SimBiology®

Getting Started Guide

R2014b

MATLAB®

MathWorks®

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

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)

Contents

1 Introduction

SimBiology Product Description **1-2**
Key Features **1-2**

SimBiology Product Overview **1-3**
Integrated Environment **1-3**
Expected Users **1-4**
Required Software **1-4**
Optional Software **1-5**
Related Software **1-6**
Compiler Setup **1-6**

Integrating SimBiology Models into Your Existing Workflow **1-7**
Existing Workflow in Experimental Research **1-7**
Workflow Incorporating Mathematical Modeling **1-8**
Workflow Using SimBiology Models for Mathematical Modeling **1-10**

Using SimBiology Desktop vs. Command Line **1-12**

SBML Support **1-13**
What Is SBML? **1-13**
Importing from SBML Files **1-13**
Exporting a SimBiology Model to SBML Format **1-13**

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 **2-4**
Tasks **2-5**

Understanding SimBiology Projects and Libraries **2-8**
What Are SimBiology Projects? **2-8**
What Are SimBiology Libraries? **2-8**

Managing SimBiology Projects and Libraries from the Desktop **2-9**
Desktop Management and Navigation Aids **2-9**
Working with Models in the Desktop **2-10**
Working with Analysis Tasks in the Desktop **2-10**
Working with Data in the Desktop **2-11**
Importing and Exporting Models, Tasks, and Data from the Desktop **2-12**
Searching a Project from the Desktop **2-13**
Generating a Project Report from the Desktop **2-13**
Using Libraries from the Desktop **2-14**

Load a Model from an SBML-formatted File **2-15**

3 Getting Started Using the Command Line

Getting Started Using the SimBiology Command Line **3-2**

About The Gene Regulation Model **3-3**
Model Diagram **3-3**
Model Reactions **3-4**
Next Steps **3-7**

Creating a SimBiology Model **3-8**
Next Steps **3-9**

Adding the Reaction for Transcription **3-10**
Next Steps **3-11**

Adding the Reaction for Translation **3-12**
Next Steps **3-12**

Adding the Reactions for Gene Regulation **3-13**
Next Steps **3-13**

Adding the Reactions for mRNA and Protein Degradation **3-14**
Next Steps **3-14**

Simulating the Model **3-15**
See Also **3-16**
Next Steps **3-16**

Simulating the Model with a Stochastic Solver **3-17**
See Also **3-20**
Next Steps **3-21**

More Examples of Using the Command Line **3-22**

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
- "SimBiology Product Overview" on page 1-3
- "Integrating SimBiology Models into Your Existing Workflow" on page 1-7
- "Using SimBiology Desktop vs. Command Line" on page 1-12
- "SBML Support" on page 1-13

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

SimBiology Product Overview

In this section...

"Integrated Environment" on page 1-3
"Expected Users" on page 1-4
"Required Software" on page 1-4
"Optional Software" on page 1-5
"Related Software" on page 1-6
"Compiler Setup" on page 1-6

Integrated Environment

SimBiology software provides an integrated environment for modeling biological processes, simulating the dynamic behavior of these processes, and analyzing the model with simulation and experimental data. Biological processes include metabolic, genetic, and signaling pathways with transform, binding, and transport reactions.

You can also create and analyze pharmacokinetic models. For more information see " Pharmacokinetic Modeling Functionality".

- **Model** — Design and build models by entering reactions, species, parameters, kinetic laws, rules, and events

  Import SBML models created with the SimBiology software or other modeling software that is compliant with the Systems Biology Markup Language (SBML) standard.

  Export models you create with the SimBiology desktop to MATLAB and continue your simulations and analysis with command-line functions.

  For more information, see "What is a Model?".
- **Analyze Structure** — Verify that the model can be simulated, and use the verification results to fix any incompatibilities in the model. Investigate the structure of your models, including determining conserved moieties, adjacency matrices, and stoichiometry matrices.

  For more information, see "Overview of Structural Analysis".

- **Simulate and Analyze** — Observe changes in species amounts and parameter values over time. Convert your model to a system of differential equations and simulate the model numerically with various differential equation solvers. The deterministic solvers include stiff and nonstiff ordinary differential equation (ODE) solvers. The stochastic solvers include a stochastic simulation algorithm with implicit and explicit tau variations. Perform multiple stochastic ensemble runs, save data from a simulation, compare simulation and experimental data, perform sensitivity analysis, species or parameter scans, and parameter estimation.

  For more information, see "Simulation and Analysis".

Expected Users

The people who use SimBiology software come from a wide range of areas including biology, systems biology, pharmacology, computer science, and engineering. This product is intended for research scientists, computational biologists, and students who need to develop and study biological pathways at the molecular and systems level, develop custom analysis applications, or implement published pathways, and visualize results.

Industry and Professional — SimBiology software allows you to model, simulate, and analyze biochemical and system pathways for applications in drug discovery and design, target identification, and pharmacokinetic modeling.

Modeling, simulating, and analyzing a biological system can test hypotheses for a pathway, identify side effects caused by drug interactions with a target compound, and identify biochemical pathways that lead to disease.

Academia — Build rigorous, dynamic, quantitative models that allow you to understand and predict system behavior at the molecular level. Build models to explore enzyme kinetics. Leverage the MATLAB foundation to easily manipulate large data sets.

Simulating the dynamic behavior of a model can confirm the validity of the models and identify behaviors and control mechanisms not apparent from studying static models. Validate models experimentally and use the model to predict *in vitro* and *in vivo* behavior.

Required Software

To use SimBiology software, you must first install the following MathWorks® product:

MATLAB	Provides a command-line interface and an integrated software environment. For instructions, see the MATLAB installation documentation for your platform. If you have installed MATLAB and want to check which other MathWorks products are installed, enter `ver` in the MATLAB Command Window.

Optional Software

Statistics Toolbox™ (Version 7.0 (R2008b) or later)	Provides fitting tools including functions required to analyze nonlinear mixed effects (NLME). **Note:** Statistics Toolbox is required to use SimBiology to perform population fitting using `sbionlmefit` or `sbionlmefitsa`, or to perform individual fitting using `sbionlinfit`.
Global Optimization Toolbox	Solve optimization problems using genetic and direct search algorithms. If this toolbox is installed, you can use various genetic and direct search algorithms for parameter estimation. If this toolbox is not installed, the software uses the optimization algorithms available in MATLAB.
Optimization Toolbox™	Optimization Toolbox extends the MATLAB technical computing environment with tools and widely used algorithms for standard and large-scale optimization. These algorithms solve constrained and unconstrained continuous and discrete problems. If the Optimization Toolbox product is installed, you can use some algorithms included this product for parameter estimation in SimBiology software. If the Optimization Toolbox product is not installed, the software uses the optimization algorithms available in MATLAB.
C Compiler	Required to prepare the model for accelerating simulations. For a list of supported compilers, see Supported and Compatible Compilers.

Related Software

Bioinformatics Toolbox™	Read, analyze, and visualize genomic, proteomic, and microarray data.

Compiler Setup

To prepare your 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 atwww.mathworks.com.
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 "Code Generation from MATLAB Code".

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.

