Journal of
open research software

PBTK Optimizer: An Open Source Application for PBTK Model Parameter Optimization in Python

SOFTWARE
METAPAPER

IAN EDHLUND
MATTHEW MACAULEY
CINDY LEE
*Author affiliations can be found in the back matter of this article

ubiquity press

ABSTRACT

Physiologically based toxicokinetic (PBTK) modeling in fish offers great promise in environmental risk assessment, potentially speeding up dose-response studies while minimizing animal testing. PBTK models are generally written as ordinary differential equations (ODEs) and have recently been modeled with Petri nets. Some limitations exist in the PBTK field, such as difficulty of model development and a lack of application specific software tools to help modelers. To address some of these limitations we introduce PBTK Optimizer, an open source tool for optimizing parameters of Petri net PBTK models. The software is demonstrated with a previously published and validated PBTK model of fluoranthene exposure in rainbow trout. We present case studies using PBTK Optimizer to evaluate different parameters of the model. The Python code and conclusions regarding the optimization methods used in this software may be adapted for ODE applications beyond PBTK modeling.

CORRESPONDING AUTHOR:
Ian Edhlund
Clemson University, US
iedhlun@g.clemson.edu

KEYWORDS:
differential equations; ODE model; physiologically based toxicokinetic modeling; PBTK; optimization; Petri net; Python

TO CITE THIS ARTICLE:
Edhlund I, Macauley M, Lee C 2021 PBTK Optimizer: An Open Source Application for PBTK Model Parameter Optimization in Python. *Journal of Open Research Software*, 9: 4. DOI: *https://doi.org/10.5334/jors.285*

(1) OVERVIEW

INTRODUCTION

Physiologically based toxicokinetic (PBTK) modelling helps risk assessors determine the bioaccumulation potential for waterborne anthropogenic compounds in fish [1]. The models simulate the absorption, distribution, metabolism, and excretion (ADME) of the compound and predict the concentration of the compound and its toxic metabolites in a specific tissue of concern where the toxic effect takes place. Since the initial application of PBTK models in fish, additional routes of exposure, compounds, fish species, etc., have been added to the literature [2–5].

PBTK models in the literature are typically in the framework of ordinary differential equations (ODEs). Users of these models are required to reimplement the ODEs in their preferred language or software, based on the description in the literature. In contrast, a Petri net (PN) is a discrete event dynamical system. A continuous PN, which incorporates vectors into the PN system, has been proposed as a method for PBTK modeling and distribution [6]. The PN PBTK models rely on open source software, such as Snoopy [7], for development, simulation, and distribution. Snoopy is a specific software for PNs, so it does not have additional features that may be useful for PBTK models, such as Monte Carlo simulations or parameter fitting routines.

Many parameters in PBTK models can be measured in vivo or are available in the literature. The mass of tissues, often given in volumes using 1.0 kg/L, can be weighed in a given specimen or may be taken from a literature review. Blood flows and ventilation volume have also been determined experimentally for certain species [8]. Biotransformation parameters have been previously measured in vitro and extrapolated for PBTK models for various compounds in certain species [9]. However, it may be impractical to measure all parameters for a specific species and compound of interest. To that end, equations to extrapolate parameters from one species or compound to another have been presented [5]. Even so, at times it may be beneficial to optimize specific parameters when measured data is unavailable or extrapolated data is unreliable.

A typical goal of optimization in PBTK modeling is for simulated time-course mass of a compound in various tissues to closely match experimental values. In mathematical terms, the goal is to find the least residual between the simulation and evaluation data. This fit can be achieved by changing or optimizing the parameters for the model. Several optimization algorithms, which may help to achieve this goal, have been previously presented in the literature and have been implemented for Python in the LmFit package [10].

The most straightforward optimization method is the grid search method, called 'brute' by LmFit. This method uses several values for each parameter to be optimized, with a small step size in between. Every possible combination of parameter values is simulated, which can be very intensive with increasing numbers of parameters to be optimized. Gradient based, Newton, and quasi-Newton optimization methods can be more efficient in determining the minimum residual values, using gradients, Jacobians, and/or Hessians. These methods have been implemented either individually or combined by LmFit in the Levenberg-Marquadt (leastsq), Truncated Newton (tnc), Limited-Memory BFGS (lbfgsb), Sequential Least Squares (slsqp), and Conjugate Gradient (cg) methods. Powell's constrained optimization by linear approximation (cobyla) method is also included. Recent advancements in the field of mathematical optimization include evolutionary and Markov Chain Monte Carlo methods, which are implemented by LmFit as differential_evolution and emcee, respectively.

The goal of the current development effort is to provide an open source software for parameter fitting in PBTK models within the Petri net framework.

IMPLEMENTATION AND ARCHITECTURE

PBTK Optimizer was developed in Python. The intended audience is diverse, with varying levels of programming experience; therefore, a graphical user environment was chosen, using tkinter [11]. The LmFit library was chosen for the optimization routines.

