DiffEqBiological Tutorial III: Steady-States and Bifurcations

Torkel Loman and Samuel Isaacson

September 3, 2021

Several types of steady state analysis can be performed for networks defined with DiffEqBiological by utilizing homotopy continuation. This allows for finding the steady states and bifurcations within a large class of systems. In this tutorial we'll go through several examples of using this functionality.

We start by loading the necessary packages:

```
using DiffEqBiological, Plots
gr(); default(fmt = :png);
```

```
Error: ArgumentError: Package DiffEqBiological not found in current path:
- Run `import Pkg; Pkg.add("DiffEqBiological")` to install the DiffEqBiolog
ical package.
```

0.0.1 Steady states and stability of a biochemical reaction network.

Bistable switches are well known biological motifs, characterised by the presence of two different stable steady states.

```
bistable_switch = @reaction_network begin
    d,    (X,Y) → ∅
    hillR(Y,v1,K1,n1), ∅ → X
    hillR(X,v2,K2,n2), ∅ → Y
end d v1 K1 n1 v2 K2 n2
d = 0.01;
v1 = 1.5; K1 = 30; n1 = 3;
v2 = 1.; K2 = 30; n2 = 3;
bistable_switch_p = [d, v1 ,K1, n1, v2, K2, n2];
```

```
Error: LoadError: UndefVarError: @reaction_network not defined
in expression starting at /var/lib/buildkite-agent/builds/1-amdci4-julia-cs
ail-mit-edu/julialang/scimltutorials-dot-jl/tutorials/models/04b-diffeqbio_
III_steadystates.jmd:2
```

The steady states can be found using the `steady_states` function (which takes a reaction network and a set of parameter values as input). The stability of these steady states can be found using the `stability` function.

```
ss = steady_states(bistable_switch, bistable_switch_p)
```

```
Error: UndefVarError: steady_states not defined
```

```
stability(ss,bistable_switch, bistable_switch_p)
```

```
Error: UndefVarError: stability not defined
```

Since the equilibration methodology is based on homotopy continuation, it is not able to handle systems with non-integer exponents, or non polynomial reaction rates. Neither of the following two systems will work.

This system contains a non-integer exponent:

```
rn1 = @reaction_network begin
    p, ∅ → X
    hill(X,v,K,n), X → ∅
end p v K n
p1 = [1.,2.5,1.5,1.5]
steady_states(rn1,p1)
```

```
Error: LoadError: UndefVarError: @reaction_network not defined
in expression starting at /var/lib/buildkite-agent/builds/1-amdci4-julia-cs
ail-mit-edu/julialang/scimltutorials-dot-jl/tutorials/models/04b-diffeqbio_
III_steadystates.jmd:2
```

This system contains a logarithmic reaction rate:

```
rn2 = @reaction_network begin
    p, ∅ → X
    log(X), X → ∅
end p
p2 = [1.]
steady_states(rn2,p2)
```

```
Error: LoadError: UndefVarError: @reaction_network not defined
in expression starting at /var/lib/buildkite-agent/builds/1-amdci4-julia-cs
ail-mit-edu/julialang/scimltutorials-dot-jl/tutorials/models/04b-diffeqbio_
III_steadystates.jmd:2
```

0.0.2 Bifurcation diagrams for biochemical reaction networks

Bifurcation diagrams illustrate how the steady states of a system depend on one or more parameters. They can be computed with the `bifurcations` function. It takes the same arguments as `steady_states`, with the addition of the parameter one wants to vary, and an interval over which to vary it:

```
bif = bifurcations(bistable_switch, bistable_switch_p, :v1, (.1,5.))
plot(bif,ylabel="[X]",label="")
plot!([[],[]],color=[:blue :red],label = ["Stable" "Unstable"])
```

```
Error: UndefVarError: bifurcations not defined
```

The values for the second variable in the system can also be displayed, by giving that as an additional input to `plot` (it is the second argument, directly after the bifurcation diagram object):

```
plot(bif,2,ylabel="[Y]")
plot!([[],[]],color=[:blue :red],label = ["Stable" "Unstable"])
```

```
Error: UndefVarError: plot not defined
```

The `plot` function also accepts all other arguments which the Plots.jl `plot` function accepts.

```
bif = bifurcations(bistable_switch, bistable_switch_p,:v1,(.1,10.))
plot(bif,linewidth=1.,title="A bifurcation diagram",ylabel="Steady State concentration")
plot!([[],[]],color=[:blue :red],label = ["Stable" "Unstable"])
```

```
Error: UndefVarError: bifurcations not defined
```

Certain parameters, like `n1`, cannot be sensibly varied over a continuous interval. Instead, a discrete bifurcation diagram can be calculated with the `bifurcation_grid` function. Instead of an interval, the last argument is a range of numbers:

```
bif = bifurcation_grid(bistable_switch, bistable_switch_p,:n1,1.:5.)
plot(bif)
scatter!([[],[]],color=[:blue :red],label = ["Stable" "Unstable"])
```

```
Error: UndefVarError: bifurcation_grid not defined
```

