

AZRsim Documentation

AstraZeneca

Contents

User Guide	5
1 Model Specification	7
1.1 Model Name and Model Notes	7
1.2 Model States	8
1.3 Model Parameters	8
1.4 Model Variables	8
1.5 Model Reactions	8
1.6 Model Functions	9
1.7 Model Events	9
2 Modeling Functions	11
2.1 create_model	11
2.2 simulate	12
2.3 plot	12
2.4 shiny_plot	13
3 Examples	15
3.1 Simple Harmonic Oscillator	15
3.2 Michaelis–Menten	17
3.3 Two Compartment Model	18

User Guide

The user guide documents the functionality for the AZRsim R package which can be used to simulate ordinary differential equations in C and NONMEM.

In order to simulate the model in the **AZRsim** package we have to,

1. Create an appropriate text file representation of the model.
2. Compile the model with the `create_model()` function.
3. Simulate the compiled model with the `simulate()` function.

Chapter 1

Model Specification

In order to simulate the time varying variables of a system of ordinary differential equations you need to provide a specific representation of your mathematical model in a text file (a file with extension `.txt`). Every text file representation of a model must contain the following blocks presented exactly as defined below. Notice that each block name is preceded with ten `*`s. It is possible to comment out a line of code using `%`.

```
***** MODEL NAME

***** MODEL NOTES

***** MODEL STATES

***** MODEL PARAMETERS

***** MODEL VARIABLES

***** MODEL REACTIONS

***** MODEL FUNCTIONS

***** MODEL EVENTS
```

The remainder of this section deals with what can be declared within each block.

1.1 Model Name and Model Notes

`Model Name` and `Model Notes` are descriptive blocks that allow the user to specify a model name and any notes about the model. The name of the model will be presented when an `azrmod` object is printed to the console.

Below are the model name and model notes blocks for the simple harmonic oscillator example in the **AZRsim** package (`sho.txt`).

```
***** MODEL NAME

Harmonic Oscillator

***** MODEL NOTES
```

Harmonic Oscillator ODE example.

This meta data can be extracted from an `azrmod` object.

1.2 Model States

The `Model States` block is where the user describes the system of differential equations followed by the initial conditions for each variable. The left hand side of each differential equation *must* be declared as $d/dt(x)$ where x is a variable name chosen by the user.

The initial conditions *must* be defined, and they *must* be defined after the system of differential equations is defined. If variable x in the preceding paragraph had an initial condition value of 3 then we can declare $x(0) = 3$ after defining the system of differential equations.

1.3 Model Parameters

The `Model Parameters` block is where the user defines the values of the parameters in the system of differential equations defined in `Model States`.

1.4 Model Variables

The `Model Variables` block provides a way to reparametrize parts of the model. Variables for intermediate calculations should be defined under `Model Variables`.

```
***** MODEL STATES
```

```
d/dt(A) = -R
```

```
d/dt(B) = R
```

```
A(0) = 1
```

```
B(0) = 0
```

```
***** MODEL PARAMETERS
```

```
k1 = 0.5
```

```
***** MODEL VARIABLES
```

```
R = k1*A
```

1.5 Model Reactions

The `Model Reactions` block provides a way to reparametrize parts of the model. Reaction rates should be defined under `Model Reactions`.

```
***** MODEL STATES
```

```
d/dt(A) = -R
```

```
d/dt(B) = R
```

```
A(0) = 1
```

```
B(0) = 0
```



```
***** MODEL PARAMETERS
```

```
k1 = 0.5
```

```
***** MODEL REACTIONS
```

```
R = k1*A
```

1.6 Model Functions

The `Model Functions` block can be used to define reoccurring calculations.

As an example, the `sho_func.txt` example model in the **AZRsim** package applies the function $f(x) = \frac{x}{100}$ to the variable y_2 in $\frac{d}{dt}y_2$ equation in the system.

```
***** MODEL NAME
```

```
Simple Harmonic Oscillator
```

```
***** MODEL NOTES
```

```
Simple Harmonic Oscillator ODE example.
```

```
***** MODEL STATES
```

```
% ODE system
d/dt(y1) = y2
d/dt(y2) = - y1 - theta * f(y2)
```

```
% initial conditions
```

```
y1(0) = 1
y2(0) = 0
```

```
***** MODEL PARAMETERS
```

```
theta = 0.15
```

```
***** MODEL VARIABLES
```

```
***** MODEL REACTIONS
```

```
***** MODEL FUNCTIONS
```

```
f(x) = x * 0.01
```

```
***** MODEL EVENTS
```

1.7 Model Events

The `Model Events` block can be used to define discrete state events.

