The Exponent Rat Dimethoate PBPK-PD Model:

Introduction and *Mathematica* primer

For use with *Mathematica*® version 11.0 or higher.

Kenneth T. Bogen, DrPH DABT
Managing Scientist
475 14th Street, Suite 400
Exponent, Inc.
Oakland, CA 94612

kbogen@exponent.com

Your *Mathematica* version number, base directory, and today's date in {year, month, day, hour, min, sec} format:

```
{$Version, $UserBaseDirectory, Date[]}
```

```
{11.1.1 for Mac OS X x86 (64-bit) (April 27, 2017),
 /Users/kbogen/Library/Mathematica, {2017, 7, 20, 2, 6, 30.754836}}
```

Contents

Introduction to the Exponent Dimethoate Model (ExDM)

General Introduction

This document provides an introduction to the Exponent rat PBPK model developed for for dimethoate and its metabolites, hereafter for convenience referred to as the Exponent Dimethoate Model (ExDM), which is written in version 11 of the *Mathematica*® programming language (Wolfram Research 2017), in which this *Mathematica* "notebook" document is written (and which was saved to pdf file format if you are now viewing a pdf file rather than a *Mathematica* notebook).

A detailed description of the ExDM model is provided in a corresponding report by Bogen and Reiss (2017) prepared for submission to the U.S. Environmental Protection Agency, with which the reader is assumed to be familiar. Basic ExDM capabilities are explained and illustrated, by evaluating *Mathematica* functions that execute, evaluate, and analyze symbolically captured output from the ExDM *Mathematica* code. To faciliate illustrating ExDM capabilities and applications, such symbolic ExDM output is also plotted using a plotting function called PlotData from a large (copyrighted) *Mathematica* "package" called *RiskQ* that was developed by Dr. Bogen. To facilitate plotting of ExDM results, the PlotData function has been inserted into the executable ExDM code, so that after that code is loaded, the PlotData function is available for applications by the user.

The introduction to ExDM is followed by a general primer on the basic *Mathematica* syntax, functionality, and applications. Although the ExDM introduction apperas first so that readers can immediately see how to get the ExDM model to work and how to plot its output, the more general Primer also should be read first before intensive applications of ExDM are undertaken, to better understand the syntax and capabilities of the *Mathematica* software platform in which ExDM runs.

Readers are directed to Wolfram Research (2017) for more extensive information, and/or to the Wolfram Documentation selection under the "Help" tab that appears at the top of the screen after opening any *Mathematica* notebook. The latter Documentation supplies complete and very extensive documentation of all *Mathematica* syntax and functionality supporting programming, numerical evaluation, symbolic mathematics, plotting, and a vast array of other features of the *Mathematica* software platform. The primer offered below provides a quick introduction allowing access to key capabilities offered by this platform.

References

Bogen K. Reisss R. 2017. Physiologically-based Pharmacokinetic/Pharmacodynamic (PBPK/PD) Models for Dimethoate. Project No. 1200734.002. July 6, 2017. Exponent, Inc., Alexandria, VA.

Wolfram Research. 2017. Wolfram Language and System Documentation Center. Wolfram Research, Inc., Champaign, IL (www.wolfram.com), http://reference.wolfram.com/language/

Bogen K. Reisss R. 2017. Physiologically Based Pharmacokinetic/ Pharmacodynamic (PBPK/PD) Models for Dimethoate. Project No. 1200734.002. July 6, 2017. Exponent, Inc., Alexandria, VA.

Introduction to the Exponent Dimethoate Model (ExDM)

Overview & DimethoateModel usage statement

General syntax and capabilities of *Mathematica* are discussed in the Primer that follows this introduction to ExDM. However, first a note on *Mathematica* notebooks and cells. This document is a *Mathematica* notebook, which consists of cells of possibly different types stacked one above the other, typically arranged automatically in groups of cells that are stacked hierarchically, with each group of cells stacked on over another within (i.e., under) a section, subsection, or subsubsection heading. This cell is a Text cell, located under a Section cell. The vertical span of each cell is shown by the right square bracket that appears along its right-hand side. If you double-click the right square bracket that surrounds this current cell and all those contained in this Section that starts with the Section cell above, then this cell and all of those in this section will be hidden. Double-clicking again on that outer bracket will make all these contained cells reappear (try it). Within a cell, pressing Shift + Command + D (on a Mac computer) will divide the current cell into two cells of the same type. After highlighting

If you click below a cell, between it and the next cell down, you will create a new Input cell, like the one below. If you click in that cell and press (at the same time) Shift + Enter, this will evaluate that Input cell and below it will appear a corresponding Output cell (try it on the cell below).

```
{1 + 1, a + b}
```

To load the ExponentDimethoateModel.m package allowing you to run the ExDM model, after opening this (or any other) *Mathematica* notebook, evaluate the following expression in an Input cell by clicking anywhere in that cell and by then pressing Shift + Enter. A semicolon at the end of an expression (like that which appears below) will block the appearance of an Output cell after evaluating an Input cell, but the evaluation of the expression in the Input cell will nevertheless still take place.

```
<< ExponentDimethoateModel`;
```

Now the ExponentDimethoateModel.m package has been loaded, which defines a DimethoateModel function that can be used to run the ExDM model, and also defines numerous ExDM model input parameter tables and parameter symbols by assigning default values to them. To lear about the syntax and capability of the DimethoateModel function, evaluate the following expression, which returns a “usage” statement that summarizes information about this function.

```
? DimethoateModel
```

DimethoateModel[bwkg, ingest, dermal, Cair, tmax, options:] is a Mathematica function that runs the Exponent rat dimethoate PBPK–PD model (ExDM). After being evaluated, an expression of the form sol = DimethoateModel[...] returs a list R of 113 rules of the form {v1[t] → f1[t], v2[t] → f2[t], ..., v113[t] → f113[t]} which associate each kth symbolic compartment variable (vk[t]) of the ExDM model expressed as a function of time t (shown on the left side of each rule) with a corresponding numeric Interpolating function of time t (fk[t], shown on the right side of each rule) that can be used to evaluate the numerical value of the 113 ExDM model variables at any time(s) t (in hours) over the period 0 ≤ t ≤ tmax. The value of vk[t] at time t=T is obtained by evaluating the expression, vk[t] /. sol /. t→T, where each /. denotes 'conditional on the following rule'. To run, DimethoateModel requires five arguments (bwkg [a rodent body weight in kg], ingest [an assumed ingested mass in mg], dermal [an assumed dermally applied load in mg/cm2––not currently implemented so set this to 0], Cair [an air concentration in mg/m3––not currently implemented so set this to 0], and tmax [a duration of integration in hours]), which may be followed by one or more Rule–type options (explained below) each of the form OptionName → optionValue.

Type Options[DimethoateModel] to see the default set of option values. All default (i.e., starting) ExDM parameters are listed in the following pre–defined list of tables: {ParTable, MetTable, PDTableCE, PDTableDegr, PDTableInhib, PDTableReact, PDTableAge, RateTable, ParTableScaled, MetTableScaled, RateTableScaled}, each of which may be printed in tabular form using the TBL function(e.g., by evaluating TBL[RateTable], or TBL/@{PDTableReact, PDTableDegr}, etc.). Tables that end in Scaled show values adjusted for body weight. Note: CE = cholinesterase, ACE = acetyl CE, BCE = butyryl CE, CaE = carboxylesterase. The latter tables refer to 10 modeled tissues (i=1,..,10), 3 esterases (ACE, BCE, CaE) (j=1,2,3), and 5 meatbolites (dimethoate (DIM), omethoate (OME), DCA, DMTP, DMP, and DTDT+DTP (m=1,...5).

Use Adjust → {{ParType1, ParList1, fList1}, {ParType2, ParList2, fList2}, ...} to adjust the default values of one or more indicated specific ExDM parameters specified in the list ParList each of type ParType by corresponding indicated multiplicative scaling factors listed in fList, where ParType is a member of {Rate, Par, Met, PD, CE, Degr, Inhib, React, PPset}, ParList is a specific parameter contained in a ParameterTable identified as described below, and fList is a corresponding list of numeric scaling factors fList = {f1, f2, f3, ..., fk} where k is the length of both ParList and fList. All paremters are initially defined using a rat weight of 0.25 kg, but become redefined using the weight bwkg specified as the 1st argument of the DimethoateModel function, by default using assumptions consistent with experimental rat bood and plasma volumes relative to body weight (Lee and Blaufox 1985). Units of tabulated paramters are defined below:

PARAMETER	TABLE	UNIT	
K–type	All (except Kinhib)	1/h	
F–type	RateTable	unitless fraction	
Kturn (list)	RateTable	esterase turnover (μmol/h)	
M–type	RateTable	mass in μmol	
QQ–type	ParTable	flow in L/h	
PP–type	ParTable	tissue: blood partition coefficient	unitless
Vmax–type	MetTable	max rxn velocity in μmol/h	
Kinhib–type	PDTableInhib	bimolecular inhibition rate in L/μmol/h	

The following ExDM model parameters defined symbolically after the DimethoateModel function is evaluated:

VV[i] = tissue i volume in L
BW = total body weight in kg
Vplasma = plasma volume in L
Vblood = blood volume in L
hematocrit = fraction of non–plasma blood volume
QQ[i] = tissue i venous blood flow in L/h
Qc = cardiac blood flow rate in L/h
Qresp = ventilation rate in L/h
Qalv = alveolar ventilation rate in L/h
All rates listed in RateTableScaled
PP[i,m] = tissue i:blood partit. coeff. (unitless)
MM[i,m] = mass in μmol in tissue i and metab m (see MetTable)
Vmax[i][m1,m2] = Vmax in tissue i for metabolite m1–to–m2 metabolism
Km[i][m1,m2] = Michaelis constant for metabolite m1–to–m2 metabolism
Kturn[i,j] = rxns per {AChE,BChE,CaE} enzyme per h
Kdegr[i,j] = tissue i esterase j degredation rate (1/h)
Kage[i,j] = tissue i esterase j ageing rate (1/h)
Kreact[i,j] = tissue i esterase j reactivation rate (1/h)
Kinhib[i,j] = tissue i esterase j inhibition rate (1/μmol/h)

DimethoateModel returns the following set of ExDM model output variables, each a function of time t. In addition to the variables listed below are Stomach[t], Intest[t], StomachUnAbs[t], IntestUnAbs[t], AlbX[t] (hypothesized albumin–bound OME), FeCar[t] (total μmol of all metabs in feces and carcass), SUM[t] (the sum of all mass–conserving compartments, and Blood[m][t] (μmol of metabolite m in blood).

MM[i,m][t] = μmol of metabolite m in tissue i
U[m][t] = μmol of metabolite m in urine
CE[i,j][t] = μmol of active esterase j in tissue i
CE0[i,j] = CE[i,j][0] = baseline CE[i,j] level at t=0
CEX[i,j][t] = μmol of inactivated esterase j in tissue i
PCE[i,j][t] = 100%(CE[i,j][t]/CE0[i,j])

One or more of the following options may be used:

Chemical → Omethoate can be used to replace the default assumption that the orally administered chemical is Dimethoate.
Adjust → { {Rate, {'Kintes2liv', 'Kurx[5]'}, {1.2, 0.95}} [where note that single quotes must here be replaced by double quotes so as to signify Rate–type parameter names as Strings rather than Symbols], {Met, {{V12, 2}, {Km24, 8}}, {1.488, 0.5}}, {Inhib, {AChE, #}&/@{2, 5, 8}, 1.205{1, 1, 1, 1, 1}} } (as an illustration) can be used to scale the Kintes2liv rate and 5th urinary excretion rate (denoted Kurx[5]) by 1.2 and 0.95 respectively; the DIM–to–OME Vmax parameter for Liver (i.e., Vmax[2][1,2]) by 1.48–fold; the OME–to–DCA Km parameter in Lung (i.e., Km[8][2,4]) by 0.5–fold; and AChE–inhibition rates in Liver, Skin, and Lung (i.e., {Kinhib[#, 1]&/@{2,5,8}}) each by 1.205–fold.
Scenario →Oral or →OralDaily must be used to specify a single or daily oral exposure scenario.
NOralPulses → n to reset (from n=1) the number of daily oral–pulse doses applied
TOralPulse → t to reset (from t = 1/60) the duration of each oral pulse.
Species → Rat to use generic (rather than empirical Lee H& Blaufox 1985) rat assumptions relating blood and plasma volumes to BW.

Toff → t to reset (to t ≤ tmax) the maximum duration of dosing (e.g., if a OralDaily scenario is applied).
Time → tList to specify a list of intermeidate times at which numerical
integration is stopped and recalibrated (e.g., if a highly discontinuous dosing scenario is applied).
Precision → n to rest (from a default of n = ~16) digits of real-number precision used for numerical integration.
Goal → n to require only n digits of accuracy & precision
Output → outputType to return the following alternative types of output: →ODE to return the set {eq, eq0} of ODE equations at times t and 0, respectively; →VAR to return the list of 113 DimetoateModel output variables; →VARmass to return the list of 75 mass-conserving DimetoateModel output variables; →TIME to return default integration-time breakpoints (see Time option); →ONOFF to return the OnOff function generated by the selected scenario; →GainLoss to return the ODE system gains and losses; or →NDSolveArgs to return the list of NDSolve arguments used by the Mathematica NDSolve function to solve the ODE system defined when DimethoateModel is evaluated.

REFERENCES
Bogen K, Reiss R. 2017. Physiologically Based Pharmacokinetic/ Pharmacodynamic (PBPK/PD) Models for Dimethoate. Project No. 1200734.002. July 6, 2017. Exponent, Inc. Alexandria, VA.
Lee HB, Blaufox MD. Blood volume in the rat. J Nucl Med 1985; 25:72–76.
Poet T, Kousba A, Dennison S, Timchalk C. In vitro rat hepatic and intestinal metabolism of the organophosphate pesticides chlorpyrifos and diazinon. Toxicol Sci 2003; 72:193–200.
Poet T, Kousba A, Dennison S, Timchalk C. Physiologically based pharmacokinetic/ pharmacodynamic model for the organophosphorus pesticide diazinon. Neurotoxicol 2004; 25:1013–1030. \n Poet T, Timchalk C, Hotchkiss J, Bartels M. Chlorpyrifos PBPK/PD model for multiple routes of exposure. Xenobiotica 2014; 44:868–881.
Wolfram Research. 2017. Wolfram Language and System Documentation Center. Wolfram Research, Inc., Champaign, IL (www.wolfram.com), http://reference.wolfram.com/language/.

DimethoateModel initial input-parameter tables

After the ExponentDimethoateModel.m package has been loaded, initial input constants parameters are all defined assuming a default body wieght of 0.25 kg. These include the data tables mentioned in the DimethoateModel usage statement, as well as constants such as the molecular weights of DIM and OME (mwD and mwO, respectively}. All values used as final input to DimethoateModel are defined as infinte-precision integer or rational integer expressions in the form a or a/b, where a and b here denote integers. This is done to prevent unnecessary loss of either accuracy or precision in internal calculations done to integrate (i.e., solve) the ODE system defined when DimethoateModel is evaluated. Any integer or rational expression J is converted to Real (floating-pint) form when it is evaluted in a way that involves mathematical combination with a Real number (i.e., a number that contains a decimal point), as in J + 1.0 or 1.*J, or by using the *Mathematica* function N to numerically evaluate J.

Note that in *Mathematica* (as explained in more detail in the Primer that follows this Introduction), an array or vector of elements is callled a “List”, containing one or more elements separated by commas and all elements surrounded within a pair of curly braces. A “Matrix” (or data table) is any list of equally-lengthed lists (e.g., each representing a data row). *Mathematica* functions Print, TableForm, or Table, as well as *RiskQ* functions TBL or Prn, can be used to select from or print (in formatted output form) data tables or values defined by DimethoateModel,

Double square brackets are used to select specified elements within any list or list of lists, as in m[[5]] (the 5th element of the list m), m[[i, j]] (the jth element of the ith list in the vector or matrix m, or m[[{3,1, 4}]] (a list containing the 3rd, 1st, and 4th lists contained in m, in that order).

```
{Qc, QQ[2], BW, vBlood, VV[8], mwD, mwO}
```

$$\left\{\frac{7}{100\sqrt{2}}, \frac{2318029219971395}{132458812569794016\sqrt{2}}, \frac{1}{4}, \frac{3}{200}, \frac{1}{800}, \frac{11463}{50}, \frac{21319}{100}\right\}$$

```
1. {Qc, QQ[2], BW, vBlood, VV[8], mwD, mwO}
```

{0.0494975, 0.0123744, 0.25, 0.015, 0.00125, 229.26, 213.19}

```
Prn[mwD]
```

$$\text{mwD}= \frac{11463}{50}$$

```
RateTable
```

$\{\{\text{Foral}, 1\}, \{\text{Finhal}, 0.047\}, \{\text{Kmuc}, \frac{1}{50}\}, \{\text{Fplasma}, 1\}, \{\text{Kstom2intes}, 0.5\}, \{\text{Kstom2liv}, 0.5\}, \{\text{Kintes2liv}, 0.5\}, \{\text{Kintes2feces}, \frac{1}{12}\}, \{\text{FKfecar}, \frac{1}{50}\}, \{\text{Kp}, \frac{1}{20\,000}\}, \{\text{Kturn}, \{1.17\times10^{7}, 3.66\times10^{6}, 108\,600.\}\}, \{\text{Pair}, \{760\,000, 1\,000\,000\,000, 1\,000\,000\,000, 1\,000\,000\,000\}\}, \{\text{Kurx}[1], 1\}, \{\text{Kurx}[2], 1\}, \{\text{Kurx}[3], 1\}, \{\text{Kurx}[4], 1\}, \{\text{Kurx}[5], 1\}, \{\text{K35}, \frac{1}{10}\}, \{\text{K45}, \frac{1}{10}\}, \{\text{Kair}, \frac{1}{10\,000}\}, \{\text{Kalb}, 0.00875\}, \{\text{KalbX}, 100\}, \{\text{Malb}, \frac{1}{100\,000}\}\}$

```
TableForm[take = Take[RateTable, 5], 1]
```

Foral	1
Finhal	0.047
Kmuc	$\frac{1}{50}$
Fplasma	1
Kstom2intes	0.5

```
? N
```

N[*expr*] gives the numerical value of *expr*.

N[*expr*, *n*] attempts to give a result with *n*-digit precision. ≫

```
TableForm[N[take], 1]
```

Foral	1.
Finhal	0.047
Kmuc	0.02
Fplasma	1.
Kstom2intes	0.5

TBL[RateTable]

Foral	1
Finhal	0.047
Kmuc	$\frac{1}{50}$
Fplasma	1
Kstom2intes	0.5
Kstom2liv	0.5
Kintes2liv	0.5
Kintes2feces	$\frac{1}{12}$
FKfecar	$\frac{1}{50}$
Kp	$\frac{1}{20\,000}$
Kturn	1.17×10^{7} 3.66×10^{6} 108 600. 760 000
Pair	1 000 000 000 1 000 000 000 1 000 000 000
Kurx[1]	1
Kurx[2]	1
Kurx[3]	1
Kurx[4]	1
Kurx[5]	1
K35	$\frac{1}{10}$
K45	$\frac{1}{10}$
Kair	$\frac{1}{10\,000}$
Kalb	0.00875
KalbX	100
Malb	$\frac{1}{100\,000}$

TBL[MetTable]

Index	Tissue	V12	Km12	V13	Km13	V24	Km24
1	Fat	0	1	0	1	0	1
2	Liver	2530	535	107	155	1000	155
3	Rapid	0	1	0	1	0	1
4	Slow	0	1	0	1	0	1
5	Brain	0	1	0	1	0	1
6	Skin	0	1	0	1	0	1
7	Diaphr	0	1	0	1	0	1
8	Lung	0	1	0	1	0	1
9	RBC	0	1	0	1	0	1
10	Plasma	0	1	0	1	0	1

TBL[MetTableScaled]

Index	Tissue	V12	Km12	V13	Km13	V24	Km24
1	Fat	0	1	0	1	0	1
2	Liver	$\frac{253}{10}$	535	$\frac{107}{100}$	155	10	155
3	Rapid	0	1	0	1	0	1
4	Slow	0	1	0	1	0	1
5	Brain	$\frac{759}{5000}$	$\frac{107}{10}$	$\frac{321}{50\,000}$	$\frac{31}{10}$	$\frac{3}{50}$	$\frac{31}{10}$
6	Skin	$\frac{49\,841}{20\,000}$	$\frac{107}{10}$	$\frac{21\,079}{200\,000}$	$\frac{31}{10}$	$\frac{197}{200}$	$\frac{31}{10}$
7	Diaphr	$\frac{759}{2000}$	$\frac{107}{10}$	$\frac{321}{20\,000}$	$\frac{31}{10}$	$\frac{3}{20}$	$\frac{31}{10}$
8	Lung	$\frac{253}{80}$	535	$\frac{107}{800}$	155	$\frac{5}{4}$	155
9	RBC	0	1	0	1	0	1
10	Plasma	0	1	0	1	0	1

TBL[ParTable]

Index	Tissue	Vf	Qf	Pdim	Pome
1	Fat	0.07	0.09	0.464	0.197
2	Liver	0.04	0.25	1.05	0.868
3	Rapid	0.04	0.37625	1.25	0.868
4	Slow	0.456	0.136	0.945	0.868
5	Brain	0.012	0.03	1.25	0.868
6	Skin	0.197	0.058	1.06	0.868
7	Diaphr	0.03	0.006	0.945	0.868
8	Lung	0.005	0.05375	0.874	0.864
9	RBC	0.0276	0.46	1	0.868
10	Plasma	0.0324	0.54	1	0.868

TBL[ParTableScaled]

Index	Tissue	Vf	Qf	Pdim	Pome
1	Fat	$\frac{7}{400}$	$\frac{463\,605\,843\,994\,279}{73\,588\,229\,205\,441\,120\sqrt{2}}$	$\frac{58}{125}$	$\frac{197}{1000}$
2	Liver	$\frac{1}{100}$	$\frac{2\,318\,029\,219\,971\,395}{132\,458\,812\,569\,794\,016\sqrt{2}}$	$\frac{21}{20}$	$\frac{217}{250}$
3	Rapid	$\frac{1}{100}$	$\frac{139\,545\,359\,042\,277\,979}{5\,298\,352\,502\,791\,760\,640\sqrt{2}}$	$\frac{5}{4}$	$\frac{217}{250}$
4	Slow	$\frac{57}{500}$	$\frac{5\,837\,999\,516\,965}{613\,235\,243\,378\,676\sqrt{2}}$	$\frac{189}{200}$	$\frac{217}{250}$
5	Brain	$\frac{3}{1000}$	$\frac{463\,605\,843\,994\,279}{220\,764\,687\,616\,323\,360\sqrt{2}}$	$\frac{5}{4}$	$\frac{217}{250}$
6	Skin	$\frac{197}{4000}$	$\frac{13\,444\,569\,475\,834\,091}{3\,311\,470\,314\,244\,850\,400\sqrt{2}}$	$\frac{53}{50}$	$\frac{217}{250}$
7	Diaphr	$\frac{3}{400}$	$\frac{463\,605\,843\,994\,279}{1\,103\,823\,438\,081\,616\,800\sqrt{2}}$	$\frac{189}{200}$	$\frac{217}{250}$
8	Lung	$\frac{1}{800}$	$\frac{19\,935\,051\,291\,753\,997}{5\,298\,352\,502\,791\,760\,640\sqrt{2}}$	$\frac{437}{500}$	$\frac{108}{125}$
9	RBC	$\frac{69}{10\,000}$	$\frac{161}{5000\sqrt{2}}$	1	$\frac{217}{250}$
10	Plasma	$\frac{81}{10\,000}$	$\frac{189}{5000\sqrt{2}}$	1	$\frac{217}{250}$

TBL[N[ParTableScaled]]

Index	Tissue	Vf	Qf	Pdim	Pome
1.	Fat	0.0175	0.00445477	0.464	0.197
2.	Liver	0.01	0.0123744	1.05	0.868
3.	Rapid	0.01	0.0186234	1.25	0.868
4.	Slow	0.114	0.00673166	0.945	0.868
5.	Brain	0.003	0.00148492	1.25	0.868
6.	Skin	0.04925	0.00287085	1.06	0.868
7.	Diaphr	0.0075	0.000296985	0.945	0.868
8.	Lung	0.00125	0.00266049	0.874	0.864
9.	RBC	0.0069	0.0227688	1.	0.868
10.	Plasma	0.0081	0.0267286	1.	0.868

```
TBL /@ {PDTableCE, PDTableInhib, PDTableAge, PDTableDegr, PDTableReact}
```

{

Index	Tissue	AChE	BChE	CaE	Refs
1	Fat	0	0	0	
2	Liver	10 200	30 000	1.94×10^6	Maxwell87=M;M;M
3	Rapid	0	0	0	
4	Slow	0	0	0	
5	Brain	440 000	46 800	288 000	M;M;Hojring76
6	Skin	0	0	0	
7	Diaphr	0	0	0	
8	Lung	22 800	86 400	1.4×10^6	M;M;M
9	RBC	33 900	0	0	Zheng2000
10	Plasma	23 300	7850	84 000	Timchalk02;Carr01;Li05

,

Index	Tissue	AChE	BChE	CaE	Refs
1	Fat	0	0	0	
2	Liver	0.054	0.0048	0.005	Herzsprung92=H;H;?
3	Rapid	0	0	0	
4	Slow	0	0	0	
5	Brain	0.054	0.0048	0.005	H;H;?
6	Skin	0	0	0	
7	Diaphr	0	0	0	
8	Lung	0.054	0.0048	0.005	H;H;?
9	RBC	0.054	0	0	H;H;?
10	Plasma	0.054	0.0048	0.005	H;H;?

