Dynamic Systems

The mathematics behind the "Simulator" computer program

The computer program "Simulator", the use of which is described in a separate manual, enables the simulation of simple dynamic systems and experimentation with them. The code is publicly available on GitHub, written in VB.NET, provided with detailed comments and can be extended as required. This requires the free community version of Microsoft Visual Studio. This document describes the mathematical basics for the "Simulator" at an elementary level.

Version 4.0 – 2024/04/01

Contents

Introduction ... 3

1. Historical Background ... 4
2. Dynamic Systems ... 5
   - 2.1. Basics ... 5
   - 2.2. Orbit and periodic Points ... 6
   - 2.3. Attractors and Repellors ... 6
   - 2.4. Chaotic dynamic Systems ... 8
   - 2.5. Histograms ... 9
   - 2.6. Protocols ... 10
3. Examples of discrete dynamic Systems ... 11
   - 3.1. The Bernoulli shift System ... 11
   - 3.2. The Tent Map ... 12
   - 3.3. Logistic Growth ... 19
   - 3.4. Conjugates of the Tent Map ... 22
   - 3.5. The "normalized" Parabola ... 26
   - 3.6. Implementation in the "Simulator" ... 28
   - 3.7. Histograms in the Chaotic Case ... 30
   - 3.8. Two-dimensional Representation and Transitivity in the chaotic Case ... 31
   - 3.9. Unimodal functions ... 32
   - 3.10. Period Doubling ... 33
   - 3.11. Implementation in the "Simulator" ... 39
   - 3.12. Exercise Examples ... 40
4. Mathematical Billiards ... 42
   - 4.1. Introduction ... 42
   - 4.2. Basics of elliptical Billiards ... 43
   - 4.3. Implementation of elliptical Billiards in the "Simulator" ... 48

4.4. Billiards in a Circle 50
4.5. Periodic Points in elliptical Billiards 54
4.6. Caustics in elliptical Billiards 57
4.7. Billiards in the Stadium 60
4.8. Implementation in the "Simulator" 65
4.9. Oval Billiards 65
4.10. Representation of the phase space 69
4.11. The convex Billiard Table 70
4.12. The C-diagram 74
4.13. Exercise Examples 81
5. Numerical methods for solving Ordinary Differential Equations 85
5.1. Ordinary differential equations 85
5.2. The classic Spring Pendulum 86
5.3. The Forward Euler Method 87
5.4. The Backward Euler Method 90
5.5. The implicit Midpoint Rule 91
5.6. The fourth-order Runge-Kutta Method 93
5.7. Implementation in the "Simulator" 95
5.8. Exercise examples 97
6. Coupled pendulums 99
6.1. The Lagrange formalism 99
6.2. The double pendulum 104
6.3. Implementation of the double pendulum 107
6.4. Investigation of the phase space for the double pendulum 110
6.5. Oscillating spring pendulum 113
6.6. Implementation of the oscillating spring pendulum 116
6.7. Shaked pendulum 119
6.8. Exercise examples 122
7. Implementation of own variants 123
Bibliography 125

Introduction

There are many programs on the Internet that enable the simulation of simple dynamic systems. However, their code is hardly public, and the underlying mathematics is also poorly documented. The fourth version of the "Simulator" enables the iteration of unimodal functions, the simulation of mathematical billiards, the investigation of numerical methods for solving ordinary differential equations and the simulation of different linked pendulums. The code of the program is written in VB.NET and is publicly available on Github in the repository "HermannBiner/Simulator". To work with it, the community version 2022 of Microsoft Visual Studio is sufficient, which can be downloaded free of charge and easily installed.

The mathematics on which the "Simulator" is based is explained at an elementary level in this document. Examples of exercises or suggestions for extending the "Simulator" are intended to encourage students to do their own work. Mathematics at high school is more than full of material. Nevertheless, to interested student's further topics as part of an optional subject or a seminar may be offered. The "Simulator" and this document are intended to contribute to this.

The following systems are implemented in version 4.0 of the program:

- Growth models and iteration of unimodal functions such as logistic growth, tent map, iteration on the parabola including related topics such as the Feigenbaum diagram.
- Mathematical billiards with various billiard table shapes: elliptical billiards, stadium billiards, oval billiards. The analogue to the Feigenbaum diagram is here the C-diagram.
- The investigation of numerical methods for solving ordinary differential equations. Some simple methods are compared here using the example of the spring pendulum.
- The simulation of coupled pendulums: Double Pendulum, oscillating Spring Pendulum, horizontally Shaked Pendulum

The use of the "Simulator" is documented in detail in a manual in German and a manual in English. The user interface language can be selected between German and English. The code contains detailed comments in English. This document with the mathematical comments is available in German and English.

The only prerequisite is mathematics which is either covered at high school or which can be made accessible to student with little effort. Topics from geometry (conic sections and plane vector geometry), analysis (continuity, calculus, and ordinary differential equations) and physics (Lagrange formalism) are covered.

The individual subject areas are largely independent, so that a selection is possible depending on interest and the time available. Apart from the first introductory chapter 2, the following can be selected in isolation.

The "Simulator" is to be expanded thematically in future versions and extended with further dynamic systems.

1. Historical Background

Classical mechanics as the study of moving bodies began in 1687, when Isaac Newton published his “Principia”. In 1736, Leonhard Euler formulated Newton's mechanics with greater mathematical precision and combined it with Leibniz's infinitesimal calculus. In 1788, Joseph-Louis Lagrange introduced the Lagrange function in the field of classical mechanics, which is based on the principle of least action and can be used to formulate the equations of motion of any mechanical system. In 1796, Pierre-Simon Laplace brought celestial mechanics to a level that allowed the until then unexplained orbital perturbations of Uranus to be traced back to the then undiscovered planet Neptune.

Impressed by these great advances in mechanics over the course of around a hundred years, Laplace formulated a statement in 1814 which says, in essence, that a demon endowed with sufficient intelligence could predict the entire future of the world based on the laws of mechanics and with sufficiently precise knowledge of the present state of the world. The future and the past would lie clearly before the eyes of this intelligence. This hypothetical intelligence later became known as "Laplace's demon".

Doubts about this view later arose in various places:

- While studying the three-body problem (how do three celestial bodies move around their common centre of gravity?), Henri Poincaré suspected around 1888 that this system reacts so sensitively to the initial conditions that its behaviour can only be predicted in the long term with ever greater effort. A closed mathematical solution (such as the elliptical orbits in the two-body problem) was not possible.
- According to Albert Einstein's theory of relativity (1905), it is not possible to grasp the entire cosmos. Due to the finite speed of light, there is a horizon of knowledge beyond which the "demon" could not see.
- Werner Heisenberg formulated the so-called uncertainty principle in a paper published in 1927. It essentially states that no deterministic statements are possible at quantum level, only statements of probability.
- Chaos theory, which began at the start of the last century with Henri Poincaré and his study of the three-body problem, also discovered phenomena that present the "demon" with unsolvable problems. In the 1960s, the mathematician and meteorologist Edward Lorenz noticed during computer simulations of weather patterns that minimal disturbances to the system can have dramatic effects on its future development. In 1972, he coined the image of the butterfly whose flapping wings can trigger a hurricane.
- Since around the middle of the last century, dynamic systems have therefore become the focus of mathematical research. The interesting thing is that certain phenomena can already be dealt with at secondary school level. In the 1990s, several articles on this topic were published in the "ETH für die Schule" program. See bibliography.

2. Dynamic Systems

2.1. Basics

Before we turn to the examples implemented in the "Simulator", it is necessary to introduce a few basic terms.

Dynamic systems are systems (often of a physical nature) whose state changes over time, but which obey a *law of motion.* Examples include planetary motion, the double pendulum, but also models of weather patterns. These examples are *continuous* dynamic systems.

It is possible that the change of state occurs through the *iteration of a mapping.* This is referred to as a *discrete* dynamic system. One example is the well-known *3n+1 problem (also known as the Collatz problem)*.

In the 3n+1 problem, the iteration of the "3n+1 mapping" is considered:

$$f(n) = \begin{cases} \frac{n}{2}, n\ even \\ \frac{3n+1}{2}, n\ odd \end{cases} \quad , n \in \mathbb{N}$$

The open question is whether the sequence approaches the number 1 regardless of the starting point and thus the cycle 1, - 2, -1, -2, ...

In general, one can define:

A *dynamic system* is a triple (T, X, f) . The quantity T is the *period*. Typically, $T = \mathbb{N}_0\ oder\ \mathbb{Z}$ for discrete dynamical systems and $T = \mathbb{R}_0^+\ oder\ \mathbb{R}$ for continuous dynamic systems. The set X is the *state space* and *f* is a mapping: $f: X \longrightarrow X$.

In *discrete* dynamic systems, a state changes to the nth step:

$$x_{n+1} = f(x_n), x_n \in X, n \in T = \mathbb{N}_0\ or\ \mathbb{Z}$$

In most cases X is a real interval or a subset of the $\mathbb{R}^n$.

In *continuous* dynamic systems, the "next" state of the system depends on the current state and the "current tendency" at this point. The latter is usually described by a system of differential equations. These are interesting if they are not integrable. An attempt is then made to approximate the behaviour of the system step by step using numerical methods. This "stroboscopic approach" transforms a continuous dynamic system into a corresponding discrete dynamic system.

We will restrict ourselves here to discrete dynamic systems.

The basic question in these studies is the *forecast of the system's future*.

Idea: If the system is precisely described by a law of motion, the global future of the system depends only on the initial conditions. This is determinism in the strict sense of Laplace's demon.

However, since the initial conditions cannot be measured with infinite precision, it is hoped that this inaccuracy will not have a major impact. Depending on this, you have a:

"Brave system": The strong principle of cause and effect applies: *similar causes have similar effects*.

"Chaotic system": The weak principle of cause and effect applies: *only exactly the same causes have the same effects*. Similar causes can have dramatically different effects!

A comprehensive introduction to the behaviour of dynamic systems can be found in [1]. [2] offers a good introduction at high school level and contains examples of exercises. We will recapitulate the most important terms below.

2.2. Orbit and periodic Points

We consider a dynamic system (T, X, f). If we choose a starting point $x_0 \in X$, an orbit of this point is created during the movement. This is called the *orbit* of x_0. For a discrete dynamic system, this is:

$$Or^+(x_0) := \{f^n(x_0) \in X, n \in \mathbb{N}_0\}$$

Conversely, you can consider the set of all points that fall on x_0 during the iteration. This set is called the inverse orbit of x_0. For a discrete dynamic system, this is:

$$Or^-(x_0) := \{x \in X \,/\exists n \in \mathbb{N}\ with\ f^n(x) = x_0\}$$

Example

With the "3n+1 mapping" is:

$$Or^+(7) = \{7, 11, 17, 26, 13, 20, 10, 5, 8, 4, 2, 1\}$$

One open question is: $Or^-(1) = \mathbb{N}$?

It is possible that for a point ξ holds: $f(\xi) = \xi$. Then ξ is called a *fix point* of the iteration. Similarly, a ξ is called a *periodic point of order k or k-periodic point of f*, if this point is mapped back onto itself after at least k iteration steps. The following applies to such a point:

$$f^k(\xi) = \xi, k \in \mathbb{N}\ and\ f^i(\xi) \neq \xi\ for\ i \in \mathbb{N}\ and\ 0 < i < k$$

A k-periodic point of f is a fixed point of f^k.

If ξ is a k-periodic point of f then the set

$$\{\xi, f(\xi), f^2(\xi), \dots, f^{k-1}(\xi)\} =: \{\xi_1, \xi_2, \dots, \xi_k\}$$

is called *k-cycle* of the iteration of f. Each point in this cycle is then a k-periodic point:

$$f^k(\xi_i) = f^k\left(f^{i-1}(\xi_1)\right) = f^{k+i-1}(\xi_1) = f^{i-1}\left(f^k(\xi_1)\right) = f^{i-1}(\xi_1) = \xi_i, for\ i \in \mathbb{N}\ and\ 1 \leq i \leq k$$

Example

In the "3n+1 mapping": {2, 1} is a 2-cycle.

2.3. Attractors and Repellors

What is interesting about a dynamic system is the *asymptotic* behaviour of the iteration starting from a starting point $x_0 \in X$. In a discrete dynamic system, this is the behaviour of the iteration when the number of iteration steps tends to infinity.

It is possible that for a fixed point $\xi \in X$ and a start value $x_0 \in X$ holds:

$$\lim_{n\to\infty} f^n(x_0) = \xi$$

Then x_0 *is asymptotically periodic* (with period 1) and analogous:

x_0 is called *asymptotically periodic with period k* if the orbit of x_0 converges to a k-cycle. This means that the orbit breaks down into convergent subsequences, each of which converges towards a point of the cycle. Or also that x_0 is asymptotically periodic with period 1 under the iteration f^k is.

Decisive for the behaviour of the iteration near a fixed point (or k-periodic point) is the value of the derivative of *f* at this point, if *f* is continuously differentiable.

Be $I \subset \mathbb{R}$ a real interval, ξ a fixed point of *f* and $f: I \longrightarrow I$ continuously differentiable. Then ξ *is called an attractive fixed point* or *attractor* for short, if: $|f'(\xi)| < 1$. The value $\lambda \coloneqq f'(x)$ is called the *multiplier* of the fixed point ξ. If even holds $|f'(\xi)| = 0$, then ξ is called *superattractive.*

If a point gets "too close" to an attractor during the iteration ξ, it can no longer "escape" from its surroundings, but will asymptotically move towards this attractor.

Reason: Be ξ be an attractor. For reasons of continuity, the following applies to an open environment

$$U(\xi) \subset I: |f'(x)| < 1 \; for \; x \in U(\xi)$$

If a point falls into this environment during the iteration, i.e. for a starting value x_0 a $n \in \mathbb{N}$ exists with $x_n = f^n(x_0) \in U(\xi)$, then according to the mean value theorem, there is a $\vartheta \in U(\xi)$ with

$$|x_{n+1} - \xi| = |f(x_n) - f(\xi)| = |f'(\vartheta)||x_n - \xi| < L|x_n - \xi| \; for \; a \; certain \; L < 1$$

The following then applies for the further iteration:

$$|x_{n+k} - \xi| < L^k |x_n - \xi| \longrightarrow 0 \; für \; k \longrightarrow \infty$$

And the sequence $(x_{n+k})_k$ strives towards the fixed point ξ.

In connection with an attractor ξ one speaks of the *basin* of the attractor. This consists of all points that converge asymptotically to the attractor:

$$B(\xi) \coloneqq \{x \in X / \lim_{n \to \infty} f^n(x) = \xi\}$$

Analogously, one can speak of the basin of an attractive cycle.

Be $I \subset \mathbb{R}$ a real interval, ξ a fixed point of *f* and $f: I \longrightarrow I$ continuously differentiable. Then ξ *is called a repulsive fixed point* or *repellor* if for its multiplier holds: $|\lambda| = |f'(\xi)| > 1$.

If a point comes "close" to a repellor ξ during the iteration, it will move away from it again in the next iteration steps, at least until it has "escaped" from the $U(\xi) with \; |f'(x)| > 1 \; for \; x \in U(\xi)$. It may enter this environment again later but will then be catapulted out again. If it does not hit the fixed point ξ exactly, it will never reach it asymptotically.

If ξ is a fixed point with multiplier $|\lambda| = |f'(\xi)| = 1$, then ξ is called *indifferent.*

Example

Look at the map: $f(x) = 2x \; mod \; 1, [0,1] \to [0,1]$

It is also known as the *Bernoulli shift system.*

If you write x as a dual fraction, you can see the effect of the mapping. Let $s_i \in \{0,1\}$ is the i-th digit of x in dual fraction notation. Then the following holds:

$$f: 0, s_1 s_2 s_3 \ldots \to 0, s_2 s_3 \ldots$$

0 is a fixed point of this mapping. Each periodic dual fraction is a cycle with its period length. Every pre-periodic dual fraction is a pre-periodic cycle. All cycles are repulsive, because

$f'(x) = 2 > 1\ on\ [0,1[$. The Bernoulli shift system will be discussed in detail later.

With a cycle, the question arises as to whether a cycle can consist of both attractors and repellors at the same time. Let us consider a cycle

$$\{\xi, f(\xi), f^2(\xi), \dots, f^{k-1}(\xi)\}$$

Then each point of the cycle is a fixed point of f^k. Whether a point $\xi_i := f^i(\xi_1). 1 \leq i < k$ is attractive or not depends on the derivative of f^k at this point, since ξ_i is a fixed point of f^k. According to the chain rule it is:

$$[f^k(\xi_i)]' = f'\left(f^{k-1}(\xi_i)\right) \cdot f'\left(f^{k-2}(\xi_i)\right) \cdots f'(\xi_i)$$

However, the function arguments on the right-hand side are currently going through the entire cycle. If you rearrange, the following applies:

$$\{f^{k-1}(\xi_i), f^{k-2}(\xi_i), f^{k-3}(\xi_i), \dots, \xi_i\} = \{\xi_1, f(\xi_1), f^2(\xi_1), \dots, f^{k-1}(\xi_1)\} = \{\xi_1, \xi_2, \dots, \xi_k\}$$

Thus the derivation

$$[f^k(\xi_i)]' = f'^{(\xi_k)} \cdot f'(\xi_{k-1}) \cdots f'(\xi_1) = \prod_{i=1}^{k} f'(\xi_i)$$

is independent of *i* and assumes the same value for all points in the cycle. All points in the cycle therefore have *the same* multiplier and are all either *attractive* or *repulsive*.

2.4. Chaotic dynamic Systems

Some of the examples implemented in the "Simulator" exhibit so-called chaotic behaviour. That is why we also want to deal with this. Further explanations and various definitions of chaotic systems can be found in [1], [2] or [3].

We will use the following definition in this document, which goes back to Robert L. Devaney (1989):

A dynamic system is called *chaotic* if it fulfils the following properties:

1) It is *sensitive* to the initial conditions. This means that two starting values that are arbitrarily close to each other $x_1, x_2 \in X$ move arbitrarily far apart during the iteration in the state space *X*.

2) It is *transitive*. This means that if any start value $x_0 \in X$ and an arbitrary target value $z \in X$ are given, then there is an alternative start value in an arbitrarily small vicinity of x_0 that comes arbitrarily close to the target value *z* during the movement.

3) The periodic orbits lie *dense* in the state space.

Sensitivity is what the mathematician and meteorologist Edward Lorenz observed in 1963. He set up a simple system of differential equations to describe the weather. With the help of a computer, he wanted to create a weather forecast. As the computer required a lot of computing time, Lorenz

rounded the initial conditions to fewer calculation points. This rounding led to a completely different weather forecast. He then formulated the so-called *butterfly effect,* which says that *the flap of a butterfly's wings in Brazil can trigger a tornado in Texas*.

Transitivity has a dramatic effect on an experimenter. Measuring instruments have a certain accuracy. Let us assume that all values within a very small environment $U_\delta(x) \subset X$ are displayed as *the same* measured value during the measurement. Then for *each* "target point" there is $z \in X$ a starting point $x_0 \in U_\delta(x)$ which during the iteration falls into an environment $U_\varepsilon(z)$ and is displayed as *z* by the measurement if $\varepsilon < \delta$. In this experiment, every start value that is below the measurement accuracy of the experimenter will therefore appear as *the same* start value. However, it can end up at any specified target value. This system appears completely random to the experimenter and there is no recognizable causality. The experiment can produce a completely different result each time it is repeated, even though the dynamic system is based on a precise law of motion. In this context, one also speaks of pseudo-randomness or deterministic chaos. The periodic orbits are all repulsive, otherwise the sensitivity and transitivity would not be given for the whole of *X.* If these lie close together in *X*, then all aperiodic points are constantly "catapulted away" because they constantly come arbitrarily close to a repellor.

In a chaotic system, the orbit of a starting point has an infinite number of accumulation points and is distributed randomly across the state space. A so-called *strange attractor is* obtained. In the aperiodic case, there is no long-term relationship between past and future and the system can only be predicted by letting it run.

The computer program "Simulator" supports various iteration functions, which become chaotic for certain parameter values. The properties of transitivity and sensitivity can then be examined and visualized in concrete terms.

There is also a "Two-dimensional representation" mode. In this mode, measurements of an experimenter are simulated which, in the chaotic case, no longer allow any conclusions to be drawn about the underlying law of motion.

2.5. Histograms

To examine an orbit in the chaotic case, the state space can be divided into disjoint intervals, the union of which results in the state space again. You then count how often an interval is hit during the iteration. This produces a histogram and shows how an orbit is "distributed" in the state space.

For a state space that is a real interval $[a, b]$ the "Simulator" breaks it down into n intervals of equal width:

$$d := \frac{b-a}{n}$$

Then these intervals are:

$$[a, a+d[\cup [a+d, a+2d[\cup \ldots \cup [a+(n-1)d, b]$$

The "Simulator" then determines how often an interval is hit by a point in the iteration and displays the histogram.

2.6. Protocols

Suppose you have two intervals $U, V \subset X, U \cap V = \emptyset$ and there are paths (x_i)which run completely in $U \cup V$ run. A protocol is created for such a path as follows:

$$p(x_i) = \begin{cases} 0, x_i \in U \\ 1, x_i \in \mathrm{V} \end{cases}$$

Example

For the 3n+1 problem, the following protocol *p* could be created:

$$p(n) = \begin{cases} 1, n\ odd \\ 0, n\ even \end{cases}$$

As we will see, there are examples of iterations where *every* 0-1 sequence occurs as a log if the start value is chosen appropriately. This gives us another definition of "chaotic":

A dynamic system is called *chaotic in the sense of the coin toss* if there are two sets U and V as described at the beginning, so that every 0-1 sequence occurs as a protocol of a path if the starting value is chosen accordingly.

Be $p = s_1 s_2 s_3 \dots s_n, s_i \in \{0,1\}$ a protocol for the first n steps of an iteration.

In this context, it is interesting to examine the following quantity:

$$A_p \coloneqq \{x_o \in X \mid The\ protocol\ of\ x_o starts\ with\ p\}$$

The following applies:

$$A_{s_0} \supset A_{s_0 s_1} \supset A_{s_0 s_1 s_2} \supset \cdots$$

In [2] it is shown that the above sequence is an interval nesting if for *f* and the sets U, V holds:

1) $f(U) \supset U \cup V$ and also $f(V) \supset U \cup V$
2) IT gives a number $\lambda > 1$ so that $|f'(x)| \geq \lambda, \forall x \in U \cup V$

Under conditions 1) and 2), any protocol can be specified. The above interval nesting then defines the start value in $U \cup V$that belongs to the specified protocol.

In the chaotic case, the "Simulator" allows any protocol to be specified (naturally only up to a limited length due to the accuracy of the calculation). It then proposes a start value for the iteration, the orbit of which delivers exactly the specified protocol.

A connection between *"chaotic in the sense of the coin toss"* and the definition in the sense of Devaney is established in [2]:

Assume that a dynamic system is chaotic in the sense of the coin toss. Now consider the set Σ of all 0-1 sequences and the set of associated starting points $\Lambda \coloneqq \{x(s) | s \in \Sigma\}$. Then, the subsystem (Λ, f) is chaotic (in the sense of Devaney).

3. Examples of discrete dynamic Systems

Preliminary remark: The tent map in section 2 is referred to repeatedly. It should not be ignored. The other sections in this chapter can be considered independently of each other.

3.1. The Bernoulli shift System

The following discrete dynamic system is referred to as a *Bernoulli shift system:*

$$f: [0,1] \longrightarrow [0,1]; f(x) = \begin{cases} 2x, x \in [0,0.5[\\ 2x-1, x \in [0.5,1] \end{cases}$$

This simple system is chaotic.

Each half of the interval [0,1] is first stretched by a factor of 2. Then both stretched halves are pushed over each other. This gives a "good mixing" of the points in the interval. The chaotic properties can be recognized by looking at *x* in dual fraction representation and then examining the effect of the mapping. Let $x_1 \in [0,1]$ a starting point of the mapping, represented as a dual fraction:

$$x_1 = 0.s_1s_2s_3 \ldots;\ s_i \in \{0,1\}$$

Then this dual fraction is shifted one place to the left at each iteration step by multiplication by 2, whereby the place before the decimal point disappears due to the assignment $x \longmapsto 2x - 1$ in the case $x \geq 0.5$. One obtains as orbit:

$$x_2 = 0.s_2s_3s_4 \ldots$$

$$x_3 = 0.s_3s_4 \ldots$$

Example:

$$x_1 = 0.010010111$$

$$x_2 = 0.10010111$$

$$x_3 = 0.0010111$$

$$x_4 = 0.010111$$

$$x_5 = 0.10111$$

...

Sensitivity and transitivity can be easily demonstrated:

Sensitivity

Select for any initial value

$$x_1 = 0, s_1s_2s_3 \ldots s_is_{i+1}s_{i+2} \ldots.$$

a second value, in which the first i digits correspond to x_1 and the other digits are complementary (i.e. "0" becomes "1" and vice versa):

$$x'_1 = 0, s_1s_2s_3 \ldots s_i\overline{s_{i+1}}\,\overline{s_{i+2}}\,\overline{s_{i+3}} \ldots$$

Then lies x'_1 lies in an arbitrarily small neighborhood of x_1:

$$x'_1 \in U_{2^{-i}}(x_1)$$

But after i steps, the distance between the two points becomes maximum in [0,1[.

Transitivity

Chose a starting point

$$x_1 = 0, s_1 s_2 s_3 \dots s_i \dots$$

And a destination point

$$z = 0, t_1 t_2 t_3 \dots t_i \dots$$

Define a new starting point:

$$x'_1 = 0, s_1 s_2 s_3 \dots s_i 000 \dots 000 t_1 t_2 t_3 \dots t_i \dots$$

x'_1 is in an arbitrarily small environment of x_1 depending on how many zeros are inserted.

And after i steps, x'_1 falls *exactly* on z.

Periodic points

Every periodic dual fraction provides a periodic orbit. These are exactly the rational numbers in the interval [0,1[and these lie dense in this interval. Every non-periodic dual fraction leads to an aperiodic orbit.

Furthermore, information about the start value is lost in bits with every step! This information disappears completely during the iteration.

Specified protocols

If we split the interval:

$$U = [0,0.5[, V = [0.5,1]$$

Then applies: $U \cap V = \emptyset$ and $U \cup V = [0,1]$

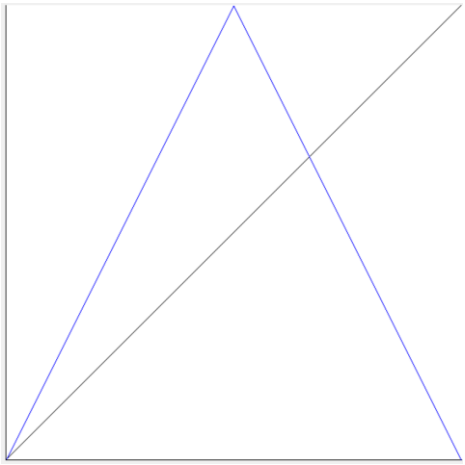
We create a protocol. We write a "0" if $x_i \in U$ and a "1" if $x_i \in V$. Then the protocol just matches the dual representation of the start value. There is therefore a bijective assignment between start values and protocols. There is a start value for any given protocol that provides this protocol!

3.2. The Tent Map

The *tent map is* a modification of the Bernoulli shift system. We consider the interval [0,1] and define the tent map as:

$$z\colon [0,1] \longrightarrow [0,1];\ z(u) = \begin{cases} 2u, u \in [0,0.5[\\ 2(1-u), u \in [0.5,1] \end{cases}$$

We have chosen the designations regarding the implementation in the "Simulator".



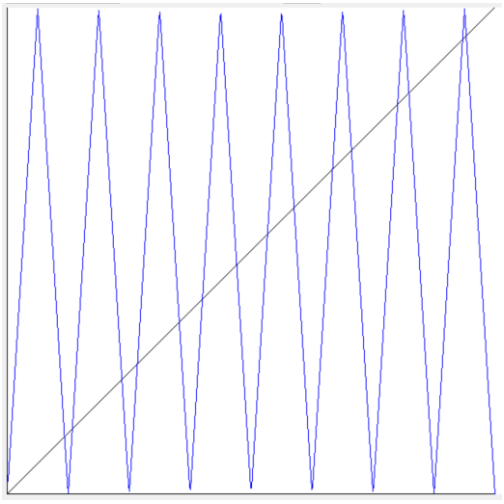
Graph of the tent map over the interval [0,1] created by the "Simulator"

The tent map intersects the 45° straight line at points 0 and 2/3. It therefore has the fixed points

$$\xi_1 = 0 \; and \; \xi_2 = 2/3$$

In these points $|f'(\xi_1)| = |f'(\xi_2)| = 2 > 1$ and both points are repulsive.

If you iterate the tent map, the unit interval is stretched and folded at each iteration. With four iterations, for example, you have the graph:



Graph of the fourfold iterated tent map f^4: $[0,1] \longrightarrow [0,1]$

The function shown above f^4 has a total of 8 fixed points (points of intersection with the 45° straight line) and all of them are repulsive (the derivative at all fixed points is > 1).

If you continue to iterate, the number of fixed points of the iterated function doubles with each iteration step. f^n has 2^n repulsive fixed points. These are repulsive n-cycles of the original function f and lie close together in the interval [0,1].

Some sentences can be elementarily proven with the tent map. We take this opportunity. However, these theorems are not relevant for understanding further chapters.

Theorem

For the tent map, every rational number from the unit interval is preperiodic or periodic. The associated cycles are repulsive. They are dense in the unit interval.

Proof: $x = 1$ goes over to the fixed point 0. A rational number from the unit interval that is less than 1 has the form $x = {}^{p}/_{q}$ with $1 \leq p < q$ and we can assume that the fraction is reduced.

If $x < 0.5, then\ 2p < q$, and *x* changes to ${}^{2p}/_{q}$. If $x \geq 0.5, then\ 2p \geq q$, and *x* transitions to $\frac{2q-2p}{q}$. If the denominator q contains the factor 2, it will be reduced and the factor 2 in the denominator will disappear during the iteration. $x = \frac{1}{2^k}$ goes to 0 during the iteration. If q contains factors other than 2, then x is always represented by a reduced fraction of the form $\frac{\cdot}{q'}$ during the iteration. Since there are only a finite number of numerators of the reduced fractions with denominator q', a fraction that has already occurred earlier must occur during the iteration. This is where the cycle begins.

The rational numbers are also dense in the unit interval. Since $|f'(x)| = 2\ in\ [0,1]\backslash\{0.5\}$ all cycles are repulsive. The point 0.5 merges into the repellor 0. □

Theorem

The tent map has for each $n \in \mathbb{N}$ a real cycle of length n.

Proof: Consider the starting value $x_1 = \frac{2}{2^n+1}$

This is multiplied by 2 at each step if the result is < 0.5. This is the case until the value

$$x_{n-1} = \frac{2^{n-1}}{2^n + 1}$$

As you can easily see, there is still $x_{n-1} < {}^{1}/_{2}$. In the next step you get:

$$x_n = \frac{2^n}{2^n + 1} > \frac{1}{2}$$

Therefore, according to the iteration rule

$$x_{n+1} = 2(1 - x_n) = 2\frac{1}{2^n + 1} = x_1$$

All occurring fractions cannot be reduced, so the cycle effectively has the length n. □

Theorem

The tent map is sensitive.

Proof: If we represent a number $x \in [0,1]$ as a dual fraction, then we see the effect of the tent map similarly to the Bernoulli shift.

The following applies in this representation and if $s_i \in \{0,1\}$ is the i-th dual digit after the decimal point:

$$f: \begin{cases} 0{,}0s_2s_3s_4 \ldots \longmapsto 0, s_2s_3s_4 \ldots \\ 0{,}1s_2s_3s_4 \ldots \longmapsto 0, \overline{s_2}\overline{s_3}\overline{s_4} \ldots \end{cases}$$

This means $\overline{s_i}$ the dual complement of s_i.

Now let an initial value be given in dual representation:

$$x_1 = 0.\, s_1s_2s_3s_4 \ldots.$$

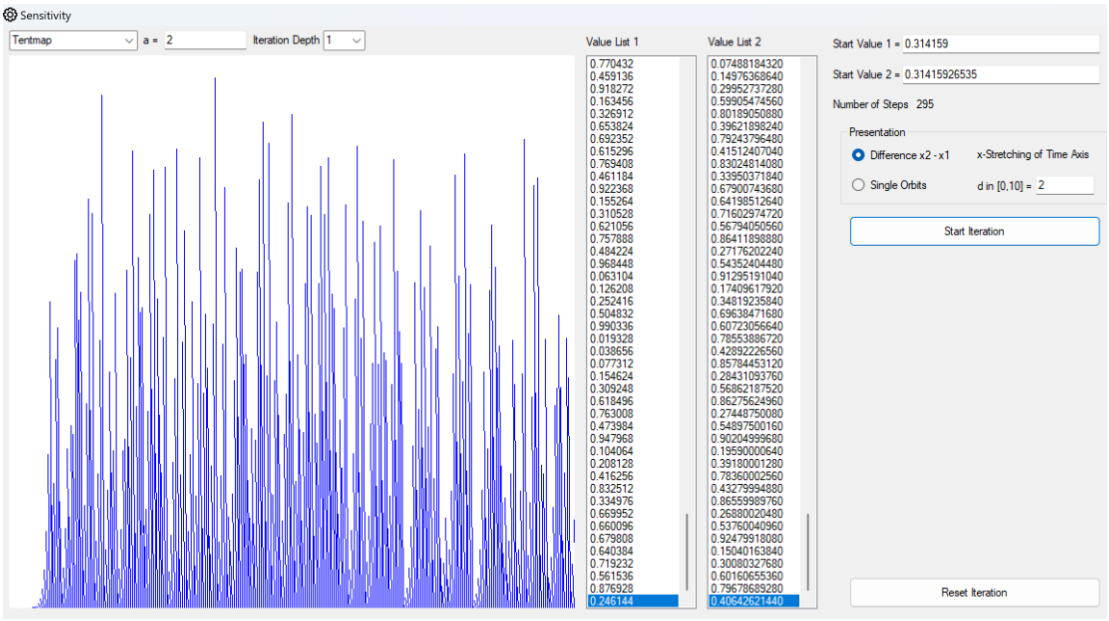
We then cut off this dual fraction after a sufficiently large number of digits (or add "0" to missing digits) so that the initial value changed in this way is close enough to the original. If it contains an odd

number "1", we select $x'_1 = 0.s_1s_2s_3s_4..00\ldots s_1s_2s_3s_4$ … In other words, we then append the digits from $x_1$to the end. With each iteration step, these digits move one position to the left, whereby the dual complement of the remaining digits is formed for each "1" occurring in the first position after the decimal point. As this occurs an odd number of times, the number will be there after the corresponding number of iteration steps: $x_n = 0.\bar{s}_1\bar{s}_2\bar{s}_3$ … , i.e. a number with a maximum distance to x_1.

We proceed in the same way if the truncated dual fraction of x_1 contains an even number "1", and set $x'_1 = 0.s_1s_2s_3s_4..00\ldots\bar{s}_1\bar{s}_2\bar{s}_3\bar{s}_4$ … with the same result. □

The "Simulator" uses this method to display the sensitivity of the marquee image in the menu item "Unimodal functions - Sensitivity". You can select two starting values that are very close to each other, and then either

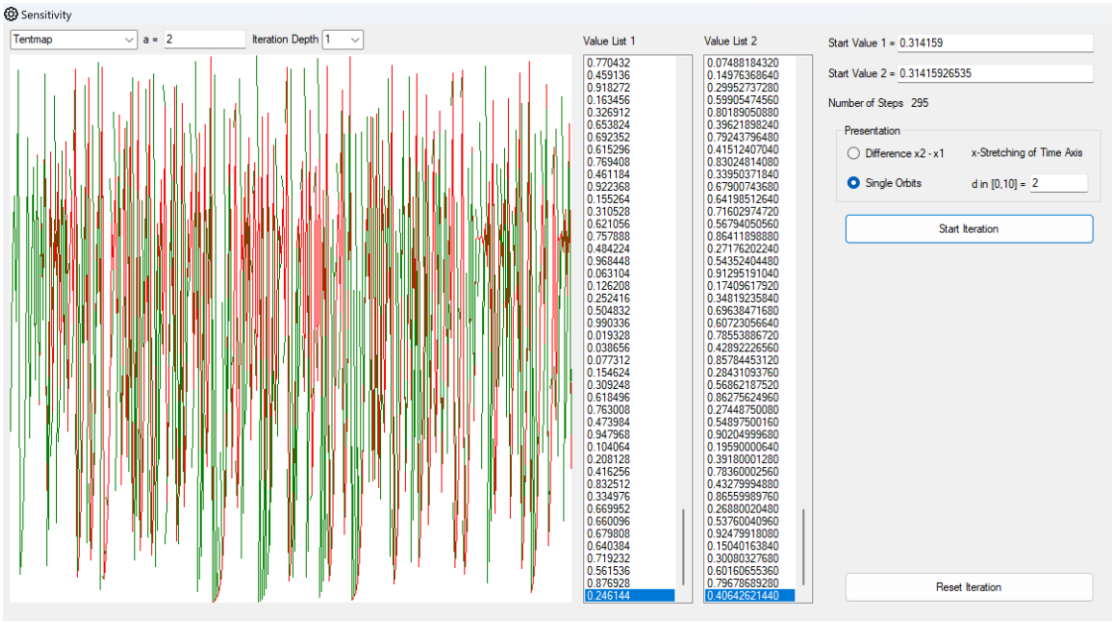
a) Display the difference between the two generated orbits during the iteration (on the "time axis" in the "Simulator"):



Difference between two orbits with starting values of the tent map that are close to each other

or

b) display each individual orbit on the time axis and compare the orbits



Different orbits of two starting values close to each other in the tent map

The two starting values only differ from the 7th digit after the decimal point, but after a few steps the orbits begin to diverge. At times, their difference reaches almost the entire interval width.

Theorem

The tent map is transitive.

Proof:

As with the detection of sensitivity, we assume a starting value $x_1 = 0.s_1s_2s_3s_4$ .... and a target value $z_1 = 0.t_1t_2t_3$ ... both in dual fraction representation. We cut off the dual fraction of x_1 again after a sufficient number of digits (or add "0" to missing digits) so that the modified start value is close enough to the original. If the remaining dual fraction contains an even number "1", we append the digits of the target value to the end. If the number "1" is odd, then we append the dual complement to the end. In both cases, after a sufficient number of iteration steps, we obtain $z_n = 0.t_1t_2t_3$ ... □

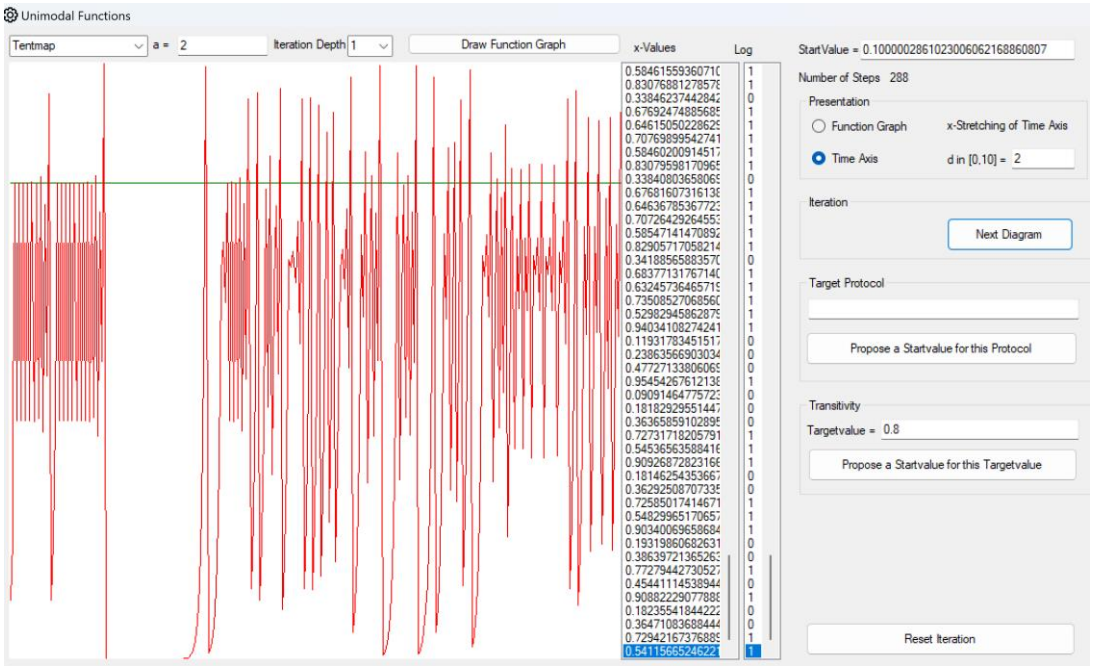
It is easy to see that the cycles are represented by periodic dual fractions and that these are dense in [0,1]. The following therefore applies:

Theorem

The tent map is chaotic.

To display the transitivity, the "Simulator" uses the same method as in the previous proof to suggest a slightly modified starting value that comes close to the specified target value. The modified dual fraction is then converted into a decimal fraction. Because of the finite number of digits and because all information about the initial value disappears over time, the iteration will again produce any values after the target value has been approached.

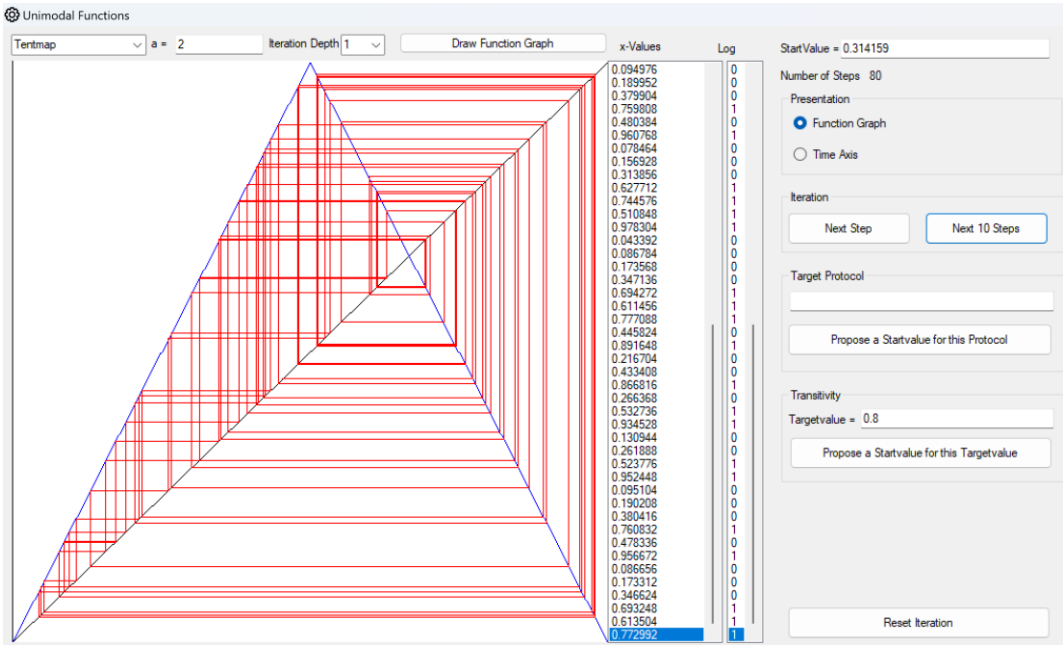
This function can be found in the menu item "Unimodal functions - Iteration".



The iteration is shown on the time axis. Green line: The target value

In the image above, the start value was set to 0.1 and the target value to 0.8. The "Simulator" has then calculated a new start value for this target value that is close to the original, namely 0.10000028610230060621688860807. The target value is very close after just 4 iteration steps.

Protocols



Iteration of the tent map shown in the function graph

In the figure above, the "Simulator" iterates the tent map with the first digits of π as the starting value. The iterated values are listed to the right of the graphs.

For the tent map, we define a protocol *p* according to the following rule:

$$p(x_n) = \begin{cases} 0, x_n \in [0,0.5[\\ 1, x_n \in [0.5,1] \end{cases}$$

This log is listed in the "log" column in the screen above.

As with the Bernoulli Shift, a start value can be found for each specified protocol that provides this protocol. To get from this specified protocol to the start value, the "Simulator" recalculates the protocol:

You start with the last digit in the protocol. Then the next algorithm is as follows:

If a "0" appears in the log, write a "0" after the decimal point and move all other digits one position to the right.

If a "1" appears in the protocol, then write a "1" after the decimal point, form the complement of all other digits and move them one position to the right.

Example

View the protocol $p = 10110$. If we work through this in 5 steps from right to left, we get a starting value with 5 digits after the decimal point.

We start with the last position "0". The fifth digit of the start value is therefore a "0".

$$x_1 = 0, s_1 s_2 s_3 s_4 0$$

The next position in the protocol is the "1". We therefore note a "1" in the fourth position in the starting value and form the dual complement of the digits to the right of it.

$$x_1 = 0, s_1 s_2 s_3 11$$

We continue with the next position in the protocol, a "1":

$$x_1 = 0, s_1 s_2 100$$

The "0" at position 2 in the protocol provides:

$$x_1 = 0, s_1 0100$$

And finally, the "1" in first place:

$$x_1 = 0{,}11011$$

You therefore obtain an initial value $x_1(s_1 s_2 s_3 s_4 \dots.)$ for a protocol *p=0.* $s_1 s_2 s_3 s_4$ … in dual representation. If the protocol has the length n, then all start values that provide the specified protocol are in the interval $[x_1(p), x_1(p) + 2^{-(n+1)}[$. To reduce the effects of computer rounding, we select the start value from the middle of the interval: $\tilde{x}_1(p) \coloneqq x_1(p) + 2^{-(n+2)}$.

You can access the function for the specified protocol in the "Unimodal functions - Iteration" menu.

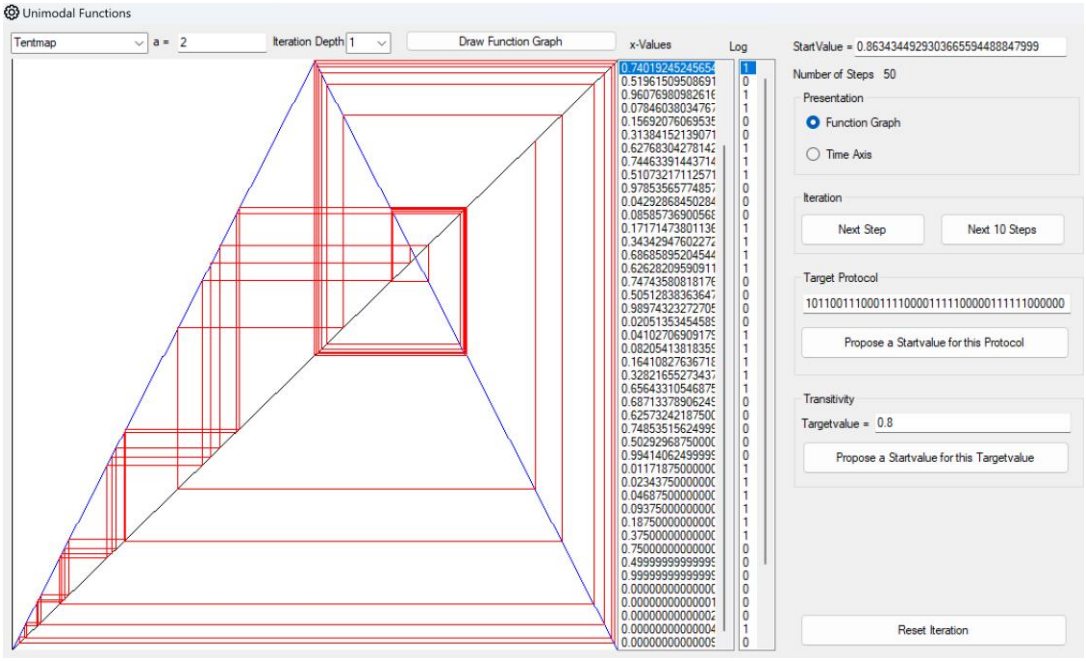
In the following example, we enter the target protocol:

101100111000111100001111100000111111000000

This target protocol has the length 42. The "Suggest start value" button is then used to suggest the start value as a decimal fraction, namely:

0.863434492930366559448884799 9

As you can see, the desired protocol is created during iteration:



Tent map with specified protocol of length 42

Tent map with a parameter a

A variant of the tent map with a parameter a is:

$$z_a: [0,1] \longrightarrow [0,1];\ z_a(u) = \begin{cases} au, u \in [0,0.5[\\ a(1-u), u \in [0.5,1] \end{cases}, a \in]0,2]$$

This has the fixed point $u = 0$ and this is attractive for $a < 1$.

For $u \geq 0.5$ can be used as a fixed point $u = \frac{a}{1+a}$ comes into question. For $\frac{a}{1+a} \geq 0.5$ it holds $a \geq 1$. The fixed point $u = \frac{a}{1+a}$ is therefore repulsive.

The following applies to the derivative of the n-fold iterated function: $|z_a^{n\prime}(u)| = a$. Thus, for $a > 1$ all potential n-cycles are repulsive.

There are $a > 1$ no attractive cycles and for $a < 1$ only the (attractive) fixed point 0.

For this reason, only the tent map defined at the beginning is usually considered with $a = 2$. However, the "Simulator" supports experimentation for $a \in]0,2]$.

3.3. Logistic Growth

We consider a simple system to simulate the growth of a population. The aim is not to design a realistic model for existing populations, but to show that even a very simplified model can have chaotic properties under certain conditions. In other words, the "chaos" is not due to the complexity of a model, but occurs even with very simple models, as we have also seen with the tent map.

We assume that the size of this population lies in the interval [0, 1]. 0 means that it is extinct and 1 means that it has reached the theoretically possible 100%. Now we denote the size of the population of the nth generation by $x_n \in [0,1], n \in \mathbb{N}$.

