deBif

A package for bifurcation analysis of ordinary differential equation systems

André M. de Roos

Institute for Biodiversity and Ecosystem Dynamics
University of Amsterdam

A.M.deRoos@uva.nl

Version March 10, 2022

Contents

Preface

This software package is developed for equilibrium and bifurcation analysis of systems of ordinary differential equations (ODEs). If you are not familiar with the bifurcation theory of continuous-time systems described in terms of ODEs the book by Kuznetsov (1995) is an excellent source to consult that I highly recommend. Many of the numerical procedures implemented in this package are taken from that book.

This package consists of two functions, called `phaseplane()` and `bifurcation()`, that both start up a `shiny` application (Chang et al., 2019). `Shiny` is a web application framework for R, which also allows to run an application on your local machine. I used `shiny` (Chang et al., 2019) and the related packages `shinydashboard` (Chang and Borges Ribeiro, 2018) and `shinydashboardPlus` (Granjon, 2019) to create a graphical user interface. Under the hood, I use the packages `deSolve` (Soetaert et al., 2010) for time integration of the ODEs and the package `rootSolve` (Soetaert, 2009) as well as my own implementation of a root finding function for the numerical solution of equilibrium and bifurcation conditions.

The package is free software and released under the GNU General Public License without any warranty or even the implied warranty of merchantability or fitness for a particular purpose (the official statement of the GPL). If you are using the software for publications, you are kindly asked to credit this software package by a reference to this documentation and the website that hosts the software package, as these are currently the only sources to be referred to.

In case you encounter any problem with the software package, please first verify the problem is not in your own model-specific file, but indeed is a bug in the general software layer. If you are convinced it is a bug in my programming, send me an email with as accurate a description of the problem as possible. Do not not forget to include your model-specific file and details about how to trigger the problems. Any comments and feedback, both on the code and on the current manual is appreciated and will be considered carefully. In particular concrete comments, for example, explicit suggestions for textual changes in the manual and/or corrections of the mistakes (they are definitely there!) will be highly valued and acknowledged.

1 Model implementation

The implementation of a model is identical to the implementation used in the package `deSolve` (Soetaert et al., 2010) for systems of ODEs. The model implementation has 3 elements: a vector of state variables, a vector of parameters and the specification of the right-hand side of the ODEs. Both the vector of state variables and the vector of parameters, should consist of named vector elements only. The right-hand sides of the ODEs have to be specified in a R function that has the following layout:

```
model <- function(t, state, parms) {
  with(as.list(c(state,parms)), {
    #
    # Define your derivatives here as a list. For example:
    #
    # derivatives <- list(c(dS1, dS2))
    #
    # where dS1 and dS2 are variables to be assigned the
    # values of state variable derivatives
    #
    return(derivatives)
  })
}
```

The name of the function (here `model`) can be chosen freely. The returned variable `derivatives` should be a list of derivative values that specify the right-hand side of the ODEs.

Notice, the order of these derivatives should be identical to the order of the state variables that are contained in the variable `state`.

After defining the right-hand side of the ODEs the functions `phaseplane()` and `bifurcation()` can be invoked using the commands:

```
phaseplane(model, state, parms)
```

and

```
bifurcation(model, state, parms)
```

in which `model` is the name of the function that specifies the right-hand side of the ODEs, `state` should be the name of the vector with values for the state variables and `parms` should be the name of the vector with values for the parameters.

1.1 Model implementation example

As an example, consider the Rosenzweig-MacArthur model for the interaction between a predator and a prey:

$$\frac{dR}{dt} = rR\left(1 - \frac{R}{K}\right) - \frac{aR}{1 + ahR}C$$
$$\frac{dC}{dt} = \epsilon\frac{aR}{1 + ahR}C - \mu C$$

with variables R and C representing the densities of the prey and predator, respectively. The parameters in this model are the prey per-capita growth rate r and its carrying capacity K, the attack rate of the predator a, its handling time h, conversion efficiency ϵ and its mortality rate μ.

To implement this model first a named vector of values for the state variables has to be defined:

```
state <- c(R = 0.05, C = 0.1)
```

and similarly a named vector of default values for the parameters:

```
parms <- c(r = 0.5, K = 0.1, a = 5.0, h = 3.0, eps = 0.5, mu = 0.05)
```

The named elements of the state variable vector and the parameter vector can now be used in the function that described the right-hand side of the ODEs:

```
rosenzweig <- function(t, state, parms) {
  with(as.list(c(state,parms)), {

    dR = r*R*(1 - R/K) - a*R*C/(1 + a*h*R)
    dC = eps*a*R*C/(1 + a*h*R) - mu*C

    return(list(c(dR, dC)))
  })
}
```

To carry out bifurcation analysis of the Rosenzweig-MacArthur model implemented above an R script can be created with the following content:

```
# The initial state of the system has to be
# specified as a named vector of state values.
state <- c(R = 0.05, C = 0.1)

# Parameters has to be specified as a named vector of parameters.
parms <- c(r = 0.5, K = 0.1, a = 5.0, h = 3.0, eps = 0.5, mu = 0.05)

# The model has to be specified as a function that returns
# the derivatives as a list. You can adapt the body below
# to represent your model
rosenzweig <- function(t, state, parms) {
  with(as.list(c(state,parms)), {

    dR = r*R*(1 - R/K) - a*R*C/(1 + a*h*R)
    dC = eps*a*R*C/(1 + a*h*R) - mu*C
```

```
    # The order of the derivatives in the returned list has to be
    # identical to the order of the state variables contained in
    # the argument `state`
    return(list(c(dR, dC)))
  })
}

bifurcation(rosenzweig, state, parms)
```

Running this script will define the vectors of the state variables and parameters as well as the function defining the system of ODEs and will start up the `shiny` application for bifurcation analysis.

The above script is in fact included in the `deBif` package as an example. It can be started up by issuing the command `deBifExample("rosenzweig")`. The function `deBifExample()` provides a list of examples included in the package when the function is called without any argument and executes an example script if the argument provided to `deBifExample()` refers to an included example.

2 The phaseplane() application

After starting the phaseplane() application a new window opens up, which is shown in Figure 2.1. The application contains a sidebar at the left, which is used for:

- specifying initial values for the state variables
- specifying initial values for the parameters
- Starting computations from the chosen initial state with the selected parameters
- loading, saving and deleting curves that are computed with the phaseplane() application.

The remainder of the application window consists of an area with up to 6 tab sheets, called Time series, Nullclines, Steady states, Vector field, Trajectories and Portrait. Only the first 3 tab sheets (Time series, Nullclines and Steady states) are shown if the system of ODEs consists of a single ODE. Switching between these tabs is achieved by clicking on the appropriate tab name. Each tab sheet includes a plot frame and a frame in which progress messages as well as errors will be reported. Notice that the plot frame differs between the different tab sheets.

In the following the Rosenzweig-MacArthur model presented in section 1.1 will be used to illustrate the different types of curves that can be computed and the different menus to adjust the computations and the graphical presentation of their results.

2.1 Time series

The first tab shows curves of the time dynamics as predicted by the system of ODEs that is being investigated. The dynamics are computed numerically through integration after pressing either the <<Compute button or the >>Compute in the left sidebar (see Figure 2.1). The program will integrate the dynamics from $t = 0$ till the maximum integration time when the >>Compute button is pressed. See section 2.1.3 for how to change the value of the maximum integration time. When the <<Compute button is pressed the integration will start at the maximum integration time and attempt to integrate backward in time till $t = 0$.

Numerical integration of the system of ODEs forward in time should always proceed without problems (if at least you specified the system of ODEs correctly), but numerical integration of the dynamics backward in time may cause problems as the values of the state variables in the ODEs may approach infinite values. Therefore, be careful with numerical integration backward in time, it is advisable to compute over a small time interval only. Hence, choose the value of the maximum integration time (see section 2.1.3) not too large.

Every time the <<Compute or the >>Compute button is pressed a new time series curve is computed from the current initial state with the current parameters. All computed curves are shown in the graph. Each curve furthermore gets a label and the first and last point of the curve are labeled as special points. These special points are added to the drop-down menu at the top of the left sidebar. Figure 2.2 shows the application window with two time series that have been computed from different initial states, while the special points drop-down menu is expanded. The drop-down menu shows the first and last point of the two computed time series

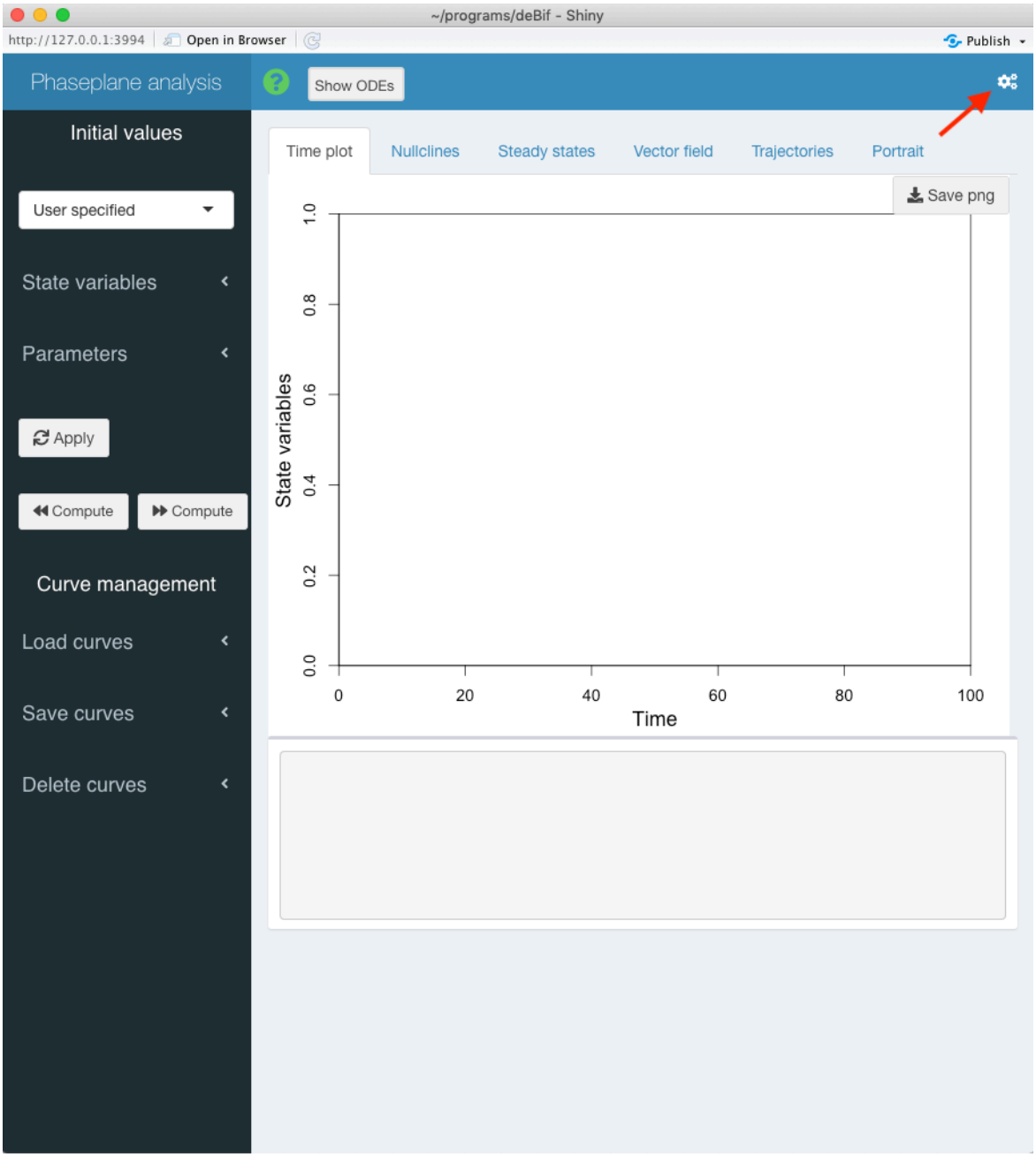

Figure 2.1: Starting view of the 'phaseplane()' application. The red arrow indicates the gears icon that opens up the right sidebar.

(started from $R = 0.05, C = 0.1$ and $R = 0.05, C = 0.5$, respectively). These special points can be selected using the special points drop-down menu, in which case the values of the state variables and parameters of the special point are used as starting point for the next time series computation.

By default the plot in this tab shows the time variable on the *x*-axis and the values of all state variables defined in the model on the *y*-axis. Which state variables are used as *y*-coordinates in the plot can be customized as discussed in section 2.1.2.

2.1.1 Changing state variables and parameters

Clicking on the `State variables` menu title in the left sidebar (see Figure 2.1) expands a sub-menu with a list of names of all state variables in the system of ODEs and input boxes that contain current values of these state variables. New values for the various state variables can be specified by editing the values in the input boxes. These new values take effect as soon as the `Apply` button is pressed.

Similarly, clicking on the `Parameters` menu title expands a sub-menu showing a list of names of all parameters and input boxes that contain the current values of the system parameters. New values of the parameters can be specified by editing the values in the input boxes. Also these new values only take effect once the the `Apply` button is pressed.

2.1.2 Customizing the plot

Curves of the model dynamics that will be shown in the time series plot include as variables the time and the values of all state variables at each time. The plot frame on the `Time series` tab sheet always shows the independent time variable on the *x*-axis with the *x*-axis ranging from 0 to 100. On the *y*-axis this plot frame shows by default the values of all state variables with the *y*-axis from 0 to 1. These default settings can be customized using the right sidebar, which can be shown by pressing the gears icon in the right-top corner of the application window (see Figure 2.1).

Figure 2.3 shows the `phaseplane()` application window with the right sidebar visible. The drop-down menus allow for selecting which variables are used as *x*- and *y*-coordinates in the time series plots. As *x*-coordinate only the time variable can be selected. As *y*-coordinate it is possible to select all state variables, in which case as many curves will be plotted as there are state variables defined in the model. All state variables will then be plotted using the same *y*-axis at the left of the plot. The boxes and drop-down menus in the right sidebar allow for customization of the minimum and maximum values on the *x*- and *y*-axis as well as whether a linear of logarithmic axis scale should be used.

Alternatively, when a single state variable is selected as *y*-coordinate the right sidebar extends and offers the possibility to add a second *y*-axis on the right-hand side of the plot, on which the value of another state variable can be plotted. If a second state variable is selected to be plotted on the second *y*-axis at the right-hand side of the plot the right sidebar panel extends even further to allow for specification of the minimum and maximum value and the scale type to use on this second *y*-axis.

If state variables are selected to be plotted on both the primary and the secondary *y*-axis, the

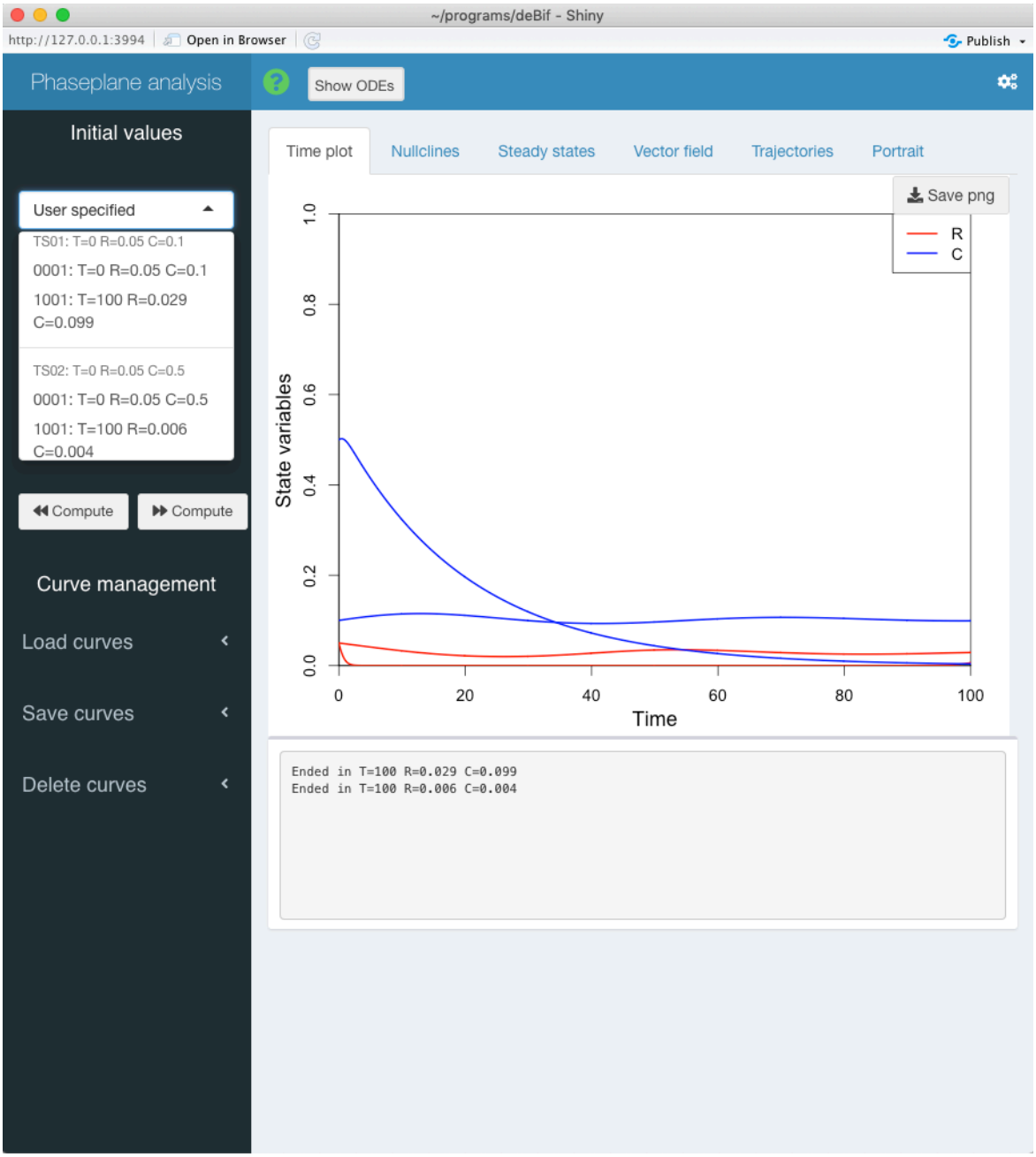

Figure 2.2: Two computed time series are shown in the 'phaseplane()' application and the special points menu is expanded, listing the first and last points of the two curves.

