Universität Osnabrück

Studienprojekt

Betreut durch

Prof. Dr. Frank Hilker

Wintersemester 2023

Host-Parasitoid Interactions

---

verfasst von

Robin
Elias

968504

Bachelor of Science

Abgegeben am January 23, 2024

Contents

List of Figures

1 Introduction

The Earth's biodiversity is made up of around 1.7 million recorded species. Of these, about 57% are insects, 25% plants, fungi and microorganisms and 20% vertebrates and other animals. Therefore there are nearly 1 million different insect species. (El-Kholy et al., 1992, S. 35ff) About 10% of all described insect species are parasitoids (Eggleton and Belshaw, 1997), which corresponds to about 100 thousand species. This leads to the conclusion that parasitism is a realy common thing to expect in a natural environment. For this reason Thompson developed the first host-parasitoid models in the beginning of the 20th century, motivated by the possibility of regulating agricultural pests through the controlled use of parasitoids. These models were very simple and their predictions rather divorced from the often complex behavior found in the field. Therefore numerous other models have been developed. But as today as then, research faces a number of challenges. Although the data has grown over the years there is still a lack of mechanistic understanding of the interactions and complex dynamics that can occur in such a real life system or corresponding model.

When such highly non-linear behavior occurs, analytical mathematic's quickly reaches it's limits. However, thanks to the rapid development of computer technology and numerical algorithms for investigating such complex system behavior, it is now possible to gain insights into such models. This report is intended to show how efficient and convincing these algorithms have become and what results can be expected and achieved. For this purpose, we use cutting-edge algorithms developed in the Julia programming language.

2 Model and Methods

2.1 The Model

The modeling of host-parasitoid interactions has a long history. Since insects often have separate generations, discrete-time models are suitable. In general all our models base on the following assumptions:

$$\begin{aligned} N_{t+1} &= g(N_t)f(N_t, P_t) \\ P_{t+1} &= bN_t[1 - f(N_t, P_t)] \end{aligned} \tag{1}$$

Given this equations $g(N_t)$ describes the growth of the hosts. Because parasitoid act lethal on the host population, only not infested one are fertile and can reproduce. Therefore $f(N_t, P_t)$ is the proportion of hosts that is not infested by parasites. The parameter b indicates the average number of parasites that emerge from one parasitized host.

Previos research by (Kaitala, Ylikarjula, and Heino, 1999) build on the assumption that the growth of host following the Ricker map given by:

$$g(N_t) = N_t e^{r(1-N_t)} \tag{2}$$

About this Ricker map is known that in respect to the intrinsic growth rate r the behavior changes from stable, over period doubling to chaos. However we want to reduce this impact, therefore we use the Beverton-Holt map. This map is also a discrete-time equivalent to logistic growth, which shows stable behavior and approaches the carring capacity in every permitted parameter set.

The Beverton-Holt Model ist given by the following equation

$$g(N_t) = \frac{\lambda N_t}{1 + \frac{(\lambda-1)N_t}{K}} \tag{3}$$

with the intrinsic growth rate $\lambda > 0$ and the capacity $K > 0$.

We assume that the interaction between host and parasitoid is density-dependent with a holling type III and that the host and the parasitoid encounter randomly N_{enc} times. Since it is possible that one host can be encountert multiple times by different parasitoids. The number each single host is encountered is poisson distributed and therefore the zero-term represents the proportion of not infected host.

This results in:

$$f(N_t, P_t) = e^{-\frac{N_{enc}}{N_t}} = e^{\left[-\frac{aTN_tP_t}{1+cN_t+aT_hN_t^2}\right]} \tag{4}$$

with the per capita encounters $\frac{N_{enc}}{N_t}$, the total time per generation T, the handling time T_h and the parasitization rate a. (Hassell, 2000, see. chapter 2.4.1)

Through dedimensionalization with

$$n_t = \frac{N_t}{K} \qquad p_t = \frac{P_t}{bK} \qquad \alpha = abTK^2 \qquad \mu = aT_hK^2 \tag{5}$$

one receives:

$$\begin{aligned} n_{t+1} &= \frac{\lambda n_t}{1+(\lambda-1)n_t} e^{\frac{-\alpha n_t p_t}{1+\mu n_t^2}} \\ p_{t+1} &= n_t \left(1 - e^{\frac{-\alpha n_t p_t}{1+\mu n_t^2}}\right) \end{aligned} \tag{6}$$

with the growth rate $\lambda > 0$, the scaled parasitization rate $\alpha > 0$ and the scaled handling time $\mu > 0$

2.2 The Algorithms

The project was implemented in Julia[1] and uses the DrWatson package. To reproduce the results look at README.md on github.com[2]

[1] https://docs.julialang.org/en/v1/

[2] https://github.com/relias96/Studienprojekt

Because of the complex behavior the existance and stability of an Attractor is hardly analytical shown. Therefore we investigate this behavior by using a few numerical methods. We choose the Julia programming Language because many required functionalities are already implemented, with easy readable code and great performance.