The next generation x_{n+1} becomes larger, the larger x_n was, due to the higher reproductive number. However, when the population reaches its natural resource limits, the population may become smaller again because not all individuals can survive. We describe this relationship using the following iteration rule:

$$x_{n+1} = f(x_n) \coloneqq ax_n(1 - x_n)\ where\ x_n \in [0,1], \forall n \in \mathbb{N}\ and\ a \in]0,4]$$

The parameter a can be understood as controlling the reproduction rate. This iteration rule is known as *logistic growth*.

We first want to investigate which values of *a* make sense. The case *a = 0* is trivial.

For $x_n \in [0,1]$, $x(1-x)$ is positive, so *a* must also be positive for the function value to be positive. On the other hand, the function *f* attains a maximum at $x = 0.5$ and it is:

$$f(0.5) = \frac{a}{4}$$

For the function value to lie in the interval [0,1], it must hold $a \leq 4$. This finally results in the condition $a \in]0,4]$. In future, we will write for the logistic growth:

$$f\colon [0,1] \longrightarrow [0,1], x \longmapsto ax(1-x), a \in]0,4]$$

We first examine which attractors exist.

Periodic points

ξ is a fixed point of the logistic growth if applies: $f(\xi) = \xi$ or $a\xi(1-\xi) = \xi$

This gives you two solutions: $\xi_1 = 0$ and $\xi_2 = 1 - {}^1/_a$

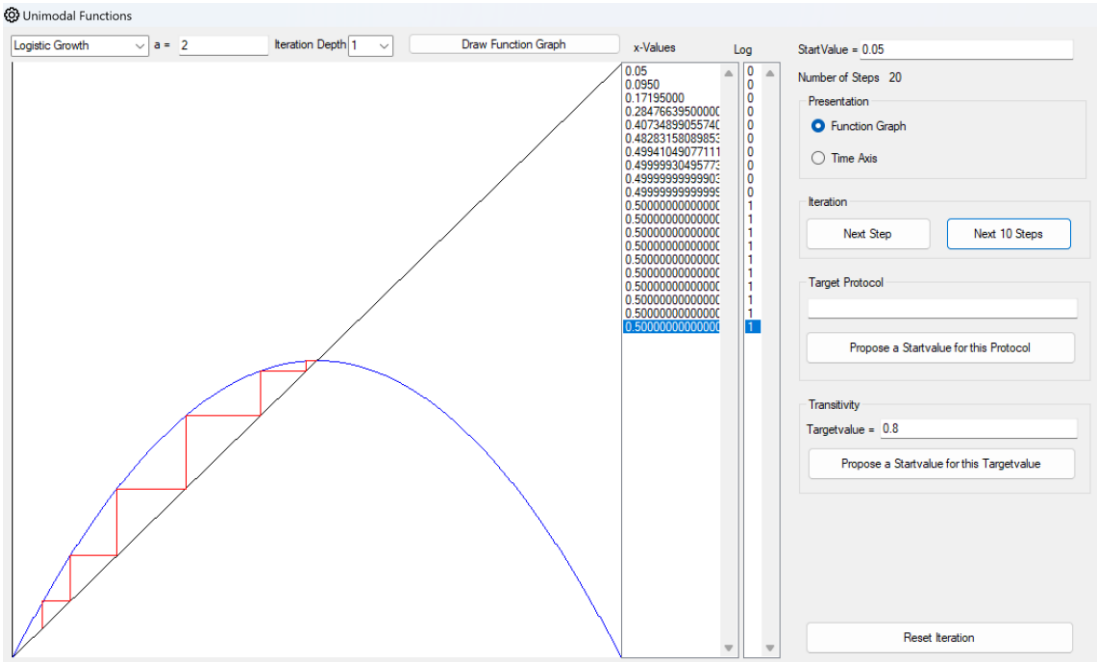
Whether these fixed points are attractive or repulsive depends on their multiplier or the value of the derivative in the absolute value at their position. It is:

$$\lambda = |f'(x)| = |a(1-2x)|$$

The first multiplier is $\lambda_1 = |f'(0)| = a$. So, the first fixed point $\xi_1 = 0$ is attractive for $a < 1$ and repulsive for $a > 1$. Furthermore, it is:

$$\lambda_2 = \left|f'(1 - {}^1/_a)\right| = |2 - a| < 1\ for\ a \in]1,3[$$

The fixed point $\xi_2 = 1 - {}^1/_a$ is attractive for a$\in]1,3[$ and repulsive for a$\in]3,4[$.



Logistic growth for a = 2 with attractive fixed point $\xi_2 = 0.5$

Cycles of period 2 are fixed points of f^2. $f^2(\xi) = \xi$ provides an equation of the fourth degree, where we already know two zeros, namely the fixed points of f: $\xi_1 = 0\ and\ \xi_2 = 1 - {}^1/_a$.

To find the other two fixed points $\xi_3 and\ \xi_4$ we use the approach:

$$\begin{cases} \xi_3 = f(\xi_4) = a\xi_4(1-\xi_4) \\ \xi_4 = f(\xi_3) = a\xi_3(1-\xi_3) \end{cases}$$

If you subtract the lower equation from the upper one, you get:

$$\xi_3 - \xi_4 = -a(\xi_3 - \xi_4) + a(\xi_3^2 - \xi_4^2)$$

Since the fixed points we are looking for are truly 2-periodic, the following applies $\xi_3 \neq \xi_4$ and we get

$$1 = -a + a(\xi_3 + \xi_4)$$

We put $\xi_4 = \frac{1+a}{a} - \xi_3$ into the second equation and after a little calculation we get

$$a^2\xi_3^2 - a(1+a)\xi_3 + 1 + a = 0$$

That delivers:

$$\xi_{3,4} = \frac{1 + a \pm \sqrt{(1+a)(a-3)}}{2a}$$

For $a > 3$ we therefore get a 2-cycle.

This is attractive if

$$|f^{2\prime}(\xi_3)| = |f'(f(\xi_3) \cdot f'(\xi_3)| = |f'(\xi_4) \cdot f'(\xi_3)| < 1$$

It is:

$$f'(x) = a(1-2x)$$

$$|f'(\xi_4) \cdot f'(\xi_3)| = a^2|1 - 2(\xi_3 + \xi_4) + 4\xi_3\xi_4|$$

With

$$\xi_3 + \xi_4 = \frac{1+a}{a} \; and \; \xi_3\xi_4 = \frac{1+a}{a^2}$$

We receive for the multiplier of the 2-cycle:

$$\lambda_{3,4} = \; |f'(\xi_4) \cdot f'(\xi_3)| = |a^2 - 2a - 4|$$

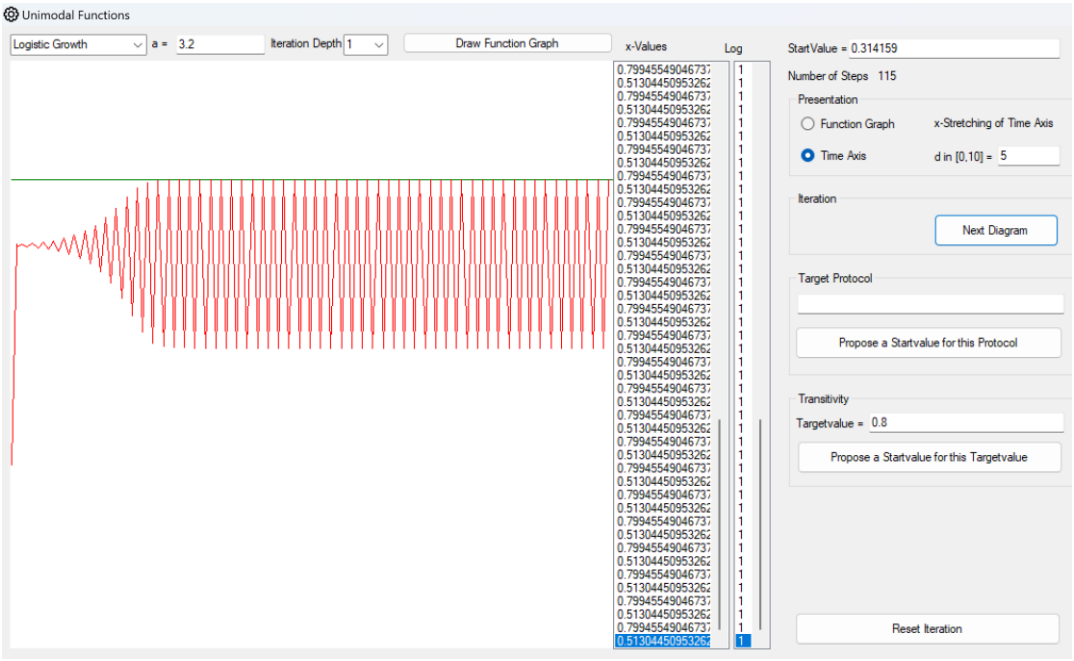
The 2-cycle is attractive for $\lambda_{3,4} < 1$ and we examine the transition points:

$$a^2 - 2a - 4 = \pm 1$$

This equation has the solutions $a = 1 \pm \sqrt{6}$ where because of $a > 0$ only comes into question:

$$a = 1 + \sqrt{6} \approx 3.449499$$

The 2-cycle is therefore attractive in the interval $a \in]3, 3.449499[$.



Logistic growth for a = 3.2, shown on the time axis

In the figure above, a = 3.2. The sequence soon settles into the attractive 2-cycle:

$$\xi_{3,4} = \begin{cases} 0.799455 \ldots \\ 0.513044 \ldots \end{cases}$$

Any further cycles are fixed points of f^k, i.e. solutions of equations of degree 2^k. The fixed points of f^k are the intersection points of the graph of f^k with the 45° straight line.

3.4. Conjugates of the Tent Map

The tent map is defined as:

$$z\colon [0,1] \longrightarrow [0,1];\; z(u) = \begin{cases} 2u, u \in [0,0.5[\\ 2(1-u), u \in [0.5,1] \end{cases}$$

Now let [a, b] be a real interval and $T: [0,1] \to [a,b]$ be a diffeomorphism. This means that both *T and* the inverse mapping T^{-1} are continuously differentiable.

Then we call the mapping *f*:

$$f \coloneqq T \circ z \circ T^{-1}: [a,b] \to [a,b]$$

A *conjugate* of the tent map *z*.

Example

The logistic growth for a = 4 is a conjugate of the tent map. The corresponding transformation is given by

$$T: [0,1] \to [0,1], u \in [0,1] \longmapsto x = T(u) = sin^2 \frac{\pi}{2} u \in [0,1]$$

T is bijective, continuously differentiable and likewise T^{-1} is continuously differentiable.

Be *f* the logistic growth. We show: $f \equiv T \circ z \circ T^{-1}\ auf\ [0,1]$. Let $u \in [0,1]$. then holds:

$$f\big(T(u)\big) = f\left(sin^2 \frac{\pi}{2} u\right) = 4sin^2 \frac{\pi}{2} u \left(1 - sin^2 \frac{\pi}{2} u\right) = (2sin \frac{\pi}{2} u \cdot cos \frac{\pi}{2} u)^2 = sin^2 \frac{\pi}{2} \cdot 2u$$
$$= \begin{cases} sin^2 \frac{\pi}{2} \cdot 2u, u \in [0,0.5[\\ sin^2 \frac{\pi}{2} \cdot 2(1-u), u \in [0.5,1] \end{cases} = \begin{cases} sin^2 \frac{\pi}{2} \cdot 2u \\ sin^2(\pi - \frac{\pi}{2} \cdot 2u) \end{cases} = T(z(u))$$

Please note: $\sin(\pi - \alpha) = sin\alpha$

This means that [0, 1]: $f \circ T \equiv T \circ z$ □

A conjugate of the tent map adopts its properties in terms of periodic and chaotic behaviour.

Theorem

Let f be a conjugate of the tent map z under the diffeomorphism T. Then holds:

1) $f^n = T \circ z^n \circ T^{-1}, \forall n \in \mathbb{N}$ *i.e.* f^n *is a conjugate of* z^n
2) $\{u_1, u_2, \dots\ u_k\}$ *is a k-cycle of z* $\Leftrightarrow \{T(u_1), T(u_2), \dots, T(u_k)\}$ *is a k-cycle of f*
3) *The k-cycle of z and the corresponding k-cycle of f have the same multiplier*
4) *Each k-cycle of f is repulsive. The set of these cycles is dense in [0,1]*
5) *f is sensitive and transitive*
6) *f has chaotic properties*

Thus, the logistic growth for a = 4 is chaotic.

Proof of the theorem

1) $f^n = (T \circ z \circ T^{-1})^n = T \circ z \circ T^{-1} \circ T \circ z \circ T^{-1} \circ \dots\dots \circ T \circ z \circ T^{-1} = T \circ z^n \circ T^{-1}$

2) $u \in [0,1]$ is a fixed point of $z^k \Leftrightarrow T(u)$ is a fixed point of f^k:

$$\text{" } \Rightarrow \text{": } z^k(u) = u \Rightarrow f^k\big(T(u)\big) = \big(T \circ z^k \circ T^{-1}\big)\big(T(u)\big) = T\left(z^k(u)\right) = T(u)$$

$$\text{" } \Leftarrow \text{": } T\left(z^k(u)\right) = T(u) \Rightarrow z^k(u) = u$$

3) Let $\xi = T(u)$ for a $u \in]0,1[$ and suppose, *u* is a fixed point of z^k, i.e. a cycle of *z* (the case u = 0 and u = 1 is trivial). Therefore, it holds $z^k(u) = u$. Then ξ is a fixed point of f^k: $f^k(\xi) = \xi$.

Also note: $T\left(T^{-1}(\xi)\right) = \xi$, thus $\left[T\left(T^{-1}(\xi)\right)\right]' = T'(u) \cdot T^{-1\prime}(\xi) = 1$, $u \in]0,1[$ according to the theorem on the derivation of the inverse mapping.

For the multipliers it holds according to the chain rule:

$$\lambda_\xi = \left|f^{k\prime}(\xi)\right| = \left|\left(T^{\circ}z^{k\circ}T^{-1}\right)'(\xi)\right| = \left|T'(z^k\left(T^{-1}(\xi)\right)) \cdot z^{k\prime}\left(T^{-1}(\xi)\right) \cdot T^{-1\prime}(\xi)\right| =$$

$$\left|T'\left(z^k(u)\right) \cdot \ z^{k'}(u) \cdot T^{-1'}(\xi)\right| = \left|T'(u) \cdot z^{k'}(u) \cdot T^{-1'}(\xi)\right| = \left|z^{k'}(u)\right| = \lambda_u$$

4) Follows directly from 3) as all cycles of the tent map are repulsive.

Due to the continuity of *T*, the cycles of *f* are dense in [0, 1]. We will dispense with the ε-justification and merely sketch the proof: Let $\xi \in [0,1]$ and $u = T^{-1}(\xi)$. Then there is a cyclic point *v* in arbitrary proximity to *u*. Since *T* is continuous, the likewise cyclic point $T(v)$ is arbitrarily close to $\xi = T(u)$.

5) Evidence sketch:

Transitivity: Let $\xi_1 \in [0,1]$ an initial value and $u_1 = T^{-1}(\xi_1)$. Let $\zeta \in [0,1]$ a target value and $v = T^{-1}(\zeta)$. Because of the transitivity of *z,* there is a point u_2 in any proximity of u_1, which comes arbitrarily close to *v* during the iteration. Because of the continuity of *T,* $T(u_2)$ comes arbitrarily close to the target point $\zeta = T(v)$.

Sensitivity: *T* is a diffeomorphism. In particular, the inverse function T^{-1} is also continuous.

Now be $\xi_1, \xi_2 \in [0,1]$ arbitrarily close to each other. Then, if these points are suitably close to each other, then, due to the continuity of T^{-1} the points $u_1 = T^{-1}(\xi_1), u_2 = T^{-1}(\xi_2)$ are arbitrarily close to each other. During the iteration of the tent map, the distance between these points can grow arbitrarily within the interval [0,1]. In particular, there is a $n \in \mathbb{N}$ such that $z^n(u_1)$ is close to 0 and $z^n(u_2)$ is close to 1 (or vice versa). Due to the continuity of T: $[0,1] \longrightarrow [a,b]$ then $f^n(\xi_1) = T(z^n(u_1))$ is close to *a* and $f^n(\xi_2) = T(z^n(u_2))$ is close to *b* (or vice versa).

Remark

This approach is a simple example of a more general situation. We have found a transformation $T(u) = sin^2\frac{\pi}{2}u, u \in [0,1]$ so that essentially the following applies:

$$f\left(T(u)\right) = T(2u)$$

When we enter the realm of complex numbers, a theorem applies that goes back to Henri Poincaré and is known as the Poincaré functional equation:

Let $f(z), z \in \mathbb{C}$ be a holomorphic function ain $z = 0$ with $f(0) = 0, \lambda \coloneqq f'(0), |\lambda| > 1$. Then there is a holomorphic function $T(Z)$ in $z = 0$, $T(0) = 0$, such that :

$$f\left(T(z)\right) = T(\lambda z)$$

If it is also required that $T'(0) = 1$, then T is uniquely determined. T is called the Poincaré function.

If an iteration of the form $x_{n+1} = f(x_n)$ is defined, the following applies under the above conditions:

$$x_n = T(\lambda^n c)$$

This is $c \in \mathbb{C}$ is a constant that depends on the start value x_0 is a constant.

However, the Poincaré function is only an elementary function in a few special cases, such as for the logistic function with $a = 4$.

Protocols

In the "Simulator" we create a log of the sequence of numbers generated by the logistic growth. We set as protocol:

$$p(x_n) = \begin{cases} 0, if\ x_n \in [0,0.5[\\ 1, if\ x_n \in [0.5,1] \end{cases}$$

In the chaotic case $a = 4$, the logistic growth is a conjugate of the tent map. The following then applies to the transformation: $T(0.5) = sin^2 \frac{\pi}{4} = 0.5$ and *T* is monotonically increasing. That implies:

The logs of the logistic growth for $a = 4$ and the tent map match in the iteration!

In the tent map, we have found a bijection between protocols and initial values by specifying an inverse mapping that assigns an initial value to any protocol, which results in this protocol during iteration.

This also gives us an inverse function for logging logistic growth:

Starting from any protocol, we calculate the corresponding start value for the tent map, which results in this protocol during iteration. We have already explicitly described this calculation in the section on tent map. We then use again the transformation

$$x = sin^2 \frac{\pi}{2} u$$

to calculate the corresponding starting value for the logistic growth. This provides exactly the specified log during iteration.

Example

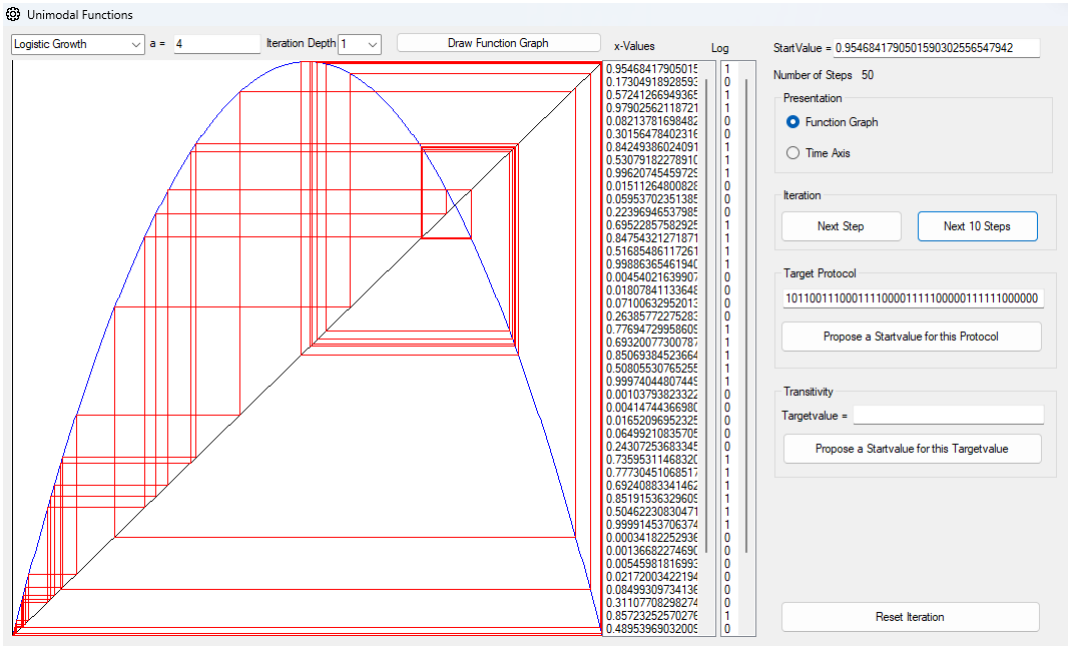
We specify the target protocol again:

101100111000111100001111100000111111000000

The starting value suggested by the "Simulator"

0.954684179050159030255654 7942

returns the target protocol. It is important that the parameter is a = 4, otherwise there may be no chaotic behaviour. The algorithm implemented in the "Simulator" works only for the case a = 4.



Logistic growth with specified log of length 42

3.5. The "normalized" Parabola

In the literature, the logistic growth function is often replaced by the parabola

$$g(y) = 1 - \mu y^2, [-1,1] \longrightarrow [-1,1], \mu \in]0,2]$$

to discuss its iteration. To distinguish from the logistic growth, we call the function above "normalized" parabola, because only the quadratic element appears in the formula.

Through the transformation

$$T: y \mapsto \frac{\mu}{a} y + \frac{1}{2}, [-1,1] \longrightarrow [0,1]$$

Whereby $a \in]2,4]$ and $\mu = \frac{a(a-2)}{4}$ this parabola can be traced back to an area of logistic growth. It applies to $y \in [-1,1]$:

$$f(T(y)) = f\left(\frac{\mu}{a} y + \frac{1}{2}\right) = a\left(\frac{\mu}{a} y + \frac{1}{2}\right) - a\left(\frac{\mu}{a} y + \frac{1}{2}\right)^2 = \frac{a}{4} - \frac{\mu^2}{a} y^2$$

And

$$T(g(y)) = T(1 - \mu y^2) = \frac{\mu}{a}(1 - \mu y^2) + \frac{1}{2} = \frac{\mu}{a} + \frac{1}{2} - \frac{\mu^2}{a} y^2 = \frac{a}{4} - \frac{\mu^2}{a} y^2$$

Because

$$\frac{\mu}{a} + \frac{1}{2} = \frac{a-2}{4} + \frac{1}{2} = \frac{a}{4}$$

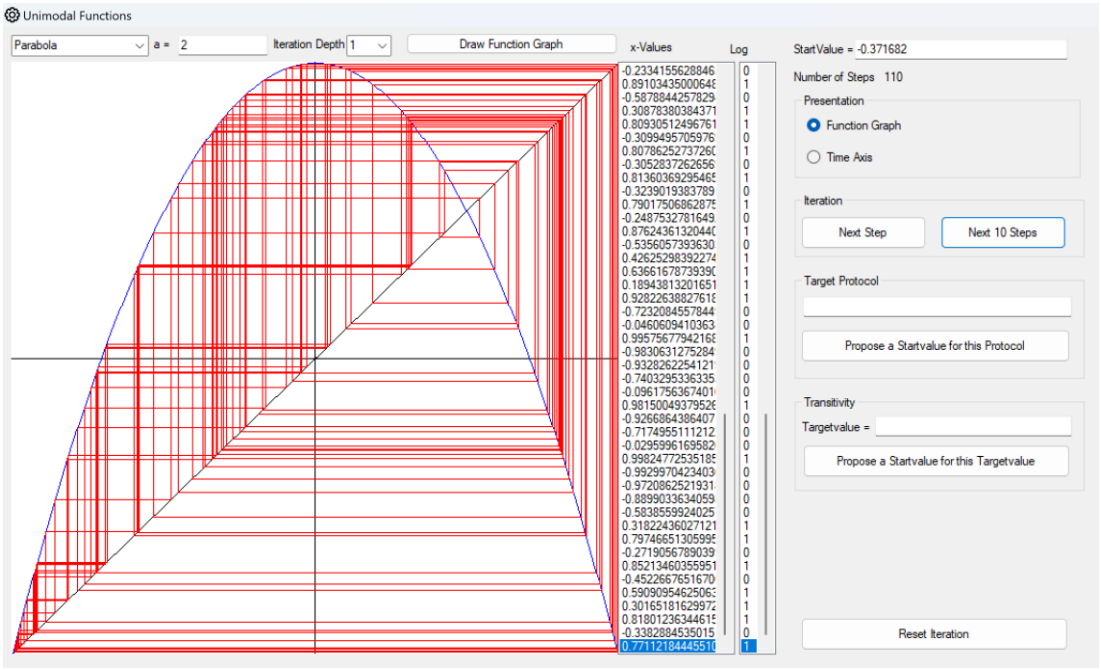
Therefore $Tg \equiv fT$ resp. $g \equiv T^{-1} fT$ on $[-1,1]$. For $a \in]2,4]$ is $\mu \in]0.2]$.

This parabola is also for $\mu = 2$ a conjugate of the tent map. The transformation is given by:

$$T\colon [0,1] \longrightarrow [-1,1], u \in [0,1] \longmapsto T(u) = cos\pi(1-u) \in [-1,1]$$

T is bijective, continuously differentiable and strictly monotonically increasing. $T'(u) = \frac{1}{\pi} sin\pi u > 0$ for $u \in [0,1]$.

This function is therefore also chaotic for $a = 2$ (in the "Simulator", the parameter is denoted by a instead of μ).



Iteration of the normalized parabola for a = 2

Like the logistic growth, this parabola is also a conjugate of the tent map, namely for the parameter value $\mu = 2$. The conjugation is given by the transformation:

$$T\colon [0,1] \to [-1,1], u \in [0,1] \longmapsto x = T(u) = \cos(\pi(1-u)), x \in [-1,1]$$

It applies to , $u \in [0,1]$:

$$f\big(T(u)\big) = f(\cos(\pi(1-u))) = 1 - 2cos^2(\pi(1-u)) = sin^2(\pi(1-u)) - cos^2(\pi(1-u)) =$$

$$-\cos\big(2\pi(1-u)\big) = -\cos(2\pi u) = \cos(\pi(1-2u))$$

And on the other hand, for the tent map:

$$T\big(z(u)\big) = \begin{cases} T(2u), u \in [0,0.5[\\ T\big(2(1-u)\big), u \in [0.5,1] \end{cases}$$

In the first case:

$$T(2u) = \cos\big(\pi(1-2u)\big)$$

In the second case:

$$T\big(2(1-u)\big) = \cos\big(\pi(1-2+2u)\big) = \cos(-\pi + 2\pi u) = \cos(\pi(1-2u))$$

For $u \in [0,1]$ therefore applies $fT \equiv Tz$ resp. $f \equiv TzT^{-1}$.

This means that the parabola we are looking at has for $\mu = 2$ chaotic properties.

Protocols

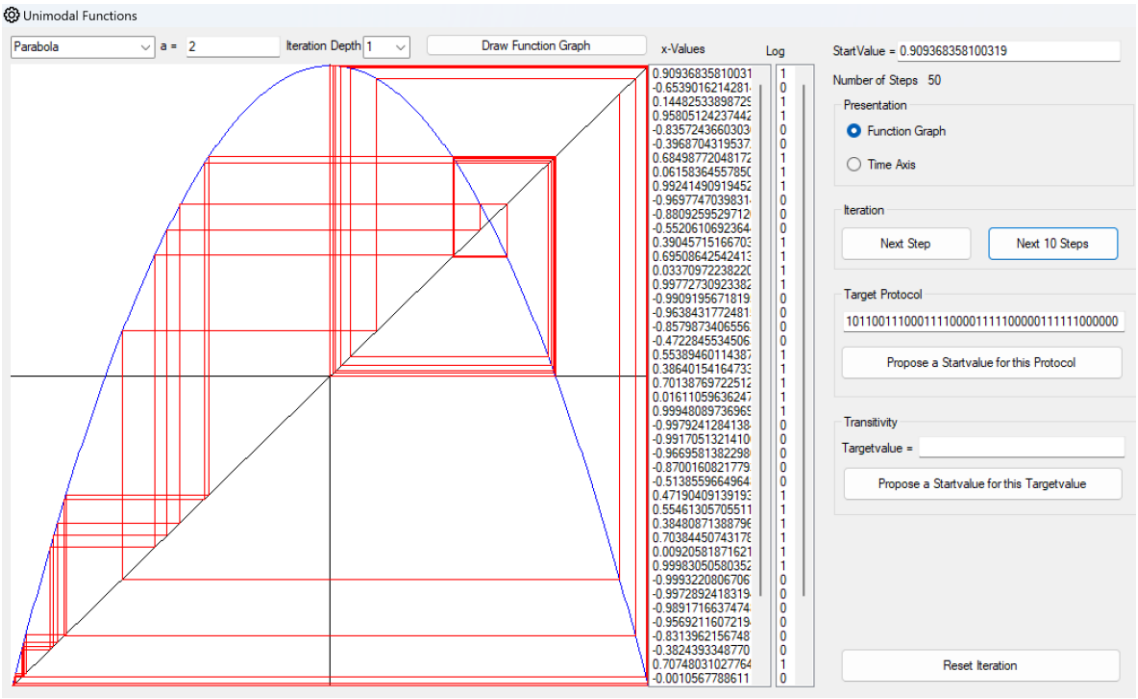
We define a protocol as follows:

$$p(x_n) = \begin{cases} 0, if\ x_n \in [-1,0] \\ 1, if\ x_n \in]0,1] \end{cases}$$

In the chaotic case $\mu = 2$ the parabola considered here is a conjugate of the tent map. The following then applies to the transformation: $T(0.5) = -cos\,{}^{\pi}/_{2} = 0$ and *T* is monotonically increasing. Thus it follows:

The logs of the normalized parabola for $\mu = 2$ *and the tent map match during iteration.*

As with the logistic growth, this makes it possible to first find a corresponding initial value for the tent map for a given log, which provides this log. The transformation *T* then provides the corresponding starting value for the normalized parabola.



Iteration of the normalized parabola with a given log of length 42

3.6. Implementation in the "Simulator"

The *FrmIteration* window is available for displaying the iteration. It enables the selection of the iterated function, the selection of the parameter, the definition of the iteration depth and the selection of the start value of the iteration. Its function is described in detail in the "Manual".

The result of the iteration can be displayed in the function graph or on the time axis. A start value can be suggested for a specified protocol, which provides the specified protocol. If a target value is specified in the case of transitivity, a new start value is calculated close to the original one so that the iteration sequence comes as close as possible to the target value.

Internally, this window works with an interface *IIteration*, which manages all the necessary properties and offers methods. These include

- Internal definition of the permissible parameter interval of the iteration and test whether a parameter value specified by the user lies within this interval
- Internal definition of the permissible iteration interval and test whether a start value specified by the user lies within this interval
- Internal definition of the iteration function and the corresponding multi-iterated function if the iteration depth is > 1.
- A method that calculates a corresponding start value for a given protocol
- A method that changes the start value for a given target value so that the target value is reached as accurately as possible during the iteration
- Implementation of the iteration steps

The *FrmIteration* then takes over the representation of the orbit of the iteration.

The individual iteration functions

- Tent Map
- Logistic Growth
- "Standardized" Parabola

are represented by corresponding classes that implement the *IIteration* interface. This makes it very easy to add further examples of iteration functions to the program. You can simply create a new class that implements the *IIteration* interface.

So that the denominations are also clear, the variables for logistic growth and the normalized parabola are consistently denominated as (x, y). The variables for the tent map, on the other hand, are referred to as (u, v). This also corresponds to the denominations in this mathematical documentation.

For all calculations, we work in a coordinate system that is defined by a value range for the x-coordinate and y-coordinate.

$$x \in [x_{min}, x_{max}], y \in [y_{min}, y_{max}]$$

A point (x, y) in this coordinate system is represented by an object of the *ClsMathPoint* class.

In the constructor, the picture box or bitmap in which the drawing is to take place and the intervals are transferred to the *ClsGraphicTool* class $[x_{min}, x_{max}], [y_{min}, y_{max}]$. The *ClsGraphicTool* class then has the necessary methods to generate drawings: *DrawCoordinatesystem, DrawPoint, DrawLine, DrawRectangle, DrawEllipse,* etc*.* The mathematical coordinates or objects of type *ClsMathPoint* are always transferred. The *ClsGraphicTool* class then converts the mathematical coordinates into pixel coordinates based on the properties of the transferred picture box or bitmap. *This means that the programmer does not have to worry about pixel coordinates but can work in mathematical coordinates.* This applies not only in the case of the unimodal functions presented here, but also later in billiards or other examples.

In order to find a start value for a given protocol or a given target value, the corresponding conjugation transformations for tent map are used: First calculate the corresponding starting value for the tent map as described in the corresponding previous section on tent map. Then the initial value for the logistic growth or the normalized parabola is obtained by applying the corresponding conjugation transformation described in the sections on these iterations.

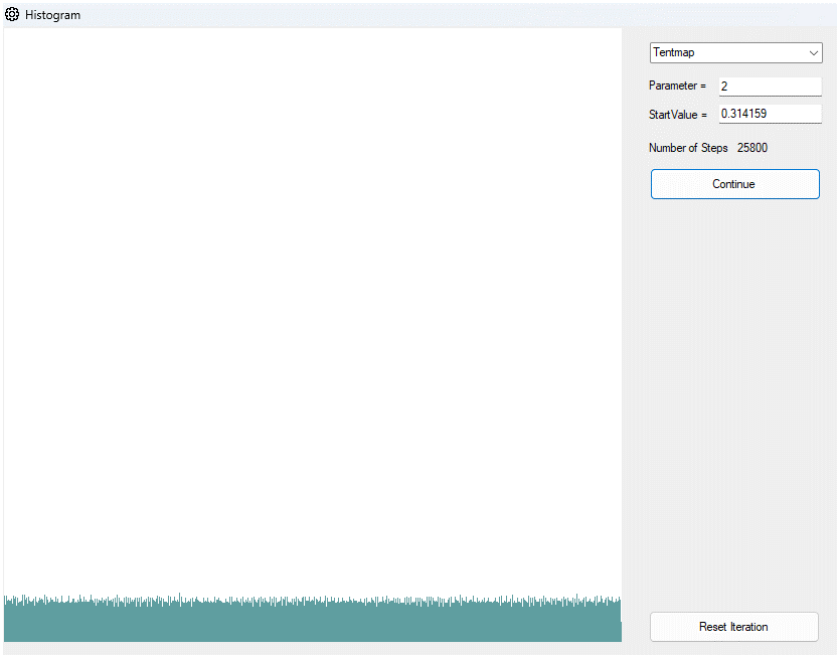
The sensitivity is displayed in the same way. *FrmSensitivity* allows you to select the corresponding parameters and settings. Its function is described in detail in the "Manual". The *IIteration* interface is also used there, which is implemented by the corresponding classes.

Again, the same variable names are used throughout as in the *FrmIteration*.

The code is provided with detailed comments (in English) throughout. The "Manual" offers the user a quick introduction to the use of the "Simulator".

3.7. Histograms in the Chaotic Case

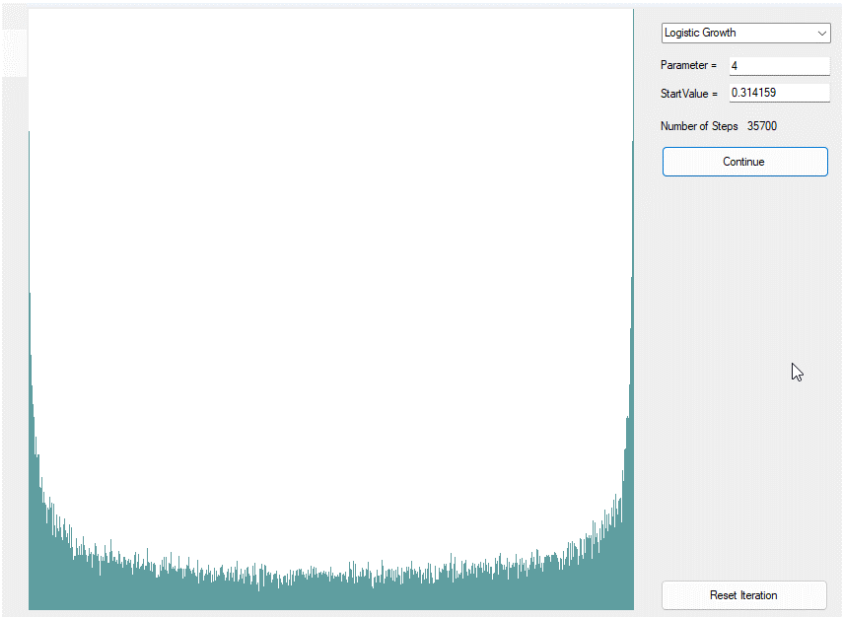
Here we consider the chaotic case, i.e. the tent map, the logistic growth with parameter a = 4 or the parabola with parameter a = 2. Other parameter values can also be selected, but then the note appears that the properties of the sensitivity are then not guaranteed. Starting from an (aperiodic) initial value $x_0 \in]0,1[$ for the tent map or logistic growth, or $x_0 \in [-1,1]$ for the parabola, we investigate how the sequence $f^n(x_0), n \in \mathbb{N}$ is "distributed" in the iteration interval. In the "Simulator", we divide this interval into small sub-intervals of pixel width 1 and create a histogram of how often the sub-interval is hit during the iteration.



Distribution of the intervals taken in the tent map

As you can see, the tent map produces a uniform distribution. This means that the iteration provides an orbit that looks random, although there is a simple law of motion behind the iteration. This effect is also known as "pseudo randomness".

A similar picture emerges with the logistic mapping, whereby the edge points of the iteration interval are obviously hit more frequently.



Histogram of logistic growth after 35,700 steps

Based on the uniformly distributed tent map, this distribution is an effect of the transformation to logistic growth given by $x = T(u) = sin^2 \frac{\pi}{2} u$:

If $p(x)$ is the probability distribution for logistic growth, then the probability that an iteration value falls into an interval Δx is approximately $p(x)\Delta x$. However, this is equal to the probability that $u = T^{-1}(x)$ falls into the corresponding interval Δu falls into the corresponding interval. However, this probability is even Δu, because we have a uniform distribution in the tent map on the interval [0,1]. It is therefore $p(x)\Delta x \approx \Delta u$. It follows in the limit:

$$p(x) = \lim_{\Delta x \to 0} \frac{\Delta u}{\Delta x} = T^{-1\prime}(u) = \frac{1}{T'(u)} = \frac{2}{\pi \sin(\pi u)}$$

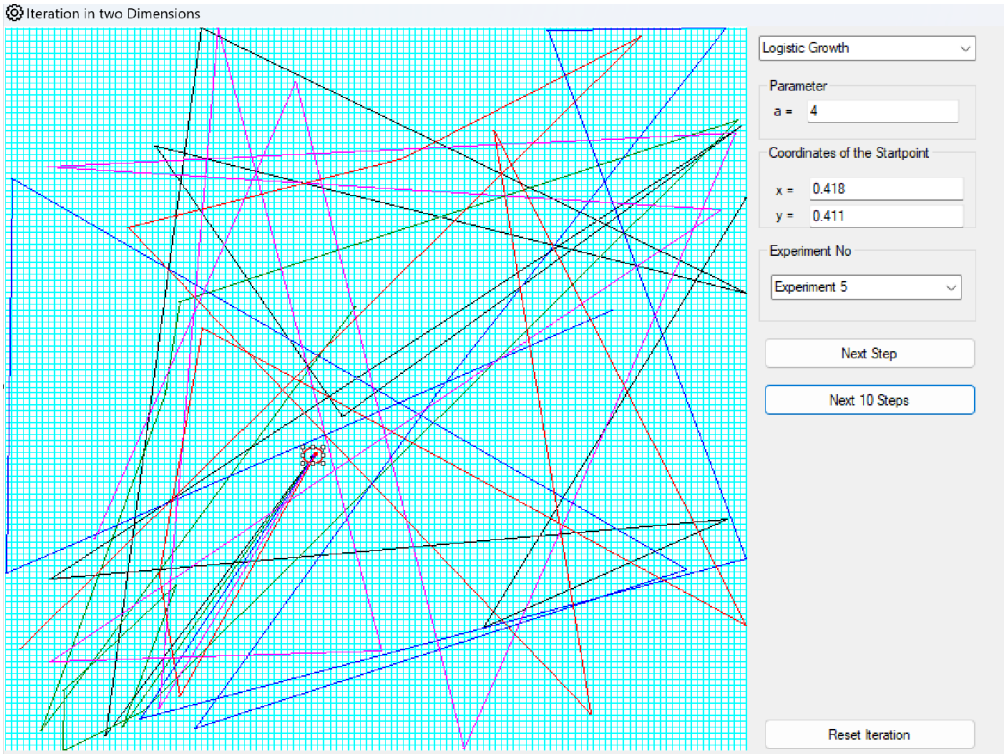
This explains the symmetry of the histogram of logistic growth with respect to $x = 0.5$ resp.

$u = 0.5$ and also the shape of the distribution at the edge of the iteration interval [0,1].

3.8. Two-dimensional Representation and Transitivity in the chaotic Case

In the "Simulator", we visualize the situation of an experimenter who is investigating a chaotic system, behind which logistic growth is the law of motion. The system moves in the state space $[0,1]x[0,1]$ and a system state is given by two parameters $(x, y) \in [0,1]x[0,1]$. The experimenter chooses a starting point $(x_0, y_0) \in [0,1]x[0,1]$ and then investigates how the system behaves from this starting point. The law of motion is then, for example, for logistic growth $x_{n+1} = 4x_n(1 - x_n)$ and $y_{n+1} = 4y_n(1 - y_n)$. It is therefore the same iteration for each component.

The experimenter measures with limited accuracy. In the "Simulator", this is represented in such a way that the state space is divided into very small squares of 5x5 pixels, represented in the diagram by a grid. For an x-value in the centre of a square, this corresponds to x-values with a deviation of ± 0.004 lie in the same square. The same applies analogously to the y-values. All starting values in such a square are perceived by the experimenter as the "same" starting value. This is marked by an orange circle in the following diagram.



State space of the experimenter in the "Simulator"

If the start values in different experiments differ only slightly enough to be in the same start square, then some target values will be reached after many steps due to transitivity. In the picture above, the experimenter carries out 5 experiments starting from the starting value marked with a small circle. In reality, each start value is minimally different. For each experiment, the orbit of the starting point is shown in a different colour. As you can see, these orbits are clearly different after a few steps.

For the experimenter, this looks like any orbits are generated starting from the same initial value. The system seems to behave randomly and a law of motion is not apparent.

3.9. Unimodal functions

For functions in which the state space is $X = [a, b] \subset \mathbb{R}$ is stretched and then folded again, chaotic properties can be determined under certain conditions. *Unimodal* mappings are among the simplest of these. All previous examples of iteration functions are unimodal.

A continuous mapping $f: [a, b] \longrightarrow [a, b], [a, b] \subset \mathbb{R}$ is called *unimodal* if a point $c \in [a, b]$ exists such that: *f* is strictly monotonically increasing in $[a, c[$ and in $]c, b]$ strictly monotonically decreasing. Furthermore, the following should apply:

$$f(c) = b \; and \; f(b) = a$$

The function *f* therefore assumes a maximum at *c* and this is the only maximum of the function. *c* is the *critical point* of the mapping *f*.

The condition $f(b) = a$ is often not required for unimodality and this is not a significant restriction.

Examples of unimodal functions:

- The Tent Map. Here c = 0.5

- The logistic Growth for parameters $a \in]0,4]$. Then c = 0.5
- The normalized parabola for all parameter values $a \in]0,2]$. Here c = 0

Unimodal functions are a generalization of the tent map and its conjugates.

A general discussion of these functions would go too far in this document. We only mention one theorem and refer to the literature, see [5].

Theorem

If a function $f: [a,b] \longrightarrow [a,b] \subset \mathbb{R}$ *is unimodal and given by a second-degree polynomial, then*

- *There is at most one attractive cycle in* $[a,b]$
- *If there is such an attractive cycle, then the critical point lies in its basin*

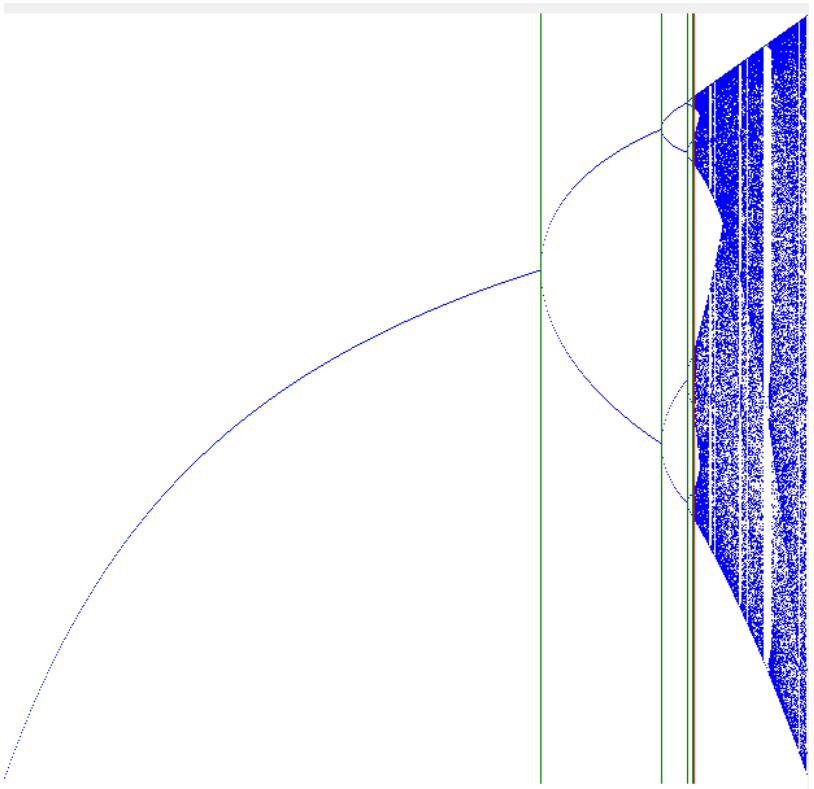
For the investigation of chaotic properties, it is already sufficient to examine unimodal functions defined by quadratic polynomials. Now we also know that it is sufficient to examine the orbit of the critical point, i.e. the point at which the quadratic function assumes the maximum.

3.10. Period Doubling

We have seen that the logistic growth for parameter values $a < 1$ has the attractive fixed point $\xi_1 = 0$. For $a \in]1,3[$, this fixed point becomes repulsive, but it is replaced by the attractive fixed point $\xi_2 = 1 - {}^{1}/_{a}$. It is attractive for $a \in]1,3[$. At the point $a = 3$, this fixed point becomes repulsive, but an attractive 2-cycle is created

$$\xi_{3,4} = \begin{cases} 0.799455 \dots \\ 0.513044 \dots \end{cases}$$

This is attractive in the area of $a \in]3,1+\sqrt{6}[$ and becomes repulsive for $a = 1+\sqrt{6}$. The computer experiment then shows that an attractive 4-cycle is created here. For increasing *a*, an 8-cycle then arises, then a 16-cycle and the period of the cycles doubles continuously up to a certain limit value of *a* and then goes over to chaotic behaviour. This is shown in the following diagram, which is named after the mathematician Mitchell Feigenbaum, who studied the phenomenon of period doubling in depth in 1975.



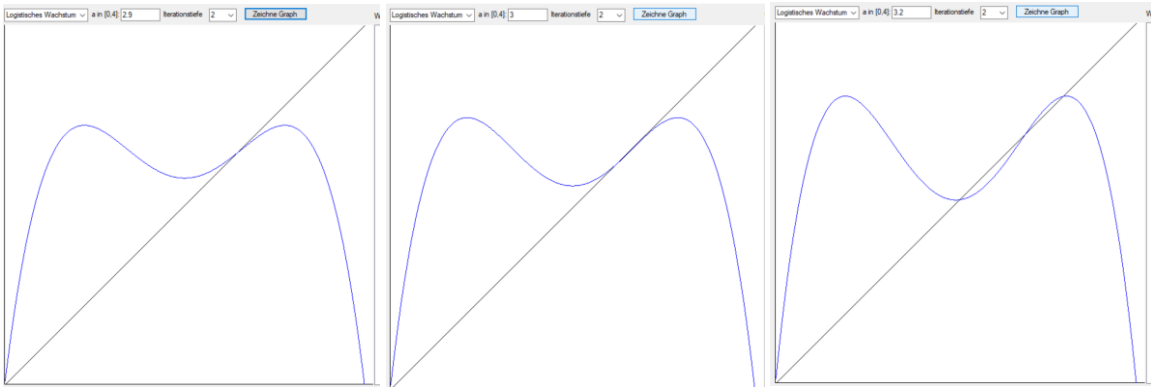
Feigenbaum diagram for the logistic growth

The values of the parameter *a* are shown on the horizontal axis. The interval in which *a* moves can be freely selected within]0,4]. The image above shows a section for $a \in [1, 4]$. For each value of *a*, the iteration is then carried out for so long that it can be hoped that the iteration either settles on one cycle or is chaotic. Further iteration steps are then carried out and the x-values of the iteration are entered vertically in the diagram.

The green lines show the split points at which an attractive cycle becomes unstable and a new cycle is created. The first split point is used for $a_1 = 3$. An attractive 2-cycle is created there. At the point $a_2 = 3.449499$ ... the period doubles and an attractive 4-cycle is created at the corresponding two split points. At the point $a_3 = 3.544090$ ... the attractive 8-cycle is created at four split points. In the further course, the period doubles at ever shorter intervals and the sequence of the a_i, i.e. the points at which further split points occur, tends towards the limit value $a_\infty = 3.569946$ .... This point is marked in the diagram with a red line.

The split points each belong to a 2^n-cycle, which becomes attractive at this point. Since all points belonging to a cycle have the same multiplier, as already stated earlier, they become attractive at the same time, i.e. they all lie on a straight line parallel to the vertical axis.