The tabs in PBTK Optimizer (pbtkoptimizer.py) are designed to be accessed from left-to-right. First, parameters are defined, followed by model definitions (and compartments chosen for evaluation). Next, a file of evaluation results is chosen. Finally, an optimization routine is selected and run.

The following paragraphs describe the use of PBTK Optimizer and algorithms behind the pbtkoptimizer.py code. Readers may find it helpful to reference pbtkoptimizer.py while reading through this section. Case studies, along with screenshots of the software, follow in the quality control section.

The 'Pick Params File' button on the 'Parameter Definition' tab allows a user to select a comma separated value (CSV) file of all the parameters used in the model. The CSV file should have a title row (which is ignored by the 'paramsbtnclick()' function). Each column in the CSV file should be placed in the order shown in ***Table 1*** (complete parameter files are shown in the supplemental material). Each column is necessary, if only as a placeholder, even when there are no values entered in the column.

For each parameter listed, 'Name' and 'Value' are required fields. 'Vary' is also required and directs the script whether to optimize the parameter (TRUE) or leave it at its initial value (FALSE). None of the remaining fields are required (null values are acceptable), except for the 'differential_evolution' optimization which requires 'Min' and 'Max'. 'Min' and 'Max' may be used to place limits when 'Vary' is set to 'TRUE' (and ignored when 'Vary' is set to 'FALSE').

NAME	VALUE	MIN	MAX	STDERR	VARY	EXPR	BRUTE_STEP
Water	20				FALSE		
Kk	0.023	0			TRUE		
Kl	0.038	0			TRUE		
Kr	0.0189	0			TRUE		
Pb	5.13				FALSE		

Table 1 Example Parameters File.

If the standard error of a parameter is known, it may be entered into 'Stderr'. 'Expr' is used to define constraints to optimization. For example, 'Expr' may be used to require that the sum of blood flows equals cardiac blood flow. 'Brute_Step' is a parameter to override the default used in the 'brute' optimization algorithm. The 'Stderr', 'Expr', and 'Brute_Step' fields were not tested in this study.

Once the properly formatted parameters file has been selected, the software loads the set of parameters into an array of 'Parameter' objects, and a table of the parameters will be shown in the 'Parameter Definition' tab. 'Min' will be set to -infinity by default and 'Max' set to infinity. 'TRUE' will show as '1' under 'Vary'; conversely, 'FALSE' will show as '0'. The user may now proceed to the 'Model Definition' tab.

The 'Pick Model File' button on the 'Model Definition' tab allows the user to select a system of ODEs from a text file, formatted according to the following specifications. The differential equation to determine the mass of contaminant in each compartment is written with a preceding 'dc_' and a trailing '_/dt'. For example, the left-hand term for the Fat (F) compartment is 'dc_F_/dt'. When used elsewhere (i.e., as part of the calculation for another compartment), the compartments are labeled with a preceding 'c_' and a trailing '_'. For example, the Fat compartment is labeled 'c_F_'. Likewise, parameters are labeled with a preceding 'p_' and a trailing '_'. For example, the volume of the fat compartment (with name 'Vf' from the parameters file), is labeled 'p_Vf_'. The leading and trailing characters allow the script to parse the ODE and change it into an equation for optimization. Equation 1 is an example equation from the system of ODEs (which is attached in the supplementary materials).

$$dc\_F\_/dt = (c\_A\_/p\_Vblood\_*p\_Qf\_)-((c\_F\_*p\_Qf\_)/(p\_Vf\_*p\_Pf\_)) \quad (1)$$

The PBTK model used as a case study was written as a Petri net in the software "Snoopy" [6], which can export a Petri net as an ODE. When parameters are named with 'p_' and '_' and compartments are named with 'c_' and '_', then the exported ODE text file will be properly formatted for use in PBTK Optimizer. The Petri net approach also ensures conservation of mass within the PBTK model and gives the user greater confidence of syntactically correct ODEs.

Once the model file is chosen, the 'modelbtnclick()' function loads the system of ODEs into the 'ODEtext' dictionary so they can later be evaluated using Python's 'eval()' function [12]. The system of ODEs is then displayed on the 'Model Definition' tab, where a checkbox is displayed for each compartment. If checked, the software expects a corresponding set of time course evaluation points for that compartment. This allows for instances when a compartment is required for the rest of the system to evaluate, but there are no corresponding evaluation measurements for that compartment. At least one checkbox must be checked to optimize the model. More evaluation measurements are encouraged for greater confidence in the optimization routines. After checking the appropriate checkboxes, the user may proceed to the 'Validation Results' tab.

The 'Pick Valid File' button on the 'Validation Results' tab allows the user to select a CSV file of time course data for each compartment checked in the previous tab. Example evaluation data is show in ***Table 2***. The file should begin with a header row, which shows the sampling time points. The first cell is left blank. Each following row comprises the sampling data for each compartment. The first cell is the compartment name, which should match with the compartment name defined in the model (less 'c_' and '_'). Each following cell is the sample data for the time point in that compartment.