2.2.1 Orbit-diagramm

An orbit-diagramm is often mistakenly called bifurcation-diagramm. One thing in common is that a parameter is varied over a range, but while an orbit-diagramm only shows attracting equilibria/cycles or possible locations of chaos, a bifurcation diagram in contrast plots all attracting and repelling periodic points (Ross and Sorensen, 2000).

While the computation of the bifurcation diagram can be very complex and time-consuming, the computation of an orbit diagram is relatively simple. Since an orbit diagramm is sufficient for our research, we will use it in the following. Because there are co-existing attractors, we implement a algorithm in the following manner:

For each parameter value on the x-axis, we randomly vary the initial conditions several times to depict even unlikely attractors. Then we evolve the system `m` times and record the following `n` states. If the attractor is chaotic, which means that the number of attractor-states with unique values is greater than the limit up to which we want to capture periodic behavior, we plot blurred dots. This ensures that the periodic and stable points are in contrast with the chaotic ones.

Algorithm 1 Orbitdiagram

```
function TIMESERIES(system, n, m)
    evolve system m times
    ts ← evolve system n times
    return ts
end function
procedure PLOTORBIT
    system ← Dict(equationOfMotion; parameter; initialState)
    m ← transient time
    n ← tracked steps
    p_values ← range(parameter)
    limit ← maximal period
    for all p ∈ parameter_values do
        repeat
            system.initialState ← u_0
            attractor ← Set(TIMESERIES(system, n, m))
            if length(attractor) < limit then
                PLOT(p,attractor) as dots
            else
                PLOT(p,attractor) as blurred dots
            end if
        until Plot is detailed enough
    end for
end procedure
```

2.2.2 Basin of attraction

If there are at least two coexisting attractors that depend on the initial condition, the basin of attraction proves to be a valuable tool. It uses a numerical approach to partition the statespace of initial conditions into different regions. Each of these regions represents a set of initial conditions that lead to its corresponding attractor.

