FluxPyt: a Python-based free and open-source software for ^{13}C-metabolic flux analyses

Trunil S. Desai[1,2] and Shireesh Srivastava[1,2]

[1] Systems Biology for Biofuels Group, International Centre for Genetic Engineering and Biotechnology, New Delhi, Delhi, India
[2] DBT-ICGEB Center for Advanced Bioenergy Research, International Centre for Genetic Engineering and Biotechnology, New Delhi, Delhi, India

ABSTRACT

^{13}C-Metabolic flux analysis (MFA) is a powerful approach to estimate intracellular reaction rates which could be used in strain analysis and design. Processing and analysis of labeling data for calculation of fluxes and associated statistics is an essential part of MFA. However, various software currently available for data analysis employ proprietary platforms and thus limit accessibility. We developed FluxPyt, a Python-based truly open-source software package for conducting stationary ^{13}C-MFA data analysis. The software is based on the efficient elementary metabolite unit framework. The standard deviations in the calculated fluxes are estimated using the Monte-Carlo analysis. FluxPyt also automatically creates flux maps based on a template for visualization of the MFA results. The flux distributions calculated by FluxPyt for two separate models: a small tricarboxylic acid cycle model and a larger Corynebacterium glutamicum model, were found to be in good agreement with those calculated by a previously published software. FluxPyt was tested in Microsoft™ Windows 7 and 10, as well as in Linux Mint 18.2. The availability of a free and open ^{13}C-MFA software that works in various operating systems will enable more researchers to perform ^{13}C-MFA and to further modify and develop the package.

Submitted 2 March 2018
Accepted 13 April 2018
Published 27 April 2018

Corresponding author
Shireesh Srivastava,
shireesh@icgeb.res.in

Academic editor
Claus Wilke

Additional Information and Declarations can be found on page 14

DOI 10.7717/peerj.4716

Subjects Biochemistry, Biotechnology, Computational Biology
Keywords Metabolic labeling studies, Flux analysis, Elementary metabolite unit, Flux maps, Mass isotopomer distribution, Central carbon metabolism

INTRODUCTION

Metabolic flux analysis (MFA) is an important component of metabolic engineering (*Stephanopoulos, 1999*). MFA measures the distribution of flux in the different metabolic pathways of an organism as well as identifies the intracellular flux changes associated with altered cellular response such as overproduction of a desired metabolite for biotechnological applications or understanding the altered pathways in diseases. The flux values from MFA can be used to constrain the solution space of the linear optimization problem in flux balance analysis to give a physiologically meaningful flux distribution (*Kim & Reed, 2012*; *Desai & Srivastava, 2015*).

Metabolic flux analysis methodology has evolved over years. Earlier studies employed an approach termed stoichiometric MFA which balanced the fluxes around the

intracellular metabolites in a stoichiometric network (*vanGulik et al., 2001*; *Srivastava & Chan, 2008*). While it provided important insights into altered metabolism under different conditions, there are several limitations of stoichiometric MFA such as the inability to determine fluxes in reversible or parallel reactions (*Wiechert et al., 2001*). ^{13}C-MFA, which employs ^{13}C-labeled substrates can resolve fluxes through such reactions. For a typical ^{13}C-MFA, the cells are fed with a labeled substrate or mixture of labeled substrates, labeled at one more carbon atoms. The cells are required to be at metabolically steady state, i.e., when the intracellular concentrations of metabolites do not change over time (*Zamboni et al., 2009*). This ensures that the differences in the labeling patterns are not due to temporal variations in the fluxes, but due to the differential activity of the pathways. Once the labeled substrate attains steady state distribution throughout the metabolic network (isotopic steady state), the relative abundances of the mass isotopomers of intracellular metabolites and proteinogenic amino acids are measured using NMR (*Szyperski et al., 1999*) or mass spectrometry to derive their respective mass isotopomer distributions (MIDs). The gas chromatography/mass spectrometry technique for measurement of MIDs of *tert*-butyldimethylsilyl derivatives of amino acids is well established. The fragmentation pattern produced by the derivatives is also well characterized and these ions are used to calculate the MIDs which are eventually used to calculate the fluxes (*Antoniewicz, Kelleher & Stephanopoulos, 2007b*; *Leighty & Antoniewicz, 2013*; *Crown, Long & Antoniewicz, 2015*; *Gonzalez, Long & Antoniewicz, 2017*). Zamboni et al. provide a detailed protocol for ^{13}C-MFA (*Zamboni et al., 2009*). A suitable metabolic model and the MID data are then fed in software tools which estimate the flux distributions in reactions included in the MFA model.

Along with the development of experimental techniques for stationary MFA, the methodology to calculate flux distribution from the labeling data has also evolved over the years. An important development in this direction was the introduction of the elementary metabolite unit (EMU) framework (*Antoniewicz, Kelleher & Stephanopoulos, 2007a*). The EMU decomposition algorithm detects minimum number of isotopomer balances needed to simulate the observed labeling pattern which greatly reduces the number of variables without any loss of information, thus making it much more efficient than the isotopomer mapping matrices approach (*Schmidt et al., 1997*). Based on the measured fractional enrichments of metabolites, the metabolic network is decomposed into elementary reactions of the EMUs. These elementary reactions form the basis of system of equations representing the relation between metabolic fluxes and the observed isotopic labeling. Recent MFA software such as WUFlux (*He et al., 2016*), Metran (*Antoniewicz, Kelleher & Stephanopoulos, 2007a*) and OpenFLUX (*Quek et al., 2009*) employ the EMUs framework. Most software for ^{13}C-MFA such as OpenFLUX (*Quek et al., 2009*), WUFlux (*He et al., 2016*), INCA (*Young, 2014*) and Metran (*Antoniewicz, Kelleher & Stephanopoulos, 2007a*) work either on the proprietary MATLAB platform or are not free and open source. Many also use the MATLAB statistics and optimization toolboxes which are proprietary. Thus, a truly free and open-source software is not available yet. Understanding the requirement for open source tools, Birkel et al. have recently developed

jQMM, an open source Python library to perform ^{13}C-MFA which uses the general algebraic modeling system (GAMS) to solve the optimization problem (*Birkel et al., 2017*). However, GAMS is a proprietary system for mathematical optimization. We believe that a truly open-source software in which all the components and packages are free and open source would make MFA more accessible to researchers and also encourage further development of ^{13}C-MFA and related tools.