There are essentially two parts to defining a discrete event in the model:

1. When does the event take place?

2. What is the event?

Below is an example of a event block.

```
***** MODEL EVENTS
```

```
event1 = eq(time, 1), y, y+40
```

```
event2 = eq(time, 2), y, y+40
```

The term `eq(time, 1)` pertains to (1). This says that the event occurs when the time step is strictly equal to 1. The term `y, y+40` pertains to (2). This is the event that is to take place. In this case the event involves variable `y` and we want to add 40 units of the drug this variable, `y+40`. (Note that events cannot take place at time 0.) In summary, the above block defines adding 40 to the value of `y` at time steps 1 and 2.

We can issue multiple events. The event defined below specifies resetting the value of `x` to 1 and `y` to 0 when `x` is equal to 0.1.

```
***** MODEL EVENTS
```

```
event = eq(x, 0.1), x, 1, y, 0
```

Below is a summary of the options that can be used to declare when an event takes place.

- `eq(a, 0)` implies variable `a` is strictly equal to 0.
- `lq(a, 0)` implies variable `a` is less than or equal to 0.
- `gq(a, 0)` implies variable `a` is greater than or equal to 0.
- `lt(a, 0)` implies variable `a` is strictly less than 0.
- `gt(a, 0)` implies variable `a` is strictly greater than 0.

Chapter 2

Modeling Functions

This section specifies the user-facing functions required to simulate the system of ordinary differential equations.

2.1 create_model

The `create_model` function is used to create the C code based on the text file representation of the model. This function returns an object of class `azrmod`. Once you create a model you can print a summary of the model in R. Below is an example of printing the simple harmonic oscillator, which is part of the **AZRsim** package.

```
library(AZRsim)

sho <- create_model(system.file("examples/sho.txt", package="AZRsim"))
sho
```

```
## AZRmodel
## =====
## Name:                Harmonic Oscillator
## Number States:       2
## Number Parameters:    1
## Number Variables:     0
## Number Reactions:     0
## Number Functions:     0
```

In this model, the system of differential equations involves two states (y_1 and y_2) and one parameter (θ).

The `azrmod` object is a list and we can extract the meta data that was declared in the `Model Name` and `Model Notes` block by using `$` to access the elements within the list. Continuing the above example, below is the code that allows the user print the meta data associated with the `sho.txt` example model to the R console.

```
sho$name

## [1] "Harmonic Oscillator"

sho$notes

## [1] "Harmonic Oscillator ODE example."
```

2.2 simulate

Once you have created an `azrmod` object with `create_model` you can use the `simulate` generic to simulate the state variables over a specified number of time steps. Note that the `simulate` function requires an object of class `azrmod`, and will return an object of class `azrsim` and `data.frame`.

In the simple harmonic oscillator example above we have two states, so calling `simulate(sho, seq(1, 100, by=0.1))` will provide a simulation of these two states over the time sequence 1 to 100 at 0.1 intervals.

```
sho_sim <- simulate(sho, seq(1, 100, by=0.1))
head(sho_sim)
```

```
##   TIME      y1      y2
## 1  1.0 1.0000000 0.0000000
## 2  1.1 0.9950290 -0.09908659
## 3  1.2 0.9802654 -0.19571796
## 4  1.3 0.9560004 -0.28896987
## 5  1.4 0.9226146 -0.37796711
## 6  1.5 0.8805758 -0.46189098
```