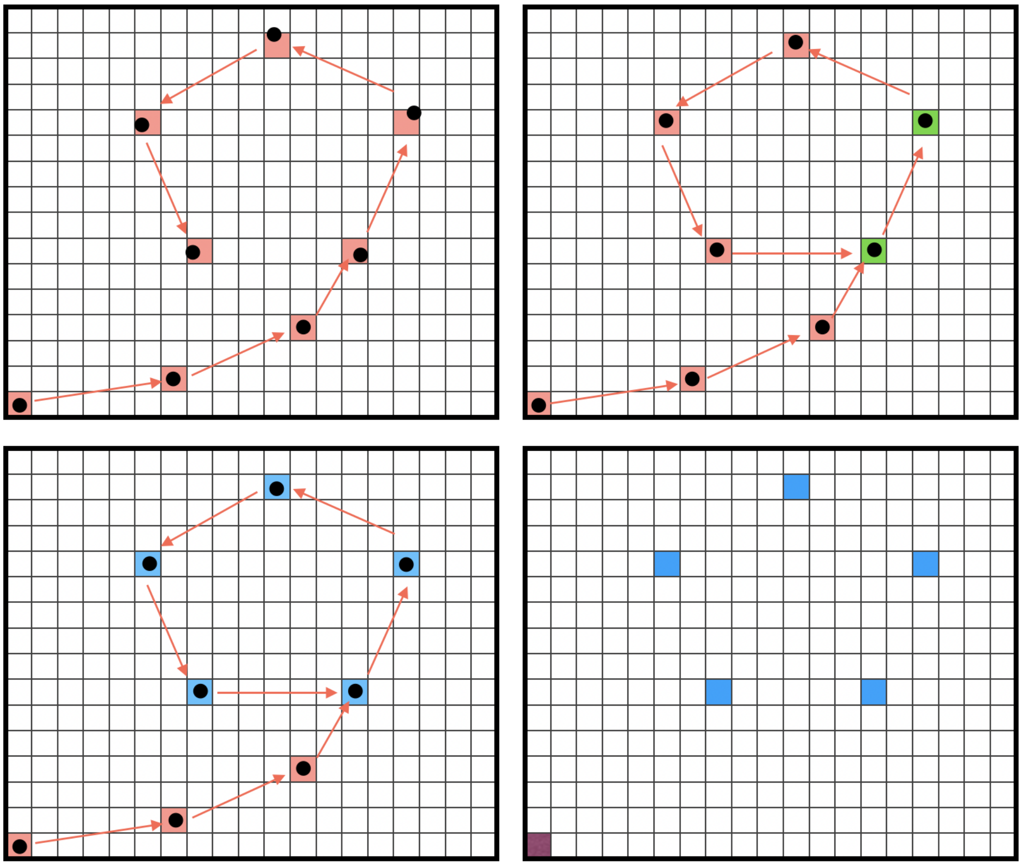

Figure 1: Basin of Attraction

Figure 1 shows the algorithm behind the function `basins_of_attraction` which relies on a method described in Chapter 7 of (Nusse and Yorke, 2012). To get the basin of attraction we use a finite state machine. The state space is devided into a discrete grid initialized with a value *unvisited*. For each inital condition we track the trajectory of the System by using the mapper `AttractorsViaRecurrences` which numerically evolves the given System using the `trajectory` function from the `DynamicalSystems` libary and tracks its path in the Statespace. Once the trajectory hits a *unvisited* cell the value changes to *visited* (upper left of Figure 1).

The Algorithm terminates when one counter hits its following corresponding threshold:

- `consecutive_recurrences` times an *visited* cells get hit
- `consecutive_basin_steps` times basin cells with same *ID* gets hit
- `consecutive_attractor_steps` times attractor cells with same *ID* get hit

When the `consecutive_recurrences` threshold is reached the new attractor is found and the cell in the grid coresponding to the initial condition gets an new *ID* which identifies the the basin to the attractor. Otherwise when `consecutive_basin_steps` or `consecutive_attractor_steps` threshold is reached the existing *ID* gets assigned to the cell, adding it to an existing basin.

Following this algorithm each inital conditions can be matched with one of the Attractors to receive the basin of attraction.

Algorithm 2 Algorithm to compute the basin of attraction

```
procedure basinOfAttraction
    system ← Dict(equationOfMotion; parameter; initialState)
    grid ← grid of empty cells
    attractors ← [ ]
    for all cell ∈ grid do
        coordinate, ID ← cell
        system.initialState ← coordinate
        ID ← AttractorsViaRecurrences(system, grid, attractors)
    end for
end procedure
```

Algorithm 3 Algorithm mapping an initial State to an Attractor

```
function AttractorsViaRecurrences(system, grid, attractors)
    tresholds
    counters
    while ∀ counters < tresholds do
        next_cell ← evolve(system, Δt)
        adjustCounters(counters, next_cell)
        if next_cell == empty then
            next_cell = visited
        end if
    end while
    if consecutive_recurrences treshold is reached then
        new_attractor ← GetAttractor(system, grid)
        attractors.add(new_attractor)
        return newID
    end if
    if consecutive_basin_steps treshold is reached then
        ID ← BasinID
        return ID
    end if
    if consecutive_attractor_steps treshold is reached then
        ID ← AttractorID
        return ID
    end if
end function
```

For a more detailed description see ‘Effortless estimation of basin of attraction’ by Datseries (Datseris and Wagemakers, 2022) or have a look into the documentation[1] .

[1] https://juliadynamics.github.io/Attractors.jl/dev/attractors/

3 Results

Since we are using dedimensionalised equations and do not have a real-life scenario to fit them into, our aim is to gain a qualitative understanding of the model. Previous research demonstrated the peculiar behaviour of their Host-Parasitoid Model when assuming host growth following a Ricker map (Kaitala, Ylikarjula, and Heino, 1999). To mitigate the destabilizing impacts caused by the chaotic behaviour of the Ricker map at certain parameter values, the Beverton-Holt map was selected for its stable behaviour across all parameters.

3.1 Transient behavior

At first we discovered long transient phases before the Model reaches its final state. This means that it takes a unexpected long time of inconclusive behavior before the behavior settles to a state. In Figure 2 the Model ($\lambda = 2.5$; $\alpha = 17.23$; $\mu = 0.1$) reaches it's Equilibrium after 8000 generations.

