

SimBiology[®]

Getting Started Guide



MATLAB[®]

R2018b



How to Contact MathWorks



Latest news: www.mathworks.com
Sales and services: www.mathworks.com/sales_and_services
User community: www.mathworks.com/matlabcentral
Technical support: www.mathworks.com/support/contact_us



Phone: 508-647-7000



The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

SimBiology[®] *Getting Started Guide*

© COPYRIGHT 2005–2018 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

September 2005	Online only	New for Version 1.0 (Release 14SP3+)
March 2006	Online only	Updated for Version 1.0.1 (Release 2006a)
May 2006	Online only	Updated for Version 2.0 (Release 2006a+)
September 2006	Online only	Updated for Version 2.0.1 (Release 2006b)
September 2006	Online only	Updated for Version 2.1 (Release 2006b+)
March 2007	Online only	Rereleased for Version 2.1.1 (Release 2007a)
September 2007	Online only	Rereleased for Version 2.1.2 (Release 2007b)
October 2007	Online only	Updated for Version 2.2 (Release 2007b+)
March 2008	Online only	Updated for Version 2.3 (Release 2008a)
October 2008	Online only	Updated for Version 2.4 (Release 2008b)
March 2009	Online only	Updated for Version 3.0 (Release 2009a)
September 2009	Online only	Updated for Version 3.1 (Release 2009b)
March 2010	Online only	Updated for Version 3.2 (Release 2010a)
September 2010	Online only	Updated for Version 3.3 (Release 2010b)
April 2011	Online only	Updated for Version 3.4 (Release 2011a)
September 2011	Online only	Updated for Version 4.0 (Release 2011b)
March 2012	Online only	Updated for Version 4.1 (Release 2012a)
September 2012	Online only	Updated for Version 4.2 (Release 2012b)
March 2013	Online only	Updated for Version 4.3 (Release 2013a)
September 2013	Online only	Updated for Version 4.3.1 (Release 2013b)
March 2014	Online only	Updated for Version 5.0 (Release 2014a)
October 2014	Online only	Updated for Version 5.1 (Release 2014b)
March 2015	Online only	Updated for Version 5.2 (Release 2015a)
September 2015	Online only	Updated for Version 5.3 (Release 2015b)
March 2016	Online only	Updated for Version 5.4 (Release 2016a)
September 2016	Online only	Updated for Version 5.5 (Release 2016b)
March 2017	Online only	Updated for Version 5.6 (Release 2017a)
September 2017	Online only	Updated for Version 5.7 (Release 2017b)
March 2018	Online only	Updated for Version 5.8 (Release 2018a)
September 2018	Online only	Updated for Version 5.8.1 (Release 2018b)

1 Introduction

SimBiology Product Description	1-2
Key Features	1-2
Compiler Setup	1-3
SBML Support	1-4
What Is SBML?	1-4
Importing from SBML Files	1-4
Exporting a SimBiology Model to SBML Format	1-5

2 Getting Started Using the SimBiology Desktop

SimBiology Desktop Overview	2-2
Opening the Desktop	2-2
Main Desktop Window	2-2
Models	2-4
Data	2-5
Tasks	2-5
Estimate Pharmacokinetic Parameters Using SimBiology Desktop	2-8
Model and Data Description	2-8
Next Step	2-8
Import and Explore Data	2-9
Import Data	2-9
Explore Data	2-10
Previous Step	2-11

Next Step	2-11
Create a PK Model	2-12
Previous Step	2-13
Next Step	2-13
Fit Data	2-14
Previous Step	2-18
Create Standalone Applications using SimBiology Desktop .	2-19

Getting Started Using the Command Line

3

Getting Started Using the SimBiology Command Line	3-2
Modeling and Simulation	3-2
Estimation	3-2
Deployment	3-2
Construct a Simple Model	3-3
Model a Gene-Regulation Pathway	3-5
About The Gene Regulation Model	3-5
Create a SimBiology Model	3-9
Add the Reaction for Transcription	3-10
Add the Reaction for Translation	3-11
Add the Reaction for Gene Regulation	3-12
Add the Reactions for mRNA and Protein Degradation	3-12
Simulate the Model	3-13

Introduction

This chapter introduces SimBiology functions and features to help you develop a conceptual model for working with the software and your biochemical data.

- “SimBiology Product Description” on page 1-2
- “Compiler Setup” on page 1-3
- “SBML Support” on page 1-4

SimBiology Product Description

Model, simulate, and analyze biological systems

SimBiology provides an app and programmatic tools to model, simulate, and analyze dynamic systems, focusing on pharmacokinetic/pharmacodynamic (PK/PD) and systems biology applications. It provides a block diagram editor for building models, or you can create models programmatically using the MATLAB® language. SimBiology includes a library of common PK models, which you can customize and integrate with mechanistic systems biology models.

A variety of model exploration techniques let you identify optimal dosing schedules and putative drug targets in cellular pathways. SimBiology uses ordinary differential equations (ODEs) and stochastic solvers to simulate the time course profile of drug exposure, drug efficacy, and enzyme and metabolite levels. You can investigate system dynamics and guide experimentation using parameter sweeps and sensitivity analysis. You can also use single subject or population data to estimate model parameters.

Key Features

- App for PK/PD and mechanistic systems biology modeling
- Ordinary differential equations (ODEs) and stochastic solvers
- Library of PK models
- Parameter estimation techniques for single-subject and population data, including nonlinear mixed-effects models
- Sensitivity analysis and parameter sweeps for investigating parameter effects on system dynamics
- Diagnostic plots for individual and population fits
- Methods for creating and optimizing dosing schedules

Compiler Setup

To prepare SimBiology models for accelerated simulations, install and set up a compiler:

- 1 Install a C compiler (if one is not already installed on your system). For a current list of supported compilers, see [Supported and Compatible Compilers](#).
- 2 Ensure that any user-defined functions in your model can be used for code generation from MATLAB, so they can convert to compiled C. For more information, see [Language, Function, and Object support for C and C++ code generation \(MATLAB Coder\)](#) or contact [MathWorks Technical Support](#).

Tip On 32-bit Windows® platforms, the LCC compiler is automatically installed. However, for better performance of the acceleration functionality, you may want to install a supported compiler other than LCC, and it will be selected automatically.

On 64-bit Windows platforms, if you have not installed another compiler, SimBiology uses the LCC64 compiler for model accelerations. If you have installed another supported compiler, it will be selected automatically.

See Also

More About

- [“Simulation”](#)
- [“Accelerating Model Simulations and Analyses”](#)

SBML Support

In this section...
“What Is SBML?” on page 1-4
“Importing from SBML Files” on page 1-4
“Exporting a SimBiology Model to SBML Format” on page 1-5

What Is SBML?

Systems Biology Markup Language (SBML) is a standard format for sharing systems biology models among various modeling and simulation software tools. The current specification is available at <http://sbml.org/documents/>.

Importing from SBML Files

Import an SBML model from a file or URL using `sbmlimport`.

SimBiology supports a subset of the SBML level 3 version 1 specification. The following SBML features are not imported into the SimBiology model:

- Piecewise kinetics — Models with piecewise kinetics are loaded, but the definitions for piecewise kinetics are ignored.
- MATLAB incompatible variable names in `UnitDefinition` — Models that have variable names incompatible with MATLAB in `UnitDefinition` are not loaded and an error message is displayed.
- The `hasOnlySubstanceUnits` attribute does not have a corresponding property in the SimBiology species object. Instead, if the species has no units, SimBiology uses the `DefaultSpeciesDimension` property to determine whether to interpret species names in expressions as substance amounts or concentrations. When you set `DefaultSpeciesDimension` to `substance` and do not specify units, SimBiology interprets species names as substance amounts and does not scale by any compartment capacity (volume). When the property is set to `concentration`, species names are interpreted as concentrations and are scaled by the appropriate compartment capacity. SimBiology does not let you set the initial value of a species as concentration or substance amount independently of how you refer to it in expressions. To provide better compatibility with SBML models having this attribute, SimBiology adds an initial assignment rule or appropriate units during the import process. For details, see the “Compatibility” section of `sbmlimport`.

- Models containing other models as submodels are loaded but submodels are ignored.
- The `fast` attribute of any reaction object is ignored.
- The `delay`, `priority`, `initialValue`, and `persistent` attributes of event objects are ignored.
- XOR, OR, and AND logical operators having three or more arguments are not supported. They are supported for binary operations only.

