Chapter 7

Examples for Phase Shifting

7.1 Introduction

This chapter is a tutorial on using the SBSI frameworks to optimise a model in which deterministic discrete events occur. Examples include the modelling of experiment conditions such as light or temperature entrainment.

There are two ways in which we may model a change in the model during the time course of the numerical analysis. The first uses explicit events whilst the second uses our ability to define functions which operate over the simulated time.

We begin with the trivial example model:

```
k1 = 0.1
A -> B at rate A * k1
```

TODO: show whole model source

Our aim is to simulate three phases each of 20 time units to a final time of 60. In the first phase we wish for our parameter $k1$ to be equal to 0.1, in the second this switches to 0.001 and in the final phase we switch the value of $k1$ back to its original value.

7.2 Using Explicit Events

Explicit events are supported in SBML, all events are defined within a `listOfEvents` element using the `event` tag. An `event` contains a `trigger` and a list of event assignments. For more information please see the SBML specification at: `www.sbml.org`. Below is the particular code for our model and phases that we wish to model. Note that there are two events for the three phases because there are only two phase switches.

```
<listOfEvents>
  <event>
    <trigger>
      <math xmlns="http://www.w3.org/1998/Math/MathML">
        <apply>
```

```
          <gt/>
          <csymbol encoding="text"
                   definitionURL="http://www.sbml.org/sbml/symbols/time"> t
          </csymbol>
          <cn>20.0</cn>
        </apply>
      </math>
    </trigger>
    <listOfEventAssignments>
      <eventAssignment variable="k1">
        <math xmlns="http://www.w3.org/1998/Math/MathML">
          <cn>0.001</cn>
        </math>
        </eventAssignment>
      </listOfEventAssignments>
  </event>
  <event>
    <trigger>
      <math xmlns="http://www.w3.org/1998/Math/MathML">
        <apply>
          <gt/>
          <csymbol encoding="text"
                   definitionURL="http://www.sbml.org/sbml/symbols/time"> t
          </csymbol>
          <cn>40.0</cn>
        </apply>
      </math>
    </trigger>
    <listOfEventAssignments>
      <eventAssignment variable="k1">
        <math xmlns="http://www.w3.org/1998/Math/MathML">
          <cn>0.1</cn>
        </math>
        </eventAssignment>
      </listOfEventAssignments>
 </event>
<</listOfEvents>
```

Important note: in the definition of any parameter altered by an event assignment, the ‘constant’ attribute must be explicitly set to ‘false’. As in the above definition:

```
<listOfParameters>
  <parameter id="k1" value="0.01" constant="false" />
</listOfParameters>
```

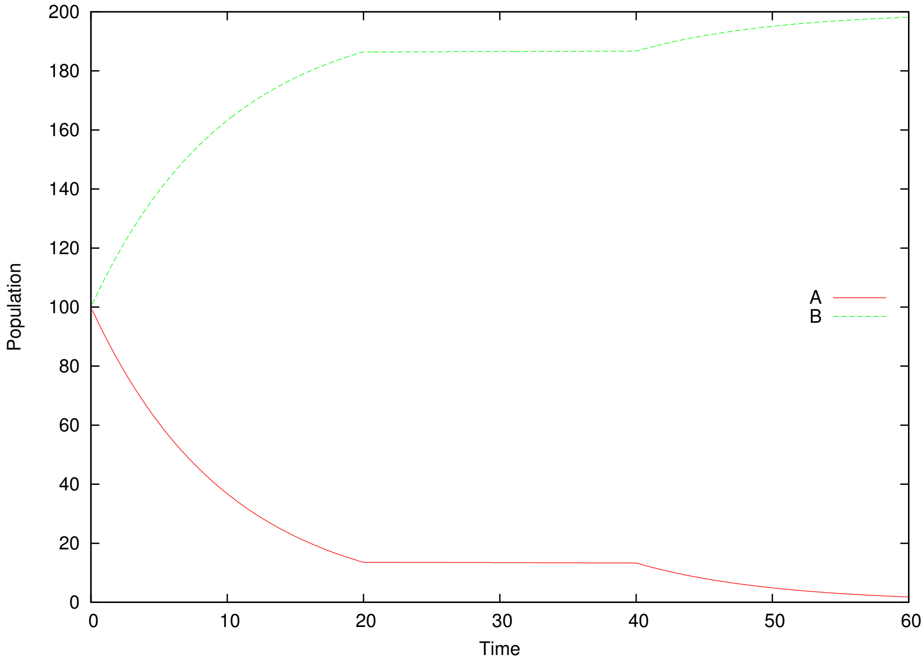

Figure 7.1: The results of numerical analysis over a model containing phase switches at times 20 and 40.