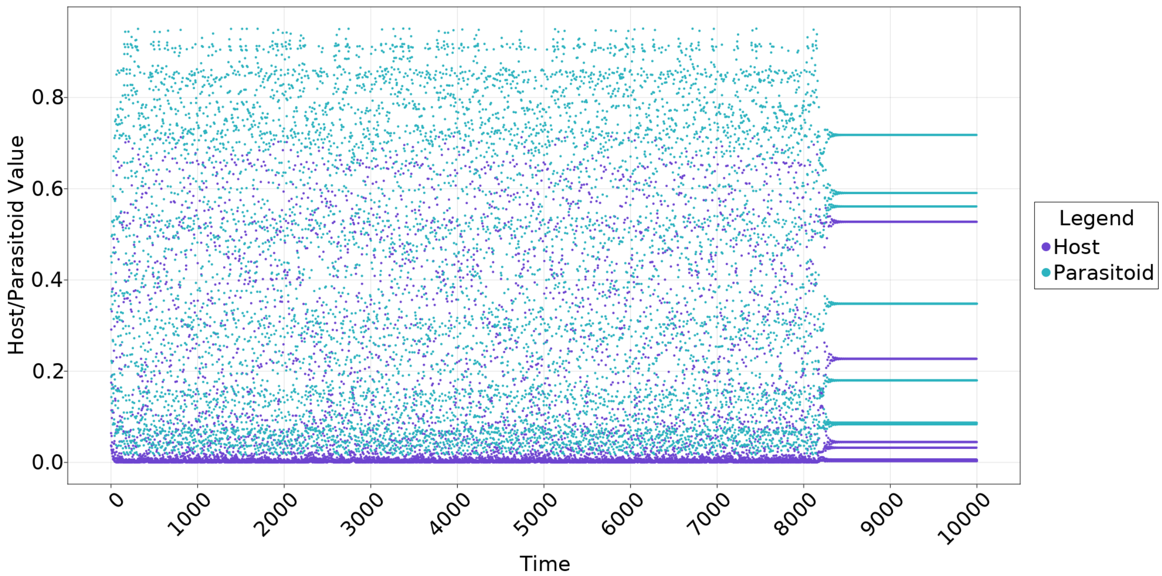

Figure 2: Transient behavior

3.2 Intermittency

For some parameter combination intermittency occurs in our Model. This means that a equlibrium is no longer stable, but instead switches randomly between states. In figure 3 the timeseries for the parameters $\lambda = 6.5$; $\alpha = 21.411$; $\mu = 0.1$ is plotted. It is rare to find in this model, but it is possible.

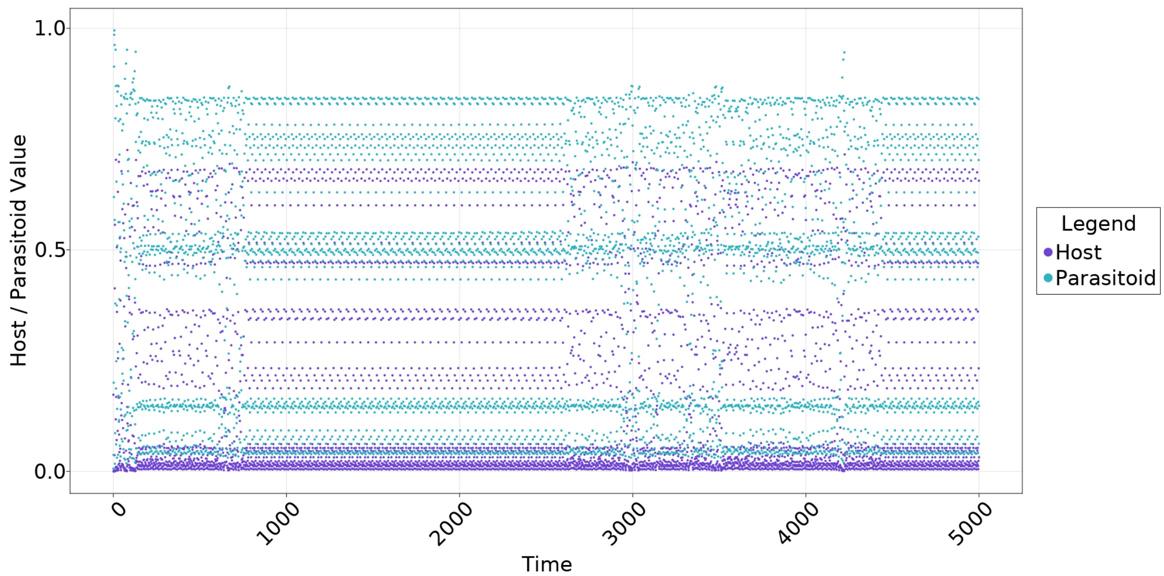

Figure 3: Intermittency

3.3 Non-unique dynamics and basin of attraction

Our Model can include several coexisting attractors. This means that for the same parameters the system can end up in different attractors. To explore the presence of the attractors we make use of the orbitdiagram defined in Algorithm 1, to track all attractors for a given parameter combination.