Read-Only Support for Function Definitions

You can import SBML models with function definitions. SimBiology replaces the function definitions with the corresponding mathematical expressions.

Support for Reaction IDs

You can also load SBML models that use reaction IDs to reference reaction rates in mathematical expressions. SimBiology replaces the references with the corresponding reaction rates.

Exporting a SimBiology Model to SBML Format

SimBiology Features Supported by SBML

The following SimBiology model information is included in the SBML format file:

- Compartments, species, parameters, reactions, rules, and events that are defined in the model and have their `Active` property set to `true`.
- All unit definitions in SBML-compliant format.
- Model component properties with SBML equivalents, such as notes, and unit values for species and parameters.
- The reaction rate equation, but not the kinetic law definition.

Example

In SimBiology, the Henri-Michaelis-Menten equation, $V_m * S / (K_m + S)$, is a definition stored in the Kinetic Laws library. Specifically, for a one-substrate enzyme-catalyzed reaction, if you assign $V_m = V_a$, $K_m = K_a$, and $S = A$, then the reaction rate equation is exported to SBML as $V_a * S / (K_a + A)$.

SimBiology Features Not Supported by SBML

The following SimBiology features are not supported by SBML and are not included in the saved SBML format file. You can store this information in a SimBiology project file, which has an `.sbproj` extension.

- **Projects** — Models, analysis tasks, and data.
- **Kinetic law information** — SimBiology models store kinetic law information such as the kinetic law name and a kinetic law definition.
- **Variant information** — Collections of quantities (compartments, species, and/or parameters) that you can use to alter a model's initial or base configuration.
- **Dosing information** — Exogenous increments to the amount (or concentration) of a species in a model.
- **Custom MATLAB function files** — Custom functions that you used in your SimBiology model.
- Features and properties specific to SimBiology software, such as **Name** (of Rule objects only), **Tag**, and **Active**.

Tip Because the previous information is not supported by SBML, we recommend saving SimBiology project files to capture this information.

Getting Started Using the SimBiology Desktop

- “SimBiology Desktop Overview” on page 2-2
- “Estimate Pharmacokinetic Parameters Using SimBiology Desktop” on page 2-8
- “Import and Explore Data” on page 2-9
- “Create a PK Model” on page 2-12
- “Fit Data” on page 2-14
- “Create Standalone Applications using SimBiology Desktop” on page 2-19

SimBiology Desktop Overview

The SimBiology desktop is a graphical user interface containing a set of integrated tools that are designed to facilitate building, simulating and analyzing models of dynamic systems with a focus on Systems Biology and Pharmacokinetics. Tasks are graphical interfaces that configure simulation and analysis settings used to investigate model behavior. One such task is parameter estimation that calibrates a model against imported data. Tasks produce results that can be further analyzed within the desktop. The models, tasks, imported data and task results are saved in a SimBiology project.

SimBiology also provides a set of libraries. These libraries provide built-in elements for defining kinetic laws, graphical blocks, and units that can be used when building a model. A library of plots provides commonly used visualizations for task results. These libraries can be extended with custom elements.

Opening the Desktop

Open the desktop by typing the following at the MATLAB command line.

```
simbiology
```

Alternatively, select **SimBiology** from the **Apps** tab.

Main Desktop Window

The main SimBiology desktop window provides tools for building models, importing and exploring data, visualizing task results, and extending SimBiology libraries. Tasks are defined and executed using the Task Editor.

Annotations on the left side of the image:

- Home Tab
- Toolstrip
- Content Button
- Info Button
- Action Button
- Address Bar

SimBiology Desktop Interface Overview:

The interface is divided into several sections:

- Home Tab:** Contains the toolstrip and content pane.
- Toolstrip:** Includes buttons for New, Save, Open, Close, Add Model, Add Data, Add Task, Layout, Preferences, Community, and Help.
- Content Pane:** Shows a tree view with 'Project' and 'lotka'. The 'lotka' folder is expanded, showing a table of reactions.
- Reactions Table:**

Type	Name	Active	Expression
1 reaction	Reaction1	<input checked="" type="checkbox"/>	$x + y1 \rightarrow 2 y1 + x, c1*y1*x$
2 reaction	Reaction2	<input checked="" type="checkbox"/>	$y1 + y2 \rightarrow 2 y2, c2*y1*y2$
3 reaction	Reaction3	<input checked="" type="checkbox"/>	$y2 \rightarrow z, c3*y2$
- Add Quantity Section:**

Add Quantity: species

Type	Scope	Name	Value	Units	Constant
1 compart...	lotka	unnamed	1.0		<input checked="" type="checkbox"/>
2 species	unnamed	x	1.0		<input type="checkbox"/>
3 species	unnamed	y1	900.0		<input type="checkbox"/>
4 species	unnamed	y2	900.0		<input type="checkbox"/>
5 species	unnamed	z	0.0		<input type="checkbox"/>
6 parameter	$x + y1 \rightarrow 2 y1 + x$	c1	10.0		<input checked="" type="checkbox"/>
7 parameter	$y1 + y2 \rightarrow 2 y2$	c2	0.01		<input checked="" type="checkbox"/>
8 parameter	$y2 \rightarrow z$	c3	10.0		<input checked="" type="checkbox"/>

Home Tab	Allows you to open and save projects, add models, tasks, and data to a project.
Toolstrip	Displays the tabs specific to the open panel.
Content button	Allows you to quickly access the project files, built-in libraries, and recent files.
Info Button	Shows you the context sensitive help when you hover over it.
Action Button	Contains additional functionality related to the open panel.
Address Bar	Shows the name of the panel that is open.

Additional Tools

You can access additional tools such as the MATLAB Code Capture Tool by selecting **Home > Layout > MATLAB Code Capture Tool**.

Models

A SimBiology model is composed of species, compartments, parameters, reactions, rules, and events. Species, compartments, and parameters are categorized as model quantities. Reactions, rules, and events are categorized as model expressions. The dynamics of the model are governed by these expressions.

A SimBiology model can also have modifiers associated with it. There are two types of modifiers, variants and doses. Variants allow you to store alternative quantity values that can be applied to a model during analysis. Doses allow you to increase the amount of a species during a simulation.

Adding a Model

Select **Home > Add Model** to add a model to a project. You can create a new model, load a model from a SimBiology project, or import a SBML file. In addition, you can add commonly used PK models from the PK library.

When a model is open in the main desktop window, the toolstrip will add tabs for working with the model. One of those tabs is the **Model** tab. It contains buttons for editing the model in tabular or diagram form, and creating tasks.

Data

Data can be imported for use with tasks such as parameter fitting. Once the data is imported you can visualize and perform preliminary data processing.

Adding Data

To add data, select **Home > Add Data**. You can import data from:

- CVS formatted files
- Microsoft® Excel® files
- SimBiology project files
- MATLAB MAT files
- SAS® XPORT formatted files
- The MATLAB workspace

When data is open in the main desktop window, you can classify data columns and assign units. The toolstrip will add tabs for working with the data. One of those tabs is the **Define Plot** tab which is used to create plots. Another tab is the **Explore Data** tab. It allows you to:

- Exclude data rows – You can exclude data rows manually or using MATLAB expressions. Excluded data is ignored when running tasks.
- Add new data columns – You can add new data columns by using MATLAB expressions on other existing data columns. You can use the new columns when running tasks.

Tasks

Tasks are analyses that can be performed on a model. There are several built-in tasks such as simulation, estimation, and sensitivity analysis. In addition, custom tasks can be created using the MATLAB language.