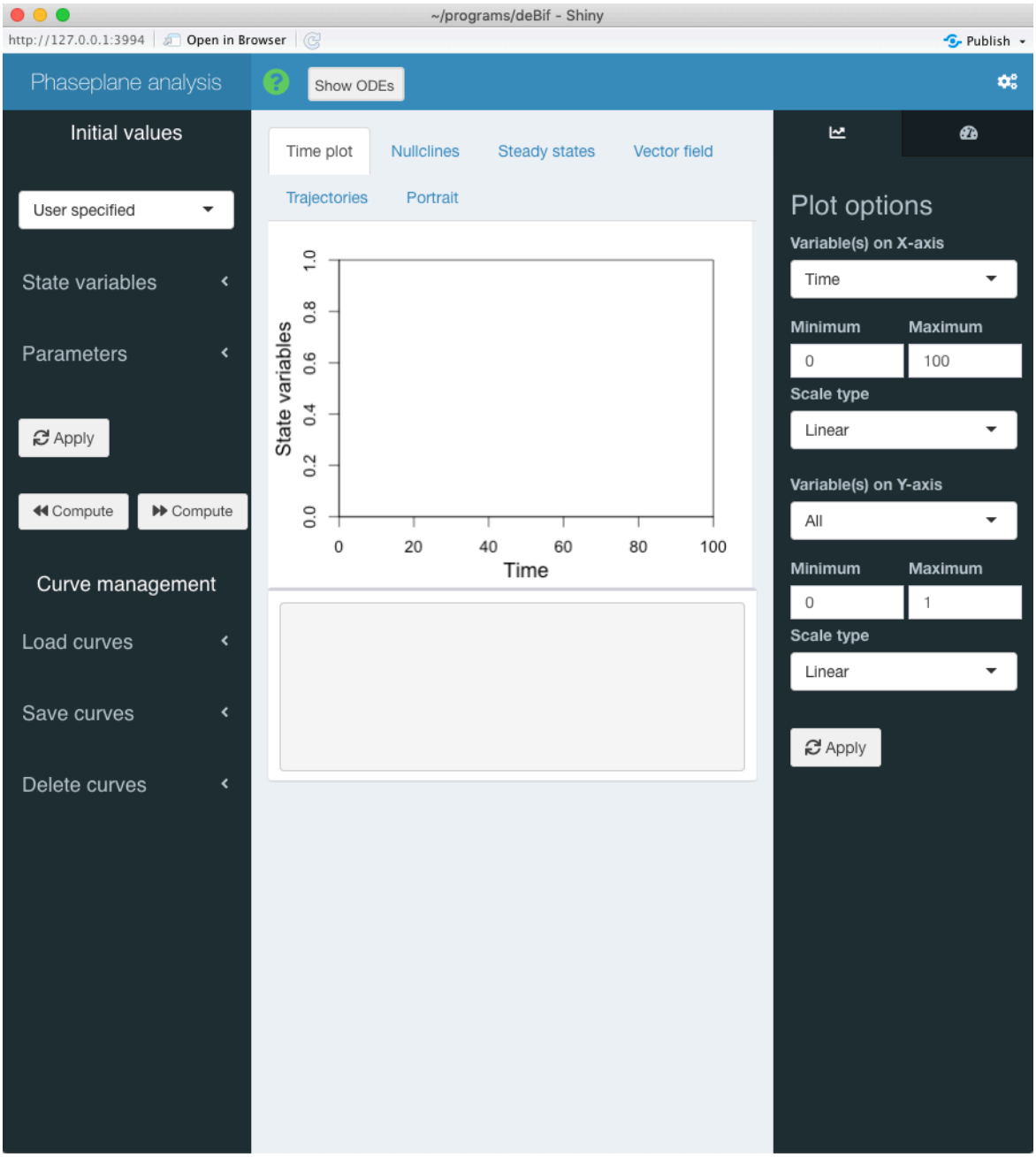

Figure 2.3: The 'phaseplane()' application window with right sidebar expanded allowing for customizing the time series plot.

right sidebar also provides the option to plot the time series curve in a 3-dimensional perspective plot. If a 3-dimensional perspective plot is selected instead of a 2-dimensional plot, the right sidebar extends even further with a slider to adjust the viewing angle of the 3-dimensional graph.

Figure 2.4 shows the 4 different configurations of the right sidebar discussed above. Which one is shown depends on whether a single or all state variables are plotted on the primary, left y-axis, whether or not a second state variable is selected to be shown on the secondary, right y-axis and whether or not a 2-dimensional or 3-dimensional plot is selected.

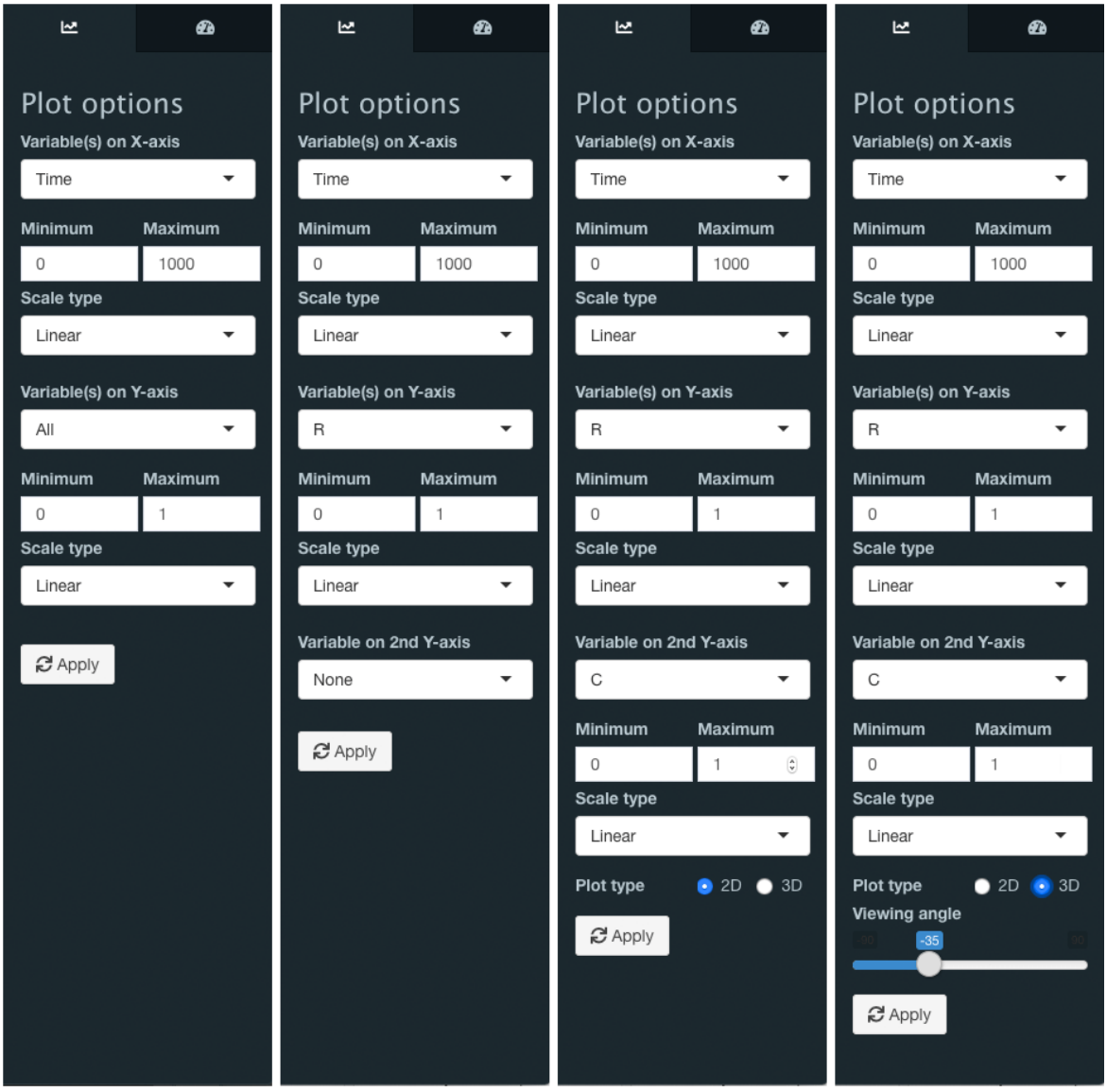

Figure 2.4: The 4 different configurations of the right sidebar of the 'phaseplane()' application window depending on which state variables are selected to be shown on the primary and secondary y-axis.

After adjusting the plot parameters in the right sidebar the `Apply` button has to be pressed to finalize the changes made. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

2.1.3 Numerical settings

In addition to the tab sheet for customizing the plot settings, the right sidebar contains a second tab sheet to customize numerical settings. This numerical options tab can be shown by pressing on the dial icon in the top-right corner of the right sidebar, whenever the latter is not hidden. Figure 2.5 shows the `phaseplane()` application window with the right sidebar showing the numerical options tab.

For the calculation of time series of the model dynamics the maximum integration time, the time step of the integration and the integration method can be adjusted. The numerical methods available for integration of the dynamics are `lsoda`, `ode23`, `ode45` and `rk4`, as provided by the `deSolve` package (Soetaert et al., 2010).

If one of the tab sheets `Steady states`, `Vector field`, `Trajectores` or `Portrait` is selected in `phaseplane()` the numerical settings in the right sidebar show an additional slider labeled `Steady state search grid`. With this slider the number of starting points can be varied, from which searches are started for a steady state in the neighborhood of the starting point. Adjusting the slider to higher values therefore refines the search and hence will reduce the chance that a particular steady state remains undetected, at the expense of longer computation times.

If the tab sheet `Portrait` is selected in `phaseplane()` the numerical settings in the right sidebar show another additional slider labeled `Portrait starting point grid`. With this slider the number of starting points can be varied, from which trajectories or orbits are computed and shown in phase plane. Adjusting the slider to higher values will show more trajectories and hence a phase portrait with higher resolution at the expense of longer computation times.

After adjusting the numerical settings the `Apply` button has to be pressed to let them take effect. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

2.2 Phaseplane analysis

The remaining tab sheets of the `phaseplane()` application all plot a phaseplane of the systems of ODEs. The dimension of the system of ODEs that is being analyzed determines what this phaseplane looks like:

- If only a single ODE is being analysed only 2 additional tab sheets are shown by the `phaseplane()` application, the `Nullclines` tab sheet and the `Steady States` tab sheet. Both tab sheet show a plot where the value of the dependent variable of the ODE is plotted on the x-axis, while the value of the time derivative of this dependent variable

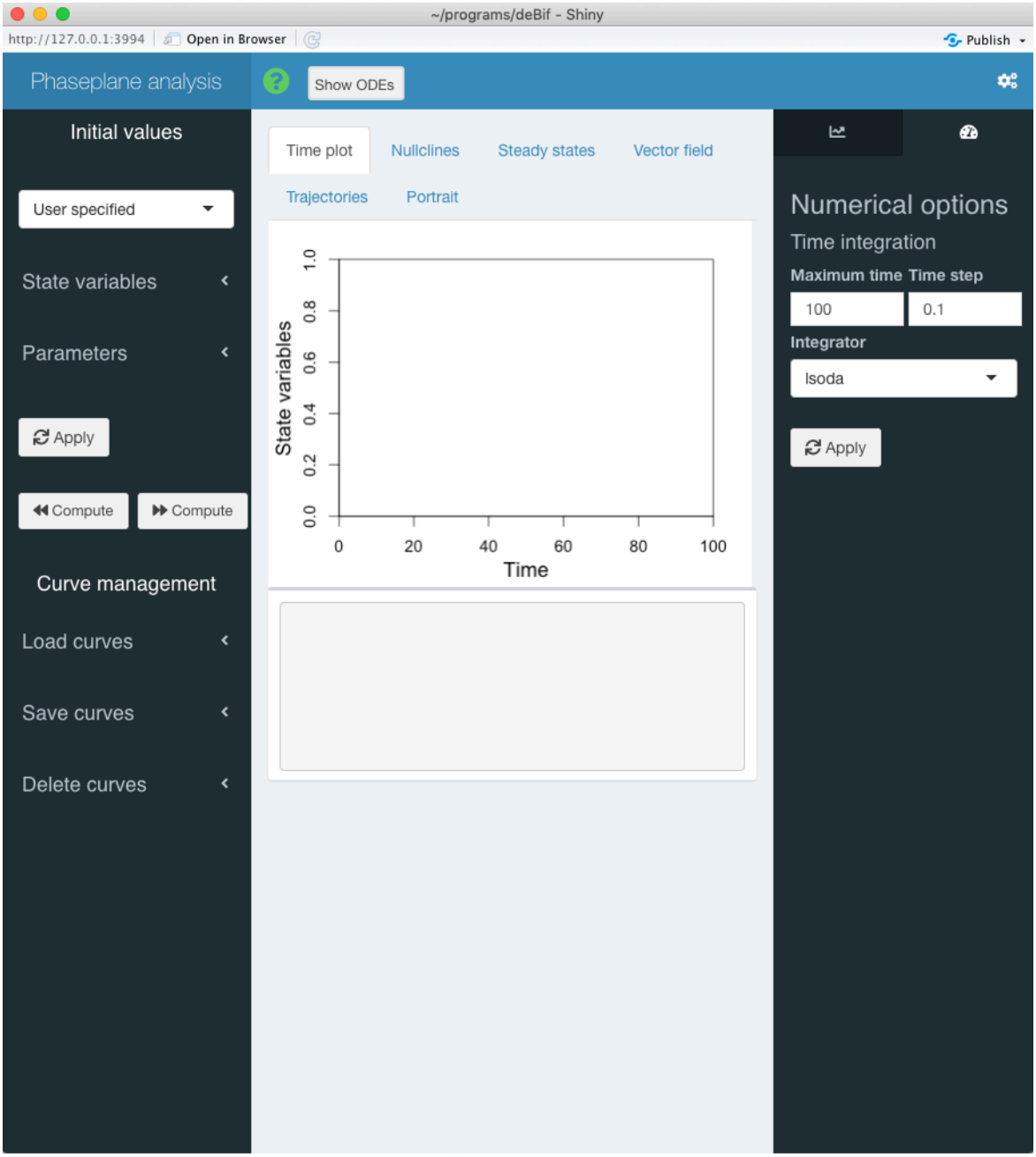

Figure 2.5: The 'phaseplane()' application window with extended right sidebar showing the numerical options tab sheet. Here numerical settings for the time series calculations can be customized.

is plotted on the *y*-axis. Taking the symbol N to refer to this dependent variable, the phaseplane plot hence is equivalent to a $(N, dN/dt)$-plot.

- If the system of ODEs includes exactly 2 ODEs, the two dependent variables in the ODEs are plotted on the *x*- and the *y*-axis. Which dependent variable is plotted on which axis can be customized, but by default the first dependent variable is plotted on the x-axis, while the second dependent variable is plotted on the y-axis.
- If the system of ODEs includes more than 2 ODEs, by default the first dependent variable is plotted on the x-axis, while the second dependent variable is plotted on the y-axis. The remaining dependent variables in the systems of ODEs are assigned constant values equal to the values that are currently given in the `State variables` sub-menu. Which dependent variable is plotted on which axis can be customized.

The phaseplane plot can be customized using the right sidebar, which can be shown by clicking the gears icon in the right-top corner of the application window (see Figure 2.1). Figure 2.6 shows the `phaseplane()` application window with the `Nullclines` tab sheet selected and the right sidebar visible. The drop-down menus allow for selecting which state variables are used as *x*- and *y*-coordinates in the phaseplane plot. The boxes and drop-down menus in the right sidebar also allow for customization of the minimum and maximum values on the *x*- and *y*-axis as well as whether a linear of logarithmic axis scale should be used.