MATERIALS AND METHODS

FluxPyt is written in Python (version 3.6.1) because Python is an open-source platform and has free, well-developed data handling packages with ability to perform the various tasks in ^{13}C-MFA. In FluxPyt, the numerical and matrix operations are conducted using the packages NumPy and SciPy (*van der Walt, Colbert & Varoquaux, 2011*), and SymPy (*Meurer et al., 2017*). The non-linear optimization (see Eq. 3 below) is conducted using the package lmfit (*Newville et al., 2014*). Python was installed as Anaconda3 (version 3.4.4.0) distribution. Anaconda3 also acts as a package manager to handle the dependency and compatibility issues between the different packages necessary for the functioning of FluxPyt. FluxPyt was developed and tested using the Spyder integrated development environment (IDE) with the IPython console. FluxPyt was tested in Windows 7, Windows 10 and Linux Mint (18.2) operating systems (OS). Detailed information of packages in the Anaconda distribution is provided in Data S1. While FluxPyt is designed to be used by people with limited programming skills, the open source nature allows further modification by advanced users. The general framework of FluxPyt is shown in Fig. 1. Table 1 describes different modules present in FluxPyt. The MFA model, isotope correction data and substrate input data are written in a csv file which can be easily edited in a spreadsheet. The reactions are divided into different types, namely, irreversible ("F"), reversible forward ("FR"), reversible reverse ("R"), metabolite balancing reactions ("B") and isotopomer balancing reactions ("S"). The metabolites which do not take part in isotopomer balancing are marked as "X" in the atom transition equations. The basis flux, which typically is the substrate uptake flux in a network, is marked with a "*". The upper bounds of other reactions are limited to 15 times this basis flux. The reverse free fluxes cannot be determined as they have no natural upper limit (*Quek et al., 2009*). The substrate information and the natural correction data are read from separate csv files which must be named substrate_input.csv and corr_file.csv, respectively. The correction file also specifies the number of mass isotopomer measurements for every amino acid provided for fitting. This is used to extract the MID vectors of length equal to the length of MID vectors provided in the measurement data. For detailed information on creating an MFA model for FluxPyt, a user guide is provided in the supplemental material (Data S2).

The model is automatically parsed into metabolite and isotopomer models. The metabolite model is a mass balance model that satisfies the metabolic steady state condition,

$$S \cdot v = 0 \tag{1}$$

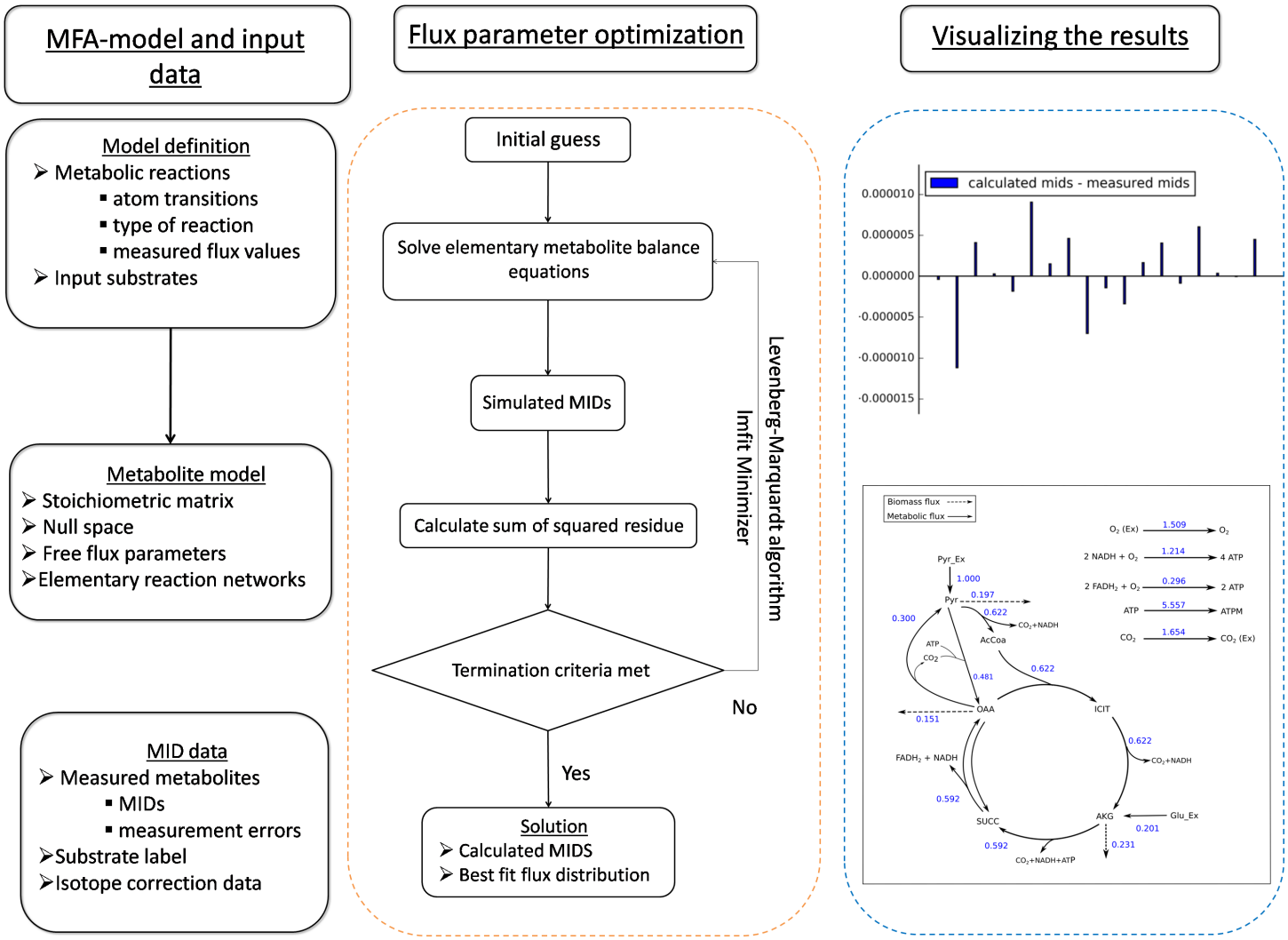

Figure 1 The general framework of FluxPyt. The user of the software provides the model definition and the mass isotopomer distribution (MID) data. The metabolite model is then calculated. The flux parameter optimization is then performed using the Levenberg–Marquardt algorithm to calculate the MIDs and the best-fit fluxes. The differences in the calculated vs. the measured MIDs are plotted. The flux maps of the calculated flux distribution is automatically generated based on the provided svg file. Full-size DOI: 10.7717/peerj.4716/fig-1

where, S is the stoichiometric matrix and v is the flux distribution vector. For a given stoichiometric matrix, the flux distribution vector can be written as a function of free fluxes (v_{free}) as follows:

$$v = N \cdot v_{free} \tag{2}$$

where N is the null space of stoichiometric matrix.