,

Index	Tissue	AChE	BChE	CaE	Refs
1	Fat	0	0	0	
2	Liver	0.022	0.12	0	Mason2000=M;M;?
3	Rapid	0	0	0	
4	Slow	0	0	0	
5	Brain	0.022	0.12	0	M;M;?
6	Skin	0	0	0	
7	Diaphr	0	0	0	
8	Lung	0.022	0.12	0	M;M;?
9	RBC	0.022	0	0	M;M;?
10	Plasma	0.022	0.12	0	M;M;?

,

Index	Tissue	AChE	BChE	CaE	Refs
1	Fat	0	0	0	
2	Liver	0.003	0.01	0.001	Timchalk02=T;T;T
3	Rapid	0	0	0	
4	Slow	0	0	0	
5	Brain	0.003	0.01	0.000754	T;T;T
6	Skin	0	0	0	
7	Diaphr	0	0	0	
8	Lung	0.003	0.01	0.001	T;T;T
9	RBC	0.003	0	0	T;T;T
10	Plasma	0.003	0.01	0.001	T;T;T

,

Index	Tissue	AChE	BChE	CaE	Refs
1	Fat	0	0	0	
2	Liver	0.019	0.03	0.005	Mason2000=M;M;?
3	Rapid	0	0	0	
4	Slow	0	0	0	
5	Brain	0.019	0.03	0.005	M;M;?
6	Skin	0	0	0	
7	Diaphr	0	0	0	
8	Lung	0.019	0.03	0.005	M;M;?
9	RBC	0.019	0	0	M;M;?
10	Plasma	0.019	0.03	0.005	M;M;?

}

```
Table[1. VV[i], {i, 10}]
```

{0.0175, 0.01, 0.01, 0.114, 0.003, 0.04925, 0.0075, 0.00125, 0.0069, 0.0081}

```
VV /@ Range[10]
```

$$\left\{\frac{7}{400}, \frac{1}{100}, \frac{1}{100}, \frac{57}{500}, \frac{3}{1000}, \frac{197}{4000}, \frac{3}{400}, \frac{1}{800}, \frac{69}{10\,000}, \frac{81}{10\,000}\right\}$$

```
1. QQ /@ Range[10]
```

{0.00445477, 0.0123744, 0.0186234, 0.00673166, 0.00148492,
 0.00287085, 0.000296985, 0.00266049, 0.0227688, 0.0267286}

```
TBL[Table[1. PP[i, j], {i, 10}, {j, 5}]]
```

0.464	0.197	0.197	0.197	0.197
1.05	0.868	0.868	0.868	0.868
1.25	0.868	0.868	0.868	0.868
0.945	0.868	0.868	0.868	0.868
1.25	0.868	0.868	0.868	0.868
1.06	0.868	0.868	0.868	0.868
0.945	0.868	0.868	0.868	0.868
0.874	0.864	0.864	0.864	0.864
1.	0.868	0.868	0.868	0.868
1.	0.868	0.868	0.868	0.868

Evaluate DimethoateModel using Output options

After the ExponentDimethoateModel.m package has been loaded, you can evaluate the DimethoateModel function as illustrated below, first using Output options for which the first four DimethoateModel function inputs can be dummy variables because they are not required to generate the specified output. For illustration though, we first here set body weight to 200 grams = 0.2 kg (BW = 1/5), to show (below) how just doing this automatically resets many input parameter values.

```
BW = 1 / 5;
dd = mwO / 1000;
vars = DimethoateModel[BW, dd, 0, 0, 78,
  Scenario → Oral, Chemical → Omethoate, Output → VAR];
varsMassConserving = varM = DimethoateModel[BW, dd, 0, 0, 78,
    Scenario → Oral, Chemical → Omethoate, Output → VARmass];
{eq, eq0} = ode = DimethoateModel[BW, dd, 0, 0, 78, Scenario → Oral,
    Chemical → Omethoate, Output → ODE];
Length /@
 {vars,
  varM}
```

{113, 75}

The ExDM model parameters are:

```
vars

{MM[1, 1][t], MM[1, 2][t], MM[1, 3][t], MM[1, 4][t], MM[1, 5][t], MM[2, 1][t],
 MM[2, 2][t], MM[2, 3][t], MM[2, 4][t], MM[2, 5][t], MM[3, 1][t], MM[3, 2][t],
 MM[3, 3][t], MM[3, 4][t], MM[3, 5][t], MM[4, 1][t], MM[4, 2][t], MM[4, 3][t],
 MM[4, 4][t], MM[4, 5][t], MM[5, 1][t], MM[5, 2][t], MM[5, 3][t], MM[5, 4][t],
 MM[5, 5][t], MM[6, 1][t], MM[6, 2][t], MM[6, 3][t], MM[6, 4][t], MM[6, 5][t],
 MM[7, 1][t], MM[7, 2][t], MM[7, 3][t], MM[7, 4][t], MM[7, 5][t], MM[8, 1][t],
 MM[8, 2][t], MM[8, 3][t], MM[8, 4][t], MM[8, 5][t], MM[9, 2][t], MM[10, 2][t],
 Stomach[t], Intest[t], StomachUnAbs[t], IntestUnAbs[t], FeCar[t], U[1][t],
 U[2][t], U[3][t], U[4][t], U[5][t], AlbX[t], CEX[1, 1][t], CEX[1, 2][t],
 CEX[1, 3][t], CEX[2, 1][t], CEX[2, 2][t], CEX[2, 3][t], CEX[3, 1][t], CEX[3, 2][t],
 CEX[3, 3][t], CEX[4, 1][t], CEX[4, 2][t], CEX[4, 3][t], CEX[5, 1][t], CEX[5, 2][t],
 CEX[5, 3][t], CEX[6, 1][t], CEX[6, 2][t], CEX[6, 3][t], CEX[7, 1][t], CEX[7, 2][t],
 CEX[7, 3][t], CEX[8, 1][t], CEX[8, 2][t], CEX[8, 3][t], CEX[9, 1][t],
 CEX[9, 2][t], CEX[9, 3][t], CEX[10, 1][t], CEX[10, 2][t], CEX[10, 3][t],
 CE[1, 1][t], CE[1, 2][t], CE[1, 3][t], CE[2, 1][t], CE[2, 2][t], CE[2, 3][t],
 CE[3, 1][t], CE[3, 2][t], CE[3, 3][t], CE[4, 1][t], CE[4, 2][t], CE[4, 3][t],
 CE[5, 1][t], CE[5, 2][t], CE[5, 3][t], CE[6, 1][t], CE[6, 2][t], CE[6, 3][t],
 CE[7, 1][t], CE[7, 2][t], CE[7, 3][t], CE[8, 1][t], CE[8, 2][t], CE[8, 3][t],
 CE[9, 1][t], CE[9, 2][t], CE[9, 3][t], CE[10, 1][t], CE[10, 2][t], CE[10, 3][t]}

varM

{MM[1, 1][t], MM[1, 2][t], MM[1, 3][t], MM[1, 4][t], MM[1, 5][t], MM[2, 1][t],
 MM[2, 2][t], MM[2, 3][t], MM[2, 4][t], MM[2, 5][t], MM[3, 1][t], MM[3, 2][t],
 MM[3, 3][t], MM[3, 4][t], MM[3, 5][t], MM[4, 1][t], MM[4, 2][t], MM[4, 3][t],
 MM[4, 4][t], MM[4, 5][t], MM[5, 1][t], MM[5, 2][t], MM[5, 3][t], MM[5, 4][t],
 MM[5, 5][t], MM[6, 1][t], MM[6, 2][t], MM[6, 3][t], MM[6, 4][t], MM[6, 5][t],
 MM[7, 1][t], MM[7, 2][t], MM[7, 3][t], MM[7, 4][t], MM[7, 5][t], MM[8, 1][t],
 MM[8, 2][t], MM[8, 3][t], MM[8, 4][t], MM[8, 5][t], Stomach[t], Intest[t],
 StomachUnAbs[t], IntestUnAbs[t], FeCar[t], U[1][t], U[2][t], U[3][t], U[4][t],
 U[5][t], AlbX[t], CEX[1, 1][t], CEX[1, 2][t], CEX[1, 3][t], CEX[2, 1][t],
 CEX[2, 2][t], CEX[2, 3][t], CEX[3, 1][t], CEX[3, 2][t], CEX[3, 3][t],
 CEX[4, 1][t], CEX[4, 2][t], CEX[4, 3][t], CEX[5, 1][t], CEX[5, 2][t],
 CEX[5, 3][t], CEX[6, 1][t], CEX[6, 2][t], CEX[6, 3][t], CEX[7, 1][t],
 CEX[7, 2][t], CEX[7, 3][t], CEX[8, 1][t], CEX[8, 2][t], CEX[8, 3][t]}
```

The ExDM ODE system and its initial values are given by (the following cells are now hidden to save viewing space; double-click the right-side outer bracket to unhide, view, then re-hide these large Output cells):

```
eq
```

$$\{\mathrm{MM}[1,1]'[t] == \Big(463\,605\,843\,994\,279\,\Big(-\frac{31\,250}{203}\,\mathrm{MM}[1,1][t] + \big(125\,\big(183\,578\,018\,058\,273\,878\,978\,017\,500\,\mathrm{MM}[1,1][t] + 29\,\big(13\,598\,371\,708\,020\,287\,331\,705\,000\,\mathrm{MM}[2,1][t] + 17\,191\,061\,513\,279\,247\,244\,741\,461\,\mathrm{MM}[3,1][t] +$$

```
        721 005 283 544 156 226 000 000 MM[4, 1][t] + 2 649 176 251 395 880
         (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
           52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
    1 656 281 674 036 871 197 067 684 016)) / (36 794 114 602 720 560 × 5^(3/4)),
MM[1, 2]'[t] == (463 605 843 994 279 (-500 000 MM[1, 2][t] / 1379 +
    (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
         MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
        319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800 MM[5, 2][
          t] + 315 294 360 736 132 054 080 MM[6, 2][t] + 214 182 720 913 855 222 944
         MM[7, 2][t] + 11 565 619 032 680 457 669 505 MM[8, 2][t])) /
    2 904 853 152 394 161 812 061 792)) / (36 794 114 602 720 560 × 5^(3/4)),
MM[1, 3]'[t] == (463 605 843 994 279 (-500 000 MM[1, 3][t] / 1379 +
    (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
         MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
        319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800 MM[5, 3][
          t] + 315 294 360 736 132 054 080 MM[6, 3][t] + 214 182 720 913 855 222 944
         MM[7, 3][t] + 11 565 619 032 680 457 669 505 MM[8, 3][t])) /
    2 904 853 152 394 161 812 061 792)) / (36 794 114 602 720 560 × 5^(3/4)),
MM[1, 4]'[t] == (463 605 843 994 279 (-500 000 MM[1, 4][t] / 1379 +
    (15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
         MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
        319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800 MM[5, 4][
          t] + 315 294 360 736 132 054 080 MM[6, 4][t] + 214 182 720 913 855 222 944
         MM[7, 4][t] + 11 565 619 032 680 457 669 505 MM[8, 4][t])) /
    2 904 853 152 394 161 812 061 792)) / (36 794 114 602 720 560 × 5^(3/4)),
MM[1, 5]'[t] == (463 605 843 994 279 (-500 000 MM[1, 5][t] / 1379 +
    (15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
         MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
        319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800 MM[5, 5][
          t] + 315 294 360 736 132 054 080 MM[6, 5][t] + 214 182 720 913 855 222 944
         MM[7, 5][t] + 11 565 619 032 680 457 669 505 MM[8, 5][t])) /
    2 904 853 152 394 161 812 061 792)) / (36 794 114 602 720 560 × 5^(3/4)),
MM[2, 1]'[t] == -107 MM[2, 1][t] / (125 (31/25 + MM[2, 1][t])) -
```

$$\frac{506\,\text{MM}[2,1][t]}{25\left(\frac{107}{25}+\text{MM}[2,1][t]\right)}+$$

$$\Big(463\,605\,843\,994\,279\times 5^{1/4}\Big(-\frac{2500}{21}\,\text{MM}[2,1][t]+\Big(125\,\Big(183\,578\,018\,058\,273\,878\,978\,017\,500\,\text{MM}[1,1][t]+29\,\Big(13\,598\,371\,708\,020\,287\,331\,705\,000\,\text{MM}[2,1][t]+17\,191\,061\,513\,279\,247\,244\,741\,461\,\text{MM}[3,1][t]+721\,005\,283\,544\,156\,226\,000\,000\,\text{MM}[4,1][t]+2\,649\,176\,251\,395\,880\,\Big(1\,724\,707\,026\,\text{MM}[5,1][t]+239\,519\,700\,\text{MM}[6,1][t]+52\,205\,\big(3496\,\text{MM}[7,1][t]+203\,175\,\text{MM}[8,1][t]\big)\Big)\Big)\Big)\Big)\Big/1\,656\,281\,674\,036\,871\,197\,067\,684\,016\Big)\Big)\Big/66\,229\,406\,284\,897\,008,$$

$$\text{MM}[2,2]'[t]==\frac{\text{Intest}[t]}{2}+\frac{\text{Stomach}[t]}{2}+\frac{506\,\text{MM}[2,1][t]}{25\left(\frac{107}{25}+\text{MM}[2,1][t]\right)}-\frac{27}{4}\,\text{CE}[2,1][t]\,\text{MM}[2,2][t]-\frac{3}{5}\,\text{CE}[2,2][t]\,\text{MM}[2,2][t]-\frac{5}{8}\,\text{CE}[2,3][t]\,\text{MM}[2,2][t]-\frac{8\,\text{MM}[2,2][t]}{\frac{31}{25}+\text{MM}[2,2][t]}+$$

$$\Big(463\,605\,843\,994\,279\times 5^{1/4}\Big(-\frac{31\,250}{217}\,\text{MM}[2,2][t]+\Big(15\,625\,\big(6\,066\,698\,389\,336\,609\,868\,160\,\text{MM}[1,2][t]+6\,693\,210\,028\,557\,975\,717\,000\,\text{MM}[2,2][t]+10\,073\,281\,092\,979\,753\,454\,085\,\text{MM}[3,2][t]+319\,395\,285\,573\,293\,160\,000\,\text{MM}[4,2][t]+2\,677\,284\,011\,423\,190\,286\,800\,\text{MM}[5,2][t]+315\,294\,360\,736\,132\,054\,080\,\text{MM}[6,2][t]+214\,182\,720\,913\,855\,222\,944\,\text{MM}[7,2][t]+11\,565\,619\,032\,680\,457\,669\,505\,\text{MM}[8,2][t]\big)\Big)\Big/2\,904\,853\,152\,394\,161\,812\,061\,792\Big)\Big)\Big/66\,229\,406\,284\,897\,008-\frac{\text{MM}[10,2][t]}{1000},\ \text{MM}[2,3]'[t]==$$

$$\frac{107\,\text{MM}[2,1][t]}{125\left(\frac{31}{25}+\text{MM}[2,1][t]\right)}-$$

$$\frac{107\,MM[2,3][t]}{1550} + \Big(463\,605\,843\,994\,279 \times 5^{1/4} \Big(-\frac{31\,250}{217} MM[2,3][t] + \big(15\,625\,(6\,066\,698\,389\,336\,609\,868\,160\,MM[1,3][t] + 6\,693\,210\,028\,557\,975\,717\,000\,MM[2,3][t] + 10\,073\,281\,092\,979\,753\,454\,085\,MM[3,3][t] + 319\,395\,285\,573\,293\,160\,000\,MM[4,3][t] + 2\,677\,284\,011\,423\,190\,286\,800\,MM[5,3][t] + 315\,294\,360\,736\,132\,054\,080\,MM[6,3][t] + 214\,182\,720\,913\,855\,222\,944\,MM[7,3][t] + 11\,565\,619\,032\,680\,457\,669\,505\,MM[8,3][t])\big) \Big/ 2\,904\,853\,152\,394\,161\,812\,061\,792\Big)\Big) \Big/$$

$$66\,229\,406\,284\,897\,008,\; MM[2,4]'[t] == \frac{8\,MM[2,2][t]}{\frac{31}{25} + MM[2,2][t]} - \frac{20}{31} MM[2,4][t] + \Big(463\,605\,843\,994\,279 \times 5^{1/4} \Big(-\frac{31\,250}{217} MM[2,4][t] + \big(15\,625\,(6\,066\,698\,389\,336\,609\,868\,160\,MM[1,4][t] + 6\,693\,210\,028\,557\,975\,717\,000\,MM[2,4][t] + 10\,073\,281\,092\,979\,753\,454\,085\,MM[3,4][t] + 319\,395\,285\,573\,293\,160\,000\,MM[4,4][t] + 2\,677\,284\,011\,423\,190\,286\,800\,MM[5,4][t] + 315\,294\,360\,736\,132\,054\,080\,MM[6,4][t] + 214\,182\,720\,913\,855\,222\,944\,MM[7,4][t] + 11\,565\,619\,032\,680\,457\,669\,505\,MM[8,4][t])\big) \Big/ 2\,904\,853\,152\,394\,161\,812\,061\,792\Big)\Big) \Big/$$