The different phaseplane tab sheets provide different types of information about the dynamics of the system of ODEs:

- The `Nullclines` tab sheet plots the nullclines (also called isoclines) of the ODEs, which represent those combinations of state variable values for which one of the right-hand sides of the ODEs equals 0. If only a single ODE is analyzed this tab sheet shows the value of the dependent variable in the ODE on the *x*-axis, while the value of the time derivative of this dependent variable is plotted on the *y*-axis. Taking the symbol N to refer to this dependent variable, the phaseplane plot hence is equivalent to a $(N, dN/dt)$-plot.
- The `Steady states` tab sheet shows in addition to the nullclines (or isoclines) the location of all equilibrium points of the system of ODEs that have been detected in the range of the state variables that is being shown in the plot. If only a single ODE is analyzed these equilibrium points correspond to the points where the curve of dN/dt as a function of N crosses the *x*-axis, while for systems of ODEs the equilibrium points are the intersections of the nullclines. Stable and unstable equilibrium points are labeled differently and the eigenvalues characterizing the dynamics in the neighborhood of the equilibrium point are reported in the console window below the plot.
- The `Vector field` tab sheet, which is only shown for systems of ODEs, plots in addition to the nullclines and the steady states the vector or flow field, which indicates the direction of the dynamics from the various positions in the phaseplane.
- The `Trajectores` tab sheet, which is only shown for systems of ODEs, plots in addition to the nullclines and the steady states the time series of the system dynamics that are also shown in the `Time plot` tab sheet. In this tab sheet these time dynamics are plotted in the phaseplane. Only those time series will be shown that have the identical parameter values as is currently used for the phaseplane plot in the `Trajectores` tab sheet. From

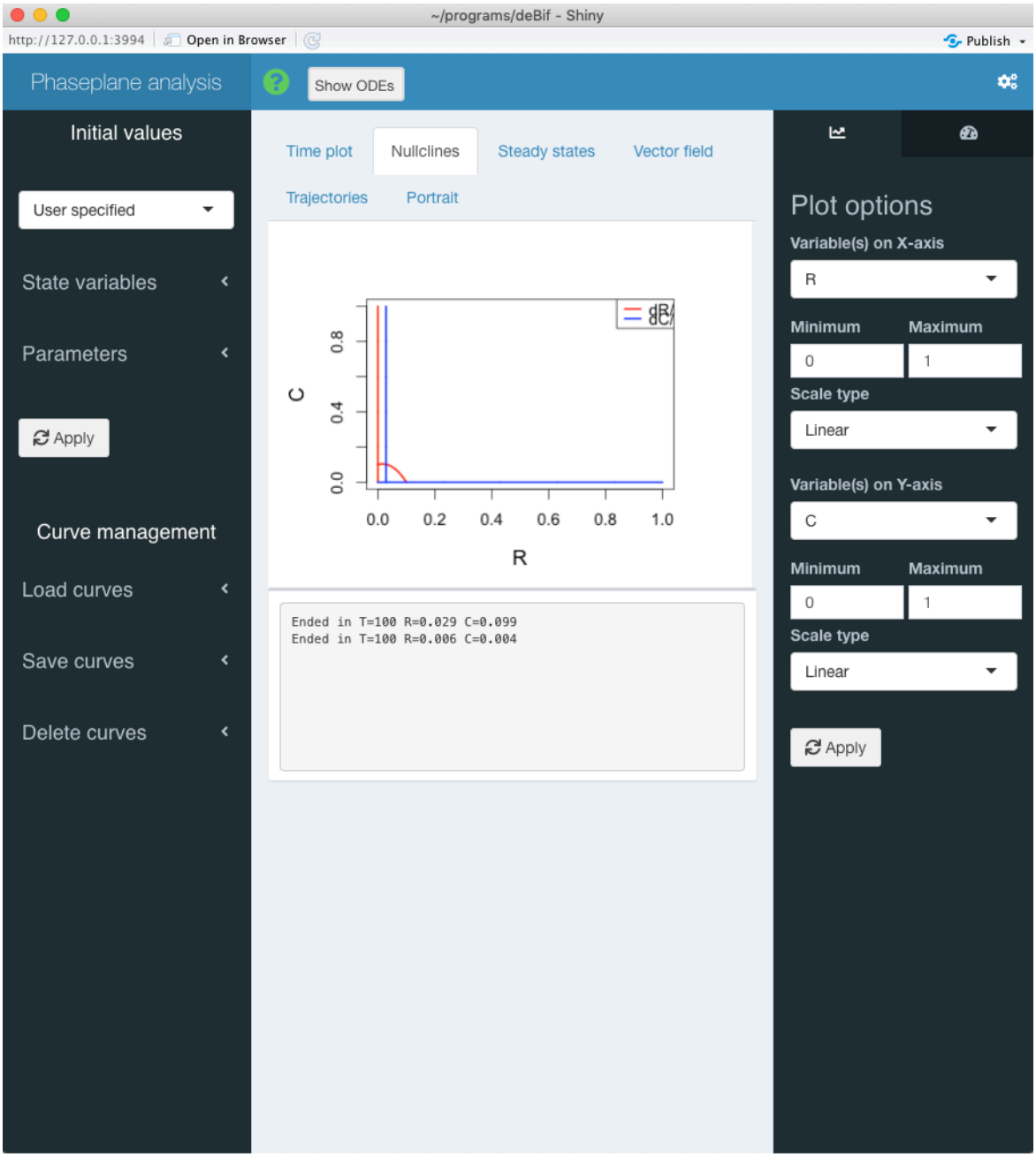

Figure 2.6: The 'phaseplane()' application window showing a phaseplane diagram with the right sidebar expanded allowing for customizing the phaseplane plot.

this tab sheet it is also possible to initiate the computation of a time series of the systems dynamics by pressing the `<<Compute` or `>>Compute` button for backward and forward integration in time, respectively.

- The `Portrait` tab sheet, which is only shown for systems of ODEs, plots in addition to the nullclines and the steady states a phase portrait characterizing the model dynamics, which is visualized by plotting a number of trajectories or orbits from a regularly spaced set of initial points in the phaseplane.

Clicking on the `State variables` menu title in the left sidebar (see Figure 2.1) expands a sub-menu with a list of names of all state variables in the system of ODEs and input boxes that contain current values of these state variables. New values for the various state variables can be specified by editing the values in the input boxes. Similarly, clicking on the `Parameters` menu title expands a sub-menu showing a list of names of all parameters and input boxes that contain the current values of the system parameters. New values of the state variables and the parameters can be specified by editing the values in the input boxes. Also these new values only take effect once the the `Apply` button is pressed. Once, the `Apply` button is pressed the phaseplane plot is redrawn with the new values of the state variables and parameters.

Notice that the success of finding all equilibrium points of the model is dependent on the range of the state variables shown on the *x*- and *y*-axis. If this range is taken large, the application might not find all equilibrium points.

2.3 Saving graphs

All tab sheets of the `phaseplane()` program show at the top-right corner of the plot frame a button that allows you to download an image of the current graph. Clicking this download icon will open up a file dialog box in which you can specify the name of the PNG or PDF file, in which to save the image of the plot. Whether the plot is saved to a PNG or a PDF file can be determined with the command-line argument `saveplotas` (see section 2.6). More elaborate graphs can of course be constructed by exporting the computed curves as explained below and using other packages in R to visualize the results.

2.4 Curve management

The bottom part of the left sidebar of the `phaseplane()` application shows 3 sub-menus that allow for manipulation of the computed curves. Curves can be read into the `phaseplane()` application from variables that are present in the global R environment from which the application was started, computed curves can be saved to variables in this global R environment and curves can be deleted. Figure 2.7 shows 4 snapshots of the left sidebar with the different sub-menus expanded and in different states.

2.4.1 Load curves

By expanding the `Load curves` sub-menu an input box shows up in which you can enter the name of a variable (*do not use any quotes!*) that is present in your global R environment (see Figure 2.7). This variable should contain time series curves. As long as the global variable you

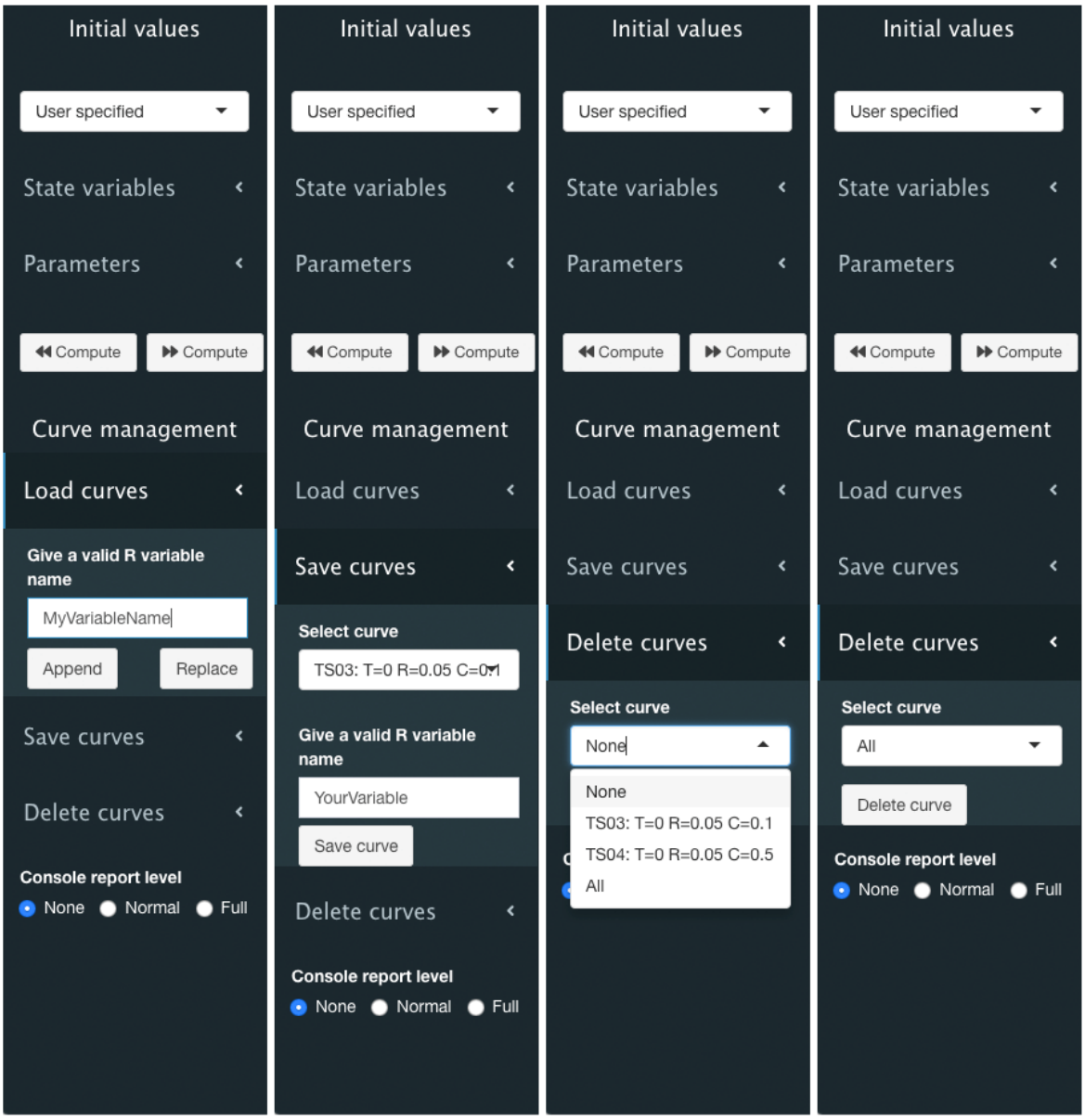

Figure 2.7: The ‘phaseplane()‘ application window with the left sidebar showing the ‘Load curves‘ submenu, the ‘Save curves‘ submenu, the ‘Delete curves‘ submenu with the curve choice drop-down menu expanded and the ‘Delete curves‘ submenu with the ‘All‘ option selected. The action buttons in each submenu allow to load, save or delete the selected curves.

want to load has been saved by the phaseplane() application, it should be possible to be read in. But this is of course only taking effect if the curves stored in the global variable pass some tests. For example, the curves in the global variable should contain all the same state variables and parameters as the state variables and parameters in the system of ODEs that is currently being analysed with the phaseplane() application.

Independent of which of the application tabs is active when you load curves from a global variable, all the valid curves in this global variables will be read in.

The two action buttons in the Load curves sub-menu allow you to either append the curves in the global variable that you are loading to the current set of curves stored by the application or replace the entire set of curves that is currently stored by the phaseplane() application. The console box below the plot frame will report on the success or failure of the load operation.

2.4.2 Save curves

Expanding the Save curves sub-menu reveals a curve selection drop-down menu, an input box in which you can enter your name of choice for the variable in your global R environment, in which to store curves, and an action button to execute the save action (see Figure 2.7). The curve to save can be selected via the curve selection drop-down menu. This drop-down menu lists in addition to the options None (the default choice) and All (the last choice) the labels of all the time series that have been computed. Either one specific curve can be selected for saving or all curves. It is not possible to save 2 out of 3 curves at the same time. In the input box you have to enter a valid name (*do not use any quotes!*) for a variable in the global R environment from which the phaseplane() application was started. Take care with specifying this variable name, the application will overwrite an already existing variable.

Pressing the Save button will save the selected curve or all curves to the variable with the specified name.The console box below the plot frame will report on the success or failure of the save operation. The variable will, however, only show up in your Rstudio Environment panel that shows the list of variables in your global R environment after the phaseplane() application has exited. I have not found a way to update the contents of this Rstudio Environment panel while the phaseplane() application is still active.

2.4.2.1 Delete curves

Expanding the Delete curves sub-menu reveals a curve selection drop-down menu and an action button to execute the delete action (see Figure 2.7). The curve to delete can be selected via the curve selection drop-down menu. This drop-down menu lists in addition to the options None (the default choice) and All (the last choice) the labels of all the computed time series. Either one specific curve can be selected for deleting or all curves. It is not possible to delete 2 out of 3 curves at the same time.

Pressing the Delete button will delete the selected curve or all curves currently stored by the program.The console box below the plot frame will report on the success or failure of the operation.

2.5 Saving program settings on exit

Whenever the program exits all the curves that the `phaseplane()` application has computed and has currently stored in memory are saved in a global variable `<model>PhaseCurves`, where the sub-string `<model>` is the name of the function describing the dynamics, which is passed as first argument to `phaseplane()`. Similarly, all programs settings, such as the plot settings on the different tabs and the numerical settings for the time integration are saved in the global variable `<model>PhaseSettings`. These two global variables will show up on return to your R environment after the programs has ended. When present, the global variables `<model>PhaseCurves` and `<model>PhaseSettings` will be read by the `phaseplane()` application the next time you start it up. They will only take effect, however, if the variables pass some tests successfully. For example, if the variable `<model>PhaseCurves` does not contain the correct set of state variables and parameters it will be ignored when you start up the `phaseplane()` application.

The variables `<model>PhaseCurves` and `<model>PhaseSettings` are overwritten every time the `phaseplane()` application ends. Therefore, if you want to save a particular set of curves and the application settings, you have to copy these variables to new variables with your own unique names. At a later time these saved variables can be assigned to the variables `<model>PhaseCurves` and `<model>PhaseSettings` again to start the `phaseplane()` application at the point where you left off.

The variables `<model>PhaseCurves` and `<model>PhaseSettings` will be ignored completely if the `phaseplane()` application is started with the command-line argument `resume = FALSE`. The default of this command-line argument is `resume = TRUE`.

2.6 Optional command-line arguments

The `phaseplane()` application allows for a number of additional arguments that can be included at the command line to tweak graphical default values used by the application. These arguments are all optional and adopt default values when not specified on the command line.

The arguments that can be included on the command line are:

- `lwd`: The line width used for plotting computed curves. The default value is 3.
- `cex`: The base font size for labeling axes and legends in the plots. The default value is 1.2.
- `tcl.len`: The length of ticks on the axes. The default value is 0.03.
- `saveplotas`: Either “pdf” or “png” to save the plot to a PDF or PNG file. The default value is “png”.

3 The `bifurcation()` application

After starting the `bifurcation()` application a new window opens up, which is shown in Figure 3.1. The application contains a sidebar at the left, which is used for:

- specifying initial values for the state variables
- specifying initial values for the parameters
- Starting computations from the chosen initial state with the selected parameters
- loading, saving and deleting curves that are computed with the `bifurcation()` application.

The remainder of the application window consists of an area with 3 tab sheets, called `Time series`, `1 parameter bifurcation` and `2 parameter bifurcation`. Switching between these tabs is achieved by clicking on the appropriate tab name. Each tab sheet includes a plot frame and 2 frames in which progress messages as well as errors will be reported. Notice that the plot frame differs between the 3 tab sheets.

In the following the Rosenzweig-MacArthur model presented in section 1.1 will be used to illustrate the different types of curves that can be computed and the different menus to adjust the computations and the graphical presentation of their results.

