Title

Berkeley Madonna Version 10-A simulation package for solving mathematical models.

Journal

CPT: Pharmacometrics & Systems Pharmacology, 11(3)

Authors

Marcoline, Frank
Furth, John
Nayak, Smita
et al.

Publication Date

2022-03-01

DOI

10.1002/psp4.12757

Received: 24 September 2021 | Revised: 24 September 2021 | Accepted: 15 October 2021

DOI: 10.1002/psp4.12757

TUTORIAL

Berkeley Madonna Version 10–A simulation package for solving mathematical models

Frank V. Marcoline | John Furth | Smita Nayak | Michael Grabe | Robert I. Macey[†]

Berkeley Madonna, Albany, California, USA

Correspondence
Michael Grabe, Berkeley Madonna, 1025 Peralta Ave, Albany, California 94706, USA.
Email: michael.grabe@berkeleymadonna.com

Funding information
No funding was received for this work

Abstract

Berkeley Madonna is a software program that provides an easy and intuitive environment for graphically building and numerically solving mathematical equations. Our users range from college undergraduates with little or no mathematical experience to academic researchers and professionals building and simulating sophisticated mathematical models that represent complex systems in the biological, chemical, and engineering fields. Here we briefly describe our recent advances including a new Java-based user interface introduced in Version 9 and our transition from a 32- to 64-bit architecture with the release of Version 10. We take the reader through an example tutorial that illustrates how to construct a mathematical model in Berkeley Madonna while highlighting some of the recent changes to the software. Specifically, we construct a standard pharmacokinetic model of the antifungal medication amphotericin B taken from the literature and discuss aspects related to model building, key numerical considerations, data fitting, and graphical visualization. We end by discussing planned functionality and features intended for future releases.

INTRODUCTION

Mathematical modeling is becoming an integral component of many scientific studies affording the ability to extract useful parameters from data, make new predictions, and provide a unifying framework for understanding and interpreting experimental findings. Mathematical models are often too complex to solve analytically, and researchers frequently turn to numerical approaches to solve the underlying equations. Common commercial mathematical application packages include MATLAB, Mathematica, and Maple, and there are also a number of free packages geared to scientific computing such R and Python. Different packages provide various advantages,[1] but there is often a trade-off between ease of use and flexibility, with the most flexible environments providing the most general solutions but also having the steepest learning curves and requiring the most time to set up and solve models. Berkeley Madonna is geared toward easily constructing and solving ordinary differential equations (ODEs). The software's primary strength lies in the simplicity with which users can examine a cartoon model of a system, as shown in Figure 1a, and translate this model into equations by either typing them directly into the Equations window in standard mathematical notation or by using a simple graphical language for defining equations through

[†]Deceased.

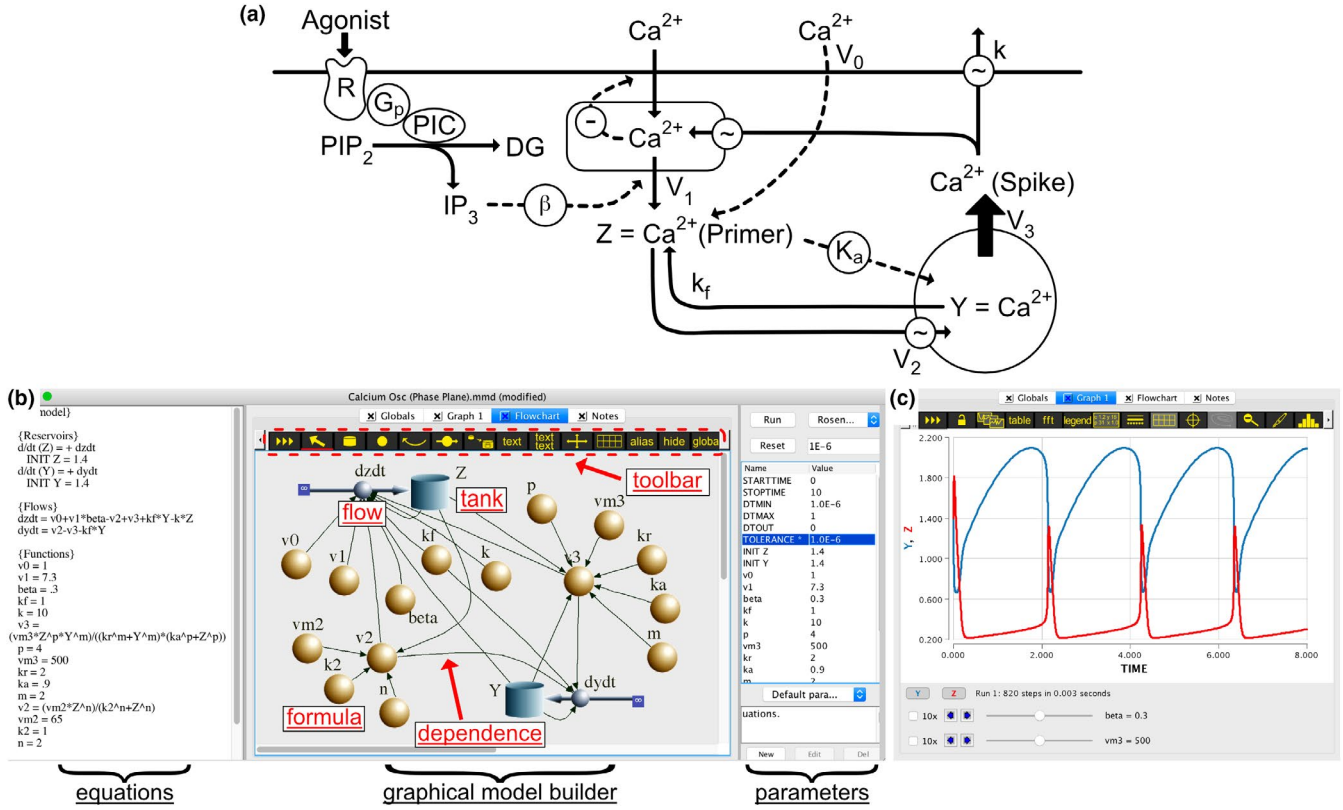

FIGURE 1 Cartoon model and Berkeley Madonna graphical user interface (GUI). (a) Cartoon model of cellular Ca^{2+} dynamics adapted from Goldbeter and colleagues.[2] (b) The GUI contains three panels (equations window, flowchart/graph window, parameters window), and models are constructed in the central window. This graphical model represents the cartoon model in panel a. Red text highlights the primary elements used to construct mathematical models: reservoirs/tanks that define time-dependent variables, flows that describe the rate of change of time-dependent variables, formula balls that allow for variable definitions, arcs that show dependence of one part of the model on another, and the toolbar from which model elements can be dragged to construct the model. The equations on the left are generated automatically from the graphical model. By first clicking Run in the Parameters window or the (>>>) symbol on the toolbar and then clicking Graph 1 in the middle panel, users can view results generated by solving the model (as shown in panel c). (c) The graphical output reveals the calcium oscillations of the inositol trisphosphate (IP_3) sensitive pool stored in intracellular vesicles (Y variable)

an intuitive user interface as illustrated in Figure 1b. Once the model equations are complete, the user selects from a list of standard numerical schemes, and the model is solved and plotted in a Graph window (Figure 1c). Within minutes, a cartoon model of a physical system, for example, a minimal model of calcium oscillations,[2] as shown in the figure, can be turned into a mathematical model in Berkeley Madonna, solved, and parameters scanned to answer quantitative questions.