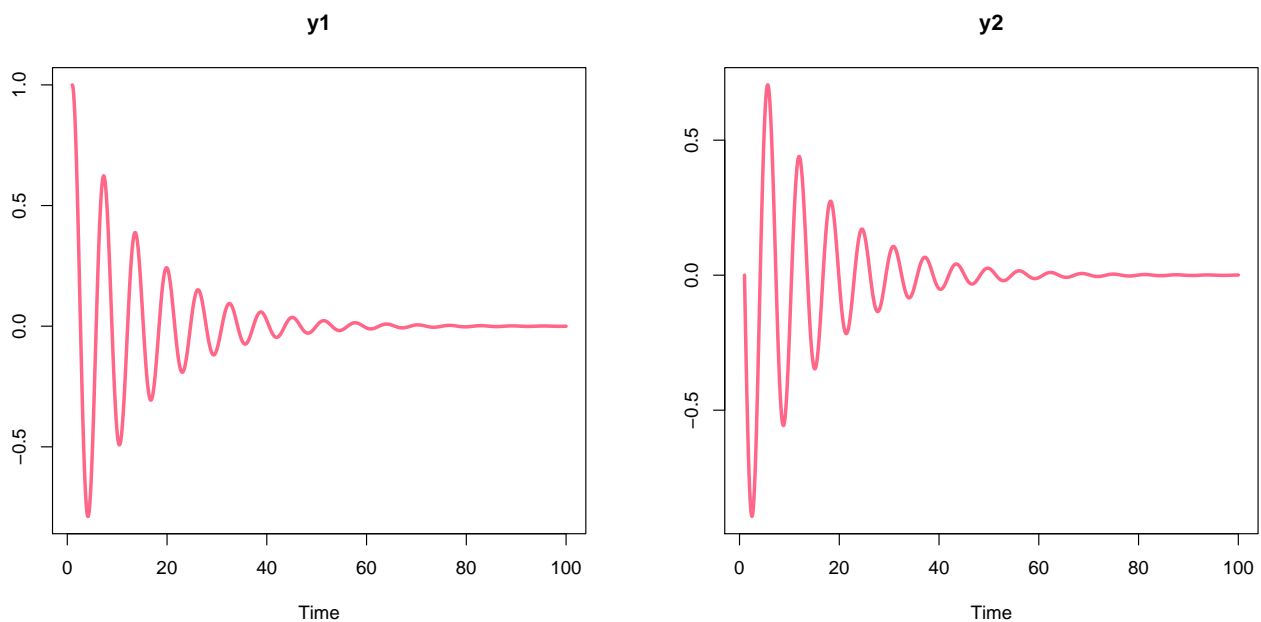
```
dim(sho_sim)
```

```
## [1] 991  3
```

2.3 plot

The `plot` generic can be called on an `azrsim` object and, by default, will construct a lattice plot of the first 9 parameters over time.

```
plot(sho_sim, col = "#FF6688", lwd = 3)
```



2.4 shiny_plot

The `shiny_plot` function must be called on an `azrsim` object and will launch a shiny app that will allow the user to interactively include/exclude any state variable estimated.

Chapter 3

Examples

3.1 Simple Harmonic Oscillator

The equations that govern a harmonic oscillator are the following,

$$\begin{aligned}\frac{d}{dt}y_1 &= y_2 \\ \frac{d}{dt}y_2 &= -y_1 - \theta y_2\end{aligned}$$

where y_1 , y_2 are state variables and θ is a parameter in this model. The text file representation of the model is provided below.

```
***** MODEL NAME

Harmonic Oscillator

***** MODEL NOTES

Harmonic Oscillator ODE example.

***** MODEL STATES

d/dt(y1) = y2
d/dt(y2) = - y1 - theta * y2

y1(0) = 1
y2(0) = 0

***** MODEL PARAMETERS

theta = 0.15

***** MODEL VARIABLES

***** MODEL REACTIONS
```

```
***** MODEL FUNCTIONS
```

```
***** MODEL EVENTS
```

