Parameter Balancing

Manual

For updates and further information, please visit www.parameterbalancing.net.

If you use parameter balancing in your research, please cite our article:
Lubitz *et al.* (2010). Parameter balancing for kinetic models of cell metabolism. J. Phys. Chem. B, 114(49):16298

Contents

1 What is parameter balancing?

Parameter balancing is a method to determine consistent parameter sets for kinetic metabolic models. Experimentally measured values, when directly inserted into a model, are likely to yield incomplete and inconsistent parameter sets. Balanced parameter sets, which are complete and consistent, are computed from kinetic constants and other data collected from experiments or the literature, based on constraints between biochemical quantities and assumptions about typical ranges, represented by prior values and bounds. The mathematical details of parameter balancing can be found in [1]. Furthermore, the articles [2, 3, 4] exemplify how parameter balancing can be embedded in a metabolic modelling workflow. Parameter balancing can be employed in five ways:

1. On www.parameterbalancing.net, users can employ an **online version** of the parameter balancing tool (see section 2).
2. The same tool can be installed as **Python package** (see section 3).
3. It can be used as a **python commandline tool** (see section 3).
4. The online version can also be installed as a **web2py standalone application** (see section 3).
5. A **matlab implementation of parameter balancing** is part of the Metabolic Network Toolbox, freely available at https://github.com/liebermeister/mnt/.

This manual describes some details of the parameter balancing implementations. If you do not find your questions answered in this manual or in the above publications, please contact Wolfram Liebermeister (wolfram.liebermeister@gmail.com) and Timo Lubitz (timo.lubitz@gmail.com).

2 Online parameter balancing

2.1 Input files

For the online parameter balancing the user can employ SBML (www.sbml.org) and SBtab (www.sbtab.net) files as input format.

2.1.1 Model file (SBML)

The user will need at least a mathematical model in the SBML format to perform online parameter balancing. This format is widespread in the mathematical modelling community and can be created with numerous tools, such as Copasi (www.copasi.org). The SBML model should contain species and reactions. The validity of an SBML model should be checked before parameter balancing on the official validation page http://sbml.org/Facilities/Validator.

2.1.2 Data, prior, and options files (SBtab)

SBtab files can be created with any text and spreadsheet editor. They are tab-separated table files (.tsv) which can hold diverse content. We use SBtab as an input format for several types of data. A specification on SBtab and an online validation tool can be found on www.sbtab.net.

SBtab QuantityData (kinetic parameter file) SBtab files of the table type QuantityData hold data on kinetic parameter values. The user can provide numeric values for the entities (species and reactions) of the SBML model. To correctly link the provided parameter values of the SBtab file to the entities of the SBML model it is of utmost importance to include the columns !Compound:SBML:species:id and !Reaction:SBML:reaction:id. The entries of these columns must comply with the IDs (attention: not the names) of the SBML model entities.

Furthermore, the SBtab file urgently requires the columns !QuantityType, !Mean, !Std, and !Unit for the provision of the parameter type, the numeric parameter value, its standard deviation, and the corresponding physical unit, respectively. The parameter types include equilibrium constants, concentration, enzyme concentration, rate constants, chemical potentials, Michaelis constants and more. More details can be found in the SBtab specification on www.sbtab.net and example files on www.parameterbalancing.net/pb/static/downloads.html.

The user need not provide a SBtab QuantityData file, this is optional. But the provision of the balancing with kinetic parameter values is crucial for its accuracy. If no SBtab parameter file is given, the parameter balancing will base its computation solely on the prior distributions of the parameter types, which is rather imprecise.

SBtab QuantityInfo (prior distribution file) Parameter balancing is a Bayesian estimation method which requires prior distributions for all included parameter types. The distributions are represented as mean values with corresponding standard deviations. We use distributions which are common for the individual parameter types and are provided with very broad standard deviations to not put too strict limitations on the estimation.

The column "PriorMedian" contains the median values of the parameter priors. For additive parameters (with "original scaling"), the median value coincides with the mean value. For multiplicative parameters (with "logarithmic scaling"), the median value does not coincide with the parameter's own mean value, but is given by exp(x), where x is the mean value of the parameter's natural logarithm.

The column "PriorMedian" contains the prior median values for all parameter types. For additive parameters, the median value directly defines the mean value. For multiplicative parameters p, the median value is given by exp(x), where x is the mean value of the natural logarithm of p. Depending on the type of parameter, the prior standard deviation is either defined in the column "PriorStd" or in the column "PriorGeometricStd". For additive parameters, the column "PriorStd" contains the prior standard deviation of the parameter itself. For multiplicative parameters, the column "PriorGeometricStd" defines the geometric standard deviation of the parameters. By applying the natural logarithm, one obtains the prior standard deviation of the natural logarithm of p. The geometric standard deviation has a simple interpretation: for example, an entry of 10 shows that a parameter's typical deviations span about 1 order of magnitude below and above the median.

Experienced users with a special interest in unusual reaction systems might want to change the prior distributions to serve their individual requirements. This can be done by providing an SBtab QuantityInfo file. The easiest way to create such a file is to download the default file from the website on www.parameterbalancing.net/pb/static/downloads.html and change the !PriorMode and/or !PriorStd columns.

As implied before, the provision of such a file is only optional and not recommended for inexperienced users.