In a single semester, we are able to teach undergraduates with little mathematical or programming training how to use Berkeley Madonna to solve mathematical models from the scientific literature. The straightforward equation entry coupled with a large library of predefined mathematical functions and suite of plotting tools makes it possible to visualize the graphical results and explore how they depend on model input parameters without being bogged down in technical details at different steps along the way, thus allowing the student to deepen their intuition about the system being studied. Our educational experience has convinced us that proficiency in modeling can be attained by the average student if he or she is provided with a simulation tool that is expressed in pictorial terms* and that makes minimal use of mathematical and computer jargon. To this end, Berkeley Madonna is used as a teaching tool for several textbooks at the undergraduate and graduate levels in the fields of general computation,[3] biomathematics,[4] chemical engineering,[5] and infectious disease modeling.[6] In general, all of the advantages that make Berkeley Madonna excellent for teaching students translate to the advanced user who also wants to quickly build a model to explore its properties with as little difficulty as possible.

The two major recent advances to Berkeley Madonna are the adoption of a Java-based graphical front-end that brings the development on both Windows and Mac into a single code base and the transition to a fully 64-bit architecture. The former advance presents the biggest change to our users because of differences in the interface. The Java front-end was adopted in Version 9 making it possible to again support Berkeley Madonna on the Mac operating

system (macOS) for the first time in more than a decade. As with earlier versions, each mathematical model involves the following four primary windows as shown in Figure 1: the Equations window, Flowchart, Parameters window, and Graph window. However, unlike Version 8 and earlier, a specific model is now contained within a single composite window rather than being composed of individual free-floating elements that can be positioned at will. Thus, the single fixed interface in Versions 9 and 10 makes it less likely that the user becomes confused about which graphs belong to which model.

Our transition to a 64-bit architecture has provided a number of advantages. The computer industry is moving away from supporting 32-bit applications, and Apple already made this move in 2019 with the release of macOS 10.15 Catalina. Version 10 is fully 64-bit compliant and therefore runs on macOS Catalina and the more recent macOS Monterey in addition to many legacy Mac operating systems as well as former and current Windows computers. To numerically solve mathematical models, Berkeley Madonna now emits C++ code that is then compiled into highly optimized code using the Clang compiler,[7] producing very fast run times two to five times faster than our previous versions. The move to 64-bit has also made it possible to execute models with high memory demands that are limited only by the amount of available RAM, whereas older versions would typically fail when attempting to access more than 1.4 GB of memory.

Another important change introduced in Version 9 was a switch away from saving model files in an obscure binary format and instead adopting a human readable, XML-like format. Thus, the new model format provides transparency and easy future support and extension. Berkeley Madonna models created in Versions 8 or earlier must be converted to the latest file format prior to running them in Version 9 or 10. Conversion happens automatically on Windows, but is not possible on Macs. We encourage Mac users to carry out the conversions themselves if they have access to a Windows computer or directly email models to our user support (MadonnaExec@gmail.com) and we will perform the conversion. Importantly, a Berkeley Madonna license is not required to convert models—even unregistered Windows copies have this conversion capability.

Here we provide a detailed tutorial showing how to build, simulate, and analyze a pharmacokinetic (PK) model in Berkeley Madonna. We use the visual flowchart to illustrate how to graphically construct a three-compartment model of a classic manuscript exploring the distribution kinetics of the antifungal agent amphotericin B.[8] We show how to provide parameter estimates by fitting experimental data and how to save multiple parameter sets corresponding to different scenarios. We also provide some insight and tips on solving numeric equations that are widely applicable to novice and expert modelers alike. In the Discussion, we end by summarizing the current state of Berkeley Madonna and discussing new functionality that will be added in the near future.

Example tutorial: Multicompartment PK models

Many Berkeley Madonna users employ the software to construct detailed PK and pharmacodynamic (PD) models, and tutorials exist showing not only how to build these models in Berkeley Madonna[9] but also how to harness its built-in graphics to aid in the discovery process.[10] Many of these models are referred to as compartment models because different regions of the body are treated as separate compartments that contain varying amounts the substances of interest as shown for the antifungal amphotericin B in Figure 2a adapted from Atkinson and Bennett.[8] The equations track the flow of material from compartment to compartment as the system evolves in time. By increasing the number of compartments, it is possible to create highly refined and accurate models. Although properly parameterizing models for a given problem can be difficult, the underlying principles can be gleaned from simple systems.

The Berkeley Madonna Flowchart editor allows users to pull down elements from the toolbar to define reservoirs/tanks (time dependent variables), flows (rates of change connected to reservoirs), and formulas (formulas of differing complexity), which can be linked together through the arcs (thin arrows) to define dependences (central panel of Figure 1b). These graphical models look very similar to PK/PD compartment-based models (compare panels a and b of Figure 2), making it particularly intuitive to construct such models with the graphical editor. Clicking on each element brings up a dialog box where mathematical formulas of arbitrary complexity can be typed to define the formulas, flows, and set initial conditions for the reservoirs. As the user works within this space, the underlying differential equations are constructed automatically in the left-hand panel. Advanced users can simply type equations in the Equations window if they discard the Flowchart or create a model without a Flowchart. Berkeley Madonna provides a tool for putting parameters from the Parameters window onto sliders that can be adjusted by the user to change their values, automatically rerunning the simulation and updating the results as shown in Figure 1c. Graphs are accessed by first running a model (click Run or ⏭) and then clicking on the Graph tab at the top of a Flowchart—Graph 1 in this case, which changes the central panel to the image shown in Figure 1c. Such visualizations enable quick and easy