$$66\,229\,406\,284\,897\,008,\; MM[2,5]'[t] == \frac{107\,MM[2,3][t]}{1550} + \frac{20}{31} MM[2,4][t] - \frac{1}{50} MM[2,5][t] + \Big(463\,605\,843\,994\,279 \times 5^{1/4} \Big(-\frac{31\,250}{217} MM[2,5][t] + \big(15\,625\,(6\,066\,698\,389\,336\,609\,868\,160\,MM[1,5][t] + 6\,693\,210\,028\,557\,975\,717\,000\,MM[2,5][t] + 10\,073\,281\,092\,979\,753\,454\,085\,MM[3,5][t] + 319\,395\,285\,573\,293\,160\,000\,MM[4,5][t] + 2\,677\,284\,011\,423\,190\,286\,800\,MM[5,5][t] + 315\,294\,360\,736\,132\,054\,080\,MM[6,5][t] + 214\,182\,720\,913\,855\,222\,944\,MM[7,5][t] + 11\,565\,619\,032\,680\,457\,669\,505\,MM[8,5][t])\big) \Big/ 2\,904\,853\,152\,394\,161\,812\,061\,792\Big)\Big) \Big/ 66\,229\,406\,284\,897\,008,$$

```
MM[3, 1]′[t] == -MM[3, 1][t] + (139 545 359 042 277 979 (-100 MM[3, 1][t] +
        (125 (183 578 018 058 273 878 978 017 500 MM[1, 1][t] +
              29 (13 598 371 708 020 287 331 705 000 MM[2, 1][t] +
                  17 191 061 513 279 247 244 741 461 MM[3, 1][t] +
                  721 005 283 544 156 226 000 000 MM[4, 1][t] +
                  2 649 176 251 395 880 (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
                      52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
          1 656 281 674 036 871 197 067 684 016)) / (2 649 176 251 395 880 320 × 5^(3/4)),
MM[3, 2]′[t] == -MM[3, 2][t] + (139 545 359 042 277 979
      (-31 250/217 MM[3, 2][t] +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
                MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
              319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800
                MM[5, 2][t] + 315 294 360 736 132 054 080 MM[6, 2][t] +
              214 182 720 913 855 222 944 MM[7, 2][t] + 11 565 619 032 680 457 669 505
                MM[8, 2][t])) / 2 904 853 152 394 161 812 061 792)) /
    (2 649 176 251 395 880 320 × 5^(3/4)), MM[3, 3]′[t] == -MM[3, 3][
     t] +
   (139 545 359 042 277 979 (-31 250/217 MM[3, 3][t] +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
                MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
              319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800
                MM[5, 3][t] + 315 294 360 736 132 054 080 MM[6, 3][t] +
              214 182 720 913 855 222 944 MM[7, 3][t] + 11 565 619 032 680 457 669 505
                MM[8, 3][t])) / 2 904 853 152 394 161 812 061 792)) /
    (2 649 176 251 395 880 320 × 5^(3/4)), MM[3, 4]′[t] == -MM[3, 4][
     t] +
   (139 545 359 042 277 979 (-31 250/217 MM[3, 4][t] +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
                MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
              319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800
                MM[5, 4][t] + 315 294 360 736 132 054 080 MM[6, 4][t] +
              214 182 720 913 855 222 944 MM[7, 4][t] + 11 565 619 032 680 457 669 505
                MM[8, 4][t])) / 2 904 853 152 394 161 812 061 792)) /
    (2 649 176 251 395 880 320 × 5^(3/4)), MM[3, 5]′[t] == -51/50
    MM[
      3,
```

```
    5][t] +
  (139 545 359 042 277 979 (- 31 250/217 MM[3, 5][t] +
       (15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
             MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
            319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800
             MM[5, 5][t] + 315 294 360 736 132 054 080 MM[6, 5][t] +
            214 182 720 913 855 222 944 MM[7, 5][t] + 11 565 619 032 680 457 669 505
             MM[8, 5][t])) / 2 904 853 152 394 161 812 061 792)) /
   (2 649 176 251 395 880 320 × 5^(3/4)), MM[4, 1]'[t] == (1 167 599 903 393 ×
   5^(1/4)
   (- 125 000 MM[4, 1][t]/10 773 +
      (125 (183 578 018 058 273 878 978 017 500 MM[1, 1][t] +
           29 (13 598 371 708 020 287 331 705 000 MM[2, 1][t] +
              17 191 061 513 279 247 244 741 461 MM[3, 1][t] +
              721 005 283 544 156 226 000 000 MM[4, 1][t] + 2 649 176 251 395 880
               (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
                 52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
       1 656 281 674 036 871 197 067 684 016)) / 306 617 621 689 338,
MM[4, 2]'[t] == (1 167 599 903 393 × 5^(1/4) (- 156 250 MM[4, 2][t]/12 369 +
      (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
            MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
           319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800 MM[5, 2][
             t] + 315 294 360 736 132 054 080 MM[6, 2][t] + 214 182 720 913 855 222 944
            MM[7, 2][t] + 11 565 619 032 680 457 669 505 MM[8, 2][t])) /
       2 904 853 152 394 161 812 061 792)) / 306 617 621 689 338,
MM[4, 3]'[t] == (1 167 599 903 393 × 5^(1/4) (- 156 250 MM[4, 3][t]/12 369 +
      (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
            MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
           319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800 MM[5, 3][
             t] + 315 294 360 736 132 054 080 MM[6, 3][t] + 214 182 720 913 855 222 944
            MM[7, 3][t] + 11 565 619 032 680 457 669 505 MM[8, 3][t])) /
       2 904 853 152 394 161 812 061 792)) / 306 617 621 689 338,
MM[4, 4]'[t] == (1 167 599 903 393 × 5^(1/4) (- 156 250 MM[4, 4][t]/12 369 +
      (15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
            MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
```

```
            319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800 MM[5, 4][
              t] + 315 294 360 736 132 054 080 MM[6, 4][t] + 214 182 720 913 855 222 944
             MM[7, 4][t] + 11 565 619 032 680 457 669 505 MM[8, 4][t])) /
        2 904 853 152 394 161 812 061 792)) / 306 617 621 689 338,
MM[4, 5]'[t] == (1 167 599 903 393 × 5^(1/4) (-(156 250 MM[4, 5][t])/12 369 +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
             MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
            319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800 MM[5, 5][
              t] + 315 294 360 736 132 054 080 MM[6, 5][t] + 214 182 720 913 855 222 944
             MM[7, 5][t] + 11 565 619 032 680 457 669 505 MM[8, 5][t])) /
        2 904 853 152 394 161 812 061 792)) / 306 617 621 689 338,
MM[5, 1]'[t] == (463 605 843 994 279 (-1000/3 MM[5, 1][t] +
        (125 (183 578 018 058 273 878 978 017 500 MM[1, 1][t] +
            29 (13 598 371 708 020 287 331 705 000 MM[2, 1][t] +
               17 191 061 513 279 247 244 741 461 MM[3, 1][t] +
               721 005 283 544 156 226 000 000 MM[4, 1][t] + 2 649 176 251 395 880
                (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
                  52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
        1 656 281 674 036 871 197 067 684 016)) / (110 382 343 808 161 680 × 5^(3/4)),
MM[5, 2]'[t] == (463 605 843 994 279 (-312 500/651 MM[5, 2][t] +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
             MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
            319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800 MM[5, 2][
              t] + 315 294 360 736 132 054 080 MM[6, 2][t] + 214 182 720 913 855 222 944
             MM[7, 2][t] + 11 565 619 032 680 457 669 505 MM[8, 2][t])) /
        2 904 853 152 394 161 812 061 792)) / (110 382 343 808 161 680 × 5^(3/4)),
MM[5, 3]'[t] == (463 605 843 994 279 (-312 500/651 MM[5, 3][t] +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
             MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
            319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800 MM[5, 3][
              t] + 315 294 360 736 132 054 080 MM[6, 3][t] + 214 182 720 913 855 222 944
             MM[7, 3][t] + 11 565 619 032 680 457 669 505 MM[8, 3][t])) /
        2 904 853 152 394 161 812 061 792)) / (110 382 343 808 161 680 × 5^(3/4)),
MM[5, 4]'[t] == (463 605 843 994 279 (-312 500/651 MM[5, 4][t] +
        (15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
```

```
                MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
               319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800 MM[5, 4][
                t] + 315 294 360 736 132 054 080 MM[6, 4][t] + 214 182 720 913 855 222 944
                MM[7, 4][t] + 11 565 619 032 680 457 669 505 MM[8, 4][t])) /
          2 904 853 152 394 161 812 061 792)) / (110 382 343 808 161 680 × 5^(3/4)),
MM[5, 5]'[t] == (463 605 843 994 279 (- 312 500/651 MM[5, 5][t] +
          (15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
                MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
               319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800 MM[5, 5][
                t] + 315 294 360 736 132 054 080 MM[6, 5][t] + 214 182 720 913 855 222 944
                MM[7, 5][t] + 11 565 619 032 680 457 669 505 MM[8, 5][t])) /
          2 904 853 152 394 161 812 061 792)) / (110 382 343 808 161 680 × 5^(3/4)),
MM[6, 1]'[t] == - (21 079 MM[6, 1][t]) / (250 000 (6107/50 000 + MM[6, 1][t])) -
    (49 841 MM[6, 1][t]) / (25 000 (21 079/50 000 + MM[6, 1][t])) +
    (13 444 569 475 834 091
       (- (250 000 MM[6, 1][t])/10 441 + (125 (183 578 018 058 273 878 978 017 500 MM[1, 1][t] +
                29 (13 598 371 708 020 287 331 705 000 MM[2, 1][t] +
                   17 191 061 513 279 247 244 741 461 MM[3, 1][t] +
                   721 005 283 544 156 226 000 000 MM[4, 1][t] +
                   2 649 176 251 395 880 (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
                      52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
           1 656 281 674 036 871 197 067 684 016)) / (1 655 735 157 122 425 200 × 5^(3/4)),
MM[6, 2]'[t] == (49 841 MM[6, 1][t]) / (25 000 (21 079/50 000 + MM[6, 1][t])) -
    (197 MM[6, 2][t]) / (250 (6107/50 000 + MM[6, 2][t])) +
    (13 444 569 475 834 091
       (- (1 250 000 MM[6, 2][t])/42 749 +
          (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
                MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
               319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800
                MM[5, 2][t] + 315 294 360 736 132 054 080 MM[6, 2][t] +
               214 182 720 913 855 222 944 MM[7, 2][t] + 11 565 619 032 680 457 669 505
```

```
          MM[8, 2][t])) / 2 904 853 152 394 161 812 061 792)) /
  (1 655 735 157 122 425 200 × 5^(3/4)), MM[6, 3]'[t] ==
  (21 079 MM[6, 1][t]) / (250 000 (6107/50 000 + MM[6, 1][t])) -
  (107 MM[6, 3][t]) / 1550 +
  (13 444 569 475 834 091
     (-(1 250 000 MM[6, 3][t]) / 42 749 +
       (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
            MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
           319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800
            MM[5, 3][t] + 315 294 360 736 132 054 080 MM[6, 3][t] +
           214 182 720 913 855 222 944 MM[7, 3][t] + 11 565 619 032 680 457 669 505
            MM[8, 3][t])) / 2 904 853 152 394 161 812 061 792)) /
  (1 655 735 157 122 425 200 × 5^(3/4)), MM[6, 4]'[t] ==
  (197 MM[6, 2][t]) / (250 (6107/50 000 + MM[6, 2][t])) -
  20/31
   MM[
    6,
    4][t] +
  (13 444 569 475 834 091 (-(1 250 000 MM[6, 4][t]) / 42 749 +
       (15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
            MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
           319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800
            MM[5, 4][t] + 315 294 360 736 132 054 080 MM[6, 4][t] +
           214 182 720 913 855 222 944 MM[7, 4][t] + 11 565 619 032 680 457 669 505
            MM[8, 4][t])) / 2 904 853 152 394 161 812 061 792)) /
  (1 655 735 157 122 425 200 × 5^(3/4)), MM[6, 5]'[t] == (107 MM[6, 3][t]) / 1550 +
  20/31
   MM[
    6,
    4][t] +
  (13 444 569 475 834 091 (-(1 250 000 MM[6, 5][t]) / 42 749 +
```

```
(15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
        MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
       319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800
        MM[5, 5][t] + 315 294 360 736 132 054 080 MM[6, 5][t] +
       214 182 720 913 855 222 944 MM[7, 5][t] + 11 565 619 032 680 457 669 505
        MM[8, 5][t])) / 2 904 853 152 394 161 812 061 792)) /
  (1 655 735 157 122 425 200 * 5^(3/4)), MM[7, 1]'[t] ==
 -(321 MM[7, 1][t])/(25 000 (93/5000 + MM[7, 1][t])) -
  (759 MM[7, 1][t])/(2500 (321/5000 + MM[7, 1][t])) +
  (463 605 843 994 279
    (-(100 000)/(567) MM[7, 1][t] + (125 (183 578 018 058 273 878 978 017 500 MM[1, 1][t] +
          29 (13 598 371 708 020 287 331 705 000 MM[2, 1][t] +
             17 191 061 513 279 247 244 741 461 MM[3, 1][t] +
             721 005 283 544 156 226 000 000 MM[4, 1][t] +
             2 649 176 251 395 880 (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
                52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
        1 656 281 674 036 871 197 067 684 016)) / (551 911 719 040 808 400 * 5^(3/4)),
 MM[7, 2]'[t] == (759 MM[7, 1][t])/(2500 (321/5000 + MM[7, 1][t])) -
  (3 MM[7, 2][t])/(25 (93/5000 + MM[7, 2][t])) +
  (463 605 843 994 279
    (-(125 000)/(651) MM[7, 2][t] +
      (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
        MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
       319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800
        MM[5, 2][t] + 315 294 360 736 132 054 080 MM[6, 2][t] +
       214 182 720 913 855 222 944 MM[7, 2][t] + 11 565 619 032 680 457 669 505
        MM[8, 2][t])) / 2 904 853 152 394 161 812 061 792)) /
  (551 911 719 040 808 400 * 5^(3/4)), MM[7, 3]'[t] ==
 (321 MM[7, 1][t])/(25 000 (93/5000 + MM[7, 1][t])) -
```

```
107 MM[7, 3][t]
─────────────── +
     1550
(463 605 843 994 279
   (- 125 000/651 MM[7, 3][t] +
     (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
           MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
          319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800
           MM[5, 3][t] + 315 294 360 736 132 054 080 MM[6, 3][t] +
          214 182 720 913 855 222 944 MM[7, 3][t] + 11 565 619 032 680 457 669 505
           MM[8, 3][t])) / 2 904 853 152 394 161 812 061 792)) /
 (551 911 719 040 808 400 × 5^(3/4)), MM[7, 4]′[t] == 3 MM[7, 2][t] / (25 (93/5000 + MM[7, 2][t])) -
20
──
31
 MM[
   7,
   4][t] +
(463 605 843 994 279 (- 125 000/651 MM[7, 4][t] +
     (15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
           MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
          319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800
           MM[5, 4][t] + 315 294 360 736 132 054 080 MM[6, 4][t] +
          214 182 720 913 855 222 944 MM[7, 4][t] + 11 565 619 032 680 457 669 505
           MM[8, 4][t])) / 2 904 853 152 394 161 812 061 792)) /
 (551 911 719 040 808 400 × 5^(3/4)), MM[7, 5]′[t] == 107 MM[7, 3][t] / 1550 +
20
──
31
 MM[
   7,
   4][t] +
(463 605 843 994 279 (- 125 000/651 MM[7, 5][t] +
     (15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
           MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
          319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800
           MM[5, 5][t] + 315 294 360 736 132 054 080 MM[6, 5][t] +
          214 182 720 913 855 222 944 MM[7, 5][t] + 11 565 619 032 680 457 669 505
           MM[8, 5][t])) / 2 904 853 152 394 161 812 061 792)) /
```

```
  (551 911 719 040 808 400 × 5^(3/4)), MM[8, 1]'[t] ==
 -(107 MM[8, 1][t])/(1000 (31/200 + MM[8, 1][t])) -
  (253 MM[8, 1][t])/(100 (107/200 + MM[8, 1][t])) +
  (19 935 051 291 753 997
     (-(500 000/437) MM[8, 1][t] + (125 (183 578 018 058 273 878 978 017 500 MM[1, 1][t] +
              29 (13 598 371 708 020 287 331 705 000 MM[2, 1][t] +
                 17 191 061 513 279 247 244 741 461 MM[3, 1][t] +
                 721 005 283 544 156 226 000 000 MM[4, 1][t] +
                 2 649 176 251 395 880 (1 724 707 026 MM[5, 1][t] + 239 519 700 MM[6, 1][t] +
                    52 205 (3496 MM[7, 1][t] + 203 175 MM[8, 1][t]))))) /
          1 656 281 674 036 871 197 067 684 016)) / (2 649 176 251 395 880 320 × 5^(3/4)),
 MM[8, 2]'[t] == (253 MM[8, 1][t])/(100 (107/200 + MM[8, 1][t])) -
  MM[8, 2][t]/(31/200 + MM[8, 2][t]) +
  (19 935 051 291 753 997
     (-(31 250/27) MM[8, 2][t] +
       (15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
               MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
             319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800
               MM[5, 2][t] + 315 294 360 736 132 054 080 MM[6, 2][t] +
             214 182 720 913 855 222 944 MM[7, 2][t] + 11 565 619 032 680 457 669 505
               MM[8, 2][t])) / 2 904 853 152 394 161 812 061 792)) /
  (2 649 176 251 395 880 320 × 5^(3/4)), MM[8, 3]'[t] ==
 (107 MM[8, 1][t])/(1000 (31/200 + MM[8, 1][t])) -
  (107 MM[8, 3][t])/1550 +
  (19 935 051 291 753 997
     (-(31 250/27) MM[8, 3][t] +
       (15 625 (6 066 698 389 336 609 868 160 MM[1, 3][t] + 6 693 210 028 557 975 717 000
               MM[2, 3][t] + 10 073 281 092 979 753 454 085 MM[3, 3][t] +
             319 395 285 573 293 160 000 MM[4, 3][t] + 2 677 284 011 423 190 286 800
```

MM[5, 3][t] + 315 294 360 736 132 054 080 MM[6, 3][t] +
214 182 720 913 855 222 944 MM[7, 3][t] + 11 565 619 032 680 457 669 505
MM[8, 3][t])) / 2 904 853 152 394 161 812 061 792))) /
(2 649 176 251 395 880 320 × $5^{3/4}$), MM[8, 4]′[t] == $\frac{\text{MM[8, 2][t]}}{\frac{31}{200} + \text{MM[8, 2][t]}}$ −
$\frac{20}{31}$
MM[
8,
4][t] +
(19 935 051 291 753 997 (− $\frac{31250}{27}$ MM[8, 4][t] +
(15 625 (6 066 698 389 336 609 868 160 MM[1, 4][t] + 6 693 210 028 557 975 717 000
MM[2, 4][t] + 10 073 281 092 979 753 454 085 MM[3, 4][t] +
319 395 285 573 293 160 000 MM[4, 4][t] + 2 677 284 011 423 190 286 800
MM[5, 4][t] + 315 294 360 736 132 054 080 MM[6, 4][t] +
214 182 720 913 855 222 944 MM[7, 4][t] + 11 565 619 032 680 457 669 505
MM[8, 4][t])) / 2 904 853 152 394 161 812 061 792))) /
(2 649 176 251 395 880 320 × $5^{3/4}$), MM[8, 5]′[t] == $\frac{\text{107 MM[8, 3][t]}}{1550}$ +
$\frac{20}{31}$
MM[
8,
4][t] +
(19 935 051 291 753 997 (− $\frac{31250}{27}$ MM[8, 5][t] +
(15 625 (6 066 698 389 336 609 868 160 MM[1, 5][t] + 6 693 210 028 557 975 717 000
MM[2, 5][t] + 10 073 281 092 979 753 454 085 MM[3, 5][t] +
319 395 285 573 293 160 000 MM[4, 5][t] + 2 677 284 011 423 190 286 800
MM[5, 5][t] + 315 294 360 736 132 054 080 MM[6, 5][t] +
214 182 720 913 855 222 944 MM[7, 5][t] + 11 565 619 032 680 457 669 505
MM[8, 5][t])) / 2 904 853 152 394 161 812 061 792))) /
(2 649 176 251 395 880 320 × $5^{3/4}$), MM[9, 2]′[t] == $\frac{1}{50 \times 5^{3/4}}$
7
((15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] + 6 693 210 028 557 975 717 000
MM[2, 2][t] + 10 073 281 092 979 753 454 085 MM[3, 2][t] +
319 395 285 573 293 160 000 MM[4, 2][t] + 2 677 284 011 423 190 286 800 MM[5, 2][
t] + 315 294 360 736 132 054 080 MM[6, 2][t] + 214 182 720 913 855 222 944
MM[7, 2][t] + 11 565 619 032 680 457 669 505 MM[8, 2][t])) /

```
      2 904 853 152 394 161 812 061 792 - (79 812 500 MM[9, 2][t])/1 356 033),
MM[10, 2]'[t] == 1/(50 × 5^(3/4)) 7 ((15 625 (6 066 698 389 336 609 868 160 MM[1, 2][t] +
            6 693 210 028 557 975 717 000 MM[2, 2][t] +
            10 073 281 092 979 753 454 085 MM[3, 2][t] + 319 395 285 573 293 160 000
             MM[4, 2][t] + 2 677 284 011 423 190 286 800 MM[5, 2][t] +
            315 294 360 736 132 054 080 MM[6, 2][t] + 214 182 720 913 855 222 944
             MM[7, 2][t] + 11 565 619 032 680 457 669 505 MM[8, 2][t])) /
      2 904 853 152 394 161 812 061 792 - (39 906 250 MM[10, 2][t])/1 400 301),
Stomach'[t] == 60 If[t ≤ 1/60, 1, 0] - Stomach[
   t], Intest'[
  t] ==
  -(7 Intest[t])/12 +
   Stomach[t]/2,
StomachUnAbs'[
  t] ==
  -StomachUnAbs[t]/2,
IntestUnAbs'[
  t] ==
  -IntestUnAbs[t]/12 +
   StomachUnAbs[t]/2,
FeCar'[
  t] ==
  (7 AlbX[t])/800 +
   1/12
    (Intest[t] +
      IntestUnAbs[t]) +
   1/40 CEX[2, 1][t] + 1/20 CEX[2, 2][t] +
   (7 CEX[2, 3][t])/1000 +
   1/40
    CEX[
      5,
      1][t] +
```

```
 1                     1627 CEX[5, 3][t]
 -- CEX[5, 2][t] + ----------------- +
 20                       250 000
 1
 --
 40
  CEX[
    8,
    1][t] +
 1                     7 CEX[8, 3][t]
 -- CEX[8, 2][t] + -------------- +
 20                       1000
 1
 --
 50
  MM[
    2,
    5][t] +
 1
 -- MM[3, 5][t], U[1]′[t] == MM[
 50
  3,
  1][
 t],
U[2]′[
 t] ==
 MM[
  3,
  2][
 t],
U[3]′[
 t] ==
 MM[
  3,
  3][
 t],
U[4]′[
 t] ==
 MM[
  3,
  4][
 t],
U[5]′[
 t] ==
 MM[
  3,
  5][
 t],
AlbX′[
 t] ==
```

```
  - 7 AlbX[t] / 800 +
    MM[10, 2][t] / 1000 ,
CEX[1, 1]′[
   t] ==
  0,
CEX[1, 2]′[
   t] ==
  0,
CEX[1, 3]′[
   t] ==
  0,
CEX[2, 1]′[
   t] ==
  - 11 / 250
    CEX[2, 1][
      t] + 27/4 CE[2, 1][
      t]
    MM[2, 2][t], CEX[2, 2]′[t] == - 4/25
    CEX[2, 2][
      t] + 3/5 CE[2, 2][
      t]
    MM[2, 2][t], CEX[2, 3]′[t] == - 3/500
    CEX[2, 3][
      t] + 5/8 CE[2, 3][
      t] MM[2, 2][
      t],
CEX[3, 1]′[t] == 0, CEX[3, 2]′[
   t] ==
  0,
CEX[3, 3]′[
   t] ==
  0,
CEX[4, 1]′[
   t] ==
  0,
CEX[4, 2]′[
   t] ==
```

```
 0,
CEX[4, 3]′[
  t] ==
 0,
CEX[5, 1]′[
  t] ==
 - 11/250
   CEX[5, 1][
    t] + 45/2 CE[5, 1][
    t]
   MM[5, 2][t], CEX[5, 2]′[t] == - 4/25
   CEX[5, 2][
    t] + 2 CE[5, 2][
    t] MM[5, 2][
    t],
CEX[5, 3]′[t] == - (2877 CEX[5, 3][t])/500 000 +
  25/12
   CE[5, 3][
    t] MM[5, 2][
    t],
CEX[6, 1]′[t] == 0, CEX[6, 2]′[
  t] ==
 0,
CEX[6, 3]′[
  t] ==
 0,
CEX[7, 1]′[
  t] ==
 0,
CEX[7, 2]′[
  t] ==
 0,
CEX[7, 3]′[
  t] ==
 0,
CEX[8, 1]′[
  t] ==
 - 11/250
   CEX[8, 1][
```

```
    t] + 54 CE[8, 1][
    t] MM[8, 2][
    t],
CEX[8, 2]′[t] == -4/25 CEX[8, 2][t] + 24/5 CE[8, 2][
    t] MM[8, 2][
    t],
CEX[8, 3]′[t] == -3/500 CEX[8, 3][t] + 5 CE[8, 3][
    t]
   MM[8, 2][t], CEX[9, 1]′[t] == -11/250
   CEX[9, 1][
    t] +
  (11 493 CE[9, 1][t] MM[9, 2][t])/4166,
CEX[9, 2]′[
  t] ==
 0,
CEX[9, 3]′[
  t] ==
 0,
CEX[10, 1]′[
  t] ==
 -11/250
   CEX[10, 1][
    t] + 1277/956 CE[10, 1][
    t]
   MM[10, 2][t], CEX[10, 2]′[t] == -4/25
   CEX[10, 2][
    t] +
  (1277 CE[10, 2][t] MM[10, 2][t])/10 755,
CEX[10, 3]′[
  t] ==
 -3/500
   CEX[10, 3][
    t] +
  (6385 CE[10, 3][t] MM[10, 2][t])/51 624,
CE[1, 1]′[
```

t] ==
0,
CE[1, 2]′[
t] ==
0,
CE[1, 3]′[
t] ==
0,
CE[2, 1]′[
t] ==

$$\frac{3\left(\frac{17}{2437500} - \mathrm{CE}[2, 1][t]\right)}{1000} +$$

$$\frac{19\,\mathrm{CEX}[2, 1][t]}{1000} -$$

$$\frac{27}{4}$$

CE[2, 1][
t] MM[2, 2][
t],

$$\mathrm{CE}[2, 2]'[t] == \frac{1}{100}\left(\frac{1}{15250} - \mathrm{CE}[2, 2][t]\right) +$$

$$\frac{3}{100}$$

CEX[2, 2][
t] -

$$\frac{3}{5}\,\mathrm{CE}[2, 2][t]\,\mathrm{MM}[2, 2][t],\ \mathrm{CE}[2, 3]'[t] ==$$

$$\frac{\frac{388}{2715} - \mathrm{CE}[2, 3][t]}{1000} +$$

$$\frac{1}{200}$$

CEX[2, 3][
t] -

$$\frac{5}{8}\,\mathrm{CE}[2, 3][t]\,\mathrm{MM}[2, 2][t],\ \mathrm{CE}[3, 1]'[t] ==$$