Integrating SimBiology Models into Your Existing Workflow

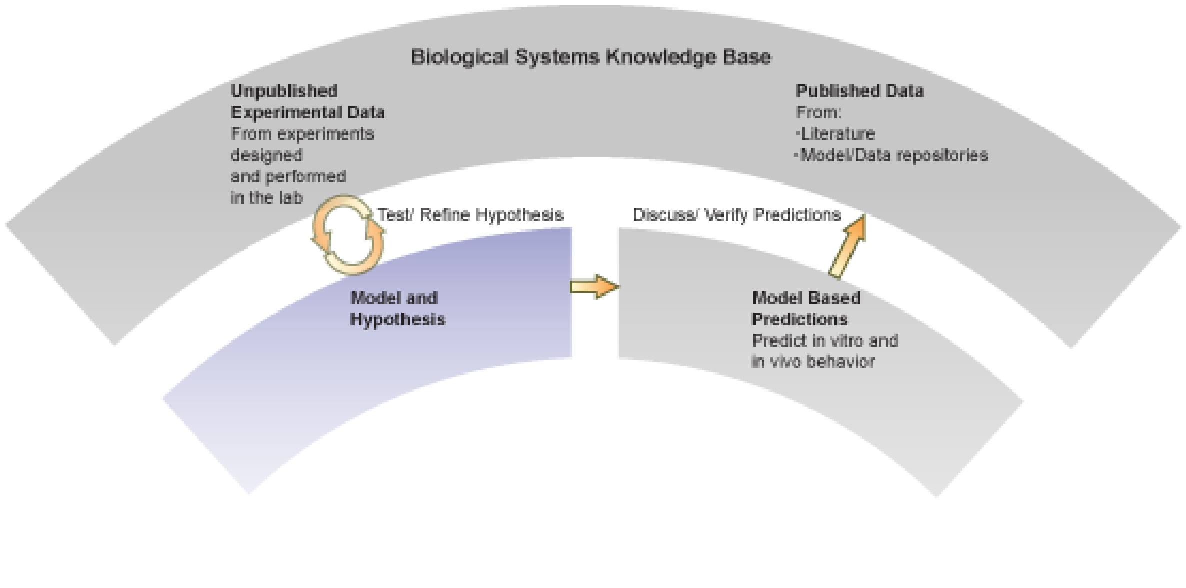
In this section...

"Existing Workflow in Experimental Research" on page 1-7
"Workflow Incorporating Mathematical Modeling" on page 1-8
"Workflow Using SimBiology Models for Mathematical Modeling" on page 1-10

Existing Workflow in Experimental Research

Consider the following typical workflow in experimental research.

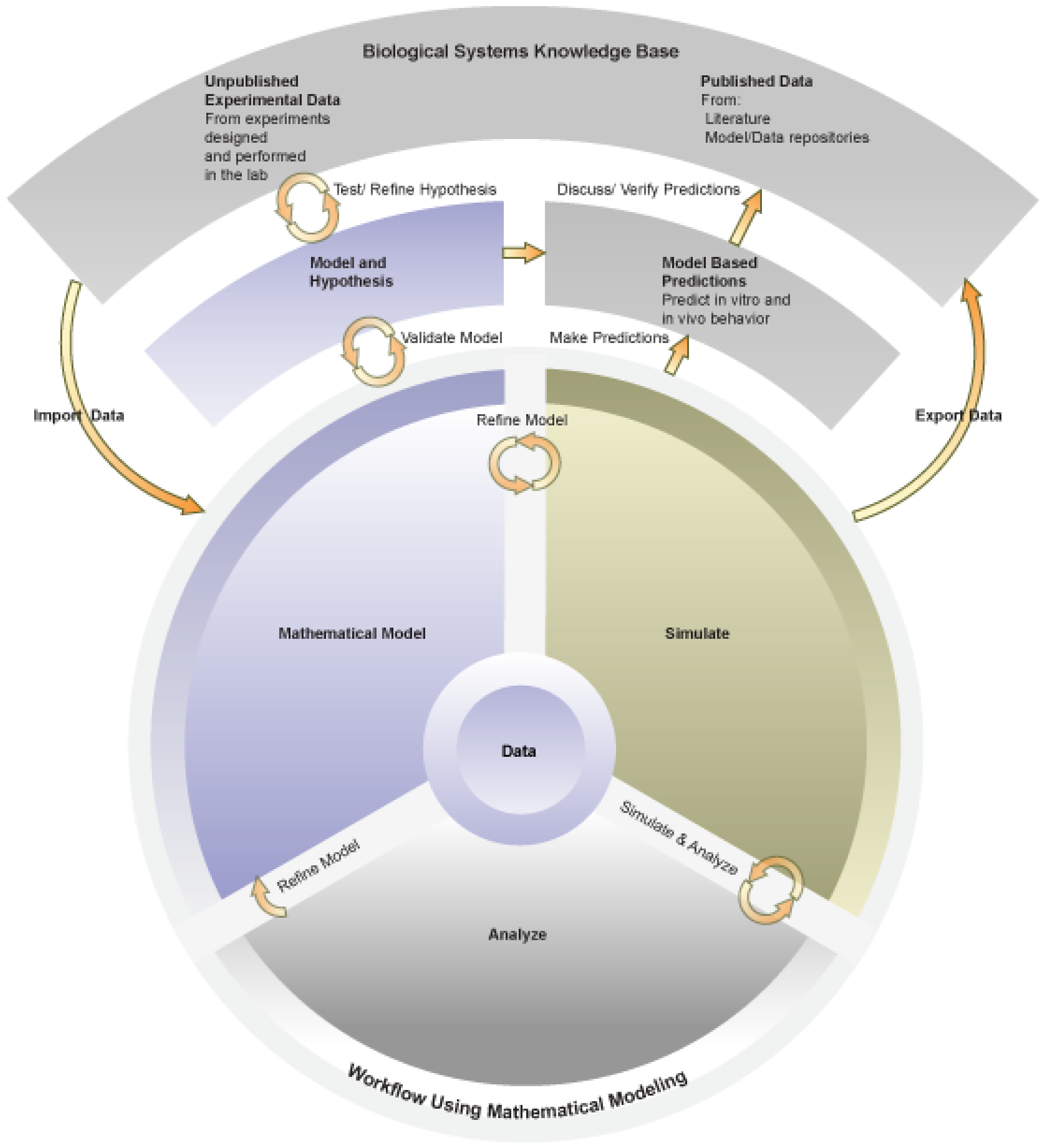
1. Use preliminary results and published data to develop a model and hypothesis in your area of interest.
2. Validate your model by making predictions and designing experiments to test your model and hypothesis.
3. Make further predictions based on your model.
4. Publish results and predictions.
5. Continue to test predictions experimentally.


Biological Systems Knowledge Base
Unpublished Experimental Data
From experiments designed and performed in the lab
Published Data
From:
-Literature
-Model/Data repositories
Test/ Refine Hypothesis
Discuss/ Verify Predictions
Model and Hypothesis
Model Based Predictions
Predict in vitro and in vivo behavior