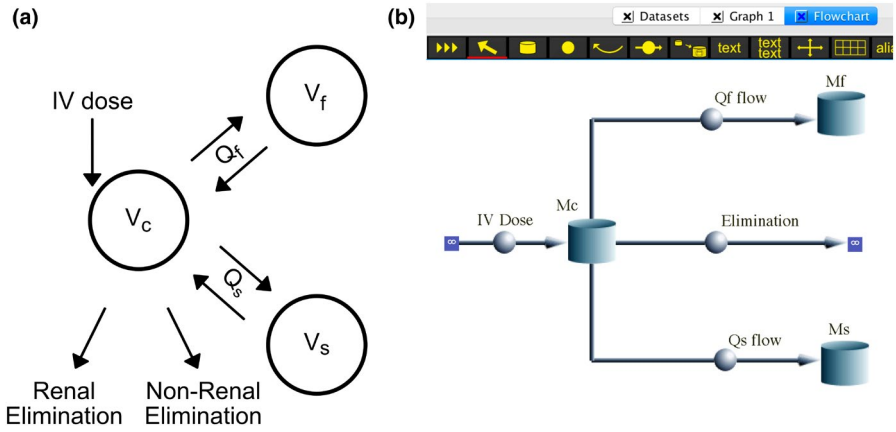

FIGURE 2 A three-compartment pharmacokinetic model. (a) Pharmacokinetic model of amphotericin B distribution and elimination modeled by a three-compartment model (figure adapted from Atkinson and Bennett[8]). Intravenous injection (IV dose) is directed into the central compartment (V_c) and then distributes to the fast (V_f) and slow (V_s) compartments before being eliminated via both renal and nonrenal pathways. The fast and slow clearance rates are Q_f and Q_s, respectively. (b) Graphical model in Berkeley Madonna representing the amphotericin B kinetics shown in panel a. The three "reservoirs" M_c, M_f, and M_s track the milligrams (mg) of the antifungal in the central, fast, and slow compartments in panel a, respectively. The "flows" into and out of each reservoir (balls with arrows through them) determine the kinetics, and they correspond to the thin arrows in panel a. Flow arrows starting or ending with an infinity sign (∞) indicate sources and sinks of the compound as they enter or leave the body; specifically, the IV Dose and Elimination flows, respectively. Note that icons representing key variables in the model are suppressed in this simplified model. See Figure 3 for the full graphical model

investigations of the model with the click of a single button. Now we demonstrate how to construct the PK model of the antifungal amphotericin B shown in Figure 2a first proposed by Atkinson and Bennett.[8]

BUILDING THE PK MODEL

1. Choose New Flowchart Document from the File menu, and an empty Flowchart window appears.
2. Position the mouse over the reservoir tool, press the mouse button, drag the reservoir to the flowchart, and release the mouse button. You now have placed a reservoir on the flowchart with the default name *R1*. The icon is colored red, showing that it is selected, and it has a question mark on it indicating that additional details must be specified for this element. For example, a numeric value or equation must be provided.
3. Change the name of the reservoir to M_c by typing the new name. It is not necessary to first click the reservoir because it is already selected. The reservoir represents the total mass in milligrams (mg) of amphotericin B in the central compartment.
4. Click the flow tool, move the mouse over the mass reservoir, press the mouse button, drag the mouse to the right a few inches, and release the mouse button. This places a flow icon on the flowchart that drains the mass reservoir into an infinite sink.
5. Make sure the reservoir is connected to the flow. If the flow looks like the image in Figure 3a, the source end is not connected to the reservoir. In this case, drag the infinite source on the left to the reservoir M_c, wait until it turns green, and then drop it. This connects the flows input to the reservoir instead of the infinite source as in Figure 3b.
6. Change the name of the flow to *Elimination* by clicking the spherical part of the flow icon (if it is not already selected) and typing the new name.
7. Click the flow tool again and move the mouse an inch to the left of the mass reservoir, press the mouse button, drag the mouse to the right until the cursor is over the reservoir, and release the mouse button. This places a flow to inject amphotericin B into the central reservoir. Rename this icon IV Dose. As before, make sure you have connected the right-hand side to the reservoir. If not, drag the infinite sink on the right onto the reservoir and drop it by releasing the mouse button. The model should now look like Figure 3c.

Constructing the elimination route

8. Place three formula icons on the flowchart near the Elimination flow icon (use the same click-and-drag technique as you did for the reservoir). Change the names of the formula icons to *Cl*, *C*, and V_c. *Cl* represents the clearance rate via both renal and nonrenal pathways (L/day) (experimental values provided in ml/min but converted here by multiplication by 24 × 60/1000), the volume V_c is in liters (L), and *C* is the concentration in mg/L or the equivalent μg/ml as reported in

