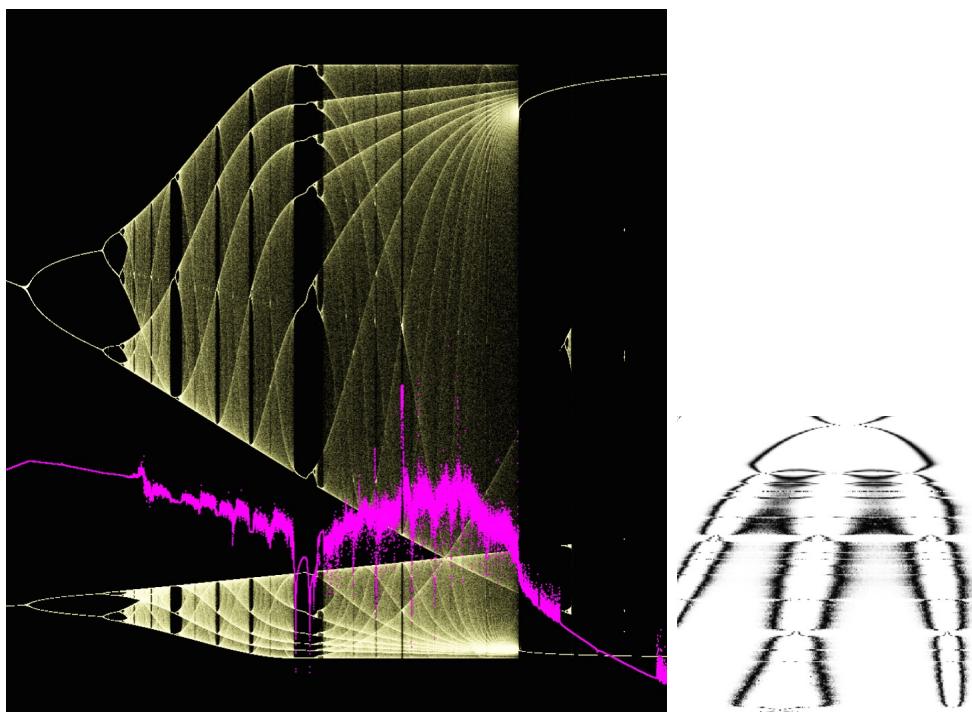


Protokoll zum Versuch Nichtlineare Dynamik und Chaos

Nicolas Heimann, Jesse Hinrichsen

Universität Hamburg

2015



Zusammenfassung

In diesem Protokoll haben wir die Eigenschaften von chaotischen Systemen anhand der logistischen Gleichung, der Sinusabbildung, der Duffing-Gleichung und dem LDR Schwingkreis numerisch bestimmt und analysiert. Ebenfalls haben wir den LDR Schwingkreis in einem Experiment anhand eines analogen Oszilloskopes untersucht. Alle Plots und Simulationen in diesem Protokoll haben wir im Rahmen des Versuches selber implementiert. Dafür wählten wir als Programmiersprache Python2.7 und nutzten OpenGL4.1 und OpenCL für Visualisierungen und Berechnungen. Der Quellcode ist über Github einsehbar: https://github.com/keksnico/gl_plotting_experimental. Bei Abbildungen ist ein entsprechender Quellcodeverweis angegeben.

Inhaltsverzeichnis

1 Logistische Abbildung	2
1.1 Bifurkationsdiagramm	5
1.2 Fixpunkte / Stabilitätsbedingung	5
1.3 Konvergenzverhalten	7
1.4 Betrachtung von Extremwerten	9
1.5 Lyapunov Exponent	9
1.5.1 Konvergenzverhalten	10
1.5.2 Betrachtung im chaotischen Bereich.	10
1.5.3 Vergleich der analytischen und renormierten Implementation	11
1.6 Feigenbaumkonstante	12
2 Sinus Abbildung	13
3 Duffing-Gleichung	15
3.1 Phasenraumdiagramm	15
3.2 Poincareschnitt	17
4 LDR-Oszillator	17
4.1 Theorie	17
4.2 Numerische Berechnungen	19
4.2.1 Phasenraumdiagramme	19
4.2.2 Poincaré-Schnitt	22
4.2.3 Bifurkationsdiagramm-Diagramm	22
4.3 Versuchsdurchführung	23
4.3.1 Versuchsaufbau	23
4.3.2 Oszillation und Phasenraumdiagramm	23
4.4 Bifurkationsdiagramm	23
4.5 Vermessung der Bifurkationspunkte	25
4.6 Auswertung	25
5 Zusammenfassung	27
5.1 Software	27
6 Appendix	28
6.1 Runge-Kutta 4ter Ordnung in 2 Dimensionen	28
6.2 Algorithmus zum bestimmen des Bifurkationspunktes	29
6.3 Lyapunov renormiert - OpenGL GLSL Quellcode	30

Im folgenden bezeichnet $f^2(x) = f(f(x))$

1 Logistische Abbildung

Die logistische Abbildung ist gegeben durch $f(x_n) = x_{n+1} = rx_n(1 - x_n)$. Es zeigt sich das diese einfache Funktionsvorschrift bereits chaotisches Verhalten an den Tag legt welches wir im folgenden Abschnitt genauer untersucht haben.

Betrachtet man die Folge x_n stellt man fest, dass diese für bestimmt (x_0, r) konvergiert und für andere (x_0, r) nicht konvergiert. In Abbildung 1 ist die Folge x_n für verschiedene Parameter r visualisiert:

- Oben links ($r = 2.8$): hier deutet sich an, dass x_n eine konvergente Folge ist. Bereits nach wenigen Iterationen ist der Grenzwert erreicht.
- Oben rechts ($r = 3.2$): es existiert kein eindeutiger Grenzwert. Vielmehr scheint x_n gegen eine alternierende Folge zu konvergieren.
- Unten links ($r = 3.45$): es zeigt sich, dass nun 4 bestimmte Werte abwechselnd abgebildet werden.
- Unten rechts ($r = 3.8$): es ist keine Korrelation zwischen den Folgegliedern zu erkennen.

Obwohl für $r = 3.2$ und $r = 3.45$ kein eindeutiger Grenzwert existiert, scheint eine Struktur erkennbar zu sein, denn hier alterniert x_n zwischen verschiedenen Werten in $[0, 1]$.

Wir haben nun für $r = 3.8$ untersucht wie sich zwei Folgen x_n und y_n entwickelt falls $y_0 = x_y + 0003$ ist. In Abbildung 2 ist der Abstand zwischen den Folgegliedern $|x_n - y_n|$ visualisiert. Die ersten 10 Iterationen ist $x_n \approx y_n$. Ab $n > 12$ ändert sich der Abstand zwischen den Folgegliedern stets. Eine kleine Änderung des Startwertes $x_0 \rightarrow x_0 + \epsilon$ hat bei $r = 3.8$ große Auswirkungen auf den Verlauf der Folge.

Die logistische Abbildung ist eine Parabel $\Rightarrow f^n(x)$ ist ein Polynom vom Grad $2n$. In Abbildung 1 wurde für einen festen Startwert die Folge x_n aufgetragen. Um nun beliebige Startwerte zu untersuchen haben wir in Abbildung 3 das Polynom $f^{100}(x)$ vom Grad 200 dargestellt:

- Oben links($r = 2.8$): $f^{100}(x)$ konvergiert gegen eine konstante Funktion $g(x) = c \in \mathbb{R}$.
- Oben recht($r = 3.05$): $\lim_{N \rightarrow \infty} f^N(x) \in \{y_0, y_1\} \forall x \in [0, 1]$.
- Unten links($r = 3.45$): $\lim_{N \rightarrow \infty} f^N(x) \in \{y_0, y_1, y_2, y_3\} \forall x \in [0, 1]$.
- Unten rechts($r = 3.56$): $\lim_{N \rightarrow \infty} f^N(x) \in \{y_0, y_1, y_2, y_3, y_4, y_5\} \forall x \in [0, 1]$.

Das Langzeitverhalten der Folge x_n hängt also nicht vom Startwert ab. Betrachtet man den Graphen für $r = 3.45$ stellt man fest, dass der Graph bei $r = 3.05$ in diesem Graph wieder mehrfach wiederzufinden ist. f_r^∞ scheint für bestimmte Parameter r ein fraktales Objekt zu sein (Selbstähnlichkeit).