Workflow Incorporating Mathematical Modeling

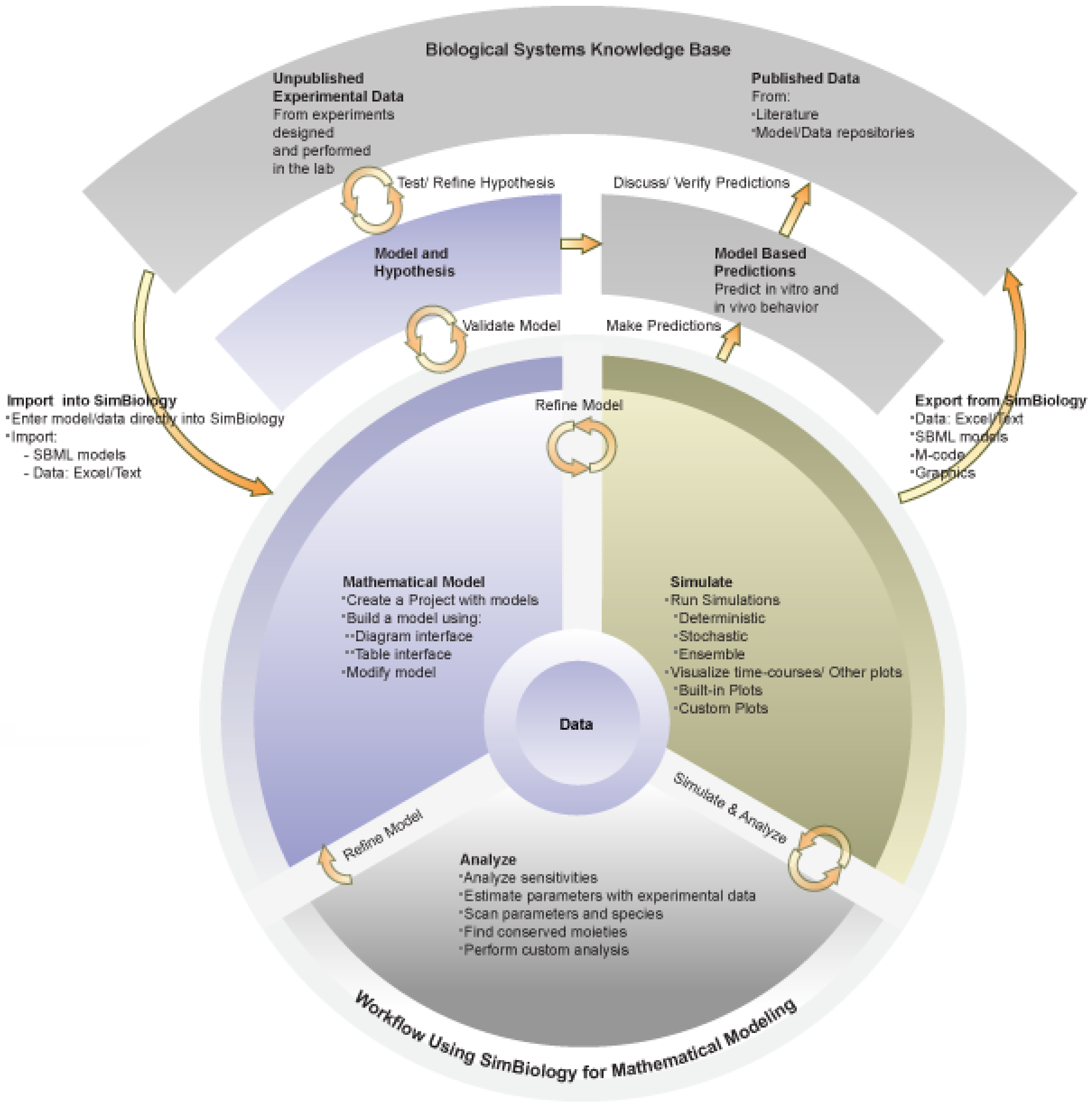
Mathematical modeling integrates into the preceding workflow by providing the means to:

- Consolidate quantitative data into the model.
- Represent large pathways and networks and lay them out in meaningful ways using a software application.
- Study emergent properties of pathways and use these predictions to guide the direction of experimental research in ways that can save time and development costs.


Biological Systems Knowledge Base
Unpublished Experimental Data
From experiments designed and performed in the lab
Published Data
From:
Literature
Model/Data repositories
Test/ Refine Hypothesis
Discuss/ Verify Predictions
Model and Hypothesis
Model Based Predictions
Predict in vitro and in vivo behavior
Validate Model
Make Predictions
Import Data
Export Data
Refine Model
Mathematical Model
Simulate
Data
Simulate & Analyze
Refine Model
Analyze
Workflow Using Mathematical Modeling

Workflow Using SimBiology Models for Mathematical Modeling

SimBiology software provides an integrated environment for mathematical modeling and analysis of biological and biochemical pathways. The following figure shows an overlay of features and tasks in the product pictured in the context of the complete workflow in experimental research.


Biological Systems Knowledge Base
Unpublished Experimental Data
From experiments designed and performed in the lab
Published Data
From:
•Literature
•Model/Data repositories
Test/ Refine Hypothesis
Discuss/ Verify Predictions
Model and Hypothesis
Model Based Predictions
Predict in vitro and in vivo behavior
Validate Model
Make Predictions
Import into SimBiology
•Enter model/data directly into SimBiology
•Import:
- SBML models
- Data: Excel/Text
Export from SimBiology
•Data: Excel/Text
•SBML models
•M-code
•Graphics
Refine Model
Mathematical Model
•Create a Project with models
•Build a model using:
••Diagram interface
••Table interface
•Modify model
Simulate
•Run Simulations
•Deterministic
•Stochastic
•Ensemble
•Visualize time-courses/ Other plots
•Built-in Plots
•Custom Plots
Data
Refine Model
Simulate & Analyze
Analyze
•Analyze sensitivities
•Estimate parameters with experimental data
•Scan parameters and species
•Find conserved moieties
•Perform custom analysis
Workflow Using SimBiology for Mathematical Modeling

Using SimBiology Desktop vs. Command Line

There are two ways to use SimBiology:

- **SimBiology desktop** — Use the desktop GUI (graphical user interface) to interactively iterate through the model building and analysis workflow. For more information, see “SimBiology Desktop Overview”.
- **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. For more information, see “Getting Started Using the SimBiology Command Line ”.

SBML Support

In this section...

"What Is SBML?" on page 1-13
"Importing from SBML Files" on page 1-13
"Exporting a SimBiology Model to SBML Format" on page 1-13

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`.

Because SimBiology supports a subset of the SBML Level 2 Version 4 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.
- Function definitions — Models containing function definitions are loaded, but a warning is displayed, and the function definitions 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` field does not have a corresponding property in the SimBiology species object. For more information on dimensions for SimBiology species, see `DefaultSpeciesDimension`.

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
- "Understanding SimBiology Projects and Libraries" on page 2-8
- "Managing SimBiology Projects and Libraries from the Desktop" on page 2-9
- "Load a Model from an SBML-formatted File" on page 2-15