After the evaluation file is selected in 'pickvalid()', a list of 'compartments' and a NumPy array, 'expdata', of the corresponding evaluation data are created [13]. A table of the evaluation results is also shown in the 'Validation Results' tab. The user is now ready to proceed to the 'Parameter Optimization' tab.

On the 'Parameter Optimization' tab, the user may select from a dropdown list of optimization methods available in the LmFit library. Some of the methods from LmFit are not included in the list, as they required additional functions in order to execute. When the appropriate optimization method is selected, the user presses the 'Optimize' button.

Several auxiliary functions are necessary for optimization to occur. The 'f()' function converts the 'ODEtext' dictionary and a set of parameters into a Python expression which can be passed to the 'g()' function for ODE simulation. The 'residual()' function compares the

	6	12	36	48	96	152	218
B	0.112842	0.165171	0.282806	0.312988	0.517278	0.730571	0.786306
P	12.312	15.225	34.0144	32.7085	68.9017	66.5103	55.1602
F	9.71499	32.2161	89.6937	127.512	364.075	478.184	548.17
G	0.055517	0.072156	0.091274	0.11179	0.147148	0.334804	0.225093
L	0.252665	0.406172	0.74976	0.784167	1.23001	2.00371	2.09576
R	1.9376	3.69466	12.6803	19.1182	33.6874	39.8261	43.6188
K	0.206384	0.293213	0.565799	0.587173	0.956094	1.20586	1.4937

Table 2 Example Evaluation Data.

results of the 'g()' simulation ('model') to the results from the evaluation ('data'), ignoring those compartments which do not have their checkbox selected in the 'Validation Results' tab.

The 'Optimize' button calls the 'pobtn1click()' function, which begins by transposing the 'expdata' array to align the data in time sequence. The 'minimize()' function from the LmFit library is called with the 'residual()' function, the 'params' object list, the evaluation data, and the selected optimization method. The 'minimize()' function passes the 'residual()' function an updated list of parameters for each iteration of the optimizing routine. The 'residual()' function then compares the results of the ODE system, solved in the 'g()' function, given the updated parameter list, to the evaluation results. The 'minimize()' function continues to update the parameter list and run the residual function until the minimum residual is found.

Upon completion of the 'minimize()' function, a plot of the model is generated using the 'matplotlib' library [14]. Lines show the modeled time course data; evaluation data for each compartment at each time point is shown as a circle. The optimized list of parameters and values are shown to the right of the plot. 'StdErr' of optimized parameters are shown when certain optimization algorithms are selected; other algorithms do not generate StdErr results.

More information about the optimization execution is shown below the plot. Notably, the Chi-Square, Reduced Chi-Square, Akaike Information Criterion (AIC), and Bayesian Information Criterion (BIC), are shown. These values may help a user to select between models and determine the confidence or applicability of the model. The 'Save Output' button provides the user an opportunity to export the pertinent information into a CSV file.

QUALITY CONTROL

We used a case study approach to demonstrate the PBTKOptimizer software. In the case study, we examined the optimization of biotransformation equations in a previously published PBTK model.

In the Smith (2003) PBTK model, from which the Edhlund and Lee (2019) model was based, the parameters which account for biotransformation of fluoranthene in the liver, kidneys, and rapidly perfused tissues (RPT) were not measured experimentally [6, 15]. Instead, they were inferred from literature reviews of other PAHs. We used PBTK Optimizer to evaluate whether we could improve on Smith's (2003) estimates. We optimized the first order and Michaelis-Menten rate constants (k and Km) and maximum enzymatic rate (Vmax) for each of the metabolizing compartments in the Petri net model to the Smith (2003) in vivo evaluation results.

In further trials, we set the Michaelis-Menten rate constants to zero to evaluate the model using only first order rates. It has been previously shown that biotransformation follows first-order kinetics when substrate concentrations are low and Michaelis-Menten kinetics when substrate concentrations are high [9].

a. Parameters File

The parameters file, params_a.csv, is shown in the supplementary material. We set the 'Vary' property to 'True' and the 'Min' property to '0' for the k, Km, and Vmax parameters in the metabolizing compartments. The 'Max' property was left empty to allow greater freedom for the optimization methods. The 'Max' property is only required for the differential evolution optimization method, which was unable to complete due to errors. 'Vary' was set to 'False' for the remaining parameters. In the subsequent runs using only first order rates, 'Vary' was set to 'False' for the Km and Vmax parameters.