The isotopomer balance model is generated based on the EMU decomposition algorithm (*Antoniewicz, Kelleher & Stephanopoulos, 2007a*). It assumes an isotopic steady state and is used to solve the non-linear optimization problem using the free fluxes as the optimization parameters to minimize the sum of squared residues (SSR) as given in Eq. (3).

Table 1 Description of modules in FluxPyt.

Module	Description
main.py	Main driver module. Asks user inputs
build_model.py	Parses csv model file. Generates metabolite and isotopomer models, stoichiometric matrix. Reads measured MIDs
create_atm_transition_equations.py	Creates list of elementary reactions from atom transition equations
input_substrate_emu.py	Reads substrate input file and calculates substrate MIDs required in the emu network
mid_vec_gen.py	Generates correction vector for a given natural isotope distribution vector and number of atomic elements
priliminary_fba.py	Generates initial feasible flux distribution using glpk as linear programming solver and sets feasible reaction bounds
glpk_solve.py	Solves linear programming problems using glpk
make_emu_networks.py	Makes EMU networks from elementary reactions and measured MIDs
free_flux_null.py	Calculates null matrix and free fluxes
solve_mfa_lm.py	Generates and solves non-linear optimization problem using lmfit
solve_mid_networks.py	Solves EMU network to find calculated MIDS with respect to parameters of the non-linear problem
mid_corr.py	Corrects calculated MIDs
draw_flux_map.py	Draws flux map from optimization result and flux map template
monte_carlo.py	Performs Monte-Carlo analysis
bootstrap.py	For bootstrapping results from Monte-Carlo analysis

$$\text{Minimize}\left[\frac{(\text{MID}_{\text{calc}}-\text{MID}_{\text{measured}})^2}{\text{Error}^2}\right] \tag{3}$$

Levenberg–Marquardt algorithm implemented in the lmfit package is used to optimize the free flux parameters in the metabolite model. The free flux assignments are done based on the reduced row echelon form (rref) of the stoichiometric matrix. Before calculating the rref, the reactions are arranged in following order:

$$\begin{bmatrix} v_{\text{rev}}^{\rightarrow} \\ v_{\text{irrev}} \\ v_{\text{free}}^{\rightarrow} \\ v_{\text{basis}} \\ v_{\text{rev}}^{\leftarrow} \end{bmatrix}$$

$v_{\text{rev}}^{\rightarrow}$ and $v_{\text{rev}}^{\leftarrow}$ are the forward and reverse reactions of reversible reactions; v_{irrev} are irreversible reactions; $v_{\text{free}}^{\rightarrow}$ are reactions that the user chooses to act as free fluxes; v_{basis} are reactions with measured or predetermined values. The arrangement of reactions in a specified order ensures that the measured fluxes are calculated as free fluxes. In addition, the user may provide a preference for certain reactions to be calculated as free fluxes ($v_{\text{free}}^{\rightarrow}$). The default number of iterations is 10, but this value can be changed by the user. The iterations are run with different random initial parameters so as to increase the chances of finding global minimum.

Flux standard deviations are calculated using the Monte-Carlo method. Multiple datasets of the measured MID values are generated using the mean and standard

deviations of the respective MIDs. The default value for the number of datasets is 500, but can be changed by the user. Each dataset of the measured MID values is then used to solve the non-linear optimization problem (*He et al., 2016*). This results in multiple optimal solutions (500 solutions by default) for each reaction. From these values the 68% and 95% confidence intervals (CIs) are determined from the upper and lower 16% and 2.5% values, respectively, using the bootstrap method (*He et al., 2016*) as follows. One thousand samples of 500 values each (if the default number of samples for Monte-Carlo analysis is used) are used to calculate the required percentiles (2.5, 16, 50, 84 and 97.5 percentiles). The calculated percentiles from each sample are stored as a separate set. The median of each percentile value is reported as the upper or lower boundary of the respective CI. For example, the median 2.5 and 97.5 percentile values represent the lower and upper boundaries of the 95% CI and the median 16 and 84 percentile values represent the lower and upper boundaries of the 68% CI, respectively. The median of the 50 percentile values represents the median of the estimated fluxes.

Goodness of fit

Assuming that the model is correct and the data are without gross errors, the SSR has a χ^2 distribution with number of degrees of freedom equal to number of measurements to be fitted (n) minus the number of independent parameters (p) (*Antoniewicz, Kelleher & Stephanopoulos, 2006*; *Leighty & Antoniewicz, 2013*). The acceptable range of SSR is between $\chi^2_{\alpha/2}(n-p)$ and $\chi^2_{1-\alpha/2}(n-p)$, where α is the chosen significance level, for example, $\alpha = 0.05$ for 95% CI.

Output files and visualization of the results

FluxPyt generates several result files during the MFA, a list of which is provided in Table S1. The data results, containing the calculated MIDs, the optimal flux distribution values and CIs are stored as separate .csv and .pckl files. FluxPyt provides graphical representation of results. The differences between the measured and the calculated MIDs of the best fit solution are automatically plotted. Flux map is automatically generated using a template flux diagram created using Inkscape. The template for flux map is a svg file that can be created using Inkscape. Templates for tricarboxylic acid cycle (TCA) and *Corynebacterium glutamicum* networks are provided in the additional data (Data S1) or can be downloaded from the FluxPyt project site. The user can easily modify the template based on their model of interest using the same drawing tool to make flux maps for their organism of interest. The users need to write the ID of the reactions at the appropriate position in the template where they want the flux value of that particular reaction to be printed (Data S1). The bootstrap results are plotted as box plots. The user can choose the specific reactions to be plotted. Additionally, the bootstrap output is stored as a pandas dataframe (*McKinney, 2010*).