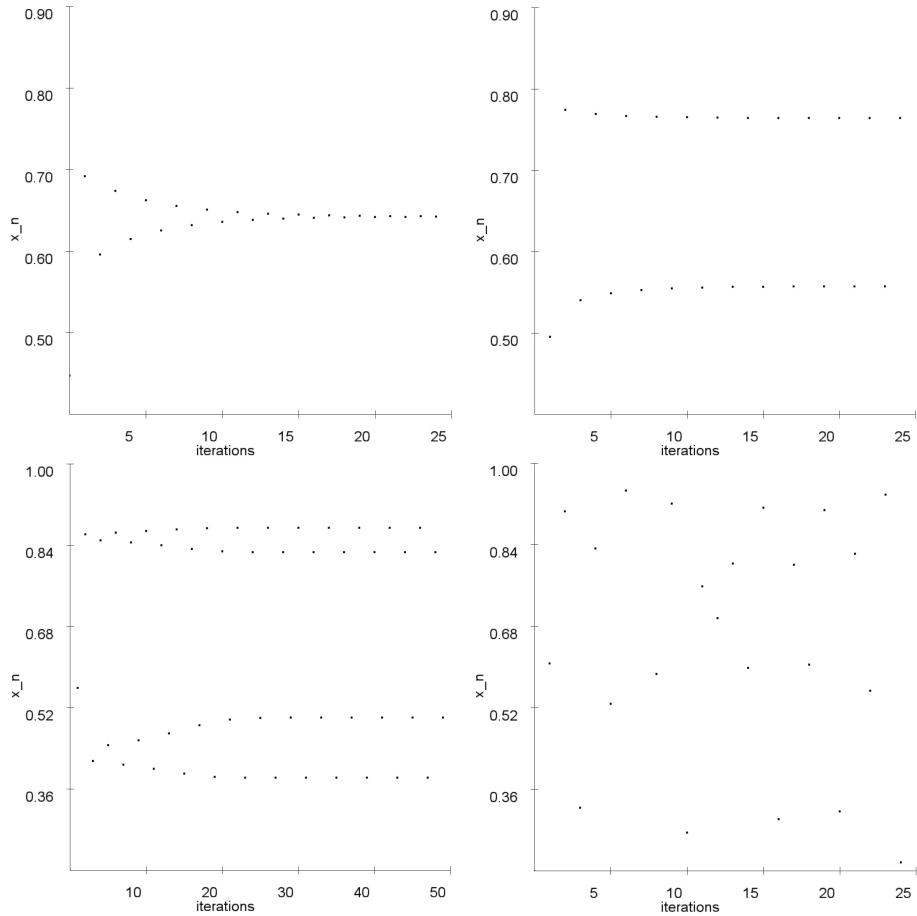


Abbildung 1: Logistische Abbildung nach 25 Iteration für den Startwert $x_0 = 0.2$. Links oben: $r = 2.8$, rechts oben: $r = 3.2$, links unten: $r = 3.45$, rechts unten: $r = 3.8$

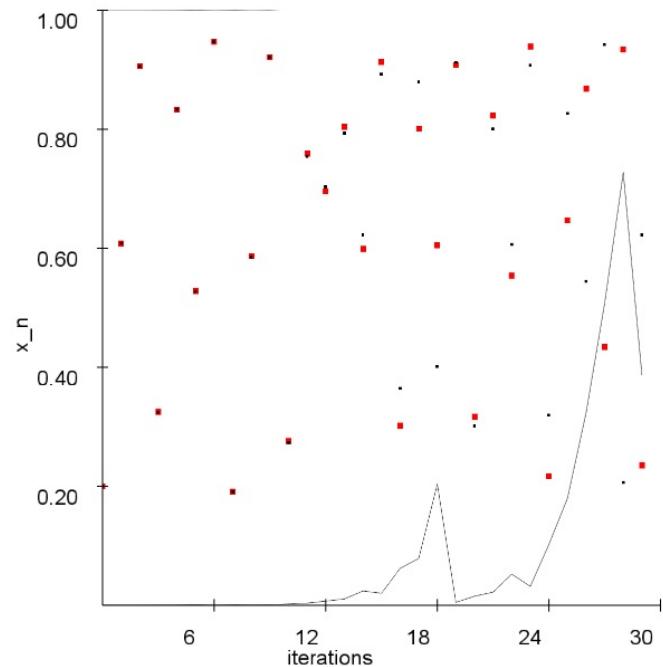


Abbildung 2: Logistische Abbildung nach 25 Iteration für den Startwert $x_0 = 0.2$ (rot) und $y_0 = 0.20003$ (schwarz). Zusätzlich als Linie geplottet (zur besseren visuellen Unterscheidbarkeit) der Abstand $|x_n - y_n|$

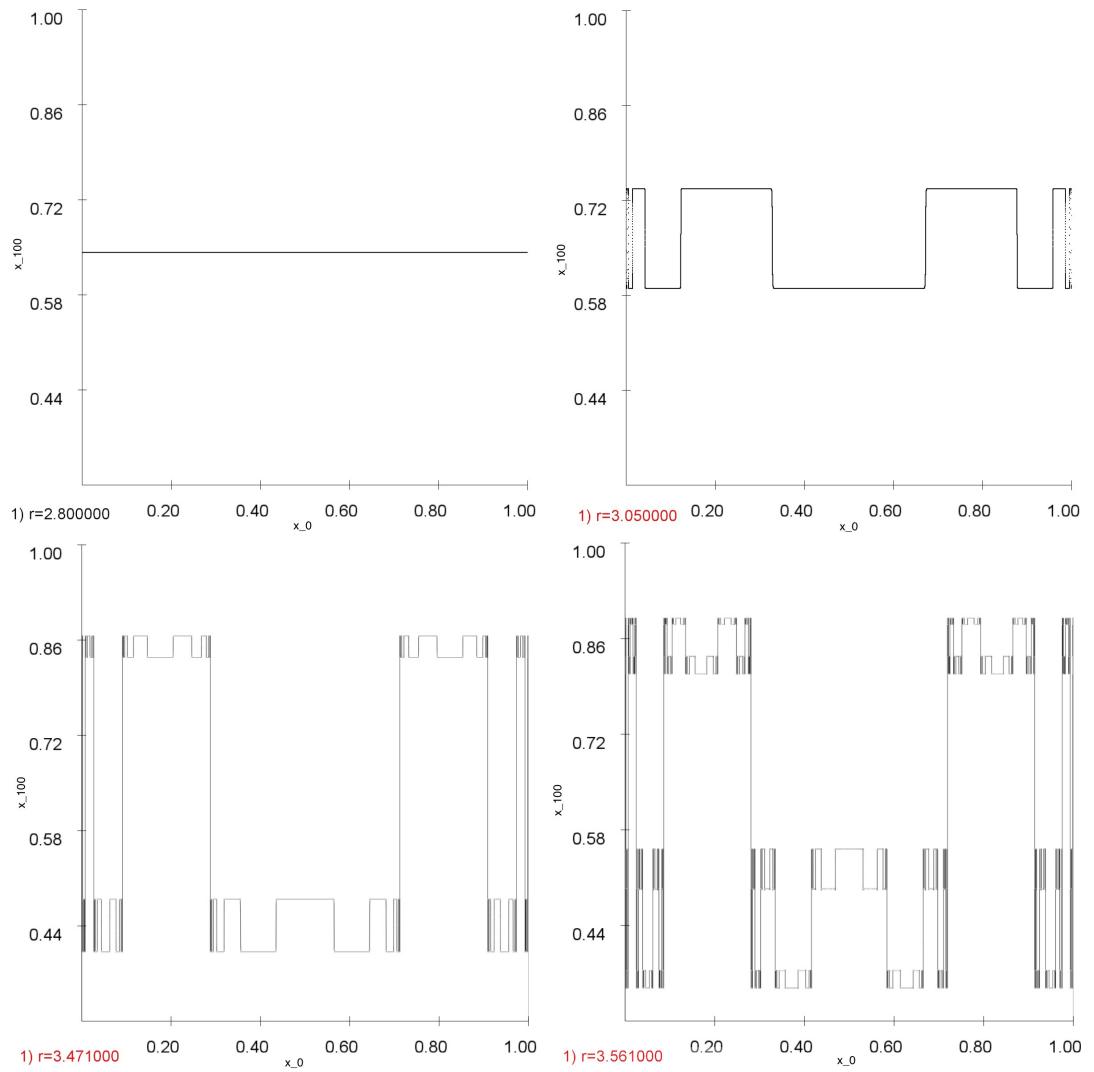


Abbildung 3: Logistische Abbildung nach 100 Iterationsschritten für $x_0 \in [0, 1]$. Links oben: $r = 2.8$, rechts oben: $r_3.05$, links unten: $r = 3.45$, rechts unten: $r = 3.56$

1.1 Bifurkationsdiagramm

Wir haben ein Bifurkationsdiagramm der logistischen Abbildung erzeugt indem wir den Parameter r gegen Iterationspunkte aufgetragen haben. Dabei fixierten wir jeweils ein r und erzeugten eine Folge $x_0 \dots x_{1000}$ von welcher wir $x_{500} \dots x_{1000}$ gegen die Ordinate aufgetragen haben (Abbildung 4). Das Bifurkationsdiagramm lässt sich in mehrere Bereiche unterteilen. Bis $r = 3$ laufen die $x_{500} \dots x_{1000}$ auf den gleichen Fixpunkt zu. An $r = 3$ gabelt sich das Diagramm in zwei Äste auf (Periodenverdoppelung). An $r = 3.449$ gibt es eine weitere Periodenverdopplung und es ist eine Selbstähnlichkeit mit dem Bereich um $r = 3$ zu erkennen (fraktale Strukturen). Ab $r = 3.569$ entsteht ein chaotischer Bereich in welchem sich aber noch Strukturen feststellen lassen (Bögen, Punkte auf Geraden, freie Bereiche). In dem Bereich um $r = 3.84$ (Abbildung 4 rechts) findet sich eine zu (Abbildung 4 links) selbstähnliche Struktur.