3.1 Time series

The first tab only shows curves of the time dynamics as predicted by the system of ODEs that is being investigated. The dynamics are computed numerically through integration after pressing either the `<<Compute` button or the `>>Compute` in the left sidebar (see Figure 3.1). The program will integrate the dynamics from $t = 0$ till the maximum integration time when the `>>Compute` button is pressed. See section 3.1.3 for how to change the value of the maximum integration time. When the `<<Compute` button is pressed the integration will start at the maximum integration time and attempt to integrate backward in time till $t = 0$.

Numerical integration of the system of ODEs forward in time should always proceed without problems (if at least you specified the system of ODEs correctly), but numerical integration of the dynamics backward in time may cause problems as the values of the state variables in the ODEs may approach infinite values. Therefore, be careful with numerical integration backward in time, it is advisable to compute over a small time interval only. Hence, choose the value of the maximum integration time (see section 3.1.3) not too large.

Every time the `<<Compute` or the `>>Compute` button is pressed a new time series curve is computed from the current initial state with the current parameters. All computed curves are shown in the graph. Each curve furthermore gets a label and the first and last point of the curve are labeled as special points. These special points are added to the drop-down menu at the top of the left sidebar. Figure 3.2 shows the application window with two time series that have been computed from different initial states, while the special points drop-down menu is expanded. The drop-down menu shows the first and last point of the two computed time series (started from $R = 0.05, C = 0.1$ and $R = 0.05, C = 0.5$, respectively). These special points can be selected using the special points drop-down menu, in which case the values of the state vari-

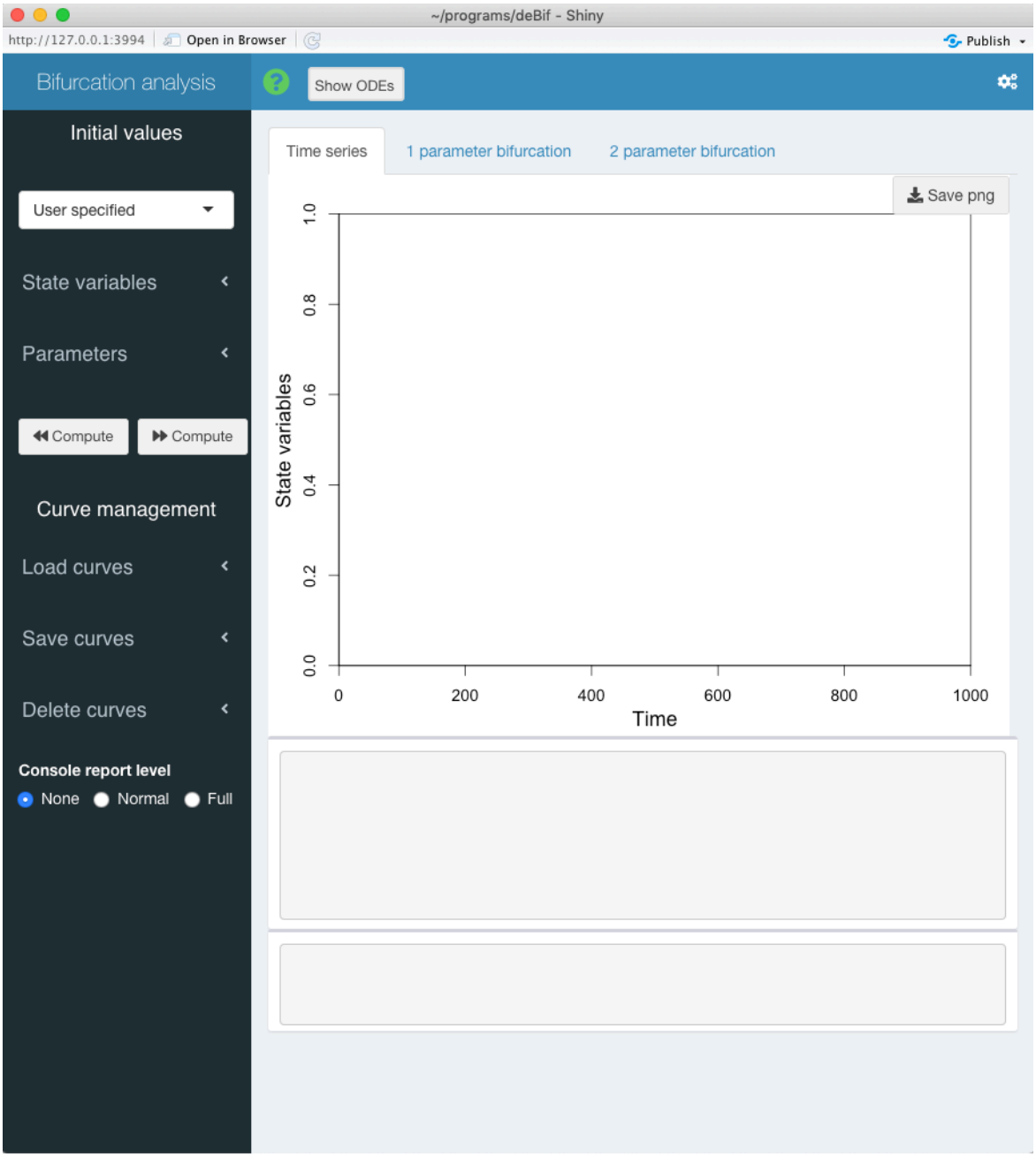

Figure 3.1: Starting view of the 'bifurcation()' application. The red arrow indicates the gears icon that opens up the right sidebar.

ables and parameters of the special point are used as starting point for the next computation (either of a time series or of an equilibrium bifurcation calculation).

By default the plot in this tab shows the time variable on the x-axis and the values of all state variables defined in the model on the y-axis. These settings can however be customized as discussed in section 3.1.2.

3.1.1 Changing state variables and parameters

Clicking on the `State variables` menu title in the left sidebar (see Figure 3.1) expands a sub-menu with a list of names of all state variables in the system of ODEs and input boxes that contain current values of these state variables. New values for the various state variables can be specified by editing the values in the input boxes. These new values take effect as soon as the `<<Compute` or the `>>Compute` button is pressed.

Similarly, clicking on the `Parameters` menu title expands a sub-menu showing a list of names of all parameters and input boxes that contain the current values of the system parameters. New values of the parameters can be specified by editing the values in the input boxes. Also these new values only take effect once the the `<<Compute` or the `>>Compute` button is pressed.

3.1.2 Customizing the plot

Curves of the model dynamics that will be shown in the time series plot include as variables the time and the values of all state variables at each time. By default the plot frame on the `Time series` tab sheet shows the independent time variable on the x-axis and the values of all state variables on the y-axis with the x-axis ranging from 0 to 1000 and the y-axis from 0 to 1. These default settings can be customized using the right sidebar, which can be shown by pressing the gears icon in the right-top corner of the application window (see Figure 3.1).

Figure 3.3 shows the `bifurcation()` application window with the right sidebar visible. The drop-down menus allow for selecting which variables are used as x- and y-coordinates in the time series plots. Only a single variable can be selected as x-coordinate, either the time variable or one of the state variables of the model. As y-coordinate it is possible to select all state variables, in which case as many curves will be plotted as there are state variables defined in the model. All state variables will then be plotted using the same y-axis at the left of the plot. The boxes and drop-down menus in the right sidebar allow for customization of the minimum and maximum values on the x- and y-axis as well as whether a linear of logarithmic axis scale should be used.

Alternatively, when a single state variable is selected as y-coordinate the right sidebar extends and offers the possibility to add a second y-axis on the right-hand side of the plot, on which the value of another state variable can be plotted. If a second state variable is selected to be plotted on the second y-axis at the right-hand side of the plot the right sidebar panel extends even further to allow for specification of the minimum and maximum value and the scale type to use on this second y-axis.

If state variables are selected to be plotted on both the primary and the secondary y-axis, the right sidebar also provides the option to plot the time series curve in a 3-dimensional perspective plot. If a 3-dimensional perspective plot is selected instead of a 2-dimensional plot, the

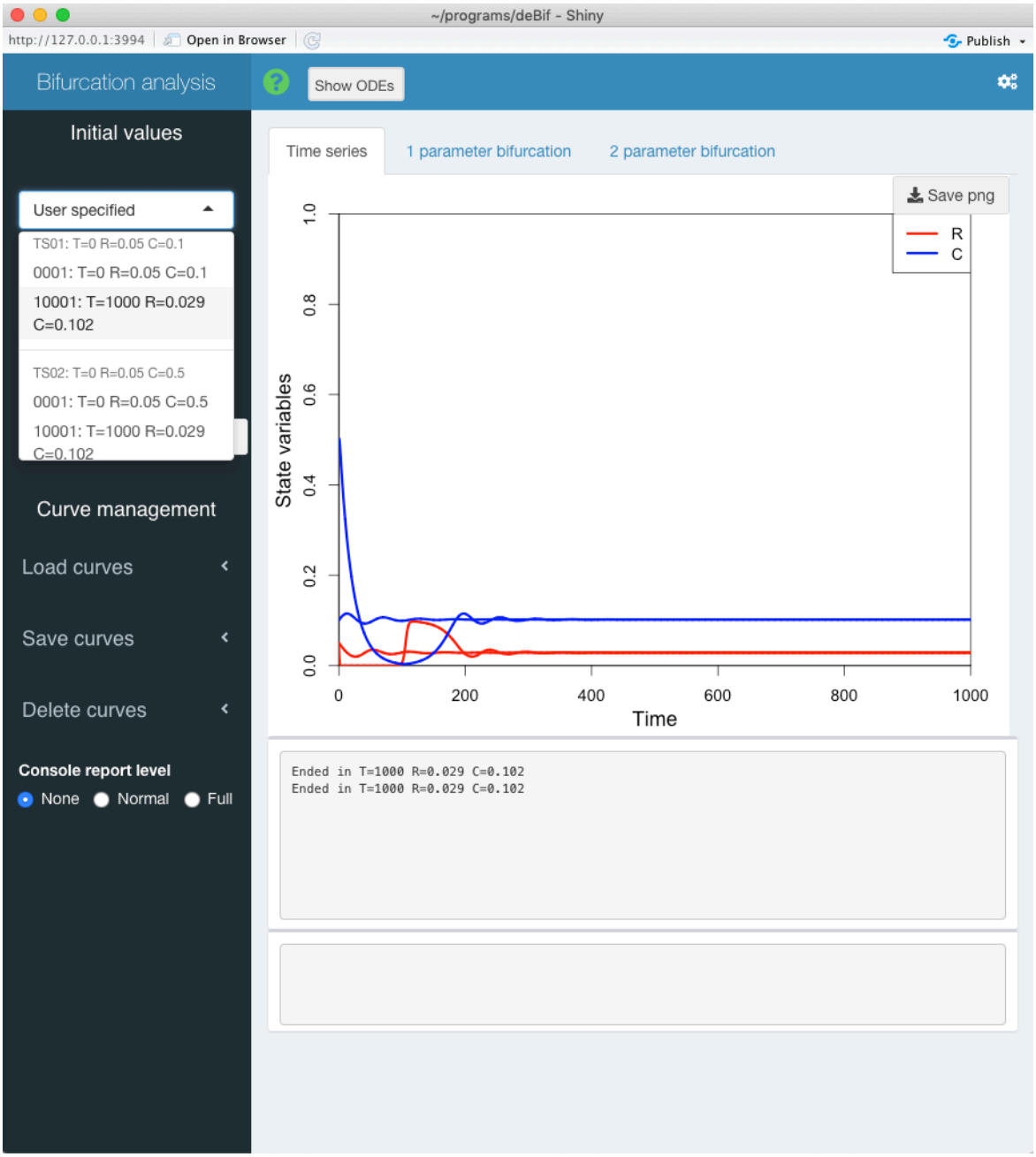

Figure 3.2: Two computed time series are shown in the 'bifurcation()' application and the special points menu is expanded, listing the first and last points of the two curves.

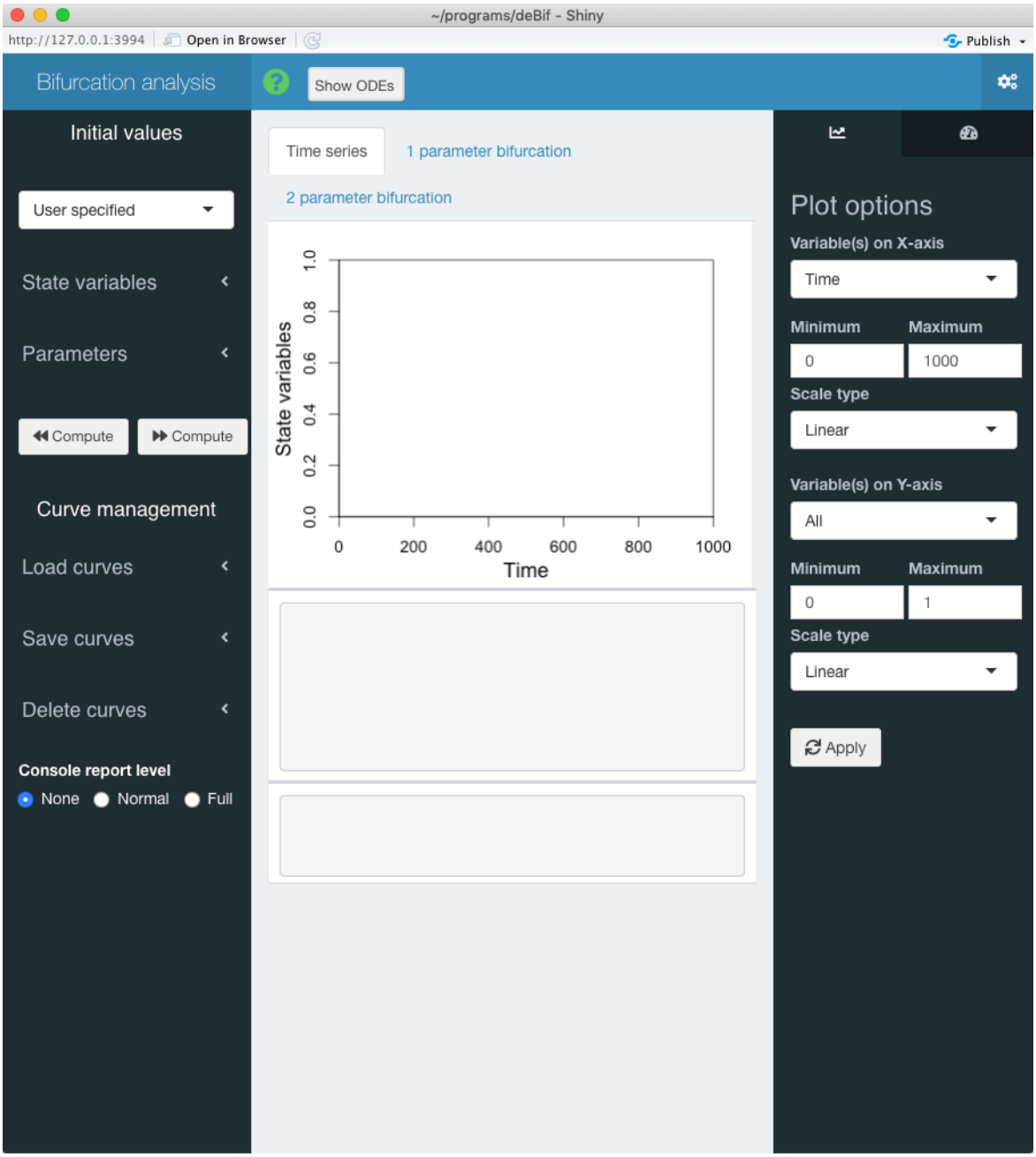

Figure 3.3: The 'bifurcation()' application window with right sidebar expanded allowing for customizing the time series plot.

right sidebar extends even further with a slider to adjust the viewing angle of the 3-dimensional graph.

Figure 3.4 shows the 4 different configurations of the right sidebar discussed above. Which one is shown depends on whether a single or all state variables are plotted on the primary, left y-axis, whether or not a second state variable is selected to be shown on the secondary, right y-axis and whether or not a 2-dimensional or 3-dimensional plot is selected.