RESULTS

We employed the models and the MID data given in OpenFLUX (*Quek et al., 2009*) for initial evaluation of FluxPyt. Both the TCA cycle and the *C. glutamicum* central metabolic network (Data S1) were compared.

Table 2 Mass isotopomer distributions (MIDs) for the TCA cycle model as calculated by FluxPyt and OpenFLUX.

		FluxPyt	OpenFLUX
VALX:11111	M+0	0.0298	0.0298
	M+1	0.2438	0.2438
	M+2	0.0589	0.0589
	M+3	0.4399	0.4399
	M+4	0.0292	0.0292
	M+5	0.1984	0.1984
LYSX:111111	M+0	0.0344	0.0344
	M+1	0.1854	0.1854
	M+2	0.1677	0.1677
	M+3	0.2499	0.2499
	M+4	0.2074	0.2074
	M+5	0.0764	0.0764
	M+6	0.0788	0.0788
ASPX:1111	M+1	0.0661	0.0661
	M+2	0.3532	0.3532
	M+3	0.2497	0.2497
	M+4	0.1590	0.1590
	M+5	0.1720	0.1720
Sum of squared residual (SSR)		0.0002	0.0002

Preliminary validation with the TCA cycle model

The TCA cycle metabolic network (see Data S1 for a detailed description of the model as well as a list of metabolites) consisted of 18 metabolite reactions and three isotopomer balancing reactions (type "S" reactions). The network has two substrate input reactions, pyruvate (V01) and glutamate (V02) and one reversible reaction (represented as two separate reactions, V07 and V08). Three of the reactions (V16, V17 and V18) have measured values relative to pyruvate flux fixed at 1. The stoichiometric matrix has seven free variables. Rearranging the reactions as mentioned in the implementation section above enables the measured flux values to be calculated as free fluxes. The three measured free flux parameters were allowed to vary within three standard deviations of their means while the remaining three free flux parameters were allowed to vary between the maximum bounds as explained in the implementation section. The only remaining parameter was the flux through pyruvate uptake reaction which was fixed to 1. A total of 18 data points were fitted to six flux parameters, therefore the maximum acceptable SSR was 21. The number of iterations was set to 10. Flux analysis results from both software converged to a similar solution. The MIDs calculated by FluxPyt were the same as those calculated by OpenFLUX (Table 2). Very minor differences in the values of two fluxes were observed (Table 3). These could be due to the different solvers and algorithms used to solve the least squares fitting problem and linear system of equations (*He et al., 2016*). The flux map, which was automatically generated (see Methods for details) based

Table 3 Comparison of the optimal flux values for the TCA cycle model as calculated by FluxPyt and OpenFLUX.

Reaction ID	Reaction formula	FluxPyt	OpenFLUX
V01	PYR_EX -> PYR	1.00	1.00
V02	GLU_EX -> AKG	0.20	0.20
V03	PYR -> ACCOA + CO_2 + NADH	0.62	0.62
V04	ACCOA + OAA -> ICI	0.62	0.62
V05	ICI -> AKG + CO_2 + NADH	0.62	0.62
V06	AKG -> 0.5 SUC + 0.5 SUC + CO_2 + NADH + ATP	0.59	0.59
V07	SUC -> OAA + FADH2 + NADH	2.62	2.61
V08	OAA + FADH2 + NADH -> 0.5 SUC + 0.5 SUC	2.02	2.02
V09	OAA -> PYR + CO_2	0.30	0.30
V10	PYR + CO_2 + ATP -> OAA	0.48	0.48
V11	2 NADH + O_2 -> 4 ATP	1.21	1.21
V12	2 FADH2 + O_2 -> 2 ATP	0.30	0.30
V13	O_2_EX -> O_2	1.51	1.51
V14	CO_2 -> CO_2_EX	1.65	1.65
V15	ATP -> ATPM	5.56	5.54
V16	PYR -> PYR_B	0.20	0.20
V17	AKG -> AKG_B	0.23	0.23
V18	OAA -> OAA_B	0.15	0.15

on the template, is shown in Fig. 2. The Monte-Carlo analysis suggested tight CIs for the TCA model (Fig. 3). The reversible reaction spanned the entire flux range, as expected. The optimal values for all fluxes lay within 2.5 and 97.5 percentile range, i.e., the 95% CI (Table 4).

MFA of *C. glutamicum* metabolic network

The *C. glutamicum* model (*Quek et al., 2009*) consisted of 67 reactions, 42 metabolites and 25 free fluxes (see Data S1 for a detailed description of the model as well as a list of metabolites). The flux measurements for 18 reactions were included in the model. The 24 variable parameters were fit to the 29 data points of amino acid labeling pattern. Here too, the calculated fluxes were similar to those calculated by OpenFLUX (Table S2).

The SSR values for the measured vs. calculated MIDs were similar for FluxPyt (690) and OpenFLUX (701) (Table 5). However, the SSR values for both the software were higher than the maximum acceptable SSR value (11.07), suggesting that either the model used was incorrect or there were measurement errors. Use of a recent MID data (*Shupletsov et al., 2014*), reduced the SSR to 10.33 (Table 6), which is within the acceptable range (1.15–11.07). This showed that the MID data previously used had measurement errors. The Fig. 4 shows the flux map for the *C. glutamicum* model for this data (see Table S3) for the list of flux values. The CIs calculated with Monte-Carlo method are shown in Table S4.