Adding a Task

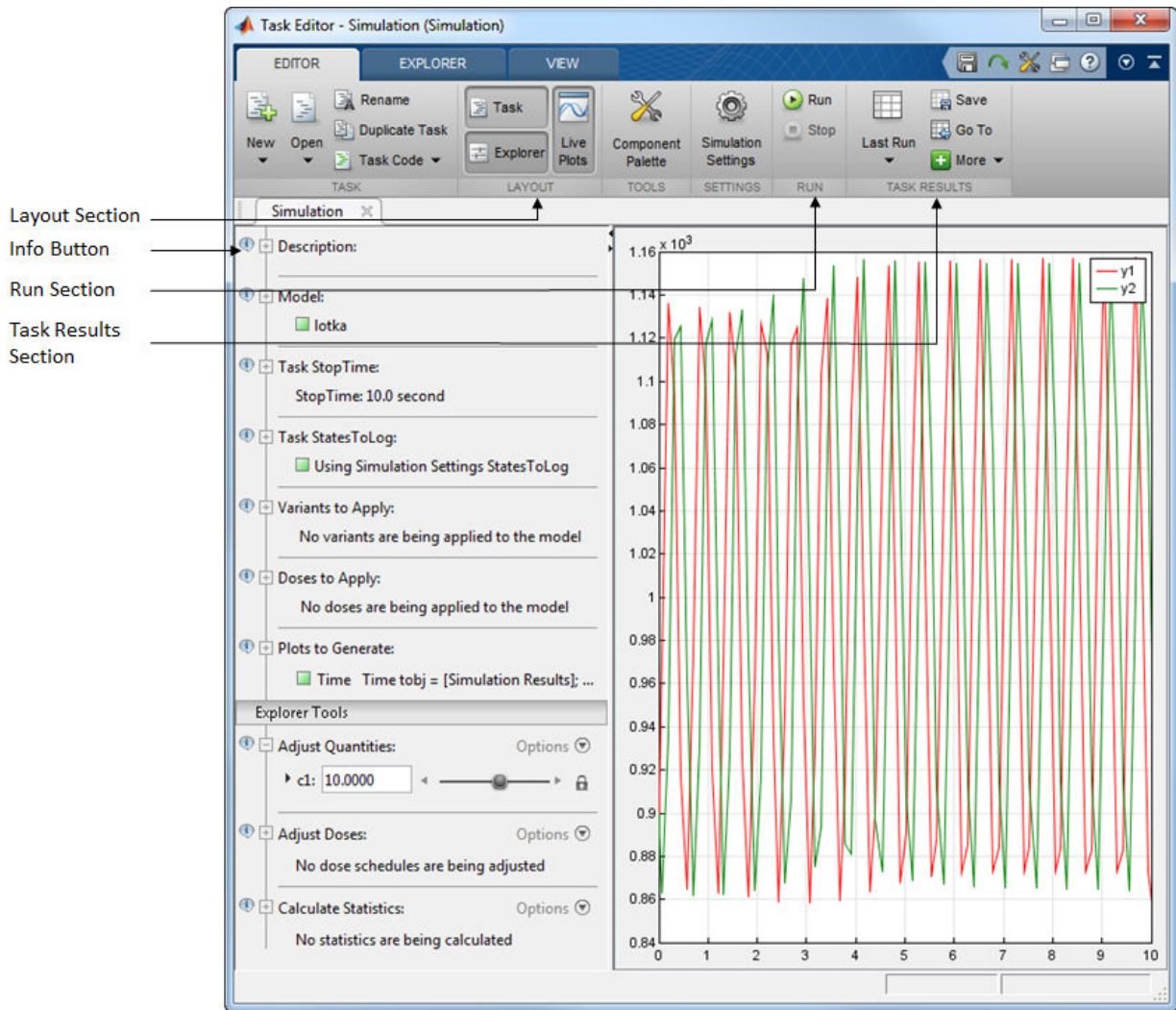
Select **Home > Add Task** to add a task to a project. Alternatively, you can select **Model > Add Task** when a model is open in the main desktop window.

A task is opened in a separate window called the **Task Editor**. The editor consists of three sections: general task setup, interactive task setup, and live visualization of results.

2 Getting Started Using the SimBiology Desktop

The general setup is used to define settings such as simulation stop time, and variants and doses applied during task execution. The interactive task setup provides sliders for exploring model behavior. The sliders and live visualization plots can be configured using the tools on the **Explorer** tab.

For example, a simulation task appears as follows.



Layout Section	Allows you to configure the open sections of the Task Editor.
Info Button	Shows you the context sensitive help when you hover over it.
Run Section	Press the Run button to execute the task.
Task Results Section	Allows you to manage results generated from executing the task.

Task Results

Task results generated at the completion of a task run are stored as **Last Run**. These results are overwritten every time the task is executed. To avoid overwriting a particular set of results, use the **Save** button in the **Task Results** section.

A list of all saved results can be accessed from the **Task Results** section. The **Go To** button will open the selected results. When task results are open in the main desktop window, the toolstrip will add tabs for working with the results. One of those tabs is the **Define Plot** tab. There you can visualize results, create additional plots, and export results to the MATLAB workspace.

See Also

Related Examples

- “Create and Simulate a Simple Model”
- “Estimate Pharmacokinetic Parameters Using SimBiology Desktop” on page 2-8

Estimate Pharmacokinetic Parameters Using SimBiology Desktop

Model and Data Description

This example shows how to estimate one-compartment pharmacokinetic (PK) parameters given experimental drug concentration data. Suppose you have measurements of drug plasma concentration from four individuals after bolus doses and want to estimate corresponding PK parameters, namely the volume of the central compartment (V) and clearance rate (Cl) of each individual. For one-compartment PK models, the drug

concentration follows an exponential decline $C_t = C_0 e^{-k_e t}$. C_t is the drug concentration at time t . C_0 is the initial concentration. k_e is the elimination rate constant that depends on

the clearance and volume of the central compartment $k_e = \frac{Cl}{V}$.

The synthetic data in this example contains measurements of drug plasma concentration of four individuals after they receive the bolus doses. The data contains the variables: ID, Time, DrugConc, and Dose. ID is the grouping variable which lists the ID of each individual, Time is the time at which each measurement was performed, DrugConc is the drug plasma concentration, and Dose is the dosing information for each individual. This data is saved as `pkProfiles_1Comp.mat` in `matlabroot\help\toolbox\simbio\examples`. `matlabroot` is the folder where you have installed MATLAB.

Estimating PK parameters using a one-, two-, or multi-compartment model in the SimBiology desktop requires the data to have at least three variables (or columns):

- Independent variable (such as time)
- Dependent variable (such as measured drug concentration at each time)
- Dosing variable (such as dose amount at each dose time)

Optionally, you can also have a grouping variable such as group or patient ID and one or more dependent and dosing variables.

Next Step

“Import and Explore Data” on page 2-9

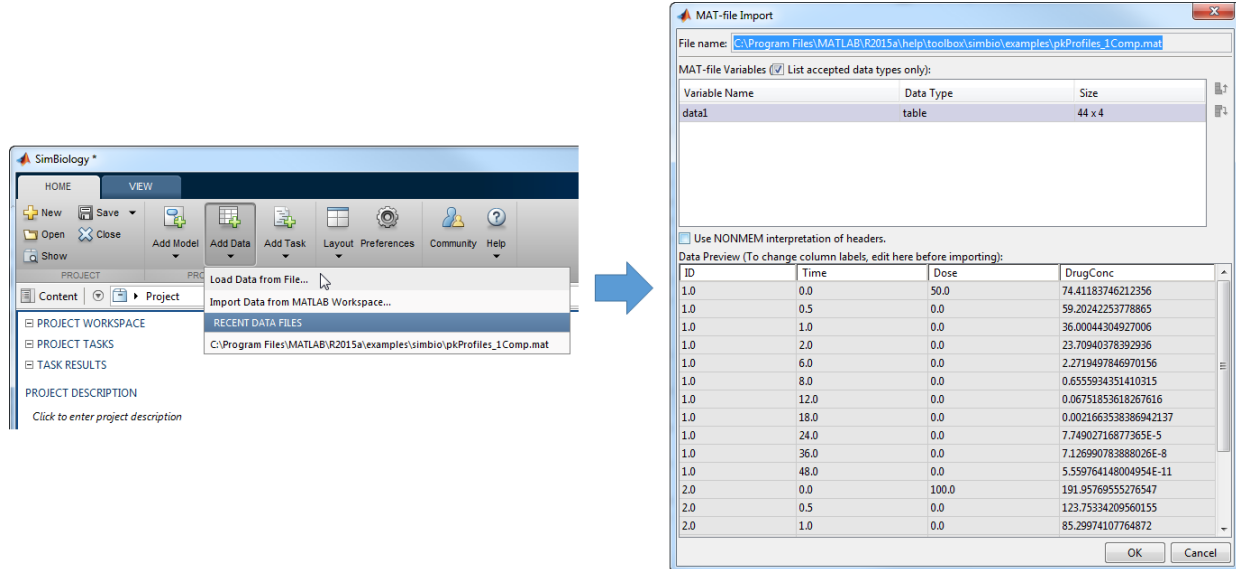
Import and Explore Data

Import Data

Open SimBiology desktop by typing `simbiology` in the MATLAB Command Window.