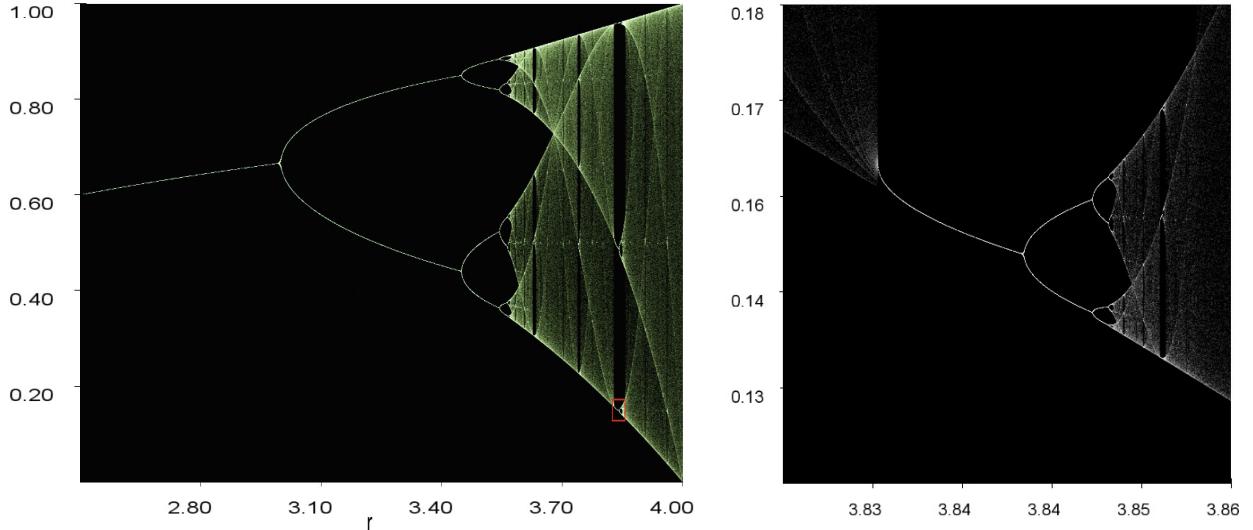


Abbildung 4: Links: Bifurkationsdiagramm der logistischen Abbildung im Bereich $r \in [2.6, 4]$. sourcecode: prak/birukation-logistisch-no-opt.py. Rechts: Vergrößerung im Bereich $r \in [3.82, 3.88]$ im linken Bild rot markiert.

1.2 Fixpunkte / Stabilitätsbedingung

Bildet die Funktion einen Punkt idempotent ab, so handelt es sich um einen Fixpunkt, es gilt: $f(x) = x \iff x$ ist Fixpunkt. Des weiteren unterscheidet man zwischen stabilen und instabilen Fixpunkten. Ist der Fixpunkt stabil so konvergiert $f^n(x)$ für beliebige x bei $n \rightarrow \infty$ gegen diesen Fixpunkt. Im Fall eines instabilen Fixpunktes entfernt sich das Ergebniss nach jeder Iteration für große n vom Fixpunkt.

Über die Ableitung der Funktion am Fixpunkt lässt sich eine Aussage über die Stabilität machen. Sei x ein Fixpunkt, dann gilt

$$|f'(x)| < 1 \iff \text{Fixpunkt - stabil} \quad (1)$$

$$|f'(x)| > 1 \iff \text{Fixpunkt - instabil} \quad (2)$$

Da wir uns für das Konvergenzverhalten interessieren genügt es das Verhalten bei großem n zu untersuchen. D.h. wir gehen davon aus, dass nach n Iterationen das Ergebnis nahe am Fixpunkt liegt. Damit können wir x als Störung des Fixpunktes x^* ausdrücken und linear approximieren. Es sei $x_{n+1} = f(x_n) = f(x^* + \delta)$, wobei x^* ein Fixpunkt von $f(x)$ ist. Dann folgt für die Taylorentwicklung ersten Grades

$$x_{n+1} = f(x^* + \delta) = f(x^*) + f'(x^*)\delta \quad (3)$$

$$\Leftrightarrow x_{n+1} - x^* = f'(x^*)\delta = (x_n - x^*)f'(x^*) \quad (4)$$

Damit x_n tatsächlich gegen x^* konvergiert also x^* ein stabiler Fixpunkt ist, muss gelten

$$|x_{n+1} - x^*| < |x_n - x^*| \Leftrightarrow |f'(x^*)| < 1 \quad (5)$$

Für die logistische Abbildung haben wir die Fixpunktgleichung $f(x) = rx(1-x) = x$. Die Lösungen der Gleichungen sind die Fixpunkte $x_0 = 0$ und $x_1 = 1 - \frac{1}{r}$. Aus der Stabilitätsbedingung (wir betrachten $f'(x) = r - 2rx$) folgt:

$$-1 < r < 1 : x_0 \text{ stabil}, r > 1 : x_0 \text{ instabil} \quad (6)$$

$$1 < r < 3 : x_1 \text{ stabil}, r < 1 \wedge r > 3 : x_1 \text{ instabil} \quad (7)$$

Abbildung 5 zeigt wie sich die Iteration bei unterschiedlichen Parametern verhalten. Für $r > 3$ ist x_1 nicht mehr stabil. So lässt sich die Aufspaltung im Bifurkationsdiagramm (Abbildung 4) für $r > 3$ verstehen. Wir haben nun die Fixpunkte des Einerzyklus numerisch bestimmt. Um die Konvergenz zu beschleunigen haben wir die folgende Abbildungsvorschrift angewendet:

$$x_0 \in [0, 1]; x_{n+1} = \frac{x_n + f(x_n)}{2} \quad (8)$$

Wir haben die Fixpunkte x^* nach 18 Iterationen numerisch auf $|x^* - f^{15}(x_0)| < 10^{-15}$ bestimmen können (Abbildung 6). Um zu untersuchen wie schnell dieses Verfahren für bestimmte x^* konvergiert, haben wir den Abstand des Fixpunktes zu dem berechneten Wert $|x^* - x_{10}|$ nach 10 Iterationen berechnet. Tabelle 1 zeigt, dass um $r = 3$ die x_{10} am dichtesten am Grenzwert x^* sind. Der Bifurkationspunkt befindet sich dort, wo der Fixpunkt weder stabil noch instabil ist:

r	$ x^* - x_{10} $	r	$ x^* - x_{10} $
0.50	1.03	2.60	2.0110^{-08}
0.73	4.3310^{-01}	2.83	6.6610^{-11}
0.97	1.7210^{-01}	3.07	1.3510^{-14}
1.20	5.8610^{-02}	3.30	5.5510^{-09}
1.43	1.6210^{-02}	3.53	7.0210^{-07}
1.67	3.3910^{-03}	3.77	1.3410^{-05}
1.90	4.9410^{-04}		
2.13	4.1410^{-05}		

Tabelle 1: Abstand der Folge x_{10} zum Fixpunkt x^* unter Anwendung der Konvergenzbeschleunigung.

$$x^* \text{ Bifurktionspunkt} \iff |f(x^*)| = 1 \Rightarrow |2 - r| = 1 \Rightarrow r = 3 \quad (9)$$

Im Bereich den Bifurkationspunktes $r = 3$ konvergiert die logistische Abbildung ohne die Konvergenzbeschleunigung sehr langsam (Abbildung 5). Allerdings zeigte sich bereits in Tabelle 1, dass um den Bereich des Bifurkationspunktes die Konvergenzbeschleunigung am besten funktioniert. Mit dem Differenzenquotienten und der Konvergenzbeschleunigung haben wir so den Bifurkationspunkt numerisch effizient bestimmen können (siehe Appendix):

$$|r_{analytisch} - r_{numerisch}| < 6 * 10^{-9} \quad (10)$$

Im folgenden untersuchen wir den Zweierzyklus $f^2(x)$. Mit der Fixpunktgleichung $f(f(x)) = x$ wissen wir, dass ein Fixpunkt von $f^n(x)$ immer auch ein Fixpunkt von $f^{n+1}(x)$ ist. Im Fall der logistischen Abbildung folgt

$$x_{n+1} = f_r(x_n) = rx_n(1 - x_n) \Rightarrow x_{n+2} = r^2x_n(1 - x_n)(1 - rx_n(1 - x_n)) \quad (11)$$

Fixpunktgleichung (Zweierzyklus):

$$x = r^2x(1 - x)(1 - rx(1 - x)) \quad (12)$$

$$\Leftrightarrow x(1 - r + rx)(1 + r - rx - r^2x + r^2x^2) = 0 \quad (13)$$

In dieser Form lassen sich, wie zu erwarten, die Lösungen $x_0 = 0$ und $x_1 = 1 - \frac{1}{r}$ wiederfinden. Die Lösung der Gleichung

$$1 + r - rx - r^2x + r^2x^2 = 0 \quad (14)$$

führt zu 2 weiteren Fixpunkten

$$x_{3,4} = \frac{\pm\sqrt{r^2 - 2r - 3} + r + 1}{2r} \quad \forall r^2 - 2r - 3 \geq 0 \Rightarrow r \leq -1 \vee r \geq 3 \quad (15)$$

Unter Berücksichtigung der Stabilitätsbedingung folgt

$$\frac{d}{dx}f^2(x) = -r^2(2x - 1)(2r(x - 1)x + 1). \quad (16)$$

Lösungen der Gleichung $|\frac{d}{dx}f^2(x_{3,4})| = 1$ sind: $r_0 = -1$, $r_1 = 3$, $r_2 = 1 - \sqrt{6} \approx -1.45$, $r_3 = 1 + \sqrt{6} \approx 3.45$.

Da die Fixpunkte $x_{3,4}$ nur für $r > 3$ und $r < -1$ existieren folgt.

$$1 - \sqrt{6} < r \leq -1 \wedge 3 \geq r > 1 + \sqrt{6} : x_{3,4} \text{ stabil} \quad (17)$$

$$r < 1 - \sqrt{6} \wedge r > 1 + \sqrt{6} : x_{3,4} \text{ instabil} \quad (18)$$

Die logistische Funktion zeigte bei einigen Parametern r Intermittenz (Abbildung 6). Intermittenz ist ein Grund für die Verdichtungen welche im chaotischen Bereich des Bifurkationsdiagrammes zu sehen sind.

