two approaches for running parallel computations with INCA. The first approach relies on the Condor job scheduler and the MATLAB Compiler Runtime (MCR) library to distribute jobs from the frontend node to compute nodes. This has been successfully tested within a 32-node Rocks 6.0 cluster (www.rocksclusters.org) running Condor 6.1 and MATLAB R2012a on the frontend node. The second approach relies on the MATLAB Parallel Computing toolbox and Distributed Computing Server.

In order to use the first approach, **Options →Run in parallel using Condor** must be checked on the main menu of the INCA GUI. Also, the full file location of the previously compiled INCA `serve` function and the previously installed MCR library must be specified using the dialog box that appears after selecting **Options →Condor settings**. The `serve` function (found in the INCA 'parallel' subdirectory) must be compiled on the frontend node using the MATLAB compiler command 'mcc -m -R -nojvm -R -nodisplay -v serve.m' and placed in a common file location where all compute nodes can access it. The MCR library must also be downloaded and installed to a location that is accessible by all compute nodes.

Once these files and options have been set, running one of the parallelizable functions on the frontend node will generate a batch of input files within the current MATLAB working directory named 'processin_#.mat', where '#' is a unique number associated with each job. The jobs will then be distributed to the compute nodes by generating a Condor submit description file called 'mat.condor' and submitting it to Condor with the command 'condor_submit'. (Before calling parallel functions, the user should change the working directory to a writable location where no other INCA parallel sessions are running. Otherwise, crosstalk could occur between the sessions. Also, the directory name should not contain spaces or errors will result when calling the `serve` function.)

The jobs will be queued and processed in order of their job number as free compute nodes become available. As each job file is processed, a text file named 'diary_#.txt' will be created to store any command line outputs returned by the compute node while running job 'processin_#.mat'. This file can be opened within a text editor while the job is still in process to monitor the progress of the calculation. Upon completion, an output file named 'processout_#.mat' will be written to the same directory where the input files are stored. This file will be read by the frontend node and then deleted. A list of completed job numbers will be updated on the frontend MATLAB command window as each output file is read. When all output files have been read, control will return to the INCA GUI window and the calculation results will be available for viewing. Two files called 'mat.err' and 'mat.log' will also be written to the MATLAB working directory that contain error messages and status messages, respectively, generated by Condor during the run. Refer to the Condor project homepage (http://www.cs.wisc.edu/condor/) for further details about Condor program operation and user commands.

The second approach for running parallel computations with INCA involves starting a pool of worker processes using the MATLAB Distributed Computing Server and Parallel Computing toolbox. The parallelizable functions in INCA have been coded with `parfor` loops, which will automatically distribute jobs to workers when they are available. No INCA options or program files need to be adjusted to use this approach. However, it has not been tested by the authors and therefore cannot be verified to work as expected. Please refer to the Parallel Computing toolbox documentation for more information on how to start worker pools and connect to them from a MATLAB client.

Appendix F: Examples

Two subdirectories are provided under the 'demo' folder with examples of INCA models that can be used to explore program features and syntax. Both examples contain a '.m' script file that can be called from the MATLAB command line to generate an INCA model object. Alternatively, the model can be loaded from the INCA GUI by opening the '.mat' file contained in the same folder.

- The 'simple' folder contains a small steady-state example based on the toy network examined by Antoniewicz et al. [1,12].
- The 'ecoli' folder contains a larger nonstationary example based on the *E. coli* network examined by Young et al. [2]. (Because this model contains several data points, it can take up to a minute to load. Please be patient!)

Copyright Notice

The INCA graphical user interface depends on the `findjobj` utility function, which is subject to the copyright and license terms shown below.

Changelog

Version 1.0
Creation date: 2013/9/1

+++

Version 1.1
Creation date: 2014/2/13

Fixed a bug in `plotms.m` that produced an occasional error when plotting MS data.