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

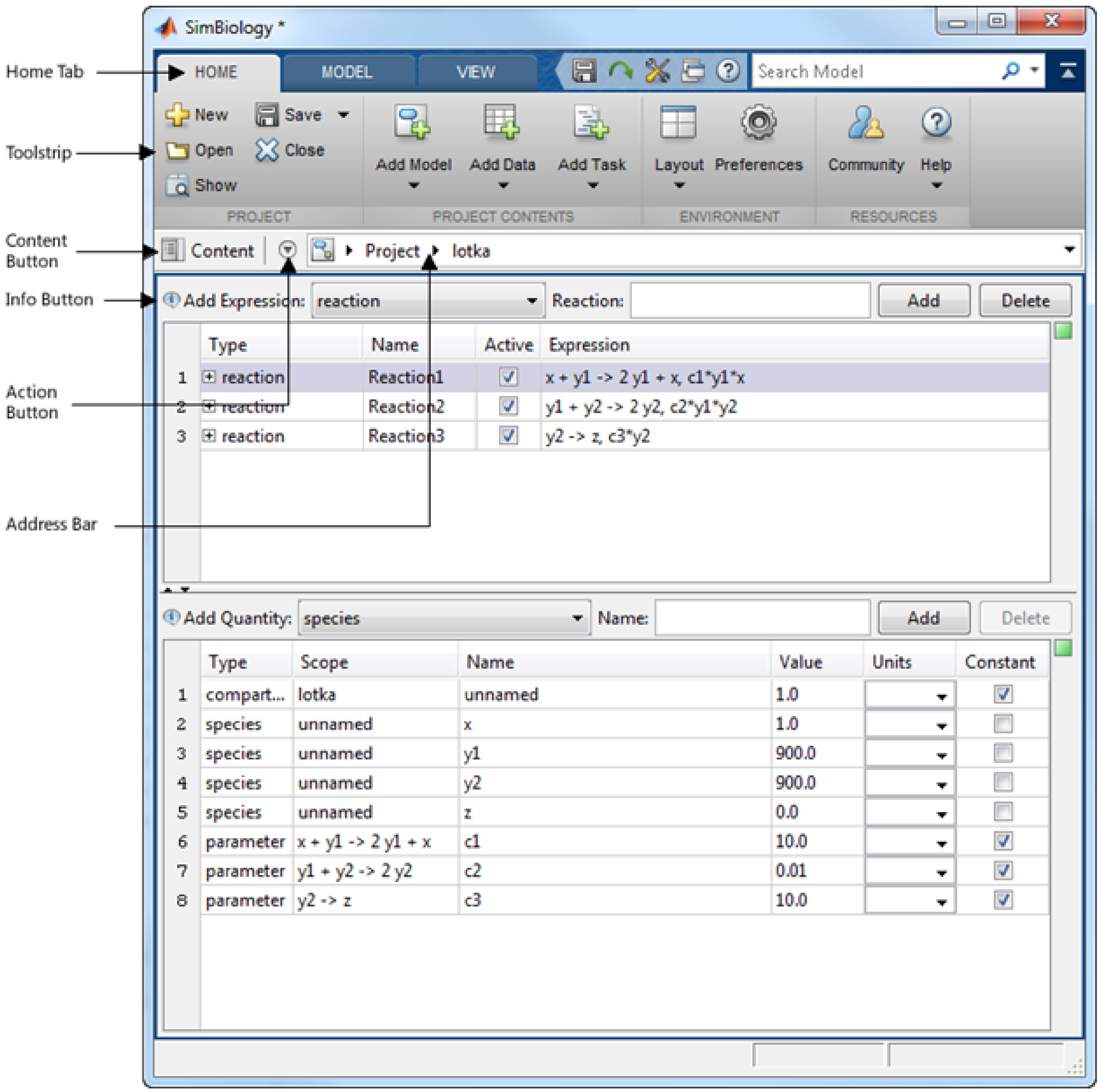
Open the desktop by typing the following at the MATLAB command line.

```
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.



	Type	Name	Active	Expression
1	reaction	Reaction1	✓	x + y1 -> 2 y1 + x, c1*y1*x
2	reaction	Reaction2	✓	y1 + y2 -> 2 y2, c2*y1*y2
3	reaction	Reaction3	✓	y2 -> z, c3*y2

	Type	Scope	Name	Value	Units	Constant
1	compart...	lotka	unnamed	1.0		✓
2	species	unnamed	x	1.0		
3	species	unnamed	y1	900.0		
4	species	unnamed	y2	900.0		
5	species	unnamed	z	0.0		
6	parameter	x + y1 -> 2 y1 + x	c1	10.0		✓
7	parameter	y1 + y2 -> 2 y2	c2	0.01		✓
8	parameter	y2 -> z	c3	10.0		✓

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 MALAB 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

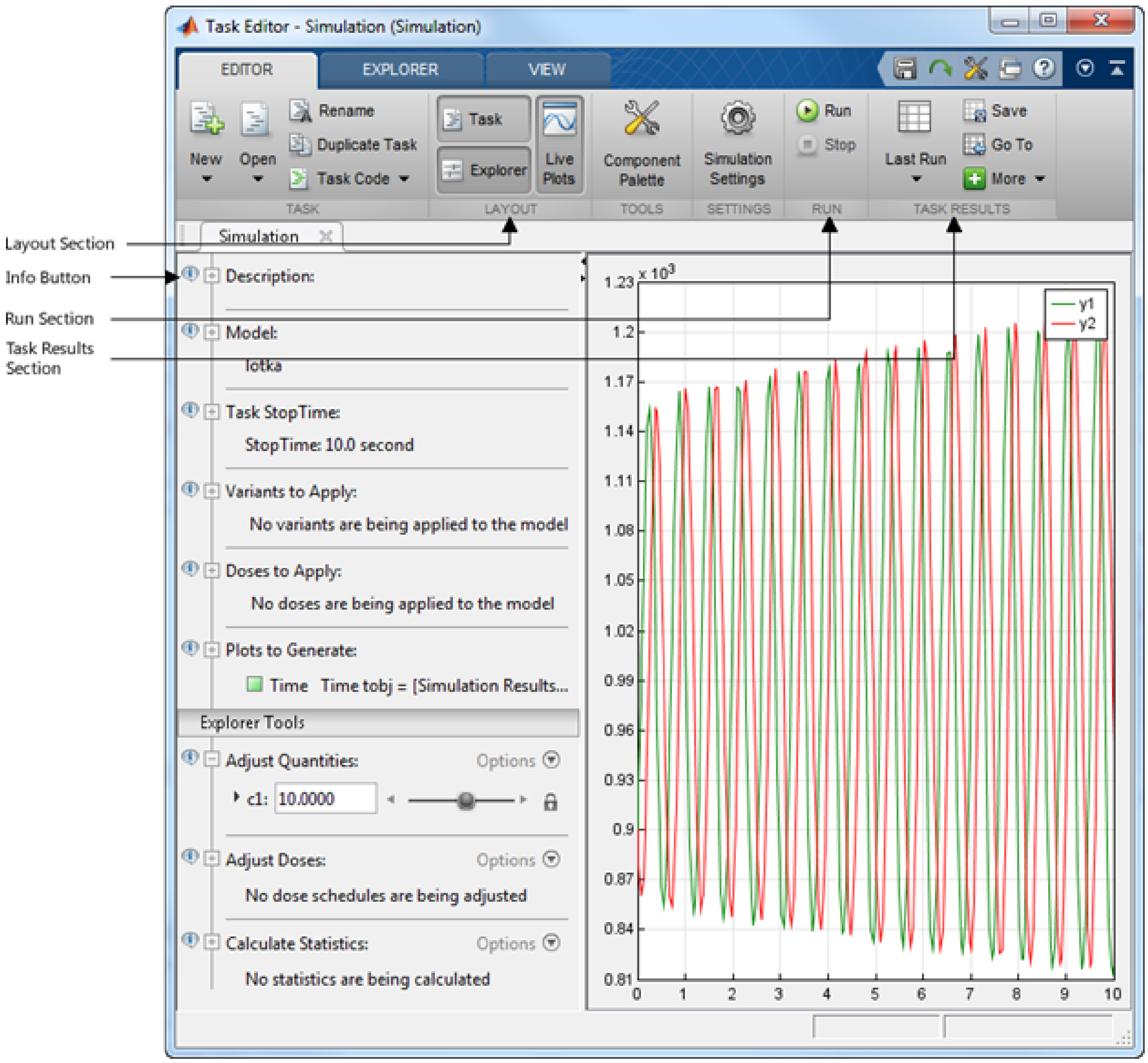
Tasks are analyses that can be performed on a model. There are several built-in tasks such as simulation, estimation, and sensitivity analysis. In addition, custom tasks can be created using the MATLAB language.

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 section: general task setup, interactive task setup, and live visualization of results. 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.



Description:

Model:
lotka

Task StopTime:
StopTime: 10.0 second

Variants to Apply:
No variants are being applied to the model

Doses to Apply:
No doses are being applied to the model

Plots to Generate:
🟩 Time Time tobj = [Simulation Results...

Adjust Quantities:
c1: 10.0000

Adjust Doses:
No dose schedules are being adjusted

Calculate Statistics:
No statistics are being calculated

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.

Related Examples

- "Create and Simulate a Simple Model"

Understanding SimBiology Projects and Libraries

What Are SimBiology Projects?

When using the desktop, *projects* let you manage the model building and analysis workflow.

A project encapsulates the following:

- **Models** — Sets of quantities (compartments, species, and parameters) and expressions (reactions, discrete events, and rules, including differential equations, algebraic constants, and assignments) that describe the dynamics of a system.
- **Tasks** — Analyses that operate on a model or project.
- **Data** — Imported data or data generated by analyses.

A project saves related models, tasks, and data in a project file with an `.sbproj` extension. For details, see "Managing SimBiology Projects and Libraries from the Desktop" on page 2-9.

What Are SimBiology Libraries?

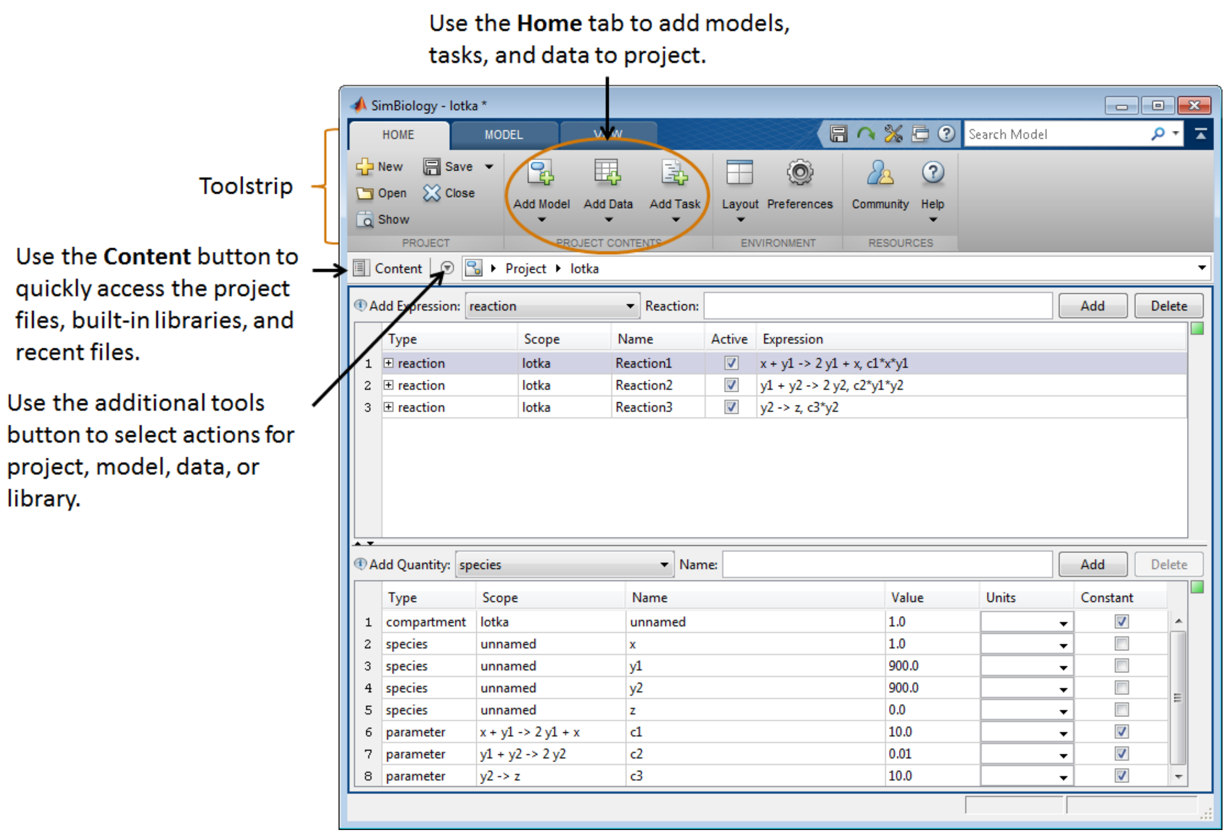
SimBiology includes the following libraries that contain both built-in and user-defined components to use when building models or plotting data:

- **Units Library**
- **Unit Prefixes Library**
- **Kinetic Laws Library**
- **Plot Types Library** — Contains plot types for plotting results of analysis tasks or plotting data.
- **Blocks Library** — Contains blocks for building models in the **Diagram** view.

The libraries are available for all projects and are not part of any one project. For details, see "Managing SimBiology Projects and Libraries from the Desktop" on page 2-9.

Managing SimBiology Projects and Libraries from the Desktop

Desktop Management and Navigation Aids


Use the **Home** tab to add models, tasks, and data to project.

Toolstrip

Use the **Content** button to quickly access the project files, built-in libraries, and recent files.

Use the additional tools button to select actions for project, model, data, or library.

	Type	Scope	Name	Active	Expression
1	reaction	lotka	Reaction1	✓	x + y1 -> 2 y1 + x, c1*x*y1
2	reaction	lotka	Reaction2	✓	y1 + y2 -> 2 y2, c2*y1*y2
3	reaction	lotka	Reaction3	✓	y2 -> z, c3*y2

	Type	Scope	Name	Value	Units	Constant
1	compartment	lotka	unnamed	1.0		✓
2	species	unnamed	x	1.0		
3	species	unnamed	y1	900.0		
4	species	unnamed	y2	900.0		
5	species	unnamed	z	0.0		
6	parameter	x + y1 -> 2 y1 + x	c1	10.0		✓
7	parameter	y1 + y2 -> 2 y2	c2	0.01		✓
8	parameter	y2 -> z	c3	10.0		✓

SimBiology Desktop With an Open Project and a Model Selected

Use the following to manage and navigate models and data in a project and to manage libraries:

- **Toolstrip** — Displays the **Home** tab and other context-sensitive tabs.
- **Home tab** — Use to open and manage projects, including adding models, tasks, and data to a project.

- Content **button** — Use to select a project, model, task, or data to view and edit in the desktop window. Use to navigate between models, tasks, and data. Use to select and view a library, and to navigate between libraries.
- **Additional tools button** — Use to select actions appropriate for the item (project, model, data, or library) selected and displayed in the desktop.

Working with Models in the Desktop

Opening, Creating, and Editing Models

After selecting a model, you can do the following from the **Model** tab or desktop window:

- Open a model page from the **Model** tab using the **Open** button.
- Create and edit the quantities and expressions that define the model.
- Create and edit modifiers (variants and doses) for the model.

Using a Model-Centric Workflow

If you want to refine a model, select the model from the address bar, then perform iterative edits on the model. Without leaving the model, you can run multiple tasks and view results to help guide you with your model edits.

Working with Analysis Tasks in the Desktop

Creating and Editing Analysis Tasks

After selecting an analysis task (for example, calculate sensitivities or run scan), you can do the following from the **Task Editor** window.

- Select a model to run the analysis on, optionally select a variant or dose, and then define the model's simulation settings.
- Define the settings of the analysis.
- Select the plots to generate when the analysis completes.
- Adjust quantities and doses using the **Explorer Tools**.
- Run the analysis.

For more information on available analysis tasks, see “Simulation and Analysis”.

Using a Task-Centric Workflow

You can perform iterative runs and edits on the task to refine an analysis task. Without leaving the task, select different models to run the task on or adjust quantities or doses using the **Explorer tools**. For example, select different models for a parameter fit task to find the model that best fits the data.

Creating Plots of Analyses

To visualize data from analyses, you can choose from a list of built-in plot types. You can also create your own plot types that extend or customize the built-in plot types. Create a custom plot type in the **Plot Types** library. For more information, see “What Are SimBiology Libraries?” on page 2-8 and “Using Libraries from the Desktop” on page 2-14.

Creating Custom Analysis Tasks

You can create a custom analysis task from scratch or use the code of an existing task by adding a **Create custom analysis** task.