1.3 Konvergenzverhalten

Wir haben das Konvergenzkriterium

$$|f_r^\infty(x_0) - f_r^m(x_0)| < \epsilon \quad (19)$$

untersucht. Zunächst haben wir den Fall $r = 3$ untersucht: Abbildung 7 (links) trägt x_n gegen n Iterationen bei einem Startwert $x = 0.4$ auf. Nach 10^7 Iterationen erhalten wir eine Abweichung vom erwarteten Wert von $\epsilon = 7.45 \cdot 10^{-5}$.

Als nächstes haben wir $\forall(x_0, r) \in [0, 1] \times [0, 4]$ berechnet wie groß $|f_r^\infty(x_0) - f_r^m(x_0)|$ ist. In einem 2d Plot (Abbildung 7 Mitte, rechts) sind (x_0, r) eingefärbt: Je dunkler (x_0, r) desto näher liegt $f_r^m(x_0)$ an $f_r^\infty(x_0)$. Wie zu erwarten ist der Fixpunkt $x_1 = 1 - \frac{1}{r}$ dunkel eingefärbt (Abbildung 7 Mitte). Der Graph scheint an der $x_0 = 0.5$ Achse gespiegelt zu sein. Dies hängt damit zusammen, dass $f(x_1^{Spiegel} = \frac{1}{r}) = x_1$. Deutlich zu sehen sind die schwarzen Bereiche um die superattraktiven Stellen und die weißen Bereiche an den Bifurkationspunkten. Je mehr Iterationen durchgeführt werden, desto mehr prägen sich die schwarzen Bereiche aus. Für $r < r_{superattraktiv}$ sind 2 Äste zu sehen, während bei $r > r_{superattraktiv}$ abhängig von der Anzahl N der Iterationen $2(N - 1)$ Äste zu sehen sind. An den Bifurkationspunkten wiederholt sich das Muster fraktal (Abbildung 7 rechts).

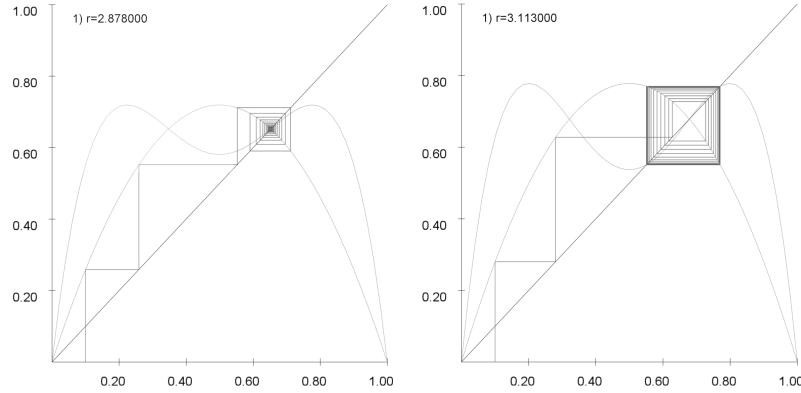


Abbildung 5: Verlauf der Iterationen bei festem Parameter r . Links: Konvergiert gegen Fixpunkt x_1 bei $r=2.878$. Rechts: Bei $r=3.113$ ist x_1 kein stabiler Fixpunkt mehr. Der Verlauf der logistischen Funktion $f(x)$, $f^2(x)$ sowie die Einheitsgerade $y = x$ sind aufgetragen. Als Linie Verbunden geplottet sind die Folgen $(x_n, 0.0), (x_n, x_{n+1}), (x_{n+1}, x_{n+1}), (x_{n+1}, x_{n+2}), (x_{n+2}, x_{n+2}), (x_{n+2}, x_{n+3}), \dots$. Sourcecode: prak/logisitsch-no-opt-behavior.py

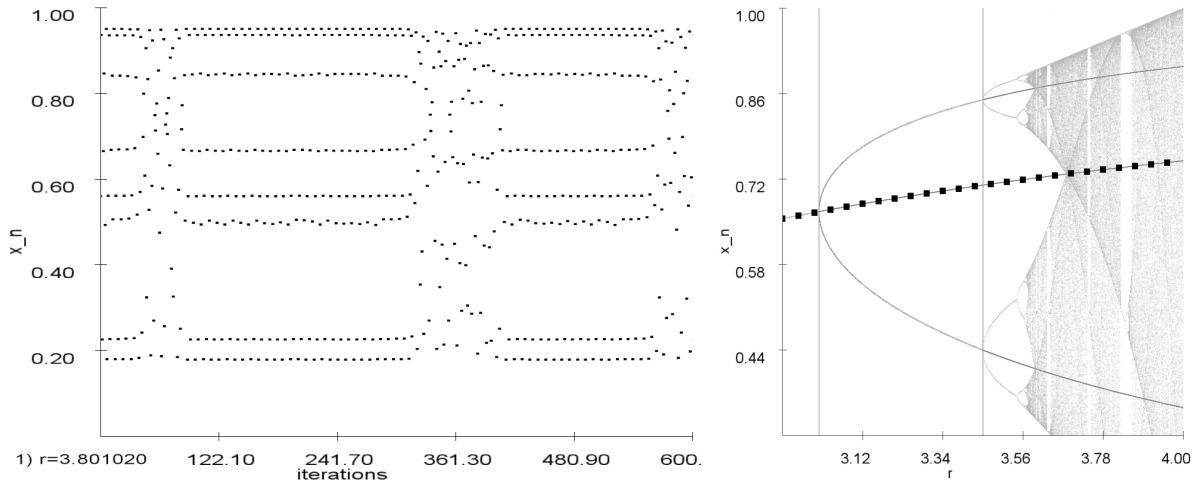


Abbildung 6: Logistische Funktion. Links: Intermittenz bei $r = 3.80102$. Rechts: Die stabile Lösung des Einerzyklus x^* und die beiden Lösungen des Zweierzyklus $x_{3,4}$. Im Hintergrund das Bifurkationsdiagramm. Die Punkte sind das Ergebnis der numerischen Berechnung der Fixpunkte x_1 . Die vertikalen Linien sind an $r = 3.0$ und $r = 3.44$ markiert und markieren die Stellen wo $|f'(x)| = 1$, $|\frac{d}{dx}f^2(x)| = 1$ sind. Sourcecode: prak/intermittenz.py, prak/logis-zyklen.py

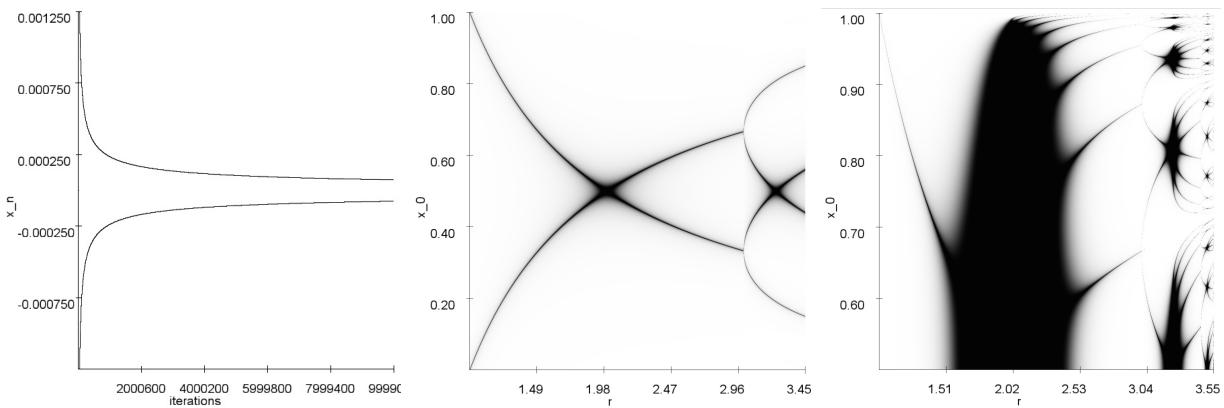


Abbildung 7: Links: Konvergenzverhalten der logistischen Abbildung bei $r = 3$. $f^n(x)$ konvergiert kontinuierlich gegen $x = \frac{2}{3}$ (Der Graph wurde um $-\frac{2}{3}$ translatiert). Mitte: Konvergenzverhalten der logistischen Abbildung. Punktfarbe $F = \exp(-\epsilon / |f_r^2(x_0) - f_r^{10000}(x_0)|)$. Rechts: Punktfarbe $F = \exp(-\epsilon / |f_r^{10}(x_0) - f_r^{10000}(x_0)|)$ mit $\epsilon = 0.00004$. sourcecode: prak/bifurk3.py, prak/bifurk-startwert.py

1.4 Betrachtung von Extremwerten

Das Bifurkationsdiagramm scheint ab $r > 4$ zu divergieren. Im folgenden haben wir diesen Bereich näher analysiert. Wegen $x^2 < x \forall x \in (0, 1)$ ist schnell klar, dass sobald $f(I) = [0, 1]$ auch $f^n(I) = [0, 1]$. Falls aber gilt $x - x^2 < 0$ dann stellt man fest:

$$f^n(x) < \dots < f^2(x) < f(x) < 0 \forall n \in \mathbb{N} \Rightarrow \lim_{n \rightarrow \infty} f^n(x) = -\infty \quad (20)$$