For example, what happens when the fixed point $\xi_2 = 1 - {}^1/_a$ changes into an attractive 2-cycle at the point *a=3,* one can see in the graph of f^2 at this point (generated by the "Simulator"):



a = 2.9a a = 3 a = 3.2

The fixed point ξ_2 is the intersection of the graph with the 45° straight line. On the left, the curve has a tangent with a slope < 1, so the fixed point is attractive. For *a = 3*, the tangent slope is exactly 1 (middle image). If the curve becomes steeper for *a = 3.2*, the fixed point ξ_2 has become repulsive with a tangent gradient > 1. Two new intersections have been created with a tangent gradient < 1. This is the newly created 2-cycle.

The first associated values of *a*, at which the respective cycles become unstable and split into a cycle of the double period, can be determined using numerical methods and are:

$$a_1 = 3, a_2 = 3.449499 \ldots, a_3 = 3.544090 \ldots, a_4 = 3.564407 \ldots,$$

$$a_5 = 3.568759 \ldots, a_6 = 3.569692 \ldots, a_7 = 3.569891 \ldots, a_8 = 3.569934 \ldots$$

The values of $a_k, k \in \mathbb{N}$ decrease geometrically and tend towards a limit value a_∞ according to:

$$a_k \approx a_\infty - c \cdot \delta^{-k}, k \in \mathbb{N}$$

$\delta \approx 4.669202 \ldots$ is the so-called FEigenbaum constant. The following applies to the logistic growth:

$$c \approx 2.6327 \ldots$$

For example it is:

$$a_\infty - c \cdot \delta^{-5} = 3.568759 \approx a_5$$

An n-cycle fulfils the condition $f^n(\xi_i) = \xi_i, \forall \xi_i \in Zyklus, 1 \leq i \leq 2$. In logistic growth, these points are therefore zeros of the polynomial $f^n(x) - x = 0$, which has the degree 2^n . Is it possible that some of these zeros are complex and the others are real? The answer is no. Assuming you know a real zero ξ_1. Then the other zeros result from the iteration $\xi_{i+1} = f(\xi_i), 1 \leq i < n$ and $\xi_1 = f(\xi_n)$. This means that all further zeros are also real.

Feigenbaum discovered that the behaviour of period doubling is a universal phenomenon and occurs in many dynamic systems during the transition to chaos. The constant $\mathcal{F}$ is always the same and also appears to be universal.

In the case of the parabola, the split points are obtained by applying the transformation between the logistic growth and the parabola: $\mu = \frac{a(a-2)}{4}$

This provides the following points for the parabola at which split points occur:

$$\mu_1 \approx 0.75, \mu_2 \approx 1.24995, \mu_3 \approx 1.3681, \mu_4 \approx 1.39405, \ldots., \mu_\infty \approx 1.401155$$

Here, too, the sequence $(\mu)_k$ follows the same law. We determine the associated constant $\tilde{c}$ by using the transformation between logistic growth and the parabola as follows:

$$\tilde{c} = \lim_{k\to\infty}(\mu_\infty - \mu_k)\delta^k = \lim_{k\to\infty}\frac{\delta^k}{4}(a_\infty(a_\infty - 2) - a_k(a_k - 2))$$

$$= \lim_{k\to\infty}\frac{\delta^k}{4}((a_\infty - a_k)(a_\infty + a_k) - 2(a_\infty - a_k)) = \lim_{k\to\infty}\frac{\delta^k}{4}(a_\infty - a_k)(a_\infty + a_k - 2)$$

$$= \lim_{k\to\infty}\frac{\delta^k}{4}c\delta^{-k}(a_\infty + a_k - 2) = \frac{c(a_\infty - 1)}{2} = 3.3829484\ldots$$

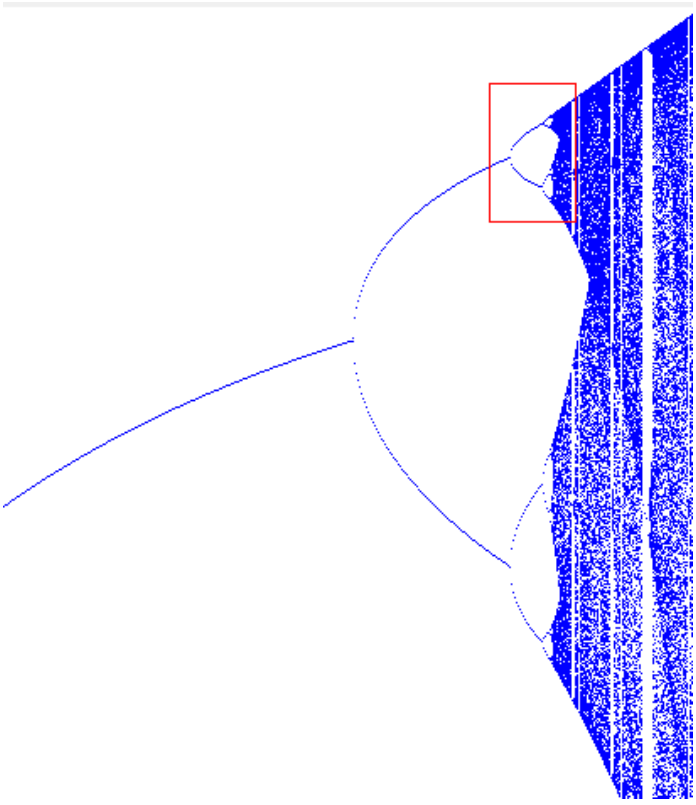
For example:

$$\mu_\infty - \tilde{c}\cdot\delta^{-4} = 1.3940375 \approx \mu_4$$

The scaling in the a-direction roughly follows the above geometric sequence with the Feigenbaum constant $\delta \approx 4.669202$ …. The scaling in the x-direction also follows a geometric sequence, whereby a universal scaling factor also occurs here, namely: $\alpha \approx 2.5029078$ …

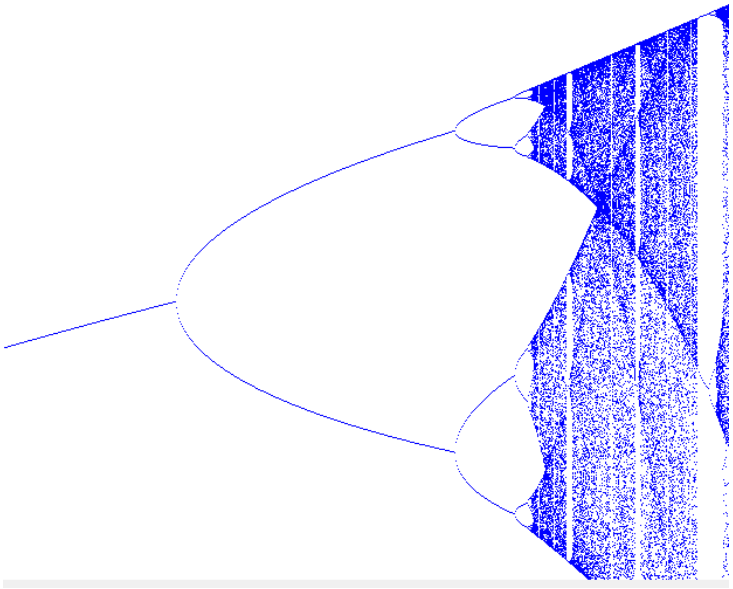
Self-similarity and renormalization

Feigenbaum used the phenomenon of self-similarity of a quadratic function and its iterates in his investigation. A good description of this approach can be found in [1] and an elementary approach in [4].



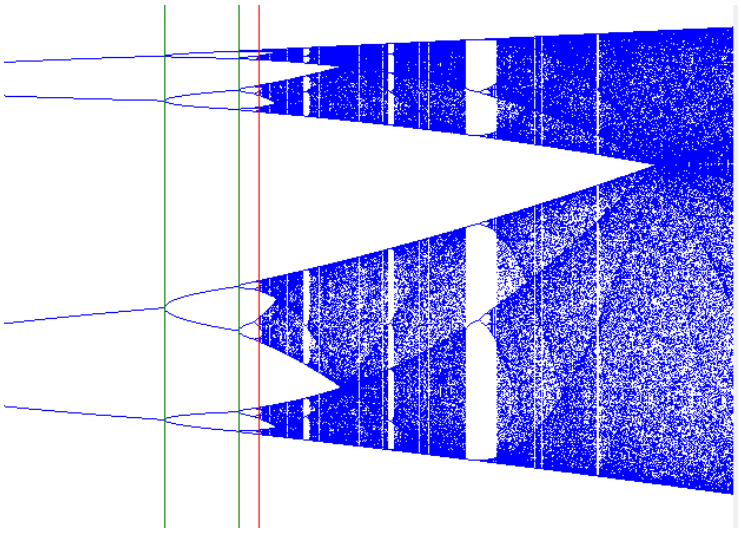
Self-similarity with period doubling

The "Simulator" makes it possible to view the section marked with a red rectangle in a correspondingly scaled iteration. The following image is obtained for the section above:



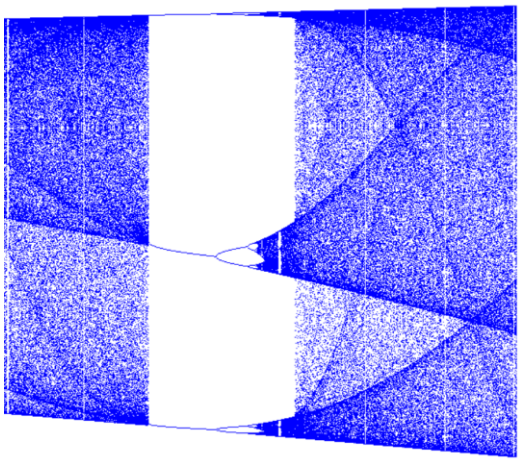
Red marked section in enlarged form

If a exceeds the value $a_\infty \approx 3.569946$, the system becomes chaotic.



Transition to chaos - the red line marks the approximate value $a_\infty \approx 3.569946$

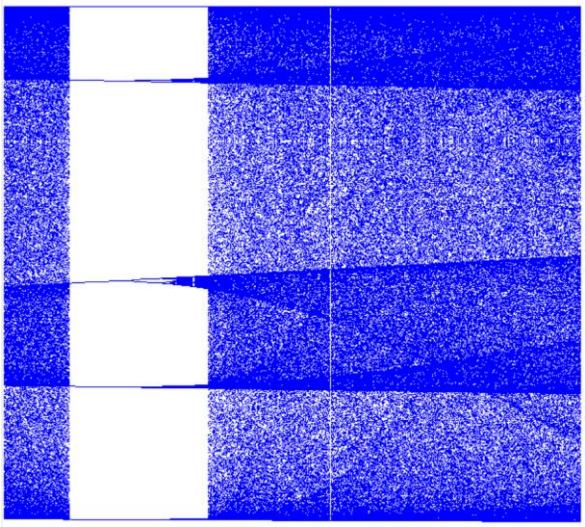
If a continues to increase, chaotic areas alternate with ever smaller windows containing attractive cycles. In the range $a \in [3.828427,3.841499]$, for example, an attractive 3-cycle appears, which then becomes unstable at the right-hand boundary of the interval. At this point, another cascade of period doublings takes place: The 3-cycle becomes a 6-cycle, then a 12-cycle and so on. At $a = 4$ chaos is definitely reached.



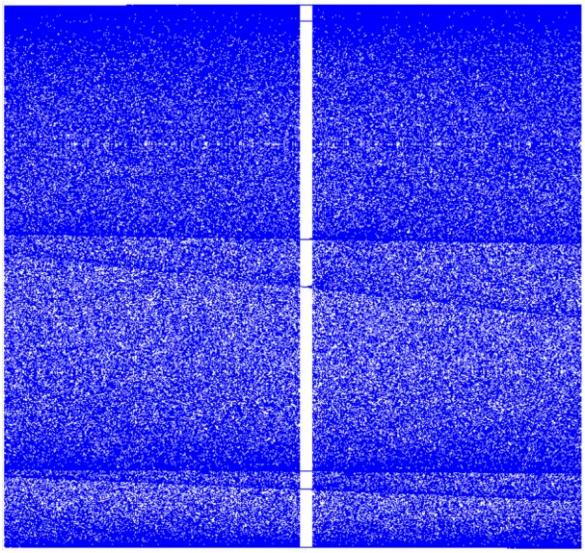
Attractive 3-cycle in the range $a \in [3.8,3.9]$ generated by the "Simulator"

In the image above, you can see how a stable 3-cycle appears out of nowhere at the point *a=3.82* after a chaotic area, which then transitions into period doubling and later ends up in chaos again. The self-similarity can be seen slightly to the right of the period doubling: Windows with stable 9-cycles also appear here, similar to the 3-cycle.

Even in the $a \in [3.905,3.91]$ windows with stable cycles still appear, as the following image shows. For example, you can see a stable 5-cycle on the left-hand side of the diagram. If you look closely, you can faintly see that it switches to period doubling as soon as it becomes unstable.



Attractive 5-cycle in the area of $a \in [3.905,3.91]$



Attractive 7-cycle in the area of $a \in [3.95,3.952]$

In this context, a theorem by Sarkovskii should be mentioned, which cannot be proven here, but gives an impression of the cycles that occur:

Movement (Sarkovskii 1964)

Consider the following order on $\mathbb{N}$*:*

$$3 > 5 > 7 > 9 > \cdots > 2 \cdot 3 > 2 \cdot 5 > 2 \cdot 7 > \cdots > 2^n \cdot 3 > 2^n \cdot 5 > \cdots > 2^n > \cdots > 4 > 2 > 1$$

Furthermore, let f be a unimodal function having a p-periodic cycle. Then f has cycles of each period $q < p$ *in the sense of the above order.*

A justification of this sentence is beyond the scope of this manuscript. However, a good approach can be found in [1]. An elementary approach can be found in [6].

Example

The logistic growth is defined by a unimodal function. The computer experiments have shown an attractive 3-cycle. Thus, the logistic growth has at least all cycles of the period

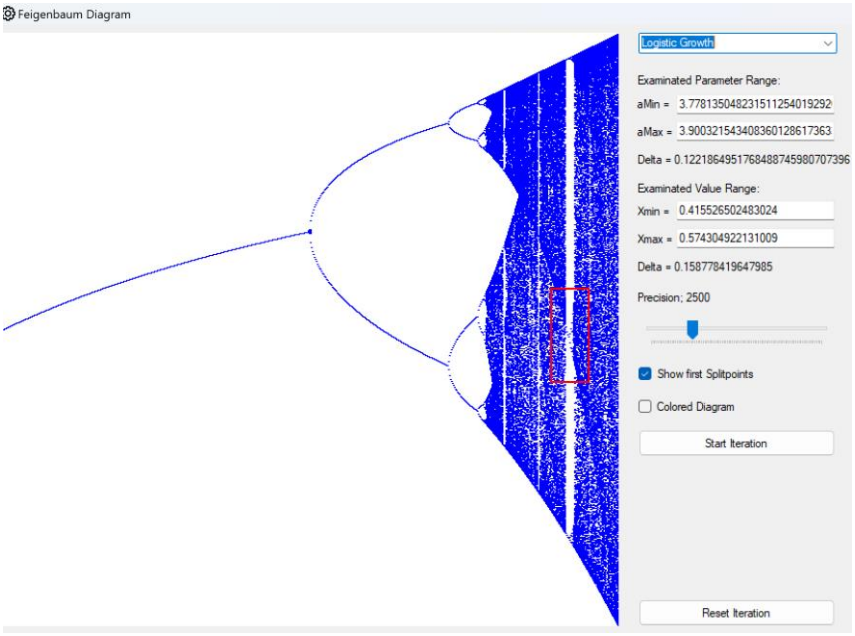
$$(2k+1) \cdot 2^n, k \in \mathbb{N}, n \in \mathbb{N}_0$$

3.11. Implementation in the "Simulator"

The *FrmFeigenbaum* supports the investigation of any cycles depending on a parameter *a* for the iteration. Here too, the *FrmFeigenbaum* uses the methods and properties of the *IIterator* interface. Classes that implement this interface then contain the specifics of the respective iteration.

The user can manually enter the parameter range in which *a* moves. He can also enter a value range for the x-values, which defines the range in which the x-values are considered. This leads to a scaling of the x-axis.

Both can also be done by selecting with the left mouse button pressed. This allows you to examine interesting areas in the Feigenbaum diagram more closely. A detailed description can be found in the "Manual".



Extract from the Feigenbaum diagram with a selection of the user

The data of the red selection rectangle is displayed in the "Simulator". If you place the red rectangle, which determines the section of the diagram, on the split points a_i you can see the difference $\Delta a_i := a_{i+1} - a_i$. You can then calculate an approximate value of the Feigenbaum constant:

$$\delta \approx \frac{\Delta a_i}{\Delta a_{i+1}}$$

You can also see the difference between two consecutive cycle points: $\Delta x_i := \xi_{i+1} - \xi_i$ if you place the rectangle on them. This provides an approximation for the scaling factor in the x-direction:

$$\alpha \approx \frac{\Delta x_i}{\Delta x_{i+1}}$$

3.12. Exercise Examples

1. The function is given:

$$f(x) = ax\sqrt{1-x^2},\ \ I \to I, a \in P$$

Where the iteration range *I* and the parameter range *P* are finite real intervals.

a) How should *I* be chosen so that the function is unimodal and can be used as an iteration rule?

b) In what range may the parameter *a* then lie?

c) Determine the fixed points and 2-cycles of the corresponding iteration. For which values of *a* are these attractive or repulsive?

d) Extend the "Simulator" with a class *ClsRoot*, which defines the above iteration and implements the interface *IIterator*. Experiment with the "Simulator".

e) Show that the iteration for a certain *a* is conjugate to the tent map (use trigonometric functions).

2. Examine the sawtooth image:

$$f(x) = \begin{cases} 2x, x \in [0, 0.5[\\ 2x - 1, x \in [0.5, 1] \end{cases}$$

Using the dual fraction representation of x, show that

a) the periodic points lie dense in the interval [0, 1] and that they are all repulsive.

b) the mapping is sensitive

c) the mapping is transitive

d) If you define a protocol: "0" if $x \in [0, 0.5[$ and "1" if $x \in [0.5, 1]$, there is a start value for each specified protocol that provides this protocol.

e) Extend the "Simulator" with a class *ClsSowTooth*, which defines the above iteration and implements the interface *IIterator*. Experiment with the "Simulator".

f) The parabola is for the parameter $a = 2$ is a conjugate of the tent map and the corresponding transformation is $T(u) = \cos(\pi(1 - u)), u \in [0,1]$. Show: The parabola is for

$a = 2$ also a conjugate of the sawtooth mapping and the corresponding transformation:

$$T(u) = \cos(\pi u), u \in [0,1]$$

3. Examine the angle doubling on the unit circle with regard to chaotic behaviour. To do this, use either the mapping $\varphi \to e^{i\varphi}$ in the complex plane or $\varphi \to \binom{cos\varphi}{sin\varphi}$ in the $\mathbb{R}^2$.

4. Examine the function $f(x) = ax^2(1 - x^2), [0,1] \to [0,1]$.

a) In which interval may the parameter a lie?

b) Determine fixed points and 2-cycles

c) For which a are these attractive? Repulsive?

d) Extend the "Simulator" with a class that implements the *IIteration* interface and represents this function.

5. Examine the Feigenbaum diagram with the "Simulator". Select the red selection triangle so that the parameter differences of consecutive split points are hit. These differences are then displayed on the right. Use this to determine approximate values for the Feigenbaum constant.

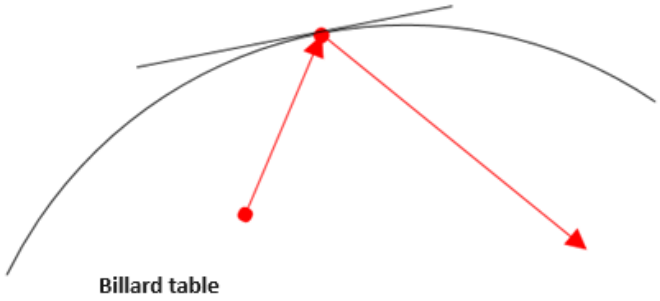
Further examples of exercises can be found in [1] and [2].

4. Mathematical Billiards

Preliminary remark: This chapter is independent of the others and only refers to chapter two.

4.1. Introduction

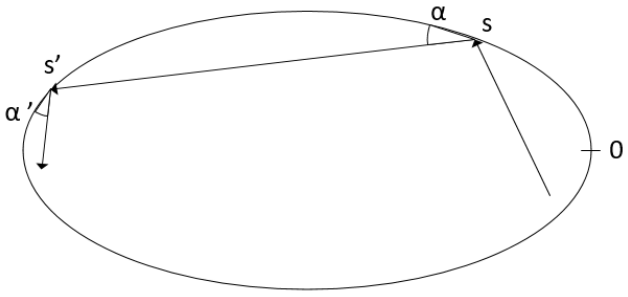
In this paper, we consider a convex region of the plane as a billiard table. An idealized (massless and point-shaped) billiard ball moves on the billiard table without friction and in a straight line until it hits the edge of the billiard table. Here it is reflected according to the law of reflection: angle of incident = angle of reflection. The angle of incidence is defined as the angle between the ball path before the impact and the tangent to the edge curve at the impact point. Similarly, the angle of reflection is the angle between this tangent at the point of impact and the path of the ball after the impact.



Reflection of the billiard ball (red) on the edge of the billiard table

In particular, we will examine the elliptical, the oval and the stadium-shaped billiard table. These table shapes are implemented in the "Simulator" program and any number of balls can be started. You can access these functions via the menu item "Mechanics - Billiards".

To parameterize this form of billiards, a zero point is often defined on the edge of the billiard table. The position of an impact point is then defined by the arc length *s* of the edge curve between this impact point and the zero point. The second parameter is the angle α between the spherical path and the tangent at the point of impact. Together with the law of reflection, this defines the law of motion for billiards.



Parameterization of the path of a billiard ball

It is a mapping which shows a point of impact *s* and an angle of deflection α the next point of impact *s'* and the next angle of deflection α' is assigned:

$$(s, \alpha) \rightarrow (s', \alpha')$$

Where *s is* a positive real number and α is an angle between 0 and π.

We will make some general considerations about billiards later. First, let's turn to specific examples.

4.2. Basics of elliptical Billiards

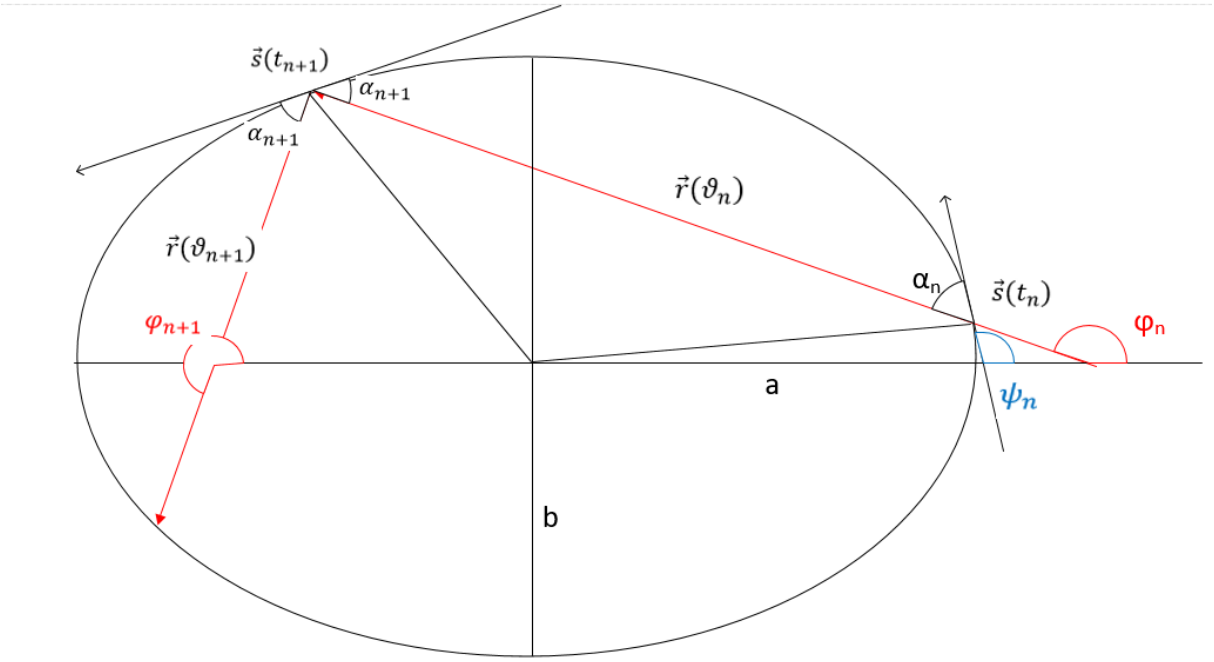
In the case of an elliptical billiard table, the parameterization by the arc length on the edge curve is not suitable for programming. Instead, we use the usual parameter representation of the ellipse.

An ellipse with main axes a and b is parameterized by:

$$\vec{s}(t) = \begin{pmatrix} acost \\ bsint \end{pmatrix}, t \in [0,2\pi[, a, b \in \mathbb{R}^+$$

Now consider a ball that rolls frictionless on the elliptical table and is reflected frictionless from the edge of the table.

Note: From an optical point of view, you could look at a beam of light reflected by an elliptical mirror instead of the spherical path. This provides the same scenario.



Elliptical billiard

After the n[th] push, the ball starts at a point

$$\vec{s}(t_n) = \begin{pmatrix} acost_n \\ bsint_n \end{pmatrix}$$

In one direction, which is defined by the angle of α_n (see sketch above).

Let ψ_n is the angle between the tangent at the point of impact $\vec{s}(t_n)$ and the positive x-axis. With φ_n denotes the angle of the ball trajectory after the n[th] impact and the positive x-axis. Then applies:

$$\varphi_n = \psi_n + \alpha_n$$

The ball moves in a direction:

$$\vec{r}(\vartheta_n) = \begin{pmatrix} acos\vartheta_n \\ bsin\vartheta_n \end{pmatrix}, \vartheta_n \in [0,2\pi[$$

Note: The parameter ϑ is not identical to the angle φ!

Since $\begin{pmatrix} acos\vartheta \\ bsin\vartheta \end{pmatrix}$ should be parallel to $\begin{pmatrix} cos\varphi \\ sin\varphi \end{pmatrix}$ is to be parallel, the following applies:

$$(1):\ \vartheta = \begin{cases} \arctan\left(\frac{a}{b} tan\varphi\right), \varphi \in [0, {}^{\pi}/_{2}\,[\\ \pi + \arctan\left(\frac{a}{b}\ tan\varphi\right), \varphi \in]\pi/2, 3\pi/2[\\ 2\pi + \arctan\left(\frac{a}{b} tan\varphi\right), \varphi \in]3\pi/2, 2\pi[\\ \varphi, \varphi \in \{{}^{\pi}/_{2}, {}^{3\pi}/_{2}\} \end{cases}$$

The branch of the arctan function is used according to the arctan function available in the program:

$$\arctan(x) : \mathbb{R} \rightarrow]-{}^{\pi}/_{2}, +{}^{\pi}/_{2}\,[$$

In the "Simulator", this conversion from φ to ϑ is handled in the class *ClsMathHelperBillard* and the interval in which φ is calculated with the sign of $cos\varphi$ and $sin\varphi$ is determined.

The ball now moves along the straight line g:

$$\vec{x}(u) = \vec{s}(t_n) + u \cdot \vec{r}(\vartheta_n), u \in \mathbb{R}$$

On the next kick, the ball hits the spot:

$$\vec{s}(t_{n+1}) = \begin{pmatrix} acost_{n+1} \\ bsint_{n+1} \end{pmatrix}$$

This point is the intersection between the straight-line g and the ellipse. We use the coordinate form of the ellipse equation:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

And insert the coordinates of a point on the line. This results in the condition for the intersection point:

$$\frac{(acost_n + u \cdot acos\vartheta_n)^2}{a^2} + \frac{(bsint_n + u \cdot bsin\vartheta_n)^2}{b^2} = 1$$

After a short calculation ($u = 0$ returns the point $\vec{s}(t_n)$) the result is

$$u = -2(cost_n cos\vartheta_n + sint_n sin\vartheta_n) = -2\cos\,(t_n - \vartheta_n)$$

From the equation

$$\begin{pmatrix} acost_{n+1} \\ bsint_{n+1} \end{pmatrix} = \begin{pmatrix} acost_n \\ bsint_n \end{pmatrix} + u \cdot \begin{pmatrix} acos\varphi_n \\ bsin\varphi_n \end{pmatrix}$$

We receive t_{n+1}:

$$t_{n+1} = \arccos\,(cost_n + ucos\varphi_n)$$

Due to the geometric situation (the ball starts at a boundary point of the ellipse and moves towards the interior of the ellipse), there is always another intersection point with the ellipse and therefore a solution to the equation.

In particular $cost_n + ucos\varphi_n \in [-1,1]$ and $\arccos\,(cost_n + ucos\varphi_n)$ is defined.

There are two solutions, namely $t_{n+1} \in [0,\pi]$ and $2\pi - t_{n+1} \in]\pi, 2\pi[$. Which of the two is the one we are looking for can be determined by substituting it into the second component of the above equation.

For the angle α_{n+1} we consider the ellipse tangent at the point $\vec{s}(t_{n+1})$:

$$\dot{\vec{s}}(t_{n+1}) = \begin{pmatrix} -asint_{n+1} \\ bcost_{n+1} \end{pmatrix}$$

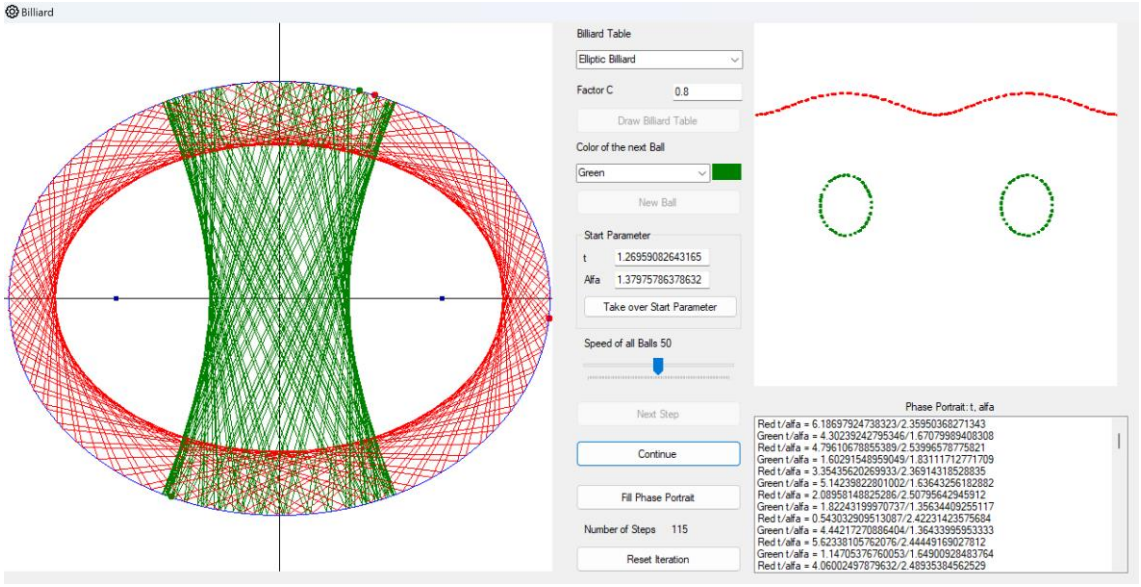
If it forms the angle with the positive x-axis, ψ_{n+1} then the following applies:

$$\alpha_{n+1} = \psi_{n+1} - \varphi_n$$

This provides a map for elliptical billiards:

$$f\colon (t_n, \alpha_n) \to (t_{n+1}, \alpha_{n+1}), [0,2\pi[\ x\ ]0,\pi[\to [0,2\pi[\ x\ ]0,\pi[$$

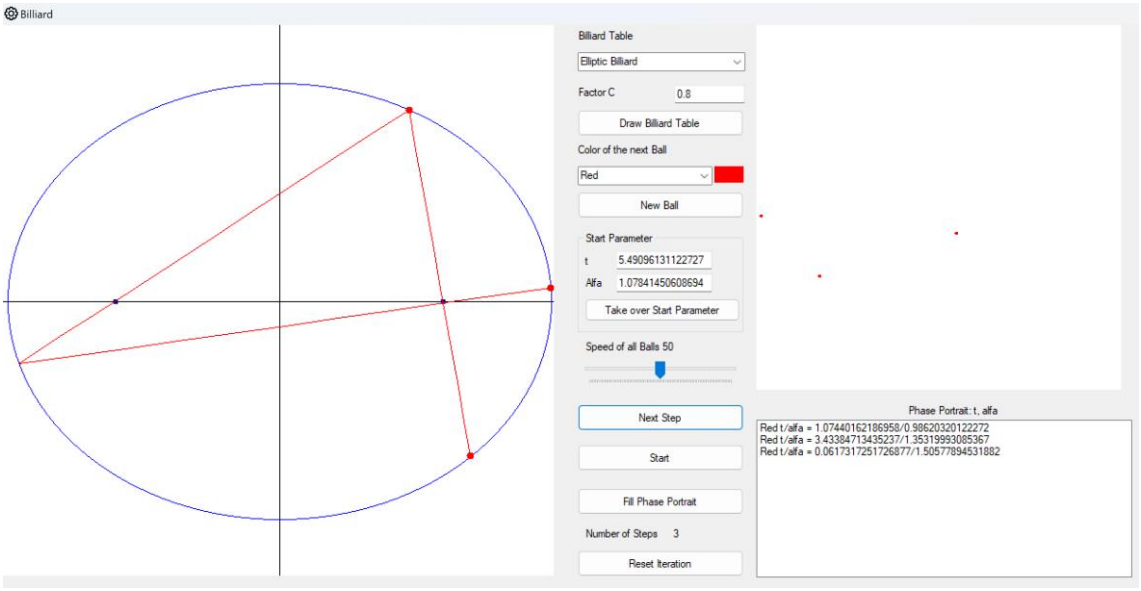
This image is iterated in elliptical billiards.



Left: Elliptical billiard table of the "Simulator". Right: The associated phase portrait

You can see this in the picture above: If the starting angle is chosen to be flat (red at the top), then the ball orbit runs around the focal points of the ellipse. In the phase portrait, the parameter *t* is plotted on the horizontal axis and the reflection angle α is plotted vertically. The image of the red orbit is a wavy line in the phase portrait. If the starting angle is steep (green above), then the ball runs up and down between the focal points. The corresponding image in the phase portrait is elliptical.

We know from optics that rays emanating from one focal point are bundled at the other focal point. A test with the "Simulator" shows similar behaviour:



A ball starts at the bottom right and the path runs through the focal point on the right at the start

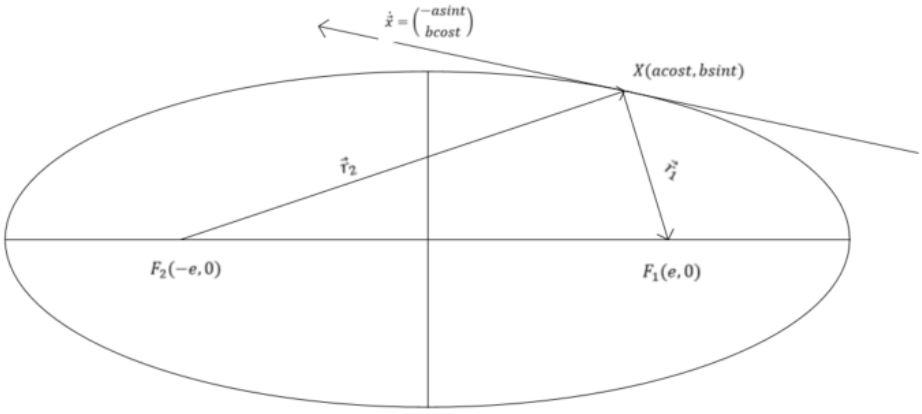
There are elegant elementary geometric proofs for the assertion that a ball that passes through one focal point passes through the other focal point after the collision on the border. Here we will give a proof using vector geometry.

The focal points of an ellipse with major axis a and minor axis b have the coordinates $(\pm e, 0)$ where $e^2 = a^2 - b^2$. Let t again be the parameter of the parameter representation of the ellipse defined at the beginning.

In preparation, we state:

$$e \cdot cost \leq e < \sqrt{e^2 + b^2} = a$$

Therefore $a - e \cdot cost > 0$



Focal points of the ellipse

For the absolute value of the vectors $\vec{r}_{1,2} = \begin{pmatrix} \pm acost + e \\ \pm bsint \end{pmatrix}$ applies:

$$\left|\vec{r}_{1,2}\right|^2 = a^2 cos^2 t + b^2 sin^2 t + e^2 \pm 2eacost$$

$$= a^2(1 + cos^2 t) + b^2(sin^2 t - 1) \pm 2eacost$$

$$= a^2 + a^2 cos^2 t - b^2 cos^2 t \pm 2eacost$$

$$= a^2 + e^2 \pm 2eacost = (a \pm ecost)^2$$

Because $a - e \cdot cost > 0$ follows: $\left|\vec{r}_{1,2}\right| = a \pm ecost$

Now we calculate:

$$\vec{r}_{1,2} \cdot \dot{\vec{x}} = -easint \mp a^2costsint \pm b^2sintcost = -easint \mp e^2sintcost = -esint(a \pm ecost)$$

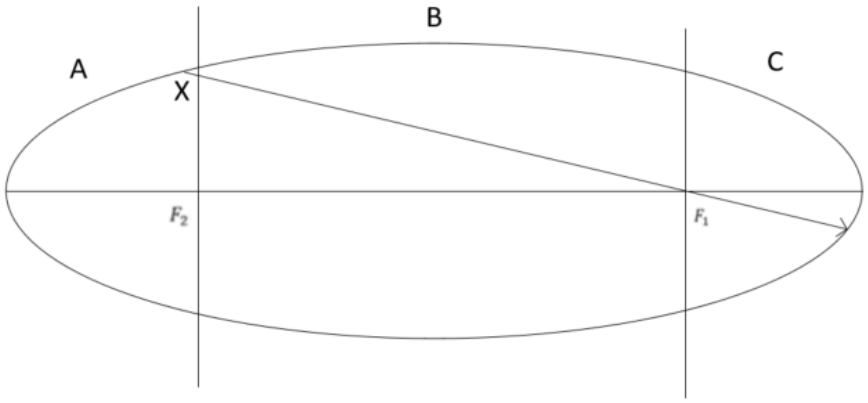
The following applies:

$$cos\alpha_{1,2} = \frac{\vec{r}_{1,2} \cdot \dot{\vec{x}}}{\left|\vec{r}_{1,2}\right|\left|\dot{\vec{x}}\right|} = \frac{-esint}{\left|\dot{\vec{x}}\right|}$$

If $\alpha_{1,2}$ is the angle between $\vec{r}_{1,2}$ and the tangent $\dot{\vec{x}}$. However, the right-hand side is independent of the sign or whether you use $\vec{r}_1$ or $\vec{r}_2$ is considered and therefore $\alpha_1 = \alpha_2$.

□

Let's look again at the case of an orbit that passes through a focal point. To do this, we divide the area of the ellipse into three parts:



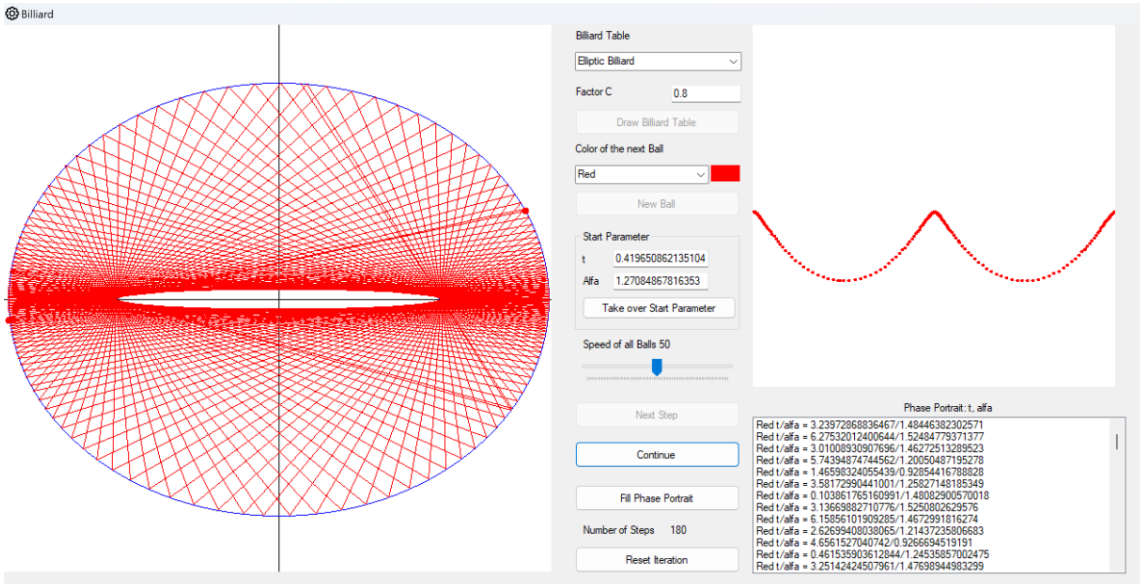
Elliptical plane divided into the vertical strips A, B, C

Case 1: The ball starts in area A at point X and then passes through the more distant focal point F_1. After the impact, it then passes through the focal point F_2 and the next impact point X'' is closer to the x-axis than X. With each further impact, the ball oscillates between the areas A and C and each time the impact point moves closer to the x-axis. Without going into the details of the proof: The path approaches the 2-cycle along the main axis of the ellipse, but only until it forms a caustic that no longer intersects it.

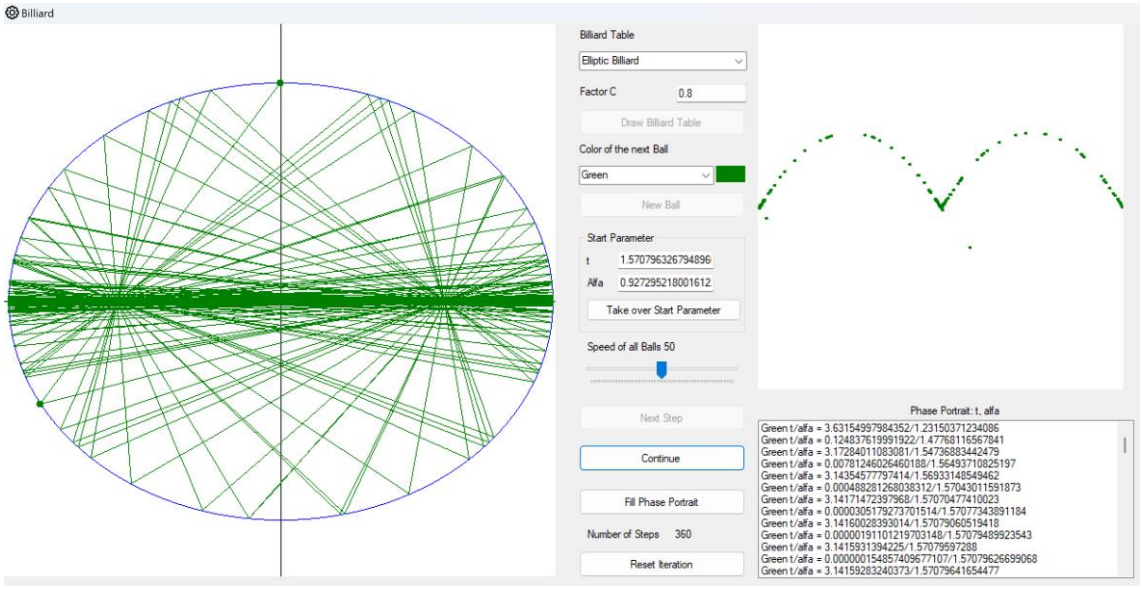
Case 2: The ball starts in area B and passes through the focal point without passing through the focal point F_2. Then we have case 1 again.

Case 3: The ball starts in area A and passes through the closer focal point F_2. Then it lies in area B after the impact, and we have case 2.

Conclusion: Orbits through the focal points come arbitrarily close to the main axis of the ellipse if they really do pass precisely through the focal points. If you do not hit the focal point precisely, the path will always swing close to the main axis, but will not come arbitrarily close to it, but will move away from it again in between. An ellipse (the so-called caustic) is left out.



The ball started in the first quadrant and just missed the left focal point



Relatively precise start at point (0, b) with explicitly calculated start parameters t and α

In the experiment above, we calculate the start parameters as precisely as possible. If we start at the point (0, b), then $t = 1.570796326794896619231321691$ (the decimal data type supports 28 decimal places). Since the axis ratio in the example above is 0.8, a = 1 and b = 0.8, so e = 0.6 and the start angle is $\alpha = \arctan\left(\frac{4}{3}\right) = 0.927295218001612232428512 4629$. We enter these values manually in the start parameters, generate a new ball and then click "Take over Start Parameter". The result is a reasonably precise image of a ball that initially oscillates along the main axis, very close to it. Over time, however, the initial error in precision becomes noticeable and the ball starts to move away from the main axis again, before oscillating close to the main axis again for a while.

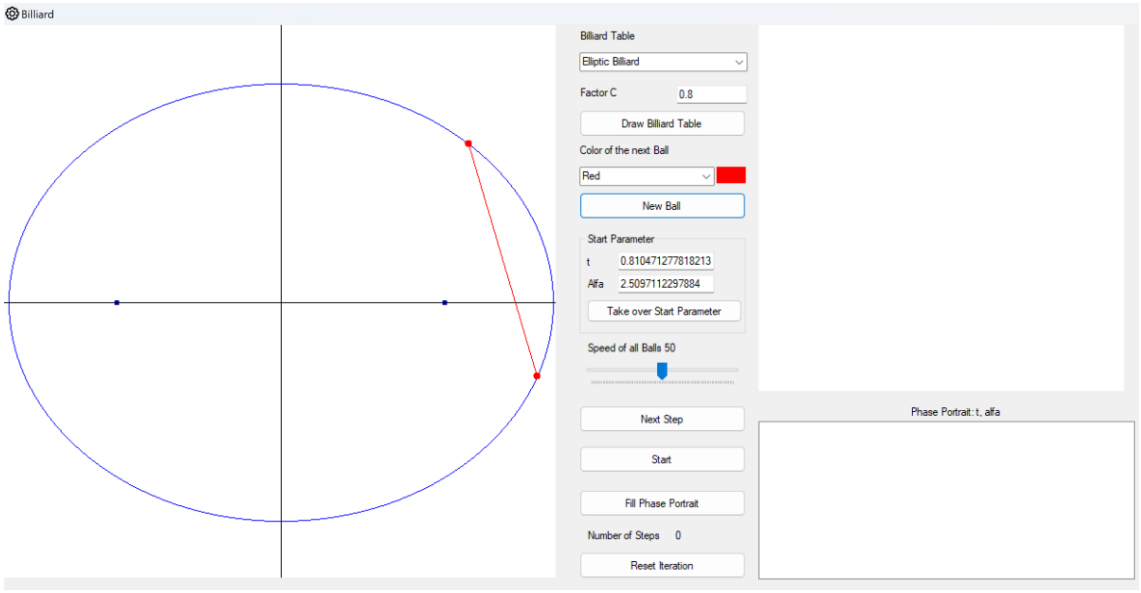
4.3. Implementation of elliptical Billiards in the "Simulator"

As angles between a vector and the positive x-axis always must be determined, this is supported by the *CalculateAngleOfDirection (DeltaX, DeltaY)* function in the *ClsMathHelperBilliard* class. The

coordinates of the vector are transferred. The return value is an angle in $[0, 2\pi[$. The function is used to determine the angles φ, ψ, ϑ and the parameter *t*. It is defined by the formula marked (1) in the previous section.

The interface between the *FrmBilliardtable* and the movement logic is provided by the interface: *IBilliardball*.

IBilliardball manages the properties of the billiard ball (e.g. colour, size, etc.). It provides methods for the movement of the ball on the table. There are also methods that enable the initial placement of the billiard ball and the direction of the first shot. The user can position these parameters with the mouse.



Positioning of a ball in *FrmBilliardTable*

Once you have defined the colour for the new ball (red at the top), you can click on "New Ball". The ball then appears by default at point (0, b). By holding down the left mouse button, you can now select the start position (slightly to the right of (0, b) at the top). If the mouse button is released, the start position is fixed. If you press the left mouse button a second time (a hand appears each time), you can select the start direction (top down to the right). If you release the mouse button, the starting direction of the ball is also fixed. The start parameter t and the start angle α are then displayed in the "Start Parameter" area.

Now you can create and place the next ball with the desired colour. Thanks to the colours, you can distinguish the balls and their paths.

Sometimes you want to calculate and specify the start parameter t and start angle α precisely and specify them. In this case, you can enter these values in the "Start Parameter" area after the ball has been created and click on "Take over Start Parameter". The start position and start angle of the ball are then adjusted accordingly and the ball is positioned.

For elliptical billiards, there is the class *ClsEllipseBilliardball*, which implements the *IBilliardball* interface.

The reader thus has the possibility to implement further forms of billiards with little effort. He must program one class for the billiard table and one for the billiard ball, which implement the interfaces mentioned.

The ball class depending on the type of billiard (for elliptical billiards the class *ClsEllipseBilliardball)* contains the general logic for the ball movement. The ball path is continuously drawn in a bitmap, which is the image of the PicDiagram picture box. The ball itself is drawn in PicDiagram. By refreshing the Picture Box, only the current position of the ball is visible, while the bitmap including the ball path is visible at the current position. To support this, the ball requires the reference of PicDiagram and the corresponding bitmap in its constructor. Since the ball works in mathematical coordinates and only the *ClsGraphicTool* class provides the conversion to pixel units, the latter also require the specification of the mathematical value ranges for x and y. This is the standard square [-1,1] x [-1,1]. It is also passed to the ball in the constructor.