On the **Home** tab, select **Add Data > Load Data from File**.

Navigate to the folder `matlabroot\help\toolbox\simbio\examples.matlabroot` is the folder where you have installed MATLAB. Select the sample file named `pkProfiles_1Comp.mat`. Click OK.



The screenshot shows the SimBiology interface with the 'Add Data' menu open, highlighting 'Load Data from File...'. A blue arrow points to the 'MAT-file Import' dialog box. The dialog box displays the file name and a table of variables. Below the table, there is a 'Data Preview' section showing a table of data points.

Variable Name	Data Type	Size
data1	table	44 x 4

ID	Time	Dose	DrugConc
1.0	0.0	50.0	74.41183746212356
1.0	0.5	0.0	59.20242253778865
1.0	1.0	0.0	36.00044304927006
1.0	2.0	0.0	23.70940378392936
1.0	6.0	0.0	2.2719497846970156
1.0	8.0	0.0	0.6555934351410315
1.0	12.0	0.0	0.06751853618267616
1.0	18.0	0.0	0.0021663538386942137
1.0	24.0	0.0	7.74902716877365E-5
1.0	36.0	0.0	7.126990783888026E-8
1.0	48.0	0.0	5.559764148004954E-11
2.0	0.0	100.0	191.95769555276547
2.0	0.5	0.0	123.75334209560155
2.0	1.0	0.0	85.29974107764872

Based on the names of variables (columns), SimBiology categorizes the variables. In this example, SimBiology is able to categorize the data into group variable, independent variable, and dose variable as shown under each data column in the table. However, **DrugConc** is not categorized as a dependent variable. From the drop-down list under **DrugConc**, select `dependent variable1`. Note that for different data sets, you may have to manually categorize one or more data variables as appropriate.

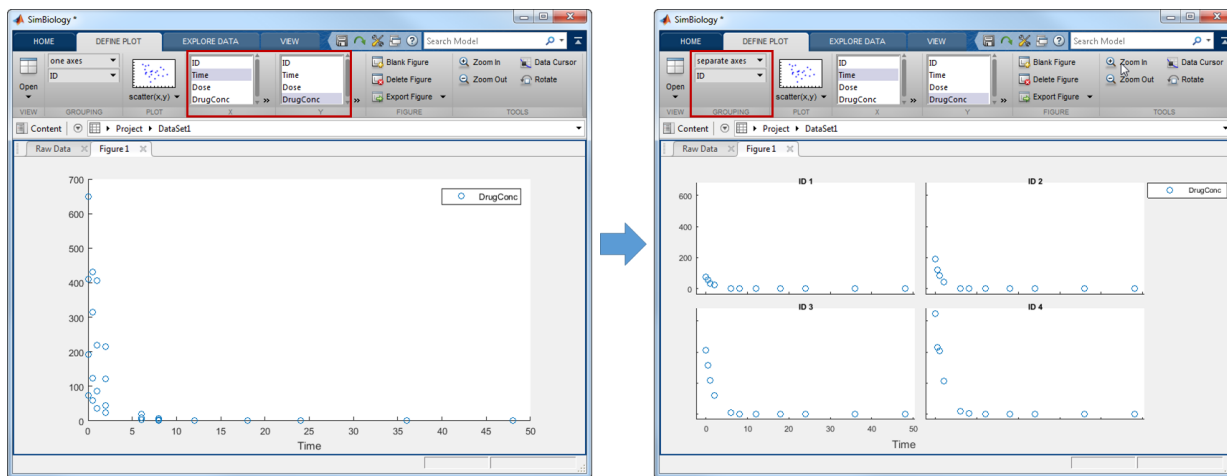
The screenshot shows the SimBiology Desktop interface. The 'EXPLORE DATA' tab is selected. In the 'DEFINE PLOT' section, the 'one axes' plot type is chosen. The X-axis is set to 'Time' and the Y-axis is set to 'DrugConc'. The data table below shows the following information:

Units	ID	Time	Dose	DrugConc
	group variable	independent variable	dose1	dependent variable1
1	1.0	0.0	50.0	74.41183746212356
2	1.0	0.5	0.0	59.20242253778865
3	1.0	1.0	0.0	36.00044304927006
4	1.0	2.0	0.0	23.70940378392936
5	1.0	6.0	0.0	2.2719497846970156
6	1.0	8.0	0.0	0.6555934351410315
7	1.0	12.0	0.0	0.06751853618267616
8	1.0	18.0	0.0	0.0021663538386942137
9	1.0	24.0	0.0	7.74902716877365E-5
10	1.0	36.0	0.0	7.126990783888026E-8
11	1.0	48.0	0.0	5.559764148004954E-11
12	2.0	0.0	100.0	191.95769555276547
13	2.0	0.5	0.0	123.75334209560155
14	2.0	1.0	0.0	85.29974107764872
15	2.0	2.0	0.0	45.50289121876351
16	2.0	6.0	0.0	3.0213224455534813
17	2.0	8.0	0.0	0.799663643095197
18	2.0	12.0	0.0	0.0600749202694836

Explore Data

After you import data, you can explore it by using the plotting tools available directly from SimBiology. From the **Define Plot** tab, select **Time** as the variable for the X-axis and **DrugConc** for the Y-axis. By default, SimBiology plots data from all individuals using one axes.

To plot each individual data on separate axes, select **separate axes** in the **Grouping** section.



To estimate PK parameters using this data, you must create a PK model on page 2-12.

Previous Step

“Model and Data Description” on page 2-8

Next Step

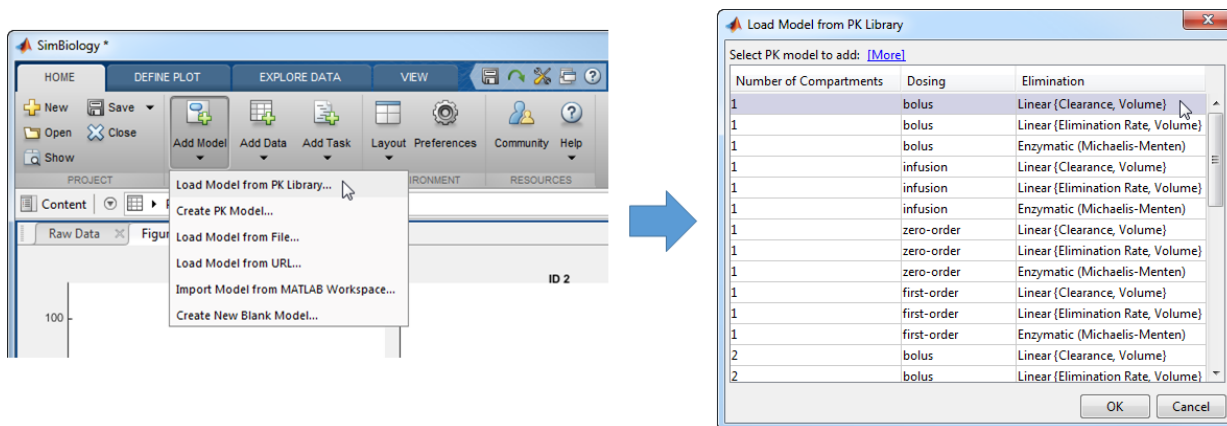
“Create a PK Model” on page 2-12

Create a PK Model

The easiest way to create a one-compartment model is to use the built-in PK library. From the **Home** tab, select **Add Model > Load Model from PK Library**. A list of model options is provided that allows you to pick features of compartmental PK models such as the number of compartments, dosing types, and elimination types.

Select the one-compartment model with the **bolus** dosing and **Linear {Clearance, Volume}** elimination option.

Note This elimination option defines the drug elimination from the body in terms of the clearance rate (Cl), $Cl = k_e * \text{Central}$. k_e is the elimination rate constant, and Central is the volume of central compartment. For details, see Elimination Types.



SimBiology builds a one-compartment model. By default, the model is displayed in the Diagram view. For other views of the model, select **Open** on the **Model** tab.

After you create a model, you can fit the model to the data.

Create a PK Model Using Wizard (Alternative Approach)

The PK wizard option provides you with more flexibility in configuring the model. Select **Add Model > Create PK Model**. You can select the number of compartments, dosing and elimination types, and specify whether a particular compartment has a response

variable associated with it. In the context of fitting, if your experimental data has the response data measured for that compartment, select the **Response** check box.