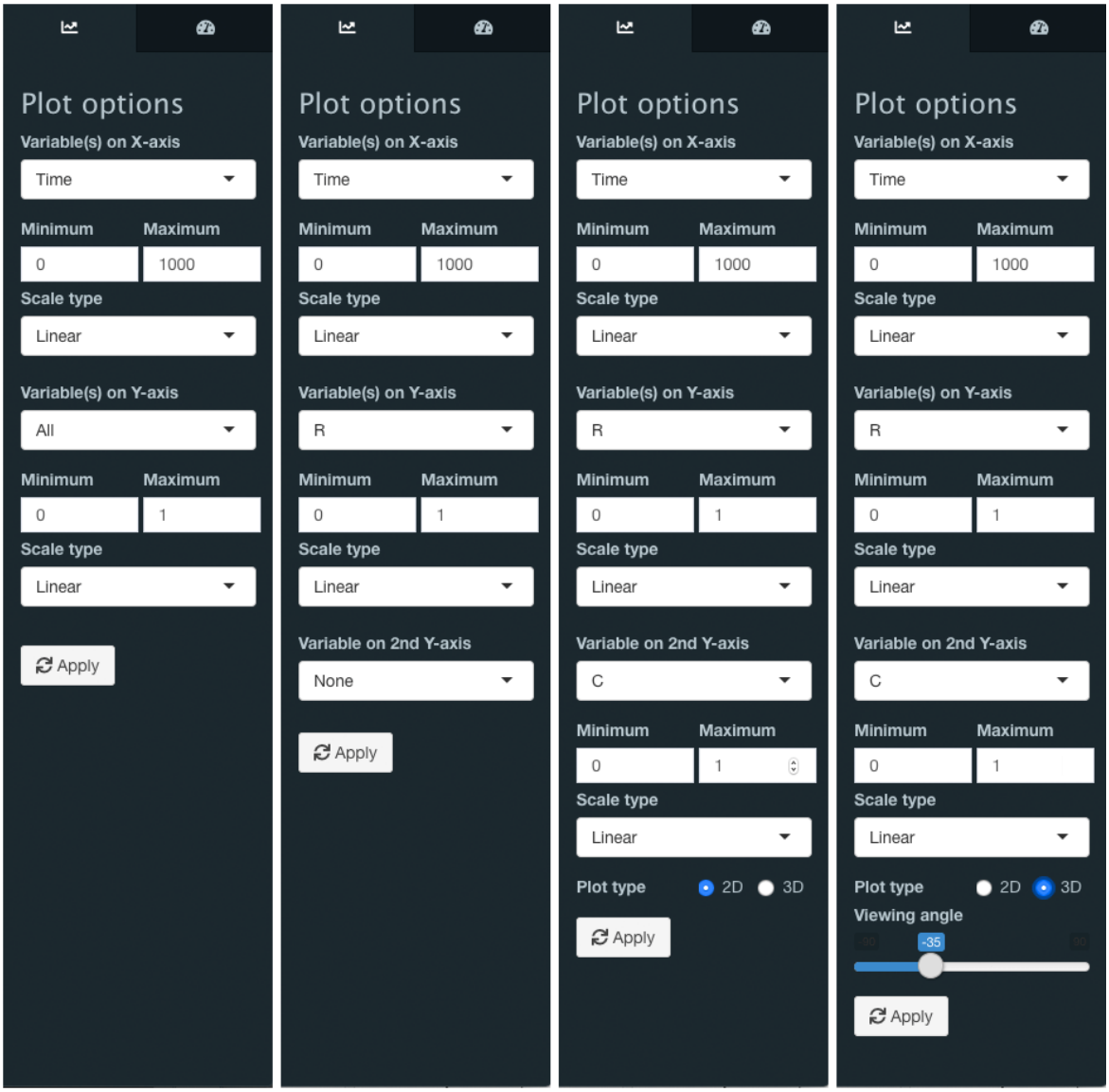

Figure 3.4: The 4 different configurations of the right sidebar of the 'bifurcation()' application window depending on which state variables are selected to be shown on the primary and secondary y-axis.

After adjusting the plot parameters in the right sidebar the `Apply` button has to be pressed to finalize the changes made. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

3.1.3 Numerical settings

In addition to the tab sheet for customizing the plot settings, the right sidebar contains a second tab sheet to customize numerical settings. This numerical options tab can be shown by pressing on the dial icon in the top-right corner of the right sidebar, whenever the latter is not hidden. Figure 3.5 shows the `bifurcation()` application window with the right sidebar showing the numerical options tab.

For the calculation of time series of the model dynamics the maximum integration time, the time step of the integration and the integration method can be adjusted. The numerical methods available for integration of the dynamics are `lsoda`, `ode23`, `ode45` and `rk4`, as provided by the `deSolve` package (Soetaert et al., 2010).

After adjusting the numerical settings the `Apply` button has to be pressed to let them take effect. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

3.2 Equilibrium curves with 1 free parameter

Choosing the bifurcation parameter

The tab sheet called `1 parameter bifurcation` is used for the computation and visualization of equilibrium curves as a function of one free parameter, the bifurcation parameter. For that reason on the x-axis of the plot in this tab sheet is always the value of one of the parameters in the system of ODEs. By default this is the first element in the named parameter vector with which the `bifurcation()` application was started up, but this can be changed through the plot option settings available through the right sidebar (see section 3.2.2). Figure 3.6 shows the default view of the tab sheet `1 parameter bifurcation` with the plot option menu in the right sidebar expanded. The first element at the top of this right sidebar is a drop-down menu, which has been expanded in figure 3.6 to show the list of system parameters to choose from as bifurcation parameter. This parameter is at the same time the parameter that will be plotted on the x-axis.

Starting a computation

The left sidebar shows in addition to all the elements that were also shown on the `Time series` tab sheet an option to select either the computation of an equilibrium curve (`Curve type to compute` equal to `EQ`) or a curve of limit cycles (`Curve type to compute` equal to `LC`), while varying the value of the bifurcation parameter plotted on the x-axis.

Computations of equilibrium curves as a function of a bifurcation parameter can in principle be started from a user-specified point, but in general there is no guarantee that an equilibrium state of the system of ODEs will be found from such a point. Therefore, it is a good idea to only start the computation of an equilibrium curve from a user-defined point if you know that the values that you specify for the variables and the parameters indeed corresponds to a steady state of the system of ODEs. A user-defined starting point can be entered using the left sidebar (see section 3.2.1). The computation of a curve of limit cycles can only be started from a Hopf bifurcation point (HP) that the program has detected while computing an equilibrium curve.

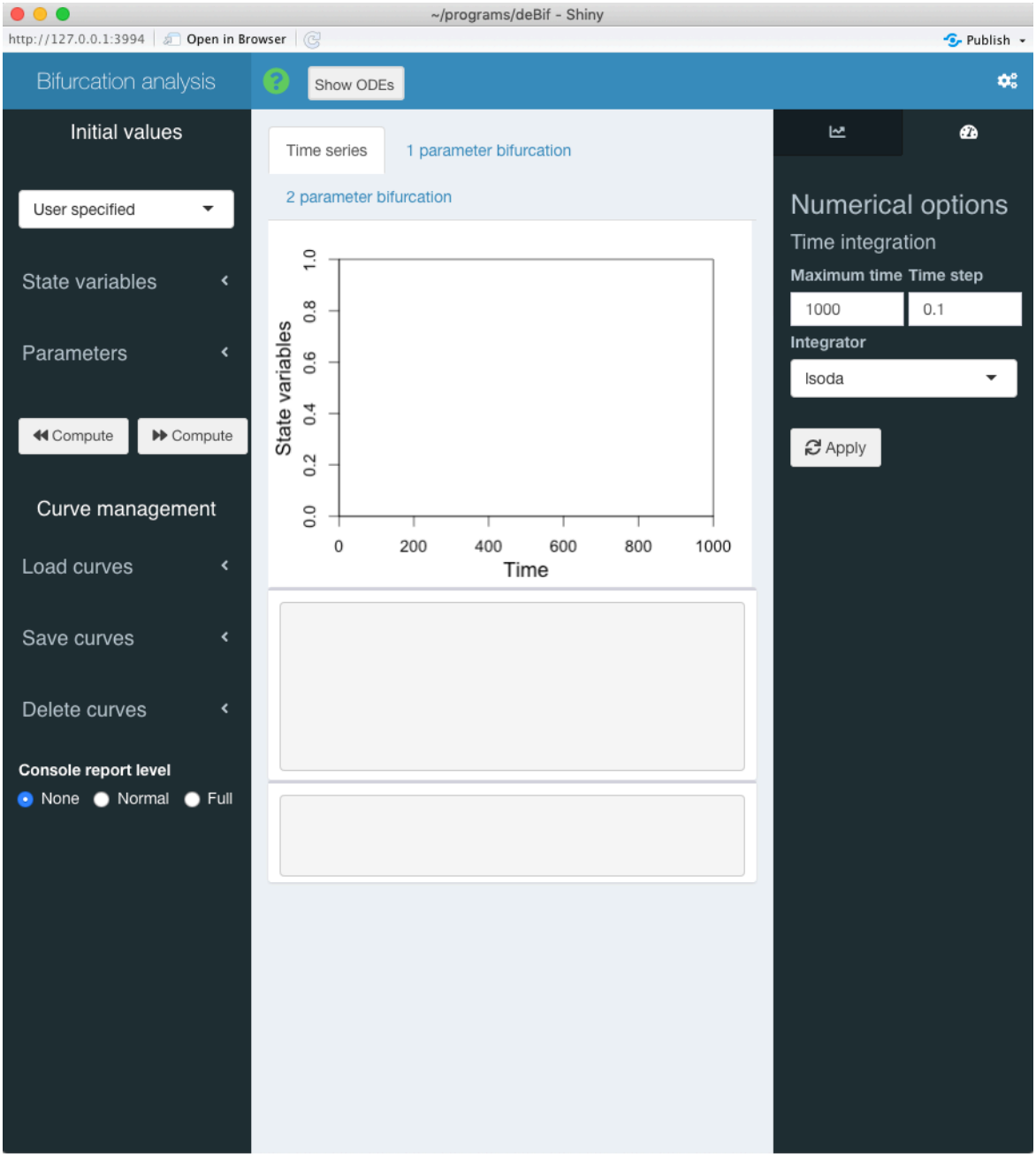

Figure 3.5: The 'bifurcation()' application window with extended right sidebar showing the numerical options tab sheet. Here numerical settings for the time series calculations can be customized.

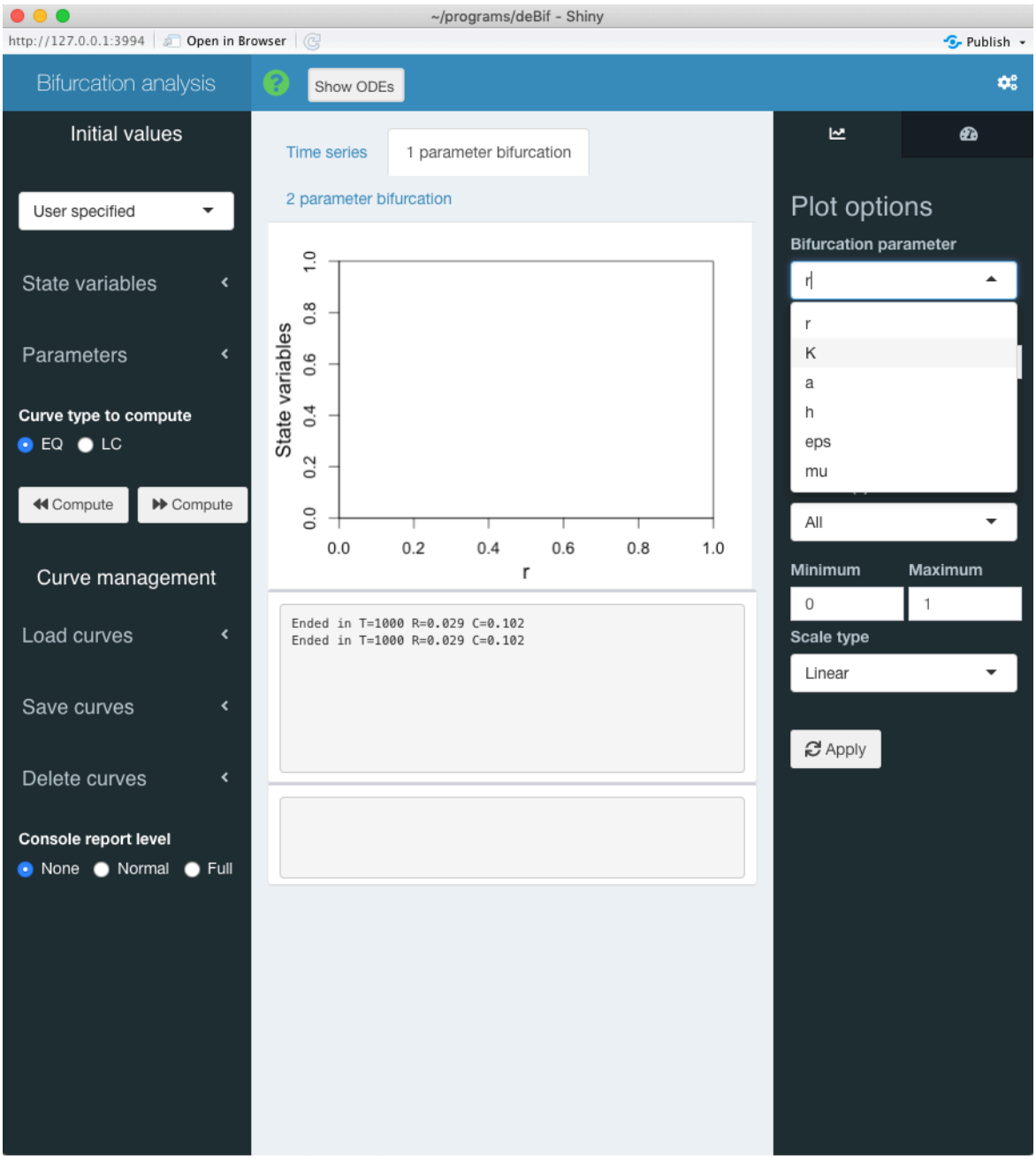

Figure 3.6: The 'bifurcation()' application with the tab sheet '1 parameter bifurcation' selected and the plot option menu in the right sidebar expanded. The first element of the right sidebar is a drop-down menu with which to select the bifurcation parameter. This parameter will be plotted on the x-axis.

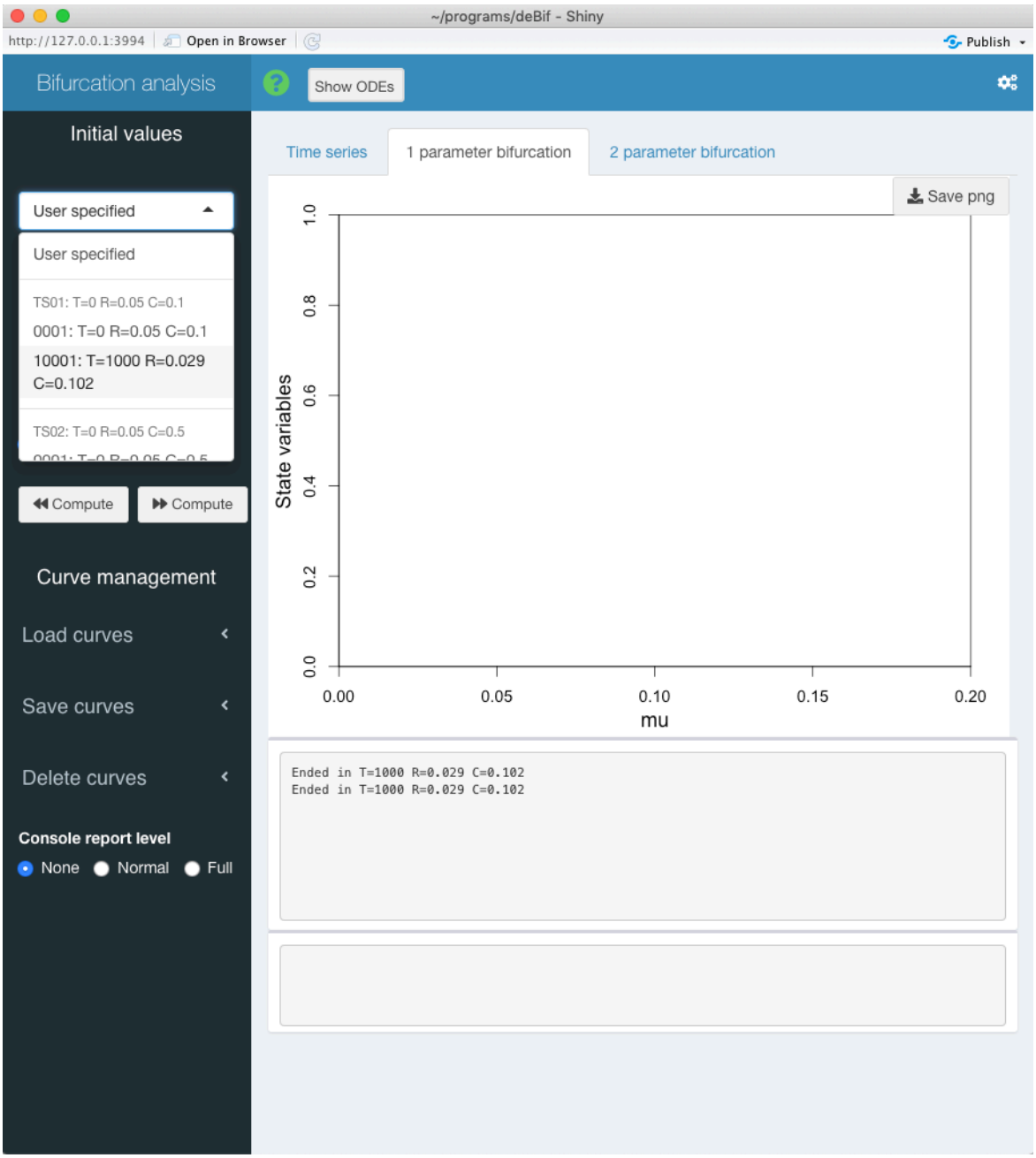

Figure 3.7: The 'bifurcation()' application with the tab sheet '1 parameter bifurcation' selected and the special points drop-down menu in the left sidebar expanded highlighting the final point of a time series curve that is going to be used as starting point for an equilibrium computation.