Damit dieses Verhalten irgendwann einsetzt muss gelten: $\exists m : f^m(x) > 1$ denn dann ist $f^{m+1}(x) < 0$.

$$f(x_0) > 1 \iff x_0 - x_0^2 > \frac{1}{r} \iff x_0 = \pm \sqrt{\frac{1}{4} - \frac{1}{r}} + \frac{1}{2} \quad (21)$$

Also $\exists x_0 \in \mathbb{R} \forall r > 4 : f(x_0) > 1 \Rightarrow f^2(x_0) < 0$ ist. Es ist also zu erwarten, dass

$$\exists m : f^m(x_0) > 1 \Rightarrow \lim_{n \rightarrow \infty} f^n(x_0) = -\infty \forall x_0 \forall r > 4 \quad (22)$$

In prak.iteration-log-extreme.py haben wir nun zu den Anfangsbedingungen $x_0 = 0.4, r \in [3.99999, 4.999989], \Delta r = 0.000001$ überprüft wie viele Punkte r nach n Iterationen gegen $-\infty$ divergieren. Dabei haben $n = 1, n = 10, n = 100, n = 1000, n = 10000$ Iterationen durchgeführt mit folgendem Ergebnis.

```
python -m prak.iteration-log-extreme
nothing at N_ITER=1
nothing at N_ITER=10
N_ITER=100 found 680 intervals. Last interval [4.072665, 4.999989]
N_ITER=1000 found 17 intervals. Last interval [4.000251, 4.999989]
N_ITER=10000 found 1 intervals. Last interval [4.000001, 4.999989]
```

Je mehr Iterationen durchgeführt werden, desto mehr Punkte divergieren ab $r \geq 4.000001$ gegen $-\infty$. Dieses Verhalten war nach Gleichung (22) zu erwarten.

1.5 Lyapunov Exponent

Der Lyapunov Exponent beschreibt mit welcher Geschwindigkeit sich zwei naheliegende Punkte voneinander entfernen. Es gibt drei Wege den Lyapunov Exponenten zu implementieren:

$$\text{Definition : } \lambda(x_0) = \lim_{N \rightarrow \infty} \lim_{\epsilon \rightarrow 0} \frac{1}{N} \log \left| \frac{f^N(x_0 + \epsilon) - f^N(x_0)}{\epsilon} \right| \quad (23)$$

$$\text{Analytisch : } \lambda(x_0) = \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{i=0}^{N-1} \log f'(x) \quad (24)$$

$$\text{Renormiert} \quad (25)$$

In Abbildung 11 ist der Lyapunov Exponent der analytischen Implementation gezeigt. Es zeigte sich, dass die Gleichung (23) sehr schlechte Ergebnisse im Vergleich zu den beiden letzten Methoden (24), (25) ergibt. Dies liegt daran, dass schon bei kleinen ϵ die Funktionswerte sehr schnell divergieren und somit die Definition des Differenzenquotienten in (23) keinen Sinn ergibt. In Abbildung 8 (links) haben wir für $r = 0.5$ das Konvergenzverhalten von (23) mit dem Konvergenzverhalten von (24) verglichen. Es ist zu sehen, dass bei $N = 60$ der Differenzenquotient beginnt von der analytischen Formel abzuweichen und schließlich bei $N = 125$ divergiert. Die

Renormierung (25) hält diesen Abstand in jedem Iterationsschritt klein, weshalb sich der Lyapunov Exponent trotzdem ausrechnen lässt. Der Lyapunov Exponent $\lambda(x_0)$ hat seine Nullstellen dort wo im Bifurkationsdiagramm Periodenverdopplung auftritt. Umgekehrt divergiert $\lambda(x_0)$ an den superattraktiven Stellen gegen $-\infty$. Im Vergleich wird nun klar warum in Abbildung 7 große dunkle Bereiche zu finden sind, nämlich genau dort wo superattraktive Stellen sind. Man erhält also Information über das Verhalten der Abbildung für bestimmte x_0 . Tatsächlich kann man den Lyapunov-Exponenten über den mittleren Informationsverlust ausdrücken $\lambda(x_0) = -\log(2) * \delta I$ (Literatur [1], S. 10). In den Appendix ist der OpenGL Quellcode der renormierten Formel des Lyapunov Exponenten beigelegt.

1.5.1 Konvergenzverhalten

Es kann passieren, dass man einen Anfangswert wählt, für welchen der Lyapunov-Exponent kein sinnvolles Ergebnis liefert. Setzt man $x_0 = 0.5$ so divergiert die analytische Formel, denn $\log(f'(x_0)) = -\infty$. Für die renormierte und analytische Implementation haben wir untersucht wie schnell bestimmte Anfangswerte x_0 gegen den Grenzwert $\lambda(x_0)$ streben. Dafür wurde untersucht wie nah $\lambda_{N=5}(r, x_0)$ an $\lambda_{N=1000}(r, x_0)$ ist:

$$\gamma = \exp^{-500 * (\lambda_{N=5}(r, x_0) - \lambda_{N=1000}(r, x_0))^2} \quad (26)$$

In Abbildung 8 (Mitte, rechts) ist das Ergebnis zu sehen: Je dunkler ein Punkt (r, x_0) ist, desto besser konvergiert $\lambda(r, x_0)$ ($\gamma \approx 1$). Es gibt also x_0 welche ein sehr schnelles Konvergenzverhalten gegenüber anderen x_0 haben. Der oben besprochene Startwert $x_0 = 0.5$ ist in der rechten Abbildung gut bestätigt, denn hier konvergiert $\lambda(r, x_0)$ nicht.

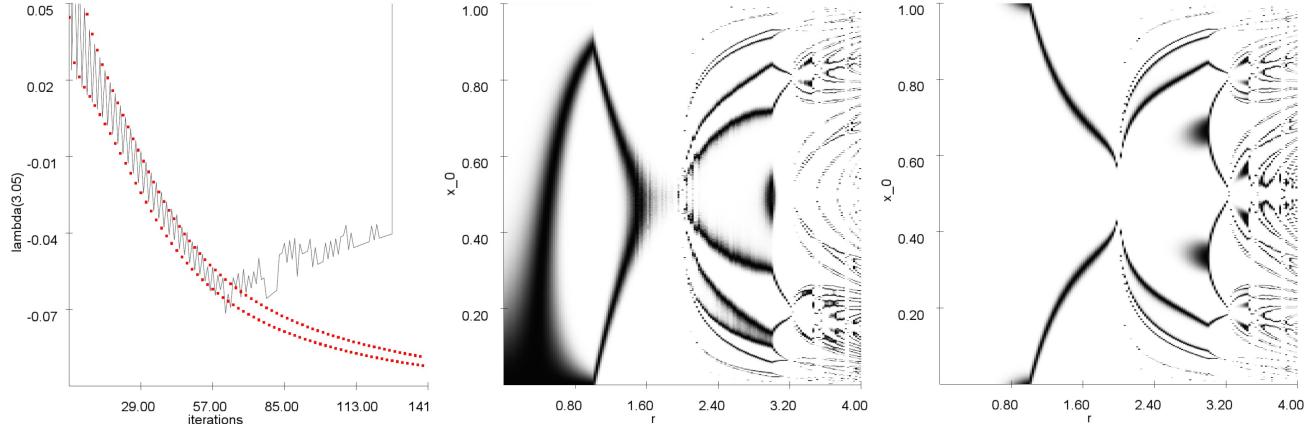


Abbildung 8: Analyse des Lyapunov Exponenten. Links: Konvergenz der analytischen Implementation Gleichung 24 (Punkte) gegen Konvergenz der Definition Gleichung 23 (Linie, zur besseren visuellen Unterscheidung wurden die diskreten Abbildungspunkte mit Linien verbunden). Mitte: Kontrollparameter r aufgetragen gegen Definitionsbereich x_0 . Je dunkler ein Punkt (r, x_0) desto besser konvergiert $\lambda(r, x_0)$ (nach Gleichung 25). Rechts: Analog zur Mittleren Abbildung für die analytische Implementation (sourcecode: prak/lyapunov-defintion-convergence.py, prak/lyapunov-rx-convergence.py)

1.5.2 Betrachtung im chaotischen Bereich.

In Abbildung 11 ist zu sehen wie der Lyapunov Exponent ab $r = 3.56$ nicht mehr stetig konvergiert. Der Verdacht liegt nahe, dass nicht genügend Iterationsschritte zur Berechnung ausgeführt wurden also erhöhten wir die Anzahl an Iterationsschritten für die Berechnung von $\lambda(x)$. Es zeigte sich allerdings kaum eine Veränderung. Die Frage nach der Konvergenz von $\lambda(x)$ im chaotischen Bereich galt es zu untersuchen: Abbildung 9 zeigt das Langzeitverhalten von $\lambda(x)$ über $19 \cdot 10^6$ Iterationen.

$\lambda(x)$ erinnert die ersten 10^6 Iterationen eher an einen Börsenkurs als an ein Objekt welches gegen einen Grenzwert konvergiert. Erst nach ca. 10^7 Iterationen deutet sich ein Konvergenzverhalten an welches aber nicht präzise ist: Falls

$$\lim_{N \rightarrow \infty} \lambda_N(x) = L \quad (27)$$

so kann man nach $2 \cdot 10^7$ Iterationen höchstens feststellen, dass der Graph in einen Epsilonschlund von $\epsilon = 0.002$ passt. Dies ist nicht sehr befriedigend angesichts der massiven Iterationslänge.