b. Model File

The Petri net model file, PN_Trout.spcontped, from Edhlund and Lee (2019) was retrieved from *https://figshare.com/articles/A_Petri_Net_Approach_to_PBTK_Modeling/6057719* and edited with Snoopy, which is available for download from *http://www-dssz.informatik.tu-cottbus.de/DSSZ/Software/Snoopy*. The parameter and compartment names were changed, as described above, and saved as PN_Trout_Opt.spcontped. In addition, PBTK_ODEs.txt, was exported as a text ODE file. PBTK_ODEs.txt is shown in the supplementary materials, PN_Trout_Opt.spcontped is available on Github. When run in PBTK Optimizer, the checkboxes for all compartments, other than ART and VEN were selected. The 'Model Definition' tab is shown in ***Figure 1***.

c. Evaluation File

Measured concentrations in each tissue for the in vivo exposures were provided by Smith (2003). The concentrations were averaged and converted to mass units. The time-course fluoranthene mass in each tissue was then formatted as described above and saved as valid_a.csv, which is available online and shown in the supplemental material. A screenshot of the 'Validation Results' tab is shown in ***Figure 2***.

d. Optimization

PBTK Optimizer was run once for each optimization method. The values of the nine optimized parameters, AIC, and number of evals were recorded for each optimization method. The results are shown in ***Table 3***. The subsequent runs, which only varied the first order rate constants and excluded the Michaelis-Menten kinetics are summarized in ***Table 4***. A screenshot of the 'Parameter Optimization' tab is shown in ***Figure 3***.

DISCUSSION

When optimizing Michaelis-Menten and first order kinetics together, the ratio of number of parameters to be optimized per evaluation data points was very high. This created a difficult scenario for the optimization methods, with the majority of the methods resorting to the initial values. The cobyla and slsqp methods found different optimized value sets. The lower AIC found by the cobyla method was obtained by moving nearly all of the metabolism to the kidney compartment, which is not physiologically likely. The AIC found by the slsqp method was much higher than the other methods. The brute and differential_evolution methods errored out. Processing of the emcee method was cancelled by the user after 24 hours.

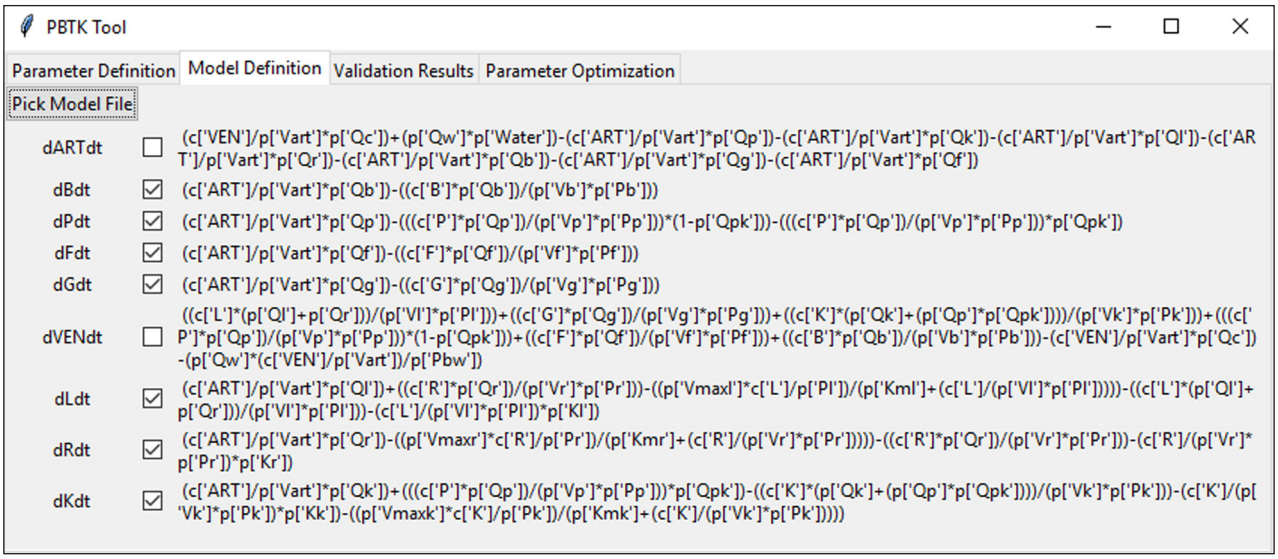

Figure 1 PBTK Optimizer Model Definition Tab.