Previous Step

“Import and Explore Data” on page 2-9

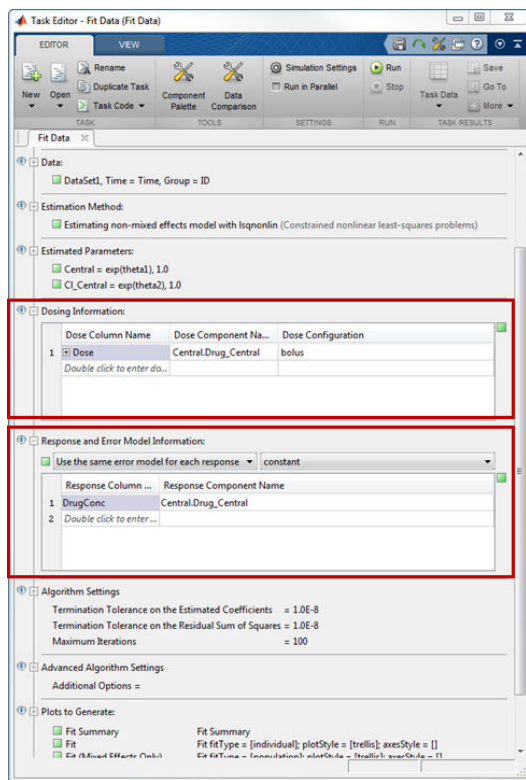
Next Step

“Fit Data” on page 2-14

Fit Data

You can estimate parameters of a model using a fit task. From the **Home** tab, select **Add Task > Fit data**.

When using a model that you create using the PK library or wizard, you can verify the automatic mapping between the model species and data variables by expanding the **Dosing Information** and **Response and Error Model Information** sections.



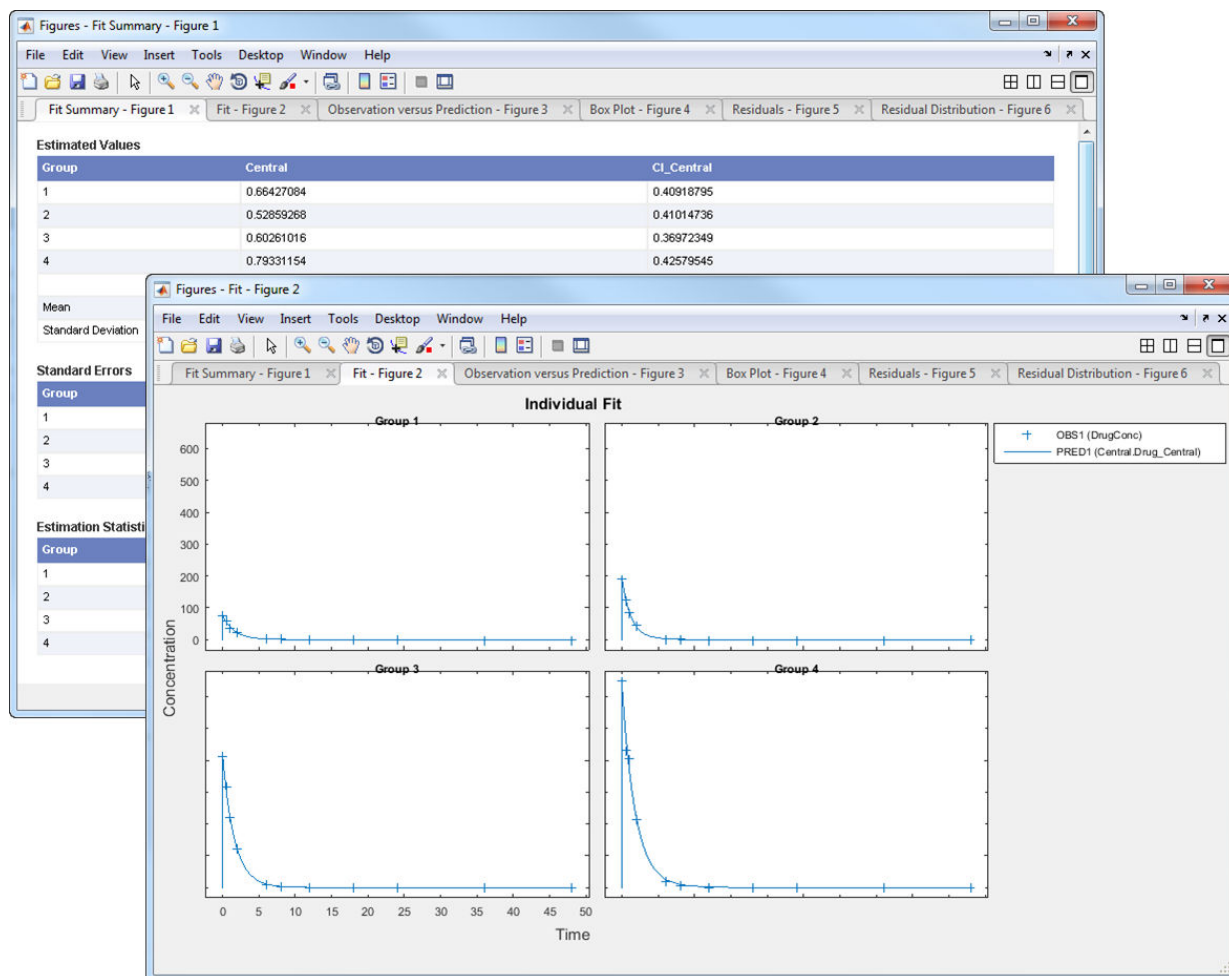
SimBiology has mapped the `Central.Drug_Central` species to the appropriate dosing variable `Dose` from `DataSet1`.



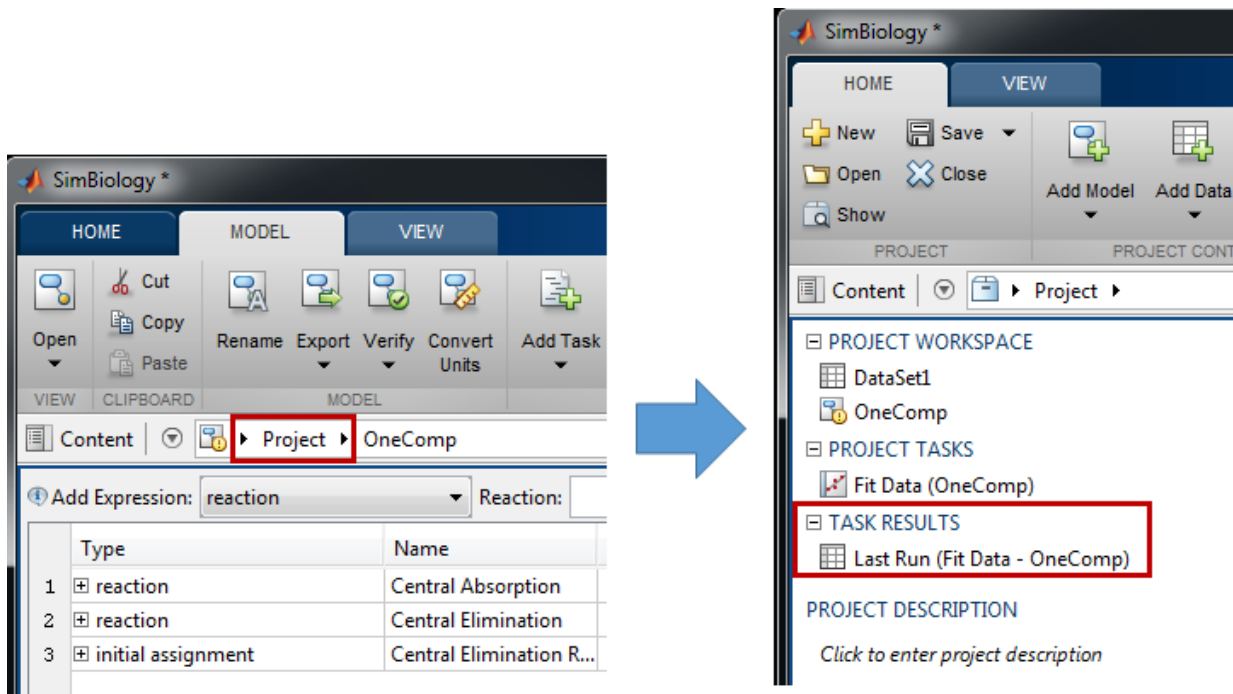
SimBiology has mapped the `Central.Drug_Central` species to the appropriate response variable `DrugConc` from `DataSet1`.