Working with Data in the Desktop

Importing and Generating Data

There are two types of data to manage in the desktop:

- **Imported data** — Data that you import from external sources to use for an analysis (such as parameter fitting) or to use to compare to results from a simulation or analysis.
- **Task data** — Data generated from analysis tasks.

After you import or generate data, select the data from the address bar to view, plot, and manipulate it as described in “Working with Imported Data” on page 2-11 and “Working with Task Data” on page 2-12.

Working with Imported Data

You can do the following with data that you import:

- **View raw data** — When viewing imported raw data, you can specify the units for the column headers.
- **Plot data** — Select from a list of built-in and user-defined plot types.

- **Exclude data** — Exclude rows of data that meet or don't meet certain criteria, using column headers and relational operators. For example, you can exclude outliers in a data set, such as rows containing a `DOSE` value > `20`.
- **Add derived data** — Create additional columns of data that you derive from the existing columns of data using expressions and relational operators. For example, you can convert a column header to different units, or create a ratio of two columns of data.
- **View data statistics** — View statistics, such as minimum, maximum, mean, and time of maximum.

Working with Task Data

You can do the following with data generated from an analysis task:

- **Save data** — When you generate data from an analysis task, you can save the data to prevent it from being overwritten by subsequent analysis tasks. Saved data includes the task analysis settings used to create it.
- **View raw data** — Inspect the raw data generated by a simulation or analysis.
- **Plot data** — Select from a list of built-in and user-defined plot types. You can plot data without having to rerun the analysis task.
- **Export data** — Export analysis task data to a file or a variable in the MATLAB Workspace.

Importing and Exporting Models, Tasks, and Data from the Desktop

Note: To export models, tasks, or data to HTML, see "Generating a Project Report from the Desktop" on page 2-13.

Models

You can import models from any of the following:

- sbproj file
- SBML (file or URL)
- MATLAB Workspace variable

You can export models to either of the following:

- SBML file
- MATLAB Workspace variable

For more information on how SimBiology supports SBML, see “SBML Support” on page 1-13.

Tasks

You can save a task as code in a MATLAB file.

Data

You can import data from any of the following:

- sbproj file
- Microsoft Excel file
- txt or CSV file
- MAT-file
- MATLAB Workspace variable

Select **Export Data** from the Actions button to export data to a MATLAB Workspace variable, MAT file, or Microsoft Excel file (Windows only).

Searching a Project from the Desktop

To do a simple search of a project, use the button above the toolstrip.

Alternately, create search criteria to find specific components in the models in a project by adding a **Search model** task. After editing or loading a project, you must execute a search to have up-to-date results. Like other tasks, a search model task is saved with the project.

Generating a Project Report from the Desktop

You can generate reports (HTML) with model-specific information, such as analysis task results, imported data, and searches, by adding a **Generate report** task. Like other tasks, a report task is saved with the project.

Using Libraries from the Desktop

After selecting a library (using the Content button or the address bar), you can:

- **Add user-defined components to a library** — You can add, modify, and delete user-defined components in a library, but you cannot modify or delete built-in components in a library.
- **Export and import libraries** — Export user-defined components from a library to an sblib file, which another SimBiology user can import.

Load a Model from an SBML-formatted File

This example shows how to import an SBML model from a file.

You can import an SBML model from a file or URL in SimBiology Desktop. Because SimBiology supports a subset of the SBML Level 2 Version 4 specification, some SBML features are not imported into the SimBiology model.

Open SimBiology desktop by typing `simbiology` in the MATLAB Command Window or clicking **SimBiology** on the **Apps** tab.

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

Navigate to the folder `\MATLAB\R2013b\toolbox\simbio\simbiodemos\` and open the sample file named `lotka.xml`.

SimBiology loads the `lotka` model.

More About

- " Importing from SBML Files"

Getting Started Using the Command Line

- "Getting Started Using the SimBiology Command Line " on page 3-2
- "About The Gene Regulation Model" on page 3-3
- "Creating a SimBiology Model" on page 3-8
- "Adding the Reaction for Transcription" on page 3-10
- "Adding the Reaction for Translation" on page 3-12
- "Adding the Reactions for Gene Regulation" on page 3-13
- "Adding the Reactions for mRNA and Protein Degradation" on page 3-14
- "Simulating the Model" on page 3-15
- "Simulating the Model with a Stochastic Solver" on page 3-17
- "More Examples of Using the Command Line" on page 3-22

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.

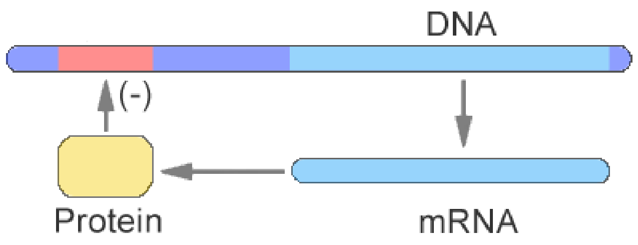
This tutorial shows how to model a simple gene-regulation pathway using the command-line interface:

- "About The Gene Regulation Model" on page 3-3
- "Creating a SimBiology Model" on page 3-8
- "Adding the Reaction for Transcription" on page 3-10
- "Adding the Reaction for Translation" on page 3-12
- "Adding the Reactions for Gene Regulation" on page 3-13
- "Adding the Reactions for mRNA and Protein Degradation" on page 3-14
- "Simulating the Model" on page 3-15
- "Simulating the Model with a Stochastic Solver" on page 3-17
- "More Examples of Using the Command Line" on page 3-22

About The Gene Regulation Model

Model Diagram

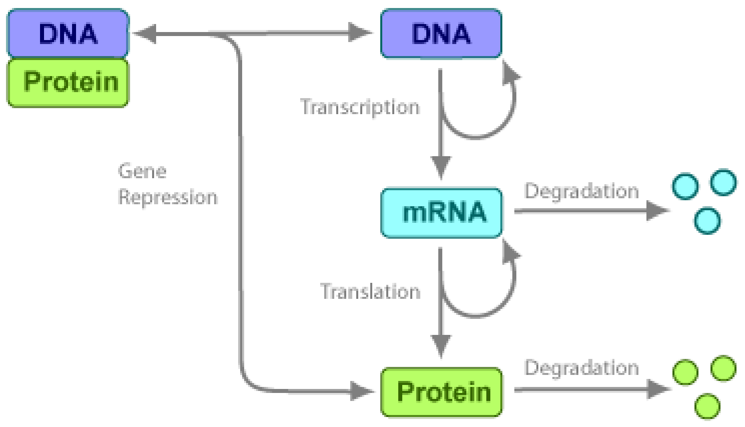
You can visualize a systems biology model with various levels of detail. One view sketches only the major species and processes. This model is an example of simple gene regulation, where the protein product from translation controls transcription. You could create a more complex model by adding the enzymes, 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 -> DNA + mRNA             (transcription)
mRNA -> mRNA + protein        (translation)
DNA + protein -> DNA_protein (binding)
DNA_protein -> 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.

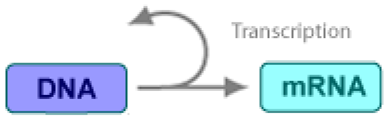


Model Reactions

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.

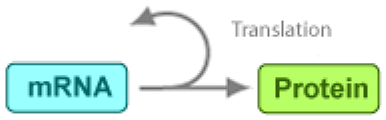


This model simplifies the transcription and the synthesis of mRNA by using a single reaction.