For the Parameters $\lambda = 7$ and $\mu = 2.5$ the following plots are produced.

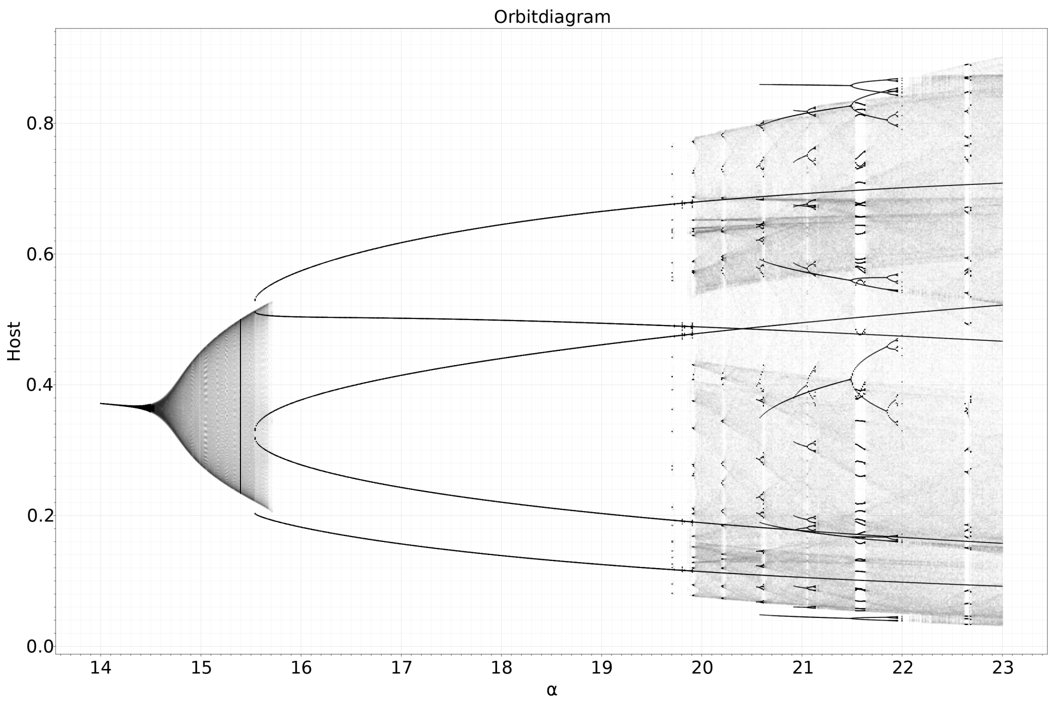

Figure 4: Orbitdiagram

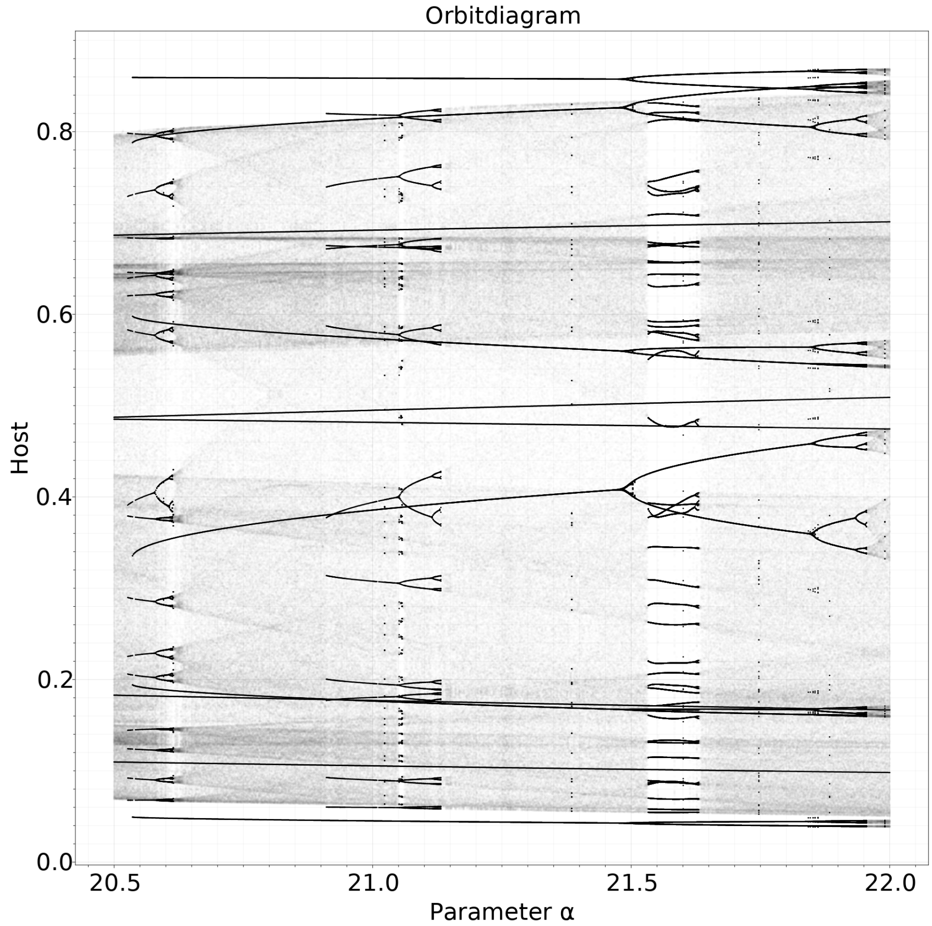

Figure 5: Detailed-Orbitdiagram

When $\alpha < 14$ there is only on stable attractor. At around 14 the stable attractor changes to an quasiperiodic attractor with an growing period. At around $\alpha = 15.5$ a periodic attractor with period 5 appears, which persists for higher α-values.

At around $\alpha = 20$ chaotic behaviour can appear. Multiple times an additional attractor arises which first goes through period doubling and then turns into a chaotic attractor or disappears. Whenever there are multiple coexisting attractors the algorithm defined in Algorithm 2 can be used to get the boundaries in the statespace to distinguish between the attractors. In Figure 6 and 7 two basins of attraction for $\lambda = 7$ and $\mu = 2.5$

Figure 6 shows a basin of attraction for $\alpha = 21.5$. The Basin on the left hand side contains three attractors, one chaotic(black) and two periodic(purple and turquoise). The turquoise one has fractal properties and the chaotic one appears as closed curve in the Statespace with its typical stretching and folding properties. In Figure 7 the chaotic attractor dissapears and instead two stable riddled attractors emerge. Therefore there are four coexisting periodic attractors.