SBtab PbConfig (parameter balancing options file) Online parameter balancing is a complex process with many optional configurations. These can be changed by using a configuration file of the SBtab table type PbConfig. Again, the default configuration file can be downloaded on www.parameterbalancing.net/pb/static/downloads.html. The available configuration options are

- **use_pseudo_values** (Bool; *True* or *False*): usage of pseudo values as substitute for missing parameter values to improve balancing (see PLOS one publication for details)
- **pH** (float; 1-14): pH value that allows parameter balancing for correction of parameter values that do not correspond to the systems target pH value (see JoPC publication for details)
- **Temperature** (float, in Kelvin): temperature that allows parameter balancing for correction of parameter values that do not correspond to the systems target temperature (see JoPC publication for details)
- **overwrite_kinetics** (Bool; *True* or *False*): should existing kinetic laws in the SBML file be overwritten?
- **cell_volume** (float; in μm^3): volume of the cell
- **parametrisation** (string; hal or cat or weg): different parametrisation types for the convenience kinetics. See [5] and [6] for details
- **enzyme_prefactor** (Bool; *True* or *False*): should a prefactor for the enzyme concentration be set in the rate law?
- **default_inhibition** (string; *complete*, *partial*, or *specific*): type of enzyme inhibition
- **default_activation** (string; *complete*, *partial*, or *specific*): type of enzyme activation
- **model_name** (string): name that is used for the result files
- **boundary_values** (string; *ignore*): ignore boundary values (temporary feature)

2.1.3 Alternative: Single input file

All of the above SBtab files can be provided in one single spreadsheet to ease and fasten up the online parameter balancing. Moreover, also the SBML model can be translated to SBtab (on www.sbtab.net/convert) and be added to the SBtab file. In sum, the user only has to upload a single input file (multiple SBtab tables in one SBtab Document) for online balancing. Here again, example files for whole systems (model, parameters, prior file, config file) can be found on the download page www.parameterbalancing.net/pb/static/downloads.html.

2.2 Using the online workflow

Online parameter balancing can be performed on

www.parameterbalancing.net/pb/default/balancing.html

and there are two options for the user: SBML + SBtab input and SBtab only input.

2.2.1 Alternative 1: Upload SBML and SBtab files

The classic balancing requires at least an SBML model file to be uploaded. As soon as it is uploaded, a large ->Balance Parameters<- button appears on the bottom of the column. By clicking it, the SBML model is processed and parameter balancing is performed on grounds of the default prior distributions and default options. This is easy and fast, but for accuracy the user should provide more information.

Optionally, the user can provide an SBtab data file with experimental values. The more data are provided, the more accurate the balancing will get.

Also, the user can upload individual prior distributions for the different parameter types and a parameter balancing options file. Example files for all table types are available on our download page.

2.2.2 Alternative 2: Upload a single SBtab file

Instead of uploading single files for SBML model and optionally parameter, prior distribution, and options SBtabs, the user can also simply upload a single SBtab file which holds all required information needed for parameter balancing. The SBML file can first be translated to SBtab (on www.sbtab.net/convert) and all other SBtab files can simply be added to the same file (also see the online download section).

No matter which of the two options is chosen, after clicking the ->Balance Parameters<- button the user is referred to the results page. Here, they can download (i) the SBML model with balanced parameters and inserted convenience kinetics (as long as this option is not switched off with the options SBtab), (ii) an SBtab parameter file with all balanced parameters, and an SBtab model data file, which holds the model information, balanced parameter values, prior file, and configuration file all in one SBtab. The files can be viewed online, downloaded, or removed.

2.2.3 Result files (SBtab and SBML)

Parameter balancing algorithm results in a joint posterior distribution of all model parameters. Internally, multiplicative parameters (see prior table) are represented by their natural logarithms. In this representation, the posterior distribution of all model parameters is a multivariate Gaussian. However, constraints on the parameters (e.g., individual upper and lower bounds) can further restrict the distribution to a convex polytope. Given such constraints, it may happen that the center of our Gaussian distribution is outside the polytope and is not a feasible point. The posterior mode is the point with the highest probability density within the feasible polytope.

Balanced parameter set The result SBtab file holds the balanced parameters for each model entity. However, there are some details that motivate a closer look. In the standard result file, the posterior distribution is characterised by different numbers:

1. The columns "UnconstrainedMean" and "UnconstrainedStd" contain the arithmetic mean values and standard deviations of the *unconstrained* posterior. For multiplicative parameters - which follow log-normal distributions - these values can become very large because of the skewed shape of the log-normal distribution. For these variables, it is preferable to have a look at the geometric mean and standard deviation instead (see next point).

2. The columns "UnconstrainedGeometricMean" and "UnconstrainedGeometricStd" are relevant only for multiplicative parameters p. To compute them, the exponential function is applied to the mean and standard deviation, respectively, of ln(p). The geometric mean and standard deviation usually give a better account of a log-normal distribution than the corresponding arithmic mean and standard deviation, which may assume extreme values.

3. The column "Mode" describes the *constrained* mode of the distribution. To compute it, we determine the point in the feasible polytope where the posterior has its maximal value. For additive parameters, the value is reported directly. For multiplicative parameters, we apply the exponential function before the value is reported.

When inserting parameter values into a model, it is advisable to use the "Mode" entries as model parameters, because any of the other values might violate the constraints. If all constraints are inactive, the standard deviations of the Gaussian distribution can be used to characterise the posterior. If some constraints are active, then the standard deviations may not be meaningful, and sampling from the posterior is required.

Sampled parameter sets In the matlab version, the user can export a second output file with parameter sets sampled from the posterior distribution.