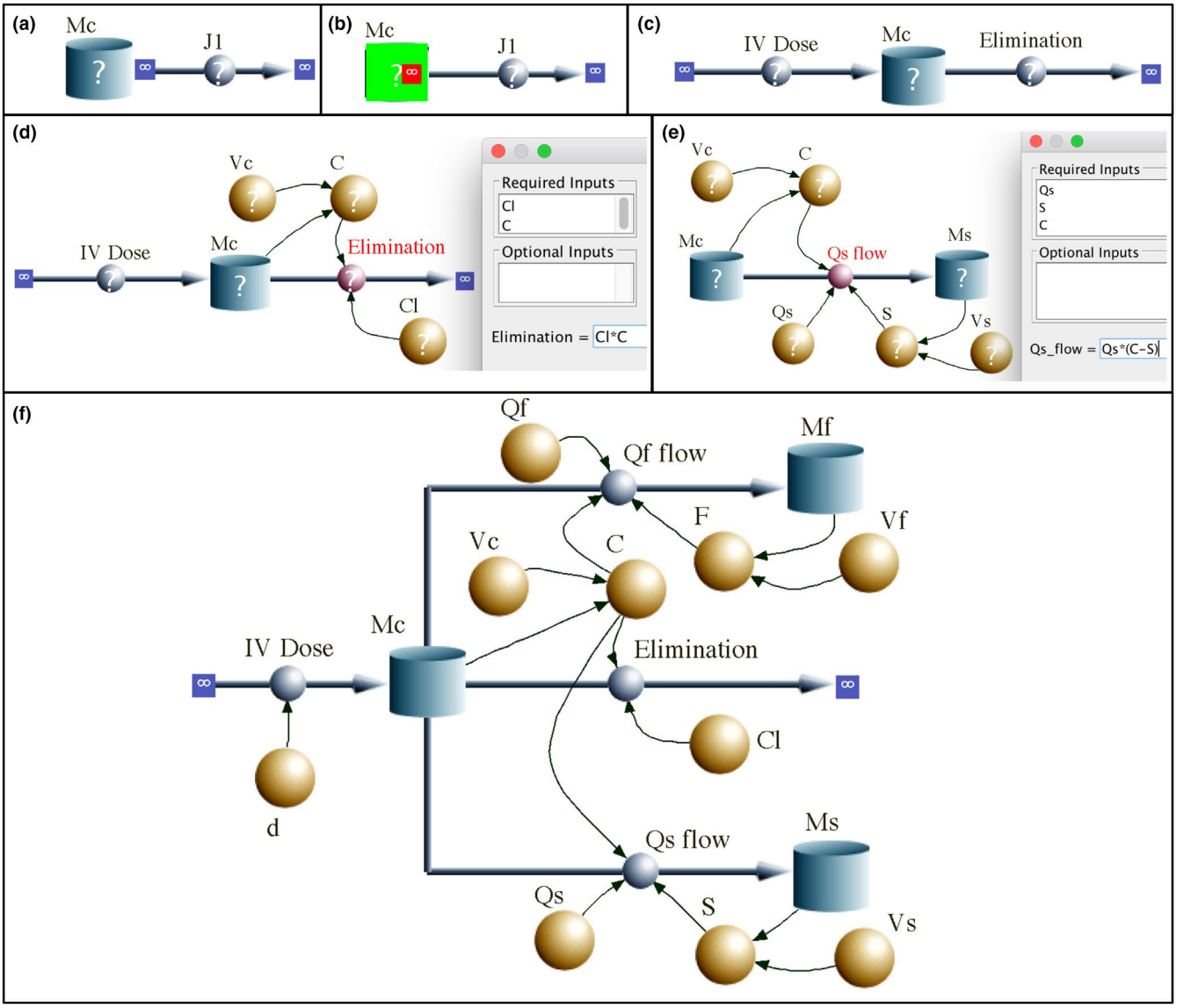

FIGURE 3 Steps in graphically building the amphotericin B pharmacokinetic model. (a) Central compartment mass reservoir (M_c) with improperly attached outflow. (b) Clicking and dragging the left ∞ sign of the flow $J1$ into the reservoir M_c lights up the reservoir in green when properly attached. Unclick the mouse to complete the attachment. (c) M_c reservoir with inward flow (IV Dose) and outward flow (Elimination). (d) Formula icons for central compartment volume (V_c), concentration (C), and elimination rate (Cl). The thin arrows connecting icons inform one element of the state or contents of the other. Elimination has arrows from C and Cl, indicating that these are both required inputs needed to define *Elimination* as shown in the equation in the dialog box. (e) Module defining the flow of drug from the central (M_c) to the slow compartment (M_s). S is the concentration in the slow compartment. The formula icons needed to define the concentrations in each compartment are present as well as the intercompartmental clearance rate (Q_s). (f) The full graphical model where additional icons for the fast intercompartmental clearance rate (Q_f) and dosage amount (d) are shown

Figure 4 here and throughout Atkinson and Bennett.[8] All model parameters can be found in Table 1.

9. Now, click the arc tool and create an arc going from the reservoir icon to the C icon (use the same technique as you did for the flow). This creates a dependency relationship between the concentration and the mass of amphotericin B. Create another arc going from the V_c icon to the C icon. Now our diagram shows that C depends on both M_c and V_c. Finally, use an arc to connect C to Elimination and use another arc to connect Cl to Elimination. These last two connections show that the rate of removal of the antifungal depends on the elimination clearance rate, which is dominated by nonrenal mechanisms as well as the drug concentration in the central compartment. Your flowchart should resemble Figure 3d, and the mathematical equation for this portion of the model is:

$$\frac{dM_c}{dt} = -\text{Elimination} = -Cl \cdot C = -Cl \cdot \frac{M_c}{V_c},$$

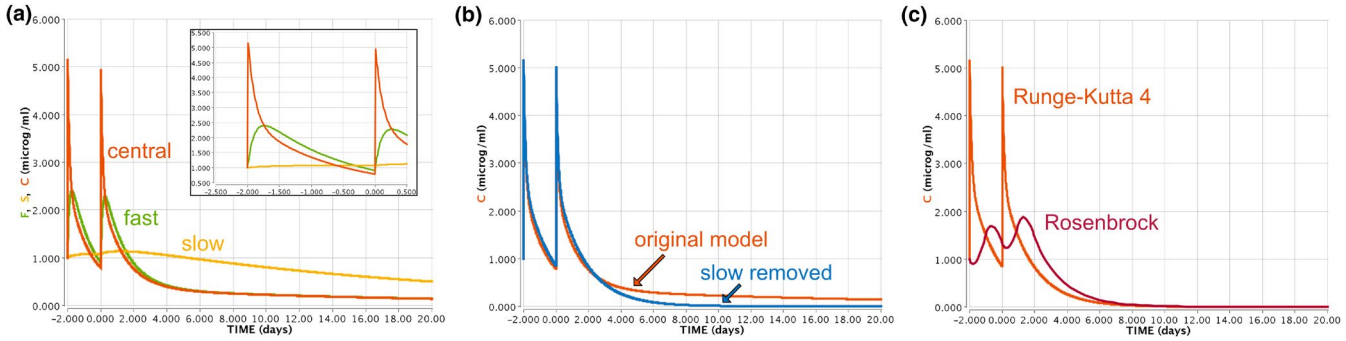

FIGURE 4 Running the model. (a) Solution to the model showing the concentrations of the three compartments in time: central (orange), fast (green), and slow (yellow). The inset zooms in on the solution during the last 2 days (−2 to 0) prior to stopping treatment to see the interplay of these compartments. (b) Concentration in the central compartment (orange) with the parameter values for Patient 217 (Table 1) compared with the equivalent two-compartment model (blue) in which Q_S was set to zero. (c) Concentration in the central compartment (orange) determined with a fixed time-step Runge-Kutta 4 method (as in panels a and b) compared with the solution with an adaptive step size method Rosenbrock (red)

TABLE 1 Model parameters

Variable	Description	Units	Patient 217	Patient 220
V_c	Central volume	L	26.3	35.4
V_f	Fast volume	L	29.2	18.9
V_s	Slow volume	L	187.4	262.6
d	Intravenous dose	mg	116[a]	116[a]
Cl	Elimination rate	L/day	34.6	52.0
Q_f	Intercompartmental clearance (fast)	L/day	114.5	152.9
Q_s	Intercompartmental clearance (slow)	L/day	12.3	14.0
$M_c\,(day=-2)$	Initial mass central compartment	mg	26.3	15.78
$M_f\,(day=-2)$	Initial mass fast compartment	mg	29.2	17.52
$M_s\,(day=-2)$	Initial mass slow compartment	mg	187.4	112.44

Note: See Atkinson and Bennett[8] for original values. All simulations use Runge-Kutta 4 (unless otherwise stated) with a solution time step of DT = 0.001 days and DTOUT = 0.01 days, which is the frequency at which solution values are graphed

[a]Values adjusted to fit the experimental data during the tutorial.

where the term on the left is the rate of change of *Mc*, Cl/V_C has the units of inverse time (L/day·1/L) = (1/day) and sets the units of time in the model to days. Double click on the Elimination flow and enter the last equation with *C* into the text box as shown in Figure 3d. Notice that the negative sign is absent in the Flowchart because the negative is accounted for by the direction of the flow arrow—it is pointing out of M_c, indicating that it removes material from M_c.