Für $r = 4$ haben wir in Abbildung 9 die 3 Varianten (23), (24), (25) miteinander verglichen. Es fällt auf, dass die renormierte Implementation (25) schneller konvergiert als die analytische Implementation (24). Die Definition (23) hingegen konvergiert gegen den Grenzwert 0 und weicht deutlich von (25), (24) ab.

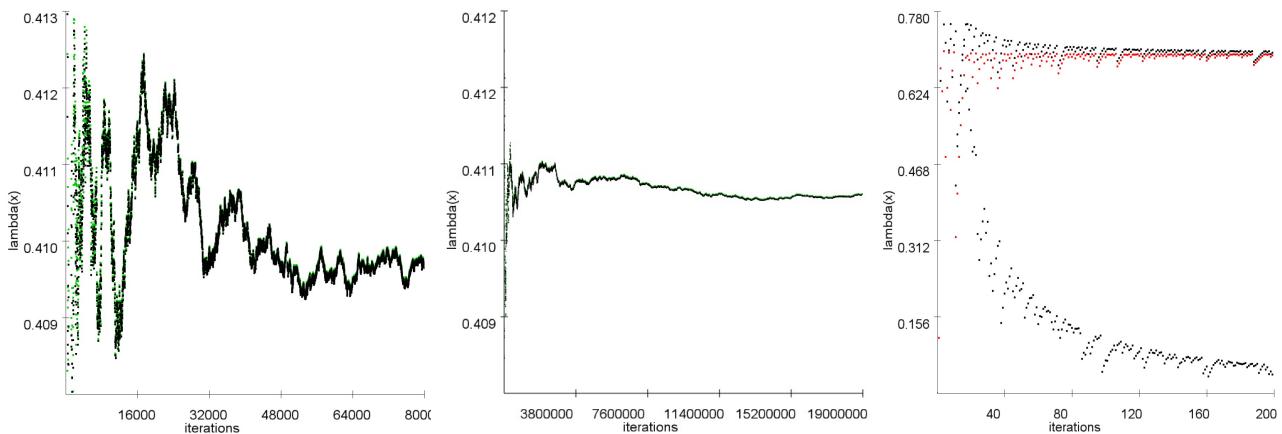


Abbildung 9: Analytischer(schwarz) und renormierter(grün) Lyapunovexponent im chaotischen Bereich bei $r = 3.78$. Links: Zoom im Bereich 0 bis $80 \cdot 10^3$ Iterationen. Mitte: Bis $2 \cdot 10^7$ Iterationen. Rechts: Vergleich zwischen den Varianten des Lyapunov Exponenten bei $r = 4$ bis $N = 200$ Iterationen. Die Definition (23) konvergiert gegen 0. Rot ist die renormierte Implementation (25). sourcecode: prak/lyapunov-definition-convergence.py, prak/lyapunov-r4.py

1.5.3 Vergleich der analytischen und renormierten Implementation

Wir haben nun die analytische und die renormierte Implementation des Lyapunovexponenten weiter untersucht. Dabei stellten wir fest, dass bei Stellen mit Periodenverdoppelung die renormierte Formel etwas schneller als die analytische Formel gegen 0 konvergiert. An einer weiteren Stelle ($r = 3.05$) ist zu erkennen wie beide Implementationen jeweils von oben (renormiert) und von unten (analytisch) gegen einen gemeinsamen Wert streben. Die Vermutung liegt nahe, dass man mit dem Mittelwert aus beiden Implementationen an solchen Stellen wesentlich schneller den Grenzwert bestimmen kann. Es zeigt sich aber, dass dieses Verhalten nicht regelmäßig auftritt weshalb wir es nicht weiter untersucht haben. Nach weiteren Stichproben scheint die analytische Implementation bis zum 500ten Iterationsschritt etwas schneller als die renormierte Implementation zu konvergieren. Beide Versionen zeigten in allen Stichproben $r < 3$, dass sie stets gegen den gleichen Grenzwert strebten.

1.6 Feigenbaumkonstante

Die Feigenbaumkonstante ist eine universelle Größe. Sie tritt in chaotischen nicht linearen Systemen auf und ist definiert als:

$$\delta_i = \frac{b_i - b_{i+1}}{b_{i+1} - b_{i+2}}, \Delta_i = \frac{s_i - s_{i+1}}{s_{i+1} - s_{i+2}} \quad (28)$$

$$\delta = \lim_{i \rightarrow \infty} \delta_i = \lim_{i \rightarrow \infty} \Delta_i = 4.669201609102991 \quad (29)$$

(Quelle: [2]) wobei s_i, b_i die Folgen der superattraktiven und periodenverdoppelnden Stellen sind. Da im Grenzfall $\lim_{i \rightarrow \infty} |\delta_i - \delta_{i+1}| = 0$ und $\lim_{i \rightarrow \infty} |\Delta_i - \Delta_{i+1}| = 0$ lässt sich die Feigenbaumkonstante sowohl mit dem superattraktiven als auch mit dem periodenverdoppelnden Stellen bestimmen. Als erstes haben wir versucht die Nullstellen des Lyapunov Exponenten zu bestimmen. Wir wendeten dabei das folgende Kriterium für Nullstellen an:

$$\lambda(x - \epsilon) < \lambda(x) \wedge \lambda(x + \epsilon) < \lambda(x) \wedge |\lambda(x)| < \epsilon \quad (30)$$

Bei genauer Betrachtung des numerisch bestimmten Lyapunov Exponenten zeigten sich große Schwankungen, weshalb dieses Kriterium nicht mit unseren verfügbaren Rechenleistungen praktikabel war (Abbildung Aufgrund dieser Probleme haben wir uns dazu entschieden die superattraktiven Stellen anstelle der Bifurkationspunkte numerisch zu bestimmen. Unser Algorithmus startet im Suchmodus bei gegebenen Startwert x_{start} und geht in kleinen Schritten Δx die x-Achse ab. In jedem Schritt wird $l_1 = \lambda(x_0)$ $l_2 = \lambda(x_0 + \Delta x)$ berechnet. Ist

$$l_2 - l_1 \leq 0 \quad (31)$$

wandert der Iterationsschritt zum superattraktiven Fall. Dies wird so lange fortgesetzt bis die Bedingung (31) nicht mehr hält. Es wird nun um Δx zurückgegangen und anschließend die Schrittweite $\Delta x \mapsto \frac{\Delta x}{10}$ verkleinert. Nun wird erneut so lange iteriert, bis (31) nicht mehr hält. Der Vorgang wiederholt sich 8 mal. Anschließend wird $\frac{2x_0 + \Delta x}{2}$ als Ergebnis gespeichert. Als nächstes befindet sich der Algorithmus im Anfangspunkt-Modus. Es wird so lange die x-Achse abgetastet bis die Bedingung

$$\lambda(x_0) < \lambda(x_0 + \Delta x) < \lambda(x_0 + 2\Delta x) \quad (32)$$

nicht mehr erfüllt ist und somit ein neues x_{start} gefunden wurde. Der Suchmodus wird aktiviert (Quellcode: prak/feigenbaum.py). Im folgenden ist die Terminal Ausgabe des Algorithmus beilegt:

```

searching from 1.9
looking for next start_r from 2.00000000002
searching from 2.99950000003
looking for next start_r from 3.23606797751
searching from 3.44927797752
looking for next start_r from 3.49856169934
searching from 3.54400769935
looking for next start_r from 3.55464086278
searching from 3.56439786279
looking for next start_r from 3.56666737986
found values [2.0000000000249916, 3.236067977509959, 3.498561699344952,
  3.554640862779951, 3.5666673798649517]
delta_0=4.70894301336
delta_1=4.68077099865
delta_2=4.66295961155

```

Somit konnten wir numerisch für $i = 3$ eine Feigenbaumkonstante von $\delta_3 = 4.66295961155$ berechnen. Dieser Wert weicht um 0.12% vom tatsächlichen Wert ab. Die Grenzen des Algorithmus sind bereits nach 5 gefunden superattraktiven Stellen erreicht. Ab $r > 3.57$ fängt die numerische Implementation des Lyapunovexpontenten an zu "rauschen" (Abbildung 11). Der Algorithmus kann daher nicht mehr präzise seinen Suchmodus ausführen. Ebenfalls liegen die nächsten superattraktiven Punkte noch dichter zusammen als es für s_4, s_5 der Fall war, was ebenfalls vom Algorithmus nicht mehr detektiert wurde. 10).