```
     Reaction: DNA -> DNA + mRNA
Reaction rate: v = k1*DNA   molecule/second
      Species: DNA  =  50   molecule
               mRNA =   0   molecule
   Parameters: k1 = 0.2 1/second
```

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.

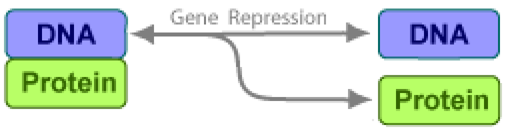


The synthesis of the protein is modeled as a single reaction.

```
     Reaction: mRNA -> mRNA + protein
Reaction rate: v = k2*mRNA  molecule/second
      Species: mRNA =  0 molecule
               protein =  0 molecule
   Parameters: k2 = 20 1/second
```

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 -> DNA_protein
Reaction rate: v = k3*DNA*protein molecule/second
      Species: DNA = 50 molecule
               protein =  0 molecule
   Parameters: k3 =  0.2 1/(molecule*second)
```

Notice how the units are described for the parameter `k3`. In the literature, some of the common ways to display second-order rate constants are mole^{-1}second^{-1} or 1/mole-second. However, software evaluations of these common ways are sometimes ambiguous.

Reverse Reaction (Unbinding)

```
     Reaction: DNA_protein -> DNA + protein
Reaction rate: v = k3r*DNA_protein molecule/second
      Species: DNA_protein = 0 molecule
   Parameters: k3r = 1 1/second
```

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

```
     Reaction: DNA + protein <-> DNA_protein
Reaction rate: v = k3*DNA*protein - k3r*DNA_protein molecule/second
      Species: DNA = 50 molecule
               protein =  0 molecule
   Parameters: k3 =  0.2 1/(molecule*second)
               k3r = 1 1/second
```

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

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

```
     Reaction: mRNA -> null
Reaction rate: v = k4*mRNA molecule/second
      Species: mRNA = 0 molecule
   Parameters: k4 = 1.5 1/second
```

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

Protein
Degradation

```
     Reaction: protein -> null
Reaction rate: v = k5*protein molecule/second
      Species: protein = 0.0 molecule
```

```
   Parameters: k5 = 1.0 1/second
```

Next Steps

“Creating a SimBiology Model” on page 3-8

Creating 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 model object. For example, to create a model with the name `cell`, in the MATLAB Command Window, type

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

MATLAB creates `Mobj`, a model object in the workspace and displays a summary of the objects in the model.

```
SimBiology Model - cell

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

2 Display all the model object properties by typing

```
get(Mobj)
```

MATLAB displays a list of properties for the model object and their current values.