SBML model with modular rate laws Finally, the parameter balancing workflow also generates an SBML model with the same network structure as the input model, but with modular rate laws [6] and balanced parameters inserted.

3 Software code for developers and experienced users

3.1 Download and installation instructions

Parameter balancing can be used in five different ways: (i) The online interface described above, (ii) as a Python package, (iii) as a web2py application for a standalone version of the online balancing interface, (iv) as simple Python scripts that can be employed in the commandline, and (v) as part of the MATLAB Metabolic Networks Toolbox.

3.1.1 Online interface

For the usage of the online interface, a user needs a browser of any choice, an internet connection, and the input files. All further details are explained in Sections 2 and 3.4.2.

3.1.2 Python package

To install Parameter Balancing as a Python3 package, first of all you need Python3. Next, you will need the pip3 installer. You can find information on how to achieve this installer here: https://pip.pypa.io/en/stable/installing/. Afterwards, install Parameter Balancing by typing in your command line:

```
> sudo pip3 install pbalancing
```

This will also install libsbml and tablib on your computer if these libraries are missing. You can now employ Parameter Balancing as a Python3 package by, e.g., writing a Python script such as

```
from pbalancing import parameter_balancing
parameter_balancing.parameter_balancing_wrapper('model.xml')
```

In this example case, 'model.xml' is the file name of an SBML model. Further optional arguments are an SBtab parameter file, an SBtab prior distribution file, and an SBtab configuration file. You will find information on all these file types in the online Download area www.parameterbalancing.net/pb/static/download.html.

3.1.3 web2py standalone version

The parameter balancing interface can also be used as a standalone version. This requires initially the download of a web2py server from

http://www.web2py.com/examples/default/download

Next, the user needs to download the parameter balancing application from

https://github.com/tlubitz/parameter_balancing/web_version.

The application folder pb needs to be moved to the web2py directory, namely to web2py/applications/pb. Now the web2py server can be initiated (see www.web2py.com for details on your specific operating system) and the parameter balancing standalone can be accessed on any browser calling

http://127.0.0.1:8000/pb/

The benefit of this version is that the user will not require a running internet connection and can still run the balancing GUI in their browser. Furthermore, they can make alterations on the parameter balancing source code in the pb directory, assuming the user is familiar with Python3 and web2py. The prerequisites for running this interface are Python3 and the libsbml library. The former can be downloaded from

https://www.python.org/downloads/release/python-2713/

and the latter from

http://sbml.org/Software/libSBML/Downloading$_$libSBML.

If the user wants to contribute to the parameter balancing project, a Github account is required, the pb repository needs to be cloned, and changes can be pushed to the repository on request.

3.1.4 Python Scripts

To run parameter balancing as a commandline tool, the package needs to be installed as explained in 4.2. Then, it can be executed in the commandline as follows:

python3 -m pbalancing.parameter_balancing model.xml

where model.xml corresponds to the path to your SBML model. If you do not want to install the package, you can simply download the files and start the scripts from within the directory by typing

python3 parameter_balancing.py model.xml

3.1.5 Matlab code

An implementation of parameter balancing in Matlab is part of the Metabolic Networks Toolbox, which can be obtained from https://github.com/liebermeister/mnt/.

3.2 Parameter balancing in bigger workflows

Parameter balancing can be used in bigger workflows of mathematical modelling. An extensive example is given in [2]. Here, a geometric FBA is employed as first step, but this is not necessarily required - any FBA that generates loopless flux distributions is appropriate. Similarly, other steps of the presented workflow can be replaced by other modelling techniques, which yield the same context as the methods used.

Furthermore, in parameter balancing we adhere to common standard file formats such as SBML and SBtab to ensure an easy interface for embedding.

4 Table of biochemical quantities

At the beginning of parameter balancing, a metabolic network structure is read from an SBML file. It is assumed that all kinetic rate laws will be substituted by modular rate laws [6] and the necessary kinetic constants and some dynamic quantities (referring to a specific metabolic state of the system) are estimated from collected data. The relevant quantities are listed below.

Quantity Type	Symbol	Unit	BiologicalElement	MathematicalType	PhysicalType	Dependence	Prior Median	PriorStd PriorStd	Prior GeomStd
Standard chemical potential	0	kJ/mol	Species	Additive	Thermodynamic	Basic	0 kJ/mol	500 kJ/mol	
Catalytic rate constant geometric mean	kV	1/s	Reaction	Multiplicative	Kinetic	Basic	10		100
Michaelis constant	kM	mM	Reaction/Species	Multiplicative	Kinetic	Basic	0.1		10
Activation constant	kA	mM	Reaction/Species	Multiplicative	Kinetic	Basic	0.1		10
Inhibitory constant	kI	mM	Reaction/Species	Multiplicative	Kinetic	Basic	0.1		10
Concentration	c	mM	Species	Multiplicative	Dynamic	Basic	0.1		10
Concentration of enzyme	u	mM	Reaction	Multiplicative	Dynamic	Basic	0.001		100
pH	pH	dimensionless	None	Additive	Dynamic	Basic	7	1	
Standard Gibbs energy of reaction	dmuO	kJ/mol	Reaction	Additive	Thermodynamic	Derived	0	500	
Equilibrium constant	keq	dimensionless	Reaction	Multiplicative	Thermodynamic	Derived	1		100
Substrate catalytic rate constant	kcat+	1/s	Reaction	Multiplicative	Kinetic	Derived	10		100
Product catalytic rate constant	kcat-	1/s	Reaction	Multiplicative	Kinetic	Derived	10		100
Chemical potential		kJ/mol	Species	Additive	Dynamic	Derived	0	500	
Reaction affinity	A	kJ/mol	Reaction	Additive	Dynamic	Derived	0	500	
Forward maximal velocity	vmax+	mM/s	Reaction	Multiplicative	Dynamic	Derived	0.01		100
Reverse maximal velocity	vmax-	mM/s	Reaction	Multiplicative	Dynamic	Derived	0.01		100
Forward mass action term	thetaf	1/s	Reaction	Multiplicative	Dynamic	Derived	1		1000
Reverse mass action term	thetar	1/s	Reaction	Multiplicative	Dynamic	Derived	1		1000
Forward enzyme mass action term	tauf	mM/s	Reaction	Multiplicative	Dynamic	Derived	1		1000
Reverse enzyme mass action term	taur	mM/s	Reaction	Multiplicative	Dynamic	Derived	1		1000
Michaelis constant product	KMprod	mM	Reaction	Multiplicative	Kinetic	Derived	1	1000	
Catalytic constant ratio	Kcatratio	dimensionless	Reaction	Multiplicative	Kinetic	Derived	1		10