Our results show that FluxPyt can be used for performing MFA with real metabolic models. It is capable of flux estimation when multiple labeled substrates are used

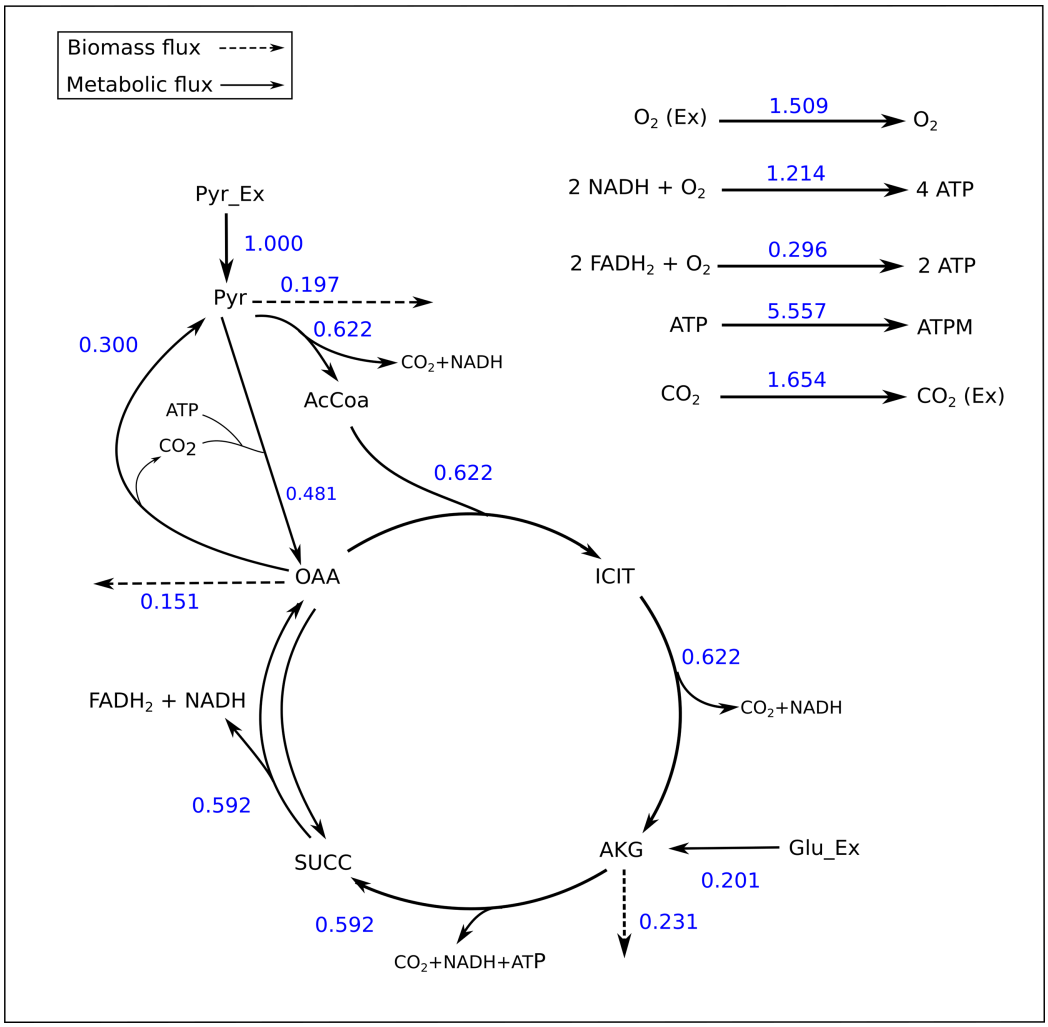

Figure 2 Flux map of the TCA generated by FluxPyt. The fluxes for the reactions of the TCA cycle model are shown as generated by the FluxPyt. Net flux values are shown for reversible reactions.
Full-size DOI: 10.7717/peerj.4716/fig-2

(e.g., the TCA model uses three labeled substrates, namely 1-^{13}C pyruvate, U-^{13}C pyruvate and 1-^{13}C Glutamate).

DISCUSSION

Metabolic flux analysis is a powerful method to measure intracellular fluxes. However, all of the currently used software have some component that is proprietary and thus makes it impossible to conduct ^{13}C-MFA without obtaining/purchasing a license. Most licenses do not allow altering the code, restricting further development. In designing this open-source software, our effort was to keep the interface simple and easy to use for scientists somewhat familiar with other MFA software. Therefore, the files for labeling-data preparation have been kept similar to the popular software OpenFLUX. The software can be run through different interfaces. The Anaconda command prompt may be convenient for people familiar with Python, while the Spyder IDE would appear more

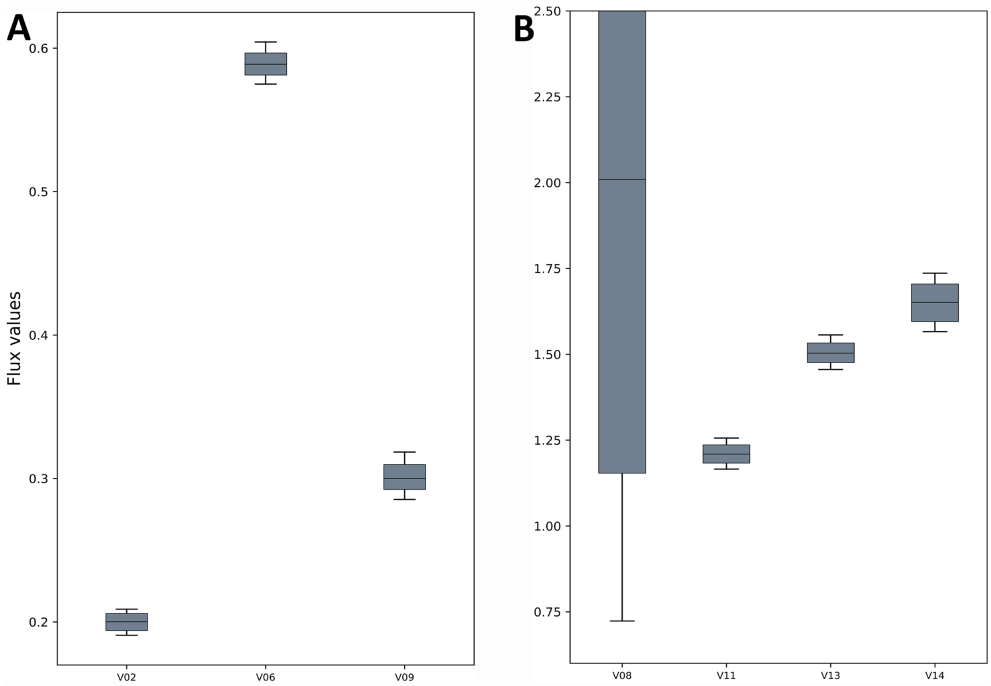

Figure 3 Flux confidence intervals (CIs) calculated by Monte-Carlo method for selected reactions. The boxes span 16 and 84 percentiles and whiskers span 2.5 and 97.5 percentiles. See the TCA model in Data S1 for full form of the reactions. (A) The flux CIs for the reactions V02, V06 and V09, (B) the flux CIs for the reactions V08, V11, V13 and V14. Full-size DOI: 10.7717/peerj.4716/fig-3

Table 4 The 95% and 68% confidence intervals for the TCA network calculated by the Monte-Carlo analysis in FluxPyt.