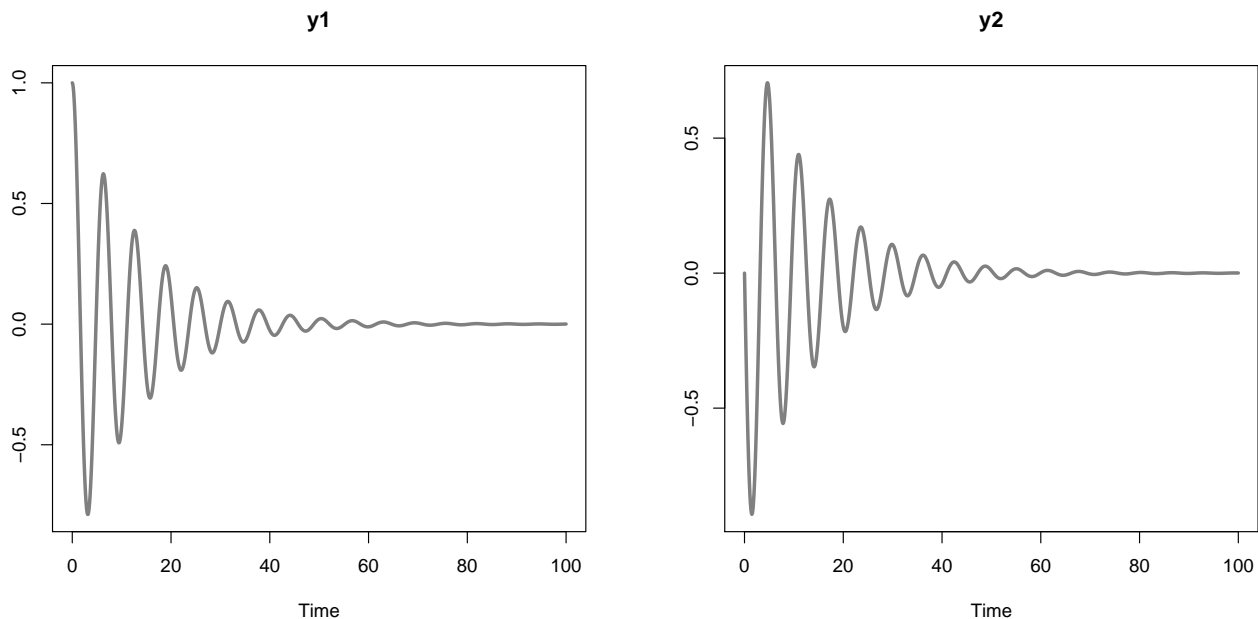
Compiling and simulating the model.

```
sho_model <- create_model(system.file("examples/sho.txt", package="AZRsim"))
sho_sim <- simulate(sho, seq(0, 100, by=0.1))
head(sho_sim)
```

```
##   TIME      y1      y2
## 1  0.0 1.000000 0.000000
## 2  0.1 0.9950291 -0.09908663
## 3  0.2 0.9802655 -0.19571802
## 4  0.3 0.9560005 -0.28896993
## 5  0.4 0.9226147 -0.37796717
## 6  0.5 0.8805759 -0.46189106
```

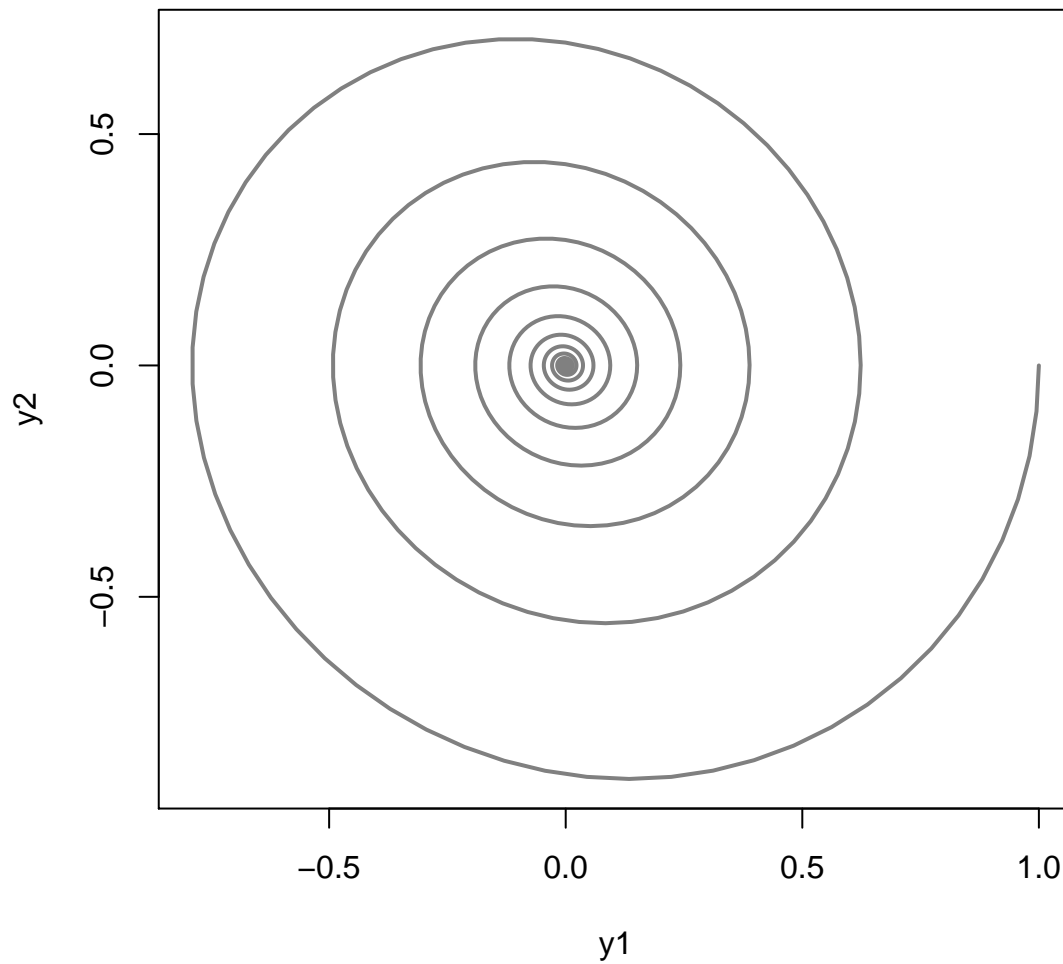
Plotting variables as a function of time.

```
plot(sho_sim, col = "#808080", lwd = 3)
```