Constructing the intravenous dosing regimen

10. Next place a formula icon near IV Dose called *d* representing the instantaneous dosing of compound (mg) into the central compartment at Day −2 and Day 0. To accomplish this dosing, we use the built-in function pulse (amount, first time, repeat interval). Create an arc connecting *d* to IV Dose, and double click on IV Dose to enter the following text into the dialog window:

$$\text{IV\_Dose} = \text{pulse}\,(d, -2, \text{STOPTIME}+3) + \text{pulse}\,(d, 0, \text{STOPTIME}+1)$$

The repeat interval for each pulse is set to 1 day past the end of the simulation (STOPTIME) so that each pulse function produces only a single infusion during the run time of the model. Double click *d* and set the value to 116 mg as shown in Table 1 for both Patients 217 and 220. The manuscript does not explicitly state the value of *d* in the simulation, but 116 g is the experimental dosing. We will see that this value is slightly high here, and we will attempt to tune the value later by fitting the experimental data.

Coupling to the fast and slow compartments

11. A slow elimination-phase half-life of the drug led the authors to posit the existence of a slowly exchanging peripheral compartment in addition to a fast exchanging compartment. The volume was estimated from the experimental data to be roughly 80% of the total volume of distribution for the slow store (V_s), and the intercompartmental clearance of amphotericin B between the central and slow compartments was found to be ≈9 ml/min or ≈13 L/day (see Table 1 for individual patient values). Click and drag the reservoir tool to create a new reservoir and name it M_s for slow compartment. Now use the flow tool to connect M_c to M_s. Click once on the flow tool and release. Now click and hold on the M_c reservoir, drag to M_s, and once M_s is highlighted green, release the mouse. Now you should see a flow connecting the two reservoirs. Create three formulas icons and label them V_s, S, and Q_s to represent the volume, concentration, and intercompartmental clearance rate for the slow compartment. Use the arc tool to create dependences between the icons as shown in Figure 3e (elimination flow not shown for clarity). Double click on all of the icons and fill in values from Table 1 for Patient 217, and use the following equation for the concentration formula $S = M_s/V_s$. Now double click on the flow tool, and enter the rate as shown in panel e:

$$Q_s \text{ Flow} = Q_s\,(C - S)$$

This equation indicates a passive flow of drug between the central and slow compartment proportional to Q_s. Moreover, the flow stops when the concentration of the drug in both compartments is equal.

12. Repeat step 11 for the fast compartment using the naming scheme shown in the final full model Figure 3f, where Q_f defines the fast intercompartmental clearance rate. The model can be downloaded from the Supplementary Material (amphoB.mmd).

The full set of equations can be written as:

$$\frac{dM_c}{dt} = \text{IV Dose} + Q_f\left(\frac{M_c}{V_c} - \frac{M_f}{V_f}\right) + Q_s\left(\frac{M_c}{V_c} - \frac{M_s}{V_s}\right) - Cl \cdot \frac{M_c}{V_c}$$

$$\frac{dM_f}{dt} = -Q_f\left(\frac{M_c}{V_c} - \frac{M_f}{V_f}\right)$$

$$\frac{dM_s}{dt} = -Q_s\left(\frac{M_c}{V_c} - \frac{M_s}{V_s}\right).$$

As with all ODEs, initial conditions must be provided for each time-dependent variable (reservoir). When modeling patients at the end of treatment, Atkinson and Bennett assumed patients were at steady state with respect to amphotericin B prior to administering the final two doses at Day −2 and Day 0, and therefore, initial masses in Table 1 were determined from figure 2 of Atkinson and Bennett,[8] indicating concentrations of 1.0 and 0.6 mg/ml for Patients 217 and 220, respectively, and multiplying by the volume of each compartment.

Running the model

Now that the model is complete with parameter values for Patient 217 (there should be no question marks—?—on any icons), we are nearly ready to run the model. First, change the STARTTIME and STOPTIME in the far right Parameters window to −2 and 20, respectively, DT (the numeric time step) to 0.001, and DTOUT (how frequently solution points are printed to the graph window) to 0.01. Then hit "Run" in the Parameters window or in the Flowchart window, and a new graph should appear plotting M_c, M_f, and M_s. However, we want to plot the concentrations, so double click anywhere on the main Graph window, and a dialog box will appear that lets you "<< Remove" the mass values and ">> Add" F, S, and C. Now press "OK" and then Run again; the concentrations should now be plotted. Next, we want all variables plotted on the left y-axis, so if any appear on the right axis, hold down the "Shift" button and click the variable button of interest at the lower left corner of the graph. This action toggles variables between the right and left y-axes. Repeat until all variables are plotted on the left axis as shown in Figure 4a.

Initially, all concentrations are 1 μg/ml but the central compartment (orange) quickly spikes and then falls off exponentially during the next 2 days as drug moves primarily to the fast compartment while also being eliminated, eventually dipping below the starting 1 μg/ml value. The fast compartment (green) also quickly rises, but smoothly peaks near 2.5 μg/ml 6 h later (−1.75 days) before starting to decrease (see Figure 4a [inset]). The concentration of the slow compartment (yellow) rises slowly during the next 2 days as some of the drug from the central compartment enters, but it does not fall below the starting value until Day 5, providing the source of the sustained concentration of the drug in the central compartment. You can see the pronounced elimination of the drug in the absence of the slow compartment by

selecting only central concentration C, hitting the overlay tool, setting Q_s to zero in the Parameters window, and hitting Run (Figure 4b). This modification essentially creates a two-compartment model, and the ability to describe the retention of drug in the body beyond 2.5 days is lost. Change Q_s back to its original value for Patient 217 by selecting the parameter in the right window and clicking "Reset."

We wanted to briefly discuss a technical point about mathematical models of which even the novice user should be informed. In the Parameters window you can see that we are using the default Runge-Kutta 4 method to solve the equations. This is a fixed time-step method that determines the solution to the equations at a regular time interval DT, which can be adjusted. The smaller the interval DT the more accurate the solution, but below a certain value, solutions are well converged to the correct value. For this model, that value is DT = 0.001 days. Experiment with the accuracy of the solution by increasing DT to 0.1 with overlay on; the height of the initial peak is significantly reduced. Then set DT to 0.0001, and you will see that the solution is virtually unchanged from the starting value of 0.001. Now deselect the overlay tool before continuing.