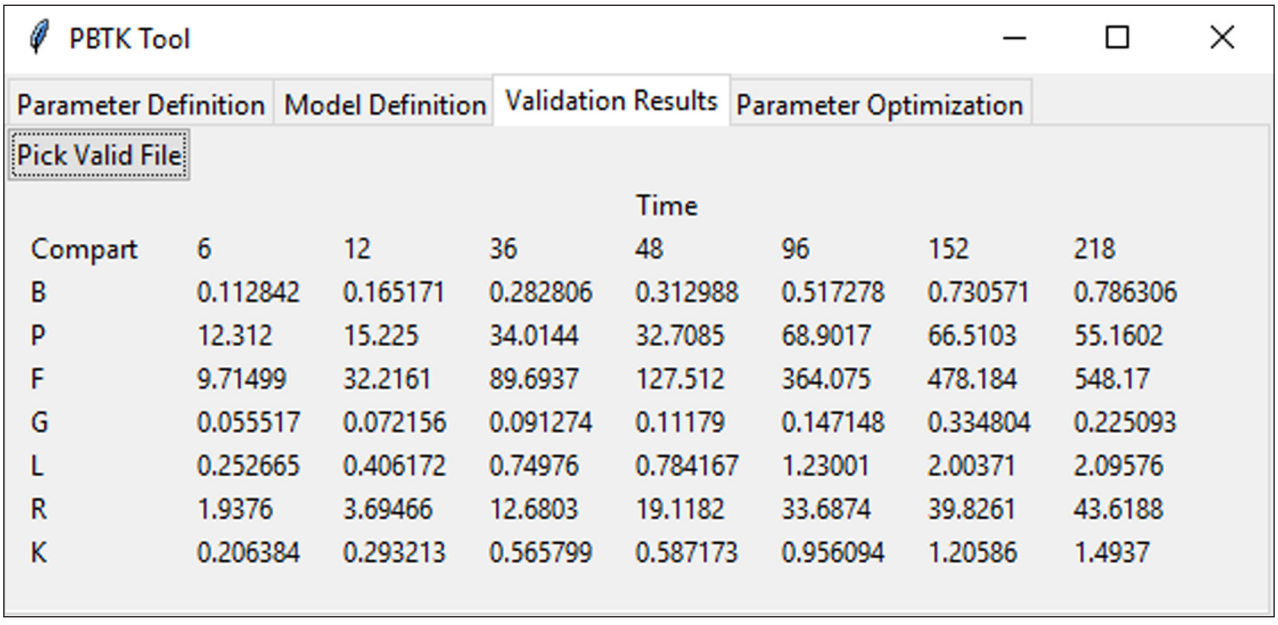

Figure 2 PBTK Optimizer Validation Results Tab.

OPT METH	AIC	RED CHI-SQ	#EVALS	KK	KL	KR	KMK	KML	KMR	VMAXK	VMAXL	VMAXR
Initial				0.0230	0.0380	0.0189	5582	3050	1525	12400	6790	3395
leatfsq	351	1093	34	0.0230	0.0380	0.0189	5582	3050	1525	12400	6790	3395
lbfgsb	351	1093	821	0.0230	0.0380	0.0189	5582	3050	1525	12400	6790	3395
cg	351	1093	1639	0.0230	0.0380	0.0189	5582	3050	1525	12400	6790	3395
cobyla	350	1062	2888	0.0743	0.0001	0.0000	5583	3051	1526	12400	6790	3395
tnc	351	1093	6	0.0230	0.0380	0.0189	5582	3050	1525	12400	6790	3395
slsqp	449	8148	821	3929	1427	1695	10161	716	2394	13703	416	3465
brute	No Result											
diff_ev	No Result											
emcee	No Result											

Table 3 Results – Michaelis-Menten and First Order Kinetics.

OPTIMIZATION METHOD	AIC	RED CHI-SQ	#EVALS	KK	KL	KR
Initial				0.0230	0.0380	0.0189
leastsq	342	1003	34	0.0761	0.0002	0.0266
lbfgsb	339	952	193	0.0294	0.0408	0.0234
cg	339	950	279	0.0306	0.0423	0.0192
cobyla	338	933	4001	0.0968	0.0084	0.0001
tnc	339	949	70	0.0307	0.0430	0.0187
slsqp	339	953	153	0.0000	0.3846	0.0067
brute	No Result					
cfiff_ev	339	946	1072	0.0259	0.0737	0.0071
emcee	338	929	372	0.1253	0.0000	0.0000

Table 4 Results – Only First Order Kinetics.

The optimization results for optimizing only First Order kinetics were more successful. The lbfgsb, cg, and tnc methods found similar minima in proximity to the initial values, which suggests that these are the most biologically relevant and preferred values. The leastsq method found a suboptimal condition with more metabolism occurring in the kidney and nearly none in the liver. Similarly, cobyla and emcee found suboptimal conditions with nearly all the metabolism occurring in the kidney. On the other hand, slsqp found a suboptimal condition with nearly all metabolism occurring in the liver. Differential_evolution found a unique condition where metabolism occurs in the liver and kidney, but not the rpt.

The optimization methods were forced to make a compromise between K values in different metabolizing compartments, with the only evaluation data being the mass of fluoranthene in those compartments. If additional evaluation data were available to direct the amount of biotransformation by each compartment, it is possible that more of the optimization methods would have found similar results. Techniques have been previously used to determine kidney versus liver clearance of PAHs in living trout [16]. This type of data could be added to the model with the addition of a compartment to capture the outflow of the metabolizing compartments. Another option would be to make use of the 'Expr' property in the K parameters to approximate the ratios of metabolism in the three metabolizing compartments. A further possibility would be to make more directed use of the 'Min' and 'Max' properties in the K parameters. The limits were intentionally set to allow for full exploration of the optimization methods. If these limits were set to biologically relevant numbers, i.e., through the use of in vitro isolated hepatocyte experiments [9], closer results between the optimization methods might be found.