Quantity Type	Lower Bound	Upper Bound	DataStd DataStd	Data GeomStd	SBML element SBML element	UseAsPrior Information	MatrixInfo MatrixInfo
Standard chemical potential	-500	500	10		Global parameter	1	[I_species, 0, 0, 0, 0, 0, 0, 0]
Catalytic rate constant geometric mean	0.00000001	10000	10	1.2	Local parameter	1	[0, I_reaction, 0, 0, 0, 0, 0, 0]
Michaelis constant	0.0000001	1000	1	1.2	Local parameter	1	[0, 0, I_KM, 0, 0, 0, 0, 0]
Activation constant	0.0001	100	1	1.2	Local parameter	1	[0, 0, 0, I_KA, 0, 0, 0, 0]
Inhibitory constant	0.0001	100	1	1.2	Local parameter	1	[0, 0, 0, 0, I_KI, 0, 0, 0]
Concentration	0.0000001	1000	1	1.2	Species (conc.)	1	[0, 0, 0, 0, 0, I_species, 0, 0]
Concentration of enzyme	0.0000001	0.5	0.05	1.2	Local parameter	1	[[-1/RT * Nt], 0, 0, 0, 0, 0, 0, 0]
pH	0	14	1		Global parameter	1	[0, 0, 0, 0, 0, 0, 0, 1]
Standard Gibbs energy of reaction	-1000	1000	10		Global parameter	0	[Nt, 0, 0, 0, 0, 0, 0, 0]
Equilibrium constant	0.00000000001	10000000000	10	1.2	Local parameter	1	[[-1/RT * Nt], 0, 0, 0, 0, 0, 0, 0]
Substrate catalytic rate constant	0.01	1000000000	10	1.2	Local parameter	1	[[-0.5/RT * Nt], I_reaction, [-0.5 * Nkm], 0, 0, 0, 0, 0]
Product catalytic rate constant	0.00000000001	1000000000	10	1.2	Local parameter	1	[[0.5/RT * Nt], I_reaction, [0.5 * Nkm], 0, 0, 0, 0, 0]
Chemical potential	-500	500	10			0	[I_species, 0, 0, 0, 0, [RT * I_species], 0, 0]
Reaction affinity	-100	100	10			0	[[-1 * Nt], 0, 0, 0, 0, 0, [-RT * Nt], 0, 0]
Forward maximal velocity	0.000000001	10000000	0.1	2	Local parameter	0	[[-0.5/RT * Nt], I_reaction, [-0.5 * Nkm], 0, 0, 0, I_reaction, 0]
Reverse maximal velocity	0.000000001	10000000	0.1	2	Local parameter	0	[[0.5/RT * Nt], I_reaction, [0.5 * Nkm], 0, 0, 0, I_reaction, 0]
Forward mass action term	0.000000001	100000000	1	2		0	[[-1/(2*RT) * h * Nt], I_reaction, - 1/2 * h * abs(Nkm), 0, 0, h * Nft, 0, 0]
Reverse mass action term	0.000000001	100000000	1	2		0	[[1/(2*RT) * h * Nt], I_reaction, - 1/2 * h * abs(Nkm), 0, 0, h * Nrt, 0, 0]
Forward enzyme mass action term	0.000000001	100000000	1	2		0	[[-1/(2*RT) * h * Nt], I_reaction, - 1/2 * h * abs(Nkm), 0, 0, h * Nft, I_reaction, 0]
Reverse enzyme mass action term	0.000000001	100000000	1	2		0	[[1/(2*RT) * h * Nt], I_reaction, - 1/2 * h * abs(Nkm), 0, 0, h * Nrt, I_reaction, 0]
Michaelis constant product	0.001	1000	1	2	Local parameter	0	[0, 0, Nkm, 0, 0, 0, 0, 0]
Catalytic constant ratio	0.00000000001	10000000000	1	2	Local parameter	0	[-1/RT * Nt], I_reaction, [-1 * Nkm], 0, 0, 0, 0, 0]