```
    Annotation: ''
          Events: [0x1 double]
            Name: 'cell'
           Notes: ''
      Parameters: [0x1 double]
          Parent: [1x1 SimBiology.Root]
         Species: [0x1 handle]
    Compartments: [0x1 double]
       Reactions: [0x1 double]
           Rules: [0x1 double]
             Tag: ''
            Type: 'sbiomodel'
        UserData: []
```

Some of the model properties are themselves objects or arrays of objects.

3. Add a compartment named `contents` to the model.

   ```
   compObj = addcompartment(Mobj, 'contents');
   ```

Next Steps

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

Adding the Reaction for Transcription

A simple model of transcription shows only the DNA and mRNA. For more details, see "Transcription" on page 3-4.

1 Add the reaction `DNA -> DNA + mRNA` to the model.

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

The species `DNA` and `mRNA` are automatically added to the model.

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 Specify the kinetics of the reaction to be Mass Action by creating a kinetic law object, `Kobj1`, and setting its `KineticLawName` property to `MassAction`.

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

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

3 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`.

```
get(Kobj1, 'ParameterVariables')
get(Kobj1, 'SpeciesVariables')

ans =

    'Forward Rate Parameter'


ans =

    'MassAction Species'
```

4 Because 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 `ParameterVariablesNames` property of `Kobj1` to `k1`.

```
Pobj1 = addparameter(Kobj1, 'k1', 0.2);
set(Kobj1, 'ParameterVariableNames', 'k1');
```

5 For Mass Action kinetics, the `SpeciesVariables` are automatically assigned to the reactants. Therefore, the `SpeciesVariablesNames` property of `Kobj1` is automatically set to `DNA`. To confirm this, retrieve the `SpeciesVariableNames` property of `Kobj1`.

```
get(Kobj1, 'SpeciesVariableNames')

ans =

    'DNA'
```

The reaction rate expression is now defined as `k1 * DNA`.

6 Set the initial amount for the species `DNA` to `50` and `mRNA` to `0`. Select the species using the function `sbioselect`, which returns an object.

```
Sobj1 = sbioselect(Mobj, 'Type', 'species', 'Name', 'DNA');
set(Sobj1, 'InitialAmount', 50);
```

The value for `mRNA` defaults to `0`.

7 Check the initial amounts for the species.

```
Mobj.Species
```

MATLAB displays a summary list.

```
SimBiology Species Array

   Index:  Compartment:  Name:  InitialAmount: InitialAmountUnits:
   1       contents       DNA    50
   2       contents       mRNA   0
```

Next Steps

"Adding the Reaction for Translation" on page 3-12

Adding the Reaction for Translation

A simple model of translation shows only the mRNA and protein product. For more details, see “Translation” on page 3-5.

1 Enter the reaction `mRNA -> mRNA + protein` with reaction rate equation `v = k*mRNA`.

```
Robj2 = addreaction(Mobj, 'mRNA -> mRNA + protein');
Kobj2 = addkineticlaw(Robj2,'MassAction');
```

2 Define the reaction rate constant `k2`.

```
Pobj2 = addparameter(Kobj2, 'k2', 20.0);
set(Kobj2, 'ParameterVariableNames','k2');
```

Next Steps

“Adding the Reactions for Gene Regulation” on page 3-13

Adding the Reactions 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 3-5.

Enter the reversible reaction for `DNA + protein <-> DNA_protein` with a forward reaction rate equation `v = k3*DNA*protein` and a reverse rate `v = k3r*DNA_protein`.

```
Robj3 = addreaction(Mobj, 'DNA + protein <-> DNA_protein');
Kobj3 = addkineticlaw(Robj3,'MassAction');
Pobj3 = addparameter(Kobj3, 'k3', 0.2);

Pobj3r = addparameter(Kobj3, 'k3r', 1.0);
set(Kobj3, 'ParameterVariableNames', {'k3', 'k3r'});
```

Next Steps

"Adding the Reactions for mRNA and Protein Degradation" on page 3-14

Adding 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.

```
Robj4 = addreaction(Mobj, 'mRNA -> null');
Kobj4 = addkineticlaw(Robj4, 'MassAction');
Pobj4 = addparameter(Kobj4,  'k4', 1.5);
set(Kobj4, 'ParameterVariableNames','k4');
```

2. Enter the reaction for protein degradation to amino acids.

```
Robj5 = addreaction(Mobj, 'protein -> null');
Kobj5 = addkineticlaw(Robj5,'MassAction');
Pobj5 = addparameter(Kobj5,  'k5', 1.0);
set(Kobj5, 'ParameterVariableNames','k5');
```

Next Steps

“Simulating the Model” on page 3-15

Simulating the Model

After you create a model, you can simulate the dynamic behavior of the model.

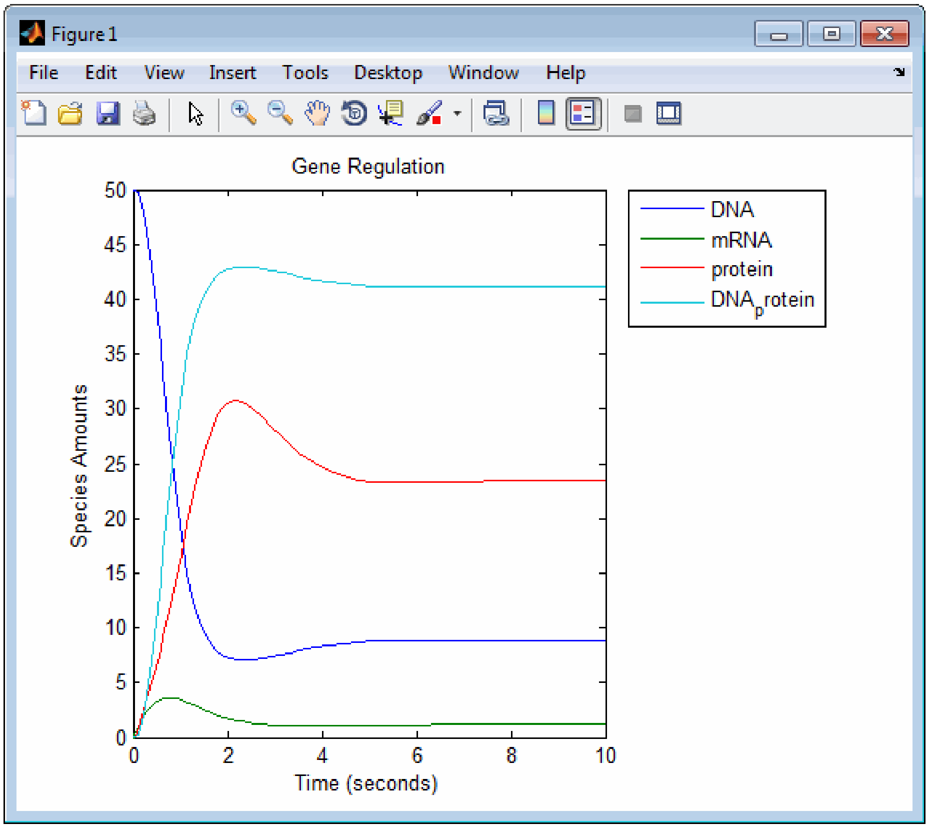
1 Run the simulation.

```
[t_ode, x_ode, names] = sbiosimulate(Mobj);
```

2 Plot the results.

```
figure;
set(gcf, 'color', 'white');
plot(t_ode, x_ode(:,1:4));
legend(names, 'Location', 'NorthEastOutside')
title('Gene Regulation');
xlabel('Time (seconds)');
ylabel('Species Amounts');
```

MATLAB plots a figure with the simulation results.



See Also

- “ Model Simulation”
- “ODE Solvers”

Next Steps

“Simulating the Model with a Stochastic Solver” on page 3-17
“More Examples of Using the Command Line” on page 3-22

Simulating the Model with a Stochastic Solver

SimBiology software includes three stochastic solvers. The stochastic solvers more accurately calculate the change in species amounts with a small number of molecules.

1 Get the active configuration set for the model, `Mobj`.