The most common way to exceed memory limits in Berkeley Madonna is by employing one of these fixed time-step solvers (Euler or the Runge-Kutta methods) with a very small time step and/or large stop time. Solutions to these models produce copious amounts of data that can exceed memory limits, and they take longer to run. Although this problem is much better in Version 10 than previous versions due to increased memory accessibility, users should employ the DTOUT feature to reduce the amount of saved data. When DTOUT is larger than DT, intermediate points used to solve the model with high fidelity (i.e., every DT interval) are discarded, and only values computed at the less frequent DTOUT interval are retained. In some cases, this approach still does not alleviate memory issues, and "stiff methods" such as Auto-stepsize or Rosenbrock are needed. These stiff methods automatically increase the step size in solution regions that are slowly varying to reduce the computational burden and memory demand and then decrease the step size in quickly changing regions where numeric errors accumulate. Rosenbrock and Auto-stepsize are generally excellent algorithms to employ, but not for models that pulse in discrete changes to reservoirs such as this one. See this for yourself by changing the solution method to Auto-stepsize or Rosenbrock and hitting run. The sharp change in concentration C is smoothed out, and it looks nothing like the desired result of an instantaneous bolus of drug (Figure 4c). Change the solution method back to Runge-Kutta 4.

Working with experimental data

A key feature of Berkeley Madonna is its ability to easily load experimental data and determine key experimental parameters by fitting the model to the data. We digitized the amphotericin B time-series data provided by Atkinson and Bennett for Patients 217 (*P217_data.txt*) and 220 (*P220_data.txt*) in figure 2 of their article, and these digitized data sets are provided as Supplemental Files. To load the data into your Berkeley Madonna model, open the Datasets panel under Model from the menu bar. Click "Import" and navigate to where the data file is stored on your computer. Import the data as a 1D vector with the names *P217* and *P220*. The numeric values will automatically be shown in the Dataset window (Figure 5a), and the points will be plotted in the Graph window. It is often useful to plot the y-axes with a log base 10 scale to accentuate features of the curve at small values. To change the axes, double click on the graph anywhere near the x or y axes. For the "Left *Y* Axis," check the box next to "log." You can also set the maximum and minimum values if desired or let the software determine this automatically.

Plotting only the *P217* data and *C* on the graph allows you to determine how well the model (solid curve) matches the experimental data (Figure 5b). The general shapes of both curves are nearly identical; however, notice that the model is slightly above the experimental points likely arising from small systematic errors that occurred as we extracted the data from the published images. To illustrate Berkeley Madonna's data-fitting functionality, let us attempt to better fit the data with the "Curve Fit" procedure found under the Parameters menu item. Selecting this feature opens a dialog box as shown in Figure 5c. Since the initial peak of *C* during each injection is slightly high, let us pick the dose amount *d* as the search parameter. Select it under the "Available" list and add hit "Add >>." The four values on the right help guide the search. Because the dose *d* must be a positive definite number, set Minimum to 0, and because 116 mg already appears too large, it is probably safe to use it as an upper bound, but let us choose 200 mg instead just to be safe. Now provide two initial guesses between the upper and lower limits to get the search algorithm started. Berkeley Madonna uses a Nelder-Mead simplex method to perform its curve fits,[11] and the algorithm attempts to identify the parameter values that minimize the root mean square difference between the data points and the model solution over the specified time range (STARTTIME to STOPTIME). Make sure your time range encompasses all of the data points you want to fit. Pick the "Fit Variable" *C* and the correct "Dataset" *#P217*. The Tolerance value in the bottom left of the dialog box is related to the exit criteria for the search: the smaller this value, the longer the search will proceed,