The ball also supports its positioning for the start as well as the initial direction for the first push, which is carried out using the mouse. The start parameters can also be entered manually.

When the iteration is started, the *ClsEllipseBilliardball* moves independently within the ellipse according to the mathematical algorithms in the previous section. From the last point of impact and the current direction of impact, the ball first calculates the next point of impact. It then calculates the tangent angle at the next point of impact and from this the direction for the next impact.

From the equations

$$\varphi_{n+1} = \psi_{n+1} + \alpha_{n+1}$$

And

$$\alpha_{n+1} = \psi_{n+1} - \varphi_n$$

Results directly for the direction of the ball after the nth impact:

$$\varphi_{n+1} = 2\psi_{n+1} - \varphi_n$$

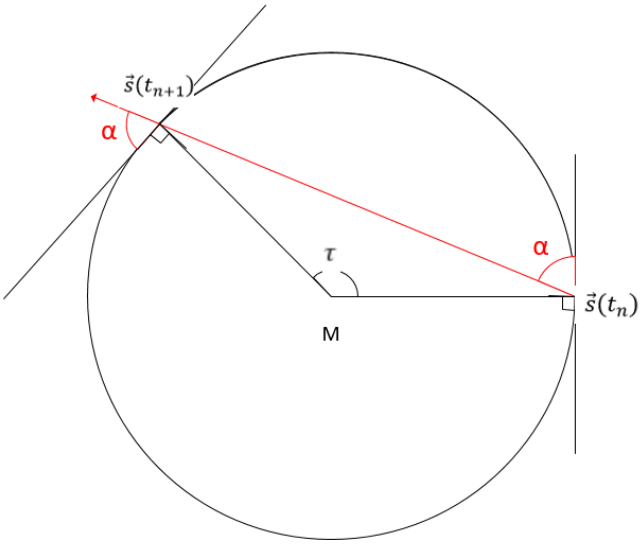
The "Simulator" works directly with this formula but logs the angle α_n together with the parameter t_n in the phase portrait.

Since the ball itself knows the rules for its movement, this makes it possible to instantiate several balls and run their movement in parallel. The balls are differentiated by their colours and five different colours can be selected. The ball speed can be changed and affects all balls.

At the same time, the ball writes the parameters (t_n, α_n) in a phase diagram on the right-hand side of the window and in a list box.

4.4. Billiards in a Circle

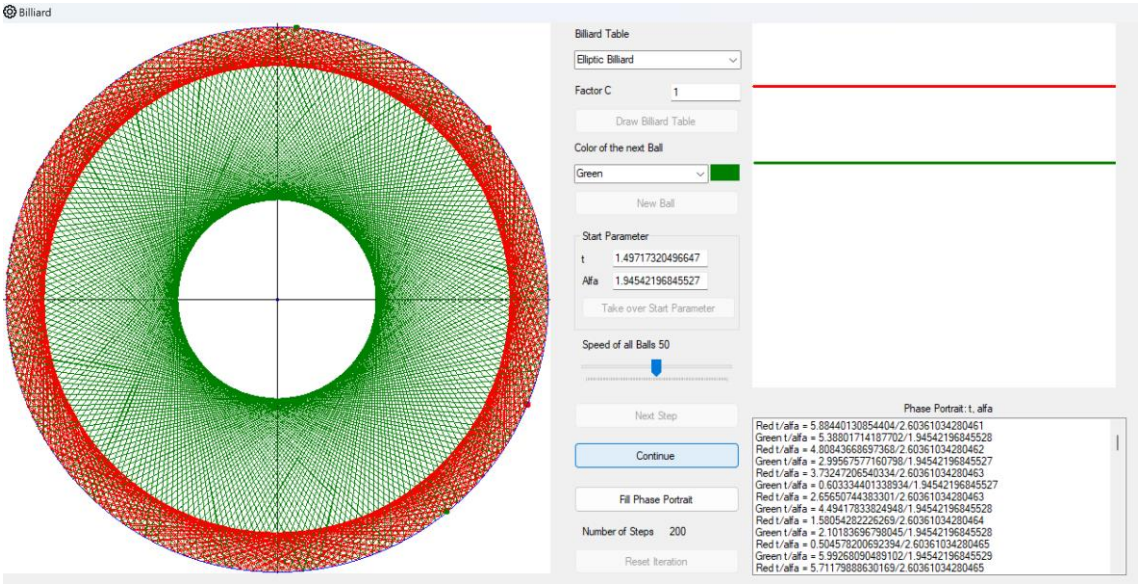
Before we examine the elliptical billiard, it is worth looking at the special case a = b, i.e. the billiard in a circle.



Billiards in a circle: The angle α is constant. Furthermore $\tau = 2\alpha$

The triangle with the two hit points and the centre point M is equilateral, as the distance M from each butt joint is equal to the radius r. The base angles of this triangle are therefore equal. α is the complement of each base angle to $\pi/2$ therefore also equal.

Here is an image generated by the "Simulator" for two different balls: red with a flat reflection angle and green with a steep one:



Billiards in a circle

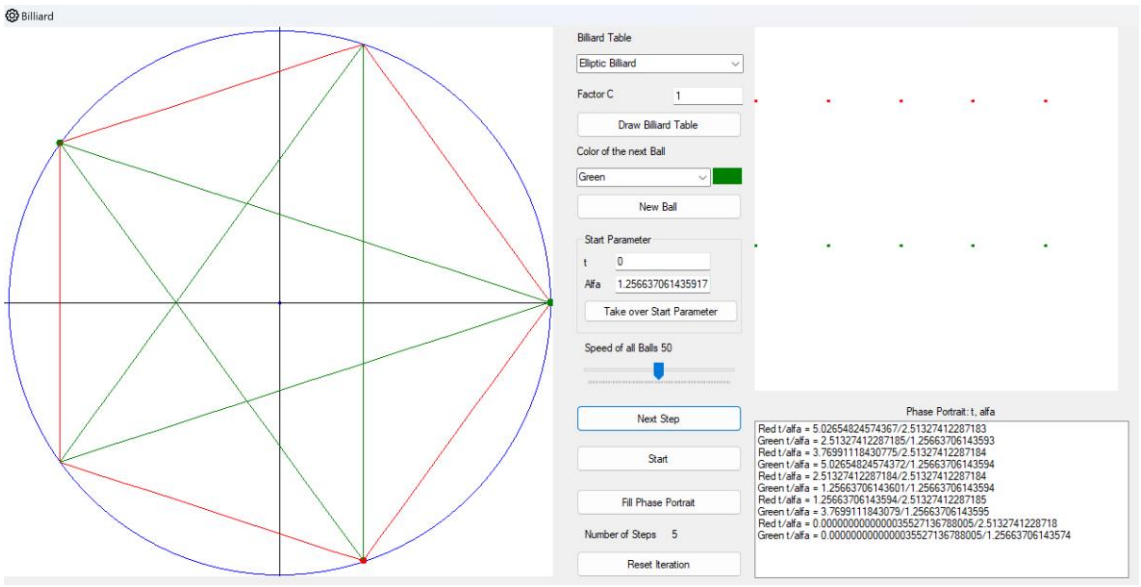
The angle α is constant here. The parameter t increases for the next impact point by the same constant value $\tau = 2\alpha$ increases. The image of the iteration in the phase portrait is therefore a straight line parallel to the t-axis.

For circular billiards, τ and the parameter t for the impact point are identical. The following applies to the angle φ between the direction of the ball path and the positive x-axis:

$$\varphi = \alpha + \pi/2 = \frac{\tau + \pi}{2}$$

Now let the direction vector φ of the path start be a rational multiple of 2π. Then τ is also a rational multiple of 2π. Let's assume that it is $\frac{p}{q}$ is a reduced fraction and $\tau = 2\pi \cdot \frac{p}{q}$. We only consider the case $p < q$ because otherwise the integer part of the fraction only results in a rotation by a multiple of 2π.

The path is periodic if τ is a multiple of 2π after *n* collisions. This is the case after *q collisions*, so q is the period of the trajectory. Then the direction angle has grown to the value $p \cdot 2\pi$. If $p < {}^{q}/_{2}$, is p the number of windings of the path around the centre of the circle in a negative clockwise direction. If $p > {}^{q}/_{2}$ then $q - p$ is the number of windings of the path around the centre of the circle in a positive clockwise direction.



Trajectories with period q = 5 and winding numbers p = 1 (red) or p = 2 (green)

If the start angle φ of the path is an irrational multiple of 2π, then the centre angle τ is also an irrational multiple of 2π. Then the orbit will never close and the orbit is aperiodic. The points of such an orbit lie close to the edge of the circle. The reason for this is provided by the following consideration:

Be ϑ_1 is the angle of rotation of the path between two successive impact points. If we consider the unit circle, this is just the arc length that lies between two impact points. Starting from a starting point x on the edge of the circle, we then perform as many pushes as possible until the path comes close to point x again for the first time. Let it be the case after n rotations. It is essential for the reasoning that the path of x will never hit point x itself exactly, because otherwise it would be a rational multiple of 2π.

We denote n rotations around the angle ϑ_1 with $D_{\vartheta_1}^n$. Then x lies between the points:

$$D_{\vartheta_1}^{n-1}(x) < x < D_{\vartheta_1}^n(x)$$

Either x is then closer to the point $D_{\vartheta_1}^{n-1}(x)$ or otherwise closer to the point $D_{\vartheta_1}^n(x)$. W.l.o.g. we assume that x lies closer to $D_{\vartheta_1}^n(x)$. This distance is then smaller than $\frac{\vartheta_1}{2}$. Thus applies:

$D_{\vartheta_1}^n =: D_{\vartheta_2}$. We therefore replace $D_{\vartheta_1}^n$ by a rotation D_{ϑ_2} with $\vartheta_2 < \frac{\vartheta_1}{2}$. Now we can apply the same argument to a multiple of the rotation D_{ϑ_2} . So, we substitute for a certain m:

$$D_{\vartheta_1}^m =: D_{\vartheta_2}$$

If we apply the same argument to D_{ϑ_2} then there is a rotation D_{ϑ_3} with $\vartheta_3 < \frac{\vartheta_2}{2}$ which is a multiple of the rotation D_{ϑ_2} . We repeat the same argument enough times until we get an arbitrarily small rotation angle $\vartheta_k < \frac{\vartheta_1}{2^{k-1}}$ where the rotation by this angle is a multiple of D_{ϑ_1}, therefore $D_{\vartheta_k} = D_{\vartheta_1}^r$ for a sufficiently large $r \in \mathbb{N}$. If we apply this rotation D_{ϑ_k} to x, the path will hit any small interval.

A tempting question is *whether the elliptical billiard could be understood as an affine image of the circular billiard*. This following on from the first chapter, where we gained a lot of clarity about logistic growth by being able to represent this as a conjugate of the tent map. The tent map was easy to investigate.

So, if $g\colon (s,\beta) \mapsto (s',\beta')$ is the circular billiard with certain starting parameters, then the elliptical billiard should be $f\colon (t,\alpha) \longmapsto (t',\alpha')$ should be representable as its conjugate in the form

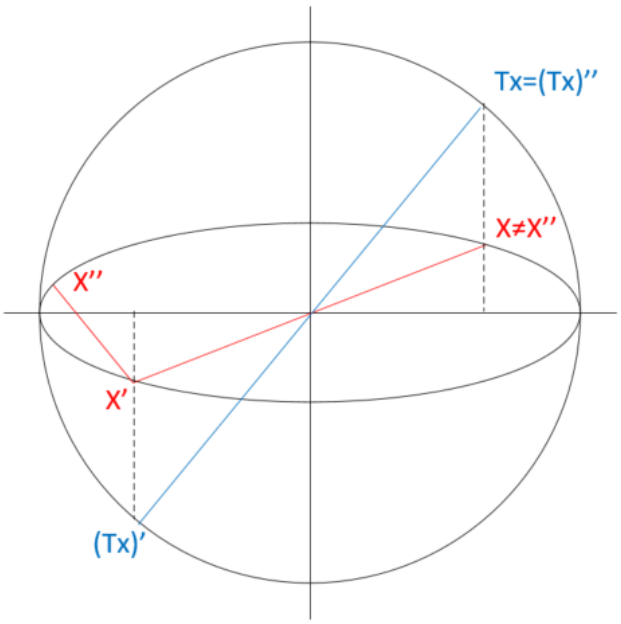
$$f(t,\alpha) = T^{-1} \circ g \circ T(t,\alpha) = T^{-1} \circ g(s,\beta) = T^{-1}(s',\beta') = (t',\alpha')$$

Where T is the elongation of the plane in the y-direction by the factor a/b factor. An ellipse with axes a and b is then transformed into a circle with radius a. For a point $\vec{x}(t)$ on the ellipse then applies:

$$T\colon \vec{x}(t) = \begin{pmatrix} acost \\ bsint \end{pmatrix} \longmapsto \vec{s}(t) = \begin{pmatrix} acost \\ asint \end{pmatrix}$$

We would therefore first map the starting point and the starting angle of the ball through T to the corresponding starting point of a ball in a circle. Then a push is performed in the circle, which leads to the next push point, whereby the start angle of the circle remains constant during the push. We then transform everything back to the ellipse and hope that we get the same result as if we had performed the push directly in the ellipse according to the law of reflection.

Can this work? For example, the focal points of the ellipse remain fixed in the figure *T* but lose their meaning in the circle. We are therefore looking for a counterexample that shows that the proposed conjugation does not work in this way. Let's look at the following figure:



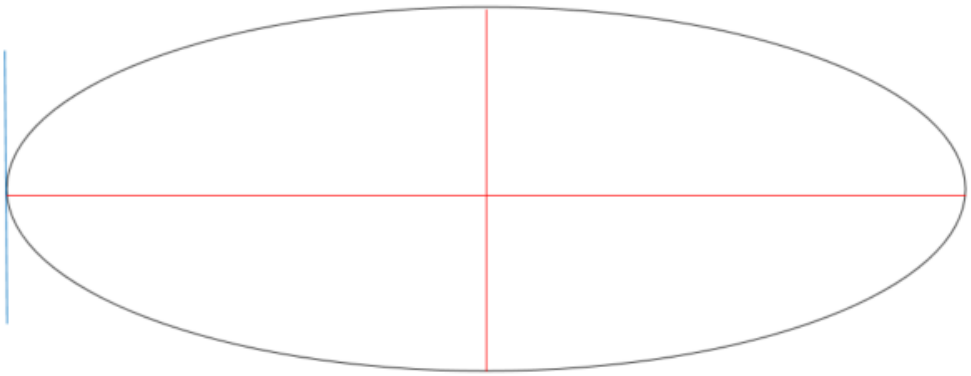
Affine mapping between ellipse and circle

We consider the point X. In billiards, it will land at X'' after two shots. The image of X under the affine mapping T is TX. This point will land at itself again after two shots and not at TX''.

4.5. Periodic Points in elliptical Billiards

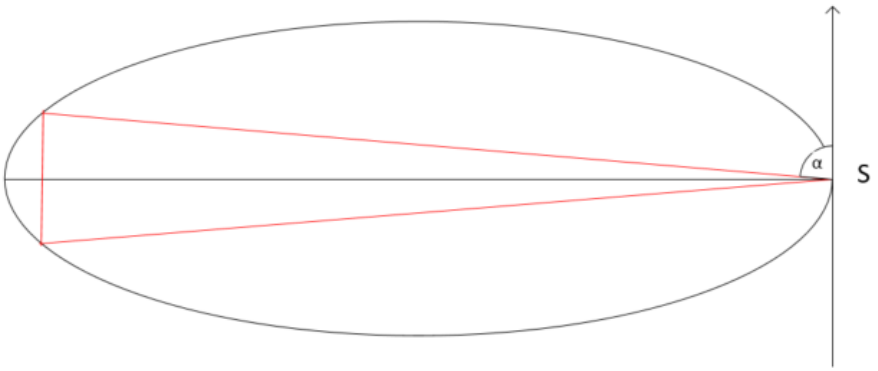
Any point on the edge of the ellipse whose starting direction is just the direction of the ellipse tangent can be regarded as a (degenerate) fixed point.

For a 2-periodic point, the direction of the incident ball-path must be rotated by the angle π so that the outgoing beam hits the original starting point again. This means that the direction of incidence is perpendicular to the tangent at the point of impact. The same applies when the ball hits the starting point again. The only possible paths are therefore



2-period tracks for elliptical billiards

Orbits of period three exist. If you look at the following sketch:



A three-period orbit

Starting from point S, α must be selected so that the ball runs vertically downwards after the first impact and arrives back at point S after the second impact. If you experiment with the "Simulator", you will see that this is possible. If α is chosen slightly too small, you end up slightly below S. If α is chosen too large, you end up above S. If α is constantly changed in between, then there must be an α such that point S is hit again.

Analytically or with elementary geometry, it seems difficult to find a solution to the problem. A vector geometric approach leads to a complicated equation for the required α.

But we can try to find at least an approximate solution with the "Simulator". To do this, we start with a slightly too small α_1 and then select the next α_2 a little too large. The α we are looking for must therefore lie somewhere in between for reasons of continuity. We now use the following algorithm:

Let's assume we have two angles α_{n-1} and $\alpha_n$where α_{n-1} is too small (i.e. we do not quite reach point S after two joints) and α_n is too large. Then we find the next pair of angles as follows:

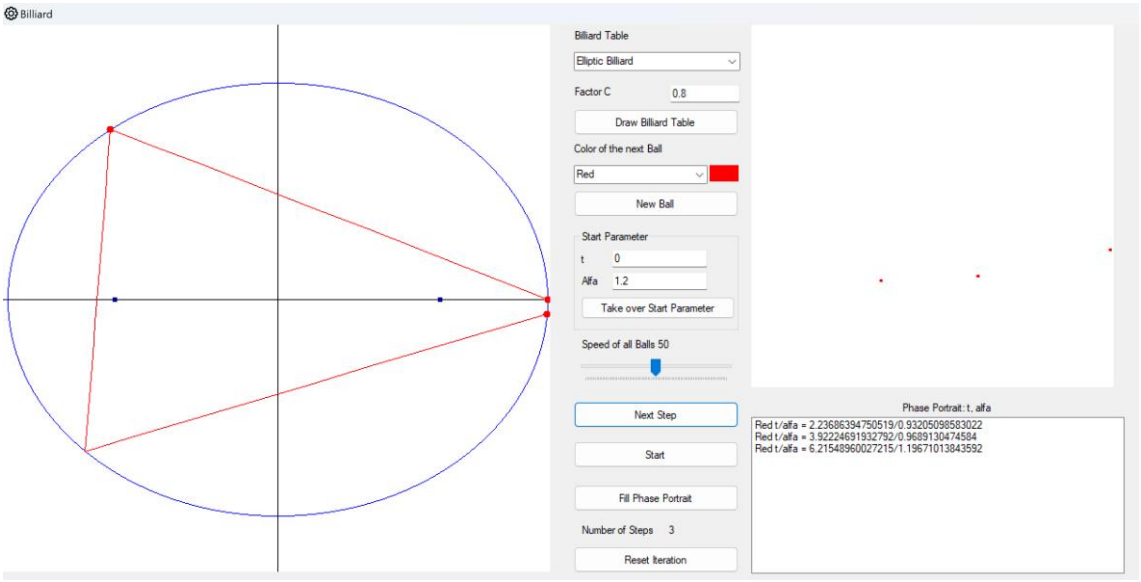
$$\delta := \frac{\alpha_{n-1} + \alpha_n}{2}$$

Then we bet:

$$\begin{cases} \alpha_{n+1} = \delta, \alpha_n = \alpha_{n-1}, if\ \delta\ is\ too\ large \\ \alpha_{n+1} = \alpha_n, \alpha_n = \delta, if\ \delta\ is\ too\ small \end{cases}$$

Again, the angle you are looking for is $\alpha \in [\alpha_n, \alpha_{n+1}]$ but this interval length has been halved in the step. If we iterate long enough, we get closer and closer to the value we are looking for.

Below is a table with some of the values found experimentally in this way for the factor $c = 0.8$:



$\alpha_1 = 1.2$ is a little too small

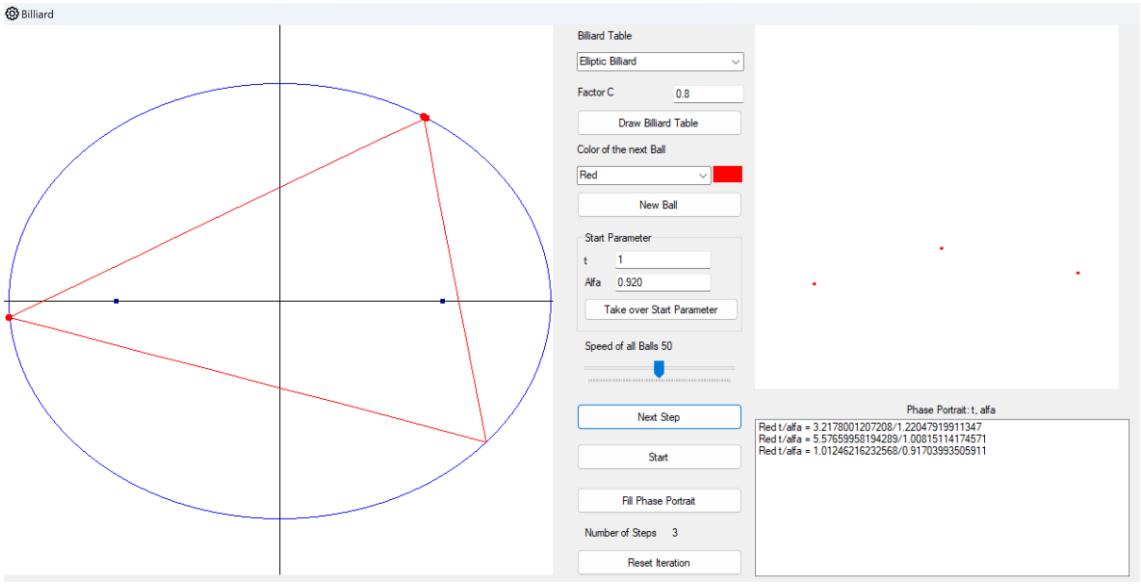
The fact that $\alpha_1 = 1.2$ is a little too small can still be seen visually here. However, you have better control on the right in the log of the parameter t. This is just below 2π. As you can also see $\alpha_2 = 1.25$ is slightly too large. We then halve this interval and see that the value 1.225 is still too large. So, we replace 1.25 with 1.225. In this way, we obtain a successive interval nesting:

α_n	α_{n+1}
1.2	1.25
1.2	1.225
1.2125	1.225
1.21875	1.225
1.221875	1.225
1.221875	1.2234375
1.22265625	1.2234375

First steps of interval nesting to find the desired start value α

This converges very slowly. But at least it is plausible that a limit value and thus a 3-cycle exists.

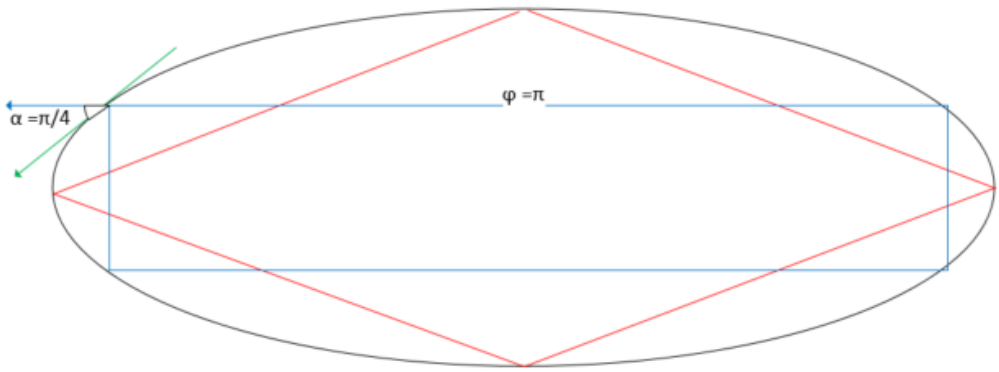
The same method can be used to find other, non-symmetrical starting positions that lead to a three-periodic path. For example, for the parameter $t = 1$ and α = 0.920 as a first approximation:



Start parameters $t = 1, \alpha = 0.920$

Further experiments show: The required α for a three-periodic trajectory with the starting point for the parameter t=1 exists and lies in the interval $[0.919, 0.920]$.

Period 4 tracks are easy to find:



Tracks of period four

Shown in red is a path that runs from one vertex to the next. It can be traversed clockwise or counterclockwise.

Another 4-periodic path is shown in blue. The impact point at angle α is easy to determine: According to section 4.1, the new direction angle φ must point vertically downwards after the impact. The direction angle of the ball before the impact is equal to π. Then the direction of the tangent must be relative to the positive x-axis $^{5\pi}/_{4}$. This provides the coordinates for the desired impact point

$$\vec{r} \approx \begin{pmatrix} -a \cdot 0.707 \\ b \cdot 3.927 \end{pmatrix}$$

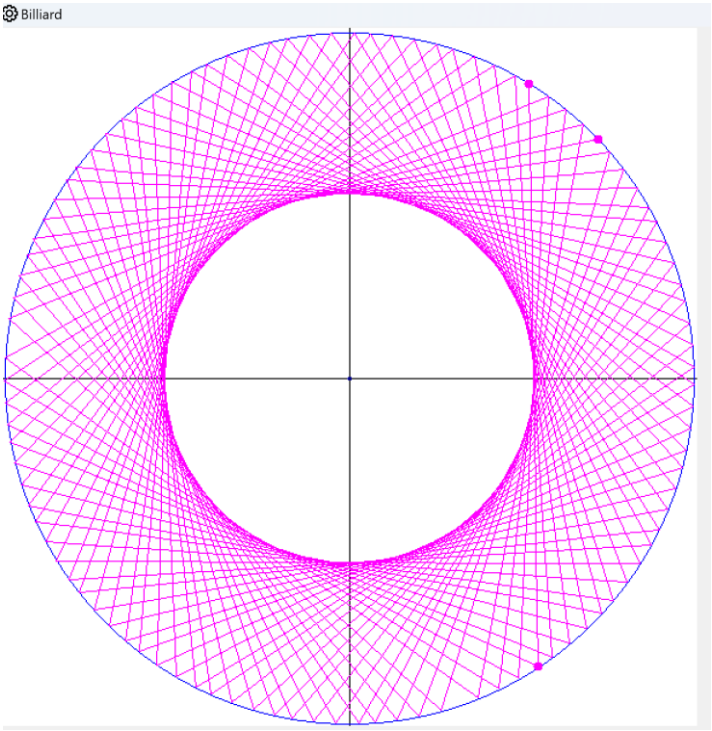
Experiments with the "Simulator" show that apparently non-symmetrical orbits of period 4 also exist, e.g. with starting point t=1. A corresponding α lies in the interval [0.721,0.722].

This method can obviously be used to find starting angles that lead to trajectories of any given period.

We will later learn a corresponding theorem for any strictly convex billiard table.

4.6. Caustics in elliptical Billiards

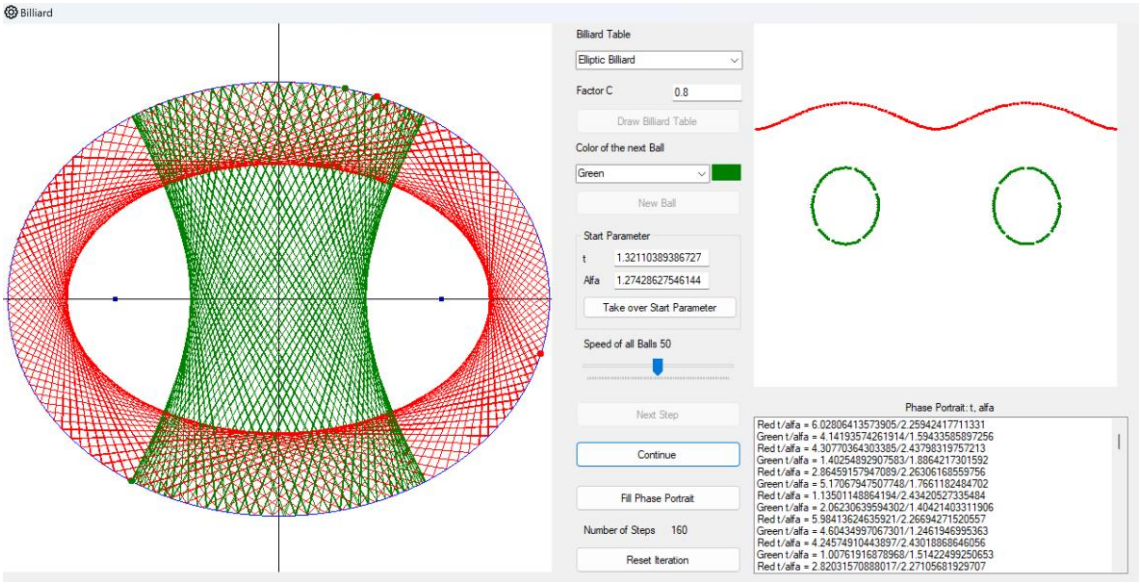
A *caustic* is *a* term from optics and describes an arc to which light rays of an optical system are tangent. In the case of billiards, the ball paths are considered instead of light rays. In the case of a circle, these are tangential to a concentric circle:



The caustic of billiards in a circle is a concentric circle

The individual sections of the spherical path are circular chords. As the angle α between the ball track and the tangent at the point of impact is constant for all track sections, all circular chords belonging to the track are of the same length. Their point of contact with the caustic is the centre of the chord. If we rotate such a circular chord, its centre describes a concentric circle with radius $cos\alpha$.

It is similar with elliptical billiards:

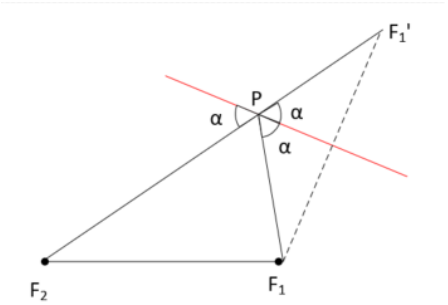


Caustics in elliptical billiards

The image above shows the paths of two different balls, one in green with the starting direction between the focal points, and one in red with the starting direction outside the focal points of the ellipse. You can see that the caustic in both cases is obviously a conic section. An ellipse in the red case and a hyperbola in the green case. These have the same focal points as the ellipse of the billiard table. The respective phase diagram is shown on the right in the corresponding colour.

We want to prove this in the case of the red orbital curve. With the help of vector geometry, this can be time-consuming. We fall back on an elementary geometric proof.

First, we must think about where a section of the path touches the caustic. We set this up so that a ball that starts at one focal point and is reflected at the path section runs to the other focal point. The law of reflection must apply.



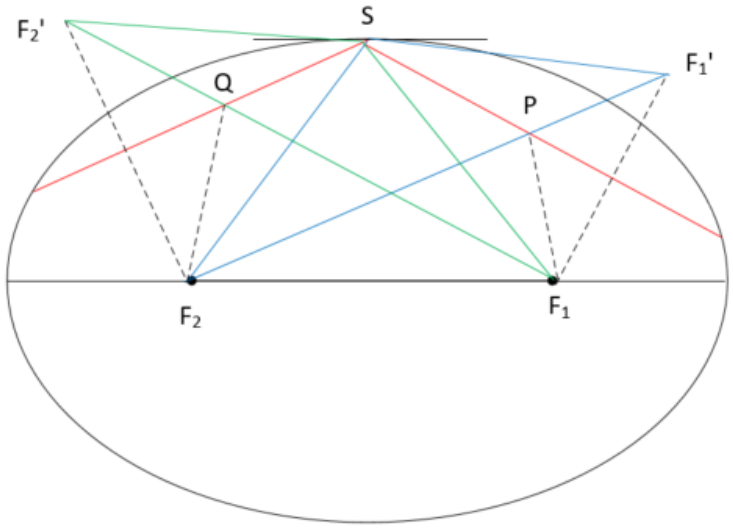
Construction of point P on the track section (red)

The section of track is shown in red above. We must select the searched meeting point P in such a way that the law of reflection applies to a focal beam. To do this, we mirror F_1 at the path section and then connect the mirrored point $F_1{}'$ with the other focal point F_2. At the intersection P with the path section, all marked angles are equal and therefore the law of refraction applies if the caustic touches the path section at the point P point. There is therefore an ellipse with focal points F_1 and F_2so that the path section is a tangent to this ellipse.

It is easy to check that we get the same point P if instead of F_1 the other focal point F_2 would have been mirrored.

Now we must show that for the following path section the next meeting point Qwhich is constructed in the same way, is on the same ellipse as P.

Let's look at the following figure:



Two consecutive path sections (red) with the impact point S and their points of contact P and Q with the searched ellipse, which also has the focal points F_1 and F_2

First, we note that the triangle marked in green $S, F_1, F_2{}'$ and the triangle marked in blue S, F_1', F_2 are congruent. This is because $|SF_2{}'| = |SF_2|$ and $|SF_1{}'| = |SF_1|$. Furthermore, the angle at the apex S is the same for every triangle because of the law of reflection on the one hand and because the angle

$\sphericalangle\ (F_1SF_2)$ is the same for both triangles. This means that the third side opposite the point S is the same length for both triangles and the following applies:

$$|F_2{}'F_1| = |F_1{}'F_2| \implies |F_2Q| + |QF_1| = |F_2P| + |PF_1|$$

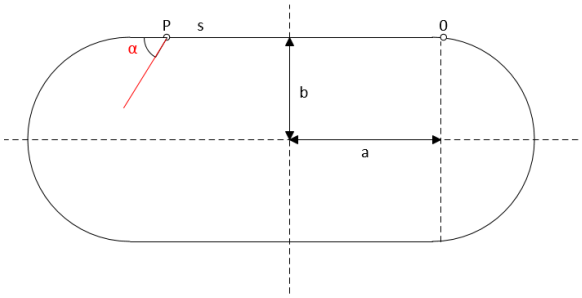
The ellipse that touches the ball track in Q is therefore identical to the ellipse that touches the ball path in P.

We will carry out an analogous consideration for the case that the path sections run between the focal points in the form of an exercise and see that in this case the caustic is a hyperbola with the focal points F_1 and F_2 appears.

In the specific case, this ellipse (or hyperbola) can be calculated by proceeding as in the construction above and calculating the respective main axis length $|F_2P| + |PF_1|$ is calculated.

Once the starting point and the first angle of reflection of a ball on the elliptical billiard table have been determined, the caustic is also defined. This means that the further orbit is only dependent on the next impact point. You simply lay a tangent from one impact point to the caustic, intersect it with the ellipse and obtain the next impact point. There are two possibilities for the tangent if the orbit runs between the focal points. The movement then only depends on the parameter t if the orbit runs outside the focal points. However, this information is already available from the plot in phase space.

4.7. Billiards in the Stadium



Stadium

The stadium is composed of a rectangle with a width $2a$ and height $2b$. On each of the vertical sides, a semicircle with a radius of b is added to each vertical side. We use a coordinate system with the zero point at the centre of the stadium and the symmetry lines as axes.

A billiard ball is reflected at the edge of the stadium according to the law of reflection.

For the parameterization, we fix a zero point on the edge, namely the point $(a,\ b)$. To describe a point P on the edge, we use the arc length s between this point and the fixed zero point on the edge. In doing so, we calculate modulo the circumference of the stadium, i.e. modulo $L = 4a + 2\pi b$. This means that each impact point of the billiard ball is uniquely defined by the parameter s. As the second parameter, we again use the angle α between the ball path and the curve tangent at the point of impact. If the ball hits a rectangular side, this tangent is just identical to the rectangular side. In the other case, the curve tangent is the circle tangent at the point of impact.

The ball movement is therefore described by an image:

$$f: [0,4a + 2\pi b[\times]0,\pi[\rightarrow [0,4a + 2\pi b[\times]0,\pi[: (s,\alpha) \rightarrow (s',\alpha')$$

These two parameters are recorded in the phase portrait for each impact.

For the implementation, we will enter for each s the coordinates (x,y) of the corresponding boundary point P for each edge point. This is done with the help of a case distinction (2):

1. $0 \leq s \leq 2a \Rightarrow P(x,y) = (a - s, b)$
2. $2a < s < 2a + b\pi \Rightarrow P(x,y) = (-a - b \cdot sin\frac{s-2a}{b}, b \cdot cos\frac{s-2a}{b})$
3. $2a + b\pi \leq s \leq 4a + b\pi \Rightarrow P(x,y) = (s - 3a - b\pi, -b)$
4. $4a + b\pi < s < 4a + 2b\pi \Rightarrow P(x,y) = (a + b \cdot sin\frac{s-4a-b\pi}{b}, -b \cdot cos\frac{s-4a-b\pi}{b})$

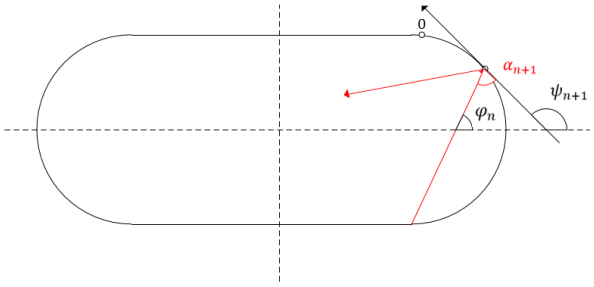
This is $s\ modulo\ L$ calculated.

We need further parameters to calculate the respective next impact point and impact angle. We use the same denominations as for elliptical billiards:

$\psi_n =$ the angle between the (directed) curve tangent at the nth point of impact and the positive x-axis

$\alpha_n =$ the reflection angle at the nth joint

$\varphi_n =$ the angle between the (directed) nth path segment and the positive x-axis



Angle at the (n+1)[th] joint

Then applies: $\alpha_{n+1} = \psi_{n+1} - \varphi_n$ and $\varphi_{n+1} = \psi_{n+1} + \alpha_{n+1}$as with elliptical billiards. It follows directly from this:

$$\varphi_{n+1} = 2\psi_{n+1} - \varphi_n$$

If we know the point of impact for the (n+1)[th] impact, we can calculate the tangent at the point of impact and determine its angle ψ_{n+1} with the positive x-axis. This immediately results in the angle φ_{n+1} that the next path section makes with the positive x-axis. For the protocol we calculate:

$$\alpha_{n+1} = \psi_{n+1} - \varphi_n$$

To determine the tangent angle ψ again requires a case distinction (3):

1. $0 \leq s \leq 2a \Rightarrow \psi = \pi$
2. $2a < s < 2a + b\pi \Rightarrow \psi = \frac{s-2a}{b}$
3. $2a + b\pi \leq s \leq 4a + b\pi \Rightarrow \psi = 0$
4. $4a + b\pi < s < 4a + 2b\pi \Rightarrow \psi = \frac{s-4a-b\pi}{b}$

At the start, the first reflection angle α_1 is given. The first impact point is also given, so that we can calculate the associated tangent and its angle ψ_1 can be calculated. The angle of the first path section is then $\varphi_1 = \psi_1 + \alpha_1$. With the second impact point, we calculate its tangent angle ψ_2 and then $\varphi_2 = 2\psi_2 - \varphi_1$. This gives us all the other angles during the iteration.

It remains to find a method for calculating the next impact point. Starting from an impact point P with a position vector $\vec{p} = \binom{u}{v}$ the next section of the ball's path is on a straight line with an angle φ to the positive x-axis, i.e. on a straight line:

$$\vec{x}(t) = \binom{u}{v} + t \cdot \binom{cos\varphi}{sin\varphi}, t \in \mathbb{R}$$

If $\varphi \neq 0, \pi$ we first intersect this line with the lines $y = \pm b$. This yields

$$t = \frac{\pm b - v}{sin\varphi}$$

If $u + tcos\varphi \in [-a, a]$ we have found the intersection.

If we have not yet found the point of intersection, we look for it on the two semicircles with centre $(\pm a, 0)$ and radius b.

These have the equations:

$$(x \mp a)^2 + y^2 = b^2$$

With the condition $x > a$ for the right-hand circle (with a negative sign for a in the brackets) and $x < -a$ for the left-hand circle with a positive sign in the brackets.

This provides the condition for the point you are looking for on the straight line:

$$(u + tcos\varphi \mp a)^2 + (v + tsin\varphi)^2 = b^2$$

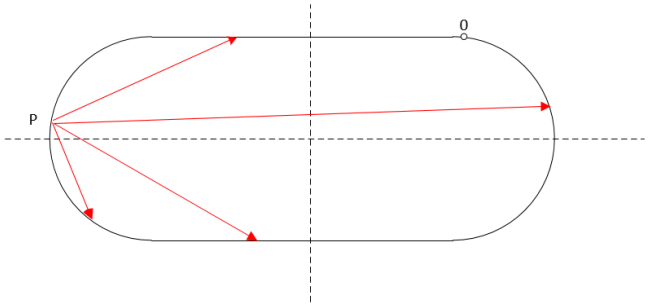
After a brief calculation, we receive for t:

$$t = -B \pm \sqrt{B^2 - C}$$

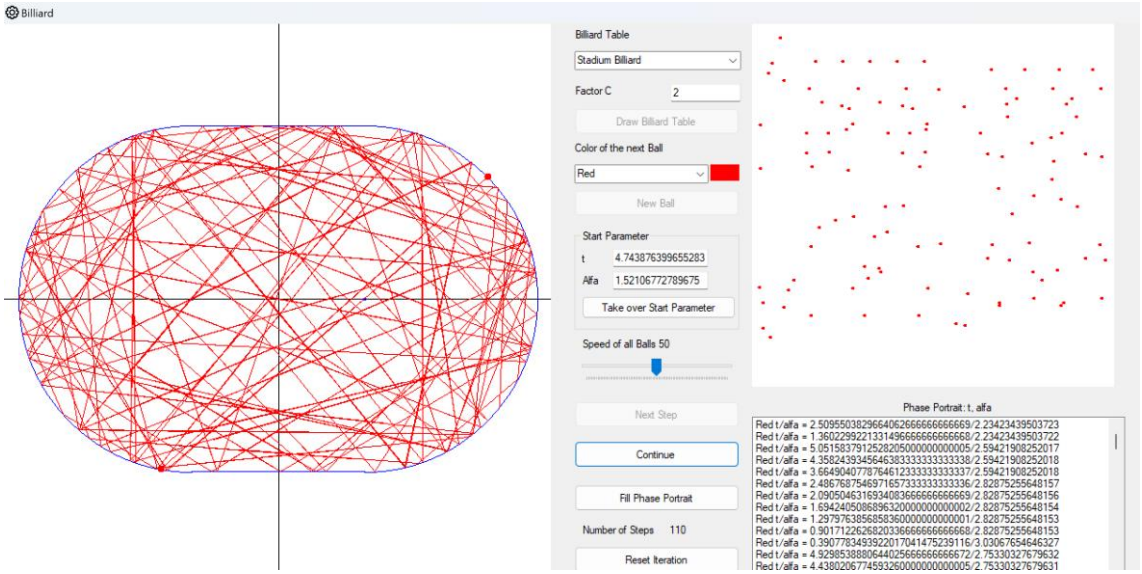
Whereby

$$B = (u \mp a)cos\varphi + vsin\varphi$$

$$C = (u \mp a)^2 + v^2 - b^2$$

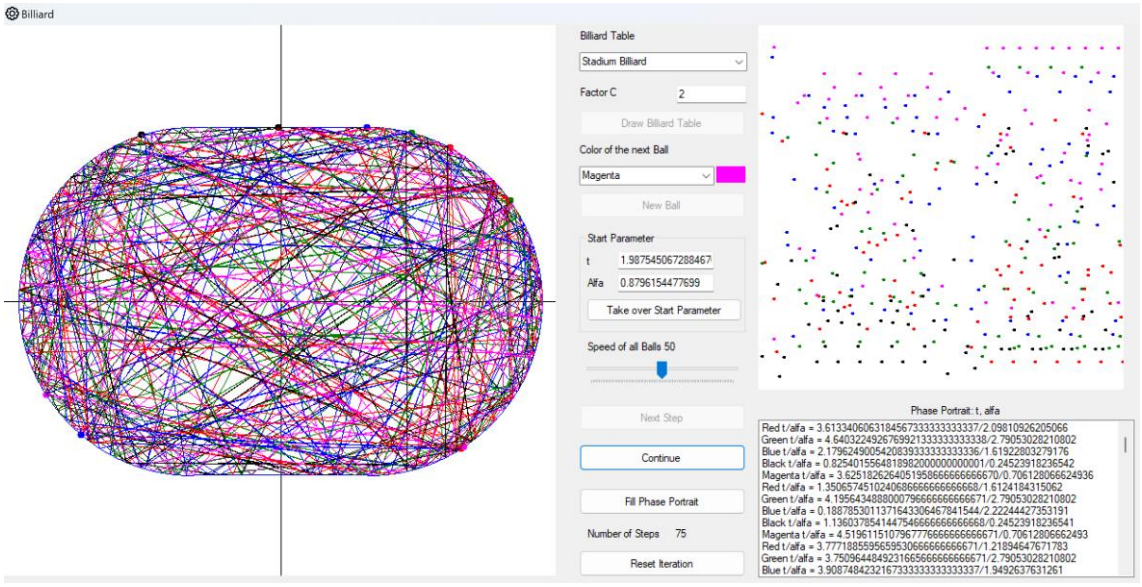


Various scenarios for calculating the next impact point



The first 110 impacts of a ball in the stadium

Subsequently, five balls of different colours were launched simultaneously and each performed 75 shots.



The billiards in the stadium are chaotic

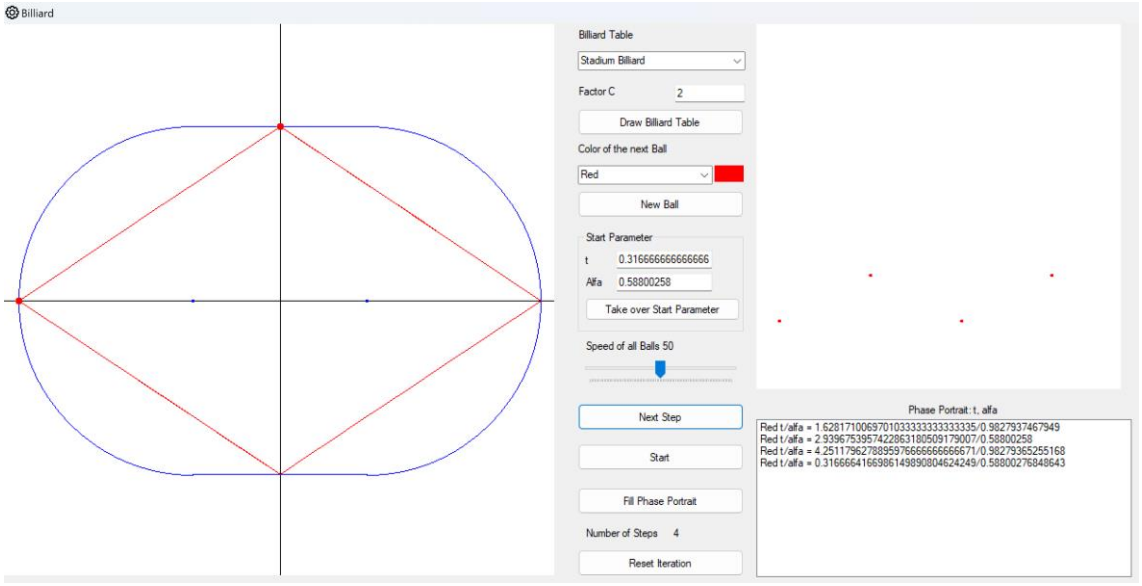
As you can see in the phase portrait on the right, the entries are distributed over the entire phase portrait. In the elliptical billiard, the entries in the phase portrait were limited to certain curves and at least the transitivity in the elliptical billiard as mapping

$$f: (t_n, \alpha_n) \to (t_{n+1}, \alpha_{n+1}), [0,2\pi[\ x\ ]0,\pi[\to [0,2\pi[\ x\ ]0,\pi[$$

is not fulfilled, as certain areas are always left out of the phase portrait.

On the other hand, chaotic behaviour has been proven for billiards in the stadium. This type of billiards is also known as "Bunimowich Stadium" after Leonid Bunimowich, who provided this evidence.

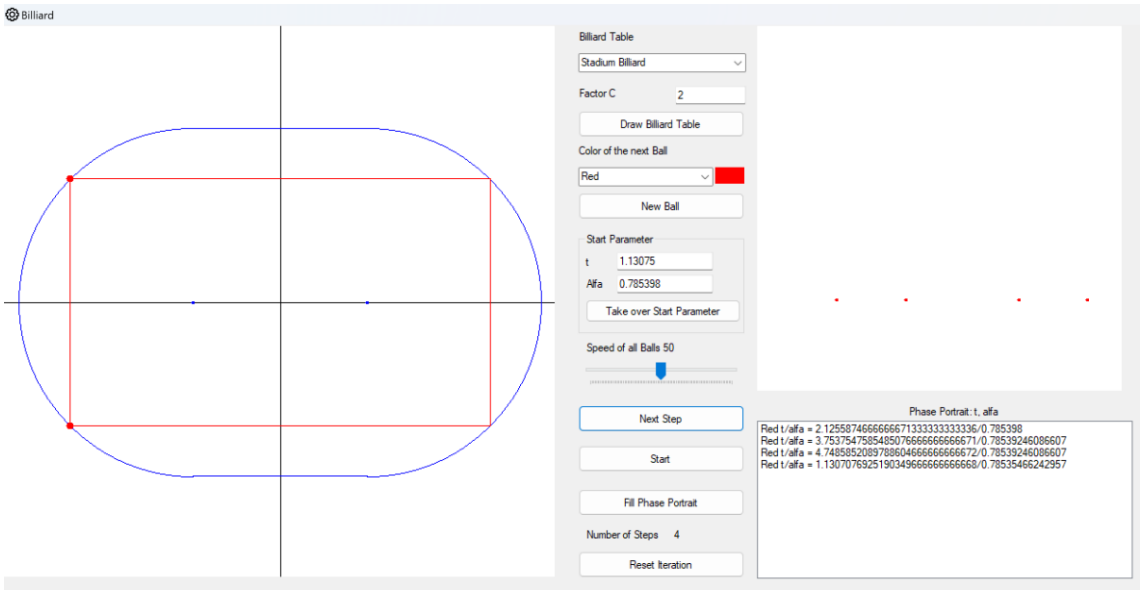
It is also easy to find periodic points for billiards in the stadium. For example, starting from an apex, there are $(0, a + b)$ paths of each period $2k, k \in \mathbb{N}$which can be found through corresponding paths in the circle.



