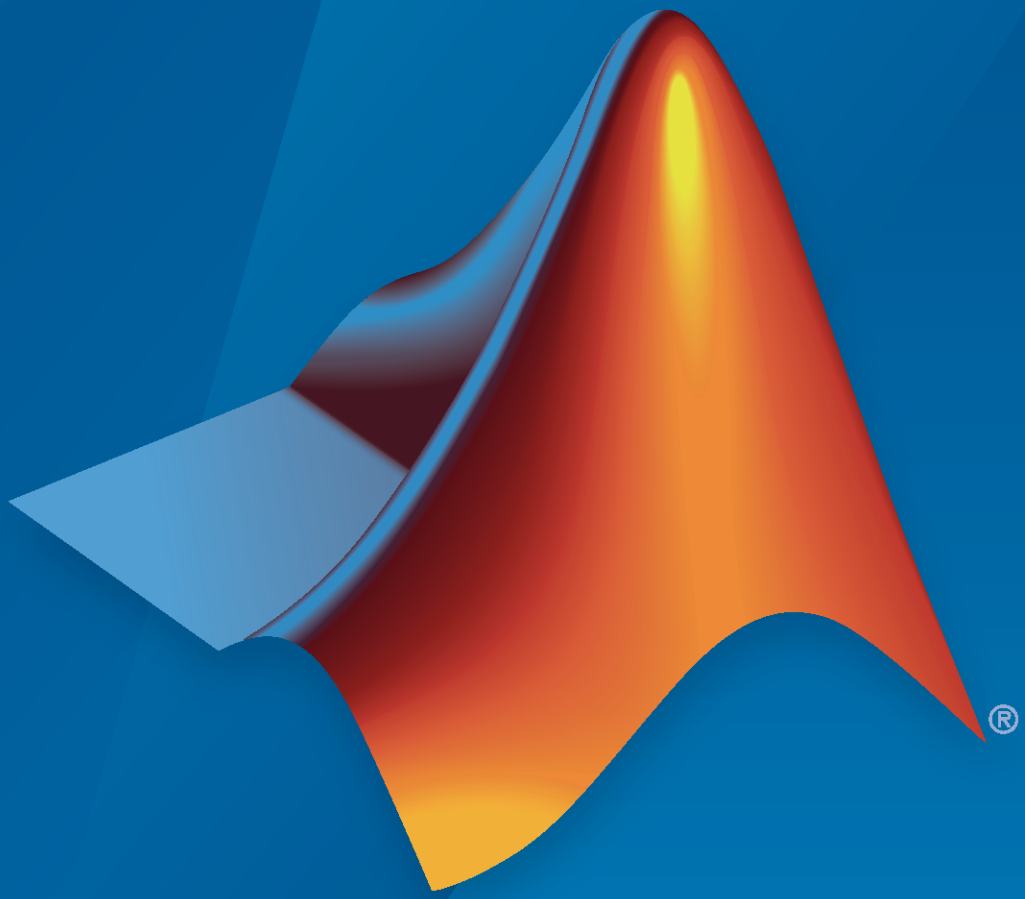
SimBiology®

Getting Started Guide



MATLAB®

R2022a

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.
1 Apple Hill Drive
Natick, MA 01760-2098

SimBiology® Getting Started Guide

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)
March 2019	Online only	Updated for Version 5.8.2 (Release 2019a)
September 2019	Online only	Updated for Version 5.9 (Release 2019b)
March 2020	Online only	Updated for Version 5.10 (Release 2020a)
September 2020	Online only	Updated for Version 6.0 (Release 2020b)
March 2021	Online only	Updated for Version 6.1 (Release 2021a)
September 2021	Online only	Updated for Version 6.2 (Release 2021b)
March 2022	Online only	Updated for Version 6.3 (Release 2022a)

Contents

1 Introduction

2 Getting Started Using the SimBiology Apps

3 Getting Started Using the Command Line

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

Model, simulate, and analyze biological systems

SimBiology provides apps and programmatic tools for modeling, simulating, and analyzing dynamic systems, focusing on quantitative systems pharmacology (QSP), physiologically-based pharmacokinetic (PBPK), and pharmacokinetic/pharmacodynamic (PK/PD) applications. You can build models interactively using the SimBiology block diagram editor or programmatically using the MATLAB® language. Your models can be created from scratch, imported as SBML formatted files, or built on the model examples provided in SimBiology.