A good strategy to start the computation of an equilibrium curve as a function of a bifurcation parameter is to use the final point of a time series computation, which has converged to constant values. In section 3.1 two such curves were computed (see Figure 3.2). Figure 3.7 shows the tab sheet `1 parameter bifurcation` with the system parameter μ selected as bifurcation parameter (plotted on the x-axis) and the special points drop-down menu at the top of the left sidebar expanded. The final point of the time series computed is highlighted as it will be selected as starting point for the computation of the equilibrium curve.

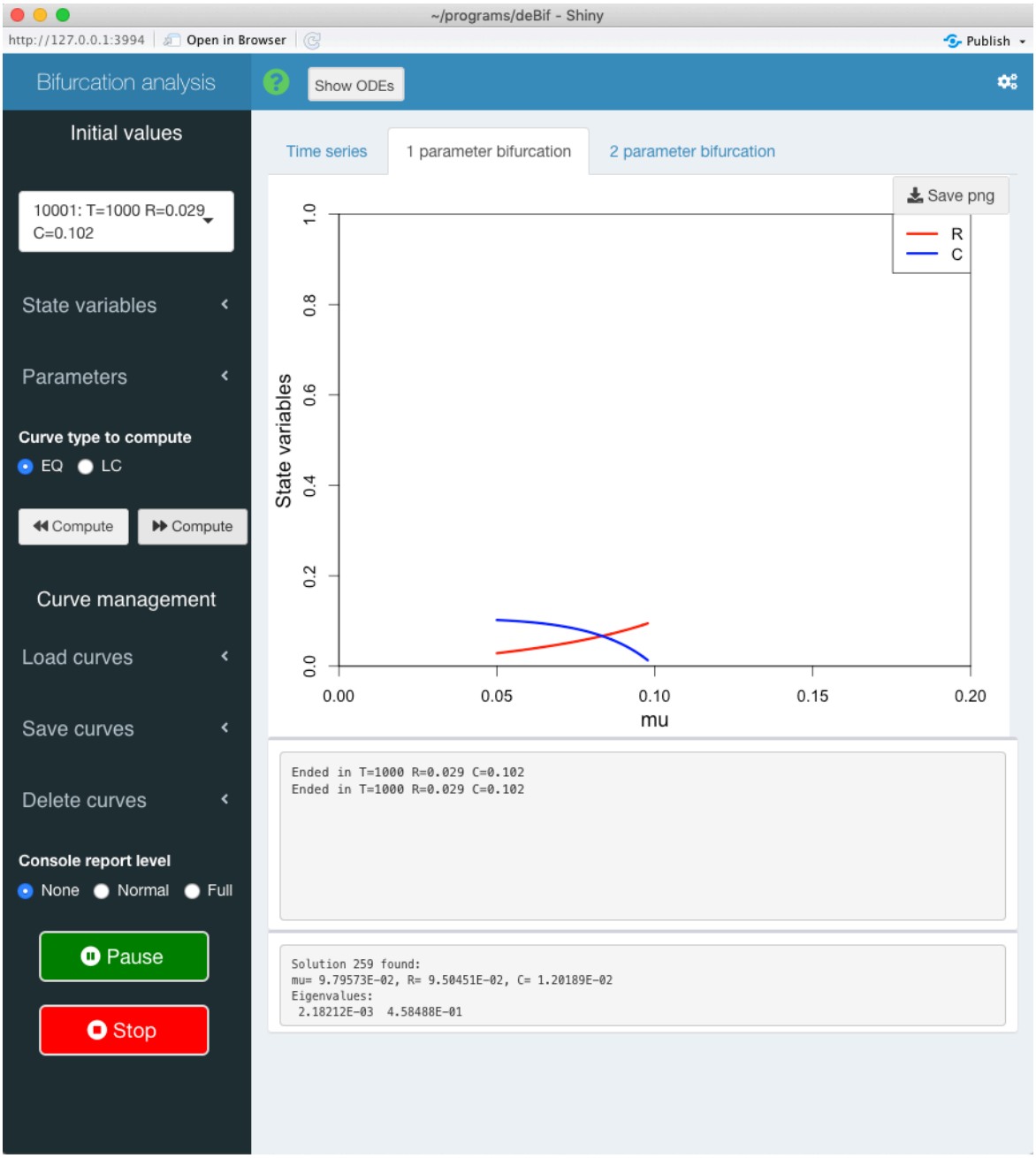

Figure 3.8: The 'bifurcation()' application while it is busy computing an equilibrium curve

Computations are started by clicking either the `<<Compute` or the `>>Compute`, which will start to compute the equilibrium curve originating from the initial point in the direction of lower and

higher values of the bifurcation parameter, respectively. During the computation of a curve a `Pause` and `Stop` button are displayed in the left sidebar (see Figure 3.8) with which the computation can be temporarily halted or stopped altogether. The progress window at the very bottom of the tab sheet `1 parameter bifurcation` reports on the solutions that are found and, importantly, also reports on the eigenvalues that characterize the dynamics in the neighborhood of the equilibrium. Inspection of these eigenvalues allows for checking whether or not the computed steady states are stable or unstable.

Computations of equilibrium curves will halt when the maximum number of solution points along the curve has been reached or when the values of the state variables in the equilibrium point are reaching values outside the range of values currently shown on the x- and y-axis. The default number of solution points along a curve is by default equal to 500, but can be changed (see section 3.2.3). Figure 3.9 shows the completed equilibrium curve, of which the initial part was already shown in Figure 3.8. This complete equilibrium curve has been constructed by 1) first starting the continuation of this curve to lower values of the bifurcation parameter μ; 2) subsequently continuing this curve to higher values of the bifurcation parameter μ; and finally, 3) Selecting the final solution point of the latter curve (point number 501) as initial point for the next calculation and computing once again forward to higher values of the bifurcation parameter. As Figure 3.9 shows two bifurcation points are detected during these computations: a Hopf bifurcation point (HP) at a value of the bifurcation parameter around 0.03 and a transcritical bifurcation or branching point (BP) at a value of the bifurcation parameter around 0.10. Figure 3.9 also shows that the curve section to the left of the HP is plotted with a short-dashed line. This indicates that the equilibrium points on that section of the curve are unstable, that is, the eigenvalues characterizing the dynamics in the neighborhood of the equilibrium have a positive real part.

While computing equilibrium curves the `bifurcation()` application can detect Hopf bifurcation points (HP), branching points (BP) and also saddle-node bifurcation points, also referred to as limit points (LP). By selecting a Hopf bifurcation point (HP) or branching point (BP) a new computation can be started as a function of a single bifurcation parameter. In a branching point (BP) two equilibrium curves intersect each other. When a branching point (BP) is selected as initial point for an equilibrium computation the `bifurcation()` application will attempt to switch to the other curve that intersects the equilibrium curve during the computation of which the BP is detected.

Figure 3.10 shows the two intersecting equilibrium curves, of which the second one has been obtained by selecting the detected BP shown in figure 3.9 as starting point and from that initial point computing the equilibrium curve both forward (to higher parameter values) and backward (to lower parameter values).

When a Hopf bifurcation point (HP) is selected as initial point the application will automatically switch to computing a curve of limit cycles as a function of the bifurcation parameter. The item `Curve type to compute` will automatically be set to `LC`. Computing a curve of limit cycles is computationally intensive and usually takes a long time. Figure 3.11 shows the `bifurcation()` application during the computation of the curve of limit cycles that originates from the Hopf bifurcation point (HP) to lower values of the bifurcation parameter. To represent curves of limit cycles the application will plot the minimum and maximum value of the state variables that are observed during the limit cycle that these variables exhibit. Therefore, limit cycle curves always

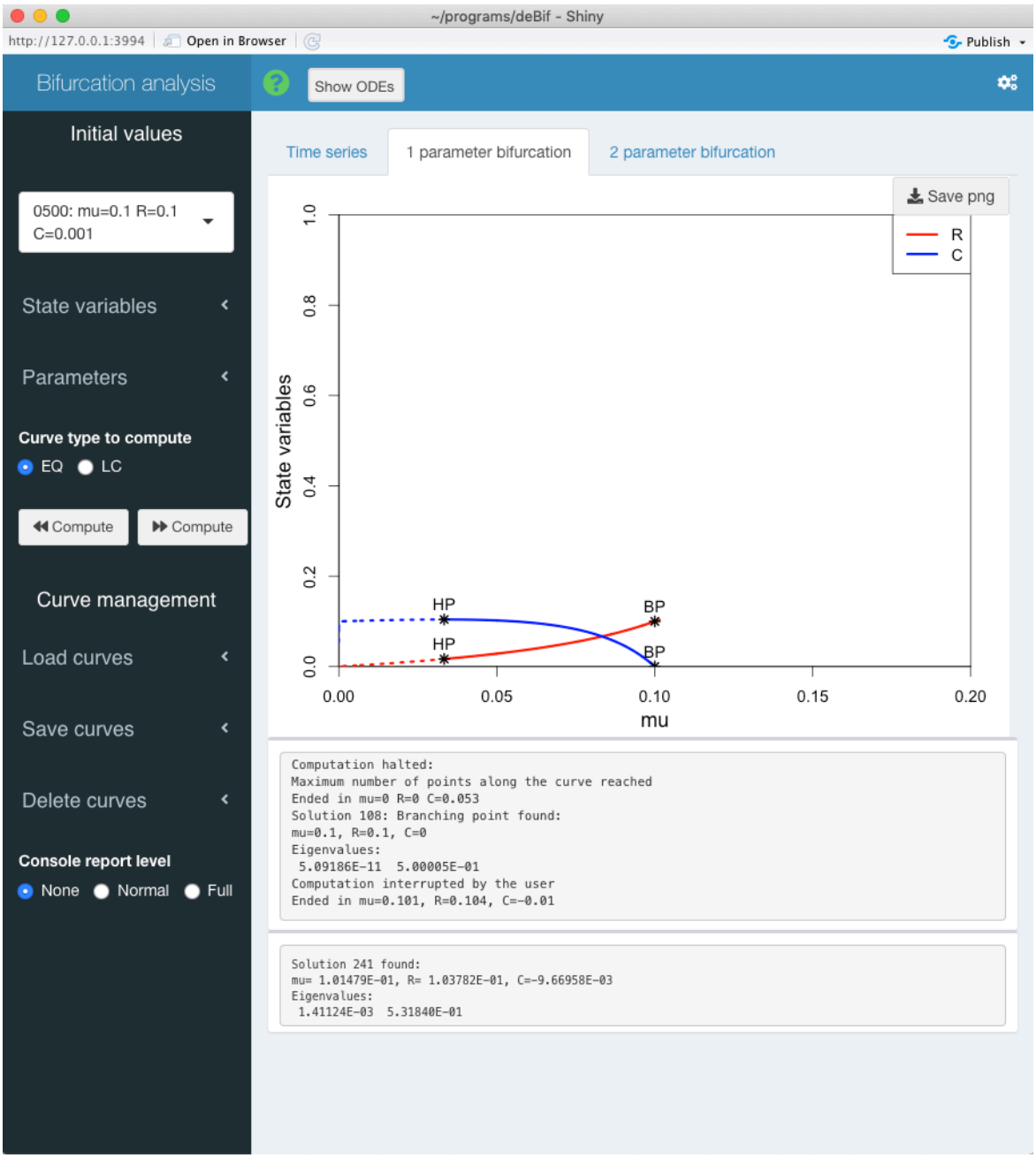

Figure 3.9: The 'bifurcation()' application with the tab sheet '1 parameter bifurcation' showing the complete equilibrium curve as a function of the bifurcation parameter, which has been constructed using both forward (to higher parameter values) and backward (to lower parameter values) continuations.

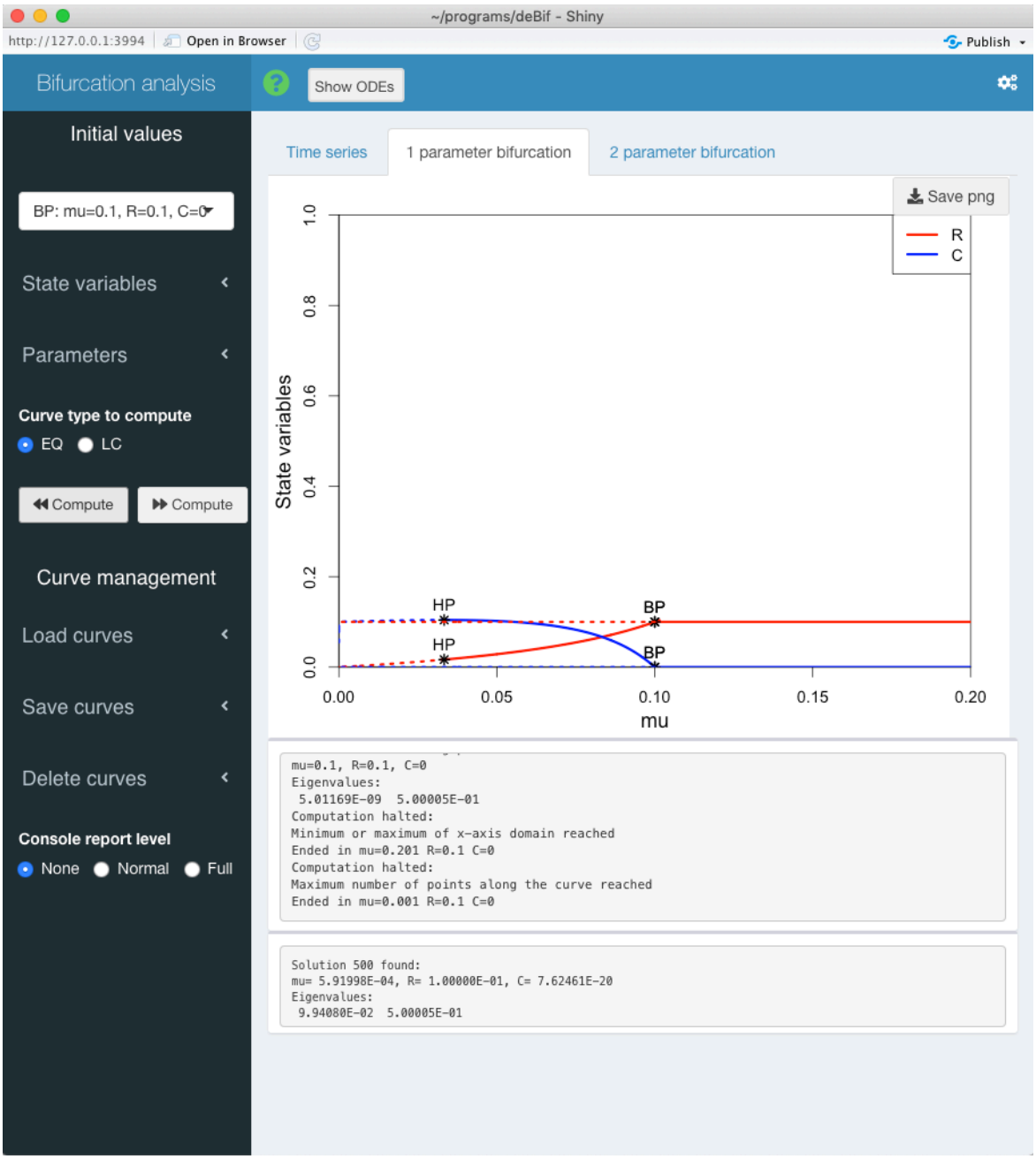

Figure 3.10: The 'bifurcation()' application with the tab sheet '1 parameter bifurcation' showing two equilibrium curve as a function of the bifurcation parameter. The second equilibrium curves has been computed by selecting the branching point (BP) as starting point and continuing the curve forward (to higher parameter values) and backward (to lower parameter values).

consist of an upper and a corresponding lower curve section.

3.2.1 Changing state variables and parameters

Clicking on the `State variables` menu title in the left sidebar (see Figure 3.6) expands a sub-menu with a list of names of all state variables in the system of ODEs and input boxes that contain current values of these state variables. New values for the various state variables can be specified by editing the values in the input boxes. These new values take effect as soon as the `<<Compute` or the `>>Compute` button is pressed.

Similarly, clicking on the `Parameters` menu title expands a sub-menu showing a list of names of all parameters and input boxes that contain the current values of the system parameters. New values of the parameters can be specified by editing the values in the input boxes. Also these new values only take effect once the the `<<Compute` or the `>>Compute` button is pressed.

3.2.2 Customizing the plot

As shown in Figure 3.6 and discussed above, on the tab sheet `1 parameter bifurcation` the variable plotted on the x-axis of the plot is always a parameter of the system of ODEs, which is at the same time the bifurcation parameter. Which parameter is selected as this bifurcation parameter can be selected via the drop-down menu in the right sidebar (see Figure 3.6), which can be shown by pressing the gears icon in the right-top corner of the application window (see Figure 3.1). By default the plot shows the values of all state variables on the y-axis. These default settings can be customized using the right sidebar.