A track of period 4

For the factor $c = 2$ is $a = 0.3\overline{16}, b = 0.6\overline{3}$. The parameter for the starting point is $s = 0.3\overline{16}$. For the start angle α applies $\alpha = arctan\frac{b}{a+b} = 0.58800258 \ldots$

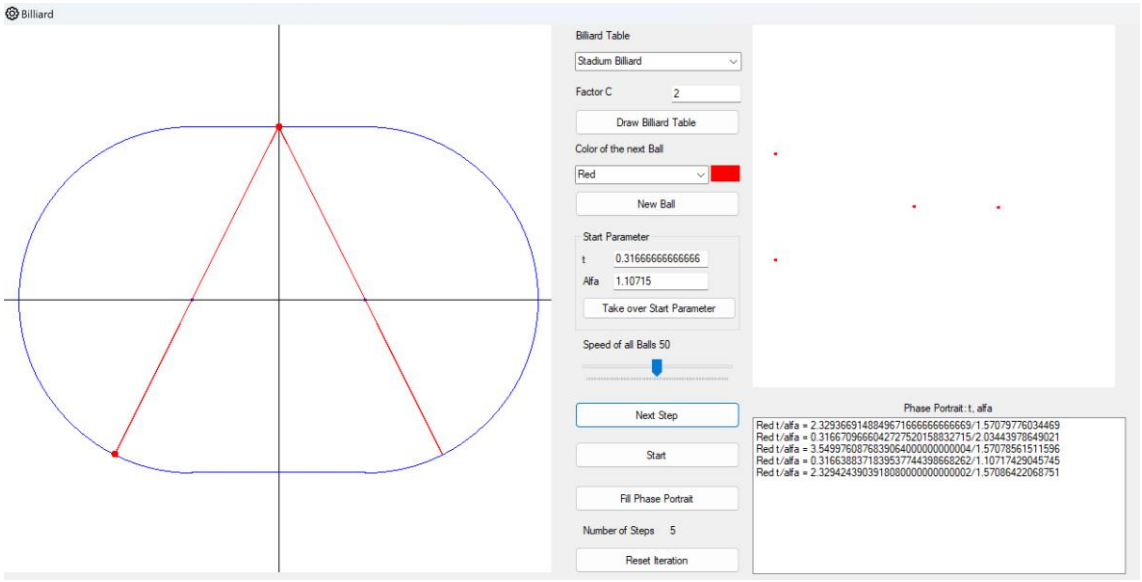
Another track with period four is the following:



An alternative track for period four

As you can easily calculate, the following applies to the above path $s = 2a + b \cdot \frac{\pi}{4}$ and $\alpha = \frac{\pi}{4}$.

If a path passes through the centre of a circle and is then mirrored at it, it passes through the centre again on the way back. This allows you to find other periodic paths.



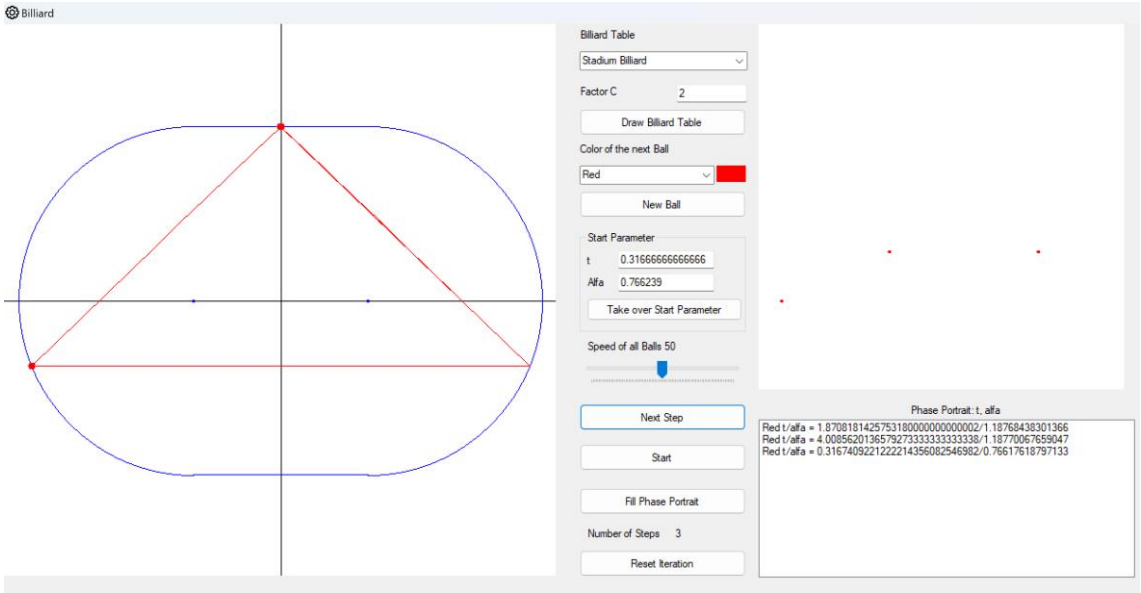
Another track of period four

In the above case, again $s = 0.3\overline{16}$ i.e. the starting point is $(0, b)$. For the angle α applies $\alpha = arctan\frac{b}{a} = 1.10715 \ldots$

For paths of odd period, we can again determine the corresponding start parameters experimentally, at least approximately, using the same approximation method as for elliptical billiards. As an

example, we are looking for a three-period trajectory with a starting point in $(0, b)$ for the factor $c = 2$. For this factor, the start parameter is $s = a = 0.31\overline{6}$. We first start with an angle $\alpha = 0.75$ and adjust it continuously so that after three impacts in the protocol, the parameter $s = 0.31\overline{6}$ for the position of the ball is reached as well as possible.

After a few iteration steps, we obtain the following approximation $\alpha \approx 0.766239$:



Approximate start parameters for a three-period orbit

4.8. Implementation in the "Simulator"

The *ClsStadiumBilliardball* class implements the *IBilliardball* interface. Nothing needs to be adjusted in the *FrmBilliardtable* window.

The billiard ball contains the entire movement logic, which is the only way we can easily instantiate any number of balls. So that the designation of the parameter for the impact point is uniform, we use the designation *t* instead of *s* for this parameter in the implementation. The case differentiation (2) for calculating the coordinates of an edge point from the parameter *t* is implemented in the class *ClsStadiumBilliardball* under the name *CalculateMathPointFromT*.

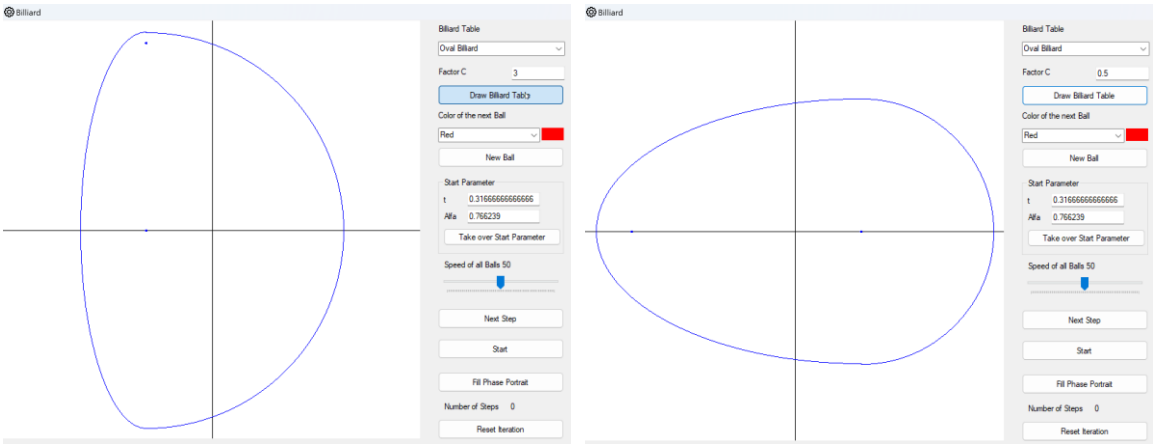
As with elliptical billiards, the starting point and the starting angle of the ball can be set at the beginning using the mouse.

The case distinction (3) for calculating the tangent angle ψ is implemented in *CalculatePsi*.

4.9. Oval Billiards

Some chickens would probably not enjoy certain eggs, which we will use as a pool table below. The billiard table is made up of a semicircle and half an ellipse. Where *b* is always the radius of the circle and if $a \geq b$, *a* is the major axis of the ellipse and *b is* its minor axis. If $a < b$ it is the other way around. These parameters are not set explicitly, only the factor $c := a/b$. Then *a* and *b* are determined so that the figure fits well into the diagram. The horizontal diameter of the oval is $a + b$,

the vertical $2b$. This maximum is set to 1.9 for better visibility. The value range of the mathematical coordinates of the diagram is $[-1,1] \times [-1,1]$, therefore has a diameter of 2.
If $a \geq b$, then $a(1+c) = a+b \geq 2b$, you therefore set $a = \frac{1.9}{1+c}$. If $b > a$, then $2ac = 2b \geq a+b$ and you set $a = \frac{1}{2c}$. In both cases, then $b = ca$.
This is implemented in the "Simulator" when the factor *c* is transferred.
To make maximum use of the diagram, the point with the x-coordinate $(a-b)/2$ is set as the center of the circle or ellipse.



On the left an "egg" or an oval with factor $c = 3$, on the right with factor $c = 0.5$

If $c = 1$ you have a circle.

We select the parameter representation of the oval:

$$\vec{s}(t) = \begin{cases} \begin{pmatrix} m + b\cos t \\ b\sin t \end{pmatrix}, t \in [-\pi/2, \pi/2] \\ \begin{pmatrix} m + a\cos t \\ b\sin t \end{pmatrix}, t \in]\pi/2, 3\pi/2[\end{cases}$$

Where $m = (a-b)/2$ is the x-coordinate of the centre point. (Its y-coordinate is 0).

The curve parameterized in this way is continuous. The following applies:

$$\lim_{t \to \pi/2^+} \vec{s}(t) = \begin{pmatrix} m \\ b \end{pmatrix} = \vec{s}(\pi/2) \quad \text{resp.} \quad \lim_{t \to 3\pi/2^-} \vec{s}(t) = \begin{pmatrix} m \\ -b \end{pmatrix} = \vec{s}(3\pi/2)$$

However, it is not differentiable at the transition points if $a \neq b$:

$$\lim_{t \to \pi/2^+} \dot{\vec{s}}(t) = \begin{pmatrix} -a \\ 0 \end{pmatrix} \neq \begin{pmatrix} -b \\ 0 \end{pmatrix} = \dot{\vec{s}}(\pi/2) \quad \text{resp.} \quad \lim_{t \to 3\pi/2^-} \dot{\vec{s}}(t) = \begin{pmatrix} a \\ 0 \end{pmatrix} \neq \begin{pmatrix} b \\ 0 \end{pmatrix} = \dot{\vec{s}}(3\pi/2)$$

However, we only need the gradient of the tangent at each point, and this is at the transition points 0 as is the limes on both sides at these points. The *slope* of the tangent is therefore continuous at these points.

Everything else is identical to the elliptical billiard, except that the ellipse/circle case distinction is used instead of the ellipse in the parameterization, depending on the parameter *t*. In the implementation, we outsource the determination of the coordinates of an edge point depending on the parameter *t* to a corresponding function *CalculateMathPointFromT.*

All relationships between the angles are the same:

$$\begin{cases} \alpha_{n+1} = \psi_{n+1} - \varphi_n \\ \varphi_{n+1} = \psi_{n+1} + \alpha_{n+1} \\ \varphi_{n+1} = 2\psi_{n+1} - \varphi_n \end{cases}$$

When calculating the new impact point, we start with an impact point $\vec{s}(t_n)$. The direction angle of the next path section is φ_n. Then the next impact point we are looking for lies on the straight line

$$\vec{x}(t) = \begin{pmatrix} p \\ q \end{pmatrix} + u \cdot \begin{pmatrix} cos\varphi_n \\ sin\varphi_n \end{pmatrix}, u \in \mathbb{R}$$

with $\begin{pmatrix} p \\ q \end{pmatrix} = \vec{s}(t_n)$.

If we intersect this straight line with the oval, we get the next point of impact we are looking for.

In the case of the ellipse, the corresponding equation is:

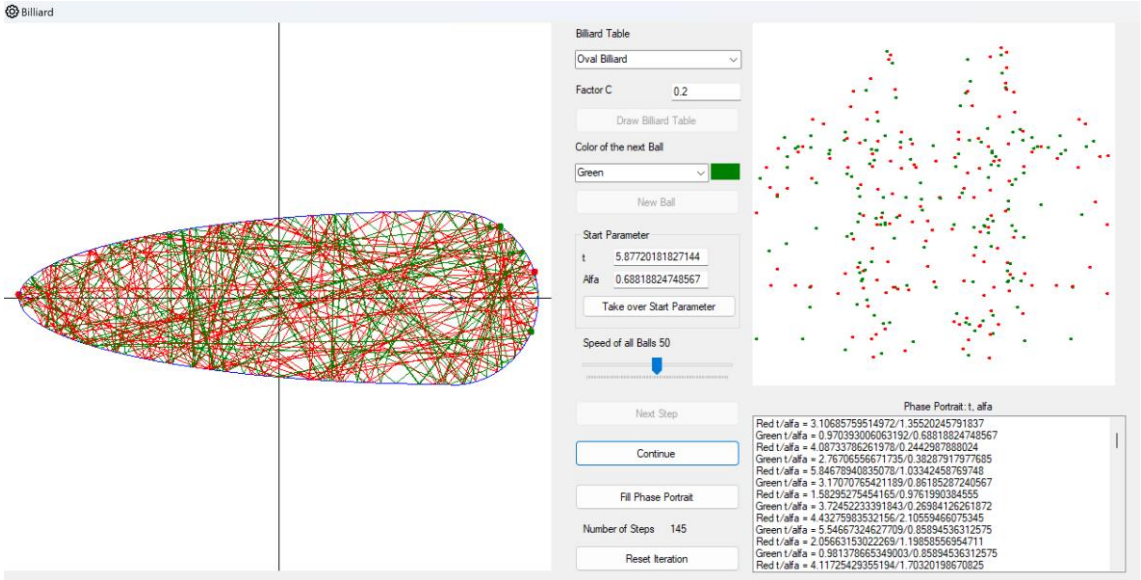
$$\frac{(x-m)^2}{a^2} + \frac{y^2}{b^2} = 1$$

Whereby $m = (a-b)/2$.

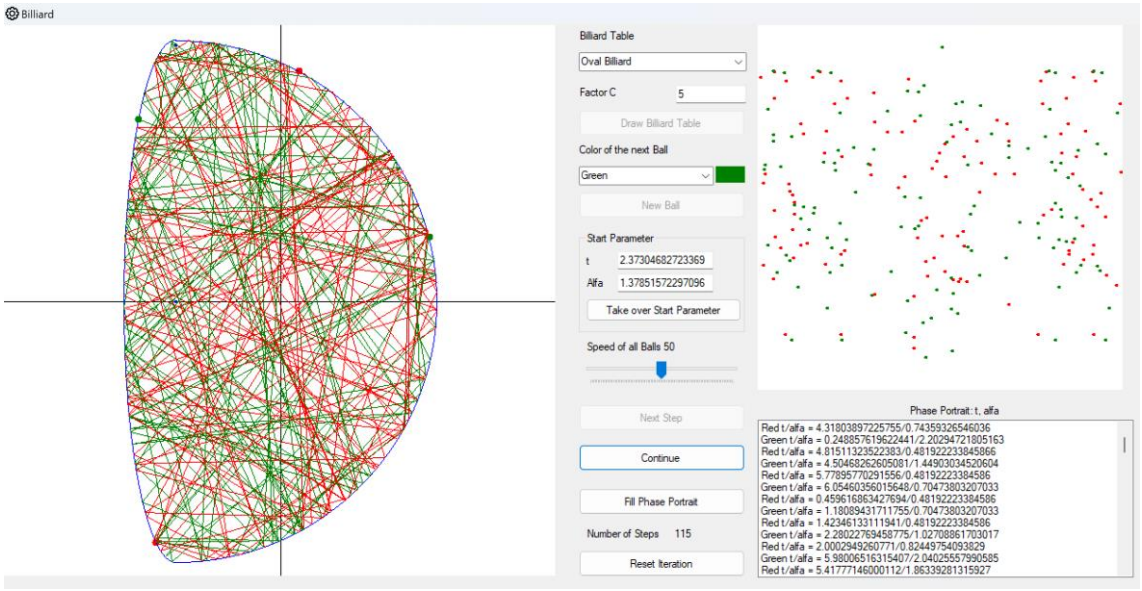
We leave it to the reader to solve the quadratic equation that now appears. In the case of the circle, the equation above is $a = b$ and we proceed analogously.

This generally produces 4 points of intersection, two with the ellipse and two with the circle. One of these is the starting point, i.e. the solution $u = 0$ which can be excluded.

For the circle intersections, their x-coordinate must be >= m. Otherwise they can be excluded. Similarly, the ellipse intersection points must have an x-coordinate < m, otherwise they can be excluded. As the straight line starting from an edge point of the oval is not tangential, you get exactly one second intersection point.



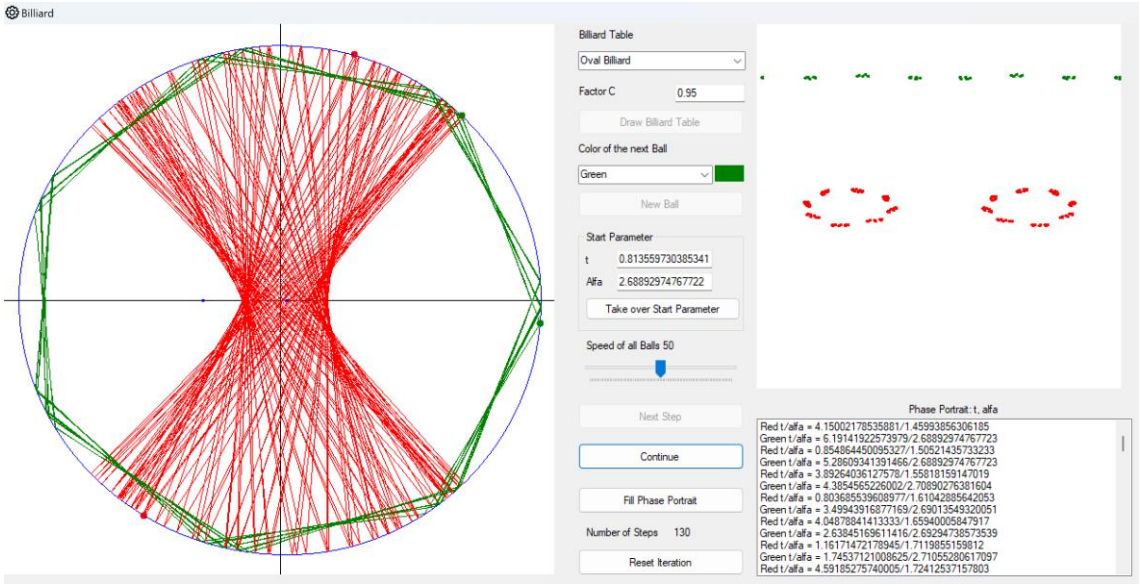
The path of two balls with c = 0.2



Two balls with c = 5

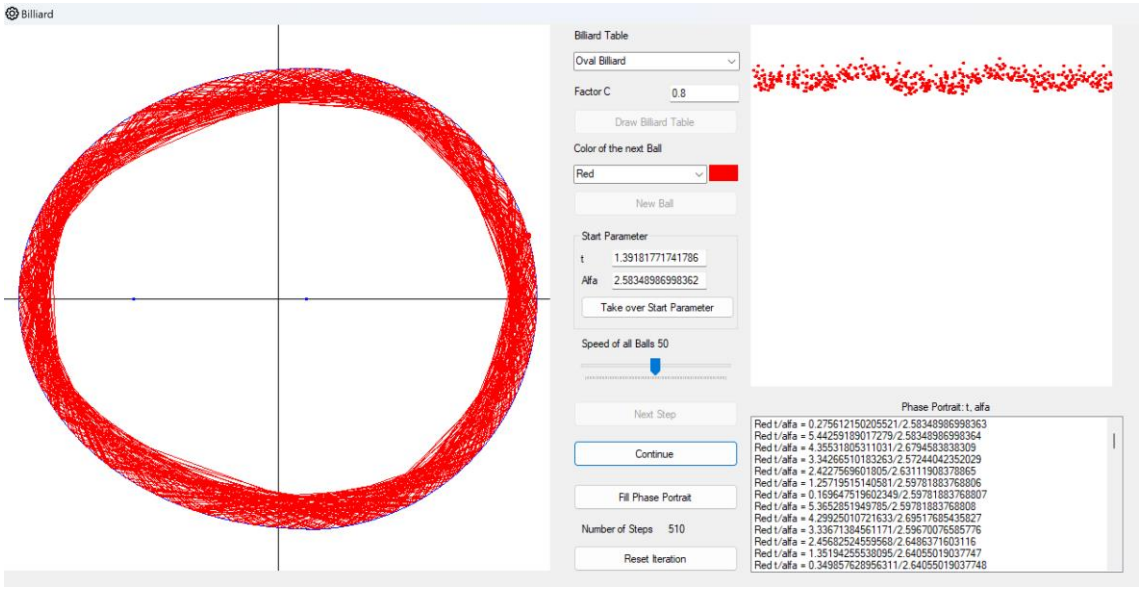
If you let the balls run for longer, the points in the phase portrait spread more and more over the entire room. This could indicate chaotic behaviour.

But if you choose *c* close to 1, then the lanes are more like those in an elliptical billiard table:



Two balls with c = 0.95

If you start the ball track flat, the ball seems to behave quite neatly at first, almost like an ellipse. After a while, however, the ball "tilts" more and more towards a chaotic appearance.

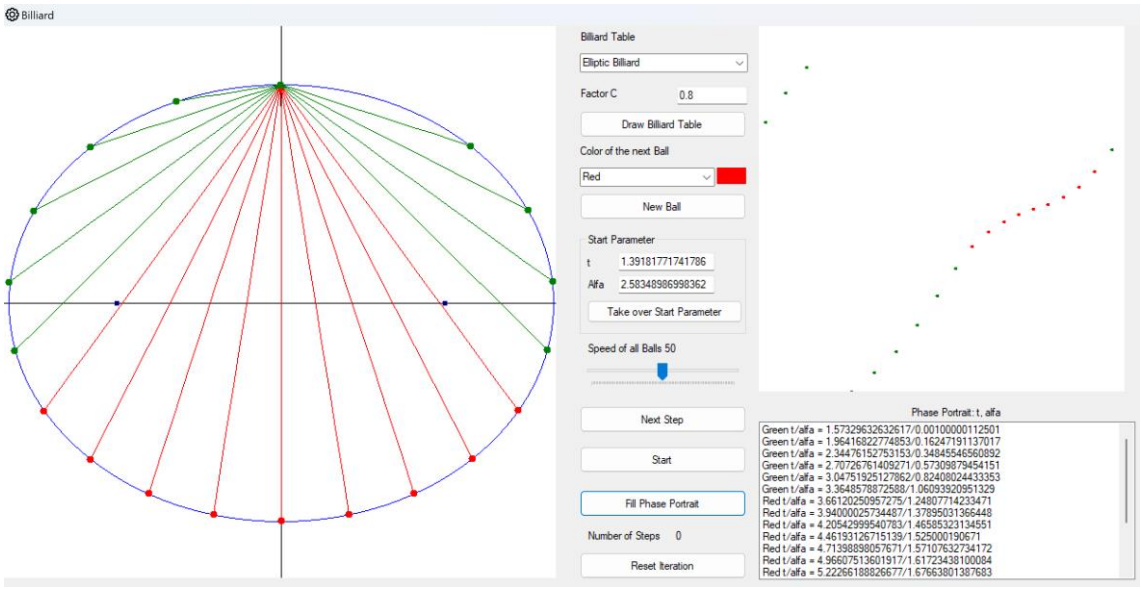


A spherical path with c = 0.8, which still comes close to a caustic after 510 impacts

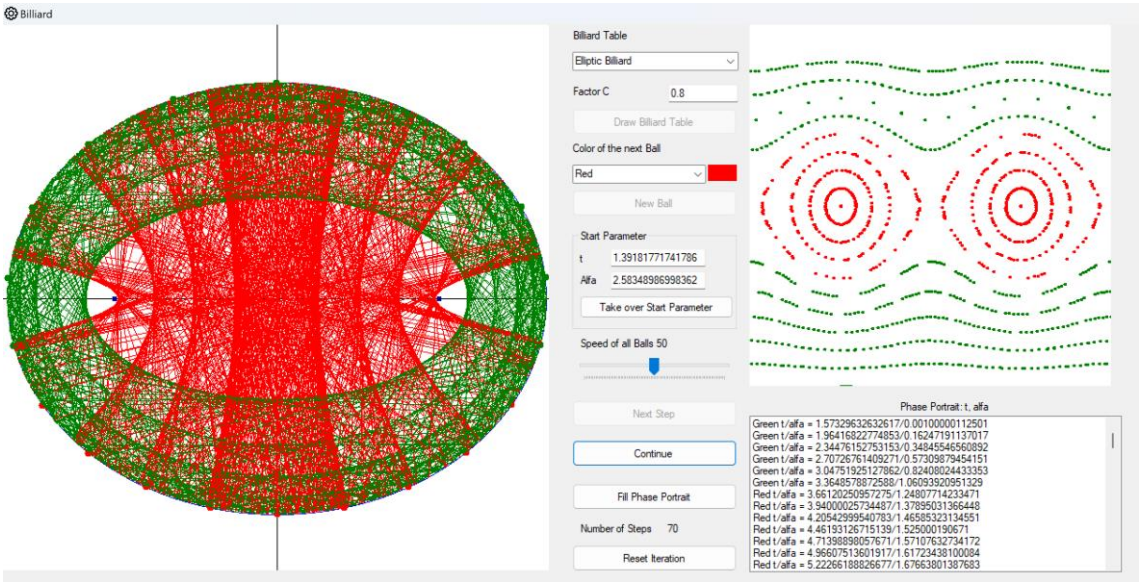
As with earlier billiard tables, the periodic points of oval billiards can also be discussed. We leave this to the reader in the form of an exercise.

4.10. Representation of the phase space

The "Fill phase space" button makes it possible to display the trajectories of many spheres with different starting conditions in phase space simultaneously. This provides an overview of the complexity of the system.



Starting position of the automatically generated balls



Phase space on the right, after each ball has performed 70 impacts

4.11. The convex Billiard Table

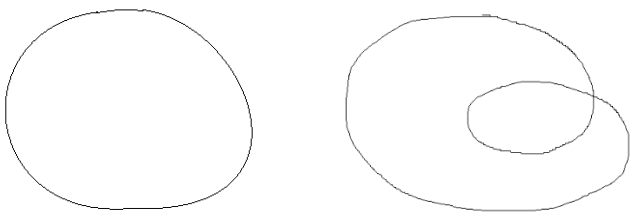
As an outlook, we consider a billiard table whose edge is a (plane) convex, closed, and smooth curve. Some elementary mathematical statements can be made here, above all as an application of the theorem of maximum and minimum of continuous functions on a closed interval.

We understand a *plane curve* to be a continuous mapping:

$$\gamma: I \subset \mathbb{R} \longrightarrow \mathbb{R}^2, t \mapsto \gamma(t)$$

The curve is called *smooth* if it is continuously differentiable. The curve is said to be *regular* if the absolute value of the derivative does not vanish, i.e. $|\dot{\gamma}(t)| \neq 0, \forall t \in I$. That the curve should be *closed* means that $I = [a, b]$ is a (finite) real interval with $\gamma(a) = \gamma(b)$, where the derivative also coincides: $\gamma'(a) = \gamma'(b), \forall k \in \mathbb{N}$.

To make the edge of a billiard table look "reasonable", the curve should be *simply closed*, i.e. the mapping γ on [a, b [is *injective.* This also means that there are no double points.



On the left a single closed curve, on the right the curve is not single closed

We will often need the derivation of $|\gamma(t)|$. You can convince yourself that applies:

$$\frac{d}{dt}|\gamma(t)| = \frac{\gamma(t) \cdot \dot{\gamma}(t)}{|\gamma(t)|}$$

The scalar product is in the numerator.

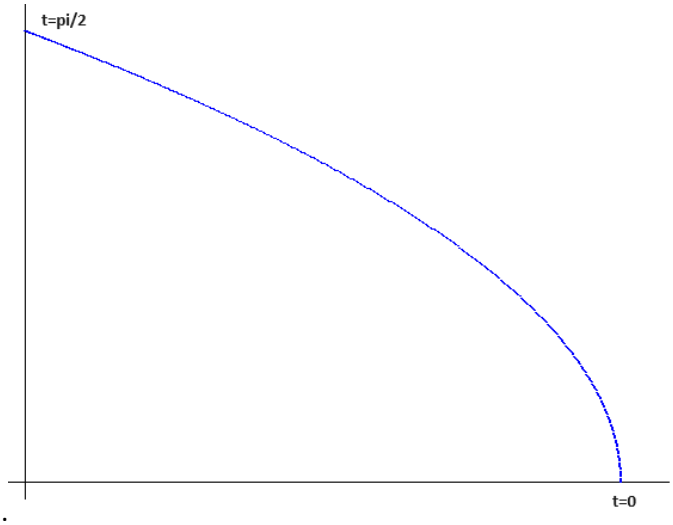
Example

The flat curve

$$\gamma(t) = \begin{pmatrix} cos^2 t \\ sint \end{pmatrix}, t \in [0, \pi[$$

Is closed: $\gamma(0) = \gamma(\pi)$. It is smooth, but not regular: $\left|\dot{\gamma}(\frac{\pi}{2})\right| = 0$.

It has "peaks" at $t = 0, \frac{\pi}{2}$.



.

Graph of the plane curve defined above

In the following, we only consider *convex* billiard tables. These are tables that are bounded by *convex* curves. We use the following definition for convexity:

A simply closed and flat curve is called convex if for any two points that lie within the area enclosed by the curve, their connecting line also lies entirely within this area.

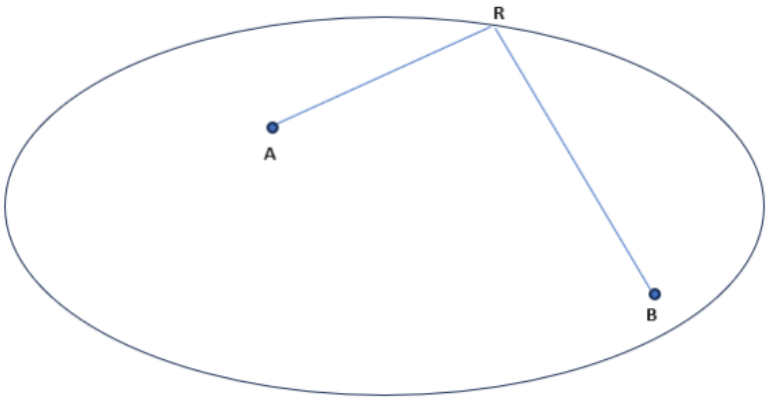
In the following, a *convex billiard table* will therefore always be bounded by a flat, simply closed, smooth, regular, and convex curve.

A convex curve can contain straight line segments. If it does not, it is called *strictly convex.*

Example

The elliptical and oval billiard is strictly convex, the billiard in the stadium is convex.

In the following, we will define the term "*point of reflection*". Let γ be the edge of a convex billiard table and *A, B* two different points inside it. Then *R* is called a *reflection point for A, B* if a path segment starting from *A* is reflected in *R* in such a way that it arrives at *B* after reflection.



R is the reflection point for *A, B*

Note: In elliptical billiards, each point on the ellipse is a reflection point for the focal points of the ellipse.

A first question that can be asked is: *Is there always a reflection point R for any points $A \neq B$that lie inside a convex billiard table always have a reflection point R?*

We consider the function

$$f(t) = \left|\overrightarrow{AR}\right| + \left|\overrightarrow{RB}\right|, t \in I$$

Where *I* is the parameter interval for the boundary curve that limits the billiard table. Let $\vec{r}(t)$ is the location vector associated with *R.* Then is:

$$f(t) = |\vec{r}(t) - \vec{a}| + \left|\vec{r}(t) - \vec{b}\right|$$

f is a real-valued, continuous function and *I* is a closed interval. If *f* is constant, then *A* and *B* are focal points of an ellipse and then each edge point is a reflection point. Otherwise, *f* effectively assumes a maximum and a minimum on *I. I* can be continued periodically, as the boundary curve is closed. This means that the extrema of *f* occur in the interior of *I* or in the interior of the periodic continuation. Because the boundary curve is smooth, *f* is continuously differentiable. The derivative of *f* at the extreme points is zero. Let t_1, t_2 be the parameter values for which *f* is maximum or minimum. Then the following applies:

$$\frac{d}{dt}f(t) = \frac{(\vec{r}(t) - \vec{a})}{|\vec{r}(t) - \vec{a}|} \cdot \dot{\vec{r}}(t) + \frac{\left(\vec{r}(t) - \vec{b}\right)}{\left|\vec{r}(t) - \vec{b}\right|} \cdot \dot{\vec{r}}(t) = 0 \;\; f\ddot{u}r\, t = t_{1,2}$$

$\frac{(\vec{r}(t)-\vec{a})}{|\vec{r}(t)-\vec{a}|} =: \vec{e}_a$ is a unit vector and likewise $\frac{(\vec{r}(t)-\vec{b})}{|\vec{r}(t)-\vec{b}|} =: \vec{e}_b$ in each direction of the corresponding path section through point *A* or *B*. The sum of the two unit vectors $\vec{e}_a + \vec{e}_b$ is the angle bisector of these two path sections. The following applies:

$$(\vec{e}_a + \vec{e}_b) \cdot \dot{\vec{r}}\left(t_{1,2}\right) = 0$$

This angle bisector for $t = t_{1,2}$ is perpendicular to the tangent at these points and therefore the connecting lines *AR* and *BR* form the same angle with the tangent.

Since the billiard table is convex, the connections *AR* and *BR* lie entirely inside the table and represent sections of the path of a ball which starts in *A*, is reflected in *R,* and then hits *B.*

As a result, we have:

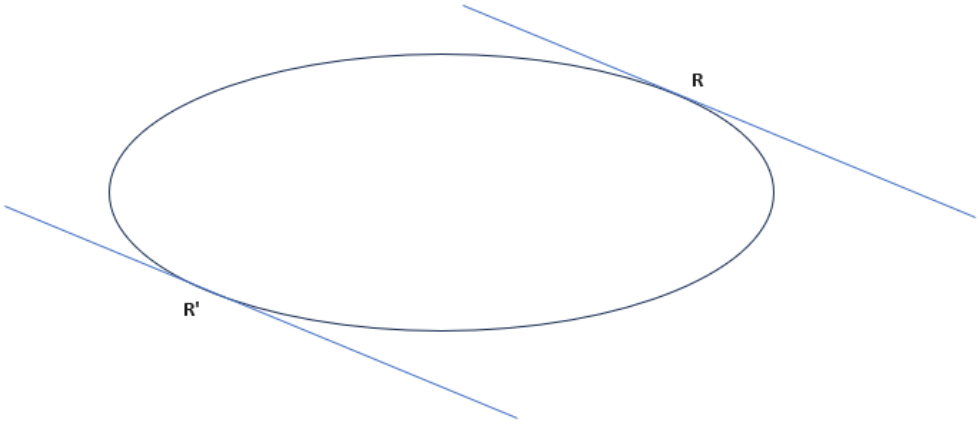
If A and B are any two points inside a convex billiard table. Then there are at least two reflection points $R_{1,2}$ for A and B. At these points, the sum of the distances $|AR| + |BR|$ has a local extremum.

You can also examine the case where *A* and *B* coincide. The same result also applies in this case: there are at least two reflection points, so that a ball emanating from *A* returns to *A* after the impact. The proof is an exercise.

In some examples, we were able to approximate start parameters that led to periodic trajectories with interval nesting. The question is whether there are always periodic trajectories for convex billiard tables.

Let us consider a (possibly non-convex) billiard table and a tangent to its edge at any edge point. Assume that there are points on both sides of the tangent that belong to the interior of the billiard table. Then it is possible to find points on different sides of the tangent whose connection is no longer completely inside the billiard table. The billiard table would then not be convex. Conversely, if the billiard table is convex and we place a tangent in any edge point, then all points that belong to the billiard table lie on the same side of the tangent.

Let's take a convex billiard table and place the tangent at an edge point *R*. If we let this edge point run around the edge of the table once, the tangent will have rotated by 2π. Due to the continuity of the derivative, there is therefore a boundary point *R'* in which the tangent is parallel to the original one. If we now consider such a pair of tangents, their distance is the "width" of the billiard table with respect to these tangents.



A pair of parallel tangents on a convex billiard table

The width of the billiard table as a function of the edge point *R* or thus of the associated parameter *t* of the ellipse edge is a periodic, restricted real function and thus assumes a maximum and a minimum. This function can be represented as:

$$B(t, t') = |\vec{r}(t) - \vec{r}(t')|$$

At the extreme points, the partial derivatives of *B* with respect to *t* and *t'* are zero. The following therefore applies:

$$\frac{d}{dt}B(t,t') = \frac{\vec{r}(t)-\vec{r}(t')}{|\vec{r}(t)-\vec{r}(t')|}\cdot\dot{\vec{r}}(t) = 0$$

This means that the vector $\overrightarrow{RR'}$ is perpendicular to the tangent at point *R*. By taking the derivative to *t'*, you can see that this vector is also perpendicular to the tangent at point *R'*. This results in a 2-periodic path. Since there are at least two extremal points, it follows:

A convex billiard table has at least two 2-period tracks. They correspond to the minimum and maximum width of the table or its diameters.

Based on our experiments with the "Simulator", we can assume that it is possible to find periodic orbits of any period on a convex billiard table. This is indeed the case, because the

Birkhoff Theorem

For a strictly convex billiard table, for every pair (p, q) of natural numbers with $q \geq 2$ *and* $p \leq [(q-1)/2]$ *there are two geometrically different q-periodic orbits with circulation number p.*

$[(q-1)/2]$ here denotes the next smaller natural number less than $(q-1)/2$.

A direct proof of this theorem can be found in [7]. This theorem is a corollary of a much more general fixed-point theorem by Poincaré-Birkhoff, which cannot be explained here. A good introduction with an application of this theorem to billiards can be found in [8].

4.12. The C-diagram

For unimodal functions, we examined the dependency of the iterated function on a parameter in the so-called Feigenbaum diagram. We saw that periodic behaviour occurred for certain parameter values and chaotic behaviour for other parameter values.

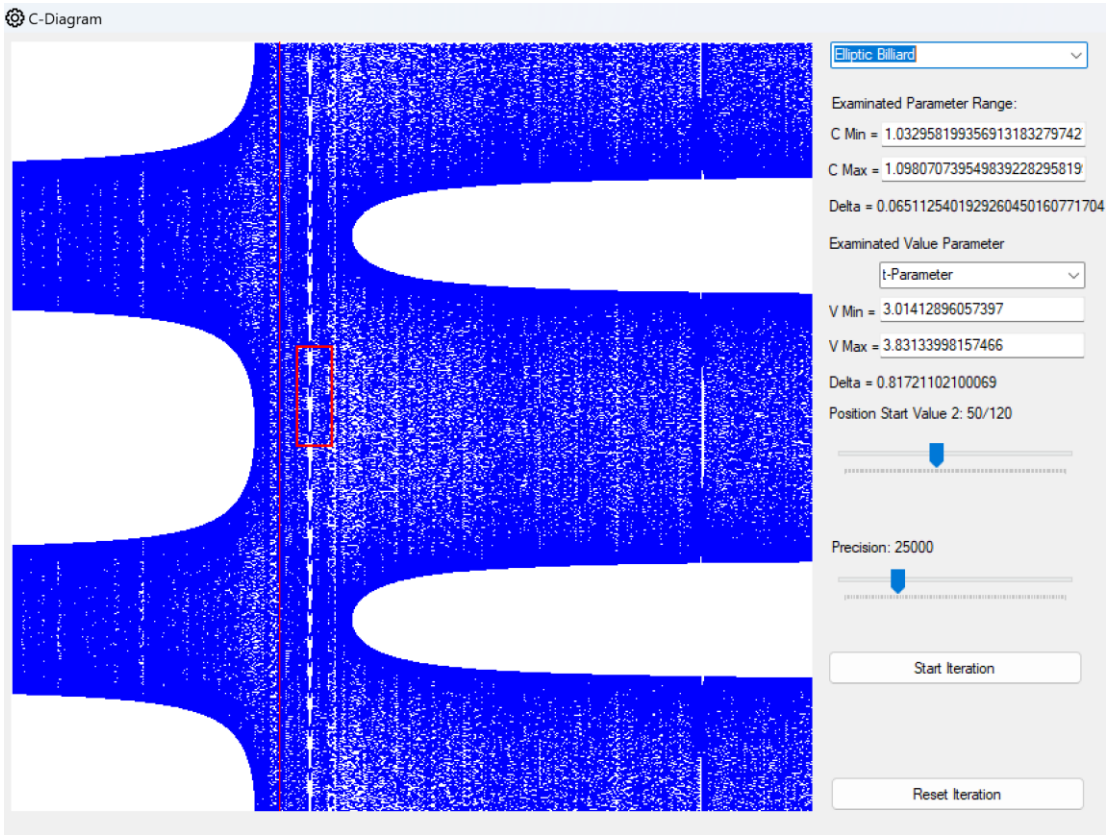
For billiards, we have defined a parameter *C* in all our examples, which determines the shape of the billiard table. It was:

- Elliptical billiard: *C* = ratio of the minor axis to the major axis of the ellipse
- Billiards in the stadium: *C* = ratio of the diameter of the circle to the width of the rectangle in the stadium
- Oval billiard: *C* = ratio of circle radius to the main axis of the ellipse

For elliptical and oval billiards, the shape of the billiard table for *C* =1 is a circle.

C influences the properties of the billiard table. We now have two parameters to describe the path of a ball on the billiard table, namely a parameter *t*, which describes the location of an impact point on the edge of the billiard table, and an angle *α,* which describes the impact angle. This is the angle between the ball path and the tangent at the impact point.

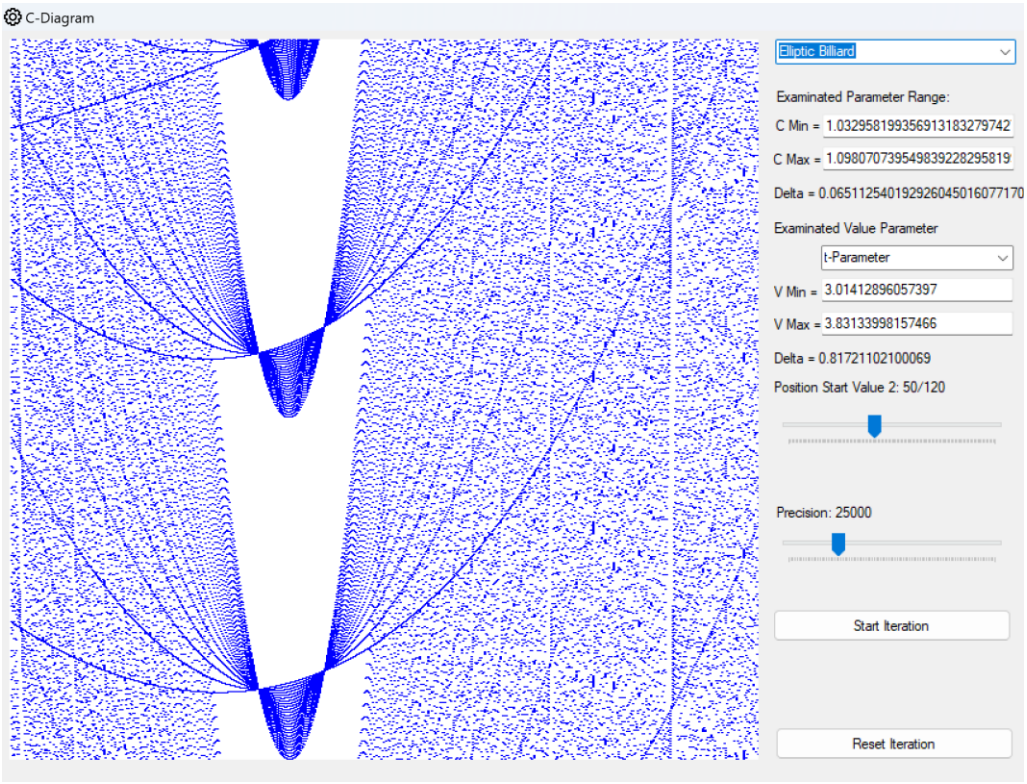
So that we can display the influence of *C* on these parameters in a two-dimensional diagram, we select one of the two parameters for the display. The "Simulator" then examines what the asymptotic behaviour of the parameter looks like for each value of *C,* as with the Feigenbaum diagram.



Dependence of the parameter *t* on *C* in elliptical billiards

The red vertical line in the diagram marks the point $C = 1$.

As with the Feigenbaum diagram, you can select sections of the image by holding down the left mouse button. The corresponding intervals for *C* and for the parameter to be examined are then adjusted on the right-hand side. If you run the iteration again, the corresponding section is displayed in enlarged form:



Enlarged view of the previously selected section

We make a few agreements for the iteration:

- *C* varies by default between 0.5 and 2. These values can be adjusted on the right in the parameter range under investigation.
- We set the start value of the parameter that is *not* to be displayed in the diagram to 1/3 of its parameter interval. In the images above, the angle $\alpha = {}^{\pi}/_{3}$.
- The start value of the parameter shown in the diagram can be set to ${}^{p}/_{12}$ of its parameter interval, where $p \in \{1,2,\dots,11\}$. In the images above $t = 2\pi \cdot \frac{5}{12}$.

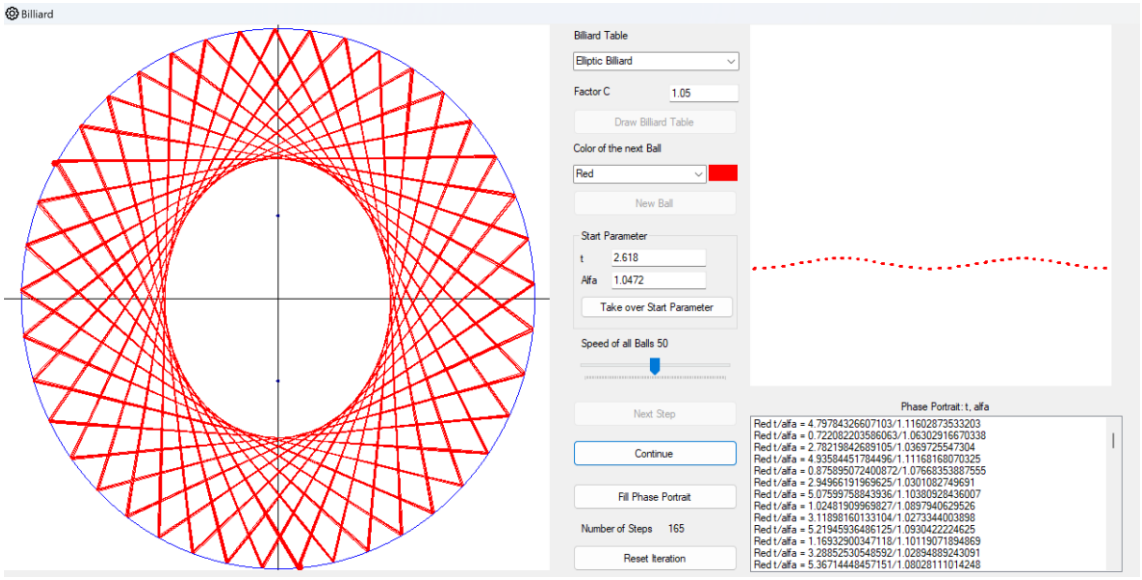
If you want to change this, you can easily adapt the corresponding code in the *FrmCDiagram* and the *DrawIterationValues* subroutine.

It is important that the *same* starting values are set for each *C* for the iteration so that the results in the diagram above are consistent and the magnifications of sections are correct.

In contrast to the Feigenbaum diagram, we are unfortunately unable to offer a mathematical analysis of the generated diagrams. However, the images can be made plausible based on the shape of the billiard table and the selected start parameters.

In the previous diagram, we can use another selection rectangle to estimate for which values of *C* the "node" visible in the centre of the diagram occurs. This is in the range $C \in [1.05, 1.06]$.