0,
CE[3, 2]′[
t] ==
0,
CE[3, 3]′[
t] ==
0,
CE[4, 1]′[
t] ==

```
 0,
CE[4, 2]′[
  t] ==
 0,
CE[4, 3]′[
  t] ==
 0,
CE[5, 1]′[
  t] ==
```

$$\frac{3\left(\frac{11}{121875} - CE[5,1][t]\right)}{1000} + \frac{19\,CEX[5,1][t]}{1000} - \frac{45}{2}\,CE[5,1][t]\,MM[5,2][t],$$

$$CE[5,2]'[t] == \frac{1}{100}\left(\frac{117}{3812500} - CE[5,2][t]\right) + \frac{3}{100}\,CEX[5,2][t] - 2\,CE[5,2][t]\,MM[5,2][t],\ CE[5,3]'[t] ==$$

$$\frac{377\left(\frac{144}{22625} - CE[5,3][t]\right)}{500000} + \frac{1}{200}\,CEX[5,3][t] - \frac{25}{12}\,CE[5,3][t]\,MM[5,2][t],\ CE[6,1]'[t] ==$$

```
 0,
CE[6, 2]′[
  t] ==
 0,
CE[6, 3]′[
  t] ==
 0,
CE[7, 1]′[
  t] ==
```

```
 0,
CE[7, 2]′[
  t] ==
 0,
CE[7, 3]′[
  t] ==
 0,
CE[8, 1]′[
  t] ==
```

$$\frac{3\left(\frac{19}{9750000}-CE[8,1][t]\right)}{1000}+\frac{19\,CEX[8,1][t]}{1000}-54\,CE[8,1][t]\,MM[8,2][t],$$

$$CE[8,2]'[t]==\frac{1}{100}\left(\frac{9}{381250}-CE[8,2][t]\right)+\frac{3}{100}CEX[8,2][t]-\frac{24}{5}CE[8,2][t]\,MM[8,2][t],\ CE[8,3]'[t]==$$

$$\frac{\frac{7}{543}-CE[8,3][t]}{1000}+\frac{1}{200}CEX[8,3][t]-5\,CE[8,3][t]\,MM[8,2][t],\ CE[9,1]'[t]==$$

$$\frac{3\left(\frac{235379}{4150250000}-CE[9,1][t]\right)}{1000}+\frac{19\,CEX[9,1][t]}{1000}-\frac{11493\,CE[9,1][t]\,MM[9,2][t]}{4166},$$

```
CE[9, 2]′[
  t] ==
 0,
CE[9, 3]′[
```

t] ==

0,

CE[10, 1]′[

t] ==

$$\frac{3\left(\frac{167061}{2075125000} - \text{CE}[10, 1][t]\right)}{1000} +$$

$$\frac{19\,\text{CEX}[10, 1][t]}{1000} -$$

$$\frac{1277}{956}$$

CE[10, 1][

t] MM[10, 2][

t],

$$\text{CE}[10, 2]'[t] == \frac{1}{100}\left(\frac{337707}{3894850000} - \text{CE}[10, 2][t]\right) +$$

$$\frac{3}{100}$$

CEX[10, 2][

t] −

$$\frac{1277\,\text{CE}[10, 2][t]\,\text{MM}[10, 2][t]}{10755},$$

CE[10, 3]′[

t] ==

$$\frac{\frac{180684}{5778425} - \text{CE}[10, 3][t]}{1000} +$$

$$\frac{1}{200}$$

CEX[10, 3][

t] −

$$\frac{6385\,\text{CE}[10, 3][t]\,\text{MM}[10, 2][t]}{51624}\}$$

eq0

{MM[1, 1][0] == 0, MM[1, 2][0] == 0, MM[1, 3][0] == 0, MM[1, 4][0] == 0, MM[1, 5][0] == 0,
MM[2, 1][0] == 0, MM[2, 2][0] == 0, MM[2, 3][0] == 0, MM[2, 4][0] == 0, MM[2, 5][0] == 0,
MM[3, 1][0] == 0, MM[3, 2][0] == 0, MM[3, 3][0] == 0, MM[3, 4][0] == 0, MM[3, 5][0] == 0,
MM[4, 1][0] == 0, MM[4, 2][0] == 0, MM[4, 3][0] == 0, MM[4, 4][0] == 0, MM[4, 5][0] == 0,
MM[5, 1][0] == 0, MM[5, 2][0] == 0, MM[5, 3][0] == 0, MM[5, 4][0] == 0, MM[5, 5][0] == 0,
MM[6, 1][0] == 0, MM[6, 2][0] == 0, MM[6, 3][0] == 0, MM[6, 4][0] == 0, MM[6, 5][0] == 0,
MM[7, 1][0] == 0, MM[7, 2][0] == 0, MM[7, 3][0] == 0, MM[7, 4][0] == 0, MM[7, 5][0] == 0,
MM[8, 1][0] == 0, MM[8, 2][0] == 0, MM[8, 3][0] == 0, MM[8, 4][0] == 0, MM[8, 5][0] == 0,
MM[9, 2][0] == 0, MM[10, 2][0] == 0, Stomach[0] == 0, Intest[0] == 0, StomachUnAbs[0] == 0,
IntestUnAbs[0] == 0, FeCar[0] == 0, U[1][0] == 0, U[2][0] == 0, U[3][0] == 0,
U[4][0] == 0, U[5][0] == 0, AlbX[0] == 0, CEX[1, 1][0] == 0, CEX[1, 2][0] == 0,
CEX[1, 3][0] == 0, CEX[2, 1][0] == 0, CEX[2, 2][0] == 0, CEX[2, 3][0] == 0,
CEX[3, 1][0] == 0, CEX[3, 2][0] == 0, CEX[3, 3][0] == 0, CEX[4, 1][0] == 0,
CEX[4, 2][0] == 0, CEX[4, 3][0] == 0, CEX[5, 1][0] == 0, CEX[5, 2][0] == 0,
CEX[5, 3][0] == 0, CEX[6, 1][0] == 0, CEX[6, 2][0] == 0, CEX[6, 3][0] == 0,
CEX[7, 1][0] == 0, CEX[7, 2][0] == 0, CEX[7, 3][0] == 0, CEX[8, 1][0] == 0,
CEX[8, 2][0] == 0, CEX[8, 3][0] == 0, CEX[9, 1][0] == 0, CEX[9, 2][0] == 0,
CEX[9, 3][0] == 0, CEX[10, 1][0] == 0, CEX[10, 2][0] == 0, CEX[10, 3][0] == 0,
CE[1, 1][0] == 0, CE[1, 2][0] == 0, CE[1, 3][0] == 0, CE[2, 1][0] == $\frac{17}{2\,437\,500}$,
CE[2, 2][0] == $\frac{1}{15\,250}$, CE[2, 3][0] == $\frac{388}{2715}$, CE[3, 1][0] == 0, CE[3, 2][0] == 0,
CE[3, 3][0] == 0, CE[4, 1][0] == 0, CE[4, 2][0] == 0, CE[4, 3][0] == 0,
CE[5, 1][0] == $\frac{11}{121\,875}$, CE[5, 2][0] == $\frac{117}{3\,812\,500}$, CE[5, 3][0] == $\frac{144}{22\,625}$,
CE[6, 1][0] == 0, CE[6, 2][0] == 0, CE[6, 3][0] == 0, CE[7, 1][0] == 0,
CE[7, 2][0] == 0, CE[7, 3][0] == 0, CE[8, 1][0] == $\frac{19}{9\,750\,000}$, CE[8, 2][0] == $\frac{9}{381\,250}$,
CE[8, 3][0] == $\frac{7}{543}$, CE[9, 1][0] == $\frac{235\,379}{4\,150\,250\,000}$, CE[9, 2][0] == 0, CE[9, 3][0] == 0,
CE[10, 1][0] == $\frac{167\,061}{2\,075\,125\,000}$, CE[10, 2][0] == $\frac{337\,707}{3\,894\,850\,000}$, CE[10, 3][0] == $\frac{180\,684}{5\,778\,425}$}

The new BW value resets Dimethoate BW-related input parameter values, as illustrated below:

```
TBL[MetTableScaled]
```

Index	Tissue	V12	Km12	V13	Km13	V24	Km24
1	Fat	0	1	0	1	0	1
2	Liver	$\frac{506}{25}$	535	$\frac{107}{125}$	155	8	155
3	Rapid	0	1	0	1	0	1
4	Slow	0	1	0	1	0	1
5	Brain	$\frac{759}{6250}$	$\frac{107}{10}$	$\frac{321}{62\,500}$	$\frac{31}{10}$	$\frac{6}{125}$	$\frac{31}{10}$
6	Skin	$\frac{49\,841}{25\,000}$	$\frac{107}{10}$	$\frac{21\,079}{250\,000}$	$\frac{31}{10}$	$\frac{197}{250}$	$\frac{31}{10}$
7	Diaphr	$\frac{759}{2500}$	$\frac{107}{10}$	$\frac{321}{25\,000}$	$\frac{31}{10}$	$\frac{3}{25}$	$\frac{31}{10}$
8	Lung	$\frac{253}{100}$	535	$\frac{107}{1000}$	155	1	155
9	RBC	0	1	0	1	0	1
10	Plasma	0	1	0	1	0	1

```
TBL[ParTableScaled]
```

Index	Tissue	Vf	Qf	Pdim	Pome
1	Fat	$\frac{7}{500}$	$\frac{463\,605\,843\,994\,279}{36\,794\,114\,602\,720\,560 \times 5^{3/4}}$	$\frac{58}{125}$	$\frac{197}{1000}$
2	Liver	$\frac{1}{125}$	$\frac{463\,605\,843\,994\,279 \times 5^{1/4}}{66\,229\,406\,284\,897\,008}$	$\frac{21}{20}$	$\frac{217}{250}$
3	Rapid	$\frac{1}{125}$	$\frac{139\,545\,359\,042\,277\,979}{2\,649\,176\,251\,395\,880\,320 \times 5^{3/4}}$	$\frac{5}{4}$	$\frac{217}{250}$
4	Slow	$\frac{57}{625}$	$\frac{1\,167\,599\,903\,393 \times 5^{1/4}}{306\,617\,621\,689\,338}$	$\frac{189}{200}$	$\frac{217}{250}$
5	Brain	$\frac{3}{1250}$	$\frac{463\,605\,843\,994\,279}{110\,382\,343\,808\,161\,680 \times 5^{3/4}}$	$\frac{5}{4}$	$\frac{217}{250}$
6	Skin	$\frac{197}{5000}$	$\frac{13\,444\,569\,475\,834\,091}{1\,655\,735\,157\,122\,425\,200 \times 5^{3/4}}$	$\frac{53}{50}$	$\frac{217}{250}$
7	Diaphr	$\frac{3}{500}$	$\frac{463\,605\,843\,994\,279}{551\,911\,719\,040\,808\,400 \times 5^{3/4}}$	$\frac{189}{200}$	$\frac{217}{250}$
8	Lung	$\frac{1}{1000}$	$\frac{19\,935\,051\,291\,753\,997}{2\,649\,176\,251\,395\,880\,320 \times 5^{3/4}}$	$\frac{437}{500}$	$\frac{108}{125}$
9	RBC	$\frac{69}{12\,500}$	$\frac{161}{2500 \times 5^{3/4}}$	1	$\frac{217}{250}$
10	Plasma	$\frac{81}{12\,500}$	$\frac{189}{2500 \times 5^{3/4}}$	1	$\frac{217}{250}$

Run DimethoateModel after adjusting some default inputs

The DimethoateModel Adjust option is used below to adjust default model inputs before running an illustrative application involving exposure to Omethoate. Sets of specified parameter-type-specific adjustment factors must first all be defined for the Adjust option to use (see the DimethoateModel usage statement), a body weight is specified (in this case, 0.200 kg), an administered oral OME dose is specified (here ~2.617 micromoles OME per kg body weight), and then the DimethoateModel function is evaluated first without and then with using the Adjust option. The output list of rules DimethoateModels generates in each case is here set equal to a variable called "sol" (for "solution", but you may name this anything), and the sol = DimethoateModel[...] expression is terminated by a semicolon to suppress the generation of (what would be a very lengthy) Output cell listing the set of all resulting 113 rules. However, one of these symbolically expressed rules (the one pertaining to MM[2,2][t], the mass of OME in liver) is shown just to illutrate what each one looks like, and to illustrate the effect of not using vs. using the specified parameter adjustments, at time t = 10 hours after oral administration.

```
adjNewD = {{Rate, {"Kstom2liv", "Kstom2intes", "Kintes2liv",
      "Kurx[1]", "Kurx[2]", "Kurx[3]", "Kurx[4]",  "Kurx[5]", "FKfecar"},
     {0.04302287018713838`, 6.447287881297196`, 24.892688718635217`,
      0.05, 0.743263161909309`, 0.5,
      0.8729947391421071`, 0.48720983462787804`, 0.48720983462787804`}},
    {Met, {{V12, 2}, {V13, 2}, {V24, 2}}, {1, 22.5, 0.19014207349320744`}},
    {PPset, {{1, 4}}, {1.781863130336682`}},
    {Inhib, {{AChE, 2}, {AChE, 5}, {AChE, 8}, {AChE, 9}, {AChE, 10}},
     {0.8503721520203437`, 0.8503721520203437`, 0.8503721520203437`,
      0.8503721520203437`, 0.8503721520203437`}}};
```

```
BW = 0.2;
umol = 1000. (dose = 3 * BW / mwD)
```

```
2.61712
```

Here DimethoateModel is evaluated WITHOUT an Adjustment option:

```
sol = DimethoateModel[0.2, 3 * BW, 0, 0, 72, Scenario → Oral];
sol[[7]]
```

```
MM[2, 2][t] → InterpolatingFunction[ Domain: {{0., 72.}} Output: scalar ][t]
```

```
{MM[2, 2][t], MM[2, 4][t], Blood[#][t] & /@ {1, 2, 3, 4, 5}} /. sol /. t → 10
```

```
{0.00113205, 0.015101, {0.302904, 0.268548, 0.829888, 2.06549, 4.91361}}
```

Here DimethoateModel is evaluated WITH an Adjustment option:

```
sol = DimethoateModel[0.2, 3 * BW, 0, 0, 72, Scenario → Oral, Adjust → adjNewD];
sol[[7]]
```

```
MM[2, 2][t] → InterpolatingFunction[ Domain: {{0., 72.}} Output: scalar ][t]
```

```
{MM[2, 2][t], MM[2, 4][t], Blood[#][t] & /@ {1, 2, 3, 4, 5}} /. sol /. t → 10
```

```
{0.00141541, 0.00866648, {0.117778, 0.344775, 0.948421, 1.12789, 8.0156}}
```

Using the latter reults, obtained using DimethoateModel WITH an Adjustment option, the code below summarizes compartment values in terms of percent of applied (micromolar OME) dose for the total of all DIM + metabolites present in those compartments at various times (in hours) listed in the left-most column. Four subsets of all the modeled mass-conserving compartments are evaluated, together with the sum of tehse four subsets, and the same sum evaluated using the DimethoateModel-generated SUM function. Note that by 72 hours, the totals (shown in the last two columns on the right) are very slightly greater than 100%. This indicates that the current version of the Dimenthoate model has a very subtle mass-balance flaw (i.e., flaw in the logic of the ODE structure), the underlying reason for which has not yeat been identified. However, runs using various model inputs have shown that the relative magnitude of this problem remains very small (<1%) even over large durations of (e,.g., repeated daily) simulated exposure. When using Dimethoate-Model, it therefore is important to verify that this remains the case for each model run, until this problem is corrected.

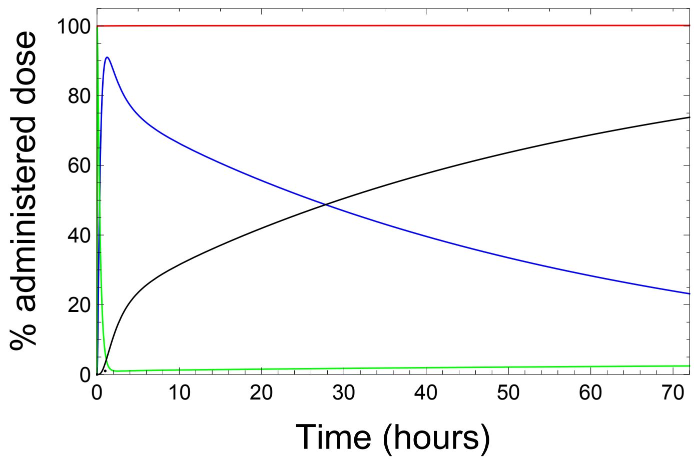
```
items[T_] := (100 / umol) {
      aa = Plus @@ Flatten[Table[MM[i, j][t], {i, 8}, {j, 5}]], bb =
        Stomach[t] + StomachUnAbs[t] + Intest[t] + IntestUnAbs[t] + FeCar[t] + AlbX[t],
      cc = Plus @@ (U[#][t] & /@ Range[5]),
      dd = Plus @@ Flatten[Table[CEX[i, j][t], {i, 8}, {j, 3}]],
      sum = aa + bb + cc + dd,
      SUM[t]} //. sol /. t → T;
```

```
tt = {0, 1 / 60, 1, 24, 48, 72};
TBL[tr = Transpose[Prepend[Transpose[items[t] /. t → # & /@ tt], tt]]]
```

0	0.	0.	0.	0.	0.	0.
$\frac{1}{60}$	0.191668	99.8083	1.13632×10^{-7}	1.13984×10^{-6}	100.	100.
1	90.0562	6.07658	3.26181	0.617803	100.012	100.012
24	51.9544	1.60613	45.5538	0.977316	100.092	100.092
48	34.6179	2.07558	62.5487	0.877444	100.12	100.12
72	23.1586	2.43393	73.7787	0.769983	100.141	100.141

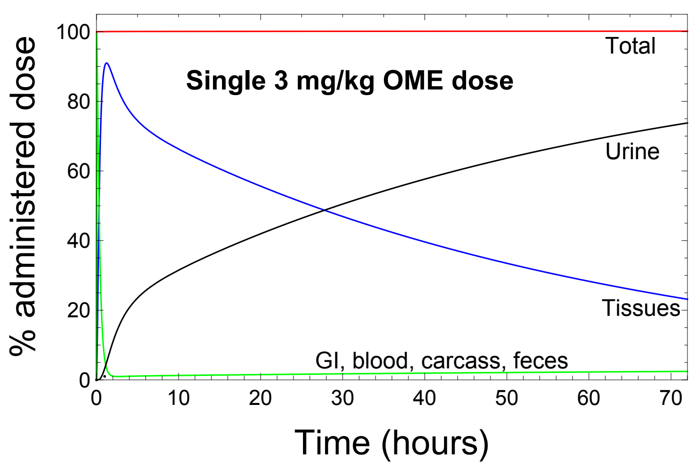
The *RiskQ* function PlotData can then be used to plot various model outputs, e.g., in relation to time t.

```
PlotData[Plot, FitTo → {items[t][[{5, 1, 2, 3}]], t}, X → {0, 72},
 Y → {0, 105}, Labels → {"Time (hours)", "% administered dose"},
 PlotStyle → {Red, Blue, Green, Black}, FontSize → 18]
```



The *Mathmatica* Drawing Tools tool (available under the Graphics tab at the top of your screen when running *Mathmatica*) can be used to annotate any plot like that above, to label, group, arrange, align, distribute, fill, color, alter the opacity or stroke or arrow heads of, or add text to, and/or otherwise modify objects that appear within the plot. Items within a plot (such as added text) can also be copied and pasted and then modified within that plot, by first double-clicking on the plot (to make it editable--after which a gray box will surround the plot), using copy and paste keys as in any Word Processing or graphics-editing application; use Shift + Click to select or paste multiple plot objects.

To do any of this without needing to replot the underlying plot in case using the Drawing Tools results in an error that irreversibly, it is strongly recommended that you highlight (by clicking on) the source plot (or on its Output cell right side bracket), copy it, and then paste it into a new Output cell that will appear when you use your Paste button. Then edit your new copy of your source plot. New evaluations of the PlotData expression used to generate the source plot will not add new objects created using Drawing Tools; rather, Drawing Tools must be applied to (preferably a copy of) each new plot generated in order to annotate that new plot.



Additional DimethoateModel applications & plots

The first illustration below involves parameter adjustment to show the effect of using optimized DimethoateModel values for OME, assuming a single oral OME dose of first 0.5 and then 10 mg/kg. Before doing this, OME plasma measures and standard error (SE) data are defined (by evaluating the closed data-definition cell), so that those data (which were obtained at both intake doses combined) may be plotted together with ExDM model predictions, which include corresponding predictions of OME in plasma (in μmol/L) evaluated using a metabolite-specific “plasma“ function newly defined below.

The second illustration involves parameter adjustment to show the effect of using optimized DimethoateModel values for DIM, assuming a single oral DIM dose of first 10 and then 100 mg/kg.

Define OME plasma and urinary data

```
xyP = {{{0.0833`, 0.06589705842185338`}, {0.1667`, 0.28829045197736447`},
    {0.3333`, 0.6482390986351865`}, {0.6667`, 0.9804068912283939`},
    {1, 1.0180762802225924`}, {1.5`, 0.8864086338975363`}, {2, 0.7218926049002709`},
    {3, 0.550044525108011`}, {4, 0.3704527820035404`}, {6, 0.17696014676065688`},
    {8, 0.09465176456512947`}, {24, 0.005440018362626245`},
    {32, 0.0038099511931019123`}, {48, 0.0020838013121856833`}},
   {{0.0833`, 0.2371029415781466`}, {0.1667`, 0.6178095480226355`},
    {0.3333`, 0.9845709013648134`}, {0.6667`, 1.220703108771606`},
    {1, 1.2861237197774074`}, {1.5`, 1.0973113661024638`}, {2, 0.9534273950997293`},
    {3, 0.714000474891989`}, {4, 0.5611772179964595`}, {6, 0.3384898532393431`},
    {8, 0.19013823543487052`}, {24, 0.011739981637373376`},
    {32, 0.00906504880689809`}, {48, 0.004931198687814317`}},
   {{0.0833`, 0.1515`}, {0.1667`, 0.45305`}, {0.3333`, 0.8164049999999999`},
    {0.6667`, 1.100555`}, {1, 1.1521`}, {1.5`, 0.9918600000000001`},
    {2, 0.8376600000000001`}, {3, 0.6320225`}, {4, 0.465815`},
    {6, 0.257725`}, {8, 0.142395`}, {24, 0.00859`},
    {32, 0.006437500000000005`}, {48, 0.0035075`}}};
XYPdata = Last[xy];
vXYPdata = {0.08560294157814662`, 0.16475954802263557`,
   0.1681659013648134`, 0.12014810877160603`, 0.13402371977740743`,
   0.1054513661024637`, 0.1157673950997292`, 0.08197797489198902`,
   0.09536221799645954`, 0.0807648532393431`, 0.047743235434870517`,
   0.0031499816373733763`, 0.0026275488068980882`, 0.001423698687814317`};

xyi = xyu =
   {{{2, 19.649432761756326`}, {4, 41.83252072076206`}, {6, 59.905484787038944`},
     {8, 70.32838997880681`}, {24, 85.21279999869634`}, {48, 86.88479962467622`}},
    {{2, 32.821567238243674`}, {4, 63.11147927923794`}, {6, 79.25851521296104`},
     {8, 87.05861002119319`}, {24, 96.07970000130365`}, {48, 96.78070037532376`}},
    {{2, 26.235500000000002`}, {4, 52.472`}, {6, 69.582`},
     {8, 78.6935`}, {24, 90.64625`}, {48, 91.83274999999999`}}};
Udata = Last[xyu];
vUdata = {6.586067238243675`, 10.639479279237936`, 9.676515212961052`,
    8.365110021193185`, 5.43345000130366`, 4.947950375323778`}^2;
```

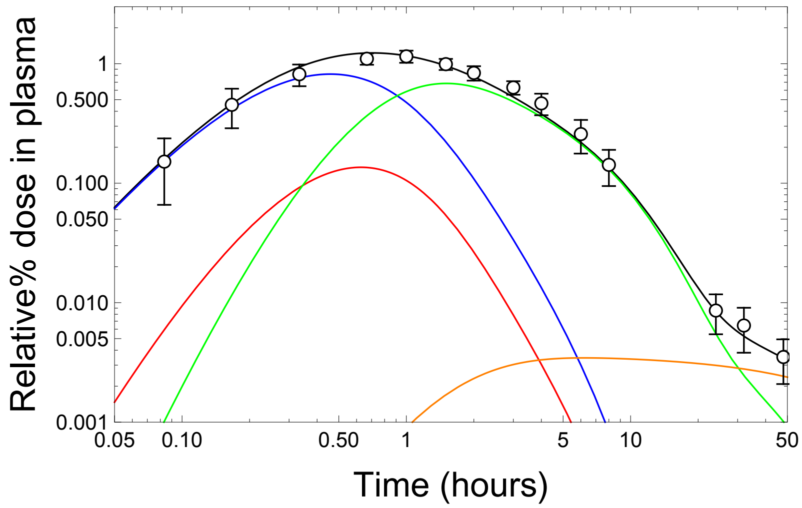
Last: Nonatomic expression expected at position 1 in Last[xy].