Figure 3.6 already shows the `bifurcation()` application window with the right sidebar visible. The drop-down menus allow for selecting which variables are used as bifurcation parameter and plotted on the x-axis and which variable(s) are used as y-coordinates in the plot of the equilibrium curves. As y-coordinate it is possible to select all state variables, in which case as many curves will be plotted as there are state variables defined in the model. All state variables will then be plotted using the same y-axis at the left of the plot. The boxes and drop-down menus in the right sidebar allow for customization of the minimum and maximum values on the x- and y-axis as well as whether a linear of logarithmic axis scale should be used.

Alternatively, when a single state variable is selected as y-coordinate the right sidebar extends and offers the possibility to add a second y-axis on the right-hand side of the plot, on which the value of another state variable can be plotted. If a second state variable is selected to be plotted on the second y-axis at the right-hand side of the plot the right sidebar panel extends even further to allow for specification of the minimum and maximum value and the scale type to use on this second y-axis.

If state variables are selected to be plotted on both the primary and the secondary y-axis, the right sidebar also provides the option to plot the time series curve in a 3-dimensional perspective plot. If a 3-dimensional perspective plot is selected instead of a 2-dimensional plot, the right sidebar extends even further with a slider to adjust the viewing angle of the 3-dimensional graph.

Figure 3.12 shows the 4 different configurations of the right sidebar discussed above. Which one is shown depends on whether a single or all state variables are plotted on the primary, left

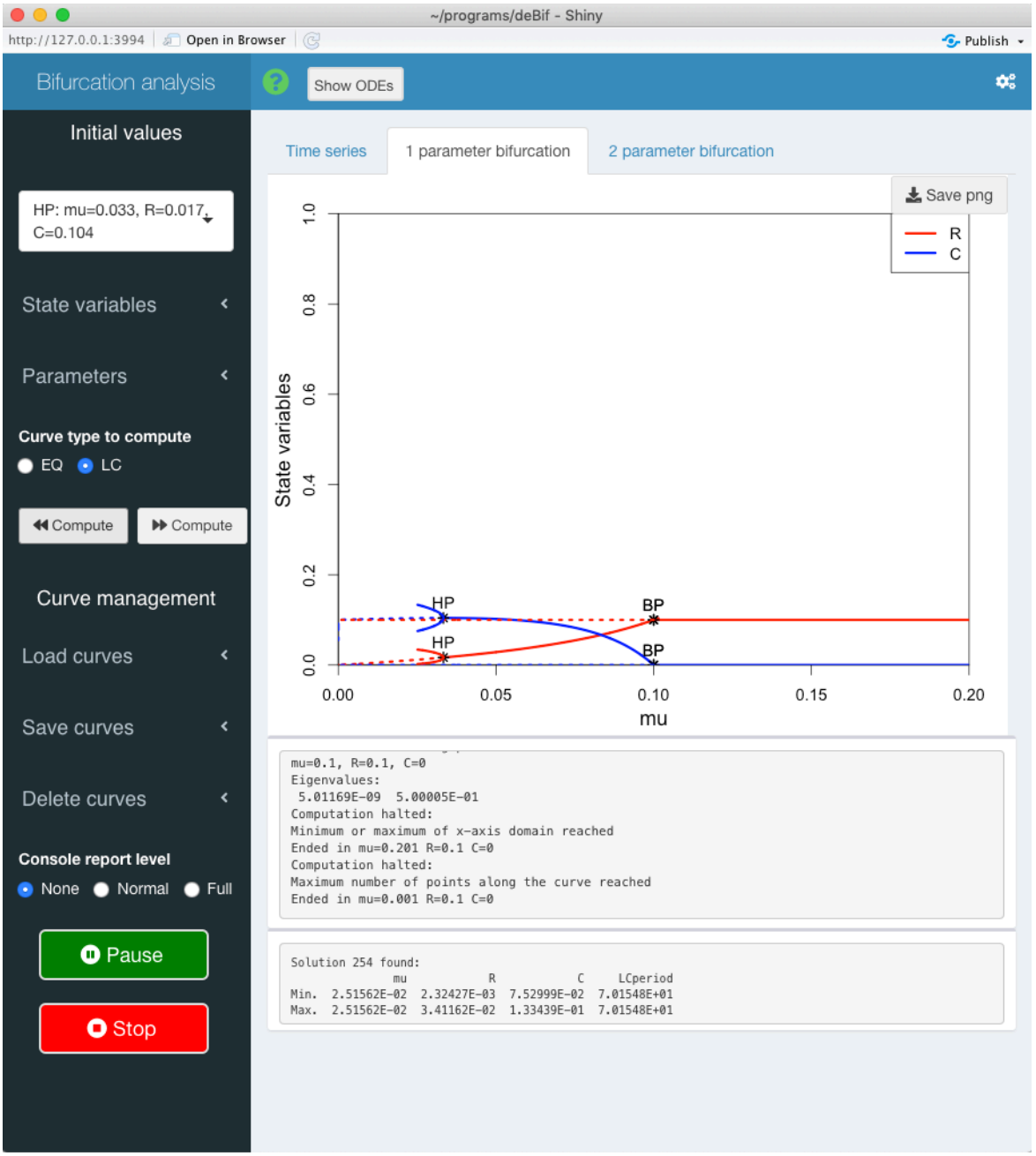

Figure 3.11: The 'bifurcation()' application during the computation of the curve of limit cycles (hence the 'Pause' and 'Stop' button) that originates from the Hopf bifurcation point (HP) to lower values of the bifurcation parameter. The maximum and minimum value of the state variables observed during the limit cycle are plotted as two separate curve sections originating from the Hopf bifurcation point (HP). The progress window reports the period of the limit cycle.

y-axis, whether or not a second state variable is selected to be shown on the secondary, right y-axis and whether or not a 2-dimensional or 3-dimensional plot is selected.

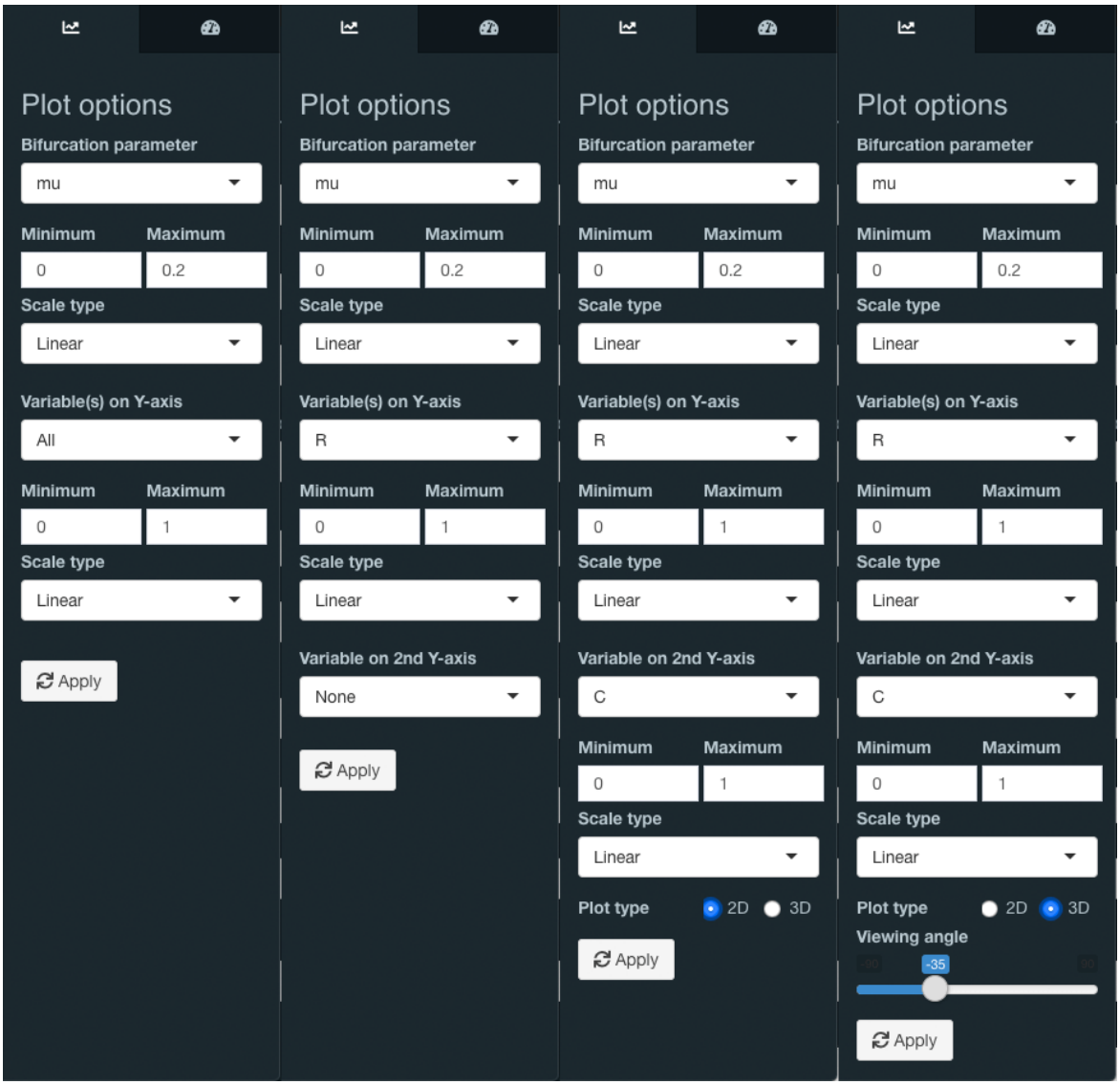

Figure 3.12: The 4 different configurations of the right sidebar of the 'bifurcation()' application window depending on which state variables are selected to be shown on the primary and secondary y-axis.

After adjusting the plot parameters in the right sidebar the `Apply` button has to be pressed to finalize the changes made. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

3.2.3 Numerical settings

The computation of steady states of the system of ODEs depends on a number of numerical settings that can be customized for better convergence to a solution point or more or less detailed

resolution along an equilibrium curve. Figure 3.13 shows the `bifurcation()` application window with the numerical options tab visible. This numerical options tab can be shown by clicking the gears icon at the right-hand side of the application window header to expand the right sidebar and subsequently clicking the dial icon in the top-right corner of the right sidebar.

The various input boxes be can used to change the following numerical settings that influence the computations of the steady states:

- *Function tolerance* and *Variable tolerance*: The root finding function implemented in the package `deBif` will consider a point to be a steady state of the system of ODEs if the norm of the right-hand side of the system of ODEs $|f(y)|$ is less than N times the function tolerance and the norm of the change in the solution point from one iteration to the next, $|y_{n+1} - y_n|$, is less than N times the variable tolerance, where N is the dimension of the solution point. Making the function or the variable tolerance smaller or larger will hence make the solutions to steady state computations more or less accurate, respectively.
- *Zero identity*: Quantities with an absolute value below this threshold value are considered equal to 0.
- *Neighbourhood tolerance*: If a solution point is found, but its relative distance to the previously computed point is larger than this threshold value, the solution point is discarded. The next solution point on a curve is hence forced to be in the close proximity of the previously computed point on the curve and is prohibited by this relative distance measure (relative to the value of the previously computed point) to jump to an entirely different solution point.
- *Jacobian perturbation*: To compute the steady state and to compute the eigenvalues of a steady state that has been located, the application computes the Jacobian matrix of the system of ODEs numerically. If the system of ODEs is written as $dx/dt = f(x)$ the Jacobian matrix is calculated by evaluating both $f(x - \Delta)$ and $f(x + \Delta)$. Here the deviation Δ is the Jacobian perturbation that can be adjusted in the numerical options tab. Taking a smaller or larger value for the Jacobian perturbation hence computes the partial derivatives by taking 2 points that are less or more apart, while surrounding the value x at which to compute the partial derivatives.
- *Minimum step size* and *Maximum step size*: These settings control the distance between the solution points that together constitute an equilibrium curve, where *Minimum step size* is an absolute and *Maximum step size* a relative measure. More specifically, the change in the fastest changing component of the solution between a computed solution point and the prediction of the next solution point is in *absolute value* never smaller than the value of *Minimum step size*. On the other hand, the change in the fastest changing component of the solution between a computed solution point and the prediction of the next solution point is in *relative value* never larger than *Maximum step size*. The realized step size is adjusted along the curve in between these two limits.
- *Maximum number of iterations*: Maximum number of iterations during one call to the root finding function. The function will return unsuccessfully if a solution point satisfying the error condition given above has not been found after adjusting the point this maximum number of times.

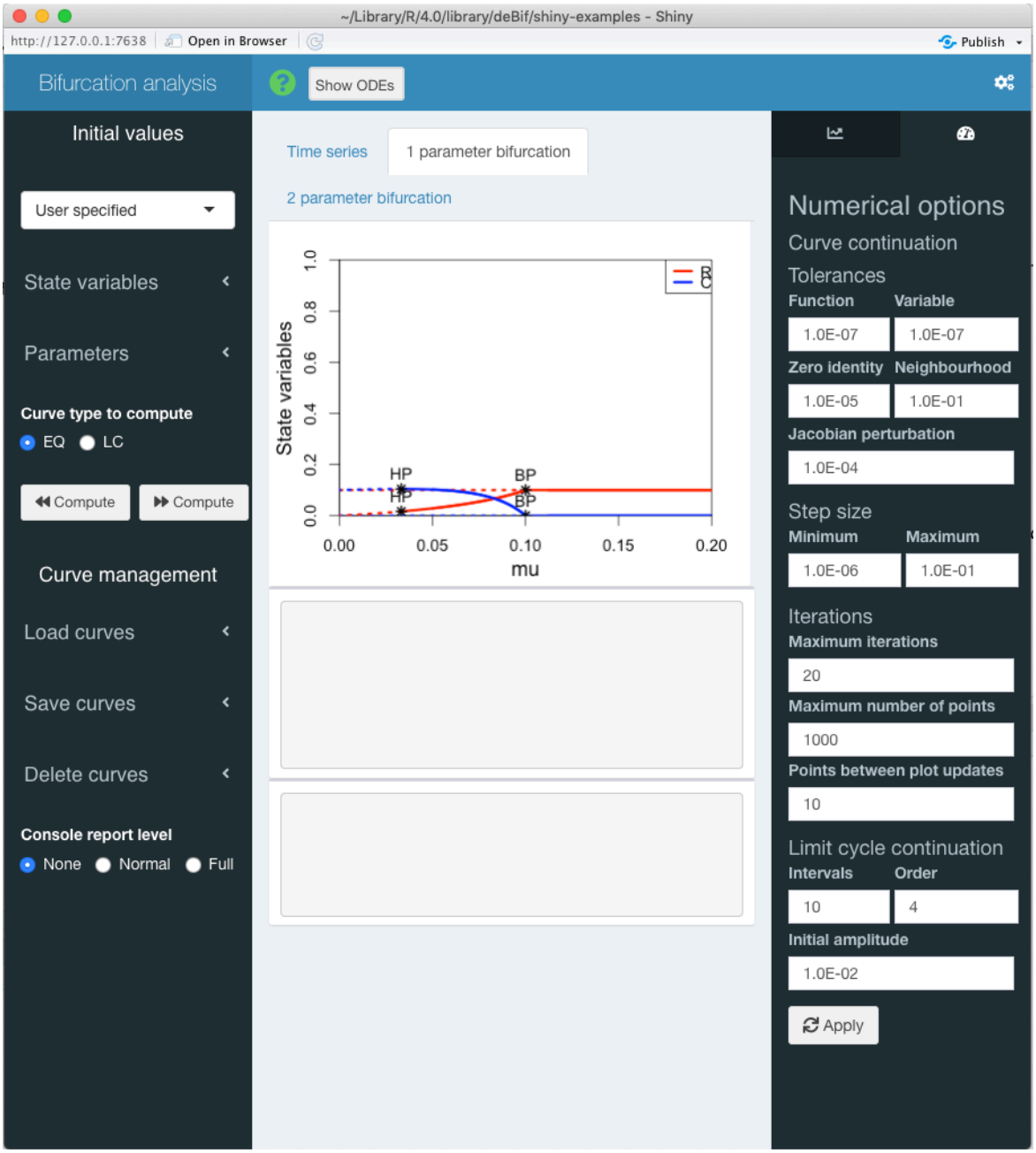

Figure 3.13: The 'bifurcation()' application window with extended right sidebar showing the numerical options tab sheet. Here numerical settings for the equilibrium curve continuations can be customized.

- *Maximum number of points*: The maximum number of points to be calculated along an equilibrium curve. Computations of equilibrium curves will halt when this maximum number of solution points along the curve has been reached or when the values of the state variables in the equilibrium point are reaching values outside the range of values currently shown on the x- and y-axis.
- *Points between plot updates*: The equilibrium curves are computed in pieces, while in between these pieces the plot frame is updated with the newly computed curve section. The number of points computed between these plot updates is given by this entry.