The results of evaluating this model are shown in Figure 7.1.

7.3 Using Time-Dependent Rate Functions

Often the model is not authored in SBML, but is authored in another modelling framework such as Bio-PEPA and the SBML automatically generated. In this situation it may be awkward, if the particular framework involved does not support SBML events, to generate the SBML and then manually add the required events. In this case it may be more convenient to use a time-dependent rate function.

In Bio-PEPA to achieve the same scenario as that above we would write the following rate function:

```
r = [ A * ( ( (H(20 - time) + H(time - 40)) * 0.1) +
            ( (H(time - 20) * H(40 - time)) * 0.001) )
    ];
```

To understand this we must understand first of all the H function in Bio-PEPA. This function returns 1 if the argument is greater than zero and 0 otherwise. So the first call to the H function, $H(20 - time)$ will be 1 if the current time is less than 20. We add this to the result of calling $H(time - 40)$ which checks that the current time is above 40. Hence the addition of these will be 1 if we are in the first or third phase. This is then multiplied by the rate at which we wish 'k1' to be, if we are in the first or third phases. The next two calls are $H(time - 20)$ and $H(40 - time)$ check that we are within the bounds of the second phase, since the first checks that the time is above 20 and the second that it is below 40. Since the results of these two calls are multiplied together the overall result will be 1 only if both of these hold true and we are in fact in the second phase. The 1 or 0 is then multiplied by the rate which we wish the 'k1' parameter to take during the second phase. These two rates are then multiplied together and the entire rate expression is equal to A multiplied by the desired value of 'k1' depending on which phase we are in.

For other formalisms we can write the helper SBML function definitions:

```
<functionDefinition id="phaseZeroToX">

<math xmlns="http://www.w3.org/1998/Math/MathML">
  <lambda>
    <bvar><ci>r</ci></bvar>
    <bvar><ci>X</ci></bvar>
    <bvar><ci>t</ci></bvar>
    <piecewise>
      <piece>
        <ci>r</ci>
        <apply>
          <gt/>
          <ci>X</ci>
          <ci>t</ci>
        </apply>
      </piece>
      <otherwise>
        <cn>0</cn>
      </otherwise>
    </piecewise>
  </lambda>
</math>
</functionDefinition>

<functionDefinition id="phaseXToInfinity">

<math xmlns="http://www.w3.org/1998/Math/MathML">
  <lambda>
    <bvar><ci>r</ci></bvar>
    <bvar><ci>X</ci></bvar>
    <bvar><ci>t</ci></bvar>
    <piecewise>
      <piece>
```

```
        <ci>r</ci>
        <apply>
          <gt/>
          <ci>t</ci>
          <ci>X</ci>
        </apply>
      </piece>
      <otherwise>
        <cn>0</cn>
      </otherwise>
    </piecewise>
  </lambda>
</math>
</functionDefinition>

<functionDefinition id="phaseXToY">

<math xmlns="http://www.w3.org/1998/Math/MathML">
  <lambda>
    <bvar><ci>r</ci></bvar>
    <bvar><ci>X</ci></bvar>
    <bvar><ci>Y</ci></bvar>
    <bvar><ci>t</ci></bvar>
    <piecewise>
      <piece>
        <piecewise>
          <piece>
            <ci>r</ci>
            <apply>
              <gt/>
              <ci>t</ci>
              <ci>X</ci>
            </apply>
          </piece>
          <otherwise>
            <cn>0</cn>
          </otherwise>
        </piecewise>
        <apply>
          <lt/>
          <ci>t</ci>
          <ci>Y</ci>
        </apply>
```

```
      </piece>
      <otherwise>
        <cn>0</cn>
      </otherwise>
    </piecewise>
  </lambda>
</math>
</functionDefinition>
```

7.4 Periodical Phase Switches

7.5 Optimisation

The previous two sections have shown how the numerical analysis of an SBML model can react to events which alter parameters within the model. In this section we combine this with optimisation to optimise for two parameters which control the rate of a reaction in different phases of the time course.