Continue with OME illustration

```
plasma[T_, umol_, sol_, i_: {1, 2, 3, 4, 5}] :=
 (100 * Fplasma / umol) Append[(Blood[#][t] & /@ i) vBlood, AlbX[t]] /. sol /. t → T

adj = {{Rate, {"Kstom2liv", "Kstom2intes", "Kintes2liv",
      "Kurx[2]", "Kurx[4]", "Kurx[5]",
      "FKfecar", "Fplasma",   "K35", "K45",    "Kalb", "KalbX", "Malb"},
     {0.25,  4,  22,
      1,  1 ,  0.35,
      3,   0.035,            0,  25,          1, 1, 1.4}},
    {Met, {{V24, 2}}, {0.475}},
    {Inhib, {AChE, #} & /@ {2, 5, 8, 9, 10}, 1.2065 {1, 1, 1, 1, 1}},
    {React, {AChE, #} & /@ {2, 5, 8, 9, 10}, (0.0088 / 0.019) {1, 1, 1, 1, 1}},
    {PPset, Join[ {#, 2} & /@ {2, 3, 4, 6},
      {3, #} & /@ {4, 5},  {#, 4} & /@ {4},  {#, 5} & /@ {4, 6}],
       Join[{0.5,  30, 0.001, 0.001}, {30, 5},  Table[0.01, {2}], Table[0.01, {1}] ] }
  };

{BW = 0.2, umol = 1000. (dose = 0.5 * BW / mwO)};
sol = DimethoateModel[BW, 0.5 * BW, 0, 0, 50, Scenario → Oral,
  Chemical → Omethoate, Adjust → adj, Precision → 22, Goal → 7];
{PCE[5, 1][t] /. sol /. t → 2.5, utot = 1. Plus @@ (U[#][t] & /@ Range[5]) /. sol /. t → 48;
 U[2][t] / utot /. sol /. t → 48, 100 * utot / umol, 100 SUM[t] //. sol / umol /. t → # & /@ {1 / 60, 1, 2, 50}}
{73.1352, 0.33968, 91.7666, {100., 100.039, 100.079, 100.198}}
```

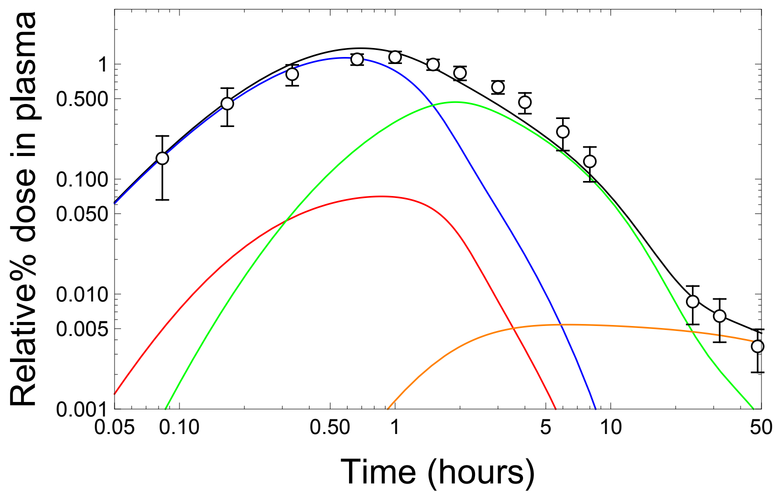
After the PlotData evaluation below, the plot was clicked and its corner dragged out a bit to enlarge the plot.

```
PlotData[xyP, Style → {M, J, M, OO}, X → {0.05, 50}, Y → {0.001, 3}, FitTo →
  {Join[{Plus @@ plasma[t, umol, sol]}, plasma[t, umol, sol][[{2, 4, 5, 6}]]], t},
 Labels → {"Time (hours)", "Relative% dose in plasma"},
 PlotStyle → {Black, Blue, Red, Green, Orange}, PlotType → LogLog]
```



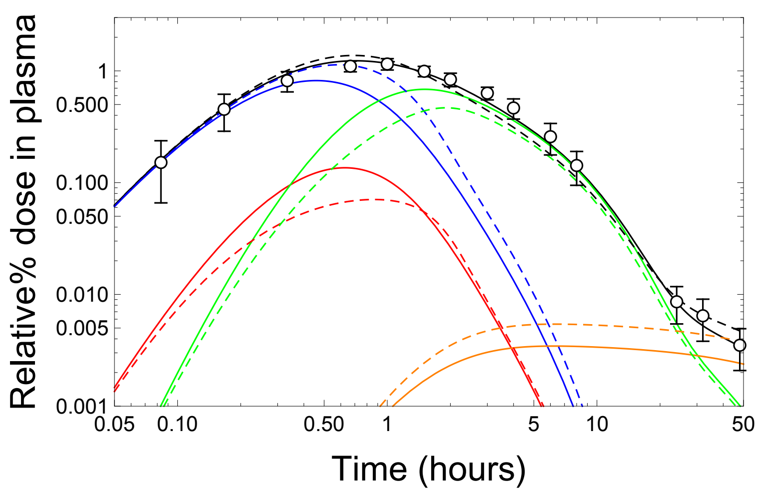

```
{BW = 0.2, umol2 = 1000. (dose = 10 * BW / mwO)};
sol2 = DimethoateModel[BW,    10 * BW, 0, 0, 50, Scenario → Oral,
  Chemical → Omethoate, Adjust → adj, Precision → 22, Goal → 7];
{PCE[5, 1][t] /. sol2 /. t → 2.5,
 utot2 = 1. Plus @@ (U[#][t] & /@ Range[5]) /. sol2 /. t → 48;
 U[2][t]/utot2 /. sol2 /. t → 48, 100 * utot2/umol2,
 100 SUM[t] //. sol2/umol2 /. t → # & /@ {1/60, 1, 2, 50}}
{0.265285, 0.522829, 93.6798, {100., 100.05, 100.097, 100.183}}
```

```
PlotData[xyP, Style → {M, J, M, OO}, X → {0.05, 50}, Y → {0.001, 3}, FitTo →
  {Join[{Plus @@ plasma[t, umol2, sol2]}, plasma[t, umol2, sol2][[{2, 4, 5, 6}]]], t},
 Labels → {"Time (hours)", "Relative% dose in plasma"},
 PlotStyle → {Black, Blue, Red, Green, Orange}, PlotType → LogLog]
```

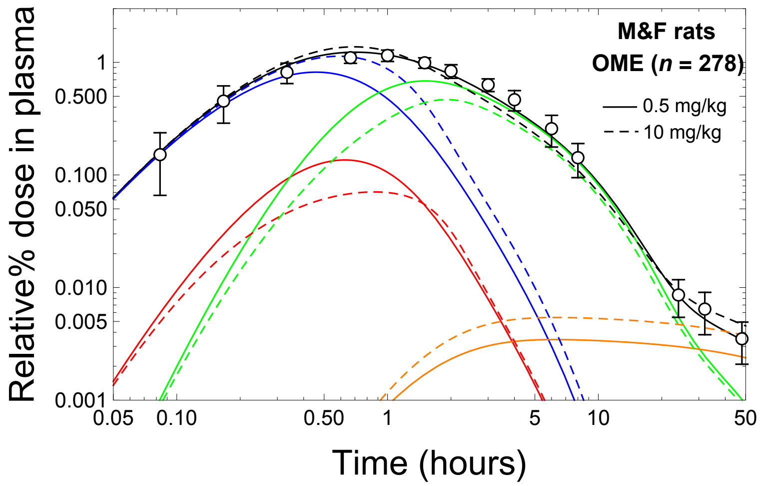


Finally, PlotData is used below to combine outputs from both the 0.5- and 10-mg/kg dosing scenarios ...

```
PlotData[xyP, Style → {M, J, M, OO}, X → {0.05, 50}, Y → {0.001, 3},
 FitTo → {Join[{Plus @@ plasma[t, umol, sol]}, plasma[t, umol, sol][[{2, 4, 5, 6}]],
    {Plus @@ plasma[t, umol2, sol2]}, plasma[t, umol2, sol2][[{2, 4, 5, 6}]]], t},
 Labels → {"Time (hours)", "Relative% dose in plasma"},
 PlotStyle → {Black, Blue, Red, Green, Orange, {Black, Dashed}, {Blue, Dashed},
   {Red, Dashed}, {Green, Dashed}, {Orange, Dashed}}, PlotType → LogLog]
```

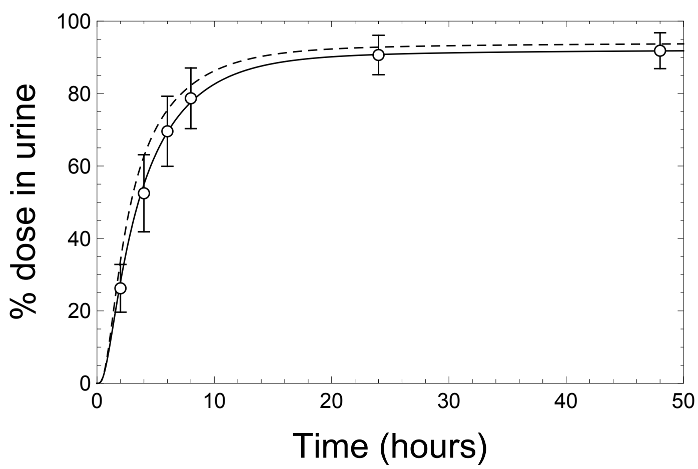


... and then Drawing Tools are used to annotate this final plot



... and PlotData is used to compare OME urinary data with corresponding ExDM model predictions.

```
PlotData[xyi, Style → {M, J, M, OO}, X → {0, 50}, Y → {0, 100},
 FitTo → { 100/{umol, umol2} {Plus @@ (U[#][t] & /@ Range[5]) //. sol,
     Plus @@ (U[#][t] & /@ Range[5]) //. sol2}, t},
 Labels → {"Time (hours)", "% dose in urine"}, PlotStyle → {Black, {Black, Dashed}}]
```



Define DIM plasma and urinary data

```
xy10 = {{{6, 71.39052814649087`}, {12, 86.33255192227215`},
    {24, 89.73220513825032`}, {48, 91.49886680378454`}, {72, 92.80714978596468`},
    {96, 93.51949168045422`}, {120, 93.98597497363033`}},
   {{6, 81.1286581668977`}, {12, 91.54820191911541`}, {24, 94.21481860555252`},
    {48, 96.02906970874518`}, {72, 96.1246702337741`}, {96, 96.21172587003286`},
    {120, 96.21786171268363`}}, {{6, 76.25959315669428`}, {12, 88.94037692069378`},
    {24, 91.97351187190142`}, {48, 93.76396825626486`}, {72, 94.46591000986939`},
    {96, 94.86560877524354`}, {120, 95.10191834315698`}}};
Udata10 = Last[xy10];
```

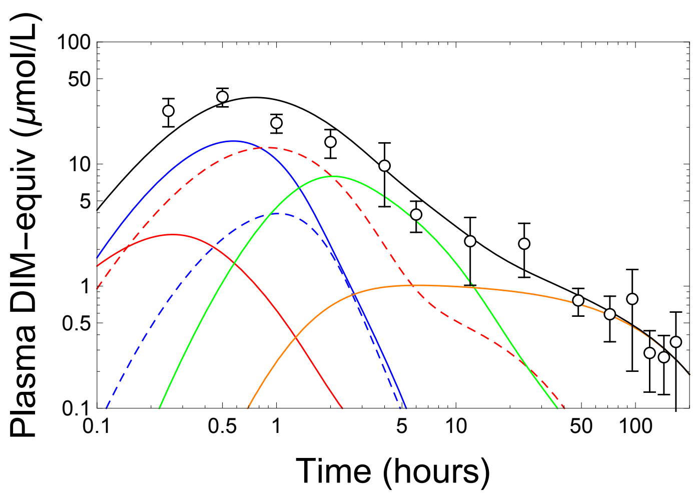
```
vUdata10 =
  {4.869065010203413`, 2.607824998421636`, 2.241306733651104`, 2.265101452480327`,
    1.6587602239047003`, 1.3461170947893255`, 1.115943369526648`}^2;

xy100 = {{{6, 52.48375082613771`}, {12, 77.98455738543913`},
     {24, 89.43882352500917`}, {48, 92.00407799590975`}, {72, 93.09594663666378`},
     {96, 93.49193111432515`}, {120, 93.78578134232318`}},
   {{6, 63.34274799661159`}, {12, 83.00159800897745`}, {24, 92.21956898252641`},
     {48, 94.09701450900339`}, {72, 94.62089253296698`}, {96, 94.79819822242604`},
     {120, 94.9317159075499`}}, {{6, 57.91324941137465`}, {12, 80.49307769720829`},
     {24, 90.82919625376779`}, {48, 93.05054625245657`}, {72, 93.85841958481538`},
     {96, 94.1450646683756`}, {120, 94.35874862493654`}}};
Udata100 = Last[xy100];
vUdata100 =
  {5.429498585236945`, 2.508520311769155`, 1.390372728758624`, 1.0464682565468229`,
    0.7624729481515955`, 0.6531335540504436`, 0.57296728261336`}^2;
xyPlasmaMF = {{{0.5`, 24.862601413242604`}, {2, 3.7948181104422924`},
     {6, 3.2277763238244783`}, {24, 0.10904649742650266`}},
   {{0.5`, 32.27776323824479`}, {2, 6.237459652795952`},
     {6, 8.113059408531798`}, {24, 0.10904649742650266`}}};
xyPlasma10 = {{{0.25`, 20.190996992206586`}, {0.5`, 29.38814878918402`},
     {1, 17.91665693807343`}, {2, 11.16596807339923`}, {4, 4.477453295848179`},
     {6, 2.758930369628143`}, {12, 1.0158368151370654`}, {24, 1.180614128821315`},
     {48, 0.5687454591014969`}, {72, 0.3503406941552417`},
     {96, 0.2012317707168626`}, {120, 0.13560302749879463`},
     {144, 0.12940983260836508`}, {168, 0.08434526933432612`}},
   {{0.25`, 34.33225172104475`}, {0.5`, 41.710167532895724`},
     {1, 25.48384903767463`}, {2, 19.14895821116852`}, {4, 14.889204647098698`},
     {6, 4.961561648168246`}, {12, 3.6513532747172492`}, {24, 3.2684829661799943`},
     {48, 0.9579055048695406`}, {72, 0.8273614780509871`},
     {96, 1.369037792224776`}, {120, 0.4314387591190193`},
     {144, 0.39401335503884777`}, {168, 0.613552314195291`}},
   {{0.25`, 27.261624356625667`}, {0.5`, 35.54915816103987`},
     {1, 21.70025298787403`}, {2, 15.157463142283873`}, {4, 9.683328971473438`},
     {6, 3.8602460088981947`}, {12, 2.3335950449271574`}, {24, 2.2245485475006546`},
     {48, 0.7633254819855188`}, {72, 0.5888510861031144`},
     {96, 0.7851347814708194`}, {120, 0.28352089330890695`},
     {144, 0.2617115938236064`}, {168, 0.3489487917648086`}}};
xyPlasma100 = {{{0.25`, 188.23775776968193`}, {0.5`, 171.3067777656402`},
     {1, 105.21002542018522`}, {2, 55.23439999144827`}, {4, 16.663219428962492`},
     {6, 37.04160593210611`}, {12, 24.938497582846523`},
     {24, 15.568732016920407`}, {48, 8.4679324878851`},
     {72, 5.274535404158911`}, {96, 1.9176116651778`}, {120, 2.199051886275236`},
     {144, 0.559358293446433`}, {168, 1.0747902649362517`}},
```

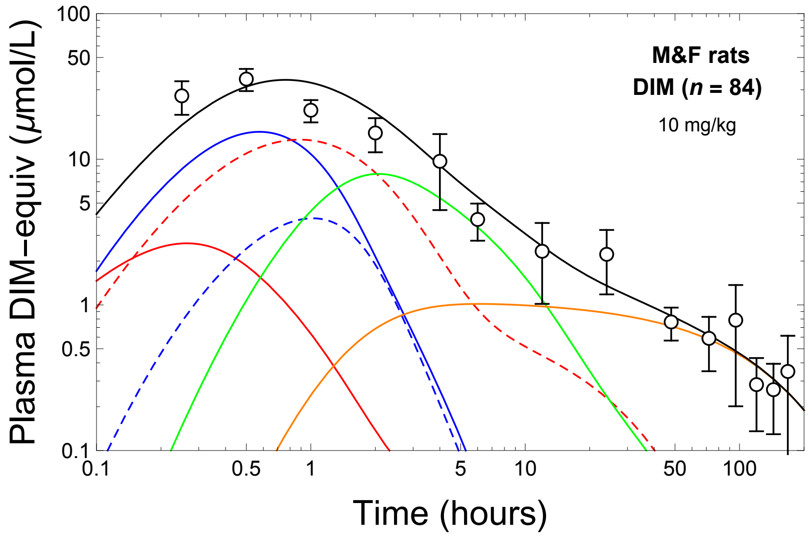
```
{{0.25`, 312.0675724231123`}, {0.5`, 425.3956561521825`},
 {1, 239.81309243726923`}, {2, 96.99451041594946`}, {4, 38.29621527399486`},
 {6, 93.81419097969709`}, {12, 48.64607888055748`}, {24, 30.75422008985793`},
 {48, 21.934230994623846`}, {72, 12.652708772758128`},
 {96, 10.208358848649297`}, {120, 10.406723216228471`},
 {144, 6.027050151114329`}, {168, 3.7668742208004664`}},
{{0.25`, 250.15266509639713`}, {0.5`, 298.3512169589113`},
 {1, 172.51155892872723`}, {2, 76.11445520369887`}, {4, 27.479717351478676`},
 {6, 65.4278984559016`}, {12, 36.792288231702`}, {24, 23.16147605338917`},
 {48, 15.201081741254473`}, {72, 8.96362208845852`},
 {96, 6.0629852569135485`}, {120, 6.302887551251854`},
 {144, 3.293204222280381`}, {168, 2.420832242868359`}}};
```

Continue with DIM illustration

```
adj = {{Rate, {"Kstom2liv", "Kstom2intes", "Kintes2liv",
     "Kurx[2]", "Kurx[4]", "Kurx[5]",      "Kurx[1]", "Kurx[3]",
     "FKfecar", "Fplasma", "K35", "K45",    "Kalb", "KalbX", "Malb"},
    {0.25,  4,  22,
     1,  1 ,  0.35,      10, 10,
     3,   1,            0.0505,  0.26 * 25,          1, 1, 23}},
   {Met, {{V12, 2}, {Km12, 2}, {V13, 2},  {V24, 2}},  {1.488, 0.5, 21.7,   0.475}},
   {Inhib, {AChE, #} & /@ {2, 5, 8, 9, 10}, 1.205 {1, 1, 1, 1, 1}},
   {React, {AChE, #} & /@ {2, 5, 8, 9, 10}, (0.0088 / 0.019) {1, 1, 1, 1, 1}},
   {PPset, Join[  {#, 2} & /@ {2, 3, 4, 6},
     {3, #} & /@ {4, 5},  {#, 4} & /@ {4},  {#, 5} & /@ {4, 6}],
      Join[{0.5,  30, 0.001, 0.001}, {30, 5},  Table[0.01, {2}], Table[0.01, {1}] ] }
  };
{BW = 0.2175, umol = 1000. (dose = 10 * BW / mwD) };
sol = DimethoateModel[BW, 10 * BW, 0, 0, 200, Scenario → Oral,
  Chemical → Dimethoate, Adjust → adj, Precision → 21, Goal → 7];
o = {PCE[5, 1][t] /. sol /. t → 1, uri[T_] := ((U[#][t] & /@ Range[5] /. sol) /. t → T);
  utot = 1. Plus @@ (uu = uri[48]);
  ur = (100 * uu)/utot, uTot = (100. * Plus @@ uri[120])/umol,
  (100 SUM[t] //. sol)/umol /. t → # & /@ {1 / 60, 1, 2, 5, 50, 120}}
{45.344772674873440625, {2.42296, 20.4811, 41.0582, 7.81461, 28.2232},
 94.9417, {100., 100.022, 100.045, 100.064, 100.101, 100.139}}
```

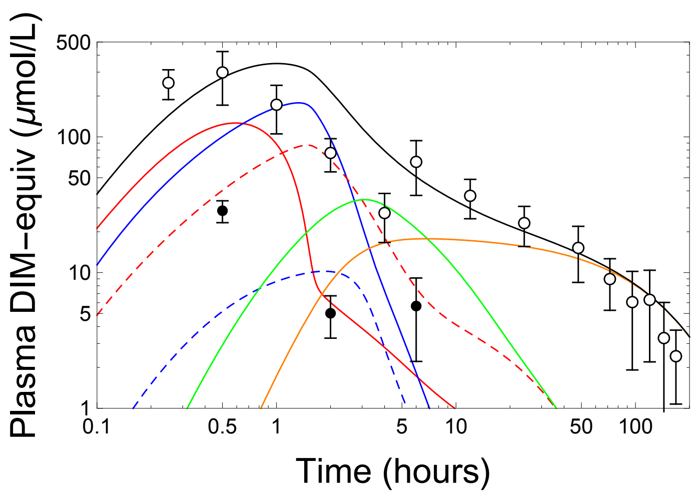
```
{BW = 0.2175, umol2 = 1000. (dose = 100 * BW / mwD)};
sol2 = DimethoateModel[BW, 100 * BW, 0, 0, 200, Scenario → Oral,
   Chemical → Dimethoate, Adjust → adj, Precision → 21, Goal → 7];
o = {PCE[5, 1][t] /. sol2 /. t → 1, uri[T_] := ((U[#][t] & /@ Range[5] /. sol2) /. t → T);
   utot2 = 1. Plus @@ (uu2 = uri[48]);
   ur2 = (100 * uu2) / utot2, uTot2 = (100. * Plus @@ uri[120]) / umol2,
   (100 SUM[t] //. sol2) / umol2 /. t → # & /@ {1/60, 1, 2, 5, 50, 120}}
{0.4355134562821032307, {15.5607, 35.7538, 28.9606, 3.71045, 16.0145},
 95.931, {100., 100.015, 100.022, 100.023, 100.031, 100.039}}

plasmaD[T_, umol_, sol_, i_: {1, 2, 3, 4, 5}] :=
 Append[Blood[#][t] & /@ i, AlbX[t] / vBlood] /. sol /. t → T

PlotData[xyPlasma10, Style → {M, J, M, OO},
 FitTo → {Append[xx = 0.16 * plasmaD[t, umol, sol], Plus @@ xx] //. sol, t},
 Labels → {"Time (hours)", "Plasma DIM-equiv (μmol/L)"},
 X → {0.1, 200}, Y → {0.1, 100}, PlotType → LogLog,
 PlotStyle → {Red, Blue, {Red, Dashed}, {Blue, Dashed}, Green, Orange, Black}]
```

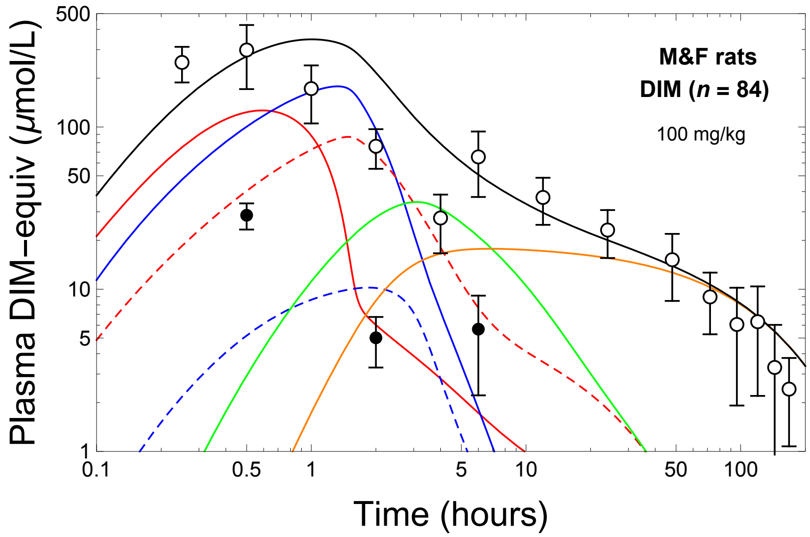





```
tt = Drop[First /@ xyPlasmaMF[[1]], -1];
yyt = Last /@ # & /@ xyPlasmaMF;
yy = Drop[Mean /@ Transpose[yyt], -1];
sy = Drop[StandardDeviation /@ Transpose[yyt], -1];
xyD = Transpose[{tt, #}] & /@ {yy - sy, yy + sy, yy};
```

```
PlotData[Join[xyPlasma100, xyD], Style → {M, J, M, OO, M, J, M, O},
 FitTo → {Append[xx = 0.16 plasmaD[t, umol, sol2], Plus @@ xx], t},
 Labels → {"Time (hours)", "Plasma DIM-equiv (μmol/L)"},
 X → {0.1, 200}, Y → {1, 500}, PlotType → LogLog,
 PlotStyle → {Red, Blue, {Red, Dashed}, {Blue, Dashed}, Green, Orange, Black}]
```





A *Mathematica*® Primer

Ken Bogen, DrPH DABT July 2017

Evaluating expressions typed into Input cells

The Mathematic user interface on your computer is called a “notebook.” When you enter text into a notebook, it becomes contained within a cell in which your cursor is located. You can see the boundary of each cell by looking at the square right-hand bracket at the right edge of the *Mathematica* notebook window. The cell expands downward as you type new text into it. When your notebook is open (which launches the *Mathematica* application), click at the top of your computer screen’s Window tab and then click the Toolbar–Formatting tab. A toggle bar appears at the top of the notebook showing the type of cell in which you are typing and also allowing you to change the cell type. For example, if you click here the cell type shown will be “Text”—but watch what happens if you click one type up and change this cell type instead to be a “Subsubsection” rather than a “Text” cell. Note that the font style changes (enlarges and get colored, a bit like the “Subsection” cell just above this one, but with slightly smaller font size and a bit different color.