Click **Run** to estimate parameters.

The task generates six figures related to the estimated results by default. If you want different figures, before you run the task, you can specify which plots to generate in the **Plots to Generate** section of the task.



SimBiology stores the results from the fit task in the **Last Run** entry under the **TASK RESULTS** section of the Project. You can navigate to the project area by clicking the **Project** on the navigation bar of the main desktop window.



Each time you run the same fit task, this **Last Run** entry is overwritten. If you plan to run the task again and want to keep the results of the last run, save the data. Right-click on the **Last Run** entry and select **Save Data** and specify a name for the data. When you rerun the task, the results are stored in the **Last Run** entry, and your previously saved data is unchanged. Note that each task has its own **Last Run** entry.

Automatic Mapping Between Data and Model

To estimate PK parameters in your model, SimBiology requires you to map your data to the model. If you construct a model using the PK library or wizard, SimBiology recognizes which model species correspond to which variables (columns) in the data set that you have imported.

In this example, since you specify a bolus dosing, the dose target is the `Drug_Central` species, which is also the measured response. Hence SimBiology maps the `Drug_Central` species to both `Dose` and `DrugConc` columns. The following diagram illustrates such mapping.

The screenshot shows the SimBiology software interface. The top menu bar includes HOME, MODEL, DIAGRAM, BLOCK, and VIEW. Below the menu is a toolbar with icons for Open, Rename, Export, Verify, Convert Units, Task Tool, Diagram, View, Tools, Select Variants, Search, Show Usages, Change Scope, and Add Task. The main workspace displays a compartmental model diagram with a central compartment (Central) and two other compartments (Dose_Central and Drug_Central). The diagram shows a flow from Dose_Central to Drug_Central, and from Drug_Central to Central. A parameter ka_Central is also shown. The left sidebar contains a list of model elements: compartment, parameter, species, reaction, initial assignment, repeated assignment, rate rule, algebraic rule, event, repeat dose, schedule dose, and variant. The central panel shows the model's parameters and their values:

Model: OneComp

- Central = 1.0
- Dose_Central = 0.0
- Drug_Central = 0.0
- Model Scoped Parameters
 - Cl_Central = 1.0
 - ka_Central = 0.0
 - ke_Central = 1.0
 - Tk0_Central = 1.0
 - Tlag_Central = 1.0
- Reaction Scoped Parameters

The diagram contains hidden blocks. To see the list select the Hidden Blocks button on the Block tab.

DataSet1

	ID	Time	Dose	DrugConc
Units	group variable	independent variable	dose1	dependent variable1
1	1.0	0.0	50.0	74.41183746212356
2	1.0	0.5	0.0	59.20242253778865
3	1.0	1.0	0.0	36.00044304927006
4	1.0	2.0	0.0	23.70940378392936
5	1.0	6.0	0.0	2.2719497846970156
6	1.0	8.0	0.0	0.6555934351410315
7	1.0	12.0	0.0	0.06751853618267616
8	1.0	18.0	0.0	0.0021663538386942137
9	1.0	24.0	0.0	7.74902716877365E-5
10	1.0	36.0	0.0	7.126990783888026E-8
11	1.0	48.0	0.0	5.559764148004954E-11
12	2.0	0.0	100.0	191.95769555276547
13	2.0	0.5	0.0	123.75334209560155
14	2.0	1.0	0.0	85.29974107764872
15	2.0	2.0	0.0	45.50289121876351
16	2.0	6.0	0.0	3.0213224455534813
17	2.0	8.0	0.0	0.799663643095197
18	2.0	12.0	0.0	0.0600749202694836

For the first order dose, the dose target is `Dose_Central`, instead of `Drug_Central`. For details, see dosing types.

Manually Mapping Data to a Custom Model

If you are using a custom model you created without using the wizard or PK library, you must manually do the mapping in the **Dosing Information** and **Response and Error Model Information** sections before you run the fit task.

Create a Variant Using Estimated Parameter Values

If you plan to use these estimated parameters later in your modeling workflow, you can optionally create a variant of the model using the mean values of parameter estimates. Open the **Last Run** or any saved result entry by double-clicking it. Go to the **Summary** tab, and scroll down to the **Estimated Values** section. Click **Create Variant from Mean Values**. When prompted, enter a name of the variant. This allows you to save the estimated parameter values without changing the original model, and reuse these values. For instance, you can apply these variant parameter values during simulation by specifying the variant in the **Variants to Apply** section of a simulation task.

Previous Step

“Create a PK Model” on page 2-12

See Also

Related Examples

- “Create and Simulate a Simple Model”

More About

- “SimBiology Desktop”
- “Nonlinear Regression”

Create Standalone Applications using SimBiology Desktop

Create standalone applications for model distribution and simulation.

You can create standalone applications for model distribution and simulation from the SimBiology desktop. The application can be an executable that runs outside of MATLAB or a MATLAB app. To deploy a standalone application, you need MATLAB Compiler™.

To build an executable for model simulation, first add a simulation task and open it. Then in the **Task Editor**, on the **Editor** tab, select **Create App > Deploy**. For a MATLAB app, select **Create App > Package as MATLAB App**.

You can include model exploration tools in the application, such as sliders to adjust quantity values and dose schedules, as well as calculated statistics. To test the user interface and see the final deployed application or MATLAB app, select **Create App > Launch**.

To run standalone applications, install the MATLAB Runtime in a target computer. For details, see “Install and Configure the MATLAB Runtime” (MATLAB Compiler). Standalone applications run on Windows, Linux®, and Mac.

Getting Started Using the Command Line

- “Getting Started Using the SimBiology Command Line” on page 3-2
- “Construct a Simple Model” on page 3-3
- “Model a Gene-Regulation Pathway” on page 3-5

Getting Started Using the SimBiology Command Line

Use the MATLAB command line to programmatically write and save scripts for batch processing, and to automate the model building and analysis workflow.

See the following tutorials to get started.

Modeling and Simulation

- “Construct a Simple Model” on page 3-3
- “Model a Gene-Regulation Pathway” on page 3-5
- “Simulate the Yeast Heterotrimeric G Protein Cycle”
- “Simulate Biological Variability of the Yeast G Protein Cycle Using the Wild-Type and Mutant Strains”
- “Create and Simulate a Model with a Custom Function”
- “Calculate Sensitivities”

Estimation

- “Fit a One-Compartment Model to an Individual's PK Profile”
- “Fit a Two-Compartment Model to PK Profiles of Multiple Individuals”
- “Estimate Category-Specific PK Parameters for Multiple Individuals”
- “Estimate a Parameter from the Yeast G Protein Model”
- “PK/PD Modeling and Simulation to Guide Dosing Strategy for Antibiotics”
- “Modeling the Population Pharmacokinetics of Phenobarbital in Neonates”

Deployment

- “Deploy a SimBiology Model”

Construct a Simple Model

This example shows you how to construct a simple model with two species (A and B) and a reaction. The reaction is $A \rightarrow B$, which follows the mass action kinetics with the forward rate parameter k . Hence the rate of change is $dA/dt = -k * A$.

Create a SimBiology model named `simpleModel`.

```
m1 = sbiomodel('simpleModel');
```

Add a reaction that involves two species A and B, where A is converted to B.

```
r1 = addreaction(m1, 'A -> B');
```

SimBiology automatically add species A and B to the model.

```
m1.species
```

```
SimBiology Species Array
```

Index:	Compartment:	Name:	InitialAmount:	InitialAmountUnits:
1	unnamed	A	0	
2	unnamed	B	0	

Set the initial amount of the first species (A) to 10.

```
m1.species(1).InitialAmount = 10;
```

Define the kinetic law of the reaction to follow the mass action kinetics. You can achieve this by adding a kinetic law object to the reaction `r1`.

```
kineticLaw = addkineticlaw(r1, 'MassAction');
```

Add a rate constant parameter to the mass action kinetic law. You must set the `ParameterVariableNames` property of the kinetic law object to the name of the parameter 'k1' so that the reaction rate can be determined.