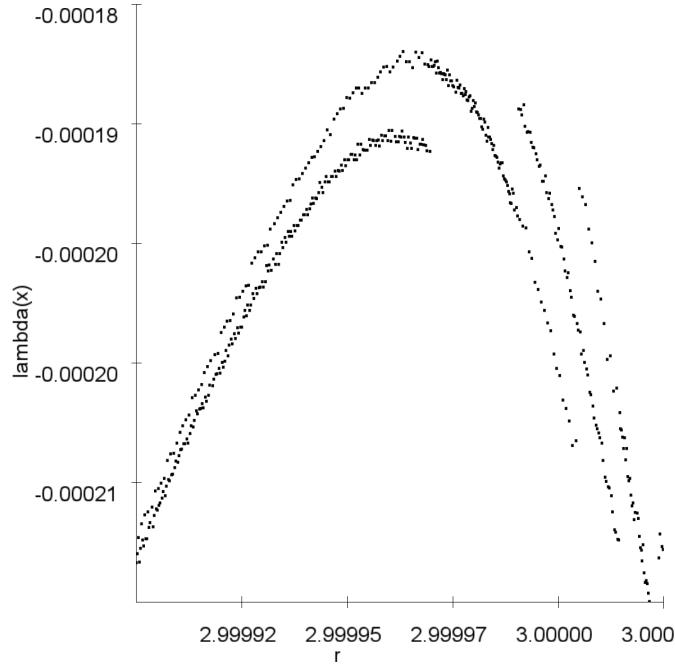


Abbildung 10: Zoom in den Lyapunovexponenten der logistischen Abbildung bei $r = 3$ mit 80000 Iterationen. Eine präzise Konvergenz ist nicht zu erkennen wodurch die numerische Bestimmung der Spitzen an den Bifurkationspunkten erschwert wird. Sourcecode: prak/lyapunov-superattraktiv.py

2 Sinus Abbildung

Die Sinusabbildung ist gegeben durch

$$f(x_n) = x_{n+1} = r \sin(x) \quad (33)$$

Wir haben die bereits implementierten Programme nun auf die Sinus Funktion angewendet. Abbildung 12 zeigt das Bifurkationsdiagramm der Sinusabbildung. Es fällt sofort auf, dass sich die Bildpunkte innerhalb einer Einhüllenden befinden. Betrachtet man die Funktion wird wegen des Bildbereiches des Sinus ($[-1, 1]$) sofort klar, dass $\sup_{n \in \mathbb{N}} |f^n(r)| \leq r$ gilt. Bei den Bifurkationsdiagrammen der logistischen Abbildung hatten wir ein festen Startwert x_0 gewählt. Für die Sinusabbildung wählten wir als Startwerte die Folge $x_{0,n} = (-1)^n x_0$, da sonst im Bereich $r \in [0, \pi]$ das Bild des Graphen stets positiv wäre. Auch dieses Phänomen lässt sich mit einfachen Überlegungen erklären:

$$\sin([0, \pi]) = [0, 1] \Rightarrow f([0, \pi]) = [0, \pi] \Rightarrow f^2([0, \pi]) = [0, \pi] \forall r \in [0, \pi] \quad (34)$$

Das Anwenden unseres Algorithmus zur Bestimmung der Feigenbaumkonstante lieferte im Vergleich zur logistischen Abbildung keine guten Ergebnisse. Die ersten superattraktiven Stellen s_i

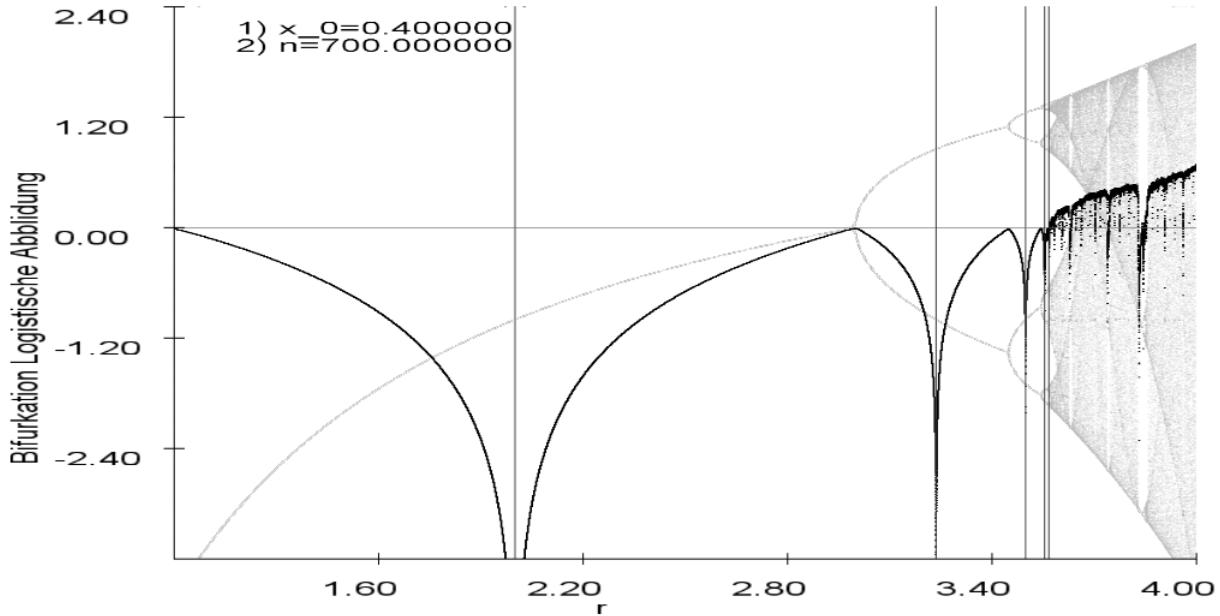


Abbildung 11: Analyse des Bifurkationsdiagrammes der logistischen Abbildung. Eingezeichnet sind die $y=0$ Achse, sowie die ersten 5 Superattraktiven Stellen. Das Bifurkationsdiagramm wurde so translatiert und skaliert, so dass es hinter dem Lyapunov Exponenten erscheint.

ließen sich ohne weiteres bestimmen. Die Abstände Δs wurden schon nach den ersten 4 Folgegliedern sehr klein und der Lyapunov Exponent beginnt zu "rauschen". Wir entwickelten einen weiteren Algorithmus, welcher nur marginal bessere Ergebnisse nach s_4 lieferte. Die Idee des Algorithmuses war es, zunächst den Lyapunov Exponenten grob für $r \in [0, 3.2]$ zu berechnen. Die so entstandene Folge $l_i = (x_i, l(x_i))$ wird nun iteriert. Sobald $l_{i,1} < M < 0$ gilt wird $A = l_{i,0}$ gesetzt. Es wird weiter iteriert bis $l_{i,1} > 0$ und schließlich wird das Intervall $[A, l_{i,1}]$ gespeichert. Dies wird bis zum Ende der Folge l_i ausgeführt. Nun wird der Algorithmus für eine tiefere Schranke auf alle ermittelten Intervalle erneut angewendet. Wir fanden so mehrere Hinweise für die nächsten superattraktiven Stellen. Schließlich wurde $M \leq 4.0$ und plötzlich fielen keine Punkte außer s_1, s_2, s_3, s_4 in das Raster. Beim betrachten des Lyapunovexponenten fiel auf, dass trotz einer Iterationstiefe von 10000 Iterationen stets $\lambda(x) > -4.0$. Wir haben dafür nur zwei plausible Erklärungen: (1.) $s_5 - s_4$ ist sehr klein und wir erwarten, dass $s_6 - s_5$ noch kleiner ist (untere Abbildung 12): Eventuell reicht unsere numerische Genauigkeit nicht aus. (2.) Der Sinus ist ungenau implementiert und verschmiert so den Grenzwert. Die folgenden superattraktiven Stellen haben wir ermittelt: $s_1 = 0.0$, $s_2 = 1.570728749999546$, $s_3 = 2.4432446071429674$, $s_4 = 2.6580499999998904$, $s_5 = 2.70599999999986$, $s_6 = 2.71600010303040$ und daraus die ersten 4 Glieder in der Feigenbaum Folge bestimmen:

$$\delta_1 = 1.8002294596$$

$$\delta_2 = 4.06188990667$$

$$\delta_3 = 4.47977878743$$

$$\delta_4 = 4.79495059736$$

Anders als die logistische Abbildung bricht das Bifurkationsdiagramm der Sinusfunktion nicht ab. Dies lässt sich auf den kompakten Bildbereich des Sinus zurückführen. Der Lyapunov-Exponent weist ab $r > 8$ eine Peridodizität von π auf weshalb weitere Betrachtungen für $r > 8$ nicht zielführend erscheinen.

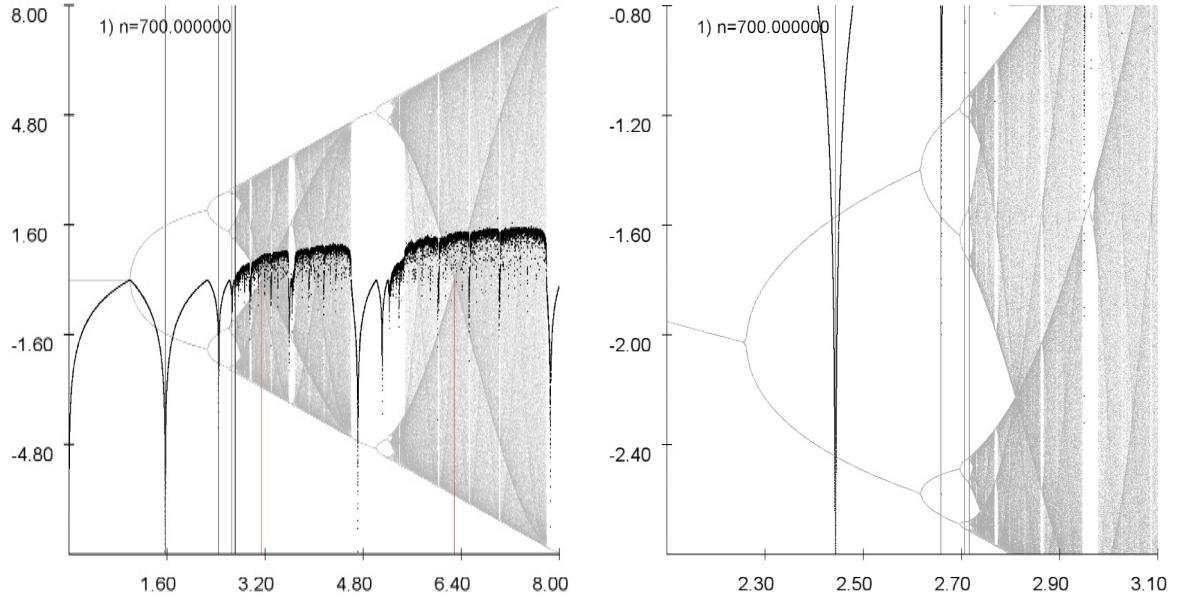


Abbildung 12: Links: Analyse des Bifurkationsdiagrammes der sinus Abbildung. Eingezeichnet sind die ersten 6 Superattraktiven Stellen (Erste Superattraktive Stelle bei $r = 0$). Ebenfalls wurden π und 2π eingezeichnet. Rechts: Vergrößerung des Ausschnittes $r \in [2.1, 3.1], y \in [-2.8, -0.8]$