```
cs = getconfigset(Mobj)

 Configuration Settings - default (active)
     SolverType:                 ode15s
     StopTime:                   10

   SolverOptions:
     AbsoluteTolerance:          1.000000e-006
     RelativeTolerance:          1.000000e-003
     SensitivityAnalysis:        false

   RuntimeOptions:
     StatesToLog:                all

   CompileOptions:
     UnitConversion:             false
     DimensionalAnalysis:        true

   SensitivityAnalysisOptions:
     Inputs:                     0
     Outputs:                    0
```

The configset object, `cs`, contains all the simulation settings, including the stop time and solver.

2 Set the `SolverType` to the `ssa` stochastic solver, and list the configuration set again.

```
set(cs, 'SolverType', 'ssa');
cs
```

Notice that the `SolverOptions` for the stochastic solver are different from the ODE options.

```
Configuration Settings - default (active)
     SolverType:                 ssa
     StopTime:                   10.000000
```

```
   SolverOptions:
     LogDecimation:                1

   RuntimeOptions:
     StatesToLog:                  all

   CompileOptions:
     UnitConversion:               false
     DimensionalAnalysis:          true

   SensitivityAnalysisOptions:
     Inputs:                       0
     Outputs:                      0
```

3 Change the value for the `LogDecimation` property.

```
cs.SolverOptions.LogDecimation = 10;
```

Increasing this setting lets you record fewer data points and decrease run time.

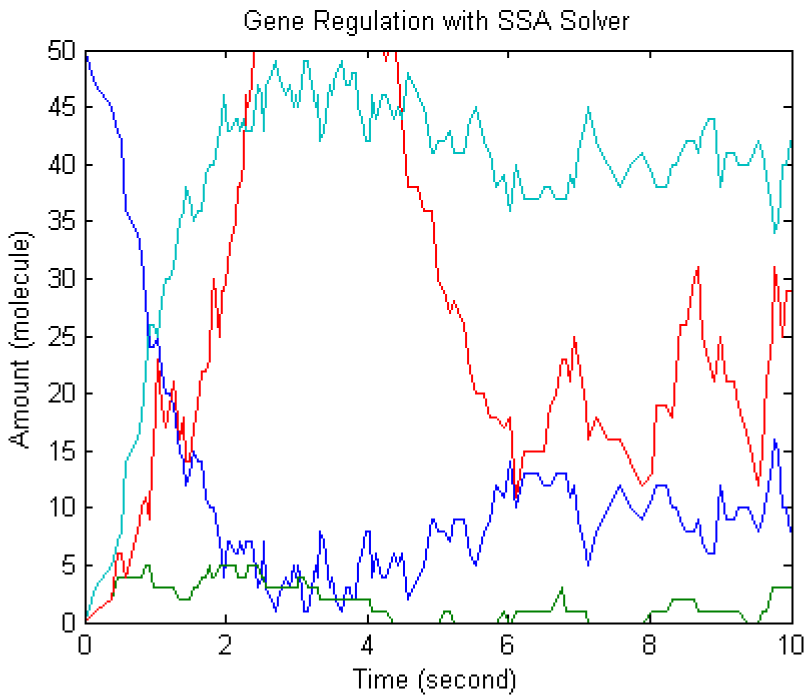
4 Run the simulation.

```
[t_ssa, x_ssa] = sbiosimulate(Mobj, cs);
```

5 Plot the results.

```
FH1 = figure;
set(gcf, 'color', 'white');
plot(t_ssa,x_ssa(:,1:4));
title('Gene Regulation with SSA Solver');
xlabel('Time (second)');
ylabel('Amount (molecule)');
axis([0, 10, 0, 50]);
```

The resulting plot may resemble the following figure.



6 Another method to visualize stochastic simulations is through ensemble runs.

```
simDataObj = sbioensemblerun(Mobj, 10, cs);
sbiosubplot(simDataObj);
```

The resulting plot may resemble the following:

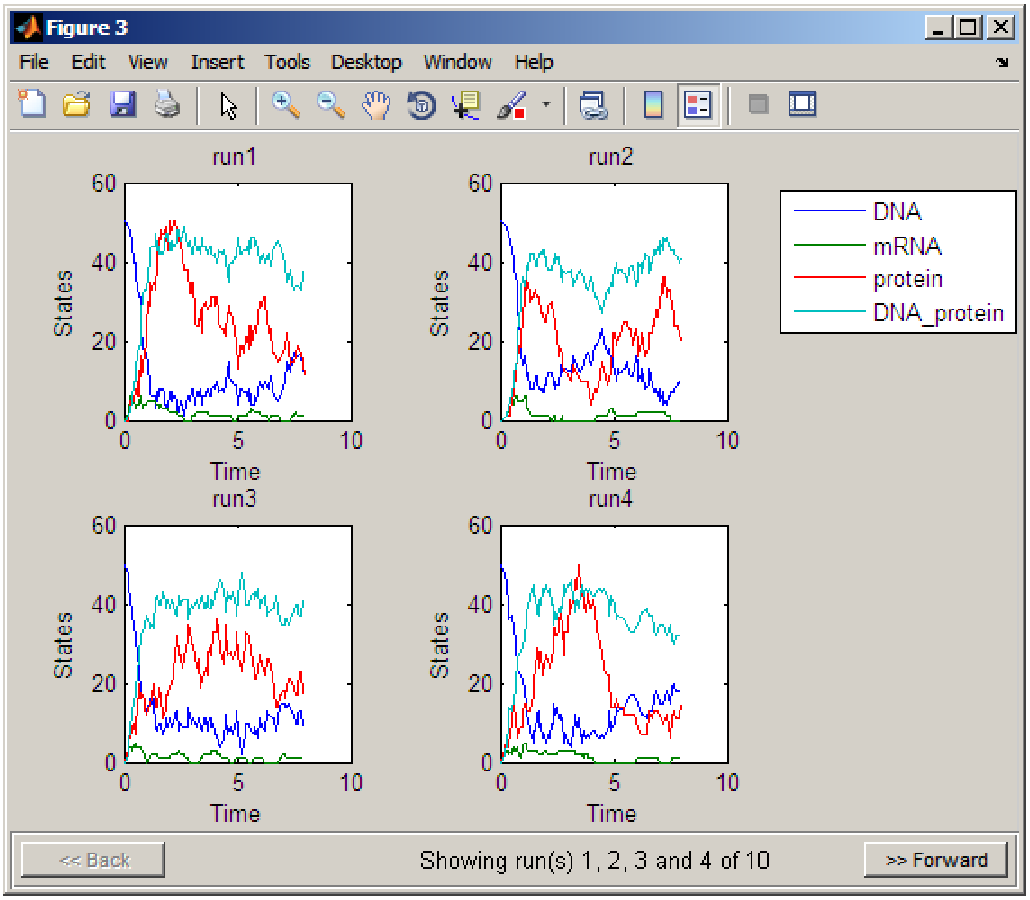


Figure 3
File Edit View Insert Tools Desktop Window Help
run1
run2
run3
run4
States
Time
DNA
mRNA
protein
DNA_protein
<< Back
Showing run(s) 1, 2, 3 and 4 of 10
>> Forward

- Click **Back** or **Forward** to navigate through the plots.

Note: To remove the model object from the **Workspace** type

```
delete(Mobj);
```

See Also

- " Model Simulation"

- "Stochastic Solvers" — Brief description of when to use stochastic solvers and the types available in the software.

Next Steps

"More Examples of Using the Command Line" on page 3-22

More Examples of Using the Command Line

For more examples of using the command line to build a model, simulate, and analyze, see:

- "Create and 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"
- "Verifying a Model"
- "Determining Conserved Moieties"
- "Simulate a Model and View Results"
- "Calculate Sensitivities"
- "Estimate Parameters of a G protein Model"