SimBiology provides a variety of techniques for analyzing ODE-based models ranging in complexity and size. You can run simulations to assess target feasibility, predict drug efficacy and safety, and identify optimal dosing schedules. You can identify key pathways and parameters using local and global sensitivity analyses and assess biological variability by running parameter sweeps. To estimate parameters you can fit data using nonlinear regression and nonlinear mixed-effects techniques and perform non-compartmental analysis (NCA).

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 Windows®, if you have not installed another compiler, SimBiology uses the lcc-win64 compiler for model accelerations. If you have installed another supported compiler, it will be selected automatically. For better performance of the acceleration functionality, you may want to install a supported compiler other than lcc-win64, and it will be selected automatically.

See Also

More About

- “Simulation”
- “Accelerating Model Simulations and Analyses”

SBML Support

In this section...

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 here.

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 Considerations” 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 and MATLAB Features Not Supported by SBML

Not all SimBiology and MATLAB features are supported by SBML. If your model contains any of those features, SimBiology issues corresponding warnings during the model export. The following SimBiology features are not supported and are not included in the saved SBML format file.

- **Projects** — Models, analysis programs, 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**.
- **Some MATLAB features** — Some of the MATLAB features and language constructs are not supported. For example, elementwise operations, such as `.*`, `./`, character vector such as `'drug'`, string such as `"drug"`, empty bracket `[]` are not supported.

Tip Because the previous information is not supported by SBML, we recommend saving as a SimBiology project file (`.sbproj`) to capture this information.

See Also

`sbmlimport`

Getting Started Using the SimBiology Apps

Getting Started Using the SimBiology Apps

SimBiology provides a set of integrated apps that are designed to facilitate building, simulating, and analyzing models of dynamic systems such as pharmacokinetic/pharmacodynamic (PK/PD) and mechanistic systems biology models.

- The **SimBiology Model Builder** app lets you build dynamic models interactively using various modeling elements. For instance, you can model a variety of biological systems such as signaling pathways, metabolic networks, and PK/PD models. It also lets you model biological variability and different dosing regimens to investigate various experimental conditions and dosing strategies.
- The **SimBiology Model Analyzer** app lets you perform analyses on models of dynamic systems. You can simulate the dynamic behavior of a model using various solvers and estimate model parameters. To investigate system dynamics and guide experimentation, you can perform sensitivity analysis and parameter sweeps.

See the following tutorials to get started.

Build Models

- “Create Model of Receptor-Ligand Kinetics”
- “Incorporate SGLT2 Inhibition into Physiologically Based Glucose-Insulin Model Using SimBiology Model Builder”
- “Undo and Redo Model Changes in SimBiology”
- “Copy SimBiology Blocks”
- “Keyboard Shortcuts for SimBiology Model Builder”
- “SimBiology Model Component Libraries”
- “Message Indicator Icons in SimBiology Model Builder”

Analyze Models and Data

- “Calculate NCA Parameters and Fit Model to PK/PD Data Using SimBiology Model Analyzer App”
- “Find Important Parameters with Local Sensitivity Analysis Using SimBiology Model Analyzer App”
- “Model Biological Variability with Virtual Patients Using SimBiology Model Analyzer App”
- “Scan Dosing Regimens Using SimBiology Model Analyzer App”

Create Standalone Applications

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 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

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 Using sbiosimulate”

Estimation

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

Deployment

- “Deploy a SimBiology Exported 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 -> B`, which follows 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
```

```
ans =
   SimBiology Species Array

   Index:    Compartment:    Name:    Value:    Units:
   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 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 `'k'` so that the reaction rate can be determined.