Reaction ID	−95% CI	−68% CI	Median	+68% CI	+95% CI
V01	1.00	1.00	1.00	1.00	1.00
V02	0.19	0.19	0.20	0.21	0.21
V03	0.59	0.60	0.62	0.64	0.66
V04	0.59	0.60	0.62	0.64	0.66
V05	0.59	0.60	0.62	0.64	0.66
V06	0.57	0.58	0.59	0.60	0.60
V07	1.30	1.74	2.59	4.17	9.41
V08	0.72	1.15	2.01	3.58	8.81
V09	0.29	0.29	0.30	0.31	0.32
V10	0.46	0.47	0.48	0.49	0.50
V11	1.17	1.18	1.21	1.24	1.26
V12	0.29	0.29	0.29	0.30	0.30
V13	1.46	1.48	1.50	1.53	1.56
V14	1.57	1.60	1.65	1.70	1.74
V15	5.33	5.41	5.53	5.66	5.76
V16	0.17	0.18	0.20	0.22	0.23
V17	0.20	0.20	0.23	0.26	0.26
V18	0.12	0.12	0.15	0.18	0.18

Table 5 The MIDs for the *C. glutamicum* metabolic network calculated by FluxPyt and OpenFLUX.

		Measured MIDs (*Quek et al., 2009*)	FluxPyt	OpenFLUX
Ala 260	M+0	0.5085	0.5096	0.5095
	M+1	0.3529	0.3537	0.3538
	M+2	0.1058	0.1062	0.1063
Val 288	M+0	0.3455	0.3472	0.3477
	M+1	0.3983	0.3979	0.3978
	M+2	0.1845	0.1840	0.1838
Thr 404	M+0	0.3330	0.3342	0.3341
	M+1	0.3764	0.3750	0.3758
	M+2	0.1957	0.1958	0.1957
Asp 418	M+0	0.3343	0.3336	0.3334
	M+1	0.3732	0.3743	0.3750
	M+2	0.1955	0.1962	0.1961
Glu 432	M+0	0.2469	0.2499	0.2493
	M+1	0.3648	0.3652	0.3657
	M+2	0.2412	0.2392	0.2396
Ser 390	M+0	0.4497	0.4483	0.4486
	M+1	0.3576	0.3582	0.3581
	M+2	0.1428	0.1437	0.1435
Phe 336	M+0	0.2712	0.2736	0.2736
	M+1	0.3816	0.3812	0.3814
	M+2	0.2282	0.2282	0.2283
Gly 246	M+0	0.7407	0.7416	0.7416
	M+1	0.1845	0.1852	0.1852
Tyr 466	M+0	0.2344	0.2356	0.2356
	M+1	0.3530	0.3560	0.3561
	M+2	0.2423	0.2448	0.2448
Tre 361	M+0	0.0613	0.0618	0.0618
	M+1	0.6040	0.6059	0.6060
	M+2	0.2070	0.2071	0.2070
SSR			690	701

familiar to scientists familiar with MATLAB interface. The IPython console in Spyder offers a more interactive environment than the command prompt. A print of FluxPyt for performing MFA of TCA network running in the Spyder console is provided in Data S3.

The fluxes calculated by MFA are dependent on the network chosen and the measurements used. In case of the *C. glutamicum* metabolic network, it was found that the MID data used by *Quek et al. (2009)* had measurement errors giving unacceptably high SSR for the MIDs calculated. Use of more recent MID data (*Shupletsov et al., 2014*) significantly reduced the SSR to acceptable values.

Table 6 The MIDs for the *C. glutamicum* measured by *Shupletsov et al. (2014)* vs. those calculated by FluxPyt.

		Measured	Simulated
Ala 260	m	0.504	0.506
	m+1	0.352	0.354
	m+2	0.107	0.109
Val 288	m	0.346	0.346
	m+1	0.397	0.397
	m+2	0.185	0.185
	m+3	0.056	0.056
Thr 404	m	0.332	0.333
	m+1	0.373	0.375
	m+2	0.195	0.196
	m+3	0.073	0.074
Asp 418	m	0.331	0.333
	m+1	0.373	0.375
	m+2	0.195	0.197
	m+3	0.074	0.074
Glu 432	m	0.248	0.248
	m+1	0.364	0.366
	m+2	0.240	0.241
	m+3	0.103	0.103
Ser 390	m	0.443	0.448
	m+1	0.353	0.358
	m+2	0.142	0.145
	m+3	0.049	0.050
Phe 336	m	0.273	0.273
	m+1	0.381	0.379
	m+2	0.229	0.228
	m+3	0.087	0.087
Gly 246	m	0.731	0.737
	m+1	0.183	0.189
	m+2	0.072	0.074
Tyr 466	m	0.235	0.235
	m+1	0.355	0.354
	m+2	0.245	0.245
	m+3	0.112	0.112
Trem 361	m	0.062	0.062
	m+1	0.605	0.605
	m+2	0.208	0.208
	m+3	0.097	0.097
SSR			10.33