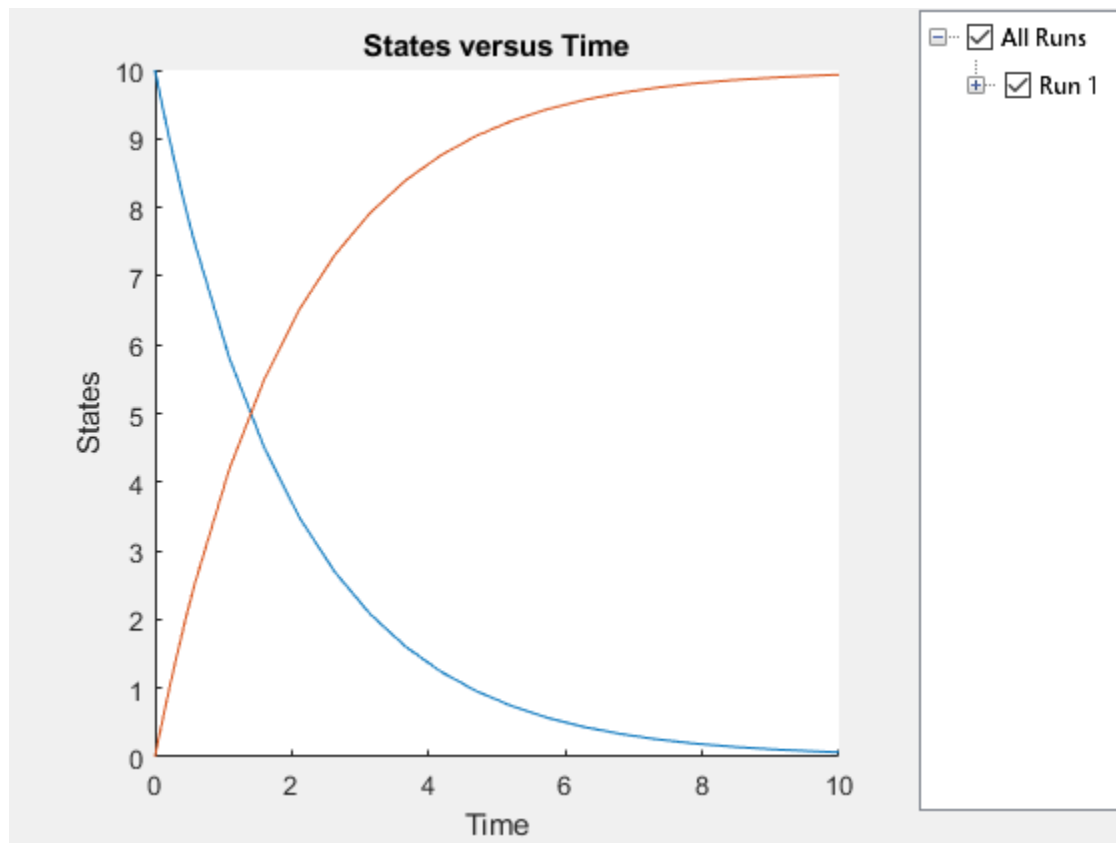
```
p1 = addparameter(kineticLaw, 'k', 0.5);
kineticLaw.ParameterVariableNames = 'k';
```

Simulate the model.

```
sd = sbiosimulate(m1);
```

Plot the simulation results.

```
sbioplot(sd);
```



Model a Gene-Regulation Pathway

In this section...

“About The Gene Regulation Model” on page 3-5

“Create a SimBiology Model” on page 3-9

“Add the Reaction for Transcription” on page 3-10

“Add the Reaction for Translation” on page 3-11

“Add the Reaction for Gene Regulation” on page 3-12

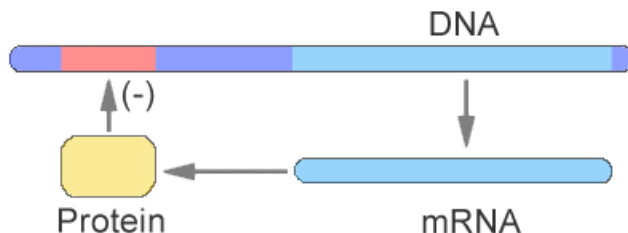
“Add the Reactions for mRNA and Protein Degradation” on page 3-12

“Simulate the Model” on page 3-13

About The Gene Regulation Model

Model Diagram

You can visualize a systems biology model with various levels of detail. One view sketches only the major species and processes. This model is an example of simple gene regulation, where the protein product from translation controls transcription. You could create a more complex model by adding the enzymes, coenzymes, cofactors, nucleotides, and amino acids that are not included in this model. The gene regulation example simplifies the regulatory mechanism by not showing the contributions of RNA polymerase and any cofactors. The protein product from gene expression binds to a regulatory region on the DNA and represses transcription.

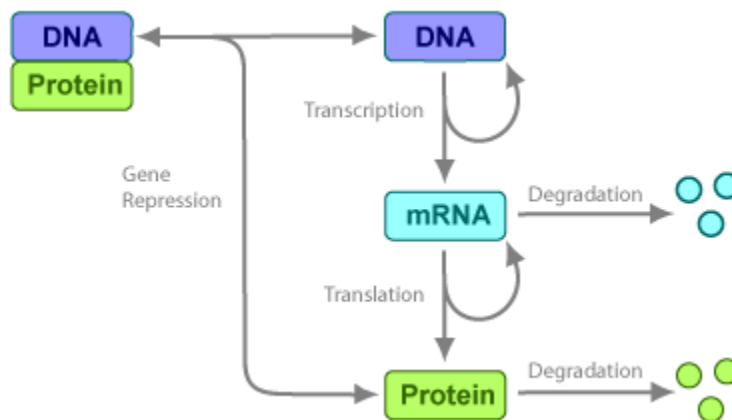


Another way of looking at a systems biology model is to list the reactions in a model with the processes they represent.



```
DNA + protein -> DNAProteinComplex    (binding)
DNAProteinComplex -> DNA + protein    (unbinding)
mRNA -> null                          (degradation)
protein -> null                        (degradation)
```

Drawing the reaction pathways will help you visualize the relationships between reactions and species. In the gene regulation example, as the amount of a protein increases, the protein forms a complex with the gene responsible for its expression, and slows down protein production.



Model Reactions

Reaction equations define a systems biology model at the level of detail needed for a software program to simulate the dynamic behavior of the model. The following reactions for transcription, translation, binding, and degradation describe a simple gene-regulating mechanism.

Transcription

Transcription is where RNAPolymerase and cofactors bind with a DNA molecule. The RNAPolymerase then moves along the DNA and combines nucleotides to create mRNA. A simple model of transcription shows only the DNA and mRNA.



This model simplifies the transcription and the synthesis of mRNA by using a single reaction.

Reaction	DNA \rightarrow DNA + mRNA
Reaction rate	$v = k_1 * \text{DNA molecule/second}$
Species	DNA = 50 molecule mRNA = 0 molecule
Parameters	$k_1 = 0.20 \text{ second}^{-1}$

Translation

After the mRNA moves from the nucleus to the cytoplasm, it can bind with ribosomes. The ribosomes move along the mRNA and create proteins with the help of tRNAs bound to amino acids. A simple model of translation shows only the mRNA and protein product.



The synthesis of the protein is modeled as a single reaction.

Reaction	mRNA \rightarrow mRNA + protein
Reaction rate	$v = k_2 * \text{mRNA molecule/second}$
Species	mRNA = 0 molecule protein = 0 molecule
Parameters	$k_2 = 20 \text{ second}^{-1}$

Gene Repression

Transcription of DNA to mRNA is regulated by the binding of the protein product from translation to the DNA. As more protein is produced, the DNA is bound with the protein more often and less time is available for transcription with the unbound DNA.



Forward Reaction (Binding)

Reaction DNA + protein -> DNAProteinComplex
Reaction rate $v = k_3 * \text{DNA} * \text{protein}$ molecule/second
Species DNA = 50 molecule
 protein = 0 molecule
Parameters $k_3 = 0.2$ 1/(molecule*second)

Reverse Reaction (Unbinding)

Reaction DNAProteinComplex -> DNA + protein
Reaction rate $v = k_{3r} * \text{DNA_protein}$ molecule/second
Species DNAProteinComplex = 0 molecule
Parameters $k_{3r} = 1$ second⁻¹

For this tutorial, model the binding and unbinding reactions as one reversible reaction.