Remarks

- **Transformed thermodynamic quantities** Note that the thermodynamic quantities refer to biochemical reactants (e.g.ATP) rather than chemical species (e.g. ATP4-). Therefore, they represent transformed quantities. In Alberty's exact notation, they would be written as K' (for k_{eq}), μ' (for μ), and μ'0 (for μ0).
- **Additive and multiplicative quantities** In parameter balancing, all quantities representing energies (in kJ/mol) are treated in their original scale, while all other quantities are converted to logarithmic scale (column "Scaling"). Since the latter quantities are described by log-normal (instead of normal) distributions, we need to distinguish between their mean and median values. For more information on this topic, see the review [7] by R. Alberty.
- **Basic quantities and derived quantities** We further distinguish between basic and derived quantities (column "Type"). The difference is that the basis quantities can be freely chosen (e.g., as a result of an estimation), while the derived quantities depend on the basic quantities and are computed from them. In parameter balancing, we define typical ranges for the basic quantities by prior distributions and for the derived quantities by pseudo values.
- **Parameters in SBML** When inserted into the SBML model, most quantities are inserted into kinetic rate laws as local parameters. Exceptions are concentrations (initial concentration attribute in species), standard chemical potentials (global parameters), and reaction affinities and chemical potentials, which could just be computed from other elements.
- **Importance of reaction orientation** Some quantities depend on the specific definition of the reaction sum formula (nominal direction, appearance of small molecules like water); in particular for equilibrium constants, catalytic rate constants, and maximal velocities, make sure that the definitions match between model and data set.
- **Chemical potentials and standard chemical potentials** Chemical potentials and standard chemical potentials are estimated during parameter balancing (e.g., based on equilibrium constants and metabolite concentrations in the data file), but they cannot be provided directly as data. The purpose of this restriction is to avoid consistency problems due to different scaling conventions (e.g., the choice of offset values in Gibbs free energies of formation)

5 Frequently asked questions

1. **What is parameter balancing?** Parameter balancing is a way to determine consistent parameter sets for kinetic models of metabolism. Inserting experimentally measured values directly into a model will probably yield incomplete or inconsistent parameter sets, violating the thermodynamic Haldane relationships. Balanced parameter sets avoid this problem. They are computed based on kinetic constants and other data collected from experiments or the literature, but also based on known constraints between biochemical quantities and on assumptions about typical ranges, represented by prior values and bounds.
2. **How can I run parameter balancing?** After preparing your model and data files, you can run parameter balancing interactively here. The workflow is described here. If you prefer working in the command line, or if you would like to include parameter balancing in your programs, you may use our code for Python and Matlab.
3. **Which parameters of a metabolic model can be balanced?** In general, parameter balancing concerns the kinetic and thermodynamic constants in kinetic metabolic models. It can also cover metabolite concentrations, chemical potentials, and reaction Gibbs free energies (or, equivalently, "reaction affinities" or "driving forces"). Metabolic fluxes cannot be balanced, but they can be included in the analysis (this is described below). There are different typical application cases:
    - Kinetic constants, where equilibrium constants are fixed and given This can be done separately for each individual reaction. If a network is large and equilibrium constants can be predefined (for instance, by parameter balancing), we suggest to split the network into single reactions and to run parameter balancing separately for every reaction.
    - Kinetic constants and equilibrium constants in a network
    - Equilibrium constants and concentrations in a network
    - Equilibrium constants, kinetic constants, and concentrations in a network

4. **What input data are needed?** Parameter balancing employs SBML (Systems Biology Markup Language) files or SBtab files for model structures and SBtab table files for data and configuration files. Parameter balancing imports a model (SBML, obligatory) and a data table (SBtab, optional) with experimental data values. Furthermore, tables withinformation on the prior distributions and balancing options are possible. Parameter balancing produces tables with balanced parameters (SBtab) and a model with rate laws and balanced parameters included (SBML). Please prepare your SBtab files as described below. The validity of these files can be checked on the SBtab website. The quantities described in your data table will be linked to elements of the SBML model, via the entries in the columns !SBML:reaction:id and !SBML:species:id. The IDs in these columns must match the IDs chosen in the SBML file.

5. **Can I also use flux data?** Metabolic fluxes do not directly fit into the dependence scheme that parameter balancing uses internally to link different quantitities. Therefore, fluxes cannot be used as input data, nor can they be predicted directly. However, they can be used in an indirect way, as described in [2]. The idea is as follows: In parameter balancing including metabolite levels and reaction Gibbs free energies, known flux directions can be used to define the signs of all reaction Gibbs free energies. The resulting rate laws and metabolite levels will be consistent with the predefined fluxes. The kinetic model, parametrised in this way, and with the balanced metabolite levels, will yield reaction rates with the same signs as the predefined fluxes. By rescaling the Vmax values (i.e., scaling the catalytic constants, enzyme levels, or both), reaction rates and fluxes can be matched. The resulting model will correlate to the predefined flux distribution by construction. Note that, in order for this towork, the predefined fluxes must be thermodynamically feasible (i.e., loop-free, and realisable for the (potentially predefined) external metabolite levels.

6. **Where can I find example files?** A number of example files (SBML models and SBtab data tables for parameter balancing) can be found on www.parameterbalancing.net.

7. **Where can I find suitable input data for my own model?** Typical input data for estimating kinetic parameters comprise catalytic constants (kcat values), Michaelis- Menten constants (KM values), equilibrium constants, standard reaction Gibbs free energies, and Gibbs free energies of formation. Typical input data for estimating metabolic states also comprise metabolite concentrations.