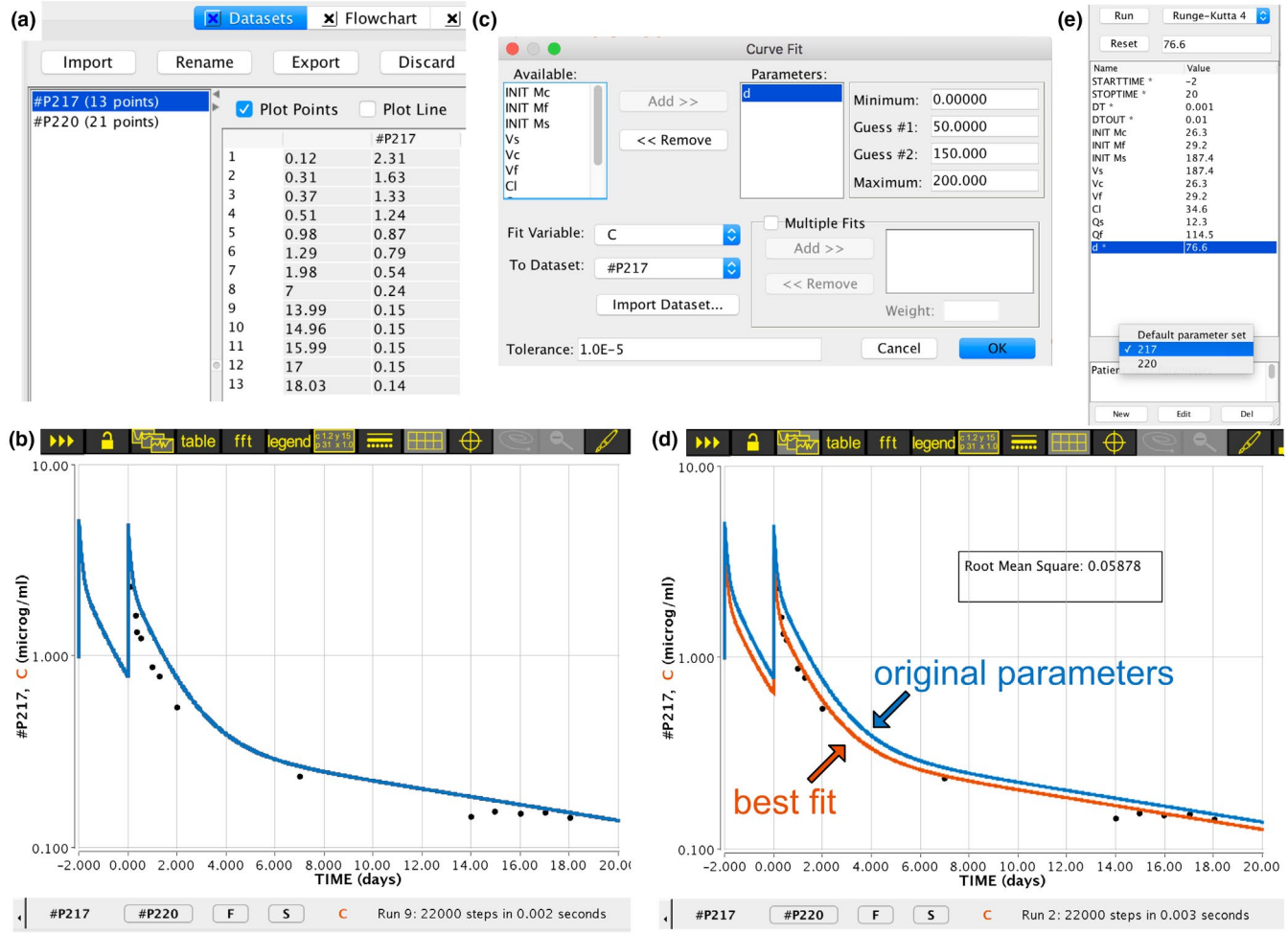

FIGURE 5 Working with experimental data. (a) Datasets window showing imported data set for Patient 217. (b) Pharmacokinetic model solution (blue curve) for Patient 217 data (black dots) using parameters in Table 1. (c) Curve Fit dialog box showing how parameters are selected for the optimization of one model variable (C) against a data set (#P217). (d) Overlay of original model solution (blue curve) with the improved fit (orange curve) to the experimental data (black dots) obtained for $d = 76.6$ mg. The root mean square error for this fit is provided in the text box. (e) Parameters window with an updated value for d has been saved as a new data set 217

potentially leading to better solutions. Here we use a value of 1E−5 (10^{-5}), and typical values range from 0.01 to 1E−6. Now hit "OK" to perform the search.

The software solves the model many times with different values of d to determine the best value that minimizes the root mean square difference between the model and the data, finally arriving at $d = 76.6$ mg (orange curve in Figure 5d). This reduced value of the intravenous dose makes the initial two concentration spikes at time = −2 days and 0 days slightly smaller and provides an excellent fit to the experimental data. Before moving on to examine the P220 data set, save the current model parameters for Patient 217 so they can be quickly recalled later. Go to the bottom of the Parameters window on the right and hit the "New" button (Figure 5e). This will bring up a dialog box with the current parameter settings, and it will allow you to give this set a Name (call it 217) and a descriptive title, say "Patient #217 Parameters." Now enter all of the information for Patient P220 from Table 1 into the Parameters window and compare the model output to the experimental data again seeing that the model values are slightly higher than the experimental data. Go through the same fitting procedure carefully choosing to select "To Dataset: #P220" in the Curve Fit dialog box and determine the optimal dosage. You should get $d = 62.4$ for Patient 220. Both data sets can be downloaded from the Supplementary Material (P217_data.txt and P220_data.txt).

Determining optimal dosing schemes

One of the most powerful uses of a PK model is its ability to predict the concentration of substances in each body compartment over time to ensure minimal inhibitory concentrations (MICs) are achieved while at the same time toxic

levels are not reached. The MIC of amphotericin B is in the range of 0.2–0.5 μg/ml.[8] Imagine Patient 220 (Table 1) enters the hospital with an aggressive aspergillosis infection. Amphotericin B, along with other medical interventions, would be the first-line treatment, and we want to identify an effective intravenous infusion regimen. Intravenous infusion will be given 4 h per day during the first 4 days, but how much should be given? We will use the model to explore two regimens: low dose (1, 5, 10, 15) mg per day and high dose (25, 25, 30, 30) mg per day.

Starting from the full model (Figure 3f), let us replace the IV Dose with an infusion scheme rather than an instantaneous pulse. To do this, we will use the built in function squarepulse (start time, duration), which is 0 before the specified start time and then jumps to 1 at the start time, lasting for the specified duration before returning to 0:

$$\begin{aligned}\text{IV\_Dose} = {} & I1 * x * \text{squarepulse}\,(0, 4/24) + \ldots \\ & I2 * x * \text{squarepulse}\,(1, 4/24) + \ldots \\ & I3 * x * \text{squarepulse}\,(2, 4/24) + \ldots \\ & I4 * x * \text{squarepulse}\,(3, 4/24),\end{aligned}$$

where $I1$, $I2$, $I3$, and $I4$ are the total amount of drug delivered (mg) during each infusion, and x is the infusion rate required to inject 1 mg in a 4-h infusion period ($x = 1/[4/24\text{ days}]/\text{mg} = 6$ mg/day). Adding these four square pulse functions together produces a drug flow into the central compartment that is constant for 4 h followed by no infusion for 20 h over 4 days (see modification in Figure 6a). The constants $I1$–$I4$ allow the amount of drug delivered to vary with each infusion. To ensure that IV Dose is behaving as expected, we can use a common trick in Berkeley Madonna: create a new reservoir (Total Drug) to track the total drug administered. To do this, create a flow into the reservoir (*J1*) and copy and paste the equation for IV Dose into *J*1. Because there is no out flow, Total Drug only changes according to the IV Dose and integrates the total change (blue and green curves in Figure 6b for low and high doses, respectively). The right axis indicates that the correct amounts of drug are delivered in each infusion, increasing linearly at a constant rate and then plateauing at the desired amount before the next infusion.

During the first 4 days, the concentration C in the central compartment under the low-dose regimen (black curve) spikes during each 4-h infusion and then slowly drops until the next infusion (Figure 6b). At the end of the 10-mg infusion on Day 3, the patient achieves a serum concentration level of 0.2 μg/ml in the central compartment consistent with the MIC (shaded region). However, once the infusion stops, the concentration quickly falls out of the desired range only to reenter the range the next day during the 15-mg infusion. Again, this fourth infusion keeps the patient in the MIC range for only half of the day. On the other hand, the high-dose regimen (orange curve) enters the MIC range during the first infusion only briefly dipping below at the end of the first day, but then remains in, or just above, the range the entire time. Obviously, this scheme will provide much more therapeutic benefit, much faster, while presumably minimizing toxicity, which can be serious.[12] This modified version of the model can be downloaded from the Supplementary Material (amphoB dosing.mmd).

DISCUSSION AND FUTURE PLANS

We are now in an excellent position to extend Berkeley Madonna in several new directions. Our most immediate