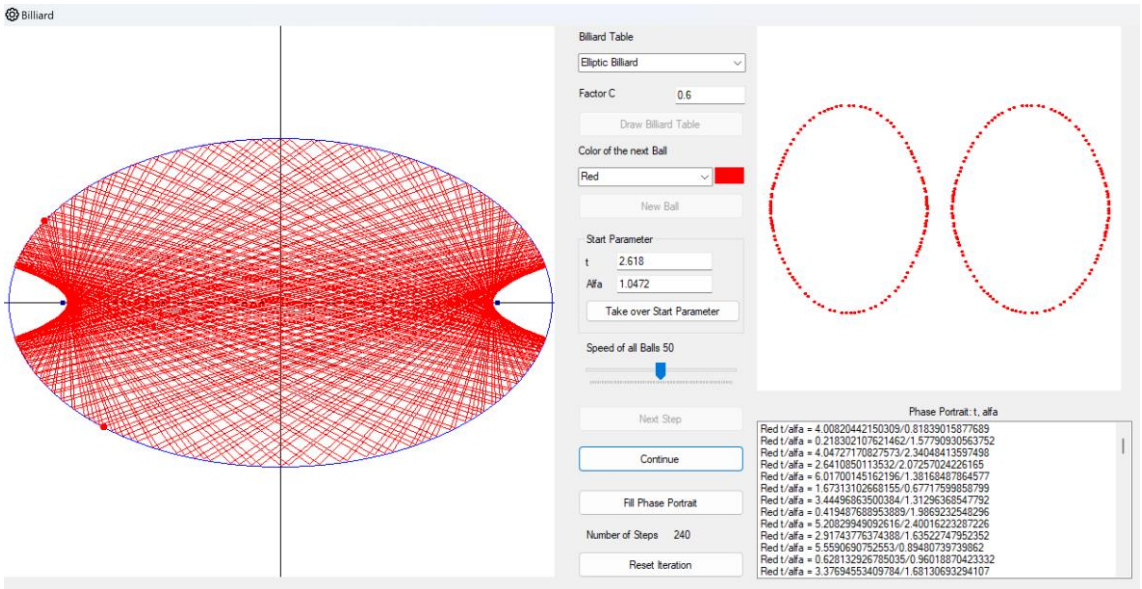
So, let's go to the elliptical billiard table, select *C* in this area and set the start values according to the C diagram: $t = 2.618, \alpha = 1.0472$.



The path of the billiard ball with the previous parameters

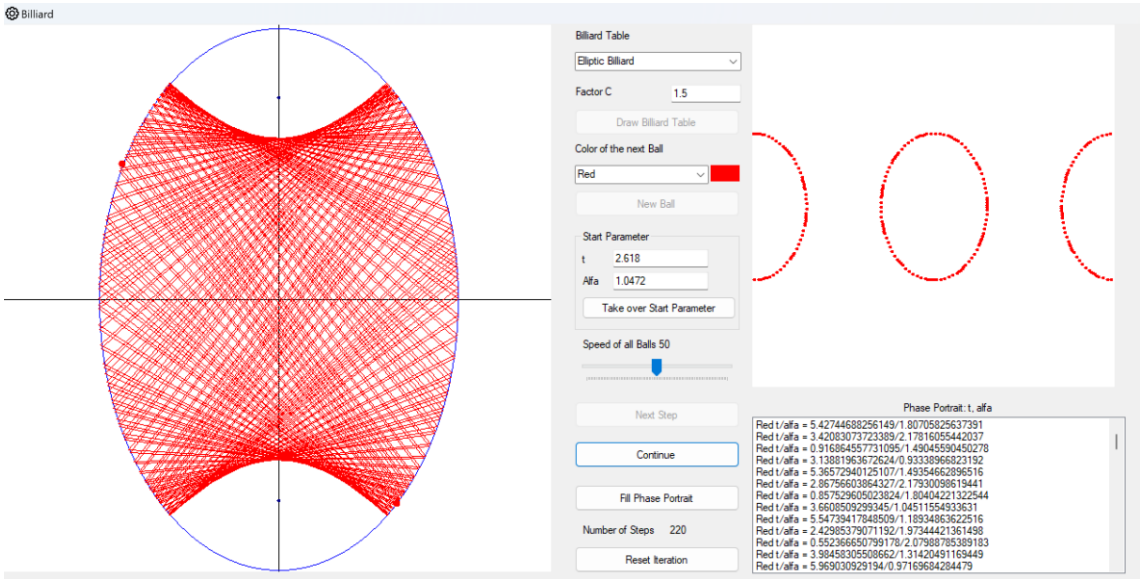
Obviously, we are close to a periodic orbit with period 40.

If we return to the first image, we see that for $C = 0.6$, the parameter *t* (always with the same starting values as before) lies in two disjoint value intervals. The corresponding image on the billiard table is:



The billiard ball track with the mentioned starting parameters and $C = 0.6$

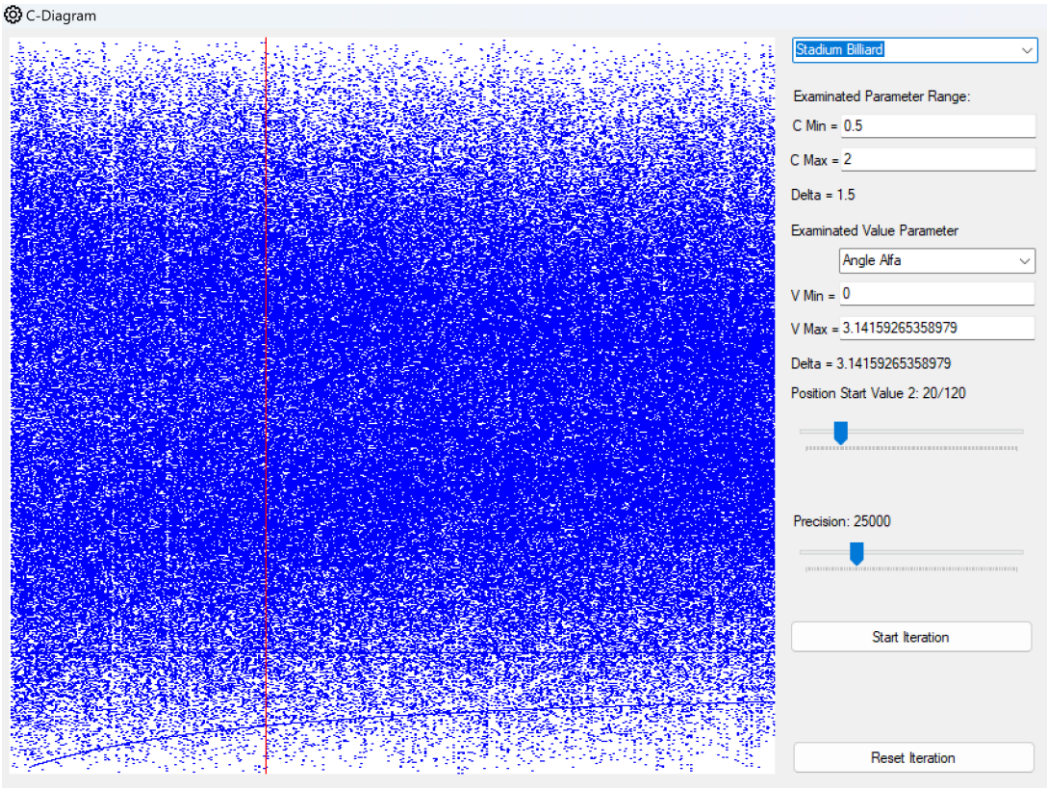
The image in the C-diagram in the case $C = 1.5$ is divided into three parts: The parameter *t* appears to move in three ranges. It should be noted that due to the periodicity of *t* and α, the C diagram is a torus, in that the lower and upper horizontal edges and the left and right vertical edges of the diagram are glued together. In this case, the parameter *t* is also in two disjoint value intervals. The picture on the billiard table confirms this.



The billiard table with the mentioned starting parameters and $C = 1.5$

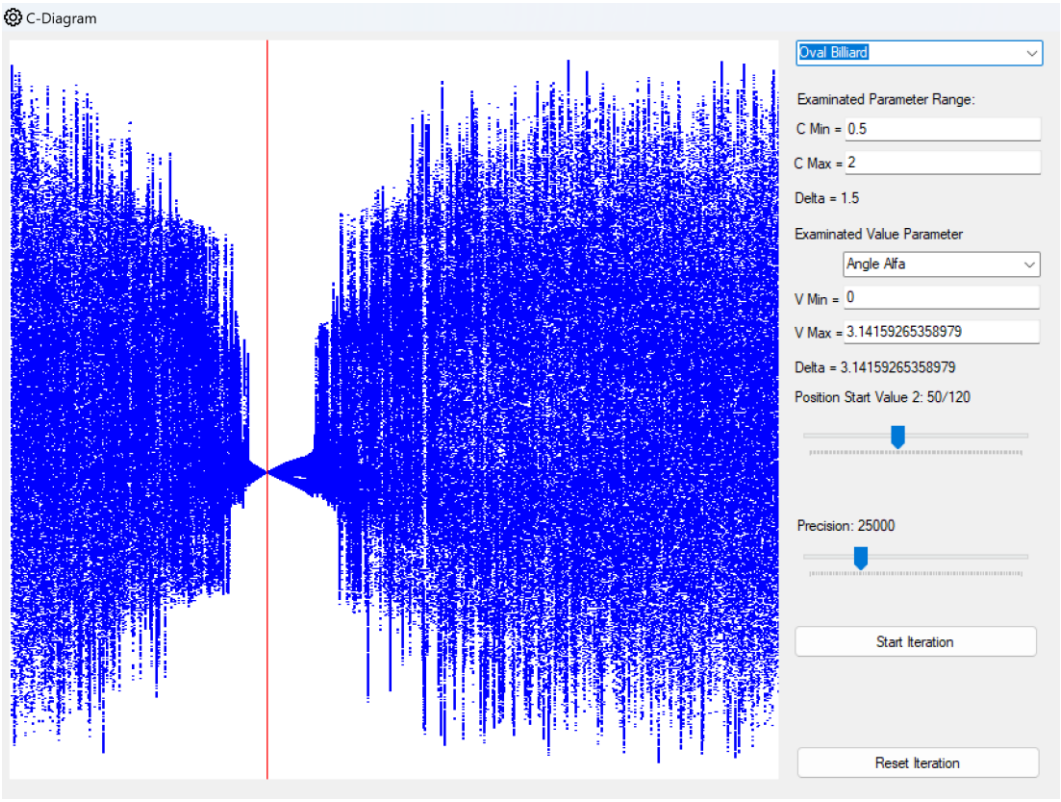
We leave further investigation on the elliptical billiard to the reader, in particular diagrams for the reflection angle α.

When playing billiards in the stadium, the C-diagram always looks quite chaotic for different starting values, as the following example illustrates.



Billiards in the stadium and diagram for the reflection angle α

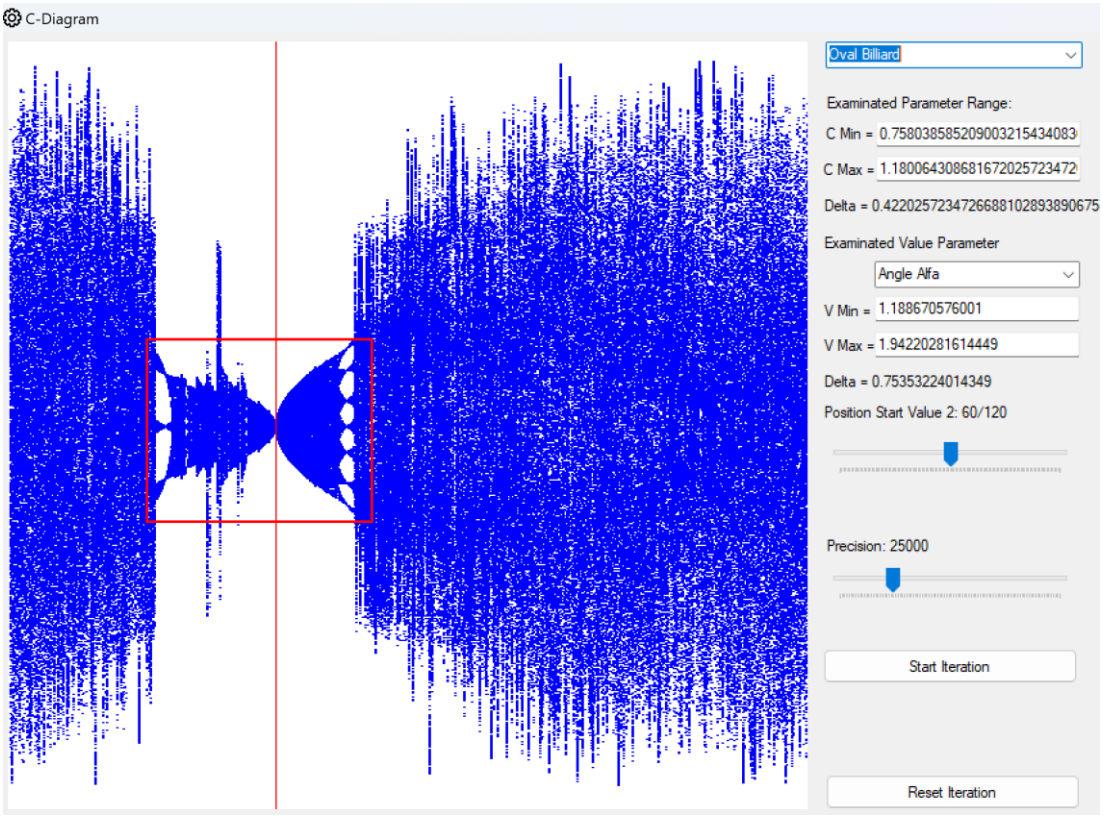
It is interesting to look at the oval billiard for the reflection angle α.



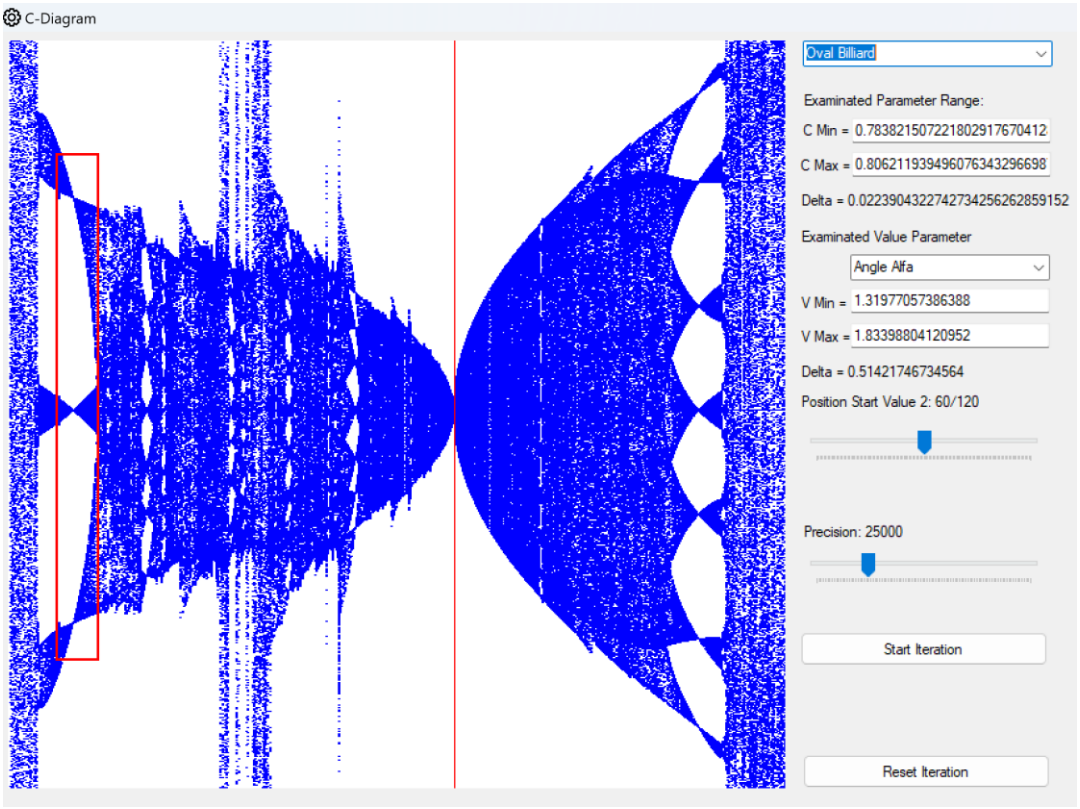
C-diagram for the oval billiard and the reflection angle α

In the area $C \approx 1$ the diagram is like the corresponding diagram for elliptical billiards, as you can see even with the "Simulator". Outside this area, it looks rather chaotic.

An interesting starting value is $\alpha = \pi \cdot \frac{6}{12}$. In the area $C \approx 1$ you can again see the similarity with the elliptical billiard. Just outside this area, however, there is a kind of periodic structure.



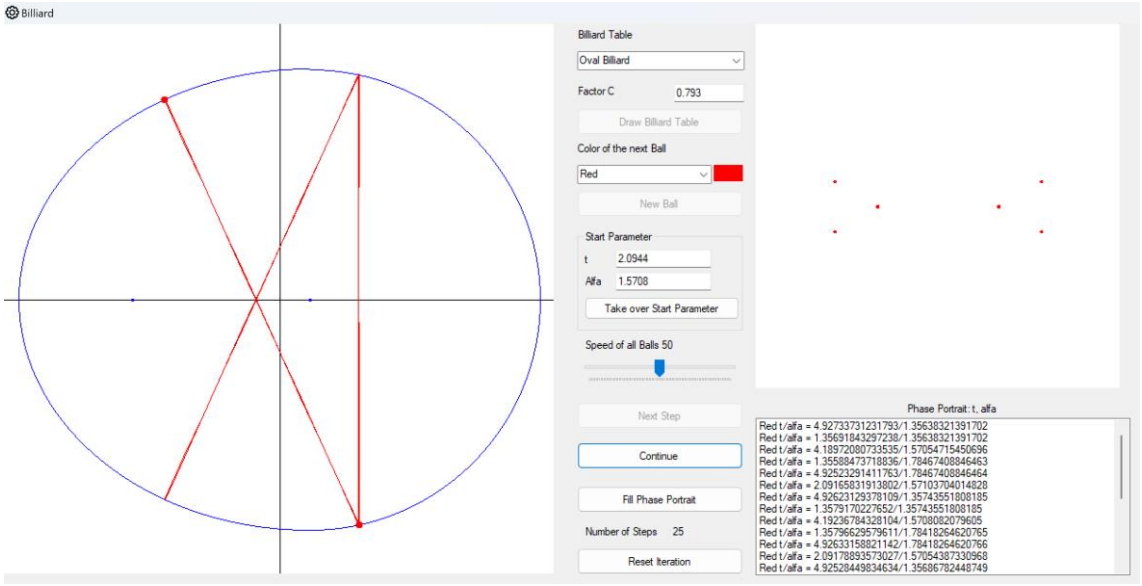
A C-diagram for the oval billiard



The enlarged section

You can see in the above image through the wider section that in the vicinity of $C \approx 0.793$ a 3-periodic orbit appears to occur.

Let's look at the oval billiard table with $C = 0.793$ and the starting parameters as in the C-diagram above: $t = 2.0944, \alpha = 1.5708$.



Path of the billiard ball with the above parameters

It is obviously a 6-period orbit.

We leave further investigations to the reader.

A mathematical analysis of the occurring images still seems to be open.

4.13. Exercise Examples

1. Examine the billiard in the rectangle. Mirror the rectangle on all sides and continue these reflections on the new rectangles. This technique is called the *unfolding method*. The ball track then becomes a straight line. What periodic paths are there? Do aperiodic paths fill the rectangle tightly? Can the path be interpreted as a path on a torus by "gluing" suitable sides of the rectangular grid together?

2. Be $\frac{p}{q}, 1 \leq p < q$ is a reduced fraction. In circular billiards, is there a ball path for each such fraction with period q and number of revolutions p?

3. In elliptical billiards, we have found an approximate starting angle α using an iteration method, which, starting from the starting point $(a, 0)$ led to a three-periodic path. Search experimentally with the help of the "Simulator" for an approximate starting angle that leads from the same starting point to a 5-periodic trajectory.

4. The hyperbola is defined as the location of all points in the plane, so that the amount of the difference between the two distances of two fixed focal points is constant. This constant is usually referred to as $2a$ analogous to the ellipse.

Be P a point on the hyperbola and F_1, F_2 are the two focal points. Then it is

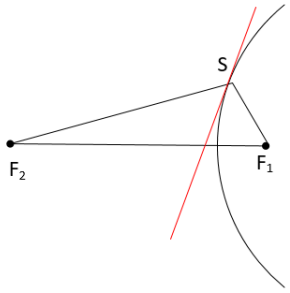
$$|PF_1| - |PF_2| = 2a$$

and

$$|PF_2| - |PF_1| = 2a$$

The hyperbola therefore has two branches.

Instead of the "reflection" of a ray emanating from a focal point, this ray is reflected at the hyperbolic tangent at the meeting point of the ray. This ray then passes through the other focal point after reflection.



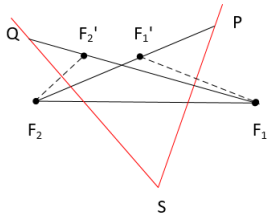
A focal beam is reflected at the hyperbolic tangent (red)

This means that in the figure above $\overline{SF_1}$ and $\overline{SF_2}$ forms the same angle with the hyperbolic tangent.

Now consider the elliptical billiard in the case that the ball trajectory passes between the focal points. Show that in this case the caustic of the trajectory is a hyperbola.

First hint: Proceed in the same way as in the case of an elliptical caustic.

If this hint does not lead to the goal, here is some further help. Look at the following figure:



Caustics for ball tracks that run between the focal points

Two consecutive ball tracks are shown in red, which meet at the point S on the elliptical billiard table. To find the points of contact of the ball track with the hyperbola you are looking for, you can proceed in the same way as we did with the ellipse as a caustic: Mirror the focal points on the respective ball track section. The points of contact you are looking for P and Q then lie on the extension of the connection between F_1F_2' or $F_1'F_2$.

Now justify that this construction really gives you the points of contact you are looking for, i.e. that e.g. $\overline{PF_2}$ forms the same angle with the right spherical path as $\overline{SF_1}$.

Furthermore, show that the triangles S, F_1, F_2' and S, F_1', F_2 are congruent. Show that this follows:

$$|F_2'F_1| = |F_1'F_2| \implies |F_1Q| - |QF_2| = |F_2P| - |PF_1|$$

This means that P and Q belong to the same hyperbola but lie on the other branch of the same hyperbola.

We have thus proved that the caustic for spherical orbits that run between the focal points is a hyperbola.

5. Consider the billiards in the stadium. A ball starts at the apex $(0, a + b)$. Explicitly state the starting angles $\alpha(k)$ for which the path of the ball has the period $2k, k \in \mathbb{N}$.

6. Look at the billiards in the stadium. A ball starts at the point $(0, b)$. Using an approximation method, find a starting angle α so that the path of the ball has period 5.

7. Discuss the periodic tracks in oval billiards.

8. Consider a billiard table bounded by a convex and differentiable curve $\gamma(t)$. Furthermore, let $\gamma(t)$ be positively oriented, i.e. the billiard table is always to the left of the curve tangent $\dot{\gamma}(t)$. Let:

$\psi_n =$ the angle between the (directed) curve tangent at the nth point of impact and the positive x-axis

$\alpha_n =$ the reflection angle at the nth hit point

$\varphi_n =$ the angle between the (directed) nth path segment and the positive x-axis

Show that applies:

$$\begin{cases} \alpha_{n+1} = \psi_{n+1} - \varphi_n \\ \varphi_{n+1} = \psi_{n+1} + \alpha_{n+1} \\ \varphi_{n+1} = 2\psi_{n+1} - \varphi_n \end{cases}$$

9. Be $\gamma: I \subset \mathbb{R} \longrightarrow \mathbb{R}^2, t \mapsto \gamma(t)$ a regular and smooth plane curve. If the curve is regarded as a function of the parameter *t*, then $\dot{\gamma}(t)$ as a (tangential) velocity vector and $\ddot{\gamma}(t)$ can be understood as acceleration. This acceleration can be divided into a vector parallel and a vector perpendicular to the velocity vector:

$$\ddot{\gamma}(t) = a^{\parallel}(t) + a^{\perp}(t)$$

Show: If γ is parameterized according to the arc length *s*, then: $a^{\parallel}(s) = 0, \forall s$.

Note:

The *arc length* of a curve is defined as:

$$L(\gamma) = \int_a^b |\dot{\gamma}(t)| dt$$

It simplifies many calculations if you select the arc length as a parameter for a curve. This is possible in the case of a regular curve. We denote this parameter by *s* and the following applies for this parameterization:

$$|\dot{\gamma}(s)| = 1, \forall s \in I$$

10. The usual parameter representation of the circle with radius r is:

$\gamma(t) = r\binom{cost}{sint}, t \in [0, 2\pi[, r > 0.$

What does a parameter representation with the arc length as a parameter explicitly look like?

11. Let a point *A* be given inside a convex billiard table. Show: Then there are at least two points of reflection *R such* that a ball starting from *A* and reflected into *R* returns to *A*.

12. Analyse the C diagram for the elliptical billiard and the angle of reflection α. Explain the value intervals that occur for α using the paths on the billiard table.

13. Analyse the C-diagram for the oval billiard and examine in particular structures that appear to be periodic.

5. Numerical methods for solving Ordinary Differential Equations

When ordinary differential equations are dealt with at the high school, this is often done in a supplementary specialization subject. This is usually limited to equations that can be solved analytically. As we will look at dynamic systems in the next section, which do not have an analytical solution but whose solution is approximated by numerical methods, this chapter is intended to introduce this topic. We will limit ourselves to simple terms and methods. For a deeper insight into the topic, sufficient literature is available. See for example [9].

Some simple numerical solution methods are explained. The "Simulator" then enables these solutions and the resulting pendulum motion to be displayed using the example of the classical spring pendulum. This allows the solution methods to be compared with each other and with the analytical solution.

5.1. Ordinary differential equations

In a differential equation, in addition to a function being searched for, there are also derivatives of this function. If the function you are looking for only depends on *one* variable, then the differential equation is called *ordinary*. For us, it is sufficient to consider ordinary differential equations *of the first order*, i.e. only the *first* derivative of the function we are looking for appears. Such a differential equation has the form:

$$y'(x) = f\big(x, y(x)\big)$$

In it is $y(x)$ is a real function. In addition, a so-called initial condition is given: $y(a) = y_0$ for $a, y_0 \in \mathbb{R}$.

Functions are often considered where the variable is the time *t.* In this case, the derivative is denoted by a point according to Newtonian notation. The first-order differential equation then looks like this:

$$\dot{y}(t) = f\big(t, y(t)\big)$$

Example

In radioactive decay, the number of atoms currently decaying depends proportionally on the number of atoms present. If $y(t)$ describes the number of atoms at time *t*, then $\dot{y}(t)$ is the change in this number at this point in time. This change is negative. The corresponding differential equation is

$$\dot{y}(t) = -\lambda y(t), \lambda > 0$$

If the decay starts at time $t = 0$ and there are y_0 atoms are present at the beginning, the initial condition is $y(0) = y_0$.

As you can easily see, this differential equation has the solution:

$$y(t) = y_0 e^{-\lambda t}$$

There are (in practice rarely occurring) cases of ordinary differential equations which have analytical solutions that can be determined using various (often tricky) methods. However, this is not the topic here.

There are three central questions in connection with ordinary differential equations:

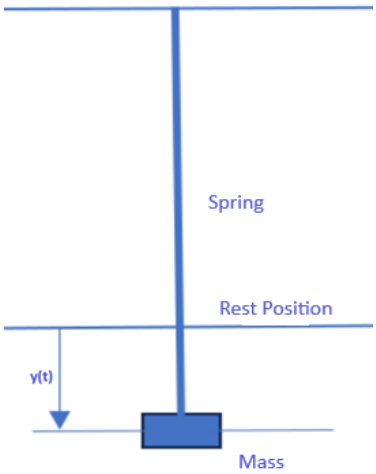
1) Is there a solution? That is, is there a function $y(x)$ which satisfies the differential equation at least in an area $x \in [a, b]$?

2) Is this solution uniquely determined by an initial condition $y(a) = y_0$?
3) Is the solution stable? This means that a small disturbance of the initial condition has little effect in the long term.

We will not discuss these questions here, but simply point out that the answer is not trivial. The third question is particularly important if the solution is only determined approximately using numerical methods.

5.2. The classic Spring Pendulum

When we later discuss numerical methods for solving ordinary differential equations, we will apply them to a spring pendulum and compare the result with the analytical solution of its equation of motion.



Spring pendulum with deflection from the rest position $y(t)$ at the time t

At rest, the spring force and gravity acting on the mass cancel each other out. If the mass is deflected by the distance $y(t)$ the (retracting) spring force increases in proportion to this deflection. The proportionality factor depends on the nature of the spring, and we denote it by *D*. The force of gravity does not change. The spring force therefore acts on the mass

$$F_{Spring} = -Dy(t)$$

This accelerates the mass. The acceleration is $\ddot{y}(t)$ and if we denote the mass by *m, the following* applies according to Newton:

$$m\ddot{y}(t) = -Dy(t)$$

Since we only want to consider an idealized pendulum, we set $m = D = 1$. This gives us the differential equation:

$$\ddot{y}(t) + y(t) = 0$$

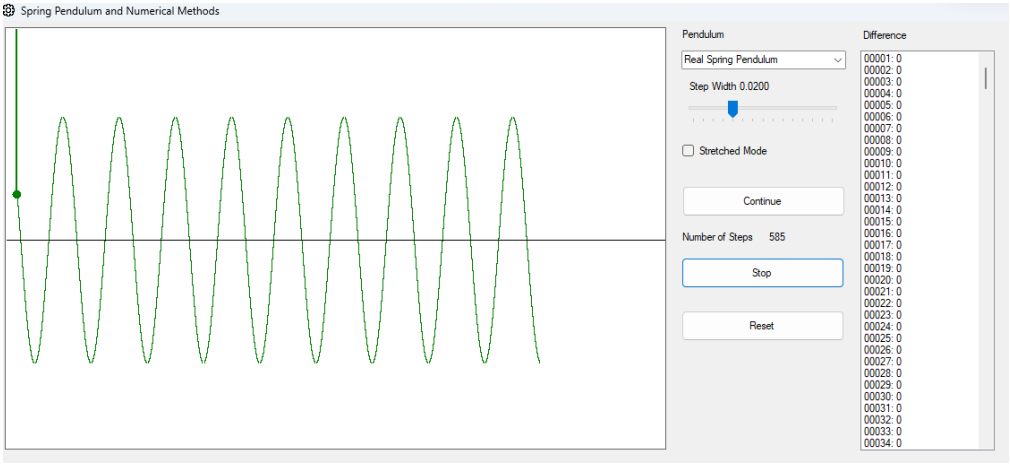
We also assume that the pendulum is deflected from its rest position by the distance y_0 from its rest position and is then released. The initial condition therefore applies: $y(0) = y_0$.

As you can easily check

$$y(t) = y_0 \cos(t)$$

A solution of this differential equation and this is uniquely determined (without us justifying this in more detail here).

In the “Simulator”, you drag the spring pendulum to the starting position with the mouse and then you can see what the oscillation looks like:



Oscillation of a spring pendulum

The time axis is slightly compressed as it depends on the speed at which the diagram is shifted to the right in the implementation. However, this applies to all pendulums to the same extent and does not affect the comparison of different methods.

5.3. The Forward Euler Method

If the differential equation has no analytical solution, you try to find the function you are looking for step by step and approximately by replacing the differential quotient $y'(x) = \frac{dy}{dy}$ with the difference quotient: $\frac{dy}{dx} \approx \frac{\Delta y}{\Delta x}$ where Δx is chosen small enough. Then set:

$$\frac{\Delta y}{\Delta x} = f(x, y(x)) \text{ resp. } \Delta y = f(x, y(x))\Delta x$$

The “Forward Euler" method now works as follows:

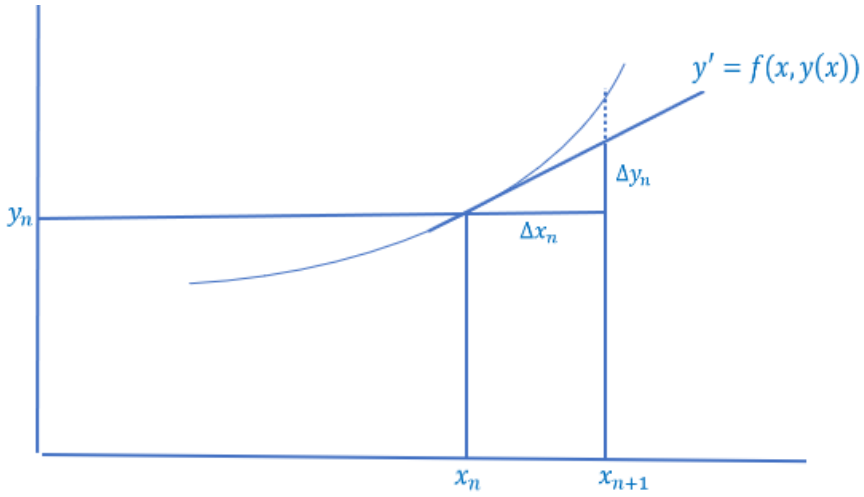
Starting from an initial condition

$$\begin{cases} x_1 = 0 \\ y_1 = y_0 \end{cases}$$

If you iteratively determine further values of *x* and $y(x)$ using the recursion formula:

$$\begin{cases} x_{n+1} = x_n + \Delta x_n \\ y_{n+1} = y_n + \Delta y_n = y_n + f(x_n, y_n)\Delta x_n \end{cases}$$

This means that we use a linear approximation to approximate the next function value using the tangent gradient at the point (x_n, y_n).



Sketch of the Forward Euler method

In the general case, the step size Δx_n can vary for each step. It could be adapted to the expected shape of the function and there are various methods for this. This is referred to as an *adaptive* approximation method.

For our purposes, however, we will choose a constant step size: $\Delta x_n := h, \forall n$ for a small positive *h*. Thus, the recursion formula becomes:

$$\begin{cases} x_{n+1} = x_n + h \\ y_{n+1} = y_n + \Delta y_n = y_n + f(x_n, y_n)h \end{cases}$$

This method is called *explicit* because the variables to be determined at each iteration step (x_{n+1}, y_{n+1}) are already on the left-hand side of the equations.

Now we want to apply this method to the spring pendulum. The corresponding differential equation is

$$\ddot{y}(t) + y(t) = 0$$

This is a second-order differential equation, but we can convert it into a first-order differential equation system by substituting:

$$\begin{cases} u(t) := y(t) \\ v(t) := y'(t) \end{cases}$$

This leads to the following system of differential equations:

$$\begin{cases} u'(t) := v(t) \\ v'(t) := -u(t) \end{cases}$$

The previous function $f(x, y(x))$ consists of two components:

$$\begin{cases} f_1\big(t, u(t), v(t)\big) = v(t) \\ f_2\big(t, u(t), v(t)\big) = -u(t) \end{cases}$$

The initial conditions are:

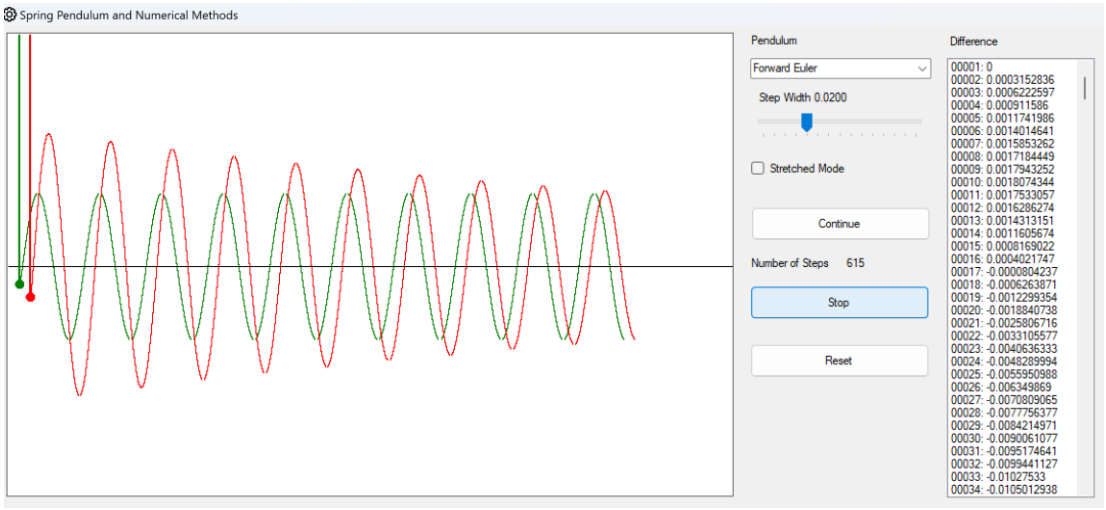
$$\begin{cases} t_1 = 0 \\ u_1 = y_0 \\ v_1 = 0 \end{cases}$$

It is $v_1 = 0$, because the pendulum has no speed in the starting position.

Now choose a small $h > 0$ as a constant increment for the growth of t and apply the Forward Euler method to the components $u(t), v(t)$ components. This provides the recursion formulas:

$$\begin{cases} t_{n+1} = t_n + h \\ u_{n+1} = u_n + v_n h \\ v_{n+1} = v_n - u_n h \end{cases}$$

We can now use the "Simulator" to check how this method behaves compared to the analytical solution for the spring pendulum. We can adjust the step size in the "Simulator". Implementation details will follow in a later section.



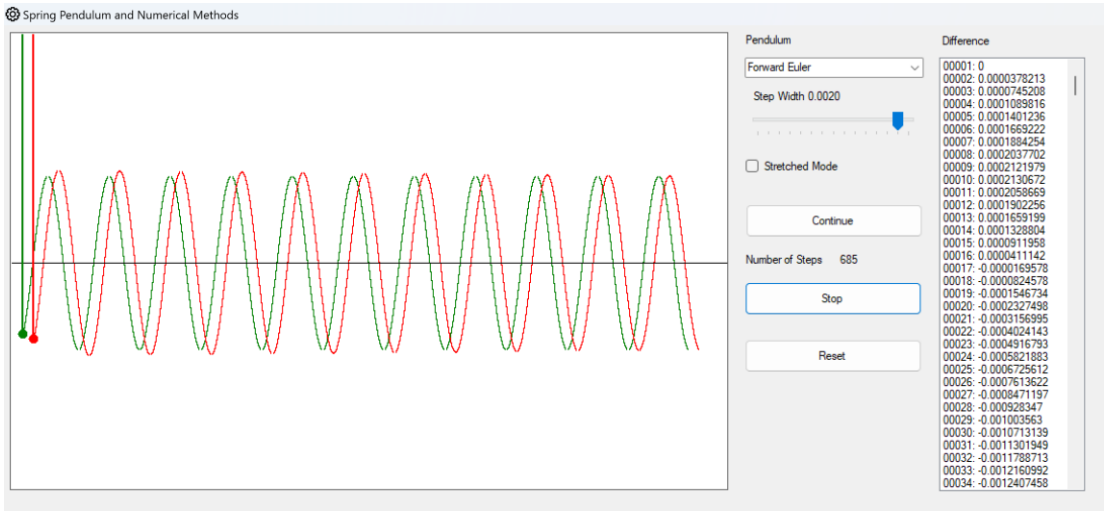
Comparison of the analytical solution with "Forward Euler"

In the picture above, the pendulum, which swings according to the analytical solution, is shown in green. The numerical approximation according to the Forward Euler method is shown in red. The method can be selected at the top right. The difference between the two oscillations at each iteration step is also listed on the right-hand side. At the start, both pendulums are moved to the same start position by holding down the left mouse button.

As you can see, the numerical approximation of the oscillation frequency is quite good, but the amplitude is "out of control".

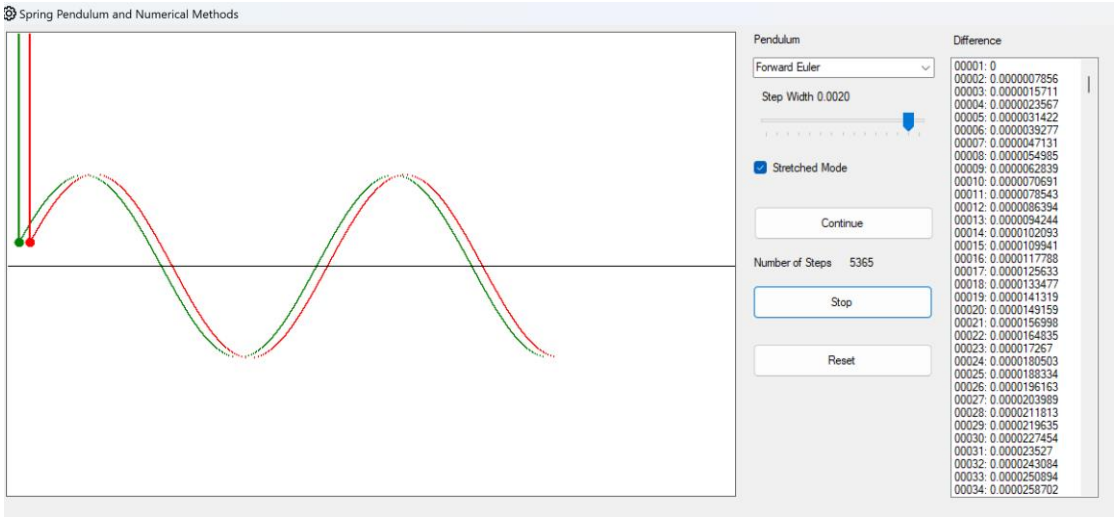
We can set the step size down. The step size for the green pendulum is constant (in the "Simulator" equal to 0.1). For the red pendulum, smaller increments of the form $\frac{0.1}{k}, k \in \mathbb{N}$ are possible. This means that the red pendulum swings k *times* until its position is recorded. This effects that the pendulums are synchronized.

Below is an example with a step size of 0.002 for the red pendulum. The effect of the escalating amplitude is still present. However, it takes longer for it to take effect.



The red pendulum swings with a smaller step width

The "Simulator" also allows *both* pendulums to swing with the same smaller increment. To do this, the *Stretched display* option is activated. This provides a kind of "zoom" on the oscillation pattern:



Both pendulums swing with the same increment

5.4. The Backward Euler Method

In the Forward Euler method, we had the recursion formula for the approximation:

$$\begin{cases} x_{n+1} = x_n + h \\ y_{n+1} = y_n + \Delta y_n = y_n + f(x_n, y_n)h \end{cases}$$

In the “Backward Euler” method, the approximation is not based on the tangent gradient at the point (x_n, y_n), but that at the point to be determined (x_{n+1}, y_{n+1}). The iteration formula is therefore:

$$\begin{cases} x_{n+1} = x_n + h \\ y_{n+1} = y_n + \Delta y_n = y_n + f(x_{n+1}, y_{n+1})h \end{cases}$$

This means that the variables being searched for are *implicit* in the system of equations, which must be solved for these variables. It is not guaranteed that this is even possible. However, we only want to point out this problem. In the case of the spring pendulum, this is possible, as we will see.

The differential equation system to be solved is still the same:

$$\begin{cases} u'(t) \coloneqq v(t) \\ v'(t) \coloneqq -u(t) \end{cases}$$

But instead of the recursion formulas

$$\begin{cases} t_{n+1} = t_n + h \\ u_{n+1} = u_n + v_n h \\ v_{n+1} = v_n - u_n h \end{cases}$$

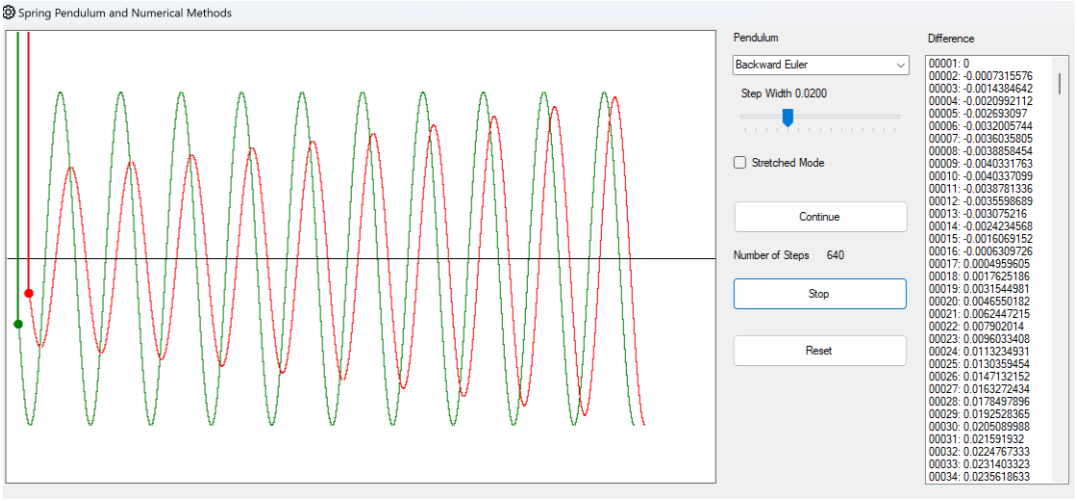
of the Forward Euler method, we have the formulas for the Backward Euler method:

$$\begin{cases} t_{n+1} = t_n + h \\ u_{n+1} = u_n + v_{n+1} h \\ v_{n+1} = v_n - u_{n+1} h \end{cases}$$

If you solve this system of equations for the required variables (u_{n+1}, v_{n+1}) you get (as you can easily calculate) the formulas:

$$\begin{cases} t_{n+1} = t_n + h \\ u_{n+1} = (u_n + v_n h)/(1 + h^2) \\ v_{n+1} = (v_n - u_n h)/(1 + h^2) \end{cases}$$

The divisor $(1 + h^2) > 1$ has a damping effect on the amplitude. This is confirmed when we look at the result in the "Simulator".



The Backward Euler method applied to the spring pendulum

5.5. The implicit Midpoint Rule

With the Forward Euler method, the amplitude has increased too much. With the Backward Euler method, it was damped too much. The question is whether you can find "something in the middle" of both methods.

This means that neither the tangent gradient at the point (x_n, y_n) nor at the point (x_{n+1}, y_{n+1}) for the approximation, but the tangent gradient at the point $(x_n + \frac{h}{2}, y_n + \frac{\Delta y_n}{2})$.

We designate: $\hat{y}_n \coloneqq y_n + \frac{\Delta y_n}{2}$.

Then the equation for determining $\hat{y}_n$:

$$\hat{y}_n = y_n + \frac{h}{2} f(x_n + \frac{h}{2}, \hat{y}_n)$$

that is, we use the tangent gradient at the point $(x_n + \frac{h}{2}, \hat{y}_n)$.

With this value we then calculate y_{n+1} using this tangent gradient again:

$$y_{n+1} = y_n + hf(x_n + \frac{h}{2}, \hat{y}_n)$$

As you can see, this is an implicit method because the first equation must be solved for $\hat{y}_n$ must be solved.

Now we apply this method to the spring pendulum. The starting point is again the system of differential equations:

$$\begin{cases} u'(t) \coloneqq v(t) \\ v'(t) \coloneqq -u(t) \end{cases}$$

First, we calculate $(\hat{u}_n, \hat{v}_n)$ according to the formula above:

$$\begin{cases} \hat{u}_n = u_n + \frac{h}{2} f_1\left(t + \frac{h}{2}, \hat{u}_n, \hat{v}_n\right) = u_n + \frac{h}{2}\hat{v}_n \\ \hat{v}_n = v_n + \frac{h}{2} f_2\left(t + \frac{h}{2}, \hat{u}_n, \hat{v}_n\right) = v_n - \frac{h}{2}\hat{u}_n \end{cases}$$

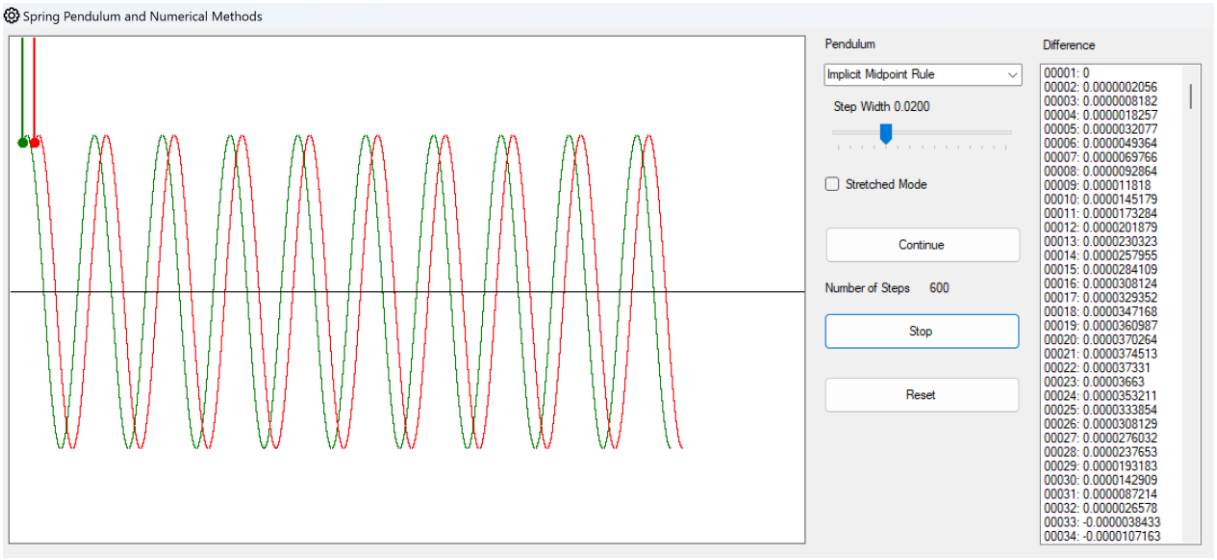
Solving the system of equations according to $(\hat{u}_n, \hat{v}_n)$ you get

$$\begin{cases} \hat{u}_n = (u_n + \frac{h}{2} v_n)/(1 + {h^2}/{4}) \\ \hat{v}_n = (v_n - \frac{h}{2} u_n)/(1 + {h^2}/{4}) \end{cases}$$

With these values we calculate (u_{n+1}, v_{n+1}):

$$\begin{cases} u_{n+1} = u_n + hf_1\left(t + \frac{h}{2}, \hat{u}_n, \hat{v}_n\right) = u_n + h\hat{v}_n = u_n + h(v_n - \frac{h}{2} u_n)/(1 + {h^2}/{4}) \\ v_{n+1} = v_n + hf_2\left(t + \frac{h}{2}, \hat{u}_n, \hat{v}_n\right) = v_n - h\hat{u}_n = v_n - h(u_n + \frac{h}{2} v_n)/(1 + {h^2}/{4}) \end{cases}$$

This recursion is implemented in the "Simulator" and now we are looking forward to the result:



The red pendulum is approximated by the implicit midpoint rule

As the picture shows, it looks pretty good. However, this has nothing to do with the fact that the implicit midpoint rule is particularly good and suitable for practical applications. It is simply that this rule provides a good approximation in the case of the spring pendulum or similar problems. However, as you can see in the list of differences on the right, these are not zero.