   - A large collection of kinetic data is provided by the BRENDA Enzyme Database [8].
   - Thermodynamic data for many reactions can be obtained from the website eQuilibrator [9]. This comprises calculated equilibrium constants and standard reaction Gibbs free energies for different values of pH and ionic strength. We provide a collection of these data for parameter balancing here.

8. **How can I define or modify the priors on model parameters?** Experimental data alone will usually not suffice to determine all model parameters. To determine underdetermined parameters, and to keep parameters in realistic ranges, parameter balancing uses prior distributions and constraints for each type of parameter. These priors and constraints are defined in a data table, which can be customised by the user. The table is available on www.parameterbalancing.net.

9. **Which kinetic rate laws are assumed in parameter balancing?** Parameter balancing is based on modular rate laws, a generalised version of the convenience kinetics. The modular rate laws include reversible mass-action and reversible Michaelis-Menten rate laws as special cases. Modular rate laws are also supported by SBMLsqueezer [10], which allows you to directly insert rate laws into SBML models. In parameter balancing, rate laws and rate constants can be directly inserted into your model at the end of the workflow. Note that all rate laws previously present in your model will be removed.

10. **What physical units are used in parameter balancing?** The units are predefined in the prior table and cannot be changed, unless you provide your own customised prior table.

11. **How does parameter balancing work mathematically?** Parameter balancing employs Bayesian estimation to determine a consistent set of all model parameters. To use it efficiently, it is good to know about some of its details. For technical reasons, all quantities are internally converted to natural scaling. This means that for energy quantities (in kJ/mol), we keep the original values while for all other quantities, we take the natural logarithms. Furthermore, we distinguish between basic quantities and derived quantities (which are uniquely determined by the basic quantities). See the overview of all quantities considered. During balancing, we integrate information from data (values and standard errors), prior distributions (typical values and spread for basic quantities), and pseudo values (typical values and spread for derived quantities). All these values and spreads are represented by normal distributions (priors, data with standard errors, pseudo values, and posteriors) for the naturally scaled quantities. When converting back to non-logarithmic values, we obtain log-normal distributions, which makes it crucial to distinguish between median and mean values. Eventually,

the median values (which are more realistic and guaranteed to satisfy the relevant constraints) are inserted into the model.

12. **Where is the parameter balancing method described?** Parameter balancing is described in [1]. If you use parameter balancing in your work, please refer to this article. A modelling workflow based on parameter balancing is described in [2].

13. **What are the known caveats?** Parameter balancing makes specific assumptions about the rate laws used. In some cases, this can lead to problems, or parameter balancing may not be suitable for your modelling. Here we list points that can typically lead to problems:

    - **Large models** Large networks lead to large parameter sets, which increase the numerical effort of parameter balancing. One possibility to avoid this problem is to use fixed, precalculated equilibrium constants. Then, the kinetic constants of each reaction can be balanced separately, which reduces the effort. In our code for parameter balancing, there is a restriction on model size to avoid numerical problems.
    - **Biomass reaction or polymerisation reactions** Many metabolic models (especially, models used in flux balance analysis) contain a "biomass" reaction that involves a large number of compounds with largely varying stoichiometric coefficients. Modular rate laws, as assumed in parameter balancing, use the stoichiometric coefficients as exponents in the formula. For biomass reactions or polyerisation reactions, this is not very realistic as assumption. Furthermore, these reactions usually do not have to be thermodynamically consistent. For both reasons, it is advisable to discard the automatically proposed kinetics, and insert a more realistic kinetics instead (see, for instance, the rate laws proposed in [11]).
    - **Very large or small parameter values** Very large or small parameter values can lead to unrealistic models and numerical problems. Extreme values should be avoided by using proper bounds. In any case, we suggest to have a look at all balanced values and to see if they are in realistic ranges.
    - **Large uncertainties and strongly shifted mean values** Each balanced parameter value comes with an uncertainty. The uncertainties are described by normal distributions for the logarithmic parameter values, so median and mean value on logarithmic scale will be identical. For the non-logarithmic values, we obtain a log-normal distribution, and median and mean value will differ. If the uncertainty is large (which can easily happen if a value is not constrained by any data values), the mean and median can become very different, and the mean value can become very high. This can be avoided by reducing the uncertainty range - by providing more data that constraints the parameter value, by using narrower priors or constraints, or by using "pseudo" values.
    - **Annotation of modifiers as activators or inhibitors** To recognise a reaction modifier, listed in an SBML model, as an activator or inhibitor, the parameter balancing code relies on SBO terms in the model. If these terms are missing (which is the case in many existing models), the allosteric regulator will be ignored and no activation or inhibition constant will be estimated.
    - **Missing standard errors in data file** For parameter balancing, each data value must come with a standard error. Missing standard errors are replaced by default values (see question below). In some cases, these default values may lead to unrealistic results. This can be avoided by providing complete data files, with your own estimates of the standard errors.

14. **What if the model cannot be simulated?** Problems in simulating the model (e.g., using COPASI) may be caused by unrealistically high or low parameter values. If you notice such parameter values in your model and would like to avoid them, you may use tighter priors or pseudo values to exclude extreme parameter values.

15. **How are enzymes handled in the balancing?** In SBML, enzymes are sometimes treated as SBML species and sometimes as SBML parameters; this is mainly up to the modeller. In parameter balancing, enzymes need to be parameters. If they are provided as species, they cannot correctly be assigned as reaction modifiers and thus are ignored.