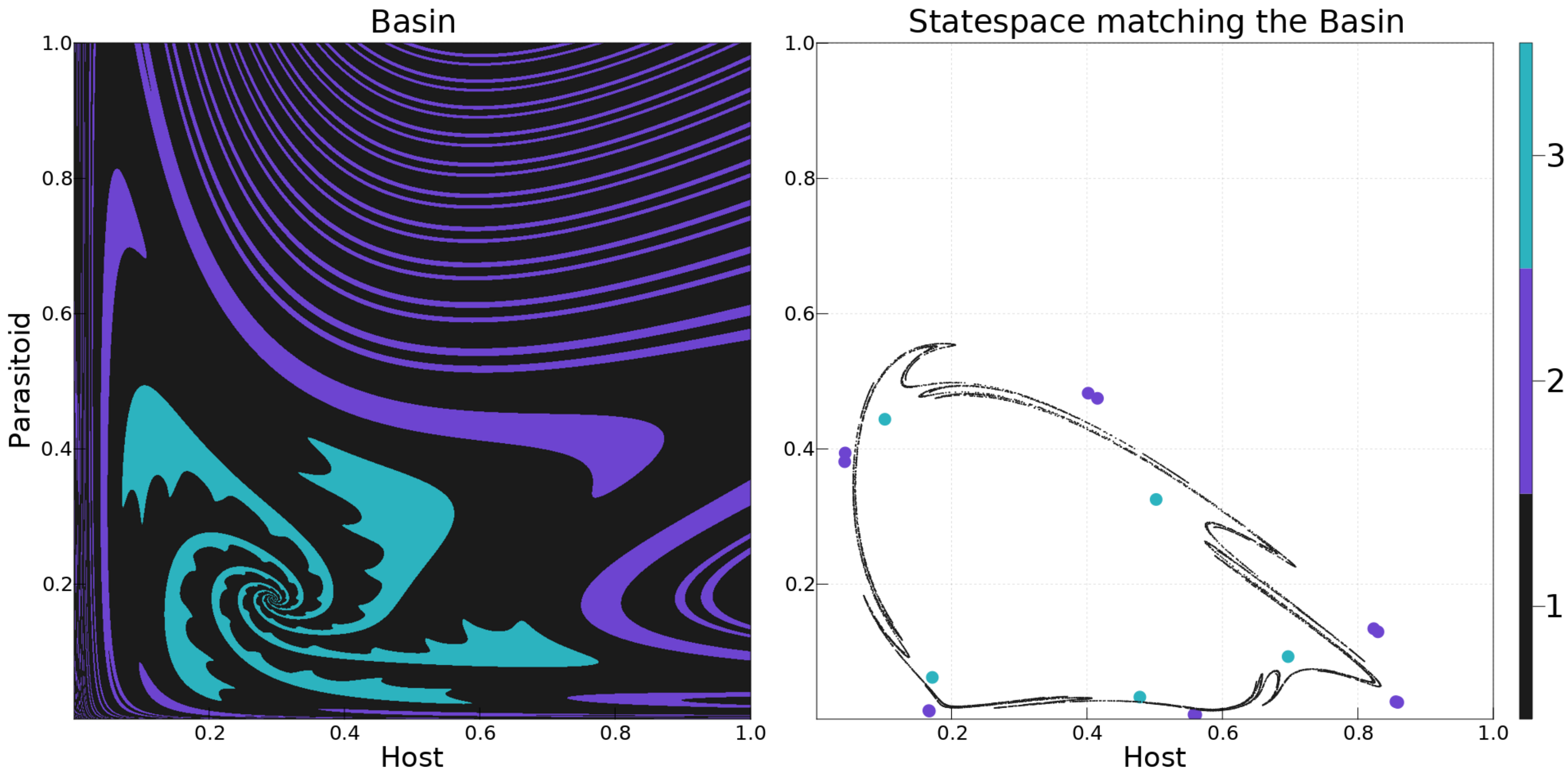

Figure 6: Basin of attraction: $\alpha = 21.5$

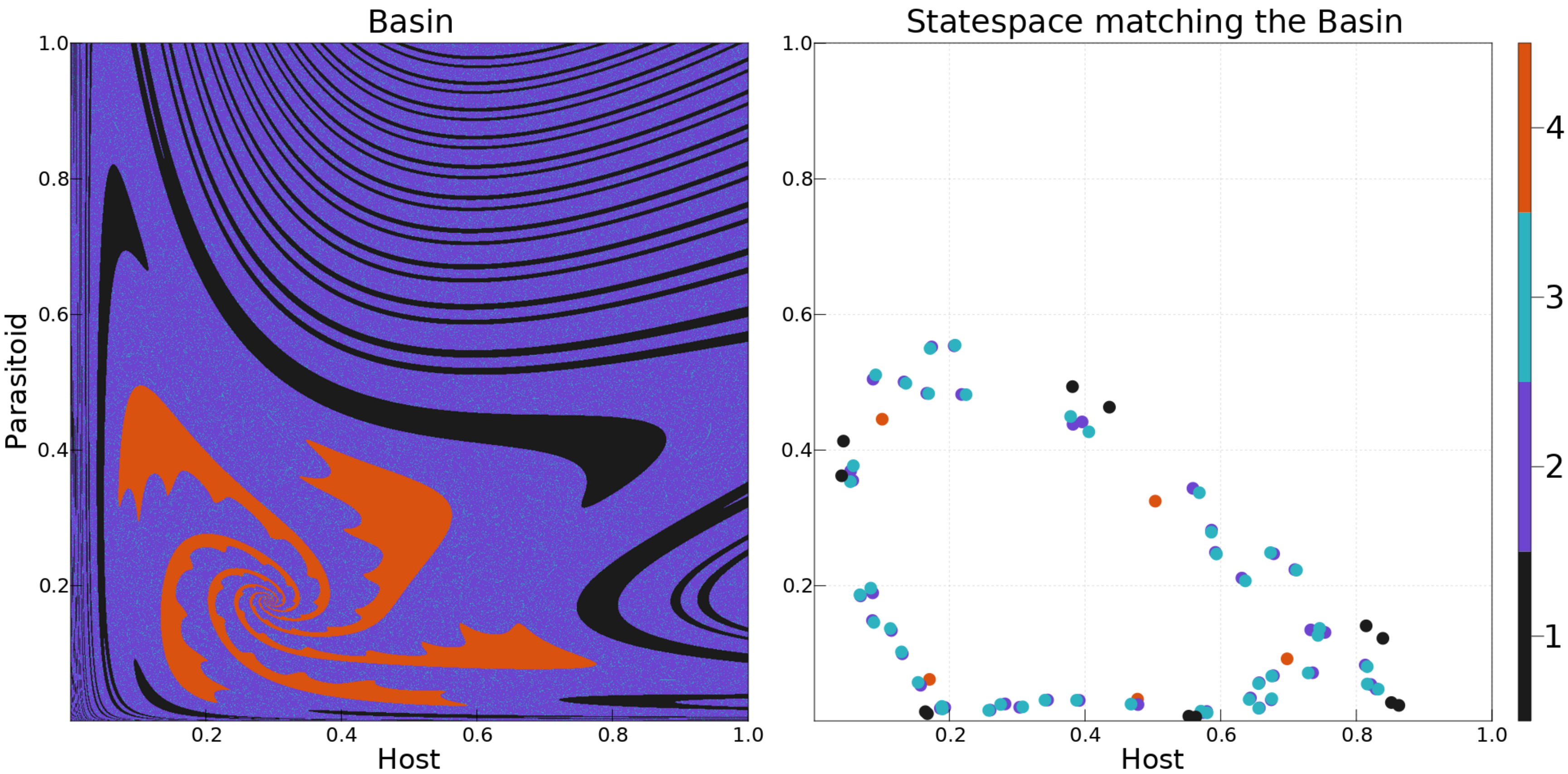

Figure 7: Basin of attraction: $\alpha = 21.6$

The results have shown that even when using the Beverton-Holt map, the behavior can be very complicated and chaotic.

4 Diskursion

Mathematically, our results show the presence of several coexisting attractors. Coexisting in the sense of arising from the same combination of parameters, but differing from each other in the initial conditions.

In reality, however, a population cannot have two different initial conditions and therefore only one attractor can be achieved. Since the model is deterministic, each of the possible attractors is stable but sensitive to population fluctuations. This means that our system is multi-stable. Thus, random changes or external influences such as immigration or emigration can change the system behaviour by switching between attractors.

It can be assumed, that the attractors have different sensitivities to fluctuations in phase space. Since the basin plot assigns an attractor to each state, and the attractor is also only a set of states, a metric can be derived from this, which determines the sensitivity of an attractor based on the neighborhood relationships.

Overall the Algorithms provided by the `DynamicalSystems` libary proofed to be very usefull and efficient. Without these it would not be possible to investigate the more complex behavior. The Julia programming language ist easy to understand and well documented and therefore not only fast in computation, but also fast to implement. The libary contains many more usefull functionalities like in the ChaosTools[1] module.