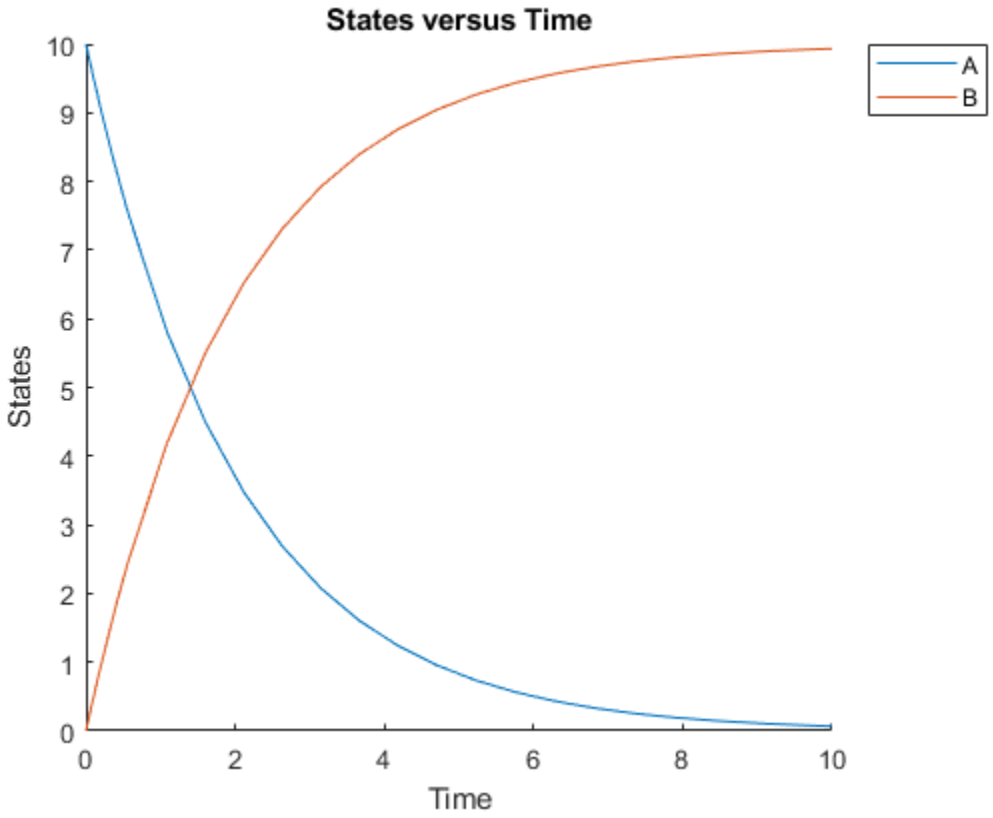
```
p1 = addparameter(kineticLaw,'k',0.5);
kineticLaw.ParameterVariableNames = 'k';
```

Simulate the model.

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

Plot the simulation results.

```
sbioplot(sd);
```



See Also

`sbiomodel` | `addreaction (model)` | `addparameter (model, kineticlaw)` | `addkineticlaw (reaction)` | `sbiosimulate`

Related Examples

- “Getting Started Using the SimBiology Command Line” on page 3-2

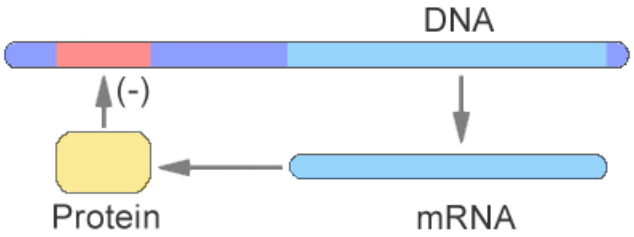
Model a Gene-Regulation Pathway

In this section...

About The Gene Regulation Model

Model Diagram

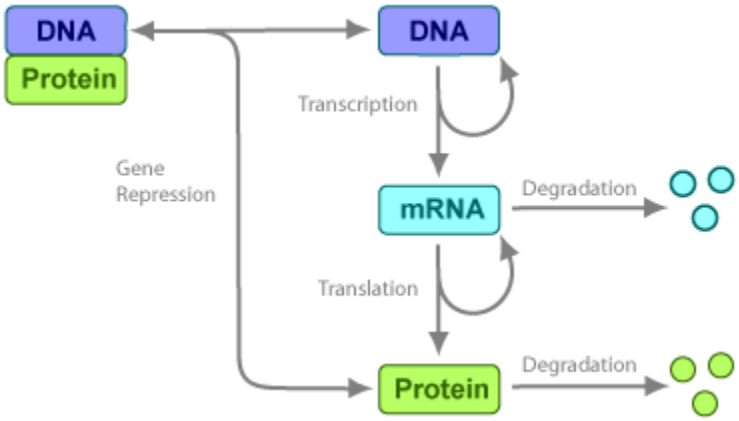
You can visualize a systems biology model with various levels of detail. One view sketches only the major species and processes. This model is an example of simple gene regulation, where the protein product from translation controls transcription. You could create a more complex model by adding the enzymes, 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 -> 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 -> DNA + mRNA`
Reaction rate	`v = k1 * DNA` molecule/second
Species	`DNA = 50` molecule
	`mRNA = 0` molecule
Parameters	`k1 = 0.20` 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 -> mRNA + protein`
Reaction rate	`v = k2 * mRNA` molecule/second
Species	`mRNA` = `0` molecule
	`protein` = `0` molecule
Parameters	`k2` = `20` 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 = k3 * DNA * protein` molecule/second
Species	`DNA` = `50` molecule
	`protein` = `0` molecule
Parameters	`k3` = `0.2` 1/(molecule*second)