Reaction DNA + protein <-> DNA_protein
Reaction rate $v = k_3 * \text{DNA} * \text{protein} - k_{3r} * \text{DNA_protein}$ molecule/
 second
Species DNA = 50 molecule
 protein = 0 molecule
Parameters $k_3 = 0.2$ 1/(second*molecule)
 $k_{3r} = 1$ second⁻¹

Degradation

Protein and mRNA degradation are important reactions for regulating gene expression. The steady-state level of these compounds is maintained by a balance between synthesis and degradation reactions. Proteins are hydrolyzed to amino acids with the help of proteases, and nucleic acids are degraded to nucleotides.



mRNA degradation is modeled as a transformation to the null species.

Reaction mRNA -> null
 Reaction rate $v = k4 * \text{mRNA molecule/second}$
 Species mRNA = 0 molecule
 Parameters $k4 = 1.5 \text{ second}^{-1}$

Likewise, protein degradation is also modeled as a transformation to the null species. The species null is a predefined species name in SimBiology models.



Reaction protein -> null
 Reaction rate $v = k5 * \text{protein molecule/second}$
 Species protein = 0 molecule
 Parameters $k5 = 1 \text{ second}^{-1}$

Create a SimBiology Model

A SimBiology model is a collection of objects that are structured hierarchically. A model object is needed to contain all the other objects.

- 1 Create a SimBiology model with the name cell.

```
clear all
Mobj = sbiomodel('cell')

SimBiology Model - cell

Model Components:
  Compartments:      0
  Events:            0
  Parameters:        0
  Reactions:         0
  Rules:             0
  Species:           0
```

- 2 Add a compartment named comp to the model and set the unit of compartment capacity.

```
compObj = addcompartment(Mobj, 'comp');
compObj.CapacityUnits = 'liter';
```

Add the Reaction for Transcription

- 1 Add the reaction `DNA -> DNA + mRNA` to the model. SimBiology automatically adds the species `DNA` and `mRNA` to the model with the default initial amount of 0.

```
Robj1 = addreaction(Mobj, 'DNA -> DNA + mRNA');
```

Note Because this example has only one compartment, you need not specify the compartment to which each species belongs. If there are multiple compartments, here is an example of the reaction syntax:

```
Robj1 = addreaction(Mobj, 'nucleus.DNA -> nucleus.DNA + cytoplasm.mRNA');
```

`nucleus` and `cytoplasm` are the names of the compartments.

- 2 Display the added species of the model.

```
Mobj.Species
```

```
SimBiology Species Array
```

Index:	Compartment:	Name:	InitialAmount:	InitialAmountUnits:
1	comp	DNA	0	
2	comp	mRNA	0	

- 3 Set the initial amount of `DNA` to 50 as well the amount units for both species.

```
Mobj.Species(1).InitialAmount = 50;  
Mobj.Species(1).InitialAmountUnits = 'molecule';  
Mobj.Species(2).InitialAmountUnits = 'molecule';
```

- 4 Specify the kinetics of the reaction to be mass action by creating a mass action kinetic law object, `Kobj1`.

```
Kobj1 = addkineticlaw(Robj1, 'MassAction');
```

For a nonreversible reaction, `MassAction` kinetics defines the reaction rate expression as *forward rate constant * reactants*.

- 5 The kinetic law serves as a map between parameters and species needed by the reaction rate expression and parameters and species in the model. To see the parameters and species that must be mapped, retrieve the `ParameterVariables` and `SpeciesVariables` properties of `Kobj1`.

```
Kobj1.ParameterVariables
```

```
ans = 1x1 cell array
      {'Forward Rate Parameter'}
```

Kobj1.SpeciesVariables

```
ans = 1x1 cell array
      {'MassAction Species'}
```

- 6** Since the kinetic law requires a forward rate parameter, create a parameter, `k1`, and set its value to `0.2`. Map the parameter `k1` to the forward rate parameter, by setting the `ParameterVariableNames` property of `Kobj1` to `k1`.

```
Pobj1 = addparameter(Kobj1, 'k1');
Pobj1.Value = 0.2;
Pobj1.ValueUnits = '1/second';
Kobj1.ParameterVariableNames = 'k1';
```

- 7** For mass action kinetics, the `SpeciesVariables` are automatically assigned to the reactant species. Therefore, the `SpeciesVariablesNames` property of `Kobj1` is automatically set to `DNA`. The reaction rate expression is now defined as follows.

Robj1.ReactionRate

```
ans =
      'k1*DNA'
```

Add the Reaction for Translation

A simple model of translation shows only the mRNA and protein product. For more details, see “Translation” on page 3-7.

- 1** Enter the reaction `mRNA -> mRNA + protein` and set its kinetic law to mass action. Also set the amount unit of the species `protein`.

```
Robj2 = addreaction(Mobj, 'mRNA -> mRNA + protein');
Mobj.Species(3).InitialAmountUnits = 'molecule';
Kobj2 = addkineticlaw(Robj2, 'MassAction');
```

- 2** Define the reaction rate constant `k2` for the reaction.

```
Pobj2 = addparameter(Kobj2, 'k2');
Pobj2.Value = 20;
Pobj2.ValueUnits = '1/second';
Kobj2.ParameterVariableNames = 'k2';
```

- 3 The reaction rate is now defined as follows.

```
Robj2.ReactionRate
```

```
ans =  
'k2*mRNA'
```

Add the Reaction for Gene Regulation

Transcription of DNA to mRNA is regulated by the binding of the protein product from translation to the DNA. As more protein is produced, the DNA is bound with the protein more often and less time is available for transcription with the unbound DNA. For more details, see “Gene Repression” on page 3-7.

- 1 Enter the reversible reaction for the binding and unbinding of DNA and protein. Add a parameter k_3 as the forward rate constant, and k_{3r} as the reverse rate constant.

```
Robj3 = addreaction(Mobj, 'DNA + protein <-> DNAProteinComplex');  
Mobj.Species(4).InitialAmountUnits = 'molecule';  
Kobj3 = addkineticlaw(Robj3, 'MassAction');  
Pobj3 = addparameter(Kobj3, 'k3', 'Value', 0.2, 'ValueUnits', '1/(molecule*second)');  
Pobj3r = addparameter(Kobj3, 'k3r', 'Value', 1.0, 'ValueUnits', '1/second');  
Kobj3.ParameterVariableNames = {'k3', 'k3r'};
```

- 2 Display the reaction rate.

```
Robj3.ReactionRate
```

```
ans =  
'k3*DNA*protein - k3r*DNAProteinComplex'
```

Add the Reactions for mRNA and Protein Degradation

Protein and mRNA degradation are important reactions for regulating gene expression. The steady-state level of the compounds is maintained by a balance between synthesis and degradation reactions. Proteins are hydrolyzed to amino acids with the help of proteases, while nucleic acids are degraded to nucleotides.

- 1 Enter the reaction for mRNA degradation to nucleotides. Add a parameter k_4 as the forward rate constant.

```
Robj4 = addreaction(Mobj, 'mRNA -> null');  
Kobj4 = addkineticlaw(Robj4, 'MassAction');
```

```
Pobj4 = addparameter(Kobj4, 'k4', 'Value', 1.5, 'ValueUnits', '1/second');
Kobj4.ParameterVariableNames = 'k4';
```

- 2 Display the reaction rate of mRNA degradation.

```
Robj4.ReactionRate
```

```
ans =
'k4*mRNA'
```

- 3 Enter the reaction for protein degradation to amino acids. Add a parameter k5 as the forward rate constant for the reaction.

```
Robj5 = addreaction(Mobj, 'protein -> null');
Kobj5 = addkineticlaw(Robj5, 'MassAction');
Pobj5 = addparameter(Kobj5, 'k5', 'Value', 1.0, 'ValueUnits', '1/second');
Kobj5.ParameterVariableNames = 'k5';
```

- 4 Display the reaction rate of protein degradation.

```
Robj5.ReactionRate
```

```
ans =
'k5*protein'
```

Simulate the Model

Simulate model to see its dynamic behavior.

- 1 First turn on the optional unit conversion feature. This feature automatically converts the units of physical quantities into one consistent system. This conversion is in preparation for correct simulation, but species amounts are returned in the unit that you specified (molecule in this example).

```
configset = getconfigset(Mobj);
configset.CompileOptions.UnitConversion = true;
```

- 2 Run the simulation.

```
[t, simdata, names] = sbiosimulate(Mobj);
```

- 3 Plot the results.

```
plot(t, simdata)
legend(names, 'Location', 'NorthEastOutside')
title('Gene Regulation');
xlabel('Time');
ylabel('Species Amount');
```