References

Datseris, George and Alexandre Wagemakers (Feb. 2022). "Effortless estimation of basins of attraction". In: *Chaos: An Interdisciplinary Journal of Nonlinear Science* 32, p. 023104. DOI: 10.1063/5.0076568.

Eggleton, P. and Robert Belshaw (Jan. 1997). "Insect parasitoids: an evolutionary overview". In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 337.1279. Publisher: Royal Society, pp. 1–20. DOI: 10.1098/rstb.1992.0079.

Hassell, Michael (June 2000). *The Spatial and Temporal Dynamics of Host-Parasitoid Interactions*. Oxford Series in Ecology and Evolution. Oxford, New York: Oxford University Press. ISBN: 978-0-19-854088-5.

Kaitala, Veijo, Janica Ylikarjula, and Mikko Heino (Apr. 1999). "Dynamic Complexities in Host–Parasitoid Interaction". In: *Journal of Theoretical Biology* 197.3, pp. 331–341. ISSN: 0022-5193. DOI: 10.1006/jtbi.1998.0878.

El-Kholy, O.A. et al. (1992). *Global Biodiversity: status of the Earth's living resources.* ENG. Publication Title: Biodiversity Heritage Library. [s.n.] DOI: 10.5962/bhl.title.44956.

[1]https://juliadynamics.github.io/ChaosTools.jl/dev/

Nusse, Helena E and James A Yorke (2012). *Dynamics: numerical explorations: accompanying computer program dynamics*. Vol. 101. Springer.

Ross, Chip and Jody Sorensen (2000). "Will the Real Bifurcation Diagram Please Stand Up!" In: *The College Mathematics Journal* 31.1, pp. 2–14. DOI: 10.1080/07468342.2000.11974102.