16. **Can I sample data sets from the posterior distribution?** The matlab version of parameter balancing allows for sampling parameter sets. To generate a sample, a parameter set is randomly sampled from the (multivariate Gaussian) posterior. If the sample violates constraints, it is replaced by a parameter vector that satisfies all constraints and is closest to the initially sampled vector, where "closeness" is defined by a quadratic norm based on the posterior covariance matrix. The resulting constrained samples satisfy all constraints and give an impression about the posterior uncertainties, but they do not strictly represent the posterior distribution (which is defined as the multi-variate unconstrained Gaussian posterior, constrained to the feasible region).

17. **What happens to reactions with many reactants?** Reactions with many reactants (e.g., biomass-producing reactions) are not properly described by modular rate laws, and parameters estimated for such reactions should

at least be taken with care. We recommend to remove such reactions from the model before running parameter balancing.

18. **What happens to reactions without any substrate or without any product?** Reactions without any substrate or without any product are not properly described by modular rate laws, and parameters estimated for such reactions should be taken with care. We recommend to remove such reactions from the model before running parameter balancing.

19. **What happens to reactions with unusual stoichiometric coefficients?** Reactions with high stoichiometric coefficients are not properly described by modular rate laws; in reactions with non-integer stoichiometric coefficients, it is likely that these coefficients do not properly describe molecularities. To avoid problems in such cases, our code allows only stoichiometric coefficients of 1 and 2. All other values (non-integer or values larger than 2) are internally replaced by values of 1. This rule is likely to yield realistic balanced parameter sets; however when checking Haldane relationships with these parameters, please not that the molecularities in these Haldane relationships must represent the adjusted (and not the original) stoichiometric coefficients. In case of doubt, we recommend to remove such reactions from the model before running parameter balancing.

20. **What happens to irreversible reactions?** Parameter balancing is designed to assume thermodynamic correctness, which implies reversible rate laws. Some reactions (e.g., macromolecule synthesis or biomass-producing reactions) are practically irreversible. In parameter balancing, these reactions will still be treated as reversible, but will obtain very large equilibrium constants. In some cases, this may lead to numerical problems. It is recommended to remove such reactions from the model before running parameter balancing. Likewise, the rate laws inserted in the SBML model will have a reversible form.

21. **What is the purpose of pseudo values?** In parameter balancing, some types of parameters are treated as basic (e.g., standard chemical potentials), while others are treated as derived (e.g., equilibrium constants). By considering independent marginal priors for all basic parameters, we obtain an uncorrelated prior distribution for the subset of basic parameters, and an ensuing distribution for the derived parameters. However, the variances of derived parameters in this distribution is still large. Therefore, the prior is modified by assuming pseudo values for derived parameters. The result is correlated prior distribution for all (basic and derived) parameter types. In this prior distribution with pseudo values, basic and derived parameters are treated on an equal footing, resulting in realistic variances for all parameters. This is the formula for posterior covariance and posterior mean (without constraints):

$$\begin{aligned} C_{\text{post}} &= \left(C_{\text{prior}}^{-1} + Q_{\text{pseudo}}^{\text{T}}\, C_{\text{pseudo}}^{-1}\, Q_{\text{pseudo}} + Q_{\text{data}}^{\text{T}}\, C_{\text{data}}^{-1}\, Q_{\text{data}}\right)^{-1} \\ \mu_{\text{post}} &= C_{\text{post}} \left(C_{\text{prior}}^{-1}\, \mu_{\text{prior}} + Q_{\text{pseudo}}^{\text{T}}\, C_{\text{pseudo}}^{-1}\, \mu_{\text{pseudo}} + Q_{\text{data}}^{\text{T}}\, C_{\text{data}}^{-1}\, \mu_{\text{data}}\right)^{-1} \end{aligned} \tag{1}$$

For a more compact form, we can set

$$\mu_{\text{pp}} = \begin{pmatrix} \mu_{\text{prior}} \\ \mu_{\text{pseudo}} \end{pmatrix}, \qquad Q_{\text{pp}} = \begin{pmatrix} \mathbf{I} \\ Q_{\text{pseudo}} \end{pmatrix}, \qquad C_{\text{pp}} = \begin{pmatrix} C_{\text{prior}} & 0 \\ 0 & C_{\text{pseudo}} \end{pmatrix}, \tag{2}$$

and obtain the formulae

$$\begin{aligned} C_{\text{post}} &= \left(Q_{\text{pp}}^{\text{T}}\, C_{\text{pp}}^{-1}\, Q_{\text{pp}} + Q_{\text{data}}^{\text{T}}\, C_{\text{data}}^{-1}\, Q_{\text{data}}\right)^{-1} \\ \mu_{\text{post}} &= C_{\text{post}} \left(Q_{\text{pp}}^{\text{T}}\, C_{\text{pp}}^{-1}\, \mu_{\text{pp}} + Q_{\text{data}}^{\text{T}}\, C_{\text{data}}^{-1}\, \mu_{\text{data}}\right)^{-1}. \end{aligned} \tag{3}$$

Please note that pseudo values are not a *replacement* for missing data values: on the contrary, a pseudo value and a data value for the same parameter will be used at the same time.

22. **Enzymes as model species** The parameter balancing tool assumes that the SBML model contains all reactions and metabolites, but not the catalysing enzymes. If enzymes appear explicitly in the model as ¡species¿, they need to be tagged as enzymes by an SBO term. Otherwise, they will be treated as metabolites. In the parameter balancing results, this may lead to redundant (and contradictory) results, defining a concentration of the enzyme, and a (contradictory) concentration of enzyme of the enzyme-catalysed reaction.