5.6. The fourth-order Runge-Kutta Method

With the implicit midpoint rule, we have the tangent gradient at a point between (x_n, y_n) and (x_{n+1}, y_{n+1}) for the approximation. An idea now is to use the tangent slope at different points near (x_n, y_n) and (x_{n+1}, y_{n+1}) and then use a weighted average of these tangent slopes for the approximation. This idea is used by the Runge-Kutta method, named after its developers Carl Runge and Martin Wilhelm Kutta at the beginning of the 20th century.

How to select the interpolation points for the tangent slopes and the weighted average of these can be found in the literature, for example in [9]. We only want to show the procedure here, namely one of order four.

As always, the starting point is a differential equation of the form:

$$y'(x) = f\big(x, y(x)\big)$$

and a recursion formula for the four-step procedure:

$$y_{n+1} = y_n + h \sum_{j=1}^{4} b_j k_j$$

In it are k_j the tangent slopes at certain points and b_j are the coefficients of the weighted mean, i.e. the following applies: $\sum_{j=1}^{4} b_j = 1$.

The coefficients k_j for the tangent slopes are now defined as:

$$k_1 = f(x_n, y_n)$$

$$k_2 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2} k_1\right)$$

$$k_3 = f\left(x_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right)$$

$$k_4 = f(x_n + h, y_n + hk_3)$$

This forms the weighted average for the approximation, which is as follows:

$$\begin{cases} x_{n+1} = x_n + h \\ y_{n+1} = y_n + h(k_1 + 2k_2 + 2k_3 + k_4)/6 \end{cases}$$

Now we apply this again to the spring pendulum. The corresponding system of differential equations was given by:

$$\begin{cases} u'(t) = f_1\big(t, u(t), v(t)\big) = v(t) \\ v'(t) = f_2\big(t, u(t), v(t)\big) = -u(t) \end{cases}$$

Since we have two components, we need two sets of coefficients for the tangent slopes. We denote them by k_j for the first component and l_j for the second.

That's what we have first:

$$\begin{cases} k_1 = f_1(t_n, u_n, v_n) = v_n \\ l_1 = f_2(t_n, u_n, v_n) = -u_n \end{cases}$$

$$\begin{cases} k_2 = f_1\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_1, v_n + \frac{h}{2}l_1\right) = v_n + \frac{h}{2}l_1 \\ l_2 = f_2\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_1, v_n + \frac{h}{2}l_1\right) = -u_n - \frac{h}{2}k_1 \end{cases}$$

$$\begin{cases} k_3 = f_1\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_2, v_n + \frac{h}{2}l_2\right) = v_n + \frac{h}{2}l_2 \\ l_3 = f_2\left(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_2, v_n + \frac{h}{2}l_2\right) = -u_n - \frac{h}{2}k_2 \end{cases}$$

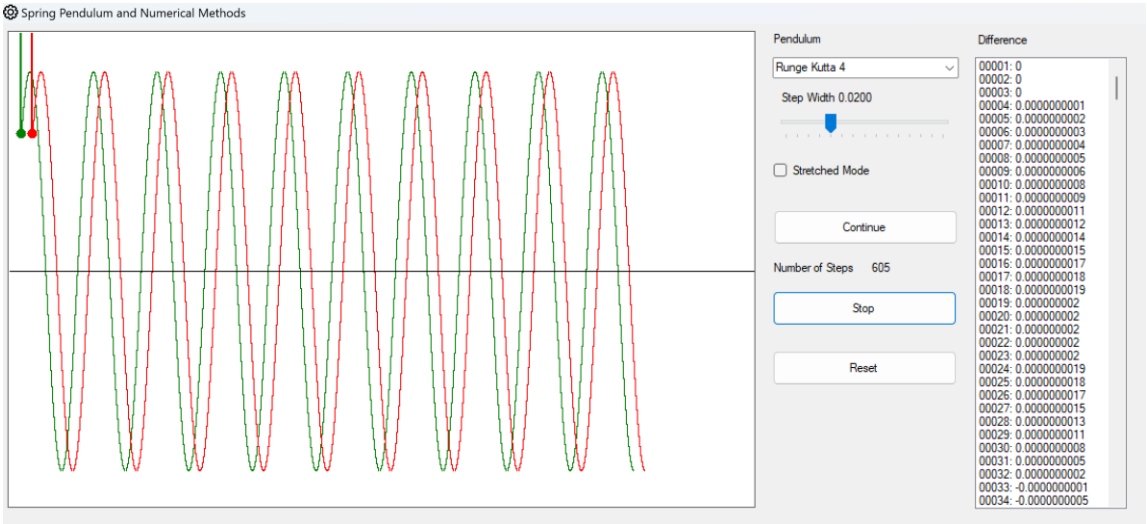
$$\begin{cases} k_4 = f_1(t_n + h, u_n + hk_4, v_n + hl_4) = v_n + hl_4 \\ l_4 = f_2(t_n + h, u_n + hk_4, v_n + hl_4) = -u_n - hk_4 \end{cases}$$

This gives you the recursion formulas:

$$\begin{cases} t_{n+1} = t_n + h \\ u_{n+1} = u_n + h(k_1 + 2k_2 + 2k_3 + k_4)/6 \\ v_{n+1} = v_n + h(l_1 + 2l_2 + 2l_3 + l_4)/6 \end{cases}$$

As you can see, this is an explicit procedure.

It is implemented in the "Simulator". The result is as follows:



The red pendulum is approximated by the Runge-Kutta method of order four

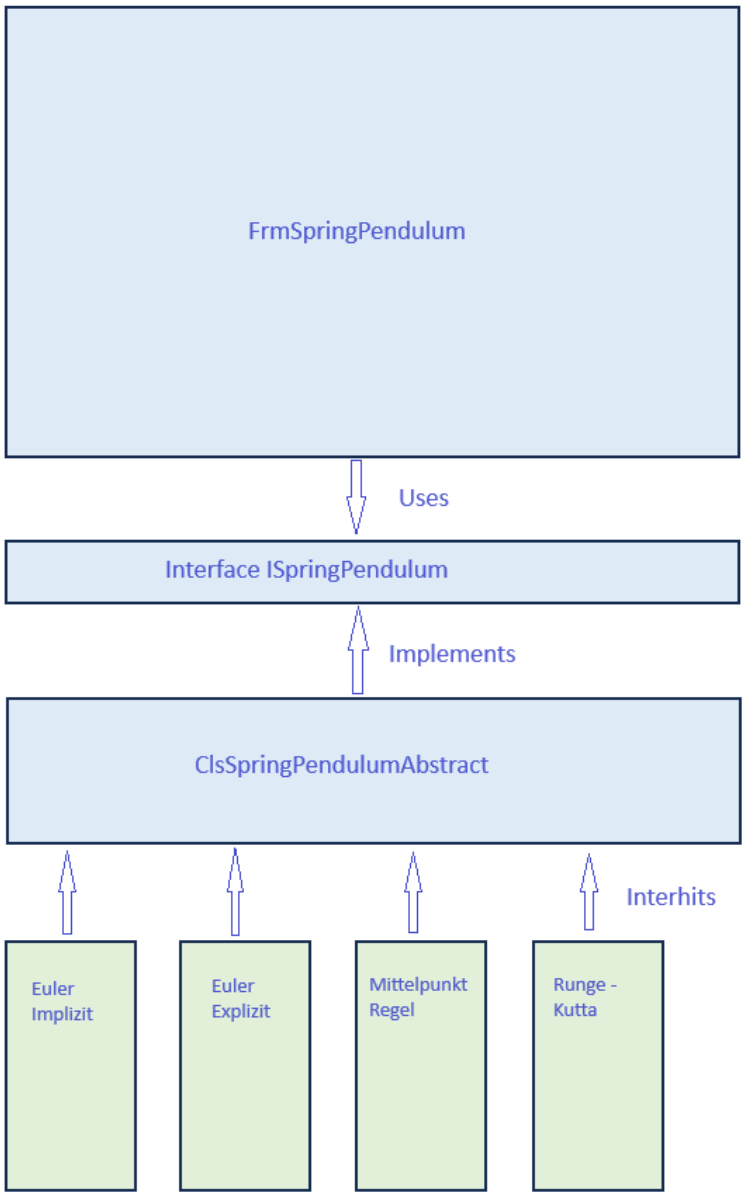
This approximation is also visually quite good. On the right-hand side, you can see in the list of differences that these have become significantly smaller.

We will use this Runge-Kutta method for the approximation in the next chapter for the simulation of different types of pendulums because it is easy to implement and at least elementary understandable from the idea. It has a good balance between accuracy and computational effort. It does not require higher derivatives of the function than the first one. In addition, the error is proportional to the fifth power of the step size, which is sufficient for our purposes.

In practice, there are much more efficient methods, some of which are multi-step methods or adaptive methods with an adapted step size per step. Anyone interested in this should refer to the extensive literature. MATLAB also offers a whole range of practical methods, which are often used by engineers, with its ODE (= Ordinary Differential Equation) suite.

5.7. Implementation in the "Simulator"

The architecture of the "Numerical methods" area is based on the same principles as the previous implementations: It should be easy for users to implement and try out their own numerical methods without disturbing the existing code. The structure is as follows:



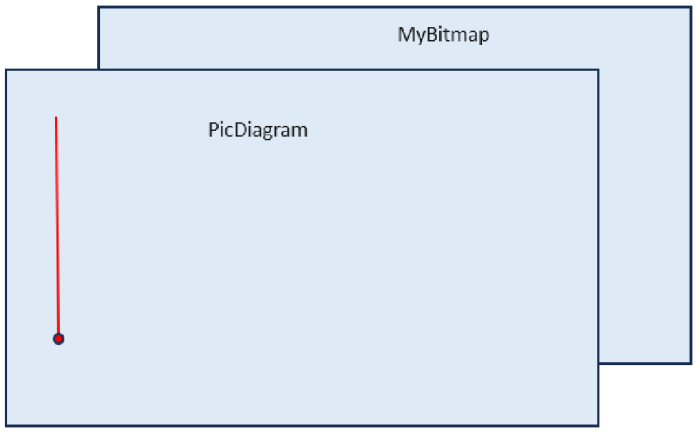
Architecture for the implementation of the spring pendulum

The user can try out his own numerical method by writing another class with this method and inheriting it from the abstract class *ClsSpringPendulumAbstract.* All he must do is implement the *Protected MustOverride Sub Iteration* routine of this class with a *Protected Overrides Sub Iteration* routine. This then contains the details of the numerical procedure.

The only tricky part is pushing the pendulum track to the right.

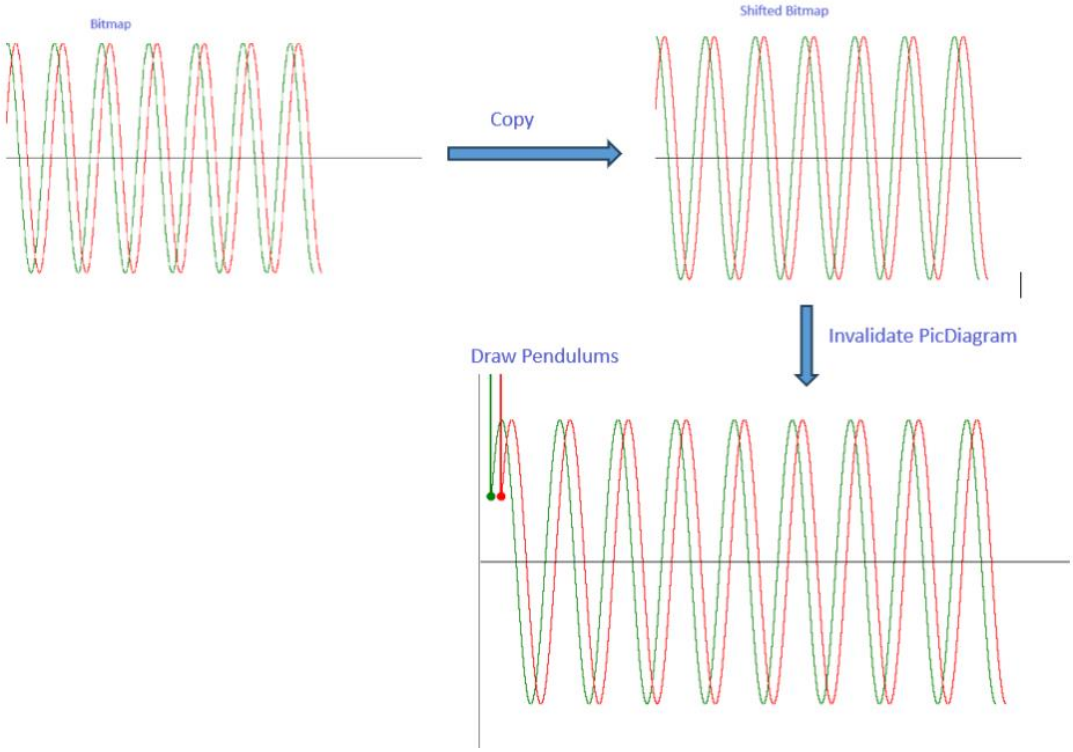
The *PicDiagram* picture box records the current position of the pendulum. This means that it is emptied at each iteration step and the pendulum is redrawn.

PicDiagram contains a bitmap *MyBitmap* in which the position of the pendulum is permanently drawn, resulting in the recording of its path.



PicDiagram and the underlying MyBitmap

Proceed as follows to ensure that "shifting to the right" works:



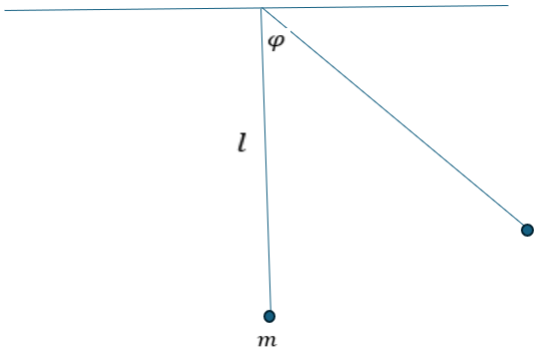
The current bitmap is copied and then emptied. The copy of the bitmap is then inserted back into the original bitmap, but with a 1-pixel shift to the right. *PicDiagram.Invalidate* transfers the shifted bitmap to the PicDiagram. Finally, the current position of the pendulums is drawn into the PicDiagram.

5.8. Exercise examples

1. With the explicit Euler method, the amplitude was damped too little, with the implicit Euler method too much. What happens if you average both methods? Implement a corresponding method in the "Simulator" and analyse the result. (See notes in the last chapter "Implementing your own variants").

2. Consider the differential equation $y' = y$ with the initial condition $y(0) = 1$. Explicitly determine the recursion formulae for the explicit and implicit Euler method and for the centre point rule.

3. Consider a thread pendulum with thread length l and pendulum mass m. Let the angle of deflection be $\varphi(t)$ where *t* is the time. g is the gravitational constant near the earth.



Thread pendulum

Derive the equation of motion: $\ddot{\varphi} + \omega^2 sin\varphi = 0$. Where $\omega = \sqrt{\frac{g}{l}}$.

Transform the equation of motion $u = \varphi, v = \dot{\varphi}$ into a system of two first-order differential equations. Then determine the recursion formulae for the centre point rule.

6. Coupled pendulums

6.1. The Lagrange formalism

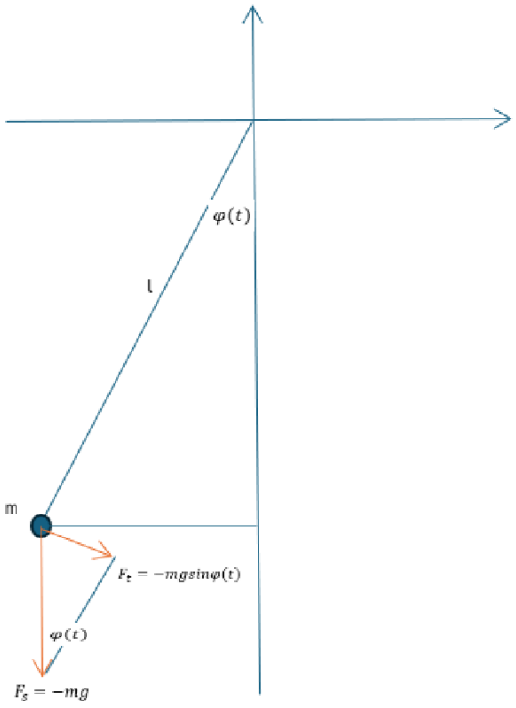
In the following sections, we will look at some examples of coupled pendulums and implement their dynamics in the "Simulator". Deriving the corresponding equations of motion on the basis of Newton's laws of force is quite difficult or even impossible. We will use the so-called Lagrange formalism for this purpose. Its derivation from Hamilton's principle is possible on the basis of high school maths with some additional effort if you know the concept of the partial derivative and the total differential and use the technique of partial integration. It can also be interesting to delve into the calculus of variations with simple examples. A good introduction can be found in [10], for example.

However, we do not want to justify the Lagrange formalism here and only introduce it to the extent that we can use it in the following sections. We will do this using an example.

The thread pendulum

Variant 1: The standard solution

The usual way to derive the equation of motion for the thread pendulum is as follows:



Thread pendulum

A mass m hangs on a (massless) thread of length l. The force of gravity acts on the mass $F_s = -mg$ in the negative y-direction. In this chapter, g is always the gravitational acceleration near the earth. This is slightly different at the equator and at the poles, but this is irrelevant for our investigations. We always calculate with $g = 9.8\, ms^{-2}$.

At time *t,* the deflection of the pendulum is $\varphi(t)$. Only the tangential force is relevant for the movement of the mass $F_t = -mgsin\varphi(t)$. The acceleration in the tangential direction is $l\ddot{\varphi}(t)$. This is Newton's law of motion:

$$ml\ddot{\varphi} = -mgsin\varphi$$

And you get the equation of motion:

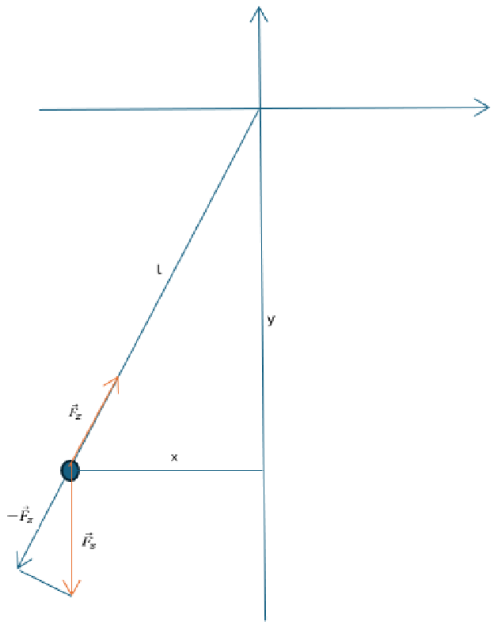
$$l\ddot{\varphi} + gsin\varphi = 0$$

In this first variant, suitable coordinates were selected and the constraining force emanating from the thread was eliminated because only the tangential force was taken into account for the acceleration of the mass.

For an explicit solution of this equation, two additional initial conditions are required, for example if the pendulum is held from an initial position with the angle φ_0 and then released: $\varphi(0) = \varphi_0 > 0$, $\dot{\varphi}(0) = 0$

Variant 2: Explicit consideration of the constraining force emanating from the thread

It could be that you are not always so lucky with the choice of the coordinate system and the eliminating constraining force. What does the derivation look like if you explicitly take the latter into account and use a Cartesian coordinate system?



Thread pendulum in Cartesian coordinates

The position vector of the mass point at time t is $\vec{r}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix}$. The force of gravity acting on the point of mass is $\vec{F}_s = \begin{pmatrix} 0 \\ -mg \end{pmatrix}$. Furthermore, the constraint applies: $x^2 + y^2 = l^2$.

For the amount of the "constraining force" $\vec{F}_z$ which is exerted by the thread and ensures that the pendulum has a constant distance from the suspension point, the following applies due to the similarity of the two triangles in the sketch:

$$\frac{|\vec{F}_z|}{|\vec{F}_s|} = \frac{|y|}{l}$$

Their direction is opposite to the position vector, i.e. points in the direction of the unit vector $\frac{-\vec{r}(t)}{l}$. The following therefore applies to the constraining force:

$$\vec{F}_z = mg\frac{-y}{l}\cdot\frac{1}{l}\begin{pmatrix}-x\\-y\end{pmatrix} = \frac{mgy}{l^2}\begin{pmatrix}x\\y\end{pmatrix}$$

The total force acting on the mass is then:

$$\vec{F}_{total} = \begin{pmatrix}0\\-mg\end{pmatrix} + \frac{mgy}{l^2}\begin{pmatrix}x\\y\end{pmatrix} = \frac{mg}{l^2}\begin{pmatrix}xy\\-l^2+y^2\end{pmatrix} = \frac{mg}{l^2}\begin{pmatrix}xy\\-x^2\end{pmatrix}$$

This is Newton's law of motion:

$$\begin{cases} l^2\ddot{x} = gxy \\ l^2\ddot{y} = -gx^2 \end{cases}$$

Substituting *y* in the first equation by $y = -\sqrt{l^2 - x^2}$ (in our sketch is $y < 0$), we obtain the system of differential equations:

$$(*)\begin{cases} l^2\ddot{x} = -gx\sqrt{l^2 - x^2} \\ l^2\ddot{y} = -gx^2 \end{cases}$$

In order to make progress here, the approach $x(t) = l \cdot sin\varphi(t)$ is obvious. Because of the constraint, then $y(t) = -l \cdot cos\varphi(t)$where in both cases $\varphi(t)$ is a function yet to be determined.

To eliminate *x* and *y* in the system of equations, we also calculate the derivatives:

$$\dot{x} = l\dot{\varphi}cos\varphi, \ddot{x} = l\ddot{\varphi}cos\varphi - l\dot{\varphi}^2 sin\varphi$$

$$\dot{y} = l\dot{\varphi}sin\varphi, \ddot{y} = l\ddot{\varphi}sin\varphi + l\dot{\varphi}^2 cos\varphi$$

We put everything in (*) and receive:

$$\begin{cases} l^3\ddot{\varphi}cos\varphi - l^3\dot{\varphi}^2 sin\varphi = -glsin\varphi \cdot lcos\varphi \\ l^3\ddot{\varphi}sin\varphi + l^3\dot{\varphi}^2 cos\varphi = -gl^2 sin^2\varphi \end{cases}$$

Now we move everything to the left-hand side and divide the first equation by $l^2 sin\varphi$. We divide the second equation by $l^2 cos\varphi$ (without specifically naming the cases where these expressions are zero). Then we get the equations:

$$\begin{cases} l\ddot{\varphi}\dfrac{cos\varphi}{sin\varphi} - l\dot{\varphi}^2 + gcos\varphi = 0 \\ l\ddot{\varphi}\dfrac{sin\varphi}{cos\varphi} + l\dot{\varphi}^2 + g\dfrac{sin^2\varphi}{cos\varphi} = 0 \end{cases}$$

If you add both equations together, the result is

$$l\ddot{\varphi}\left(\frac{cos\varphi}{sin\varphi} + \frac{sin\varphi}{cos\varphi}\right) + g\left(cos\varphi + \frac{sin^2\varphi}{cos\varphi}\right) = 0$$

Now we multiply this equation by the factor $sin\varphi \cdot cos\varphi$ and get

$$l\ddot{\varphi}(cos^2\varphi + sin^2\varphi) + g \cdot sin\varphi(cos^2\varphi + sin^2\varphi) = 0$$

This again provides the familiar equation:

$$l\ddot{\varphi} + gsin\varphi = 0$$

However, the effort involved was much greater.

Insert: The Lagrange formalism

The Lagrange formalism makes it possible to work with parameters that uniquely describe a system but can otherwise be chosen arbitrarily. This is referred to as *generalised coordinates*. Ideally (and this will be the case in the following sections), these can be chosen in such a way that no constraints have to be taken into account, but that these are already implicitly taken into account with the choice of generalised coordinates. Then the number of these coordinates corresponds to the number of independent degrees of freedom of a system.

The generalised coordinates are generally time-dependent and are referred to as $q_1(t), q_2(t), q_3(t), \ldots$. The Cartesian coordinates (of perhaps several mass points in the system) then depend on the generalised coordinates. The following then applies to the i-th Cartesian coordinate:

$$x_i(t) = x_i(q_1(t), q_2(t), \ldots) = x_i(\vec{q})$$

It will suffice for us to consider only systems in which the external (i.e. non-coercive) forces can be derived from a potential $V(\vec{q})$ in which the energy theorem applies.

The Lagrange function is now defined as the difference between kinetic energy *T* and potential energy *V*. It generally depends on $\vec{q}, \dot{\vec{q}}$ and *t*.

$$L\left(\vec{q}, \dot{\vec{q}}, t\right) = T - V$$

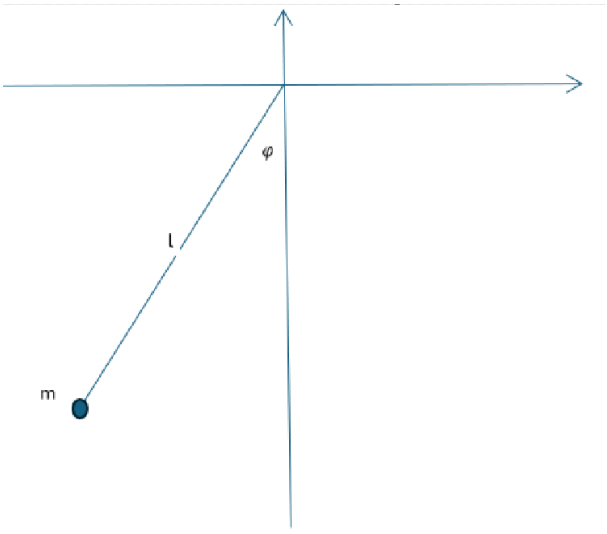
Since a potential *V* is only determined up to a constant, this also applies to the Lagrange function.

A mass point will now move along an orbit so that the so-called Hamiltonian principle of stationary action is minimised. As we do not derive the Lagrange equations, we will not go into this in detail. The result is that the following equations apply component by component:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} = \frac{\partial L}{\partial q_i}$$

And for all $1 \leq i \leq number\ of\ generalized\ coordinates$.

<u>Variant 3: The thread pendulum and the Lagrange formalism</u>



Thread pendulum with generalised coordinate φ

The Cartesian coordinates then depend as follows on φ as follows:

$$\begin{pmatrix} x(\varphi(t)) \\ y(\varphi(t)) \end{pmatrix} = l \begin{pmatrix} sin\varphi(t) \\ -cos\varphi(t) \end{pmatrix}$$

The potential is the gravitational field near the earth and is given by $V(\varphi) = mgl(1 - cos\varphi)$ if we normalise it with $V(0) = 0$.

Thus the Lagrange function (the first term is the kinetic energy) is as follows:

$$L(\varphi, \dot{\varphi}, t) = \frac{1}{2}m(\dot{x}^2 + \dot{y}^2) - mgl(1 - cos\varphi) = \frac{m}{2}l^2\dot{\varphi}^2 - mgl(1 - cos\varphi)$$

The left-hand side of the Lagrange equation is then:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} = \frac{d}{dt}ml^2\dot{\varphi} = ml^2\ddot{\varphi}$$

The right-hand side of the Lagrange equation is:

$$\frac{\partial L}{\partial \varphi} = -mglsin\varphi$$

If we equate both sides, we get:

$$ml^2\ddot{\varphi} = -mglsin\varphi$$

And thus the familiar equation again:

$$l\ddot{\varphi} + gsin\varphi = 0$$

□

As a further example, let us consider the case of a mass point on which no force acts. Then V is constant and with appropriate normalisation $V \equiv 0$.

Since there are also no constraining forces, Cartesian coordinates are the obvious coordinates to describe the movement of the mass point. The Lagrange function then only consists of the kinetic energy:

$$L = \frac{1}{2}m(\dot{x}^2 + \dot{y}^2 + \dot{z}^2)$$

Then applies:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = \frac{d}{dt}m\dot{x} = m\ddot{x} = \frac{\partial L}{\partial x} = 0$$

It is therefore $m\dot{x} = constant$. But this is nothing other than the momentum component in the x-direction.

The same applies to the coordinates *y,z*.

Conclusion: In force-free space, the momentum of a mass point is constant.

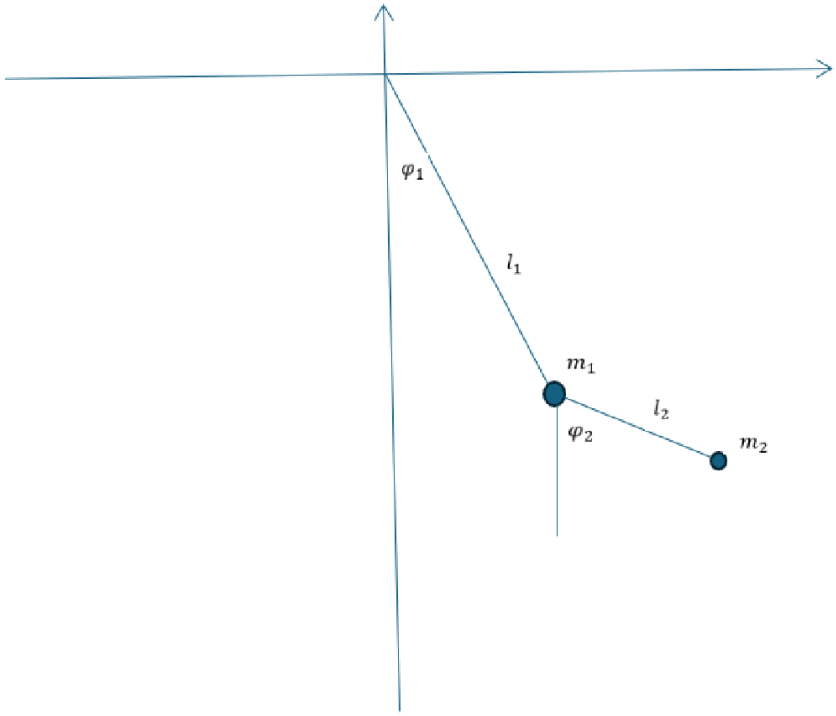
□

This raises the question of whether the Lagrange equation depends on the choice of generalised coordinates. This is not the case and this has to do with the fact that Hamilton's principle of stationary action is independent of this choice. It is similar to determining an extreme value of a function. The question of how to construct a cylindrical box with a given volume from as little sheet

metal as possible also does not depend on the choice of coordinates. However, proving the independence of Hamilton's principle from the choice of coordinates would go too far here. A justification for this can be found in [10].

6.2. The double pendulum

As a first example of a coupled pendulum, we consider the double pendulum in the "Simulator".



Double pendulum

The generalised coordinates are $\begin{pmatrix}\varphi_1(t)\\ \varphi_2(t)\end{pmatrix}$. They completely describe the system at time t. $l_1 > 0, l_2 > 0$ remain constant during the movement.

Now we will not be able to avoid a certain amount of calculation.

The position vector of the mass m_1 is:

$$\vec{r}_1 = \begin{pmatrix}x_1\\ y_1\end{pmatrix} = l_1\begin{pmatrix}sin\varphi_1\\ -cos\varphi_1\end{pmatrix}, \dot{\vec{r}}_1 = l_1\dot{\varphi}_1\begin{pmatrix}cos\varphi_1\\ sin\varphi_1\end{pmatrix}$$

And the position vector of the mass m_2:

$$\vec{r}_2 = \begin{pmatrix}x_2\\ y_2\end{pmatrix} = l_1\begin{pmatrix}sin\varphi_1\\ -cos\varphi_1\end{pmatrix} + l_2\begin{pmatrix}sin\varphi_2\\ -cos\varphi_2\end{pmatrix}$$

$$\dot{\vec{r}}_2 = l_1\dot{\varphi}_1\begin{pmatrix}cos\varphi_1\\ sin\varphi_1\end{pmatrix} + l_2\dot{\varphi}_2\begin{pmatrix}cos\varphi_2\\ sin\varphi_2\end{pmatrix}$$

The kinetic energy is given by

$$E_{kin} = \frac{1}{2}m_1\left|\dot{\vec{r}}_1\right|^2 + \frac{1}{2}m_2\left|\dot{\vec{r}}_2\right|^2$$

$$= \frac{1}{2}m_1 l_1^2\dot{\varphi}_1^2 + \frac{1}{2}m_2\{l_1^2\dot{\varphi}_1^2 + 2l_1 l_2\dot{\varphi}_1\dot{\varphi}_2(cos\varphi_1 cos\varphi_2 + sin\varphi_1 sin\varphi_2) + l_2^2\dot{\varphi}_2^2\}$$

$$= \frac{1}{2} m_1 l_1^2 \dot{\varphi}_1^2 + \frac{1}{2} m_2 \{ l_1^2 \dot{\varphi}_1^2 + 2 l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 \cos(\varphi_1 - \varphi_2) + l_2^2 \dot{\varphi}_2^2 \}$$

And the potential energy:

$$V = m_1 g y_1 + m_2 g y_2 = -m_1 g l_1 cos\varphi_1 - m_2 g l_1 cos\varphi_1 - m_2 g l_2 cos\varphi_2$$

This gives you the Lagrange function:

$$L = E_{kin} - V$$

$$= \frac{1}{2}(m_1 + m_2) l_1^2 \dot{\varphi}_1^2 + m_2 l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 \cos(\varphi_1 - \varphi_2) + \frac{1}{2} m_2 l_2^2 \dot{\varphi}_2^2$$

$$+ g(m_1 + m_2) l_1 cos\varphi_1 + g m_2 l_2 cos\varphi_2$$

The Lagrange equation for the first generalised coordinate is

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}_1} = \frac{\partial L}{\partial \varphi_1}$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}_1} = \frac{d}{dt}\{ l_1^2 (m_1 + m_2) \dot{\varphi}_1 + m_2 l_1 l_2 \cos(\varphi_1 - \varphi_2) \dot{\varphi}_2 \}$$

$$= l_1^2 (m_1 + m_2) \ddot{\varphi}_1 + m_2 l_1 l_2 \cos(\varphi_1 - \varphi_2)\, \ddot{\varphi}_2 - m_2 l_1 l_2 \sin(\varphi_1 - \varphi_2) \dot{\varphi}_2 (\dot{\varphi}_1 - \dot{\varphi}_2)$$

$$\frac{\partial L}{\partial \varphi_1} = -m_2 l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 \sin(\varphi_1 - \varphi_2) - g(m_1 + m_2) l_1 sin\varphi_1$$

We insert these expressions into the Lagrange equation and move everything to the left. Furthermore, it can be reduced by l_1. This gives us

$$l_1 (m_1 + m_2) \ddot{\varphi}_1 + m_2 l_2 \cos(\varphi_1 - \varphi_2)\, \ddot{\varphi}_2$$

$$- m_2 l_2 \sin(\varphi_1 - \varphi_2) \dot{\varphi}_2 (\dot{\varphi}_1 - \dot{\varphi}_2) + m_2 l_2 \dot{\varphi}_1 \dot{\varphi}_2 \sin(\varphi_1 - \varphi_2) + g(m_1 + m_2) sin\varphi_1 = 0$$

If the third addend is multiplied out, a part of the fourth addend is cancelled out and it remains:

$$l_1 (m_1 + m_2) \ddot{\varphi}_1 + m_2 l_2 \{ \cos(\varphi_1 - \varphi_2)\, \ddot{\varphi}_2 + \sin(\varphi_1 - \varphi_2)\, \dot{\varphi}_2^2 \} + g(m_1 + m_2) sin\varphi_1 = 0$$

We set:

$$\Delta\varphi := \varphi_1 - \varphi_2, \mu := \frac{m_2}{m_1 + m_2}$$

When $m_1 > 0$ is $\mu < 1$.

This gives us:

$$(1) \quad l_1 \ddot{\varphi}_1 + \mu l_2 cos\Delta\varphi \ddot{\varphi}_2 + \mu l_2 sin\Delta\varphi \dot{\varphi}_2^2 + g sin\varphi_1 = 0$$

The Lagrange equation for the second generalised coordinate is analogous:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}_2} = \frac{\partial L}{\partial \varphi_2}$$

A corresponding calculation, which we are happy to leave to the reader as an exercise, provides the equation:

$$(2) \quad l_2 \ddot{\varphi}_2 + l_1 cos\Delta\varphi \ddot{\varphi}_1 - l_1 \sin\Delta\varphi\, \dot{\varphi}_1^2 + g sin\varphi_2 = 0$$

Both equations have a high degree of symmetry: If the indices are swapped $1 \leftrightarrow 2$, then one almost merges into the other. They then only differ by the factor μ and $\Delta\varphi$ merges into $-\Delta\varphi$.

If we multiply the second equation (2) by $\mu cos\Delta\varphi$ we get the equations:

$$\begin{cases} l_1\ddot{\varphi}_1 + \mu l_2 cos\Delta\varphi\ddot{\varphi}_2 + \mu l_2 sin\Delta\varphi\dot{\varphi}_2^2 + gsin\varphi_1 = 0 \\ l_1\mu cos^2\Delta\varphi\ddot{\varphi}_1 + \mu l_2 cos\Delta\varphi\ddot{\varphi}_2 - \mu l_1 sin\Delta\varphi cos\Delta\varphi\dot{\varphi}_1^2 + \mu cos\Delta\varphi gsin\varphi_2 = 0 \end{cases}$$

Now we subtract the lower equation from the upper equation and, if we solve for $\ddot{\varphi}_1$:

$$\ddot{\varphi}_1 = \frac{-\mu sin\Delta\varphi(l_2\dot{\varphi}_2^2 + l_1 cos\Delta\varphi\dot{\varphi}_1^2) + g(\mu cos\Delta\varphi sin\varphi_2 - sin\varphi_1)}{l_1(1 - \mu cos^2\Delta\varphi)}$$

If we multiply the first equation (1) by $cos\Delta\varphi$ we get the equations:

$$\begin{cases} l_1 cos\Delta\varphi\ddot{\varphi}_1 + \mu l_2 cos^2\Delta\varphi\ddot{\varphi}_2 + \mu l_2 sin\Delta\varphi cos\Delta\varphi\dot{\varphi}_2^2 + gcos\Delta\varphi sin\varphi_1 = 0 \\ l_1 cos\Delta\varphi\ddot{\varphi}_1 + l_2\ddot{\varphi}_2 - l_1 sin\Delta\varphi\dot{\varphi}_1^2 + gsin\varphi_2 = 0 \end{cases}$$

Again, we subtract the lower equation from the upper one:

$$l_2(\mu cos^2\Delta\varphi - 1)\ddot{\varphi}_2 + \mu l_2 sin\Delta\varphi cos\Delta\varphi\dot{\varphi}_2^2 + gcos\Delta\varphi sin\varphi_1 + l_1 sin\Delta\varphi\dot{\varphi}_1^2 - gsin\varphi_2 = 0$$

Resolved according to $\ddot{\varphi}_2$ this provides:

$$\ddot{\varphi}_2 = \frac{sin\Delta\varphi(l_1\dot{\varphi}_1^2 + \mu l_2 cos\Delta\varphi\dot{\varphi}_2^2) + g(cos\Delta\varphi sin\varphi_1 - sin\varphi_2)}{l_2(1 - \mu cos^2\Delta\varphi)}$$

When $m_1 > 0$ is $\mu < 1$ and therefore the denominator is greater than 0 in both cases.

The oscillation frequency of an isolated pendulum would be $\omega_i = \sqrt{\frac{g}{l_i}}$, $i \in \{1,2\}$.

To calculate the equations for $\ddot{\varphi}_1$ and $\ddot{\varphi}_2$ into a first-order system of equations, we set:

$$u_1 = \varphi_1, v_1 = \dot{\varphi}_1, u_2 = \varphi_2, v_2 = \dot{\varphi}_2$$

$$\begin{cases} \dot{u}_1 = v_1 =: f_1(t, u_1, v_1, u_2, v_2) \\ \dot{v}_1 = \dfrac{-\mu sin\Delta u(l_2 v_2^2 + l_1 cos\Delta u v_1^2) + g(\mu cos\Delta u sinu_2 - sinu_1)}{l_1(1 - \mu cos^2\Delta u)} =: g_1(t, u_1, v_1, u_2, v_2) \\ \dot{u}_2 = v_2 =: f_2(t, u_1, v_1, u_2, v_2) \\ \dot{v}_2 = \dfrac{sin\Delta u(l_1 v_1^2 + \mu l_2 cos\Delta u v_2^2) + g(cos\Delta u sinu_1 - sinu_2)}{l_2(1 - \mu cos^2\Delta u)} =: g_2(t, u_1, v_1, u_2, v_2) \end{cases}$$

This is $\Delta u = u_1 - u_2$.

The four-step Runge-Kutta method with a step size *h* somewhat complex for the n+1-th step because we have to take four parameters u_1, v_1, u_2, v_2 into account. To document the implementation in the "Simulator", we prepare it in the following algorithm. This contains $u_{1n}, v_{1n}, u_{2n}, v_{2n}$ the values of the parameters after the nth iteration step. The (constant) step size is d, which must be sufficiently small. Now we try to keep the representation somewhat compact. We carry out the following steps one after the other:

$$\vec{x}_{1n} \coloneqq (u_{1n}, v_{1n}, u_{2n}, v_{2n})$$

$$\begin{cases} k_{i1} \coloneqq f_i(t_n, \vec{x}_{1n}) \\ h_{i1} \coloneqq g_i(t_n, \vec{x}_{1n}) \end{cases}, i \in \{1,2\}$$

$$\vec{x}_{2n} := (u_{1n} + \frac{d}{2}k_{11}, v_{1n} + \frac{d}{2}h_{11}, u_{2n} + \frac{d}{2}k_{21}, v_{2n} + \frac{d}{2}h_{21})$$

$$\begin{cases} k_{i2} := f_i(t_n + \frac{d}{2}, \vec{x}_{2n}) \\ h_{i2} := g_i(t_n + \frac{d}{2}, \vec{x}_{2n}) \end{cases}, i \in \{1,2\}$$

$$\vec{x}_{3n} := (u_{1n} + \frac{d}{2}k_{12}, v_{1n} + \frac{d}{2}h_{12}, u_{2n} + \frac{d}{2}k_{22}, v_{2n} + \frac{d}{2}h_{22})$$

$$\begin{cases} k_{i3} := f_i(t_n + \frac{d}{2}, \vec{x}_{3n}) \\ h_{i3} := g_i(t_n + \frac{d}{2}, \vec{x}_{3n}) \end{cases}, i \in \{1,2\}$$

$$\vec{x}_{4n} := (u_{1n} + k_{13}, v_{1n} + h_{13}, u_{2n} + k_{23}, v_{2n} + h_{23})$$

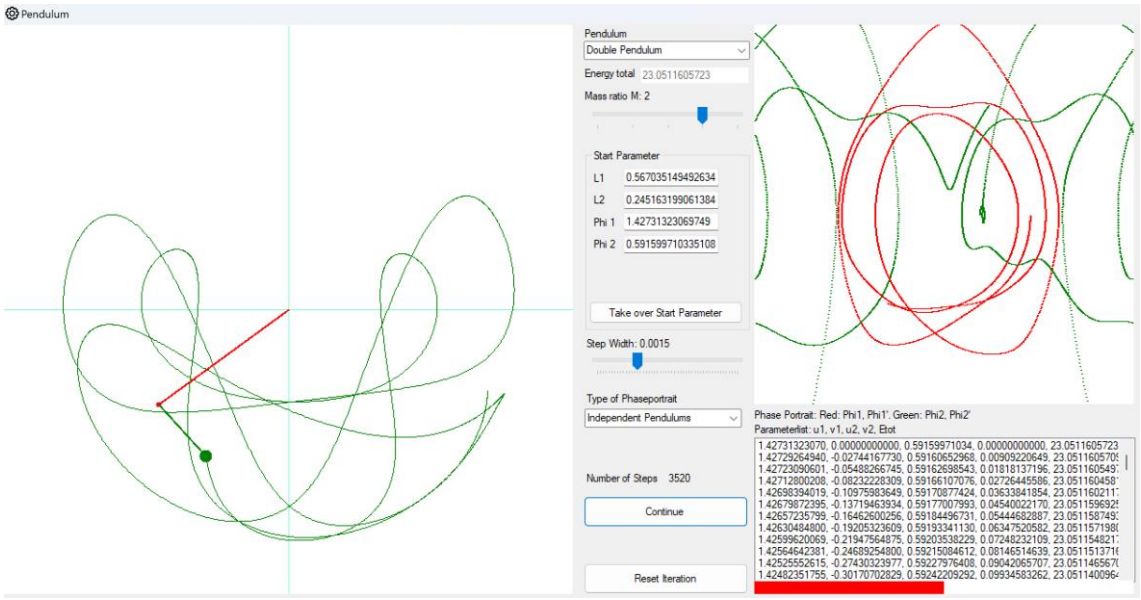
$$\begin{cases} k_{i4} := f_i(t_n + d, \vec{x}_{4n}) \\ h_{i4} := g_i(t_n + d, \vec{x}_{4n}) \end{cases}, i \in \{1,2\}$$

$$\begin{cases} t_{n+1} = t_n + d \\ u_{i(n+1)} = u_{in} + \frac{d(k_{i1} + 2k_{i2} + 2k_{i3} + k_{i4})}{6} \\ v_{i(n+1)} = v_{in} + \frac{d(h_{i1} + 2h_{i2} + 2h_{i3} + h_{i4})}{6} \end{cases}, i \in \{1,2\}$$

When experimenting, we will see that the Runge Kutta method is only of limited use for the double pendulum, namely only for low energy values or pendulum deflections.

6.3. Implementation of the double pendulum

Like similar forms, *FrmPendulum* offers the display of current pendulum positions in *PicPendulum* and the recording of the pendulum's path in the *MapPendulum* bitmap. On the right-hand side, the movement of the pendulums is recorded as a phase diagram in *PicPhasePortrait* and the individual parameter values are logged in *LstParameterList*.



FrmPendulum

Up to 6 parameters can then be specified, which are constant or variable depending on the pendulum (this determines the implementation of the pendulum). These parameters are set automatically when the start position of the pendulum is set manually with the mouse. However, they can also be entered manually and transferred to the pendulums. The factor C is defined for each pendulum, as is an optional additional parameter (*AdditionalParameter*), which is set using a shift register.

The communication between *FrmPendulum* and the implementation of the individual pendulums is defined by the *IPendulum* interface.

For a better understanding of the code, here are some explanations:

- The freely selectable parameters are labelled *TxtP1...TxtP6* in the *FrmPendulum.*
- These are transferred to the pendulum class (e.g. *ClsDoublePendulum*) via an interface as vectors *MyConstants* and *MyVariables*
- One or more *SetandDrawStartPosition* routines support the setting of the start positions with the mouse. You then also set *MyConstants* and *MyVariables*, whose values are entered in the *FrmPendulum* with *TxtP1...TxtP6.*
- The positions of the pendulums are held in *position* as *ClsMathPoint* and are also set as the start value by *SetandDrawStartPosition.*
- The position of the pendulum is continuously drawn in *DrawPendulum* based on the *position.*

The following remarks on iteration:

- The relevant iteration parameters are based on the formulae of the Runge Kutta method, for example for the double pendulum *u1, v1, u2, v2*. Their start value is also set by *SetandDrawStartposition.*
- At the start of an iteration step, *OldPosition = Position* is set. *OldPosition* is therefore the old position of the pendulum before the iteration step.
- For the iteration, a *ClsVector x(3) is* used for the double pendulum based on the mathematical formulae. It plays the role of the individual $\vec{x}_{in}$.
- k11, k12, k13, k14 is managed as *ClsVector(3).* Likewise *k2i, h1i* and *h2i*.
- At the end of each iteration step, the current position of the pendulum is drawn based on *Position* and the pendulum track is drawn in *DrawTrack* based on *OldPosition* and *Position*.

Testing and monitoring

If the "Test mode" checkbox is activated, the functions in the Runge Kutta method are replaced so that both pendulums swing independently of each other like normal string pendulums (but with possibly large deflection). For both pendulums, the differential equation

$$\ddot{\varphi}_i = \frac{-gsin\varphi_i}{l_i}\ , i \in \{1,2\}$$

is used.

It is obvious that the Runge Kutta method is unreliable in the case of the double pendulum. For small oscillations, it seems to fulfil the expectations of a double pendulum. For large oscillations, some movements appear unnatural. It should also be noted that the simulation does not reflect the actual time course, but depends on the speed of each calculation step in the Runge Kutta method.

What can be demanded as a minimum, however, is that the total energy of the double pendulum remains constant during the movement, at least to some extent. This does not guarantee the reliability of the method. But conversely, it is obvious when the process is "out of control".