You can use cell types to organize and document your entries into and calculations done within a notebook. Note that (by default) cells automatically are grouped hierarchically, as shown by the corresponding grouping of right-hand sqaure brackets (which in this case are shown with a down arrow at the bottom of the enclosing bracket) that appear on the right side of this notebook. By double-clicking on a right square bracket that encloses a group of cells (such as the group of text cells in this subsection), all the cells in that group become hidden; repeating this double-click process will then unhide that group of cells.

When your notebook is an active window (by clicking anywhere in the notebook), you can use the Divide option under the Cell tab at the top of your computer screen (or, on a Mac computer, type Shift + Command + D) to divide any cell into two or more cells at the point where the cursor is, or (after highlighting two vertically adjacent right-hand square brackets at the right side of a notebook---but be sure all those cells are the same type!) use the Merge option (or type Shift + Command + M) to merge those cells into a single cell of that type.

If the cell being typed in is an “Input”-type cell, then the expressions that appear in that cell are all evaluated by pressing Shift+Return, and (if/as applicable) a corresponding “Output” cell appears below the evaluated cell that contains (any) output of that evaluation, or set of evaluations each separated by a semicolon. A semicolon at the end of a line signifies that no output is requested to be returned from expression preceeding the semicolon. The numbered Input and Output definitions all disappear when a notebook is closed. However, simply closing a notebook does NOT erase all the definitions that may have been produced during a *Mathematica* session. To make *Mathematica* forget ALL such definitions, such that they must be re-evaluated (or re-loaded if a “package” is used to autoload a set of definitions, as explained more below) in order to be operable once again, click the Quit Kernel option of the Evaluation tab that appears on the top of the computer screen when the *Mathematica* application is running. If you are running *Mathematica* now, try doing this on this subsection of cells.

If a set of Input cells is grouped (e.g., into a section, subsection, or subsubsection), these Input cells need not be evaluated in order within that group, one cell at a time. Rather, simply click on the right-side down-arrow-bracket that defines the group of cells to highlight this bracket, and then press Shift-Return, to rapidly evaluate all the grouped cells one at a time automatically from top to bottom. The grouped set of cells need not even be visible to do this type of automatic sequential evaluation---that is, all the cells that are being evaluated one at a time need not be looked at and evaluated one at a time, but rather this can happen with all the cells being hidden in the closed group that is evaluated group wise, top to bottom, when the right-hand bracket of the entire group is highlighted and Shift-Enter is then pressed.

Special Symbols, usage notes, and operators

Some special *Mathematica* symbols (each always written with an initial **capital** letter if spelled out, or defined using a special symbol discussed below) are pre-defined such that thar are “Protected” and cannot be re-defined. Examples are E (Euler’s constant, denoted ⅇ which symbol can be accessed by the Basic Math Input palette of the Palettes tab at the top of the screen when running *Mathematica*), Pi (also denoted π), I (capital “i”, the impaginary square root of negative one, also denoted by the special symbol ⅈ), and Infinity (also denoted ∞). Such special symbols are said to have Protected as their Attribute, or as one of their Attributes. Remember, you need to press Shift-Return in order to evaluate the expression(s) in a cell. This has already been done to produce all of the Output cells that appear below in this notebook.

```
E
```

```
ⅇ
```

```
1. * E
```

```
2.71828
```

If a reassignment of the value of a Protected symbol, such as E, is attempted, *Mathematica* will complain and the attempt will fail.

```
E[x_] := x^2
```

SetDelayed::write : Tag ⅇ in ⅇ[x_] is Protected. >>

```
$Failed
```

Placing a question mark before a symbol (with no space) will, when evaluated, generate a “usage” meassage about that symbol. Using two question marks yields additional information about that symbol. Note the >> that appears at the end of each usage message. If you click on that << symbol, a new *Mathematica* window will appear containing complete *Mathematica* documentation for that function.

```
? E
```

E is the exponential constant ⅇ (base of natural logarithms), with numerical value ≃ 2.71828. >>

```
? Attributes
```

Attributes[*symbol*] gives the list of attributes for a symbol. >>

```
?? E
```

E is the exponential constant ⅇ (base of natural logarithms), with numerical value ≃ 2.71828. >>

```
Attributes[ⅇ] = {Constant, Protected, ReadProtected}
```

```
Attributes[E]
```

```
{Constant, Protected, ReadProtected}
```

In *Mathematica*, square **brackets** tha are placed directly after and adjacent to a symbol mean that the symbol represents either an indexed variable, or a function of any argument(s) that appear within the brackets (how to program function definitions is discussed further below). Once a function is defined (as explained below), it can be evaluated to perform its pre-defined operations on its argument(s). Thus directly above, the built-in symbol E is used as an argument of the predefined *Mathematica* function called Attributes. There are many predefined *Mathematica* symbols and functions. All pre-defined *Mathematica* symbols and functions begin with a capital letter. Many special symbols (such as Greek symbols, special mathematical operator symbols, superscript notation, and subscript notation) can be accessed using buttons contained the Basic Math Input palatte that may be accessed by clicking on the Palettes tab that appears at the top of your computer screen when running *Mathematica*.

In *Mathematica*, some special symbols are used as shorthand to denote mathematical operators that always also have longer function names that can be used equivalently without requiring function-type notation. Many of these are the same as in languages such as Fortran, C++, and Excel. Thus, * denotes multiplication (as in 2*2, also written Times[2,2]), the carrot or hat (^) denotes exponentiation (note that the double-asterisk ** is not a synonym for exponentiation in *Mathematica*), + denotes addition (as in 2+2, also written Plus[2,2]), / denotes division (also denoted using the Divide function), etc.

Note that in Mathematica, a space between expressions denotes that the expressions are multiplied together; thus, A B is the same as A*B.

```
2 + 2
```

```
4
```

```
Plus[2, 2]
```

```
4
```

```
Divide[2, 3]
```

$\frac{2}{3}$

$\sqrt{3}$

$\sqrt{3}$

```
Sqrt[3]
```

$\sqrt{3}$

```
Sqrt[3.]
```

```
1.73205
```

Note that *Mathematica* uses infinte-precision integer (or rational number) arithmetic. Here we define (using the = sign) the symbol xx to be a power of a rational number (here created using the Basic Math Input palette). Also shown is a use of the Equals (logical) operator that also is denoted == (a double equals sign) which tests for equality.

```
xx = (3 659 245 / 32)^35
```

```
52 324 138 062 709 450 247 173 031 492 939 096 825 340 617 853 687 538 644 935 629 577 351 517 \
  133 847 850 247 139 474 609 391 637 774 398 690 061 851 034 817 070 572 132 940 323 182 014 550 \
  495 359 365 854 893 775 235 605 247 675 578 524 370 984 986 712 031 535 003 486 348 432 488 739 \
  490 509 033 203 125 / 47 890 485 652 059 026 823 698 344 598 447 161 988 085 597 568 237 568
```

```
xx == (3 659 245 / 32) ^35
```

```
True
```

```
1. xx
```

1.09258×10^{177}

```
Pi
```

π

```
1. Pi
```

```
3.14159
```

```
N[Pi, 200]
```

```
3.14159265358979323846264338327950288419716939937510582097494459230781640628620899 \
 86280348253421170679821480865132823066470938446095505822317253594081284811174502 \
 8410270193852110555964462294895493038820
```

Comments, In, and Out

Note that any text that appears between (* and *) is ignored and thus represents an unevaluated comment. Each input evaluated is set equal to a consecutively and thus uniquely numbered value of the *Mathematica* function called In, and likewise each output is assigned as the value of a corresponding value of the Output function. The In and Out values appear on the left side of each evaluated Input cell and of each resulting Output cell. These can be handy if a cell is accidentally deleted and there is a need to retrieve its contents.

```
(* This is an unevaluated comment. *)
```

Equality and assignment or setting of one or more values to symbolic variables

In *Mathematica,* there are different forms of equality or assignment. A symbolic variable, say x, is set equal to a value by an expression of the form x = 2, after which x will be replaced by 2 unless/until the value of x is reassigned or cleared. Before x is defined in this way, it remains an unassigned symbol (with an implied Head called Symbol). Multiple applications of = (also denoted as the Set function) can be used within a single line or single expression. Delayed equality (also denoted as the SetDelayed function) is specified using :=, which is how

```
x
```

```
x
```

```
Head[x]
```

```
Symbol
```

```
x = 3.25
```

```
3.25
```

```
Head[x]
```

```
Real
```

```
Clear[x]
```

```
x
```

```
x
```

```
Head[x]
```

```
Symbol
```

List, Map, and MapThread

In *Mathematica,* a "list" of items (which may be numbers or symbols) is denoted by curly brackets (i.e., braces) the enclose each item separated by a comma. In other programming languages such lists are called arrays, and in mathematics thay are called vectors. Using the FullForm function reveals that brace notation is shorthand for a function that is named (i.e., has a "Head" called) List. The ith element of any list is denoted using double enclosing brackets; e.g., if A = {11, 2, 30, 41, 5}, then A[[3]] is automatically defined as the 3rd element 30 in the list A, and A[[{2,4}]] denotes a list of the 2nd and 4th elements of A (here, {2, 41}). A "Tensor" is any list of lists, and a Matrix is a list of lists that all have the same length. The syntax A@@B replaces the Head of expression A with the symbol B. For example, if m is defined to be the following tensor, m = {{a,b,c}, {1, 2, 3, 4}, {30,20}}, then evaluating m[[3,2]] will return 30 (the 2nd element of the 3rd list contained in m).

Note that adding spaces around elements within a list has no effect.

```
FullForm[zz = {a, b, c}]
```

```
List[a, b, c]
```

```
Head[zz]
```

```
List
```

```
List[a, b, c] == {a, b, c}
```

```
True
```

```
Plus @@ zz
```

```
a + b + c
```

```
Times @@ zz
```

```
a b c
```

```
VectorQ[{1, 2, Pi}, NumericQ]
```

```
True
```

```
VectorQ[{1, 2, Pi}, NumberQ]
```

```
False
```

```
Head[zz = {{B, 3, 4}, {A, 5, 3}, {1, 2, 3}}]
```

```
List
```

```
MatrixQ[zz]
```

```
True
```

```
? Map
```

Map[f, $expr$] or f/@$expr$ applies f to each element on the first level in $expr$.

Map[f, $expr$, $levelspec$] applies f to parts of $expr$ specified by $levelspec$.

Map[f] represents an operator form of Map that can be applied to an expression. >>

```
{Plus @@ #, Times @@ #} & /@ zz^2
```

$\{\{25 + B^2, 144\ B^2\}, \{34 + A^2, 225\ A^2\}, \{14, 36\}\}$

```
Expand[Sum[(x[i] + y[6 - i]) ^4, {i, 6}]]
```

$x[1]^4 + x[2]^4 + x[3]^4 + x[4]^4 + x[5]^4 + x[6]^4 + 4\,x[6]^3\,y[0] + 6\,x[6]^2\,y[0]^2 + 4\,x[6]\,y[0]^3 + y[0]^4 + 4\,x[5]^3\,y[1] + 6\,x[5]^2\,y[1]^2 + 4\,x[5]\,y[1]^3 + y[1]^4 + 4\,x[4]^3\,y[2] + 6\,x[4]^2\,y[2]^2 + 4\,x[4]\,y[2]^3 + y[2]^4 + 4\,x[3]^3\,y[3] + 6\,x[3]^2\,y[3]^2 + 4\,x[3]\,y[3]^3 + y[3]^4 + 4\,x[2]^3\,y[4] + 6\,x[2]^2\,y[4]^2 + 4\,x[2]\,y[4]^3 + y[4]^4 + 4\,x[1]^3\,y[5] + 6\,x[1]^2\,y[5]^2 + 4\,x[1]\,y[5]^3 + y[5]^4$

```
Remove[vM]
```

```
t1 = Table[vM[i] '[t] == (kMU + kMF) vM[i][t], {i, 5}]
```

```
{vM[1]′[t] == (kMF + kMU) vM[1][t],
 vM[2]′[t] == (kMF + kMU) vM[2][t], vM[3]′[t] == (kMF + kMU) vM[3][t],
 vM[4]′[t] == (kMF + kMU) vM[4][t], vM[5]′[t] == (kMF + kMU) vM[5][t]}
```

```
t2 = {ode1, ode2};
```

```
eqn = Join[t1, t2]
```

```
{vM[1]′[t] == (kMF + kMU) vM[1][t], vM[2]′[t] == (kMF + kMU) vM[2][t],
 vM[3]′[t] == (kMF + kMU) vM[3][t], vM[4]′[t] == (kMF + kMU) vM[4][t],
 vM[5]′[t] == (kMF + kMU) vM[5][t], ode1, ode2}
```

```
? Join
```

Join[$list_1$, $list_2$, …] concatenates lists or other expressions that share the same head.

Join[$list_1$, $list_2$, …, n] joins the objects at level n in each of the $list_i$. >>

```
? DSolve
```

DSolve[eqn, y, x] solves a differential equation for the function y, with independent variable x.

DSolve[{eqn_1, eqn_2, …}, {y_1, y_2, …}, x] solves a list of differential equations.

DSolve[eqn, y, {x_1, x_2, …}] solves a partial differential equation. >>

```
? vM
```

```
Global`vM

vM[i_][t_] := kMU vM[i][t]

? Table
```

Table[*expr*, {i_{max}}] generates a list of i_{max} copies of *expr*.
Table[*expr*, {i, i_{max}}] generates a list of the values of *expr* when i runs from 1 to i_{max}.
Table[*expr*, {i, i_{min}, i_{max}}] starts with $i = i_{min}$.
Table[*expr*, {i, i_{min}, i_{max}, di}] uses steps di.
Table[*expr*, {i, {i_1, i_2, …}}] uses the successive values i_1, i_2, ….
Table[*expr*, {i, i_{min}, i_{max}}, {j, j_{min}, j_{max}}, …] gives a nested list. The list associated with i is outermost. >>

```
Table[737, {10}]

{737, 737, 737, 737, 737, 737, 737, 737, 737, 737}

737 & /@ Range[10]

{737, 737, 737, 737, 737, 737, 737, 737, 737, 737}

Map[737 &, Range[10]]

{737, 737, 737, 737, 737, 737, 737, 737, 737, 737}

Map[Sin[#]^2 &, Range[10]]
```

$\{\mathrm{Sin}[1]^2, \mathrm{Sin}[2]^2, \mathrm{Sin}[3]^2, \mathrm{Sin}[4]^2, \mathrm{Sin}[5]^2, \mathrm{Sin}[6]^2, \mathrm{Sin}[7]^2, \mathrm{Sin}[8]^2, \mathrm{Sin}[9]^2, \mathrm{Sin}[10]^2\}$

```
save = Flatten[so = Table[{i^2, j}, {i, 5}, {j, 8}]]

{1, 1, 1, 2, 1, 3, 1, 4, 1, 5, 1, 6, 1, 7, 1, 8, 4, 1, 4, 2, 4, 3, 4, 4, 4, 5, 4, 6, 4,
 7, 4, 8, 9, 1, 9, 2, 9, 3, 9, 4, 9, 5, 9, 6, 9, 7, 9, 8, 16, 1, 16, 2, 16, 3, 16, 4,
 16, 5, 16, 6, 16, 7, 16, 8, 25, 1, 25, 2, 25, 3, 25, 4, 25, 5, 25, 6, 25, 7, 25, 8}

so

{{{1, 1}, {1, 2}, {1, 3}, {1, 4}, {1, 5}, {1, 6}, {1, 7}, {1, 8}},
 {{4, 1}, {4, 2}, {4, 3}, {4, 4}, {4, 5}, {4, 6}, {4, 7}, {4, 8}},
 {{9, 1}, {9, 2}, {9, 3}, {9, 4}, {9, 5}, {9, 6}, {9, 7}, {9, 8}},
 {{16, 1}, {16, 2}, {16, 3}, {16, 4}, {16, 5}, {16, 6}, {16, 7}, {16, 8}},
 {{25, 1}, {25, 2}, {25, 3}, {25, 4}, {25, 5}, {25, 6}, {25, 7}, {25, 8}}}

tr = Transpose[Flatten[so, 1]]

{{1, 1, 1, 1, 1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 9, 9, 9, 9, 9, 9,
  9, 9, 16, 16, 16, 16, 16, 16, 16, 16, 25, 25, 25, 25, 25, 25, 25, 25},
 {1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5,
  6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8}}
```

```
tr
{{1, 1, 1, 1, 1, 1, 1, 1, 4, 4, 4, 4, 4, 4, 4, 4, 9, 9, 9, 9, 9, 9,
  9, 9, 16, 16, 16, 16, 16, 16, 16, 16, 25, 25, 25, 25, 25, 25, 25, 25},
 {1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5,
  6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8, 1, 2, 3, 4, 5, 6, 7, 8}}
```

```
? MapThread
```

MapThread[f, {{a_1, a_2, …}, {b_1, b_2, …}, …}] gives {f[a_1, b_1, …], f[a_2, b_2, …], …}.
MapThread[f, {$expr_1$, $expr_2$, …}, n] applies f to the parts of the $expr_i$ at level n.
MapThread[f] represents an operator form of MapThread that can be applied to an expression. >>

```
MapThread[{#1, #2} &, {{1, 2, 3}, {a, b, c}}]
{{1, a}, {2, b}, {3, c}}
```

```
x1 = MapThread[(Sin[#1] ^2 + Cos[#2 ^2]) &, tr]
{Cos[1] + Sin[1]^2, Cos[4] + Sin[1]^2, Cos[9] + Sin[1]^2, Cos[16] + Sin[1]^2,
 Cos[25] + Sin[1]^2, Cos[36] + Sin[1]^2, Cos[49] + Sin[1]^2, Cos[64] + Sin[1]^2,
 Cos[1] + Sin[4]^2, Cos[4] + Sin[4]^2, Cos[9] + Sin[4]^2, Cos[16] + Sin[4]^2,
 Cos[25] + Sin[4]^2, Cos[36] + Sin[4]^2, Cos[49] + Sin[4]^2, Cos[64] + Sin[4]^2,
 Cos[1] + Sin[9]^2, Cos[4] + Sin[9]^2, Cos[9] + Sin[9]^2, Cos[16] + Sin[9]^2,
 Cos[25] + Sin[9]^2, Cos[36] + Sin[9]^2, Cos[49] + Sin[9]^2, Cos[64] + Sin[9]^2,
 Cos[1] + Sin[16]^2, Cos[4] + Sin[16]^2, Cos[9] + Sin[16]^2, Cos[16] + Sin[16]^2,
 Cos[25] + Sin[16]^2, Cos[36] + Sin[16]^2, Cos[49] + Sin[16]^2, Cos[64] + Sin[16]^2,
 Cos[1] + Sin[25]^2, Cos[4] + Sin[25]^2, Cos[9] + Sin[25]^2, Cos[16] + Sin[25]^2,
 Cos[25] + Sin[25]^2, Cos[36] + Sin[25]^2, Cos[49] + Sin[25]^2, Cos[64] + Sin[25]^2}
```

```
Flatten[so, 1]
{{1, 1}, {1, 2}, {1, 3}, {1, 4}, {1, 5}, {1, 6}, {1, 7}, {1, 8}, {4, 1}, {4, 2}, {4, 3},
 {4, 4}, {4, 5}, {4, 6}, {4, 7}, {4, 8}, {9, 1}, {9, 2}, {9, 3}, {9, 4}, {9, 5},
 {9, 6}, {9, 7}, {9, 8}, {16, 1}, {16, 2}, {16, 3}, {16, 4}, {16, 5}, {16, 6}, {16, 7},
 {16, 8}, {25, 1}, {25, 2}, {25, 3}, {25, 4}, {25, 5}, {25, 6}, {25, 7}, {25, 8}}
```

```
x2 = (Sin[#[[1]]] ^2 + Cos[#[[2]] ^2]) & /@ Flatten[so, 1]
{Cos[1] + Sin[1]^2, Cos[4] + Sin[1]^2, Cos[9] + Sin[1]^2, Cos[16] + Sin[1]^2,
 Cos[25] + Sin[1]^2, Cos[36] + Sin[1]^2, Cos[49] + Sin[1]^2, Cos[64] + Sin[1]^2,
 Cos[1] + Sin[4]^2, Cos[4] + Sin[4]^2, Cos[9] + Sin[4]^2, Cos[16] + Sin[4]^2,
 Cos[25] + Sin[4]^2, Cos[36] + Sin[4]^2, Cos[49] + Sin[4]^2, Cos[64] + Sin[4]^2,
 Cos[1] + Sin[9]^2, Cos[4] + Sin[9]^2, Cos[9] + Sin[9]^2, Cos[16] + Sin[9]^2,
 Cos[25] + Sin[9]^2, Cos[36] + Sin[9]^2, Cos[49] + Sin[9]^2, Cos[64] + Sin[9]^2,
 Cos[1] + Sin[16]^2, Cos[4] + Sin[16]^2, Cos[9] + Sin[16]^2, Cos[16] + Sin[16]^2,
 Cos[25] + Sin[16]^2, Cos[36] + Sin[16]^2, Cos[49] + Sin[16]^2, Cos[64] + Sin[16]^2,
 Cos[1] + Sin[25]^2, Cos[4] + Sin[25]^2, Cos[9] + Sin[25]^2, Cos[16] + Sin[25]^2,
 Cos[25] + Sin[25]^2, Cos[36] + Sin[25]^2, Cos[49] + Sin[25]^2, Cos[64] + Sin[25]^2}
```

```
x1 == x2
```

```
True
```

Sort, Reverse, Tally, Count, etc. (operations on lists)

The *Mathematica* Sort function can be used to sort elements within any list. Other functions like those listed below can also be used to manipulate or analyze elements in a list.

```
? Length
```

Length[*expr*] gives the number of elements in *expr*. >>

```
? Sort
```

Sort[*list*] sorts the elements of *list* into canonical order.

Sort[*list*, *p*] sorts using the ordering function *p*. >>

```
? Reverse
```

Reverse[*expr*] reverses the order of the elements in *expr*.

Reverse[*expr*, *n*] reverses elements at level *n* in *expr*.

Reverse[*expr*, {n_1, n_2, …}] reverses elements at levels n_1, n_2, … in *expr*. >>

```
? Tally
```

Tally[*list*] tallies the elements in *list*, listing all distinct elements together with their multiplicities.

Tally[*list*, *test*] uses *test* to determine whether pairs of elements should be considered equivalent, and gives a list of the first representatives of each equivalence class, together with their multiplicities. >>

```
? Count
```

Count[*list*, *pattern*] gives the number of elements in *list* that match *pattern*.

Count[*expr*, *pattern*, *levelspec*] gives the total number of subexpressions matching *pattern* that appear at the levels in *expr* specified by *levelspec*.

Count[*pattern*] represents an operator form of Count that can be applied to an expression. >>

```
? Cases
```

Cases[{e_1, e_2, …}, *pattern*] gives a list of the e_i that match the pattern.