23. **How can I impose lower or upper bounds on data values?** For each parameter type (e.g. substrate catalytic constant), lower and upper bounds are defined in the (modifiable) pb_prior file. Lower and upper bounds for individual parameters (e.g., the substrate catalytic constant of a specific reaction in the model), lower and upper bounds can be defined in the data file. Specifically in the case of catalytic constants, lower bounds could derived from measured flux and proteomics data, and an upper bound is given by the diffusion limit.

24. **How are the data values preprocessed?** When the data file is read, some checks are performed and changes are made in the following order. (i) If a zero value is given for a multiplicative quantitity, this value is ignored. (ii) If a data value is outside the allowed bound for this type of parameter, it is ignored. (iii) If a zero standard deviation is given for a value, this standard deviation is ignored. (iv) If a data value has no standard deviation, a default standard deviation (column DataStd from the prior file) is inserted. In the matlab version, it is also possible to use default geometric standard deviations (column DataGeomStd from the prior file) instead. (v) If several data values are given for the same parameter (e.g., several values for the same equilibrium constant), the arithmetic mean of these values is used as the final data value (both for additive and multiplicative quantities), and the arithmetic mean of the standard deviations is used as the final standard deviation. After this procedure, each model parameter has either no or one data value, and a data value is characterised by arithmetic mean and standard deviation). For multiplicative parameters, the arithmetic mean and standard deviation (for an assumed log-normal distribution) and then translated into the arithmetic mean and standard deviation of the logarithmic parameter values (assumed to follow a normal distribution). After the conversion, all standard deviations smaller than 0.001 are replaced by 0.001 to avoid numerical problems.

25. **What happens if the data file contains measured values without standard errors?** If standard errors are missing in the data file, the missing standard deviations are replaced by default values, which depend on the type of parameter and are read from the column !DataStd in the prior information file. Note that, in this case, smaller or larger data values will end up having the same standard deviations (and therefore, very different relative standard deviations). In the matlab version (and again, in the case of missing standard deviations), the user can choose to assume the same *relative* standard deviations instead. If this option is chosen, the algorithm uses the numbers from the column !DataGeometricStd (containing default geometric standard deviations) to determine a standard deviation specifically for each data value.

6 Contact

Parameter balancing for kinetic metabolic models has been developed at Humboldt Universität zu Berlin (Theoretical Biophysics), Charité Universitätsmedizin Berlin (Institute of Biochemistry), and INRA Jouy-en-Josas (Unité MaIAGE) by Timo Lubitz and Wolfram Liebermeister, with help and support by Marvin Schulz, Falko Krause, Edda Klipp, and Elad Noor.

We are happy to hear from you. For questions or feedback, please contact Wolfram Liebermeister (wolfram.liebermeister@gmail.com) and Timo Lubitz (timo.lubitz@gmail.com).

References

[1] T. Lubitz, M. Schulz, E. Klipp, and W. Liebermeister. Parameter balancing for kinetic models of cell metabolism. *J. Phys. Chem. B*, 114(49):16298–16303, 2010.

[2] N.J. Stanford, T. Lubitz, K. Smallbone, E. Klipp, P. Mendes, and W. Liebermeister. Systematic construction of kinetic models from genome-scale metabolic networks. *PLoS ONE*, 8(11):e79195, 2013.

[3] E. Noor, A. Flamholz, A. Bar-Even, D. Davidi, R. Milo, and W. Liebermeister. The protein cost of metabolic fluxes: prediction from enzymatic rate laws and cost minimization. *PLoS Comput Biol*, 12(10):e1005167, 2016. doi:10.1371/journal.pcbi.1005167.

[4] M.T. Wortel, E. Noor, M. Ferris, F.J. Bruggeman, and W. Liebermeister. Metabolic enzyme cost explains variable trade-offs between microbial growth rate and yield. *PLoS Computational Biology*, 14(2):e1006010, 2018.

[5] W. Liebermeister and E. Klipp. Bringing metabolic networks to life: convenience rate law and thermodynamic constraints. *Theor. Biol. Med. Mod.*, 3:41, 2006.

[6] W. Liebermeister, J. Uhlendorf, and E. Klipp. Modular rate laws for enzymatic reactions: thermodynamics, elasticities, and implementation. *Bioinformatics*, 26(12):1528–1534, 2010.

[7] R.A. Alberty. Biochemical thermodynamics. *Biochem. Biophys. Acta*, 1207:1–11, 1994.

[8] I. Schomburg, A. Chang, C. Ebeling, M. Gremse, C. Heldt, G. Huhn, and D. Schomburg. BRENDA, the enzyme database: updates and major new developments. *Nucleic Acids Research*, 32:Database issue:D431–433, 2004.

[9] A. Flamholz, E. Noor, A. Bar-Even, and R. Milo. eQuilibrator – the biochemical thermodynamics calculator. *Nucleic Acids Research*, 40(D1):D770–D775, 2012.

[10] A. Dräger, N. Hassis, J. Supper, A. Schröder, and A. Zell. SBMLsqueezer: a CellDesigner plug-in to generate kinetic rate equations for biochemical networks. *BMC Systems Biology*, 2:39, 2008.

[11] J.S. Hofmeyr, O.P.C. Gqwaka, and J.M. Rohwer. A generic rate equation for catalysed, template-directed polymerisation. *FEBS Letters*, 587:2868–2875, 2013.