During the computation of a curve of limit cycles the following numerical settings are still relevant:

- *Interval* and *Order*: A limit cycle is approximated by subdividing it into intervals and representing the curve in each interval with a spline function of a specified order. The number of intervals and the order of the interpolating spline functions can be changed here. Larger values of these entries will result in more accurate approximations of the limit cycle at the expense of longer computation times.
- *Initial amplitude*: The amplitude of the initial limit cycle, which is used when starting the computation of a limit cycle curve from a Hopf bifurcation point (HP). Increase this value to force the application to speed up the computation of the initial part of a limit cycle curve.

After adjusting the numerical settings the `Apply` button has to be pressed to let them take effect. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

3.3 Bifurcation curves with 2 free parameters

The tab sheet called `2 parameter bifurcation` is used for the computation and visualization of the location of Hopf bifurcation points (HP), branching points (BP) and limit points (LP), which have been detected while computing equilibrium curves, as a function of two free parameters. For that reason both the x-axis and the y-axis of the plot in this tab sheet always represent the values of two different parameters in the system of ODEs. By default these parameters are the first two elements in the named parameter vector with which the `bifurcation()` application was started up, but this can be changed through the plot option settings available through the right sidebar (see section 3.3.1). Figure 3.14 shows the default view of the tab sheet `2 parameter bifurcation` with the plot option menu in the right sidebar expanded. The two drop-down menus labeled `1st bifurcation parameter` and `2nd bfiurcation parameter` can be used to select which parameters are varied during the computation and are hence plotted on the x- and y-axis.

Computing curves of Hopf bifurcation points (HP), branching points (BP) and limit points (LP) as a function of two free parameters, should always be started by selecting as initial point the corresponding bifurcation point that has been detected in an equilibrium curve. Hence, only after computing an equilibrium curve and only if bifurcation points are detected during its computation, the drop-down menu at the top of the left sidebar can be used to select an initial

Figure 3.14: The 'bifurcation()' application with the tab sheet '2 parameter bifurcation' selected and the plot option menu in the right sidebar expanded. The two drop-down menus in the right sidebar can be used to determine which parameters are plotted on the x- and y-axis. These parameters will also vary during the computation of curves.

point for the calculation of a bifurcation curve as function of two parameters. The computation of such a curve is started by clicking either the `<<Compute` or the `>>Compute` button, which starts computation of the curve in the direction of lower and higher values of the parameter on the x-axis, respectively.

If you encounter difficulties while computing a particular type of curve, you could try to switch the parameters on the x- and y-axis. Even though the curve will be the same, it is possible that its computation will proceed more easily and be more successful.

3.3.1 Customizing the plot

Apart from selecting which parameters in the system of ODEs are varied during the computations and are hence plotted on the x- and y-axis the plot options tab only allows for adjusting a limited number of settings. More specifically, using the input boxes the minimum and maximum value to be plotted on the x- and y-axis can be set, while the drop-down menus offer a choice between a linear and a logarithmic scale on the x- and y-axis.

After adjusting the plot parameters in the right sidebar the `Apply` button has to be pressed to finalize the changes made. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

3.3.2 Numerical settings

The computation of the bifurcation curves as a function of two parameters depends on the same numerical settings that applied to the computations of equilibrium points discussed in section 3.2.3. Refer to Figure 3.13 for the layout of the numerical options tab that allows for customization of these numerical settings, which may be opened up by clicking the gears icon at the right-hand side of the application window header and subsequently clicking the dial icon in the top-right corner of the right sidebar. The settings that apply to the computation of curves as a function of two parameters are the following:

- *Function tolerance* and *Variable tolerance*: The root finding function implemented in the package `deBif` will consider a point to be a steady state of the system of ODEs if the norm of the right-hand side of the system of ODEs $|f(y)|$ is less than N times the function tolerance and the norm of the change in the solution point from one iteration to the next, $|y_{n+1} - y_n|$, is less than N times the variable tolerance, where N is the dimension of the solution point. Making the function or the variable tolerance smaller or larger will hence make the solutions to steady state computations more or less accurate, respectively.
- *Zero identity*: Quantities with an absolute value below this threshold value are considered equal to 0.
- *Neighbourhood tolerance*: If a solution point is found, but its relative distance to the previously computed point is larger than this threshold value, the solution point is discarded. The next solution point on a curve is hence forced to be in the close proximity of the previously computed point on the curve and is prohibited by this relative distance measure

(relative to the value of the previously computed point) to jump to an entirely different solution point.

- *Jacobian perturbation*: To locate the bifurcation points the application numerically computes the Jacobian matrix of the system of equations that has to be solved. If this system of equations is written as $f(x) = 0$ the Jacobian matrix is calculated by evaluating both $f(x - \Delta)$ and $f(x + \Delta)$. Here the deviation Δ is the Jacobian perturbation that can be adjusted in the numerical options tab. Taking a smaller or larger value for the Jacobian perturbation hence computes the partial derivatives by taking 2 points that are less or more apart, while surrounding the value x at which to compute the partial derivatives.
- *Minimum step size* and *Maximum step size*: These settings control the distance between the solution points that together constitute an equilibrium curve, where *Minimum step size* is an absolute and *Maximum step size* a relative measure. More specifically, the change in the fastest changing component of the solution between a computed solution point and the prediction of the next solution point is in *absolute value* never smaller than the value of *Minimum step size*. On the other hand, the change in the fastest changing component of the solution between a computed solution point and the prediction of the next solution point is in *relative value* never larger than *Maximum step size*. The realized step size is adjusted along the curve in between these two limits.
- *Maximum number of iterations*: Maximum number of iterations during one call to the root finding function. The function will return unsuccessfully if a solution point satisfying the error condition given above has not been found after adjusting the point this maximum number of times.
- *Maximum number of points*: The maximum number of points to be calculated along a curve. Computations of bifurcation curves will halt when this maximum number of solution points along the curve has been reached or when the two free parameters are reaching values outside the current ranges of the *x*- and *y*-axis.
- *Points between plot updates*: The curves are computed in pieces, while in between these pieces the plot frame is updated with the newly computed curve section. The number of points computed between these plot updates is given by this entry.

After adjusting the numerical settings the `Apply` button has to be pressed to let them take effect. Subsequently, the gears icon at the right-top corner of the application window can be pressed to hide the right sidebar again.

3.4 Saving graphs

All tab sheets of the `bifurcation()` program show at the top-right corner of the plot frame a button that allows you to download an image of the current graph. Clicking this download icon will open up a file dialog box in which you can specify the name of the PNG or PDF file, in which to save the image of the plot. Whether the plot is saved to a PNG or a PDF file can be determined with the command-line argument `saveplotas` (see section 3.7). More elaborate graphs can of course be constructed by exporting the computed curves as explained below and using other packages in R to visualize the results.

3.5 Curve management

The bottom part of the left sidebar of the `bifurcation()` application shows 3 sub-menus that allow for manipulation of the computed curves. Curves can be read into the `bifurcation()` application from variables that are present in the global R environment from which the application was started, computed curves can be saved to variables in this global R environment and curves can be deleted. Figure 3.15 shows 4 snapshots of the left sidebar with the different sub-menus expanded and in different states.

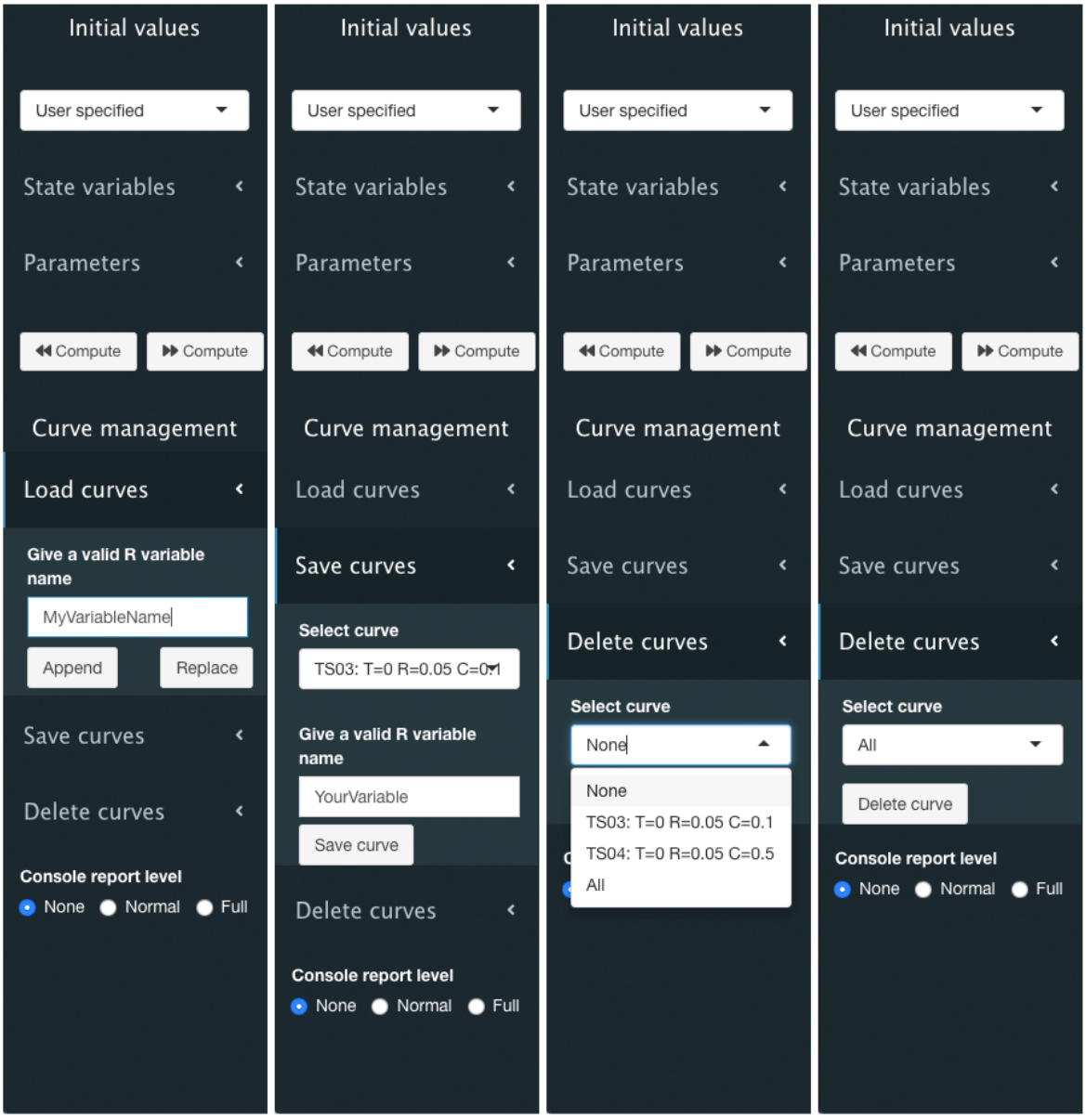

Figure 3.15: The 'bifurcation()' application window with the left sidebar showing the 'Load curves' submenu, the 'Save curves' submenu, the 'Delete curves' submenu with the curve choice drop-down menu expanded and the 'Delete curves' submenu with the 'All' option selected. The action buttons in each submenu allow to load, save or delete the selected curves.

3.5.1 Load curves

By expanding the `Load curves` sub-menu an input box shows up in which you can enter the name of a variable (*do not use any quotes!*) that is present in your global R environment (see Figure 3.15). This variable can contain time series curves, equilibrium bifurcation curves with 1 free parameter or bifurcation curves with 2 free parameters. As long as the global variable you want to load has been saved by the `bifurcation()` application, it should be possible to be read in. But this is of course only taking effect if the curves stored in the global variable pass some tests. For example, the curves in the global variable should contain all the same state variables and parameters as the state variables and parameters in the system of ODEs that is currently being analysed with the `bifurcation()` application.

Independent of which of the 3 application tabs (`Time series`, `1 parameter bifurcation` or `2 parameter bifurcation`) is active when you load curves from a global variable, all the valid curves in this global variables will be read in, independent of their type.

The two action buttons in the `Load curves` sub-menu allow you to either append the curves in the global variable that you are loading to the current set of curves stored by the application or replace the entire set of curves that is currently stored by the `bifurcation()` application. The console box below the plot frame will report on the success or failure of the load operation.

3.5.2 Save curves

Expanding the `Save curves` sub-menu reveals a curve selection drop-down menu, an input box in which you can enter your name of choice for the variable in your global R environment, in which to store curves, and an action button to execute the save action (see Figure 3.15). The curve to save can be selected via the curve selection drop-down menu. This drop-down menu lists in addition to the options `None` (the default choice) and `All` (the last choice) the labels of all the curves that are stored for the currently active tab (`Time series`, `1 parameter bifurcation` or `2 parameter bifurcation`). Either one specific curve can be selected for saving or all curves. It is not possible to save 2 out of 3 curves at the same time. In the input box you have to enter a valid name (*do not use any quotes!*) for a variable in the global R environment from which the `bifurcation()` application was started. Take care with specifying this variable name, the application will overwrite an already existing variable.

Pressing the `Save` button will save the selected curve or all curves to the variable with the specified name.The console box below the plot frame will report on the success or failure of the save operation. The variable will, however, only show up in your Rstudio Environment panel that shows the list of variables in your global R environment after the `bifurcation()` application has exited. I have not found a way to update the contents of this Rstudio Environment panel while the `bifurcation()` application is still active.

3.5.2.1 Delete curves

Expanding the `Delete curves` sub-menu reveals a curve selection drop-down menu and an action button to execute the delete action (see Figure 3.15). The curve to delete can be selected via the curve selection drop-down menu. This drop-down menu lists in addition to the options `None` (the default choice) and `All` (the last choice) the labels of all the curves that are stored for the currently active tab (`Time series`, `1 parameter bifurcation` or `2 parameter`

bifurcation). Either one specific curve can be selected for deleting or all curves. It is not possible to delete 2 out of 3 curves at the same time.

Pressing the Delete button will delete the selected curve or all curves in the currently active tab (Time series, 1 parameter bifurcation or 2 parameter bifurcation).The console box below the plot frame will report on the success or failure of the operation.

3.6 Saving program settings on exit

Whenever the program exits all the curves that the bifurcation() application has computed and has currently stored in memory are saved in a global variable <model>BifCurves, where the sub-string <model> is the name of the function describing the dynamics, which is passed as first argument to bifurcation(). Similarly, all programs settings, such as the plot settings on the 3 different tabs and the numerical settings for the time integration and curve continuation are saved in the global variable <model>BifSettings. These two global variables will show up on return to your R environment after the programs has ended. When present, the global variables <model>BifCurves and <model>BifSettings will be read by the bifurcation() application the next time you start it up. They will only take effect, however, if the variables pass some tests successfully. For example, if the variable <model>BifCurves does not contain the correct set of state variables and parameters it will be ignored when you start up the bifurcation() application.

The variables <model>BifCurves and <model>BifSettings are overwritten every time the bifurcation() application ends. Therefore, if you want to save a particular set of curves and the application settings, you have to copy these variables to new variables with your own unique names. At a later time these saved variables can be assigned to the variables <model>BifCurves and <model>BifSettings again to start the bifurcation() application at the point where you left off.

The variables <model>BifCurves and <model>BifSettings will be ignored completely if the bifurcation() application is started with the command-line argument resume = FALSE. The default of this command-line argument is resume = TRUE.

3.7 Optional command-line arguments

The bifurcation() application allows for a number of additional arguments that can be included at the command line to tweak graphical default values used by the application. These arguments are all optional and adopt default values when not specified on the command line.

The arguments that can be included on the command line are:

- lwd: The line width used for plotting computed curves. The default value is 3.
- cex: The base font size for labeling axes and legends in the plots. The default value is 1.2.
- tcl.len: The length of ticks on the axes. The default value is 0.03.
- bifsym: The symbol used to mark a bifurcation point in an equilibrium curve. The default value is 8.

- `biflblpos`: The location of label of a bifurcation point. Values of 1, 2, 3 and 4, respectively, indicate positions below, to the left of, above and to the right of the symbol marking the bifurcation point. The default value is 3.
- `unstablelty`: The line style to be used for curve sections representing unstable equilibrium points. The default value is 3, which specifies dotted lines to be used.
- `saveplotas`: Either "pdf" or "png" to save the plot to a PDF or PNG file. The default value is "png".

References

Chang, W. and Borges Ribeiro, B. (2018). *shinydashboard: Create Dashboards with 'Shiny'*. R package version 0.7.1.

Chang, W., Cheng, J., Allaire, J., Xie, Y., and McPherson, J. (2019). *shiny: Web Application Framework for R*. R package version 1.4.0.

Granjon, D. (2019). *shinydashboardPlus: Add More 'AdminLTE2' Components to 'shinydashboard'*. R package version 0.7.0.

Kuznetsov, Y. A. (1995). *Elements of applied bifurcation theory*. Springer-Verlag, Heidelberg.

Soetaert, K. (2009). *rootSolve: Nonlinear root finding, equilibrium and steady-state analysis of ordinary differential equations*. R package 1.6.

Soetaert, K., Petzoldt, T., and Setzer, R. W. (2010). Solving differential equations in R: Package deSolve. *Journal of Statistical Software*, 33(9):1–25.