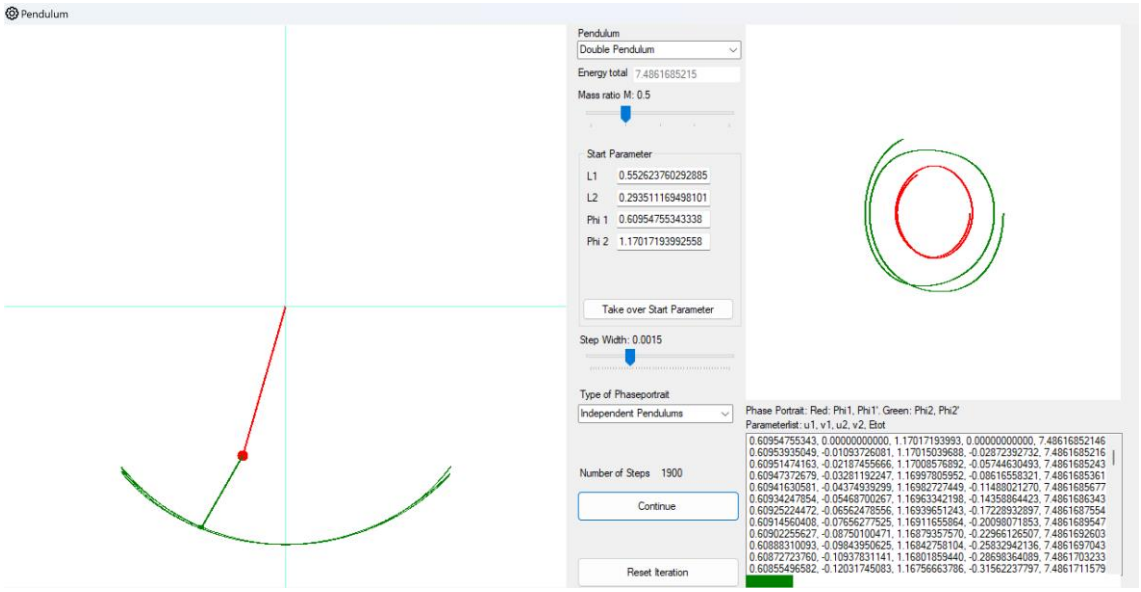
The following applies to the kinetic energy of the double pendulum:

$$E_{kin} = \frac{1}{2} m_1 l_1^2 \dot{\varphi}_1^2 + \frac{1}{2} m_2 \{ l_1^2 \dot{\varphi}_1^2 + 2 l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 \cos(\varphi_1 - \varphi_2) + l_2^2 \dot{\varphi}_2^2 \}$$

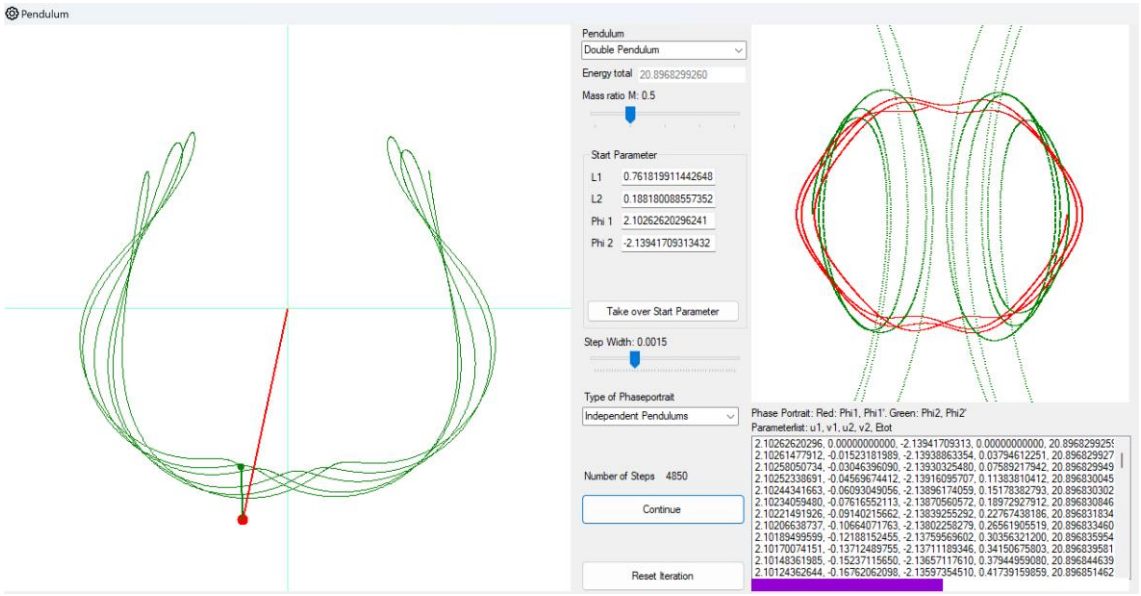
For the potential energy, we set the zero level to $y = -1$. This becomes:

$$E_{pot} = g\{(1 - l_1 cos\varphi_1)(m_1 + m_2) - m_2 l_2 cos\varphi_2\}$$

During the iteration, the current energy of the double pendulum is continuously calculated and compared with the starting energy. The result is shown in the image above by a horizontal bar at the bottom right. The total interval corresponds to the range of possible starting energy and the coloured bar to the current energy of the double pendulum. If this bar is green, this means that the deviation of the pendulum energy is less than 10% relative to the total interval. If the pendulum energy is higher, the bar turns red. If it is lower, the bar turns purple.



Test mode and monitoring of the total energy on the right at low energy:
The total energy remains in the green range



Test mode and monitoring of the total energy on the right at high energy:
The total energy is sometimes too high (red) or too low (purple)

During implementation, we rely on $m_1 = 1$ and $m_2 = M$. We also use:

$$\varphi_i = u_i, \dot{\varphi}_i = v_i, i \in \{1, 2\}$$

This provides us with energy:

$$E_{tot} = \frac{1}{2} l_1^2 \dot{\varphi}_1^2 + \frac{1}{2} M\{l_1^2 \dot{\varphi}_1^2 + 2 l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 \cos(\varphi_1 - \varphi_2) + l_2^2 \dot{\varphi}_2^2\} + g\{(1 - l_1 cos\varphi_1)(1 + M) - M l_2 cos\varphi_2\}$$

The double pendulum is started from the rest position. There is $E_{kin} = 0$. The potential energy is minimum for $\varphi_1 = \varphi_0 = 0$ and maximum for $\varphi_1 = \varphi_0 = \pi$. We denote these values by:

$$E_{min} = g\{(1 - l_1)(1 + M) - M l_2\}$$

$$E_{max} = g\{(1 + l_1)(1 + M) + M l_2\}$$

The pendulum energy is therefore in the interval $[E_{min}, E_{max}]$. The current pendulum energy exceeds the limit of 10% if:

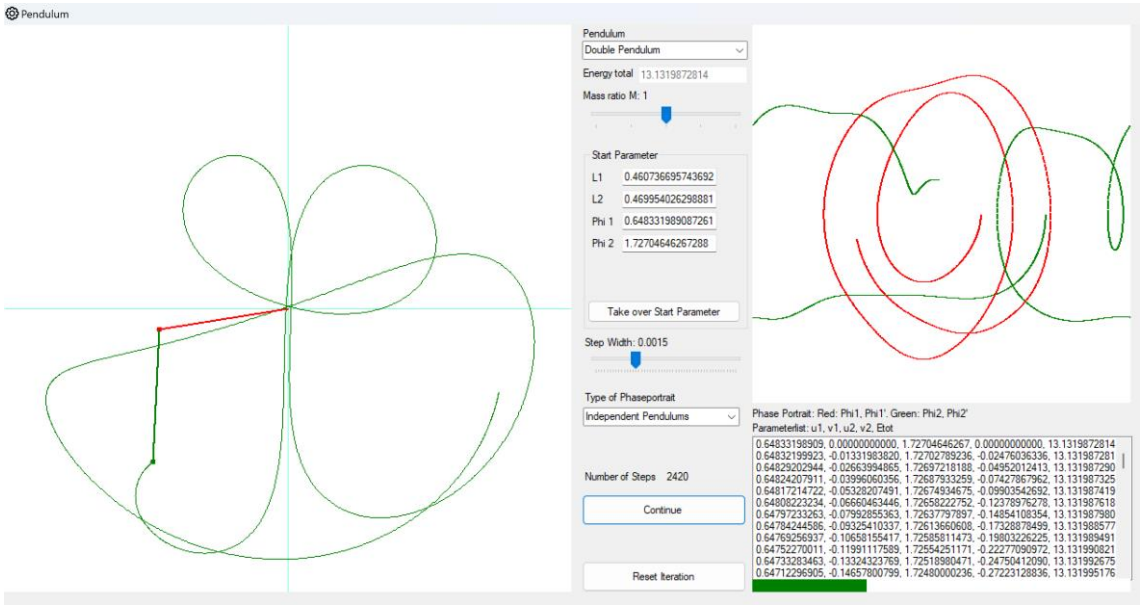
$$E_{aktuell} \geq E_{Start} + 0.1(E_{max} - E_{min})$$

And falls short of it if:

$$E_{aktuell} \leq E_{Start} - 0.1(E_{max} - E_{min})$$

6.4. Investigation of the phase space for the double pendulum

In the previous chapter, we visualised the movement of the pendulums individually in phase space. In other words, starting from a starting point, we have the curves $(\varphi_1, \dot{\varphi}_1)$ and $(\varphi_2, \dot{\varphi}_2)$ individually. This results in a picture of the movement with a constant energy starting from the starting point. If you now want to gain an overview of the behaviour of the double pendulum across all possible energy states, this representation is unsuitable.



Visualisation of the individual pendulums in phase space

A better way of investigating the behaviour of a dynamic system in phase space is the so-called Poincaré section. In the double pendulum, the phase space is four-dimensional and the coordinates of a point are denoted by $(\varphi_1, \dot{\varphi}_1, \varphi_2, \dot{\varphi}_2)$ denoted. During the movement of the double pendulum, this point moves in phase space along a trajectory, i.e. a curve in phase space.

If you now place a hyperplane (this is an (n-1) dimensional subspace of the phase space) in the phase space, you can analyse the points at which the trajectory of the path intersects this hyperplane.

In our case, we want to define the hyperplane by the condition $\varphi_1 = 0$. We therefore consider the points in phase space at the time when the upper pendulum passes straight through the y-axis. This gives us a *discrete* dynamic system for the double pendulum. A point of intersection with the hyperplane is mapped to the point at which the trajectory intersects the hyperplane on the next pass:

$$(0, \dot{\varphi}_1, \varphi_2, \dot{\varphi}_2)_n \mapsto (0, \dot{\varphi}_1, \varphi_2, \dot{\varphi}_2)_{n+1}$$

The total energy along a trajectory is constant. For $\varphi_1 = 0$ it is given by the previous terms and $m_1 = 1,\ m_2 = M$ is given by:

$$E_{tot} = \frac{1+M}{2} l_1^2 \dot{\varphi}_1^2 + M l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 cos\varphi_2 + \frac{M}{2} l_2^2 \dot{\varphi}_2^2 + g((1-l_1)(1+M) - M l_2 cos\varphi_2)$$

$$= \frac{1+M}{2} l_1^2 \dot{\varphi}_1^2 + M l_1 l_2 \dot{\varphi}_1 \dot{\varphi}_2 cos\varphi_2 + \frac{M}{2} l_2^2 \dot{\varphi}_2^2 + E_{pot}(0, \varphi_2)$$

In the implementation, we will change the position $\varphi_1 = 0$ by changing the sign of the corresponding parameter u_1 in the Runge Kutta method.

Now we project the hyperplane $(\dot{\varphi}_1, \varphi_2, \dot{\varphi}_2)$ onto $(\varphi_2, \dot{\varphi}_2)$which means that we only enter the coordinates of the lower pendulum into the phase space. If $(\varphi_2, \dot{\varphi}_2)$ are given, then $\dot{\varphi}_1$ is also determined by the above energy equation. This means that $\dot{\varphi}_1$ is not an independent parameter, as the energy is constant. If we transform the energy equation slightly, we obtain an equation for $\dot{\varphi}_1$:

$$\frac{1+M}{2} l_1^2 \dot{\varphi}_1^2 + M l_1 l_2 \dot{\varphi}_2 cos\varphi_2 \dot{\varphi}_1 + \frac{M}{2} l_2^2 \dot{\varphi}_2^2 + E_{pot}(0, \varphi_2) - E_{tot} = 0$$

We divide by the first coefficient of $\dot{\varphi}_1$:

$$\dot{\varphi}_1^2 + 2 \cdot \frac{M}{1+M} \cdot \frac{l_2}{l_1} \dot{\varphi}_2 cos\varphi_2 \cdot \dot{\varphi}_1 + C = 0$$

Whereby

$$C = \frac{1}{(1+M)l_1^2}\left(Ml_2^2\dot{\varphi}_2^2 + 2E_{pot}(0,\varphi_2) - 2E_{tot}\right)$$

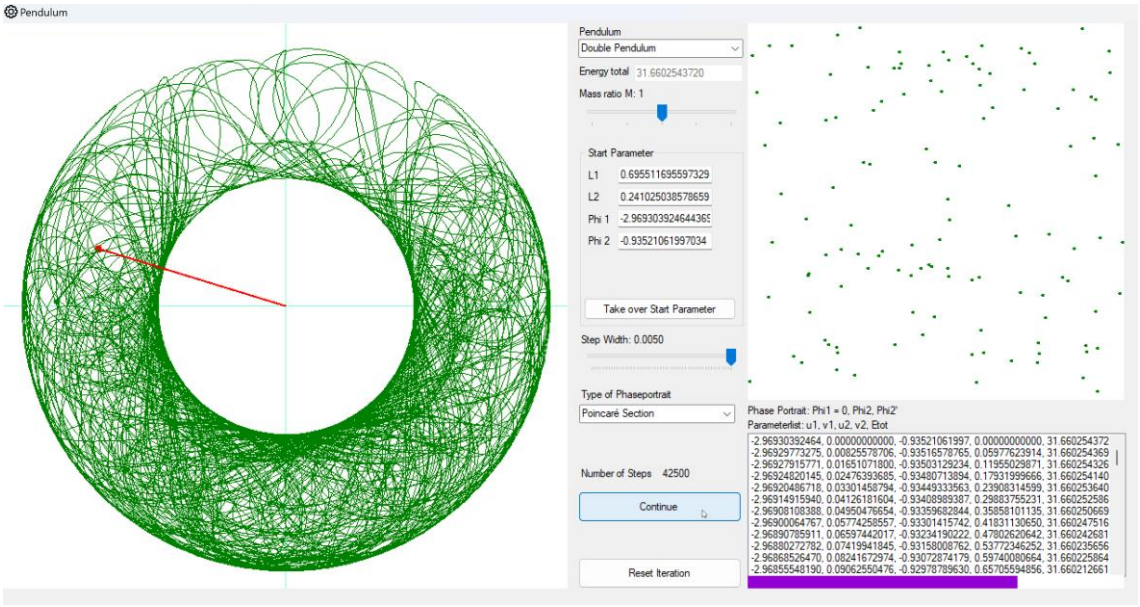
This provides for $\dot{\varphi}_1$:

$$\dot{\varphi}_1 = -\frac{Ml_2}{(1+M)l_1}\dot{\varphi}_2 cos\varphi_2 \pm \sqrt{(\frac{Ml_2}{(1+M)l_1}\dot{\varphi}_2 cos\varphi_2)^2 - C}$$

In the implementation, we will additionally require as a condition for the plotting of $(\varphi_2, \dot{\varphi}_2)$ that applies:

$$\dot{\varphi}_1 + \frac{M}{1+M}\frac{l_2}{l_1}\dot{\varphi}_2 cos\varphi_2 \geq 0$$

So that the points $(0, \dot{\varphi}_1, \varphi_2, \dot{\varphi}_2)$ are clearly defined.

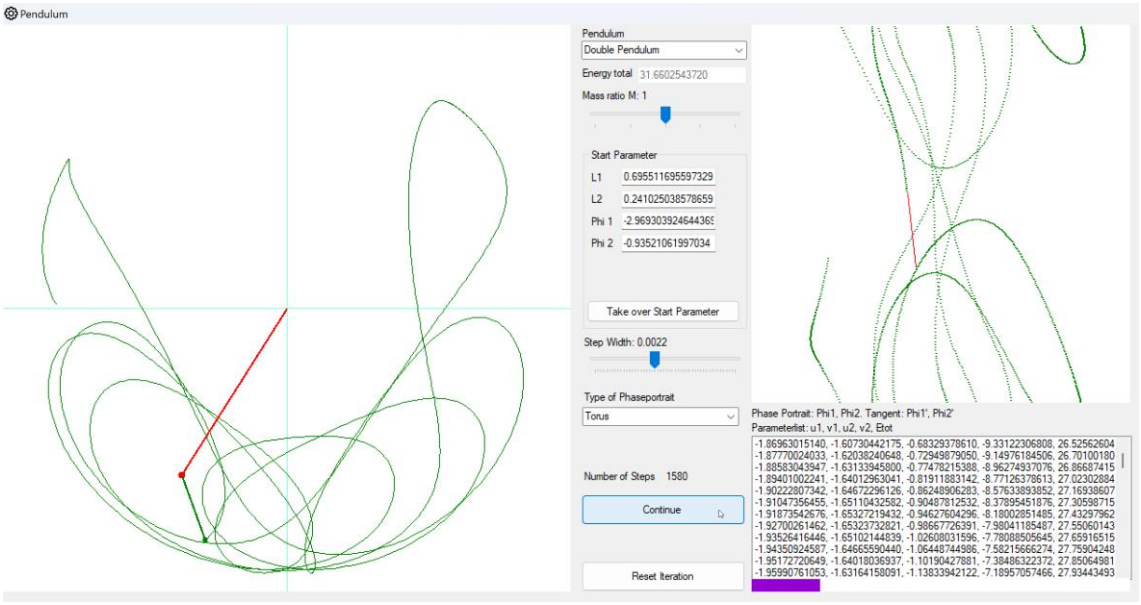


Poincaré section: It should be noted that the simulation does not represent a real double pendulum

In the Runge Kutta parameters, the condition holds:

$$v_1 + \frac{M}{1+M}\frac{l_2}{l_1}v_2 cosu_2 \geq 0$$

Another representation of the phase space is obtained using the torus. The parameters (φ_1, φ_2) are 2π-periodic and lie on a torus. Their derivative at the point (φ_1, φ_2) is the tangent to the corresponding curve and lies in the tangential plane at this point. In the representation on the torus, the orbit of the parameter pair (φ_1, φ_2) is shown in green and the respective tangent $(\dot{\varphi}_1, \dot{\varphi}_2)$ by a red line starting from the point (φ_1, φ_2).

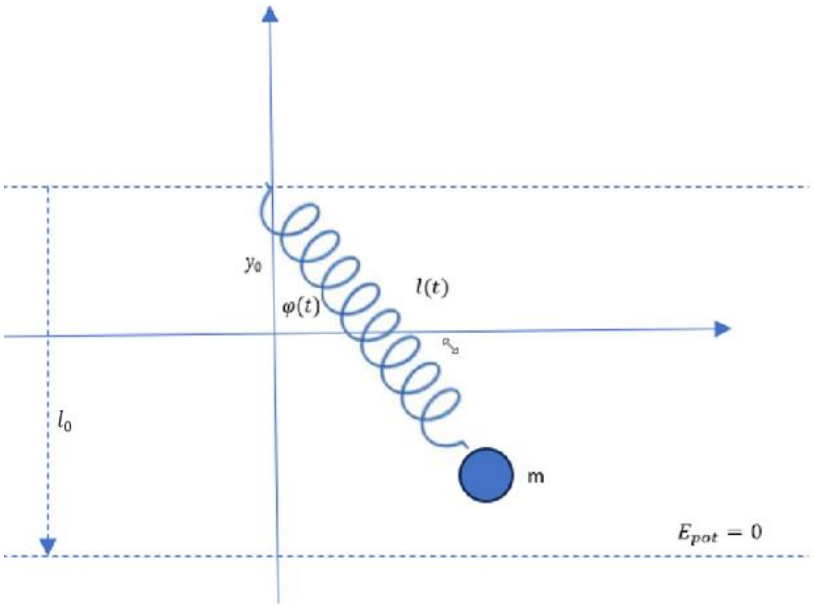


Visualisation of the movement on the torus

In the *FrmPendulum* you can then choose which representation in phase space (both pendulums independently of each other, torus or Poincaré section) you want to see.

6.5. Oscillating spring pendulum

Here we are looking at a spring pendulum, which can also swing back and forth like a thread pendulum.



Oscillating spring pendulum

At the point $(0, y_0)$ of the coordinate system, a spring is attached with a mass *m* hanging from its other end. This mass also swings back and forth. The deflection angle $\varphi(t)$ as well as the spring length $l(t) > 0$ depends on the time *t*.

Let it be l_0 is the length of the relaxed spring. Let the zero level of potential energy in relation to gravity be at the height $y = y_0 - l_0$ or at $\varphi = 0, l = l_0$. Let *D* denote the spring constant.

The position vector of the mass *m* is

$$\vec{r}(t) = l(t)\begin{pmatrix} sin\varphi(t) \\ -cos\varphi(t) \end{pmatrix} + \begin{pmatrix} 0 \\ y_0 \end{pmatrix}$$

$$\dot{\vec{r}} = \dot{l}\begin{pmatrix} sin\varphi \\ -cos\varphi \end{pmatrix} + l\dot{\varphi}\begin{pmatrix} cos\varphi \\ sin\varphi \end{pmatrix}$$

$$\left|\dot{\vec{r}}\right|^2 = \dot{l}^2 + 2l\dot{l}\dot{\varphi}[sin\varphi cos\varphi - cos\varphi sin\varphi] + l^2\dot{\varphi}^2$$

$$E_{kin} = \frac{1}{2}m\left(\dot{l}^2 + l^2\dot{\varphi}^2\right)$$

Then the following applies to the entire potential (spring energy plus gravitational energy):

$$V = \frac{1}{2}D(l - l_0)^2 + mg(l_0 - lcos\varphi)$$

This gives us the Lagrange function:

$$L = \frac{m}{2}\left(\dot{l}^2 + l^2\dot{\varphi}^2\right) - \frac{D}{2}(l - l_0)^2 - mg(l_0 - lcos\varphi)$$

For the first coordinate $l(t)$ applies:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{l}} = \frac{d}{dt}\left(m\dot{l}\right) = m\ddot{l}$$

$$\frac{\partial L}{\partial l} = ml\dot{\varphi}^2 - D(l - l_0) + mgcos\varphi$$

This is provided by the Lagrange equation:

$$\ddot{l} = l\dot{\varphi}^2 - \frac{D}{m}(l - l_0) + gcos\varphi$$

For the second coordinate $\varphi(t)$ you have:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} = \frac{d}{dt}(ml^2\dot{\varphi}) = 2ml\dot{l}\dot{\varphi} + ml^2\ddot{\varphi}$$

$$\frac{\partial L}{\partial \varphi} = -mglsin\varphi$$

This is provided by the Lagrange equation:

$$ml^2\ddot{\varphi} = -2ml\dot{l}\dot{\varphi} - mglsin\varphi$$

$$\ddot{\varphi} = -(2\dot{\varphi}\dot{l} + gsin\varphi)/l$$

Only the ratio $\frac{D}{m}$ is relevant for the movement. However, this is the square of the oscillation frequency of the spring pendulum. We therefore set $\omega = \sqrt{\frac{D}{m}}$ and obtain the equations:

$$\begin{cases} \ddot{l} = l\dot{\varphi}^2 - \omega^2(l - l_0) + gcos\varphi \\ \ddot{\varphi} = -\dfrac{2\dot{\varphi}\dot{l} + gsin\varphi}{l} \end{cases}$$

In order to convert the equations into a first-order differential equation system, we introduce the following variables:

$$u_1 = l, v_1 = \dot{l}, u_2 = \varphi, v_2 = \dot{\varphi}$$

This gives us the system of equations for the Runge Kutta method:

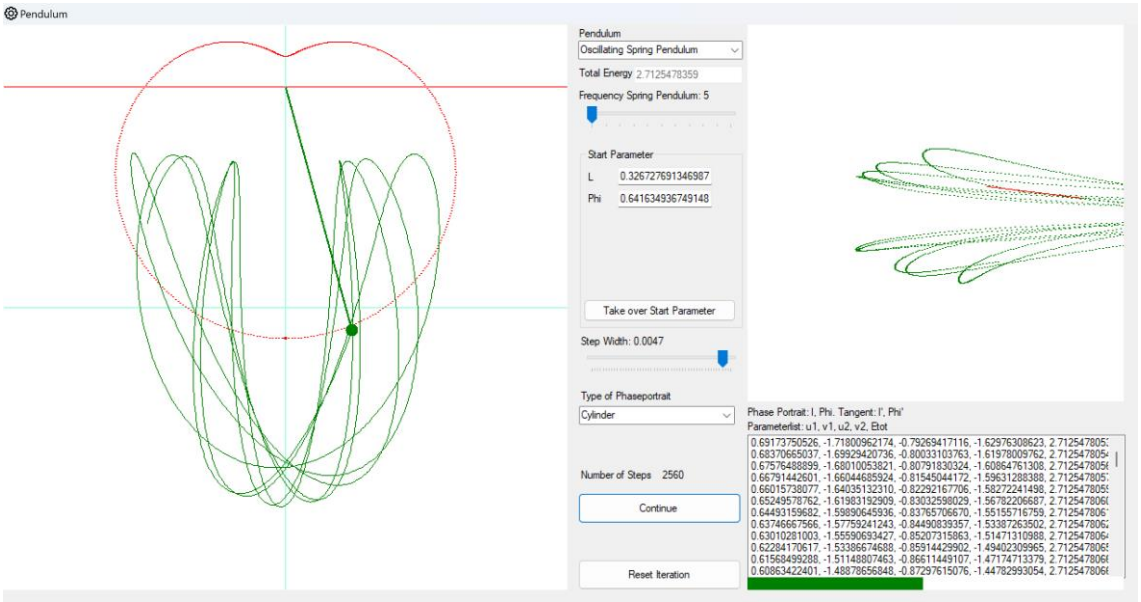
$$\begin{cases} \dot{u}_1 = v_1 \\ \dot{v}_1 = u_1 v_2^2 - \omega^2(u_1 - l_0) + gcosu_2 \\ \dot{u}_2 = v_2 \\ \dot{v}_2 = -\dfrac{2v_2 v_1 + gsinu_2}{u_1} \end{cases}$$

The same formalism is used for the implementation as for the double pendulum.

Since $u_1$the length of the spring pendulum, is in the denominator of the last equation, then $\dot{v}_2$ can become very large if the pendulum swings in the direction of the point $(0, y_0)$ and comes very close to it. In the case of a physical pendulum, the spring, whose length has a minimum dimension, blocks this effect. In the implementation, we therefore assume that the length of the spring pendulum should not fall below a value l_{min} should not fall below this value. This value is defined during implementation.

If now during the iteration $u_1 < l_{min}$ this is picked up and replaced by the Runge Kutta formulae $u_1 = l_{min}, v_1 = -v_1$ formulae. The total energy changes insignificantly at most. Visually, this looks as if the mass *m* on the sphere with radius l_{min} and centre $(0, y_0)$ without friction.

The operation of the *FrmPendulum* is the same again.



Oscillating spring pendulum

For any given angle φ there is a starting point of the pendulum at which the spring force just compensates for the gravitational force. In an exercise, the reader can show that this is the case if $l(\varphi) = l_0 + \frac{gcos\varphi}{\omega^2}$. The corresponding locus line is a cardioid and is shown in red in the picture above.

There are again three options for visualisation in phase space:

- Movement of the spring pendulum and the plane oscillation independent of each other
- Representation of the parameter pair (l, φ) on a cylinder and the tangent $(\dot{l}, \dot{\varphi})$ as a red line at the respective current curve point
- Poincaré section: Here the hyperplane is defined by the condition $\varphi = 0$ condition, i.e. at the point where the pendulum passes the negative y-axis.

The Runge Kutta method seems more reliable here. At the very least, the total energy of the system is retained.

6.6. Implementation of the oscillating spring pendulum

The implementation follows the same logic as the implementation of the double pendulum. The corresponding Runge Kutta formulae are implemented. In addition, the starting conditions for the pendulum should be limited as much as possible so that the movement is visible within the *PicDiagram.*

As a test case for the implementation, let's consider the equilibrium state when $\varphi = 0$ is. The length of the spring in this position is l_1.

In this case the following applies:

$$D(l_1 - l_0) = mg$$

For the implementation we set $m = 1$then $D = \omega^2$ and thus:

$$l_1 = l_0 + \frac{g}{\omega^2}$$

Whereby in the implementation $g = 9.81$ is set.

The total energy displayed in the second highest field is then

$$E_{0,tot} = E_{kin} + E_{Feder} + E_{pot} = 0 + \frac{D}{2}(l_1 - l_0)^2 + mg(l_0 - l_1) = -\frac{g^2}{2\omega^2}$$

If you observe the equilibrium condition and the values for the implementation. You can now calculate the corresponding values manually, set the start values accordingly and have them accepted. You can then check the total energy displayed and the pendulum should not move.

The upper equilibrium point can be considered in the same way: $\varphi = \pi$ and the length l_2 of the spring is just long enough to compensate for the force of gravity. An analogous calculation is then performed:

$$l_2 = l_0 - \frac{g}{\omega^2}$$

And the energy in this point is:

$$E_{1,tot} = \frac{g^2}{2\omega^2}$$

If we analyse the total energy of the system, where the zero level of the gravitational potential energy is again at the height $y = y_0 - l_0$ then the following applies to the energy (as before $m = 1, D = \omega^2$):

$$E_{tot} = E_{kin} + E_{Feder} + E_{pot} = \frac{1}{2}m\left(\dot{l}^2 + l^2\dot{\varphi}^2\right) + \frac{D}{2}(l - l_0)^2 + mg(l_0 - lcos\varphi) \geq$$

$$\frac{\omega^2}{2}(l - l_0)^2 + g(l_0 - l) =: \tilde{E}(l)$$

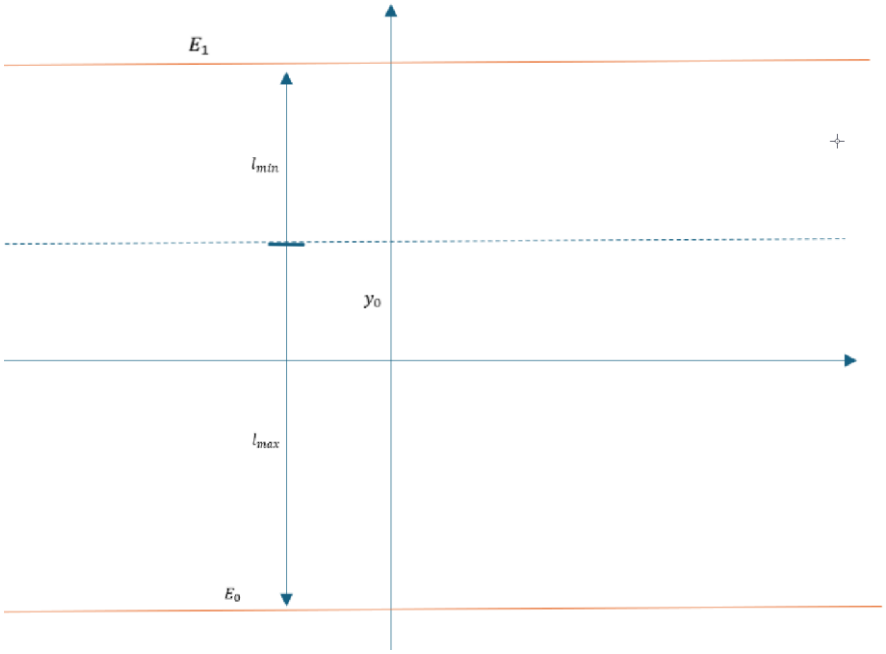
$\tilde{E}(l)$ is an open parabola and we are looking for its minimum:

$$\frac{d\tilde{E}}{dl} = \omega^2(l - l_0) - g = 0$$

The minimum is assumed to be in the lower equilibrium state: $l_{min} = l_0 + \frac{g}{\omega^2}$ and thus applies:

$$E_{tot} \geq -\frac{g^2}{2\omega^2}, \forall l, \varphi$$

During implementation, we need to clarify how the parameters l_0, y_0, ω should be selected so that the pendulum has room to swing in the square $[-1,1]x[-1,1]$.



Analysing the maximum pendulum movement

First, we take care of an optimal value of y_0.

Let's assume that the pendulum starts from the bottom position and has a maximum length of l_{max} and let it be $\varphi = 0$. The pendulum energy at this point is E_0. At the maximum height reached by the pendulum, the pendulum length is minimal l_{min}. The energy at this height is E_1. A total length of 2 is available in the vertical direction, i.e. $l_{max} + l_{min} = 2$.

Due to the conservation of energy, the following applies $E_0 = E_1$. Because we set m=1 in the implementation and the upper point is $\varphi = \pi$ this results in:

$$\frac{\omega^2}{2}(l_{max} - l_0)^2 + g(l_0 - l_{max}) = \frac{\omega^2}{2}(l_{min} - l_0)^2 + g(l_0 + l_{min})$$

After a short calculation we get the following:

$$\frac{\omega^2}{2}\left(l_{max}^2 - l_{min}^2 - 2L(l_{max} - l_{min})\right) - g(l_{max} + l_{min}) = 0$$

Now we use: $l_{max} + l_{min} = 2$ and get:

$$\frac{\omega^2}{2}\big(2(l_{max} - l_{min}) - 2L(l_{max} - l_{min})\big) = 2g$$

$$l_{max} - l_{min} = \frac{2g}{\omega^2(1-L)}$$

The denominator is defined as $L < 1$. Now is

$$2y_0 = l_{max} - l_{min}$$

This means that the suspension point of the pendulum is sufficiently well positioned when it is set:

$$y_0 = \frac{g}{\omega^2(1-l_0)}$$

Since holds: $y_0 \leq 1$ a condition for ω can be derived from this:

$$\frac{g}{\omega^2(1-l_0)} \leq 1 \implies \sqrt{\frac{g}{1-l_0}} \leq \omega$$

All these conditions are taken into account during implementation.

In addition, the upper equilibrium point should be "possible". As we have seen above, the following applies to this point:

$$l_2 = l_0 - \frac{g}{\omega^2}$$

Since $l_2 > 0$ should be, follows $l_0 - \frac{g}{\omega^2} > 0$ and therefore $\omega > \sqrt{\frac{g}{l_0}}$. To summarise:

$$\omega > \max\left(\sqrt{\frac{g}{l_0}}, \sqrt{\frac{g}{1-l_0}}\right)$$

For reasons of symmetry, this suggests that the implementation $l_0 = 0.5$ in the implementation.

So that we can realistically display the total energy in the status bar at the bottom right, we still need the bandwidth in which the energy can lie at the start. To monitor the current energy in the course of the pendulum movement, this is then displayed in an interval $[E_{min}, E_{max}]$ in the status bar at the bottom right of the window. We have already established that the following applies:

$$E_{tot} \geq -\frac{g^2}{2\omega^2} =: E_{min}$$

The total energy at the start is:

$$E_{tot} = \frac{\omega^2}{2}(l - l_0)^2 + g(l_0 - lcos\varphi)$$

In other words, a parabola that is open at the top. This assumes the maximum value at the edge of the definition interval. We have defined the pendulum length during implementation:

$$l \in [l_{min}, 1 + y_0 cos\varphi]$$

The maximum value of *l* is therefore dependent on φ and is the same: $l(\varphi) = 1 + y_0 cos\varphi$. Thus

$$E_{tot}(\varphi) = \frac{\omega^2}{2}(1 + y_0 cos\varphi - l_0)^2 + g(l_0 - (1 + y_0 cos\varphi)cos\varphi)$$

We are now looking for the extremum of this function. As you can see from the calculation $\frac{dE_{tot}}{d\varphi} = 0$ for $\varphi = 0, \pi$.

If you calculate the difference between the energy values at these points, you get

$$E_{tot}(\pi) - E_{tot}(0) =$$

$$\frac{\omega^2}{2}(1 - y_0 - l_0)^2 + g(l_0 + 1 - y_0) - \frac{\omega^2}{2}(1 + y_0 - l_0)^2 + g(l_0 - 1 - y_0) =$$

$$2y_0\omega^2(l_0 - 1) + 2g \geq 0$$

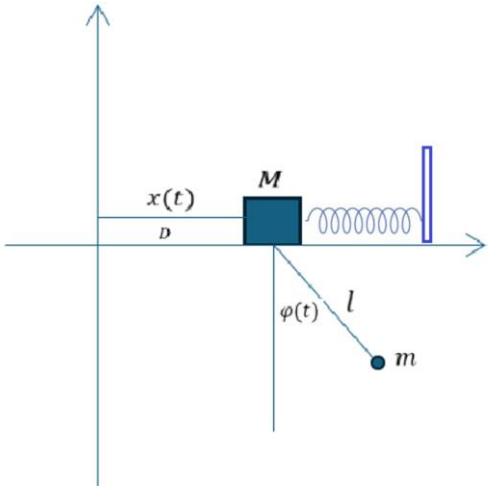
Because we have set:

$$y_0 = \frac{g}{\omega^2(1 - l_0)}$$

Therefore $E_{max} = E_{tot}(\pi)$.

6.7. Shaked pendulum

In this section we consider a shaker pendulum in which a mass *M* swings back and forth horizontally without friction like a spring pendulum and to which a thread pendulum is attached.



Vibrating pendulum

The position vector of the mass *m* is

$$\vec{r}(t) = \begin{pmatrix} x(t) + lsin\varphi(t) \\ -lcos\varphi(t) \end{pmatrix}$$

This gives us:

$$\dot{\vec{r}} = \begin{pmatrix} \dot{x} + l\dot{\varphi}cos\varphi \\ l\dot{\varphi}sin\varphi \end{pmatrix}$$

$$\left|\dot{\vec{r}}\right|^2 = \dot{x}^2 + 2\dot{x}\dot{\varphi}lcos\varphi + \dot{\varphi}^2l^2$$

$$E_{kin} = \frac{1}{2}m[\dot{x}^2 + 2\dot{x}\dot{\varphi}lcos\varphi + \dot{\varphi}^2l^2] + \frac{1}{2}M\dot{x}^2$$

Let *D* be the spring constant. Let the spring be relaxed at the point $x = 0$. Let the zero level of the potential energy of the mass *m be at* the level $y = 0$. Then is:

$$V = \frac{1}{2}Dx^2 - mglcos\varphi$$

And the Lagrange function:

$$L = \frac{1}{2}m[\dot{x}^2 + 2\dot{x}\dot{\varphi}lcos\varphi + \dot{\varphi}^2l^2] + \frac{1}{2}M\dot{x}^2 - \frac{1}{2}Dx^2 + mglcos\varphi$$

For the coordinate φ we receive:

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{\varphi}} = \frac{d}{dt}(m\dot{x}lcos\varphi + m\dot{\varphi}l^2) = ml(\ddot{x}cos\varphi - \dot{x}\dot{\varphi}sin\varphi + \ddot{\varphi}l)$$

$$\frac{\partial L}{\partial \varphi} = -m\dot{x}\dot{\varphi}lsin\varphi - mglsin\varphi = -ml(\dot{x}\dot{\varphi} + g)sin\varphi$$

This provides the first equation:

$$ml(\ddot{x}cos\varphi - \dot{x}\dot{\varphi}sin\varphi + \ddot{\varphi}l) = -ml(\dot{x}\dot{\varphi} + g)sin\varphi$$

$$\ddot{x}cos\varphi + \ddot{\varphi}l + gsin\varphi = 0$$

For the coordinate *x* we get

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} = \frac{d}{dt}(m\dot{x} + \dot{\varphi}lcos\varphi + M\dot{x}) = (m + M)\ddot{x} + \ddot{\varphi}lcos\varphi - \dot{\varphi}^2lsin\varphi$$

$$\frac{\partial L}{\partial x} = -Dx$$

This provides the second equation:

$$(m + M)\ddot{x} + \ddot{\varphi}lcos\varphi - \dot{\varphi}^2lsin\varphi + Dx = 0$$

If we multiply the first equation by $cos\varphi$ and form the difference of the equations, we get

$$\ddot{x}\big(cos^2\varphi - (m + M)\big) + gsin\varphi cos\varphi + \dot{\varphi}^2lsin\varphi - Dx = 0$$

During implementation, we will ensure that $m + M > 1$ is. Then applies:

$$\ddot{x} = \frac{\dot{\varphi}^2lsin\varphi + gsin\varphi cos\varphi - Dx}{m + M - cos^2\varphi}$$

We can use this result for $\ddot{x}$ into the first equation and after a short transformation we obtain

$$\ddot{\varphi} = \frac{Dxcos\varphi - (m + M)gsin\varphi - \dot{\varphi}^2lsin\varphi cos\varphi}{l(m + M - cos^2\varphi)}$$

To obtain a first-order differential equation system, we set :

$$u_1 = x, v_1 = \dot{x}, u_2 = \varphi, v_2 = \dot{\varphi}$$

This gives us the system of equations for the Runge Kutta method:

$$\begin{cases} \dot{u}_1 = v_1 \\ \dot{v}_1 = \dfrac{v_2^2 l sinu_2 + g sinu_2 cosu_2 - Du_1}{m + M - cos^2 u_2} \\ \dot{u}_2 = v_2 \\ \dot{v}_2 = \dfrac{Du_1 cosu_2 - (m+M) g sinu_2 - v_2^2 l sinu_2 cosu_2}{l(m + M - cos^2 u_2)} \end{cases}$$

During implementation l is determined by the manual positioning of the mass *m.* Furthermore, we normalise in the implementation $m = 1$. The choice of D, M will be discussed later.

We will need the total energy of the system for the implementation. As the following applies at the start $\dot{x} = 0, \dot{\varphi} = 0$ this is only the potential energy:

$$E_{Start} = \frac{1}{2} Dx^2 - mglcos\varphi$$

Their range is: $E_{min} = -mgl, E_{max} = \frac{D}{2} + mgl$, as $x \in [-1,1], \varphi \in [-\pi, \pi]$.

The energy at any point in time:

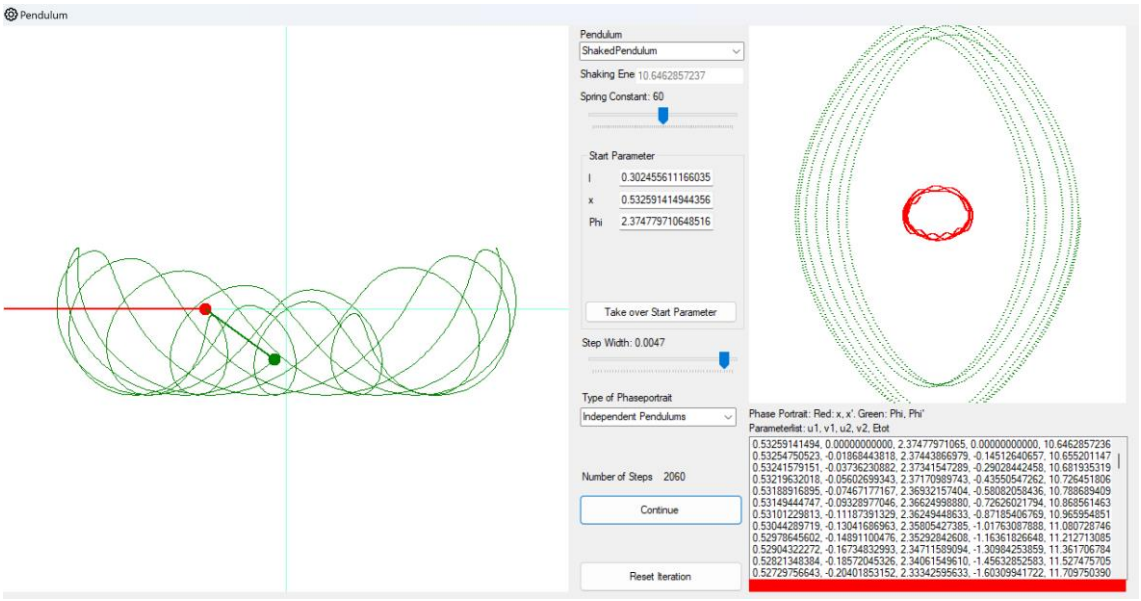
$$E_{tot} = \frac{1}{2} m[\dot{x}^2 + 2\dot{x}\dot{\varphi} lcos\varphi + \dot{\varphi}^2 l^2] + \frac{1}{2} M\dot{x}^2 + \frac{1}{2} Dx^2 - mglcos\varphi$$

During implementation, l must be selected so that the thread pendulum remains visible. For reasons of symmetry, it is sufficient to consider the case $x > 0$ case. If also $\varphi > 0$ is, then the following should apply:

$$x + lsin\varphi < 1$$

If $x < 0$ and $\varphi < 0$ applies analogously: $|x| + l|sin\varphi| < 1$. This is realised in the implementation.

After some experimentation, the implementation $M = 2$ and the spring constant *D was* made variable as an additional parameter.



Shaked pendulum

6.8. Exercise examples

1. For an oscillating spring pendulum, investigate for any angle $\varphi$for which values of $l(\varphi)$ the spring just cancels the gravitational force.

2. Suppose you have a double pendulum and the following applies: $l_1 = l_2 = {}^1/_2, m_1 = m_2 = 1$. At the start is $\varphi_1 = \varphi_2 > 0$. Does then also apply after further iteration steps $\varphi_1 = \varphi_2$? (Examine the corresponding formulae of the Runge Kutta method).

3. Experiment with the "Simulator" and investigate for which initial conditions the total energy of each pendulum remains more or less constant during the pendulum motion.

4. Examine a shaker pendulum in which the spring pendulum swings up and down in a vertical direction. A thread pendulum is attached to this spring pendulum.

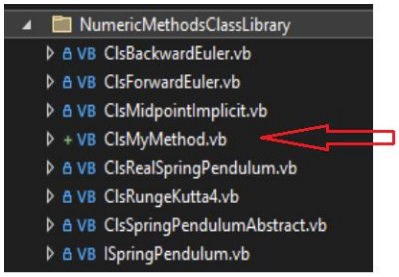
    a) Calculate the Lagrange function for this pendulum
    b) Use this to derive the differential equations that describe the movement
    c) Use suitable substitution to convert these differential equations into a system of first-order differential equations
    d) Use this to create the iteration equations for the Runge Kutta method
    e) The pendulums should start from a rest position, i.e. only potential energy is present at the start and the kinetic energy is zero at the start. In which interval does the total energy of the system lie?
    f) Implement this pendulum in the "Simulator"

7. Implementation of own variants

If the user of the "Simulator" wants to programme his own variants of the existing implemented systems, this is very easy to do. We would like to explain this using the example of a customised variant of a numerical method.

1. Step

Create your own *ClsMyMethod* class and add it to the *NumericMethodsClassLibrary* folder:



It is important to adhere to the convention that class names begin with the prefix *Cls.*

2. Step

This class must inherit the *ClsSpringPendulumAbstract* class. The only part of the code that changes compared to the existing classes is the algorithm of the numerical method:

```
0 Verweise
Public Class ClsMyMethod
    Inherits ClsSpringPendulumAbstract

    Private u As Decimal
    Private v As Decimal


    5 Verweise
    Protected Overrides Sub Iteration()

        Dim i As Integer


        With MyActualParameter
            For i = 1 To MyNumberOfApproxSteps
                .Component(0) += MyH

                'Component(1) holds the y-value
                u = .Component(1)
                v = .Component(2)

                'this is the numerical approximation
                .Component(1) = My own formula For the first component
                .Component(2) = My own formula For the second component
            Next

            'the Conponent(0) holds the "time" t with 2*pi period
            .Component(0) = .Component(0) Mod CDec(2 * Math.PI)

        End With

    End Sub
End Class
```

The code could then look as shown above.

Further steps are not necessary! In particular, the combo box in the *FrmSpringPendulum* is filled with the selection of numerical methods by *reflection.* So, no change of the code there is necessary. Your own method is listed as *MyMethod* in the combo box. This means that the prefix *Cls* is cut out of the name.

3. Step (optional)

If you want to choose different names in German and English than *MyMethod*, you can enter a corresponding key in the resource files *LabelsEN* and *LabelsDE.* This is done as follows:

In *LabelsEN*:

ClsMyMethod	My own numeric method	Cls

In *LabelsDE*:

ClsMyMethod	Meine eigene numerische Methode	Cls

In both cases, the key (name of the entry) is the class name, the value is your own selected designation and the comment 'Cls'.

The same applies to other custom extensions for unimodal functions or billiards. The corresponding interface (*IIterator* or *IBilliardball*) must be implemented there. However, this can be more complex from a mathematical point of view, especially for billiards.

Bibliography

Listed here are a few sources where individual chapters are still within an elementary accessible framework. Based on these sources, you will find further literature references.

[1] Wolfgang Metzler: Nonlinear Dynamics and Chaos, Teubner Studienbücher 1998

[2] Urs Kirchgraber: Mathematics in Chaos, Mathematische Semesterberichte, Springer 1992

[3] Urs Kirchgraber: Chaotic behavior in simple systems, Berichte über Mathematik und Unterricht, ETHZ, 1992

[4] Urs Kirchgraber, Niklaus Sigrist: Feigenbaum Universalität: Beschreibung und Beweisskizze, Berichte über Mathematik und Unterricht, ETHZ, 1995

[5] Pierre Collet, Jean-Pierre Eckmann: Iterated maps on the Interval as dynamical Systems, Birkhäuser 1986

[6] Moritz Adelmeyer: Theorem of Sarkovskii, Berichte über Mathematik und Unterricht, ETHZ, 1990

[7] Serge Tabachnikov: Geometry and Billiards, Springer Spektrum 2013

[8] Anna-Maria Vocke: The Poincaré-Birkhoff fixed point theorem and applications to billiards, Bachelor thesis, Mathematical Institute, Westfälische Wilhelms-Universität Münster, 2014

[9] Bastian von Harrach: Numerik von Differentialgleichungen, Lecture Notes, Goethe University Frankfurt am Main, Institute of Mathematics, 2022

[10] Robin Santra: Einführung in den Lagrange- und Hamilton-Formalismus, Springer Spektrum 2022

[11] Stefan Frei: Numerische Mathematik, Vorlesungsskript an der Universität Konstanz, WS 2021,2022

[12] Hans Jürgen Korsch, Hans-Jörg Jodl, Timo Hartmann: Chaos, Springer Verlag 2008

[13] Dierk Schleicher, Malte Lackmann, *Hrsg*.: Eine Einladung in die Mathematik, Springer Spektrum 2013

[14] Reinhold Remmert. Georg Schumacher: Funktionentheorie 1, Springer Verlag 2002