Cases[{e_1, …}, *pattern* → *rhs*] gives a list of the values of *rhs* corresponding to the e_i that match the pattern.

Cases[*expr*, *pattern*, *levelspec*] gives a list of all parts of *expr* on levels specified by *levelspec* that match the pattern.

Cases[*expr*, *pattern* → *rhs*, *levelspec*] gives the values of *rhs* that match the pattern.

Cases[*expr*, *pattern*, *levelspec*, *n*] gives the first *n* parts in *expr* that match the pattern.

Cases[*pattern*] represents an operator form of Cases that can be applied to an expression. >>

```
? Select
```

Select[*list*, *crit*] picks out all elements e_i of *list* for which *crit*[e_i] is True.

Select[*list*, *crit*, *n*] picks out the first *n* elements for which *crit*[e_i] is True.

Select[*crit*] represents an operator form of Select that can be applied to an expression. >>

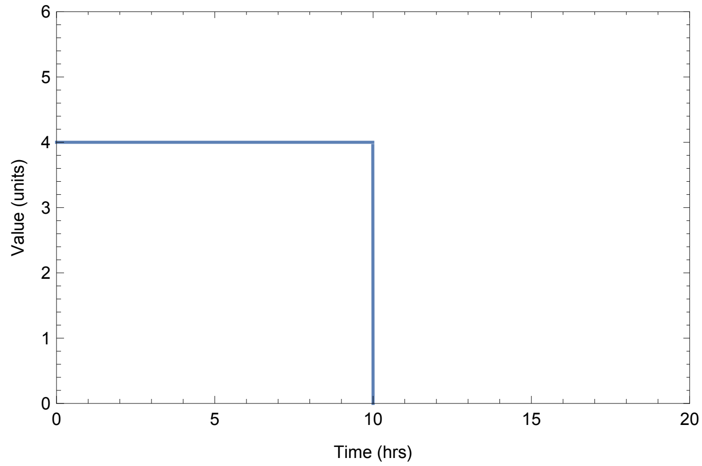
```
Sort[qq = {a, b, w, z, "This is a String", K, b, 4^2, s, k, 3, 1, s, s, w, 5, t}]
{1, 3, 5, 16, This is a String, a, b, b, k, K, s, s, s, t, w, w, z}

Length[qq]
17

Reverse[qq]
{t, 5, w, s, s, 1, 3, k, s, 16, b, K, This is a String, z, w, b, a}

Tally[qq]
{{a, 1}, {b, 2}, {w, 2}, {z, 1}, {This is a String, 1},
 {K, 1}, {16, 1}, {s, 3}, {k, 1}, {3, 1}, {1, 1}, {5, 1}, {t, 1}}

Count[qq, w]
2

{Cases[qq, _Integer], Cases[qq, _String]}
{{16, 3, 1, 5}, {This is a String}}

Select[qq, ((# < 7) || StringQ[#]) &]
{This is a String, 3, 1, 5}
```

Basic plotting using the *Mathematica* Plot function

Constant input

```
{dose, tend} = {40, 10};
rate = dose / tend;
g[t_] := If[0 ≤ t ≤ tend - 10^-2, rate, 10^2 rate (tend - t)]
```

```
Plot[g[t], {t, 0, 20}, AxesOrigin → {0, 0}, PlotRange → {{0, 20}, {0, 6}},
 Frame → True, FrameLabel → {"Time (hrs)", "Value (units)"}, PlotPoints → 2500]
```



A symbolic expression for exponential input from a Dose compartment is defined below. Note that *Mathematica* can do symbolic integration and dffierentiation (and many other symbolic mathematical operations) entirely analytically, yeilding symbolic output.

```
Clear[k];
Assuming[k > 0, Integrate[k * Xo * E ^ (-k * t), {t, 0, Infinity}]]
```

```
Xo
```

```
? DSolve
```

DSolve[eqn, u, x] solves a differential equation for the function u, with independent variable x.

DSolve[eqn, u, {x, x_{min}, x_{max}}] solves a differential equation for x between x_{min} and x_{max}.

DSolve[{eqn_1, eqn_2, …}, {u_1, u_2, …}, …] solves a list of differential equations.

DSolve[eqn, u, {x_1, x_2, …}] solves a partial differential equation.

DSolve[eqn, u, {x_1, x_2, …} ∈ Ω] solves the partial differential equation eqn over the region Ω.

DSolve[eqn, u, {t, t_{min}, t_{max}}, {x_1, x_2, …} ∈ Ω]
 solves the time–dependent partial differential equation eqn over the region Ω. ≫

```
DSolve[{Dose'[t] == -k * Dose[t], Dose[0] == Do}, Dose[t], t]
```

$\{\{\text{Dose}[t] \to 1.\ \text{Do}\ e^{-0.53319\, t}\}\}$

```
DSolve[{Dose'[t] == -k * Dose[t], Dose[0] == Do}, Dose[t], t][[1, 1]]
```

$\text{Dose}[t] \to \text{Do}\ e^{-k\, t}$

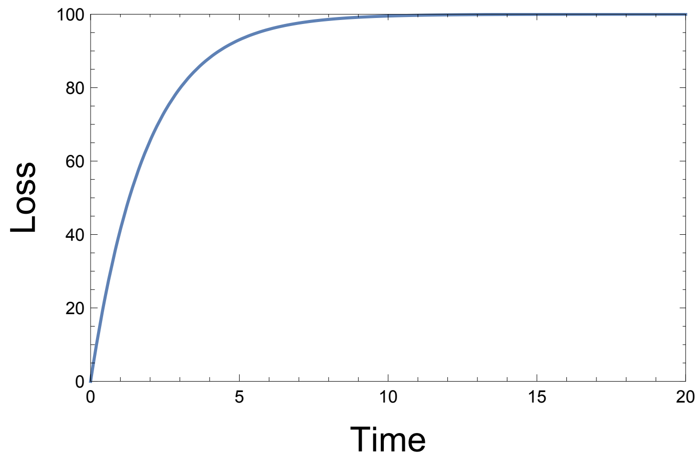
```
{dose, k} = {100, Log[2] / 1.3};
lossRate[t_] := dose * k * E ^ (-k * t);
integratedLoss = Integrate[lossRate[t], {t, 0, Infinity}]
```

```
100.
```

```
integratedLoss == dose
```

```
True
```

```
Plot[Integrate[lossRate[t], {t, 0, s}], {s, 0, 20}, Frame → True,
 FrameLabel → (Style[#, FontWeight → Plain, FontColor → Black,
      FontFamily → "Arial", FontSize → 18] & /@ {"Time", "Loss"}),
 AxesOrigin → {0, 0}, PlotRange → {{0, 20}, {0, 100}}]
```



Plotting via the PlotData function of Dr. Bogen's *RiskQ* package for *Mathematica*

After the *RiskQ* (or ExponentDimethoateModel) package is loaded, expressions like those that follow may be evaluated to yield plots like those illustrated below.

Details about PlotData function syntax are summarized below. If quotes appear surrounding the usage statement, evaluate the ?PlotData expression a second time and they will disappear.

```
? PlotData
```

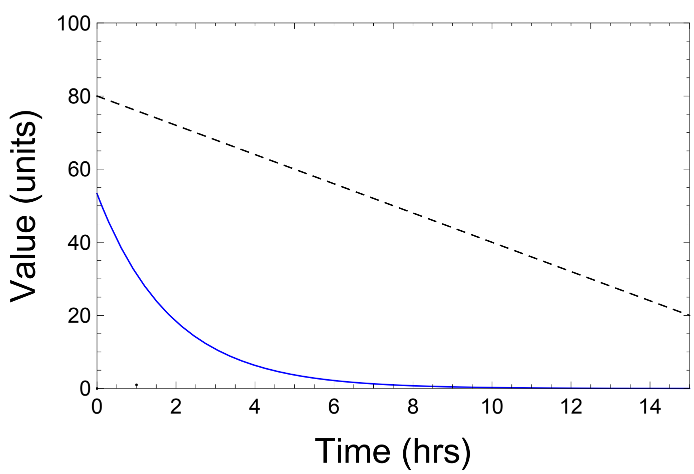
PlotData[data, options:] plots an N–by–2 (or (x vs y) data set, or a list of n such sets, with points joined by lines (unless JoinPoints→False is used). Change point style with Style→stylelist which by defualt is {OO, OA, OB, OV, OD, O, A, B, V, D} = {open Point, Triangle, Square, InvTriangle, Diamond,...& their solid counterparts}; use {TO, TA, TB, TV, TD} for transparent open symbols; use {P,X,M,I} for {plus, cross, dash, bar}; & use J to join points from adjacent data sets. E.g, if data = {xyListLo, xyListHi, xyListHat} is a matrix listing corresponding lower–bound, upper–bound, and best–estimate points, use Style→{M,J,M,OO} to plot error bars around open–circle best estimates. Note that Length[stylelist] ≤ (n + nJ) must hold with nJ ≤ 2n–1, where nJ is the number of (non–adjacent) J's, and stylelist cannot end with J. Use Tweak→k to induce a k–fold change in the opaque central disc of points made using an 'OO' Style. (If PlotType →Prob or →LogProb [see below], stylelist elements all should instead be either Mathematica Special Characters palette symbols [use Style→Markers to list examples] or Style expressions.)

Use Color→color (where color is a list {red,green,blue} of arguments to the Mathematica RGBColor function, or one of the Mathematica 'Named Colors') to color lines joining the specified point set(s); use DotColor to similarly color plotted points. JoinPoints, Color, DotColor or DotSize option (see PlotOptions) values may each also be a list (of length ≤n), to pertain respectively to each point set to be joined. Use FitTo→ {f[x], x} to include a plot of f[x] (which may be a list of functions) vs. any specified symbol x. Use JPL→True (where JPL denotes Join Points by Lines) to set JoinPoints→True, DotSize→0.0001, Style→O.

If the FitTo option is used, or otherwise to a lesser extent (see PlotOptions), Mathematica Plot options (Axes, AxesLabel, AxesStyle, Background, BaseStyle, ColorOutput, Compiled, Epilog, Filling, FillingStyle, FrameStyle, FrameTicks, GridLines, GridLinesStyle, ImageSize, MaxBend, PlotDivision, PlotPoints, PlotStyle [e.g., to color one or more non–DASHed FitTo functions], Prolog, Ticks,FormatType, and TextStyle) may also be used. Use THICK→thick or DASH→dash, where thick and dash may each be a list pertaining to each respective function. Use data=Plot to plot functions only.

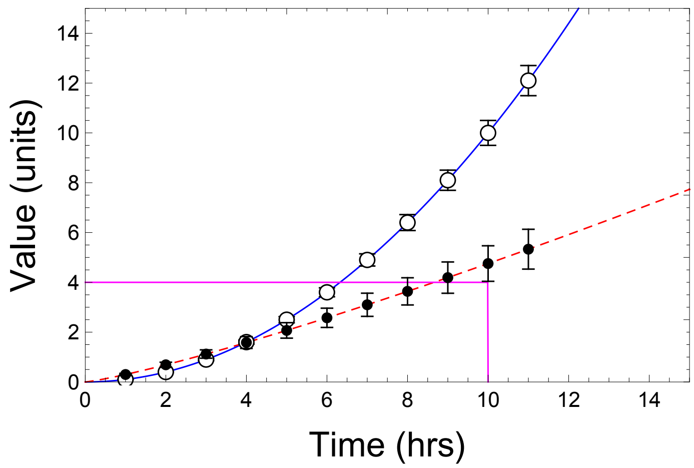
ONLY IF the FitTo option is used, PlotType→type may also be used, with type =Log, =LinLog, =LinearLog for x vs. Ly, type= LogLin, or =LogLinear, for Lx vs. y, type =LogLog for Lx vs. Ly, type =Prob for x vs. Py, or type =LogProb for Lx vs. Py, where x = the abscissa, y = the ordinate, L denotes base–10 logarithm, and Py = Φ(y=z) = the cumulative normal distribution function of plotted ordinate arguments z but shown scaled along a Y–axis showing corresponding frame tick percentages Pyi that may be input as the list PctTicks→Pyi. If PlotType →Prob or →LogProb, use Prob→Z if ordinate values yi = Last/@xy and y=f[x] (specified using FitTo) are all standard normal variate values (i.e., if y = yi = inv(Py) = z) = (y=z) = the cumulative normal distribution function of plotted ordinate arguments z but shown scaled along a Y–axis showing corresponding frame tick percentages Pyi that may be input as the list PctTicks→Pyi. If PlotType →Prob or →LogProb, use ProbPlotY→Z if ordinate values yi = Last/@xy and y=f[x] (specified using FitTo) are all standard normal variate values (i.e., if y = yi = Φinv(Py) = z) rather than probability values Py. Use Output→ PlotRange to return only the plotrange list {{Xmin, Xmax}, {Ymin, Ymax}} used for plotting. For more details, see PlotOptions.

```
PlotData[Plot, X → {0, 15}, Y → {0, 100}, FitTo → {{lossRate[t], 80 - 4 t}, t},
 Labels → {"Time (hrs)", "Value (units)"}, PlotStyle → {Blue, {Black, Dashed}}]
```



```
xi = Range[11];
yhat1 = 0.1 xi^2;
yhat2 = 0.3 xi^1.2;
{sy1, sy2} = {0.05 yhat1, 0.15 yhat2};
d1 = Transpose[{xi, #}] & /@ {yhat1 - sy1, yhat1 + sy1, yhat1};
d2 = Transpose[{xi, #}] & /@ {yhat2 - sy2, yhat2 + sy2, yhat2};
```

```
PlotData[Join[d1, d2], Style → {M, J, M, OO, M, J, M, O},
 DotSize → 0.01 {2, 2, 2, 1.5, 1.5, 1.5}, X → {0, 15}, Y → {0, 15},
 FitTo → {{0.1 t^2, 0.3 t^1.2, g[t]}, t}, Labels → {"Time (hrs)", "Value (units)"},
 PlotStyle → {Blue, {Red, Dashed}, Magenta}, PlotPoints → 2500]
```



Mathematica syntax examples (rules, random numbers, interpolation)

In *Mathematica*, a right arrow (entered either as -> or instead as → which symbol can be accessed using the Basic Math input palattte in the Palettes tab at the top of your screen when you run *Mathematica*) signifies a Rule, which also may be specifed using the Rule function. A Rule tells Mathematica to replace the expression on the left (whenever it is encountered) with that on the right, after conditioning an expression by that specific Rule. In Mathematica, the syntax A /. B means the expression A is conditioned on a Rule B (where B must be defined as or have the form of a Rule).

```
√5 → R2
√5 → R2
```

```
Rule[√5, R2]
√5 → R2
```

```
Head[√5 → R2]
Rule
```

```
xyz^R2
xyz^R2
```

```
xyz^(R1+√5) /. {√5 → R2, R1 → 4 π}
```

$xyz^{4\pi+R2}$

```
Clear[y];
ruleLists = {{x → 1/2 (1 - √5), y → 5 x, 1 - √5 → R1}, {x → 1/2 (1 + √5), y → x^3,
    z → w^3, 1 + √5 → R2}, {x → (1 + √5)^2, y → x^4, z → x^5, (√5 → R2)}};
```

In *Mathematica*, the syntax A&/@B means that the expression or function A is mapped onto a list B, where lists are denoted by curly braces that enclose a list of objects separated by commas (more about Lists and mapping below). If theh expression A contains a pound sign (#), then that sign means “the current member of the list” on which A is being mapped. This type of notation is “called functional programming”, and it provides a very efficient way to do repeated operations without having to define an indexed array or table or “Do-loop” without having to pre-define any index or index dimensions.

```
(x /. #) & /@ ruleLists
```

$\left\{\frac{1}{2}\left(1-\sqrt{5}\right), \frac{1}{2}\left(1+\sqrt{5}\right), \left(1+\sqrt{5}\right)^2\right\}$

```
(x //. #) & /@ ruleLists
```

$\left\{\frac{R1}{2}, \frac{R2}{2}, (1+R2)^2\right\}$

```
x /. ruleLists[[1]]
```

$\frac{1}{2}\left(1-\sqrt{5}\right)$

```
? Equal
```

lhs == *rhs* returns True if *lhs* and *rhs* are identical. ≫

```
Table[#[], {20}] & /@ {RandomReal, RandomInteger}
```

```
{{0.864641, 0.096901, 0.0734126, 0.0327095, 0.18955, 0.147362,
  0.576299, 0.711064, 0.161991, 0.765453, 0.443932, 0.683322, 0.220742,
  0.297363, 0.512075, 0.811861, 0.810201, 0.125032, 0.259219, 0.93902},
 {1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 1}}
```

```
RandomReal[{-2, 5}, 20]
```

```
{1.31037, 0.968886, 2.43988, 3.98995, -0.178683, -0.209656,
 0.211267, 1.16452, 0.495143, 2.40793, -1.73286, -1.34917, 2.97235,
 4.04169, 1.01865, 1.71291, 3.91366, 1.94066, -1.09391, 2.20839}
```

```
Table[2 (RandomReal[] - 1/2), {21}]
```

```
{-0.630129, -0.520293, 0.279354, -0.486307, 0.217877, -0.319721, 0.948429,
 0.936208, 0.48241, -0.917222, -0.00803415, 0.987559, 0.0402087, 0.582291,
 -0.707809, -0.0823892, -0.591894, 0.814933, 0.963221, -0.903378, 0.703196}
```

The functions Interpolation[xyList] and Interpolation[xyList, InterpolationOrder→n] return InterpolatingFunction objects intFxn that are defined between x1 and xk for any list of points xyList = {{x1,y1}, {x2,y2}, ..., {xk,yk}}, where xyList must be ordered with respect to x (i.e., sorted by x-value), with xi (i=1,...,k) specifying unique (i.e., non-repeated, typically increasing) values along the X-axis. By default (without using an InterpolationOrder option), n > 1 and n is optimized to yield a reasonably smooth fit to all the data. intFxn[xo] returns the corresponding interpolated value along the Y-axis at the X-axis value X = xo, provided x1 ≤ xo ≤ xk.

```
? Interpolation
```

Interpolation[{f_1, f_2, …}] constructs an interpolation of the function values f_i, assumed to correspond to x values 1, 2, … .

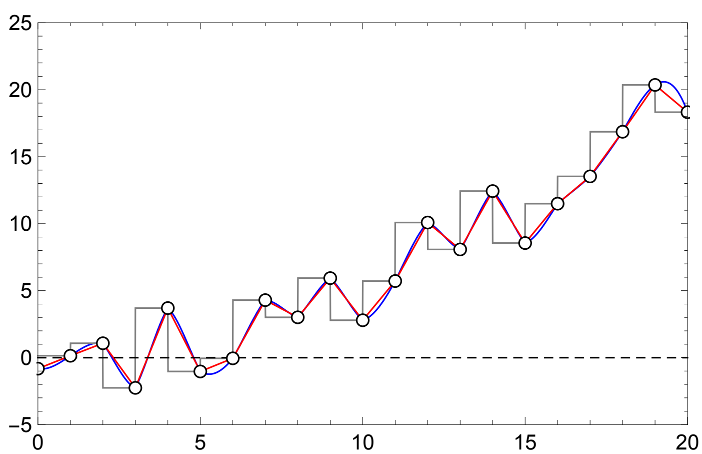
Interpolation[{{x_1, f_1}, {x_2, f_2}, …}] constructs an interpolation of the function values f_i corresponding to x values x_i.

Interpolation[{{{x_1, y_1, …}, f_1}, {{x_2, y_2, …}, f_2}, …}] constructs an interpolation of multidimensional data.

Interpolation[{{{x_1, …}, f_1, df_1, …}, …}] constructs an interpolation that reproduces derivatives as well as function values.

Interpolation[*data*, *x*] find an interpolation of *data* at the point *x*. »

```
xyData = Transpose[
   {xi = Range[0, 20], yi = 0.05 (60 * Table[2 (RandomReal[] - 1/2), {21}] + xi^2)}];
intfxn = Interpolation[xyData];
intfxn1 = Interpolation[xyData, InterpolationOrder → 1];
intfxn0 = Interpolation[xyData, InterpolationOrder → 0];
PlotData[xyData, FitTo → {{intfxn[x], intfxn1[x], intfxn0[x], x * 10^-20}, x},
 X → {0, 20}, Y → {-5, 25}, PlotStyle → {Blue, Red, Gray, {Dashed, Black}}]
```




```
intfxn
InterpolatingFunction[{{0., 20.}}, <>]
```

```
intfxn1
InterpolatingFunction[{{0., 20.}}, <>]
```

Note that NDSolve returns a list of rules that define all of the ODE-system variables listed in a variable list (e.g., vars = {A[t], B[t], ... }) as a function of the independent variable defined in the last argument of the NDSolve expression used, such as t in the expression sol = NDSolve[eqnAll, vars, {t, 0, tmax}]. In the latter expression, note that sol is used to capture, in a convenient and symbolic way, the list of rules returned by NDSolve, so that this list may be referred to easily.