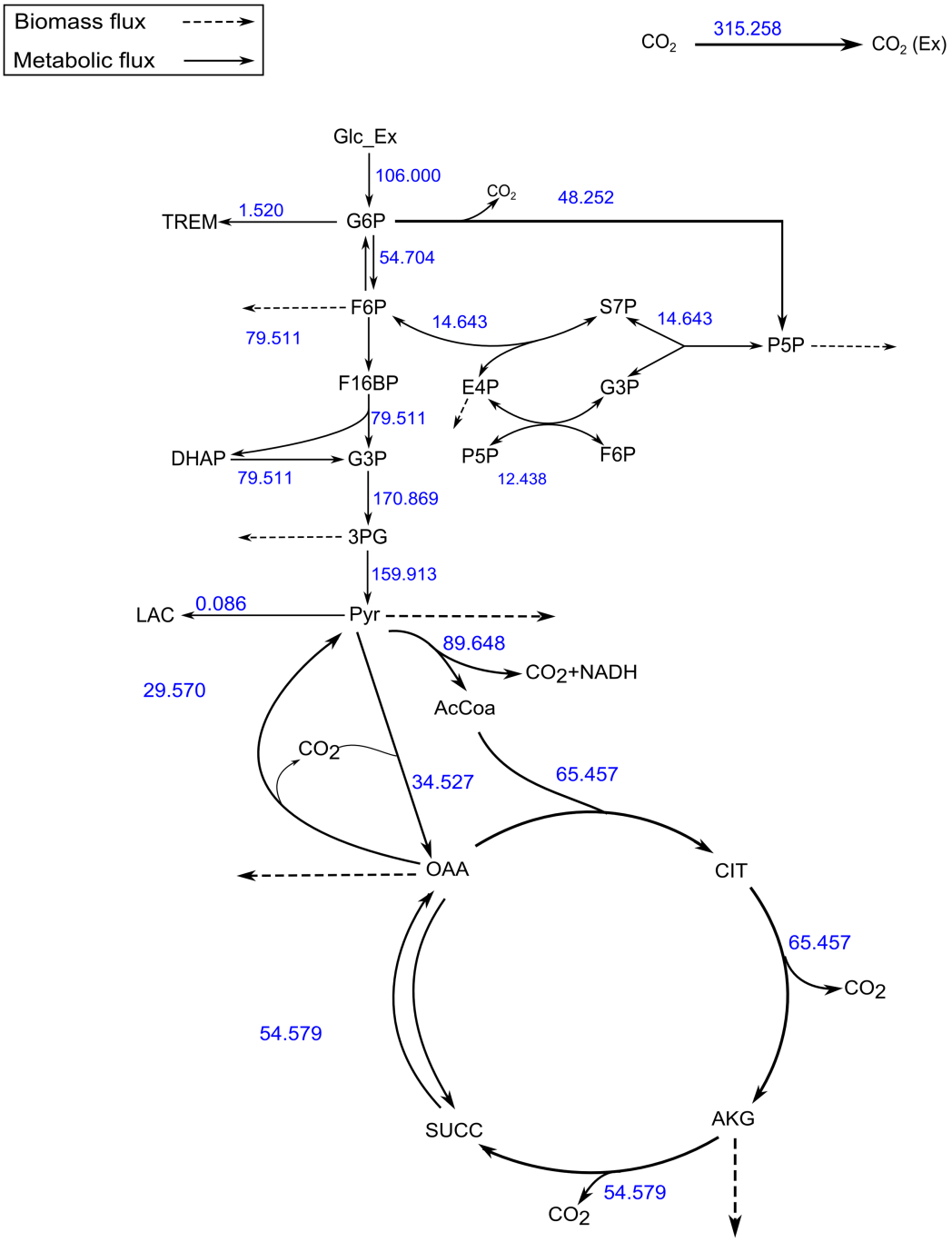

Figure 4 A map of the *C. glutamicum* fluxes generated by FluxPyt. The fluxes of the metabolic network of the *C. glutamicum* calculated using the data from *Shupletsov et al. (2014)* are shown on the metabolic network. The figure is generated automatically by the software.

Full-size DOI: 10.7717/peerj.4716/fig-4

At this point the software is released for use in Microsoft Windows and Linux OS. Similarly, the CI calculations currently use the Monte-Carlo method. The open-source nature makes it possible for advanced users to adapt it to other OS as well as add other methods for calculating CIs.

CONCLUSION

FluxPyt is free and open-source software for conducting stationary ^{13}C-MFAanalyses with performance comparable to other available software. It can be used for flux estimation with experimental data of practical scale. Automatic generation of flux maps allows better visualization of results. This tool will be useful to researchers interested in stationary ^{13}C-MFA studies.

ADDITIONAL INFORMATION AND DECLARATIONS

Funding

This work was supported by the Department of Biotechnology (DBT), Ministry of Science and Technology, India, through the grant no. BT/PB/Center/03/ICGEB/2011-Phase II. Trunil S. Desai's PhD fellowship is funded by the Council for Scientific and Industrial Research (CSIR), India. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Grant Disclosures

The following grant information was disclosed by the authors:
Department of Biotechnology (DBT), Ministry of Science and Technology, India: BT/PB/Center/03/ICGEB/2011-Phase II.
Council for Scientific and Industrial Research (CSIR), India.

Competing Interests

The authors declare that they have no competing interests.

Author Contributions

- Trunil S. Desai conceived and designed the experiments, performed the experiments, analyzed the data, contributed reagents/materials/analysis tools, prepared figures and/or tables, authored or reviewed drafts of the paper, approved the final draft.
- Shireesh Srivastava conceived and designed the experiments, analyzed the data, contributed reagents/materials/analysis tools, prepared figures and/or tables, authored or reviewed drafts of the paper, approved the final draft.

Data Availability

The following information was supplied regarding data availability:

Sourceforge: https://sourceforge.net/projects/fluxpyt/

Operating system: Windows 10, Windows 7 and Linux; Programming language: e.g. Python (~3.6). Other requirements: e.g. NumPy, SciPy, SymPy, lmfit, csvkit, glpk, pandas. License: BSD 3-clause. Any restrictions to use by non-academics: None.

Supplemental Information

Supplemental information for this article can be found online at http://dx.doi.org/10.7717/peerj.4716#supplemental-information.

REFERENCES

Antoniewicz MR, Kelleher JK, Stephanopoulos G. 2006. Determination of confidence intervals of metabolic fluxes estimated from stable isotope measurements. *Metabolic Engineering* **8(4)**:324–337 DOI 10.1016/j.ymben.2006.01.004.

Antoniewicz MR, Kelleher JK, Stephanopoulos G. 2007a. Elementary metabolite units (EMU): a novel framework for modeling isotopic distributions. *Metabolic Engineering* **9(1)**:68–86 DOI 10.1016/j.ymben.2006.09.001.