Plotting the variables as a function of each other.

```
plot(sho_sim[, "y1"], sho_sim[, "y2"], type = "l", col = "#808080", lwd = 2, xlab = "y1", ylab = "y2")
```

3.2 Michaelis-Menten

***** MODEL NAME

model_1cpt_MM

***** MODEL NOTES

One compartmental distribution model with linear and saturable elimination and with inputs allowing to realize first order absorptions into central compartment and direct administration into central compartment.

***** MODEL STATES

$\frac{d}{dt}(Ad) = -ka*Ad + Fabs1*INPUT1$
 $\frac{d}{dt}(Ac) = ka*Ad - CL/Vc*Ac - VMAX*Cc/(KM+Cc) + INPUT2$

Ad(0) = 0

Ac(0) = 0

```

***** MODEL PARAMETERS

Fabs1      = 1      % Bioavailability
ka         = 0.2     % Absorption rate parameter
CL         = 0.4     % Clearance
Vc         = 12     % Central volume
VMAX       = 5      % Maximum rate of elimination of nonlinear term
KM         = 0.1    % Value of concentration at which half the maximum nonlinear elimination is achieved

***** MODEL VARIABLES

% Calculation of concentration in central compartment
Cc         = Ac/Vc

% Defining an output (only needed when interfacing with NLME
% parameter estimation tools such as NONMEM and MONOLIX)
OUTPUT1    = Cc

***** MODEL REACTIONS

***** MODEL FUNCTIONS

***** MODEL EVENTS

```

3.3 Two Compartment Model

Here we consider a simple two compartment model where, for example, we model the concentration of an oral drug in the intestines and in the blood. The ODE system takes the following form,

$$\begin{aligned}\frac{d}{dt}y_1 &= -a \cdot y_1 + u_t \\ \frac{d}{dt}y_2 &= a \cdot y_1 - b \cdot y_2\end{aligned}$$

where y_1 denotes the concentration of the drug in the intestine, y_2 denotes the concentration of the drug in the blood, a and b are parameters, and u_t denotes the uptake of the drug at time period t .

The text file representation of the model above is provided below.

```

***** MODEL NAME

Simple Two Compartment Dosing Model

***** MODEL NOTES

A simple two compartment dosing model where
ten equally spaced doses are given to the subject.

***** MODEL STATES

```

```

d/dt(y1) = -a * y1
d/dt(y2) = a * y1 - b * y2

y1(0) = 0
y2(0) = 0

***** MODEL PARAMETERS

a = 6
b = 0.6

***** MODEL VARIABLES

***** MODEL REACTIONS

***** MODEL FUNCTIONS

***** MODEL EVENTS

% adding 40 units at a specified time step
event1 = eq(time, 1), y1, y1+2
event2 = eq(time, 2), y1, y1+2
event3 = eq(time, 3), y1, y1+2
event4 = eq(time, 4), y1, y1+2
event5 = eq(time, 5), y1, y1+2
event6 = eq(time, 6), y1, y1+2
event7 = eq(time, 7), y1, y1+2
event8 = eq(time, 8), y1, y1+2
event9 = eq(time, 9), y1, y1+2