The behaviour of PBTKOptimizer is not unique to this software, rather it is common to optimization algorithms in general. The algorithms work best when the number of parameters to optimize are low and the number of data points to fit are high. PBTKOptimizer, like any data fitting software, relies on the quality of the data and the experimental design.

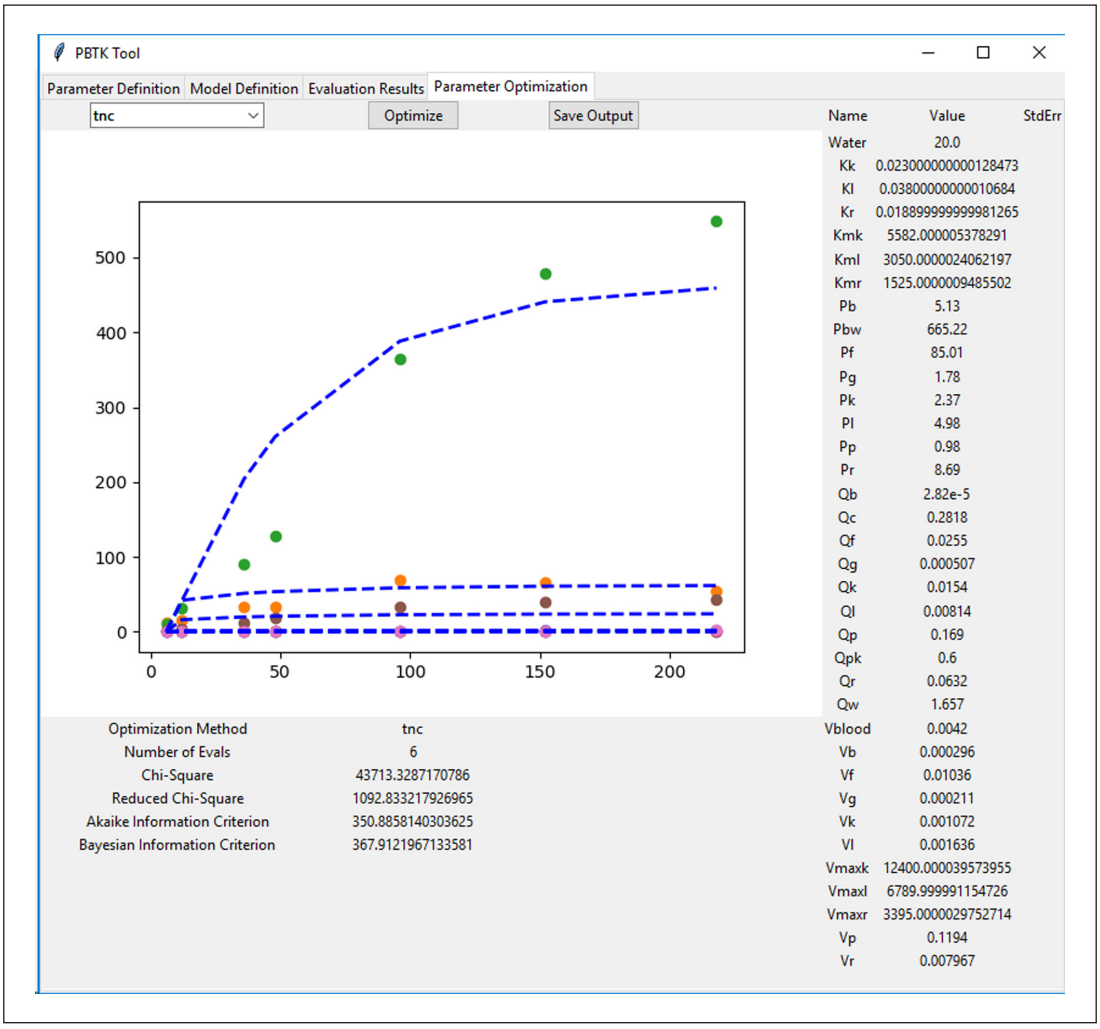

Figure 3 Parameter Optimization Tab.

FUTURE DEVELOPMENT/REUSE

PBTKOptimizer makes use of previously implemented packages for parameter optimization. At its heart, it is a tool to optimize parameters in a set of ordinary differential equations to previously established data points. As such, the code supplied could be adapted to any number of ODE parameter fitting scenarios.

The brute optimization method was never successful. Its errors suggest that not all of the necessary parameters were being passed. Contact with the LmFit developers may be necessary to debug the method. However, since its efficiency is questionable, further effort was suspended. This method may offer promise, however, when combined with multi-threading, i.e., cluster computing.