Antoniewicz MR, Kelleher JK, Stephanopoulos G. 2007b. Accurate assessment of amino acid mass isotopomer distributions for metabolic flux analysis. *Analytical Chemistry* **79(19)**:7554–7559 DOI 10.1021/ac0708893.

Birkel GW, Ghosh A, Kumar VS, Weaver D, Ando D, Backman TWH, Arkin AP, Keasling JD, Martín HG. 2017. The JBEI quantitative metabolic modeling library (jQMM): a Python library for modeling microbial metabolism. *BMC Bioinformatics* **18(1)**:205 DOI 10.1186/s12859-017-1615-y.

Crown SB, Long CP, Antoniewicz MR. 2015. Integrated ^{13}C-metabolic flux analysis of 14 parallel labeling experiments in *Escherichia coli*. *Metabolic Engineering* **28**:151–158 DOI 10.1016/j.ymben.2015.01.001.

Desai T, Srivastava S. 2015. Constraints-based modeling to identify gene targets for overproduction of ethanol by *Escherichia coli*: the effect of glucose phosphorylation reaction. *Metabolomics* **5**:145 DOI 10.4172/2153-0769.1000145.

Gonzalez JE, Long CP, Antoniewicz MR. 2017. Comprehensive analysis of glucose and xylose metabolism in *Escherichia coli* under aerobic and anaerobic conditions by ^{13}C metabolic flux analysis. *Metabolic Engineering* **39**:9–18 DOI 10.1016/j.ymben.2016.11.003.

He L, Wu SG, Zhang M, Chen Y, Tang YJ. 2016. WUFlux: an open-source platform for ^{13}C metabolic flux analysis of bacterial metabolism. *BMC Bioinformatics* **17(1)**:444 DOI 10.1186/s12859-016-1314-0.

Kim J, Reed JL. 2012. RELATCH: relative optimality in metabolic networks explains robust metabolic and regulatory responses to perturbations. *Genome Biology* **13(9)**:R78 DOI 10.1186/gb-2012-13-9-r78.

Leighty RW, Antoniewicz MR. 2013. COMPLETE-MFA: complementary parallel labeling experiments technique for metabolic flux analysis. *Metabolic Engineering* **20**:49–55 DOI 10.1016/j.ymben.2013.08.006.

McKinney W. 2010. Data Structures for Statistical Computing in Python. In: van der Walt S, Millman J, eds. *Proceedings of the 9th Python in Science Conference*. Austin, Texas: SciPy.org, 51–56.

Meurer A, Smith CP, Paprocki M, Čertík O, Kirpichev SB, Rocklin M, Kumar A, Ivanov S, Moore JK, Singh S, Rathnayake T, Vig S, Granger BE, Muller RP, Bonazzi F, Gupta H, Vats S, Johansson F, Pedregosa F, Curry MJ, Terrel AR, Roučka Š, Saboo A, Fernando I, Kulal S, Cimrman R, Scopatz A. 2017. SymPy: symbolic computing in Python. *PeerJ Computer Science* **3**:e103 DOI 10.7717/peerj-cs.103.

Newville M, Stensitzki T, Allen DB, Ingargiola A. 2014. LMFIT: Non-Linear Least-Square Minimization and Curve-Fitting for Python DOI 10.5281/zenodo.11813.

Quek L-E, Wittmann C, Nielsen LK, Krömer JO. 2009. OpenFLUX: efficient modelling software for ^{13}C-based metabolic flux analysis. *Microbial Cell Factories* **8(1)**:25 DOI 10.1186/1475-2859-8-25.

Schmidt K, Carlsen M, Nielsen J, Villadsen J. 1997. Modeling isotopomer distributions in biochemical networks using isotopomer mapping matrices. *Biotechnology and Bioengineering* **55(6)**:831–840 DOI 10.1002/(SICI)1097-0290(19970920)55:6<831::AID-BIT2>3.0.CO;2-H.

Shupletsov MS, Golubeva LI, Rubina SS, Podvyaznikov DA, Iwatani S, Mashko SV. 2014. OpenFLUX2: ^{13}C-MFA modeling software package adjusted for the comprehensive analysis of single and parallel labeling experiments. *Microbial Cell Factories* **13(1)**:152 DOI 10.1186/s12934-014-0152-x.

Srivastava S, Chan C. 2008. Application of metabolic flux analysis to identify the mechanisms of free fatty acid toxicity to human hepatoma cell line. *Biotechnology and Bioengineering* **99(2)**:399–410 DOI 10.1002/bit.21568.

Stephanopoulos G. 1999. Metabolic fluxes and metabolic engineering. *Metabolic Engineering* **1(1)**:1–11 DOI 10.1006/mben.1998.0101.

Szyperski T, Glaser RW, Hochuli M, Fiaux J, Sauer U, Bailey JE, Wüthrich K. 1999. Bioreaction network topology and metabolic flux ratio analysis by biosynthetic fractional ^{13}C labeling and two-dimensional NMR spectroscopy. *Metabolic Engineering* **1(3)**:189–197 DOI 10.1006/mben.1999.0116.

van der Walt S, Colbert SC, Varoquaux G. 2011. The NumPy array: a structure for efficient numerical computation. *Computing in Science & Engineering* **13(2)**:22–30 DOI 10.1109/MCSE.2011.37.

vanGulik WM, Antoniewicz MR, deLaat WT, Vinke JL, Heijnen JJ. 2001. Energetics of growth and penicillin production in a high-producing strain of *Penicillium chrysogenum*. *Biotechnology and Bioengineering* **72(2)**:185–193 DOI 10.1002/1097-0290(20000120)72:23.0.co;2-m.

Wiechert W, Möllney M, Petersen S, De graaf AA. 2001. A universal framework for ^{13}C metabolic flux analysis. *Metabolic Engineering* **3(3)**:265–283 DOI 10.1006/mben.2001.0188.

Young JD. 2014. INCA: a computational platform for isotopically non-stationary metabolic flux analysis. *Bioinformatics* **30(9)**:1333–1335 DOI 10.1093/bioinformatics/btu015.

Zamboni N, Fendt S-M, Rühl M, Sauer U. 2009. ^{13}C-based metabolic flux analysis. *Nature Protocols* **4**:878–892 DOI 10.1038/nprot.2009.58.