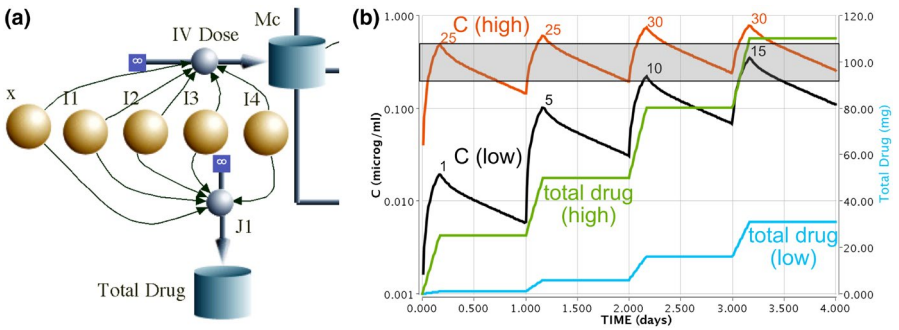

FIGURE 6 Optimal dosing regimen. (a) IV Dose with an infusion model with total drug delivery $I1$, $I2$, $I3$, and $I4$ over four 4-h infusions. x is a constant base infusion rate for a 1 mg infusion. The flow $J1$ is identical to IV Dose, and the reservoir Total Drug accumulates the total delivery during the time course. (b) The drug concentration in the central compartment for a low-dose (black curve) and high-dose (orange curve) regimen. The shaded region indicates typical range of minimal inhibitory concentrations 0.2 to 0.5 μg/ml. The total amounts of injected drug are plotted on the right axis for the low-dose (blue curve) and high-dose (green curve) regimens, providing a validation check. Injected values during each infusion (provided in milligrams) are next to each concentration peak. STARTTIME and STOPTIME are 0 and 4 days, respectively. Initial values for M_c, M_f, and M_s are all set to 0 at time 0, and all other parameters are those for Patient 220 in Table 1

goal is to release Berkeley Madonna 10 on Linux operating systems. As with macOS, it has been many years since we have supported Berkeley Madonna on Linux, but the similarity between the macOS and Linux operating systems has facilitated porting to Linux. We have a working beta version running in house, and we expect to have a fully tested version ready for release in early 2022. Another benefit of using a modern external compiler is that we can easily add support for 64-bit CPU architectures other than x86-64, such as Apple's M1 ARM processor, without rewriting the code generator. Our current release runs well on Apple ARM processors in emulated mode, but we also have a fully native release version that runs significantly faster than the x86-64 release.

We intend to incorporate several new features related to the generation of graphics and how images are saved. As previously discussed, the graphics windows are handled differently in Versions 9 and 10 compared with earlier versions. All windows are free floating in Version 8, allowing the user to stack graphics windows next to each other to compare how results change when the model is modified and recompiled. In Versions 9 and 10, the windows for a given model are all in a fixed interface (Figure 1). We appreciate the need to see results side by side, and to address this need, we are implementing multipanel graphics plots so that results can be tiled in N rows by M columns in a single window. We will make it possible to populate each subplot with data computed from different runs performed after a model has been recompiled. We will also provide increased control over the line styles and line colors so that particular curves can be modified after they are drawn. In addition, throughout the entire interface, we will provide greater control over the font, font style, and font size. Finally, because figures currently can only be saved as raster graphics, specifically JPEG format, they cannot be easily manipulated in a graphics program after saving, and they suffer from pixilation when magnified. We will enable graphics to be saved as PDFs, providing improved resolution and ease of editability of the embedded text and vector graphics in Adobe Illustrator or other drawing programs once saved to file. The move to a vector graphics format will greatly aid in the production of high-quality graphics for reports and scientific publications.

It is becoming more routine for researchers to provide statistical analyses when reporting results. In addition, PK and PD models that incorporate individual-level variations in specific parameters (implemented by drawing values from a probability distribution for each run) must explicitly account for the confidence intervals and outcome distributions that result from running a model many times. Such models can already be created and solved in Berkeley Madonna by running in Batch Run mode to repeat runs many times, and the new histogram plotting tool can be used to visualize the simulation output. However, we are adding several new probability distributions to the language (gamma, beta, log normal, exponential, logistic, and triangular) to meet our users' needs, and we intend to enable running on multiple CPUs to complete independent batch runs faster. We also plan to make it easier for our users to compute additional statistical measures, such as areas under the curve, standard deviations, and 95% confidence intervals, by adding built-in functions for such quantities. Finally, we are exploring methods for reporting error estimates on optimized parameters that result from fitting models to experimental data with the curve fit function.

The major advances we made to Berkeley Madonna Version 10 outlined here will greatly facilitate our ability to incorporate all of these proposed new additions. As we continue to improve the software, we look forward to obtaining feedback from our users so that we can provide other features not discussed here.

ACKNOWLEDGMENTS

The authors thank Mike Gittelsohn and Goran Mitrovic for their aid in developing this software. They also thank Deanna Kroetz (University of California, San Francisco) for help with constructing the model used in the tutorial.

CONFLICT OF INTEREST

All authors are employees of Berkeley Madonna, Incorporated.

ENDNOTE

* Users can also construct models by writing out the equations directly in the Equations window without using the graphical Flowchart.

REFERENCES

1. Lin Z, Jaberi-Douraki M, He C, et al. Performance assessment and translation of physiologically based pharmacokinetic models from acslX to Berkeley Madonna, MATLAB, and R Language: oxytetracycline and gold nanoparticles as case examples. *Toxicol Sci*. 2017;158(1):23-35.
2. Goldbeter A, Dupont G, Berridge MJ. Minimal model for signal-induced Ca2+ oscillations and for their frequency encoding through protein phosphorylation. *Proc Natl Acad Sci USA*. 1990;87(4):1461-1465.
3. Shiflet AB, Shiflet GW. *Introduction to Computational Science: Modeling and Simulation for the Sciences*. Princeton University Press; 2014.
4. Robeva R, Kirkwood JR. *Laboratory Manual of Biomathematics*. Academic Press; 2007.
5. Ingham J, Dunn IJ, Heinzle E, Prenosil JE, Snape JB. *Chemical Engineering Dynamics: An Introduction to Modelling and Computer Simulation*. 3rd ed. Wiley-VCH Verlag GmbH & Co. KGaA; 2007.
6. Vynnycky E, White R. *An Introduction to Infectious Disease Modelling*. Oxford University Press; 2010.

7. Clang: a C language family frontend for LLVM. Accessed February 19, 2021. https://clang.llvm.org/
8. Atkinson AJ Jr, Bennett JE. Amphotericin B pharmacokinetics in humans. *Antimicrob Agents Chemother*. 1978;13(2):271-276.
9. Dua P, Hawkins E, van der Graaf PH. A tutorial on Target-Mediated Drug Disposition (TMDD) models. *CPT Pharmacometrics Syst Pharmacol*. 2015;4(6):324-337.
10. Krause A, Lowe PJ. Visualization and communication of pharmacometric models with Berkeley Madonna. *CPT Pharmacometrics Syst Pharmacol*. 2014;3:e116.
11. Nelder JA, Mead R. A simplex-method for function minimization. *Comput J*. 1965;7(4):308-313.
12. Laniado-Laborín R, Cabrales-Vargas MN. Amphotericin B: side effects and toxicity. *Rev Iberoam Micol*. 2009;26(4):223-227.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: Marcoline FV, Furth J, Nayak S, Grabe M, Macey RI. Berkeley Madonna Version 10–A simulation package for solving mathematical models. *CPT Pharmacometrics Syst Pharmacol*. 2022;11:290-301. doi:10.1002/psp4.12757