The Parameter optimization tab shows the model results compared to mean in-vivo data at each time point in each compartment. Future versions of the software should include bars to show the variation in in-vivo data, as shown in ***Figure 4***.

Parameter optimization is just one of the computational tasks required of PBTK model development. PBTK modellers may also find themselves with other computational tasks such as Monte Carlo simulation for population modelling. PBTKOptimizer is envisioned as one piece of a suite of tools necessary for PBTK modelling.

(2) AVAILABILITY

OPERATING SYSTEM

PBTKOptimizer has been tested on 64 bit workstations running Windows 10 and Linux Mint 20.

PROGRAMMING LANGUAGE

Python 3.6.4

ADDITIONAL SYSTEM REQUIREMENTS

PBTKOptimizer was tested on a 64 bit laptop with 8GB of RAM and a 2.60GHz Intel(R) Core(TM) i5–7300U CPU processor. It is possible, but not guaranteed that the software will work with lesser hardware.

DEPENDENCIES

Python was loaded with the following packages:
lmfit 0.9.8
matplotlib 2.1.2
numpy 1.14.1
scipy 1.0.0

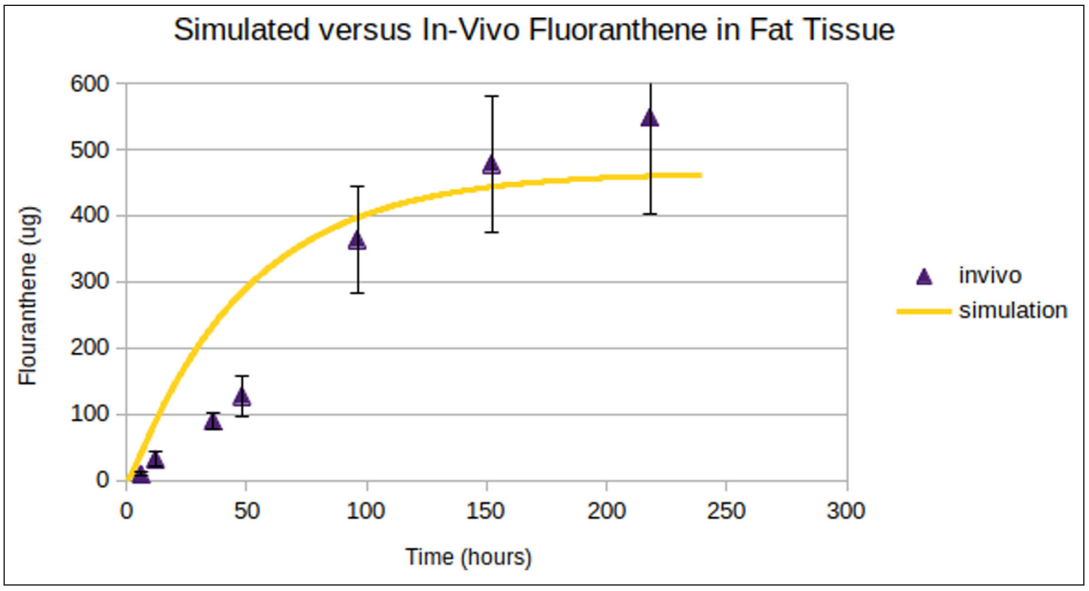

Figure 4 Simulated versus In-Vivo Results with Variation.

LIST OF CONTRIBUTORS

Ian Edhlund wrote the PBTK Optimizer software.

SOFTWARE LOCATION

Archive

Name: Zenodo

Persistent identifier: *10.5281/zenodo.3275923*

Licence: GNU General Public License version 3.0 (GPLv3)

Publisher: Ian Edhlund

Version published: 1.0

Date published: 07/09/19

Code repository

Name: GitHub

Identifier: *https://github.com/ianedhlund/PBTKOptimizer/*

Licence: GNU General Public License version 3.0 (GPLv3)

Date published: 07/09/19

LANGUAGE

English

(3) REUSE POTENTIAL

PBTKOptimizer makes use of previously implemented packages for parameter optimization. At its heart, it is a tool to optimize parameters in a set of ordinary differential equations to previously established data points. As such, the code supplied could be adapted to any number of ODE parameter fitting scenarios.