0.0.3 Bifurcation diagrams over two dimensions

In addition to the bifurcation diagrams illustrated above, where only a single variable is varied, it is also possible to investigate the steady state properties of s system as two different parameters are varied. Due to the nature of the underlying bifurcation algorithm it is not possible to continuously vary both parameters. Instead, a set of discrete values are selected for the first parameter, and a continuous interval for the second. Next, for each discrete value of the first parameter, a normal bifurcation diagram is created over the interval given for the second parameter.

```
bif = bifurcation_grid_diagram(bistable_switch, bistable_switch_p,:n1,0.:4.,:v1,(.1,5.))
plot(bif)
plot!([[],[]],color=[:blue :red],label = ["Stable" "Unstable"])
```

```
Error: UndefVarError: bifurcation_grid_diagram not defined
```

In the single variable case we could use a `bifurcation_grid` to investigate the behavior of a parameter which could only attain discrete values. In the same way, if we are interested in two parameters, both of which require integer values, we can use `bifrucation_grid_2d`. In our case, this is required if we want to vary both the parameters `n1` and `n2`:

```
bif = bifurcation_grid_2d(bistable_switch, bistable_switch_p,:n1,1.:3.,:n2,1.:10.)
plot(bif)
scatter!([[],[]],color=[:blue :red],label = ["Stable" "Unstable"])
```

```
Error: UndefVarError: bifurcation_grid_2d not defined
```

0.0.4 The Brusselator

The Brusselator is a well know reaction network, which may or may not oscillate, depending on parameter values.

```
brusselator = @reaction_network begin
    A, ∅ → X
    1, 2X + Y → 3X
    B, X → Y
    1, X → ∅
```

```
end A B;
A = 0.5; B = 4.;
brusselator_p = [A, B];
```

```
Error: LoadError: UndefVarError: @reaction_network not defined
in expression starting at /var/lib/buildkite-agent/builds/1-amdci4-julia-cs
ail-mit-edu/julialang/scimltutorials-dot-jl/tutorials/models/04b-diffeqbio_
III_steadystates.jmd:2
```

The system has only one steady state, for $(X, Y) = (A, B/A)$ This fixed point becomes unstable when $B > 1 + A^2$, leading to oscillations. Bifurcation diagrams can be used to determine the system's stability, and hence look for where oscillations might appear in the Brusselator:

```
bif = bifurcations(brusselator,brusselator_p,:B,(0.1,2.5))
plot(bif,2)
plot!([[],[],[],[]],color=[:blue :cyan :orange :red],label = ["Stable Real" "Stable
Complex" "Unstable Complex" "Unstable Real"])
```

```
Error: UndefVarError: bifurcations not defined
```

Here red and yellow colors label unstable steady-states, while blue and cyan label stable steady-states. (In addition, yellow and cyan correspond to points where at least one eigenvalue of the Jacobian is imaginary, while red and blue correspond to points with real-valued eigenvalues.)

Given `A=0.5`, the point at which the system should become unstable is `B=1.25`. We can confirm this in the bifurcation diagram.

We can also investigate the behavior when we vary both parameters of the system:

```
bif = bifurcation_grid_diagram(brusselator,brusselator_p,:B,0.5:0.02:5.0,:A,(0.2,5.0))
plot(bif)
plot!([[],[],[],[]],color=[:blue :cyan :orange :red],label = ["Stable Real" "Stable
Complex" "Unstable Complex" "Unstable Real"])
```

```
Error: UndefVarError: bifurcation_grid_diagram not defined
```

0.1 Getting Help

Have a question related to DiffEqBiological or this tutorial? Feel free to ask in the DifferentialEquations.jl Gitter. If you think you've found a bug in DiffEqBiological, or would like to request/discuss new functionality, feel free to open an issue on Github (but please check there is no related issue already open). If you've found a bug in this tutorial, or have a suggestion, feel free to open an issue on the SciMLTutorials Github site. Or, submit a pull request to SciMLTutorials updating the tutorial!

```
Error: MethodError: no method matching tutorial_footer(::String, ::String;
remove_homedir=true)
Closest candidates are:
  tutorial_footer(::Any, ::Any) at /var/lib/buildkite-agent/builds/1-amdci4
-julia-csail-mit-edu/julialang/scimltutorials-dot-jl/src/SciMLTutorials.jl:
79 got unsupported keyword argument "remove_homedir"
```

```
  tutorial_footer(::Any) at /var/lib/buildkite-agent/builds/1-amdci4-julia-
csail-mit-edu/julialang/scimltutorials-dot-jl/src/SciMLTutorials.jl:79 got
unsupported keyword argument "remove_homedir"
```