```

Below the model is compiled and simulated and the 10 initial and final values are presented.

```
two_cpt <- create_model(system.file("examples/two_cpt.txt", package="AZRsim"))
```

```
## Warning in check_azrmod(model): check_azrmod: AZRmodel contains parameter
## names with a single character name. Try to avoid that if you plan to use
## NONMEM or MONOLIX
```

```
two_cpt_simulation <- simulate(two_cpt, simtime = 10)
head(two_cpt_simulation, n = 10)
```

```
##      TIME y1 y2
## 1  0.00  0  0
## 2  0.01  0  0
## 3  0.02  0  0
## 4  0.03  0  0
## 5  0.04  0  0
## 6  0.05  0  0
## 7  0.06  0  0
## 8  0.07  0  0
## 9  0.08  0  0
## 10 0.09  0  0
```

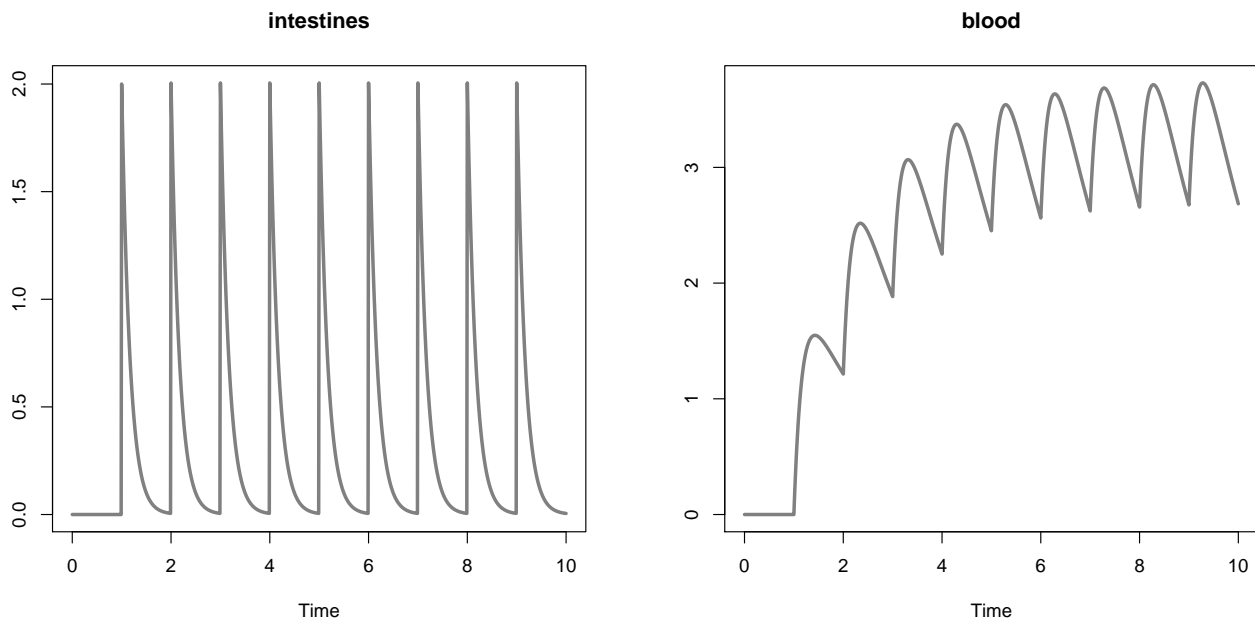
```
tail(two_cpt_simulation, n = 10)
```

```
##      TIME      y1      y2
## 992   9.91 0.008530545 2.830656
```

```
## 993  9.92 0.008033781 2.814218
## 994  9.93 0.007565934 2.797850
## 995  9.94 0.007125330 2.781552
## 996  9.95 0.006710385 2.765326
## 997  9.96 0.006319605 2.749174
## 998  9.97 0.005951668 2.733095
## 999  9.98 0.005605222 2.717091
## 1000 9.99 0.005279009 2.701162
## 1001 10.00 0.004971828 2.685310
```

Using the `plot` generic we can plot the simulation to visualize the concentration of the drug in both compartments over time.

```
plot(two_cpt_simulation, col = "#808080", lwd = 3, plot_names = c("intestines", "blood"))
```



Below we simulate the model with a different parameter value for b .

```
two_cpt_simulation2 <- simulate(two_cpt, simtime = 10, parameters = c("a" = 6, "b" = 1))
plot(two_cpt_simulation2, col = "#808080", lwd = 3, plot_names = c("intestines", "blood"))
```