The source code for PBTKOptimizer is available on GitHub. Ongoing support of the software is not provided.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR AFFILIATIONS

Ian Edhlund *orcid.org/0000-0003-3122-9759*
Clemson University, US

Matthew Macaule *orcid.org/0000-0002-4409-2248*
Clemson University, US

Cindy Lee *orcid.org/0000-0003-4058-8251*
Clemson University, US

REFERENCES

1. **Nichols JW, McKim JM, Andersen ME, Gargas ML, Clewell HJ, Erickson RJ.** A physiologically based toxicokinetic model for the uptake and disposition of waterborne organic chemicals in fish. *Toxicology and Applied Pharmacology*. 1990 Dec 1; 106(3): 433–47. DOI: *https://doi.org/10.1016/0041-008X(90)90338-U*
2. **Brinkmann M, Schlechtriem C, Reininghaus M, Eichbaum K, Buchinger S, Reifferscheid G,** et al. Cross-species extrapolation of uptake and disposition of neutral organic chemicals in fish using a multispecies physiologically-based toxicokinetic model framework. *Environmental Science & Technology*. 2016 Feb 16; 50(4): 1914–23. DOI: *https://doi.org/10.1021/acs.est.5b06158*
3. **McKim JM, Nichols JW, Lien GJ, Hoffman AD, Gallinat CA, Stokes GN.** Dermal absorption of three waterborne chloroethanes in rainbow trout (Oncorhynchus mykiss) and channel catfish (Ictalurus punctatus). *Toxicological Sciences*. 1996; 31(2): 218–228. DOI: *https://doi.org/10.1093/toxsci/31.2.218*
4. **Nichols JW, Fitzsimmons PN, Whiteman FW, Dawson TD, Babeu L, Juenemann J.** A physiologically based toxicokinetic model for dietary uptake of hydrophobic organic compounds by fish I. Feeding studies with 2,2′,5,5′-tetrachlorobiphenyl. *Toxicol Sci*. 2004 Feb 1; 77(2): 206–18. DOI: *https://doi.org/10.1093/toxsci/kfh033*
5. **Stadnicka J, Schirmer K, Ashauer R.** Predicting concentrations of organic chemicals in fish by using

toxicokinetic models. *Environmental Science & Technology*. 2012 Mar 20; 46(6): 3273–80. DOI: *https://doi.org/10.1021/es2043728*

6. **Edhlund I, Lee C.** A Petri Net approach to physiologically based toxicokinetic modeling. *Environmental Toxicology and Chemistry*. 2019; 38(5): 978–87. DOI: *https://doi.org/10.1002/etc.4390*
7. **Heiner M, Herajy M, Liu F, Rohr C, Schwarick M.** Snoopy – a unifying Petri Net tool. In: Haddad S, Pomello L (eds.), *Application and Theory of Petri Nets*. Berlin, Heidelberg: Springer; 2012. p. 398–407. (Lecture Notes in Computer Science). DOI: *https://doi.org/10.1007/978-3-642-31131-4_22*
8. **Erickson RJ, McKim JM.** A simple flow-limited model for exchange of organic chemicals at fish gills. *Environmental Toxicology and Chemistry*. 1990 Feb 1; 9(2): 159–65. DOI: *https://doi.org/10.1002/etc.5620090205*
9. **Nichols JW, Schultz IR, Fitzsimmons PN.** In vitro–in vivo extrapolation of quantitative hepatic biotransformation data for fish: I. A review of methods, and strategies for incorporating intrinsic clearance estimates into chemical kinetic models. *Aquatic Toxicology*. 2006 Jun 10; 78(1): 74–90. DOI: *https://doi.org/10.1016/j.aquatox.2006.01.017*
10. **Newville M, Stensitzki T, Allen DB, Ingargiola A.** LMFIT: non-linear least-square minimization and curve-fitting for Python. *Zenodo* [Internet]. 2014 Sep 21 [cited 2018 Dec 4]; Available from: *https://zenodo.org/record/11813#.XAa9ueJRdhE*
11. **Python Software Foundation.** tkinter [Internet]. 2018 [cited 2018 Dec 4]. Available from: *https://docs.python.org/3/library/tkinter.html*
12. **Python Software Foundation.** eval [Internet]. 2018 [cited 2018 Dec 4]. Available from: *https://docs.python.org/3/library/functions.html#eval*
13. **Oliphant T.** *A Guide to NumPy*. USA: Trelgol Publishing; 2006.
14. **Hunter JD.** Matplotlib: A 2D Graphics Environment. *Computing in Science Engineering*. 2007 May; 9(3): 90–5. DOI: *https://doi.org/10.1109/MCSE.2007.55*
15. **Smith CA.** The Development, Evaluation and Application of a Physiolgically-based Toxicokinetic Model for Fluoranthene in Rainbow Trout (Onchorhyncus Mykiss) [Internet]. Oxford, OH: Miami University; 2003 [cited 2016 Aug 22]. Available from: *http://rave.ohiolink.edu/etdc/view?acc_num=miami1068216356*
16. **Kennedy CJ.** Toxicokinetic studies of chlorinated phenols and polycyclic aromatic hydrocarbons in rainbow trout (Oncorhynchus mykiss) [PhD Dissertation]. Simon Fraser University; 1990.

TO CITE THIS ARTICLE:
Edhlund I, Macauley M, Lee C 2021 PBTK Optimizer: An Open Source Application for PBTK Model Parameter Optimization in Python. *Journal of Open Research Software*, 9: 4. DOI: *https://doi.org/10.5334/jors.285*

Submitted: 09 July 2019 Accepted: 17 February 2021 Published: 14 April 2021