Programming (i.e., defining functions) in *Mathematica*

Asignment or “setting” of values to be associated with symbols can be immediate (using =, which is called “Set”), or can be delayed (using :=, which is called

"SetDelayed"):

The right side of an immediate definition is evaluated when the definition is made:

```
x = Random[];
{x, x, x}
```

```
{0.243356, 0.243356, 0.243356}
```

The right side of an immediate definition is evaluated each time the definition is used. Note that after a symbol has been defined, its color changes from blue to black. If a red color appears for a symbol in an expression, this indicates that something is wrong with the syntax being used, and a message button appears at the right with a "+" in it, which when clicked opens an error message.

```
a * .b
```

Syntax: "a*" cannot be followed by ".b".

```
y := Random[];
{y, y, y}
```

```
{0.0665561, 0.450893, 0.474645}
```

Make definitions for special and general cases using immediate and delayed assignments:

```
factorial[1] = 1;
factorial[n_] := n fact[n - 1]
```

```
factorial[50]
```

```
30 414 093 201 713 378 043 612 608 166 064 768 844 377 641 568 960 512 000 000 000 000
```

Here the same (exact, 65-digit!) result is obtained using *Mathematica's* built-in factorial (!) function:

```
{result = 50!, Length[IntegerDigits[result]]}
```

```
{30 414 093 201 713 378 043 612 608 166 064 768 844 377 641 568 960 512 000 000 000 000, 65}
```

Programming by defining functions

Most programming in *Mathematica* consists of defining useful functions to generate the output you want as a function of values of the input arguments used by such functions. Here are some illustrations of simple functions. Note that you do not need a semi-colon after a 1-line function definition. Note that the argument-variable name of a function on the left side of a definition is always followed by an underscore, which stands for a "pattern" that matches the form (typically a symbol) the precedes the underscore. The variable used to name the argument thus "names" the pattern to be recognized as that argument.

```
myFxn[x_] := x^2 + 3
```

```
myFxn[5]
```

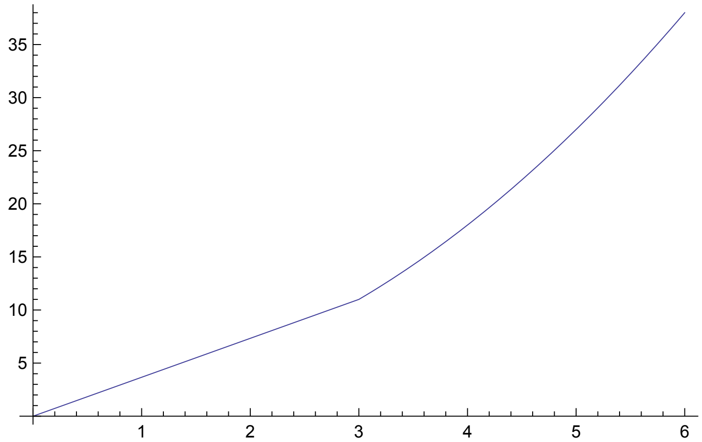
```
28
```

```
myFxn /@ {1, 2, 3, 4, a, b, c, π, 2.2}
```

$\{4, 7, 12, 19, 3 + a^2, 3 + b^2, 3 + c^2, 3 + \pi^2, 7.84\}$

```
myFxn[x_] := If[x < 3, (11/3) x, x^2 + 2]
```

```
Plot[myFxn[x], {x, 0, 6}]
```



Here's how to place a condition on the value of the argument x, such that the functin is returned unevaluated if the condition does not yield True. Note that "/;" means "conditional on". Since we are not changing function names yet, but we are adding a condition, we need to first remove the previous function definition of myFxn so that *Mathematica* will not get confused. Unless you use a new function name, removal of a previously defined function **must also** be done **whenever** changes are made to function arguments or to any members of the list of internal variables that are defined using a Module construct when defining functions (as discussed below). We will call these types of function changes "fundamental" changes. You only need to remove once before making a fundamental change. Below, note that == tests for equality (but may return an unevaluated expression involving ==), whereas === forces such a test to yield either False or True (instead of possibly returning an unevaluated expression involving ==).

```
Remove[myFxn];
myFxn[x_, y_] := ExpandAll[(x^2 + 3)^y]
```

```
myFxn[R, 4]
```

$81 + 108\, R^2 + 54\, R^4 + 12\, R^6 + R^8$

```
myFxn[R]
```

myFxn[R]

```
Remove[myFxn];
myFxn[{x_, y_}, z_] := ExpandAll[(x^2 + 3)^y] / z
```

```
myFxn[3, 4, 5]
```

myFxn[3, 4, 5]

```
myFxn[3, 4]
```

myFxn[3, 4]

```
myFxn[{R, 3}, 5]
```

$\frac{1}{5}\left(27 + 27\, R^2 + 9\, R^4 + R^6\right)$

```
Remove[myFxn];
myFxn[x_] := If[x < 3, (11 / 3) x, x^2 + 2] /; NumericQ[x]
```

```
myFxn /@ {1, 2, 3, 4, a, b, c, π, 2.2}
```

$\left\{\frac{11}{3}, \frac{22}{3}, 11, 18, \text{myFxn}[a], \text{myFxn}[b], \text{myFxn}[c], 2+\pi^2, 8.06667\right\}$

As always, there are usually several ways to make the same condition. For example, suppose we want myFxn to operate only on x if x is a list. Here's 3 ways to do this. Note that here, y is just an "auxiliary variable" defined to make the function definition easier to write down. Also, here, parentheses are used to **group** a series of expressions separated by semicolons that together define the function, and the condition is placed on that parenthesized **group** (not **list**!) of expressions.

```
Remove[myFxn];
myFxn[x_] := (y = Max[Select[x, NumericQ]];
   If[y < 3, (11/3) y, y^2 + 2] ) /; VectorQ[x]
```

```
myFxn[6]
```

```
myFxn[6]
```

```
myFxn[{a, 3.3, 4}]
```

```
18
```

```
Remove[myFxn];
myFxn[x_] := (y = Max[Select[x, NumericQ]];
   If[y < 3, (11/3) y, y^2 + 2] ) /; Head[x] === List
```

```
myFxn[6]
```

```
myFxn[6]
```

```
myFxn[{1, b, 3.3, 4}]
```

```
18
```

```
Remove[myFxn];
myFxn[x_List] := (y = Max[Select[x, NumericQ]];
  If[y < 3, (11/3) y, y^2 + 2] )
```

```
myFxn[6]
```

```
myFxn[6]
```

```
myFxn[{1, b, 3.3, 4}]
```

```
18
```

Here we define a two-argument function, with arguments named x and y, where x must be a list, and z must be an Integer.

```
Remove[myFxn];
myFxn[x_List, z_Integer] := (y = Max[Select[x, NumericQ]];
  If[y < 3, (11/3) y, y^z + 2] )
```

```
myFxn[3.3, 4]
```

```
myFxn[3.3, 4]
```

```
myFxn[{1, b, 3.3, 4}, 5.5]
```

```
myFxn[{1, b, 3.3, 4}, 5.5]
```

```
myFxn[3.3, 4.3]
```

```
myFxn[3.3, 4.3]
```

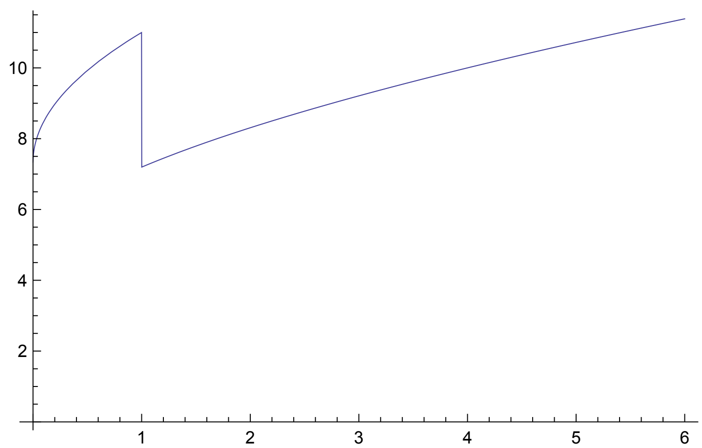
```
myFxn[{1, b, 3.3, 4}, 4]
```

```
258
```

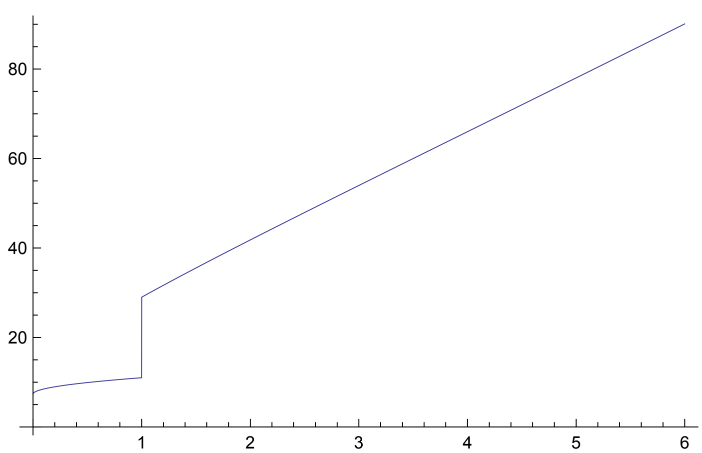
Here we define a two-argument function, with arguments named x and z, where the last argument (y) is optional and has a default value of 3 if it absent.

```
Remove[myFxn];
myFxn[x_, z_: 3] := (y = Sqrt[x] + 2;
  If[y < 3, (11/3) y, y^z + 2])
```

```
Plot[myFxn[x, 1.5], {x, 0, 6}, AxesOrigin → {0, 0}]
```




```
Plot[myFxn[x], {x, 0, 6}, AxesOrigin → {0, 0}]
```



For discontinuous functions, the built-in function Piecewise is useful. Here we also define two auxiliary variables, y and w.

```
? Piecewise
```

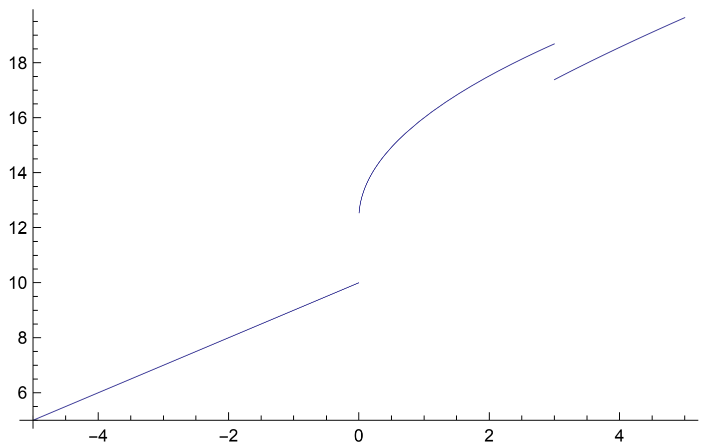
Piecewise[{{val_1, $cond_1$}, {val_2, $cond_2$}, …}] represents
a piecewise function with values val_i in the regions defined by the conditions $cond_i$.
Piecewise[{{val_1, $cond_1$}, …}, val] uses default value val if none of the $cond_i$ apply. The default for val is 0. >>

```
Remove[myFxn];
myFxn[x_, z_: 3] := (w = 10;
  y = Sqrt[x] + 2;
  Piecewise[{{x + w, x < 0}, {(11/3) y + w/2, 0 ≤ x < 3}, {y^z + 8, True}}] )
```

```
myFxn[a]
```

$$\begin{cases} 10 + a & a < 0 \\ 5 + \frac{11}{3}\left(2 + \sqrt{a}\right) & 0 \leq a < 3 \\ 8 + \left(2 + \sqrt{a}\right)^3 & \text{True} \end{cases}$$

```
Plot[myFxn[x, 1.7], {x, -5, 5}, AxesOrigin → {-5, 5}]
```



It is not good programming style to use auxiliary variables that create definitions for those variables that hang around and may confuse things later when you may have forgotten you already defined such variables earlier as auxiliary variables. If later you re-define such a variable, it will then (typically unintentionally and disasterously) affect later calculations, as shown below.

```
w = -50;
xx = w + 200
```

```
150
```

```
myFxn[a]
```

$$\begin{cases} 10 + a & a < 0 \\ 5 + \frac{11}{3}\left(2 + \sqrt{a}\right) & 0 \leq a < 3 \\ 8 + \left(2 + \sqrt{a}\right)^3 & \text{True} \end{cases}$$

```
xx = w + 200
```

```
210
```

The solution to this problem, and a key aspect of programming function definitions in *Mathematica*, is to use *Mathematica*'s built-in Module function in each (nontrivial) function definition you make, in order to (for each such function) define a function-specific set of auxiliary variables that only have defintions that are always unique and are local to that function, so the internal values of those local variables can never confuse other calculations in your notebook.

Within a Module function, the first argument contains a list of each and every local auxiliary variable you want to define. Note that these variables (as well as all other arguments defined for a function) all turn green so you can see all of these more easily. Within the local-variable list, you can also assign values to any or all of those variables using an equals sign. Using a Module function this way eliminates the need for parentheses that were used above.

On the left side of a defintion of a function such as myFxn, if an argument-variable name followed by an underscore (such as z_) is then followed by a colon (as in z_:) that is itself followed by another expression (say, X) as in the expression "z_:X" without the quotes, then if the argument z does not appear when myFxn is used, then z is assumed by default to equal X. Also, in a function definition, an argument like z_ can be followed one of the following **expression types:** Symbol, List, Integer, Rational, Real, Complex. You can get more fancy about retricting or testing such patterns, e.g., if a parameter is defined as "z_?NumericQ" (without the quotes, then only a numeric value will be recognized for that parameter and otherwise the function is returned unevaluated. These pattern tests can also be used in built-in pattern-testing functions, such as Cases. For more information on this, enter (i.e., search for) "guide/Patterns" and also see "ref/PatternTest" in the *Mathematica* Documentation Center.

```
Clear[x, y];
Cases[{1, 2, 3.5, x, y, {a, b, c}, 4}, _?NumberQ]
```

```
{1, 2, 3.5, 4}
```

```
Remove[myFxn];
myFxn[x_, z_Integer: 3] := Module[{w = 10, y},
   If[NumericQ[x], If[x < -20, Return[Bad]]];
   y = Sqrt[x] + 2;
   Piecewise[{{x + w, x < 0}, {(11/3) y + w/2, 0 ≤ x < 3}, {y^z + 8, True}}]
  ] /; NumericQ[x] || Head[x] === Symbol
```

```
myFxn[a]
```

$$\begin{cases} 10 + a & a < 0 \\ 5 + \frac{11}{3}\left(2 + \sqrt{a}\right) & 0 \le a < 3 \\ 8 + \left(2 + \sqrt{a}\right)^3 & \text{True} \end{cases}$$

```
myFxn[a, 2]
```

$$\begin{cases} 10 + a & a < 0 \\ 5 + \frac{11}{3}\left(2 + \sqrt{a}\right) & 0 \le a < 3 \\ 8 + \left(2 + \sqrt{a}\right)^2 & \text{True} \end{cases}$$

```
myFxn[5]
```

$$8 + \left(2 + \sqrt{5}\right)^3$$

```
myFxn[5, 2.5]
```

```
myFxn[5, 2.5]
```

```
myFxn[v == 6, 2.5]
```

```
myFxn[v == 6, 2.5]
```

```
myFxn[-25]
```

```
Bad
```

Note that on the left side, argument variables are typically defined by being followed by a single underscore. However, an argument-variable name followed by **two** underscores (as in: myFxn[x__]:= ...) means a **sequence** of **one or more** input expressions (of any kind) each separated by a comma (as in the expressions, myFxn[5, z] or myFxn[5, {3,4,5}, zzz]). An argument-variable name followed by **three** underscores (as in: myFxn[x___]:= ...) means **either no argument or a sequence** of **one or more** input expressions (of any kind) each separated by a comma (as in the expressions, myFxn[ ] or myFxn[5] or myFxn[5,6,7).

```
Remove[myFxn];
myFxn[x_, z__] := Module[{w = 10, y},
    If[NumericQ[x], If[x < -20, Return[Bad]]];
    y = Sqrt[x] + 2;
    Piecewise[{{x + w, x < 0}, {(11/3) y + w/2, 0 ≤ x < 3}, {y^{z} + 8, True}}]
   ] /; NumericQ[x] || Head[x] === Symbol
```

```
myFxn[5, 6, 7, K]
```

$\left\{8 + \left(2 + \sqrt{5}\right)^6, 8 + \left(2 + \sqrt{5}\right)^7, 8 + \left(2 + \sqrt{5}\right)^K\right\}$

```
Remove[myFxn];
myFxn[x_, z___:] := Module[{w = 10, y},
    If[NumericQ[x], If[x < -20, Return[Bad]]];
    y = Sqrt[x] + 2;
    Piecewise[{{x + w, x < 0}, {(11/3) y + w/2, 0 ≤ x < 3}, {y^{z} + 8, True}}]
   ] /; NumericQ[x] || Head[x] === Symbol
```

For non-trivial functions, a key to efficient function-definition programming in *Mathematica* (as with other modern object-oriented programs) is that each function should be smart and flexible enough to interrogate its fairly simple set of arguments to return output that can be useful in a variety of different but related contexts. This obviates the needs to write different function definitions to address a number of related but non-identical problems. This goal is facilitated by allowing "options" to be used with a function. The following example shows how to do this.

Note, the local variable name used to denote all the options (by convention, typically the name “options” is used) is always followed by three underscores and the Rule, to denote a sequence of zero, one, or more option-rules that a user may enter as the “final” argument(s) of a function. When using options, the default options values are defined separately, by using an Options function applied to the name of the function for which options are being defined, as shown below. The value of each option must be defined in the function, using two “conditional on” symbols (/.) that surround the options-sequence variable name with braces, to be able to treat the seuqence as a true list; this is followed by Options[function-Name] in order to condition the options on their respective intended default values. Function-definition code is more neat & clear if the options definitions precede the rest of the code but followed by a blank line before that remaining code. It also helps to unbold the input text used to define the function options, and then insert a blank line preceding a function definition. The symbol Automatic is often used as a dummy default value for options intended only to change things if some alternative option value is used.

To debug code used to program a function, it helps (during the debugging stage) to insert Print[variable(s) to print each separated by a comma, and/or text in quotes] followed by a semicolon at judicious places to print out intermediate calculations. A related trickk is to define a Verbose rule that can be set to True when more detail is desired (i.e., to give information hopefully relevant to might have gone wrong during code implementation). Note that Undefined is a special *Mathematica* symbol that returns as Undefined even if various functions are applied to it (thus, Undefined^2 returns Undefined).

```
Remove[myFxn];
Options[myFxn] = {Route → Oral, W → 10, Verbose → False};

myFxn[x_, options___] :=
 Module[{y, w, v, putMoreLocalvariablesHere1, putMoreLocalvariablesHere2,
         putMoreLocalvariablesHere3},
   route = Route /. {options} /. Options[myFxn];
      w =     W /. {options} /. Options[myFxn];
      v = Verbose /. {options} /. Options[myFxn];

   y = If[NumericQ[x],
     Which[route === Oral, 5 + x, route === Ingestion, x/3, True, Undefined]];
   If[TrueQ[v], Print["Route= ", route]; Print["y= ", y]];
   Piecewise[{{x + w, x < 0}, {(11/3) y + w/2, 0 ≤ x < 3}, {y^z + 8, True}}]
  ] /; NumericQ[x] || Head[x] === Symbol
```

```
myFxn[5]
```

$8 + 10^z$

```
myFxn[5, Route → Inhalation, Verbose → True]
```

```
Route= Inhalation
y= Undefined
Undefined
```

```
myFxn[2]
```

$\frac{92}{3}$

```
myFxn[2, Route → Ingestion]
```

$\frac{67}{9}$

Ordinary Differential Equation (ODE) system analysis (exponential loading of, e.g., skin)

Define constants

```
dose = 40;
k = 0.38629;
kAbsDermal = 103 / 1000 ;
kSB = 7 / 200;
kBS = 408 / 125;
kBM = 372 / 125;
kMU = 43 / 200;
kMF = 239 / 10 000;
fMet = {544, 131, 19, 182, 124} / 496;
```

```
t0 = {Dose '[t] == -k * Dose[t],
     R '[t] == k * Dose[t] - kAbs * R[t],
     B '[t] == kAbs * R[t] + kSB * S[t] - (kBM + kBS) * B[t],
     S '[t] == kBS * B[t] - kSB * S[t]
    } /. kAbs → kAbsDermal;
t1 = Table[vM[i] '[t] == fMet[[i]] * kBM * B[t] - (kMU + kMF) * vM[i][t], {i, 5}];
t2 = Table[U[i] '[t] == kMU * vM[i][t], {i, 5}];
t3 = Table[F[i] '[t] == kMF * vM[i][t], {i, 5}];
eqn = Join[t0, t1, t2, t3];
vars = #[[1, 0, 1]][t] & /@ eqn;
eqn0 = Append[Rest[(# == 0) & /@ (vars /. t → 0)], Dose[0] == dose];
eqnAll = Join[eqn, eqn0]
```

$$\Big\{\text{Dose}'[t] == -0.38629\,\text{Dose}[t],\ R'[t] == 0.38629\,\text{Dose}[t] - \frac{103\,R[t]}{1000},$$
$$B'[t] == -\frac{156\,B[t]}{25} + \frac{103\,R[t]}{1000} + \frac{7\,S[t]}{200},\ S'[t] == \frac{408\,B[t]}{125} - \frac{7\,S[t]}{200},$$
$$vM[1]'[t] == \frac{408\,B[t]}{125} - \frac{2389\,vM[1][t]}{10\,000},\ vM[2]'[t] == \frac{393\,B[t]}{500} - \frac{2389\,vM[2][t]}{10\,000},$$
$$vM[3]'[t] == \frac{57\,B[t]}{500} - \frac{2389\,vM[3][t]}{10\,000},\ vM[4]'[t] == \frac{273\,B[t]}{250} - \frac{2389\,vM[4][t]}{10\,000},$$
$$vM[5]'[t] == \frac{93\,B[t]}{125} - \frac{2389\,vM[5][t]}{10\,000},\ U[1]'[t] == \frac{43}{200}\,vM[1][t],$$
$$U[2]'[t] == \frac{43}{200}\,vM[2][t],\ U[3]'[t] == \frac{43}{200}\,vM[3][t],\ U[4]'[t] == \frac{43}{200}\,vM[4][t],$$
$$U[5]'[t] == \frac{43}{200}\,vM[5][t],\ F[1]'[t] == \frac{239\,vM[1][t]}{10\,000},\ F[2]'[t] == \frac{239\,vM[2][t]}{10\,000},$$
$$F[3]'[t] == \frac{239\,vM[3][t]}{10\,000},\ F[4]'[t] == \frac{239\,vM[4][t]}{10\,000},\ F[5]'[t] == \frac{239\,vM[5][t]}{10\,000},$$
$$R[0] == 0,\ B[0] == 0,\ S[0] == 0,\ vM[1][0] == 0,\ vM[2][0] == 0,\ vM[3][0] == 0,\ vM[4][0] == 0,$$
$$vM[5][0] == 0,\ U[1][0] == 0,\ U[2][0] == 0,\ U[3][0] == 0,\ U[4][0] == 0,\ U[5][0] == 0,$$
$$F[1][0] == 0,\ F[2][0] == 0,\ F[3][0] == 0,\ F[4][0] == 0,\ F[5][0] == 0,\ \text{Dose}[0] == 40\Big\}$$

```
Length /@ {eqn, eqn0}
```

```
{19, 19}
```

```
vars
```

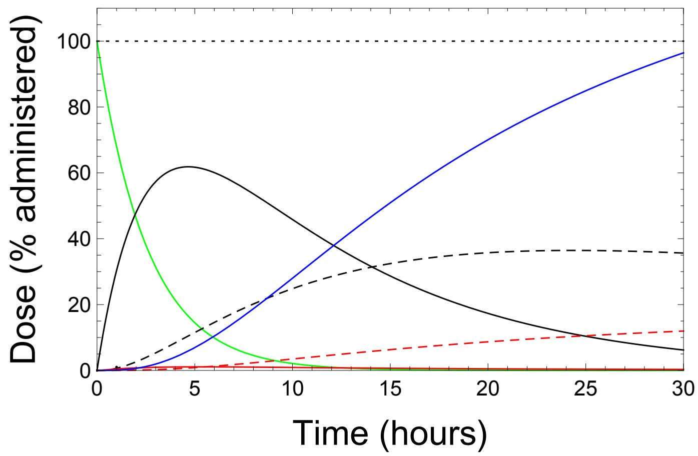
```
{Dose[t], R[t], B[t], S[t], vM[1][t], vM[2][t], vM[3][t], vM[4][t], vM[5][t], U[1][t],
 U[2][t], U[3][t], U[4][t], U[5][t], F[1][t], F[2][t], F[3][t], F[4][t], F[5][t]}
```

```
Mtot[t_] := Plus @@ Table[vM[i][t], {i, 5}];
Utot[t_] := Plus @@ Table[U[i][t], {i, 5}];
Ftot[t_] := Plus @@ Table[F[i][t], {i, 5}];
sum[t_] := Dose[t] + Mtot[t] + Utot[t] + Ftot[t] + R[t] + S[t] + B[t];
tmax = 200;
nsol = NDSolve[eqnAll, vars, {t, 0, tmax}][[1]]
```

```
{Dose[t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 R[t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 B[t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 S[t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 vM[1][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 vM[2][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 vM[3][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 vM[4][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 vM[5][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 U[1][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
 U[2][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],
```

```
U[3][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

U[4][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

U[5][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

F[1][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

F[2][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

F[3][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

F[4][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t],

F[5][t] → InterpolatingFunction[ Domain: {{0., 200.}} Output: scalar ][t]}

Dose[t] /. nsol /. t → 0
40.
```

```
PlotData[Plot, FitTo →
  {100/dose {Dose[t], R[t], S[t], B[t], U[5][t], Utot[t], dose + t * 10^-20} /. nsol, t},
 X → {0, 30}, Y → {0, 110}, Labels → {"Time (hours)", "Dose (% administered)"},
 PlotStyle → {Green, Black, {Dashed, Black}, Red,
   {Dashed, Red}, Blue, {Black, Dotted}}, FontSize → 18]
```



More information

More about *Mathematica*®

More about Mathematica can be found at:

http://reference.wolfram.com/language/

http://www.wolfram.com/mathematica/

More about the *RiskQ* package for *Mathematica*®

Kenneth T. Bogen, DrPH DABT (kbogen@exponent.com)
Exponent Inc., 475 14th Street, Suite 400
Oakland, CA 94612 USA

ity for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights. Reference herein to any specific commercial products, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the U.S. Government, UC or Exponent, Inc.

Bogen KT. RiskQ 4.2: An interactive approach to probability, uncertainty and statistics for use with Mathematica®. UCRL-MA-110232 Rev. 3. Lawrence Livermore National Laboratory, Livermore, CA, 2002.

Wolfram Research. 2017. Wolfram Language and System Documentation Center. Wolfram Research, Inc., Champaign, IL (www.wolfram.com), http://reference.wolfram.com/language/