Reverse Reaction (Unbinding)

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

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/(second*molecule)
	`k3r` = `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 * mRNA` molecule/second
Species	`mRNA = 0` molecule
Parameters	`k4 = 1.5` 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 * protein` molecule/second
Species	`protein = 0` molecule
Parameters	`k5 = 1` 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`.

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

```
Mobj =
   SimBiology Model - cell

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

```
ans =
   SimBiology Species Array

   Index:    Compartment:    Name:    Value:    Units:
   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 `ParameterVariablesNames` 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 3-6.

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 `k3` as the forward rate constant, and `k3r` 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 `k4` 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).

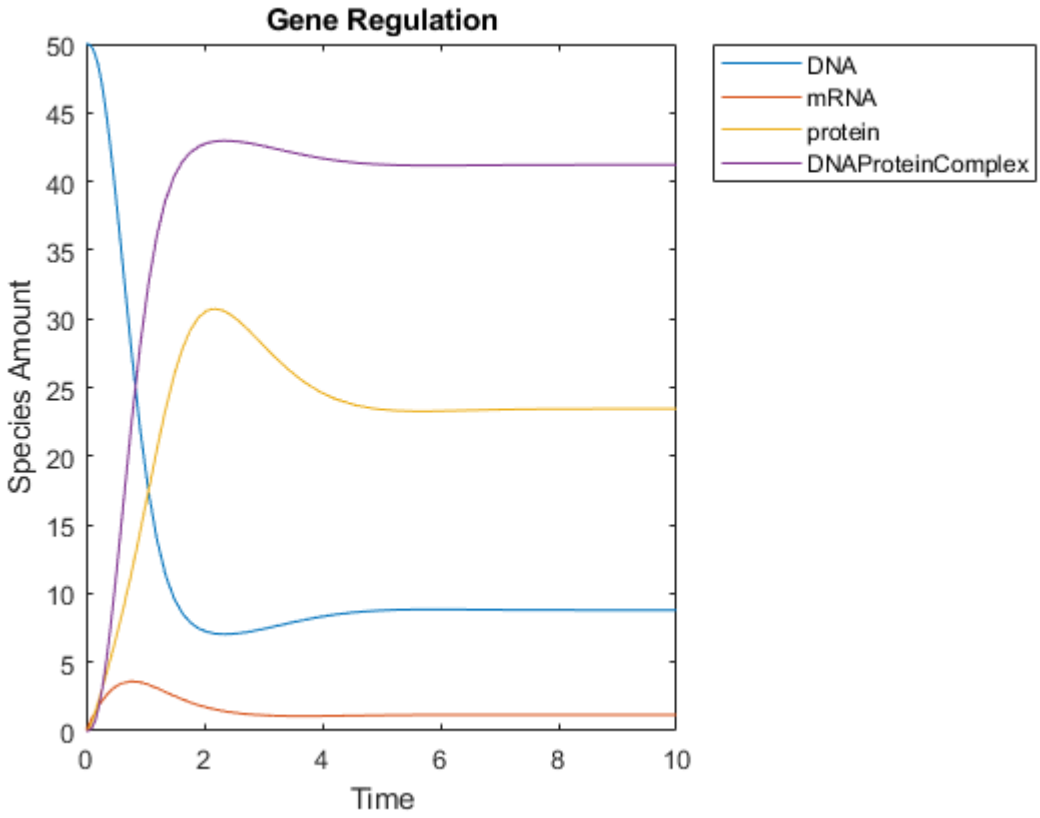
```
configset = getconfigset(Mobj);
configset.CompileOptions.UnitConversion = true;
```

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');
```


Gene Regulation
Species Amount
Time
0
5
10
15
20
25
30
35
40
45
50
0
2
4
6
8
10
DNA
mRNA
protein
DNAProteinComplex