3 Duffing-Gleichung

Wir betrachten nun einen angetriebenen und gedämpften Oszillatoren. Als Unterschied zum gewöhnlichen Harmonischen Oszillatoren wird der Term mit der Federkonstante kubisch. Dies ist vergleichbar mit der Schwingung eines Metallstabes.

$$\ddot{x} + \lambda \dot{x} + \beta x^3 = \epsilon \cos \Omega t \quad (35)$$

Diese DGL lässt sich nicht analytisch lösen. Im Folgenden lösen wir die Gleichung sowohl mit der Euler-Methode [4] als auch mit dem Runge-Kutta Verfahren [5]. Um dies zu ermöglichen lässt sich (35) in folgendes DGL System umformen.

$$\frac{dy}{dt} = \epsilon \cos \theta - \lambda y - \beta x^3 \quad (36)$$

$$\frac{dx}{dt} = y \quad (37)$$

$$\frac{d\theta}{dt} = \Omega \quad (38)$$

3.1 Phasenraumdiagramm

Im folgenden verwenden wir als Parameter $\epsilon = 0, 2$, $\lambda = 0, 08$, $\beta = 1$ und $\Omega = 1$ und untersuchen die Unterschiede der verwendeten numerischen Verfahren bei gleichen Anfangswerten. Dazu wählen wir $x_0 = 0.21$ und $y_0 = 0.02$ und betrachten das Phasenraumdiagramm bei unterschiedlichen Schrittgrößen h . Für kleine Parameter h erhalten wir sowohl bei dem Euler-Cauchy, als auch beim Runge-Kutta Verfahren (Runge-Kutta 4ter Ordnung 3-Dimensional, siehe Appendix) Attraktoren

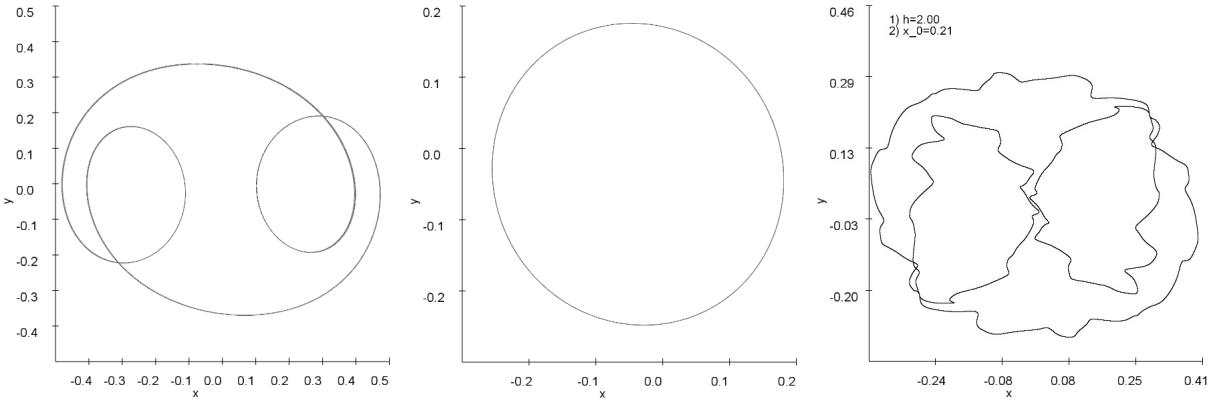


Abbildung 13: Phasenraumdiagramm. Links: Schrittweite $h = 0.1$ mit Euler-Cauchy. Mitte: Schrittweite $h = 0.08$ mit Euler-Cauchy identisch bei $h = 0.2$ mit Runge-Kutta. Rechts: Runge-Kutta bei Schrittweite $h = 2$

wie in Abbildung 13 (Mitte) dargestellt. Für $0.2 < h < 0.08$ erhalten wir mit dem Euler-Cauchy Verfahren einen anderen Attraktor (Abbildung 13 links) und für $h > 0.2$ divergiert die Trajektorie. Bei dem Runge-Kutta-Verfahren können wir $h \approx 0.2$ wählen und erhalten weiterhin einen Attraktor wie in Abbildung 13 (Mitte). Wählen wir noch größere Schrittweiten so beobachten wir plötzlich bei $h \approx 2$ einen Attraktor wie in Abbildung 13 rechts. Es ist offensichtlich, dass eine Schrittweite von $h = 2$ keinen Sinn macht, da x und y nie größer als 0.5 werden. Dennoch ist es erstaunlich, dass es tatsächlich einen Grenzzyklus gibt, der gewisse Ähnlichkeit zu Abbildung 13 links hat. Die weitere Untersuchung dieses Phänomens soll hier nicht fortgesetzt werden.

Im folgenden sind alle Berechnungen in diesem Kapitel mit dem Runge-Kutta-Verfahren 4ter Ordnung implementiert. Ganz bedeutend ändert sich der Attraktor bei Variation der Anfangsbedingungen. Variieren wir lediglich x_0 und lassen $y_0 = 0.02$, bei einer Schrittweite von $h = 0.01$, so erhalten wir bei $x_0 = 0.18$ einen Attraktor wie in Abbildung 13 (Mitte) während wir bei $x_0 = 0.17$ ein Phasenraumdiagramm ähnlich zu Abbildung 13 (links) erhalten.

Abbildung 14 zeigt Attraktoren bei unterschiedlichen Anfangsbedingungen. Im Fall des Duffing-

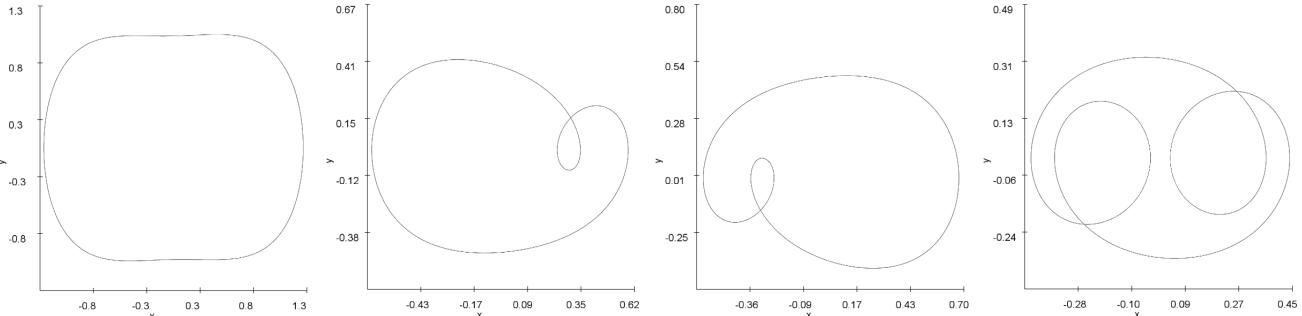


Abbildung 14: Phasenraumdiagramme mit Runge-Kutta bei einer Schrittweite von $h = 0.01$. Von links nach rechts: 1. $x_0 = 1.05$, $y_0 = 0.77$. 2. $x_0 = -0.67$, $y_0 = 0.02$. 3. $x_0 = -0.46$, $y_0 = 0.30$. 4. $x_0 = -0.43$, $y_0 = 0.12$

Oszillators beobachten wir folglich Chaos im Sinne, dass die Attraktoren sich deutlich unterscheiden wenn die Anfangsbedingungen nur marginal verändert werden. Weiterhin erhalten wir einen Übergang von Ordnung zu Chaos wenn wir z.B. die Anregungsamplitude ϵ ändern. Ganz analog zum Parameter r bei der logistischen Abbildung.

3.2 Poincareschnitt

Im folgenden variieren wir ϵ , also die Amplitude der Anregung. Ab einem bestimmten ϵ erhalten wir keinen eindeutigen Attraktor mehr, egal wie viele Zeitschritte wir berechnen. Dennoch stellt sich heraus, dass trotzdem nie der gesamte Phasenraum ausgefüllt wird. Solche Attraktoren werden seltsame Attraktoren genannt. Im folgenden untersuchen wir diese seltsamen Attraktoren und deren Poincaré-Schnitt. Die gezeigten Phasenraumpورtraits sind Projektionen des dreidimensionalen Phasenraums (x, y, θ) auf die (x, y) Ebene. Der Poincaré-Schnitt ist eine Abbildung aller (x, y) welche eine bestimmte Ebene im Phasenraum schneiden. In Abbildung 15 (rechts) sieht man einen Poincaré-Schnitt des Duffing-Oszillators bei $\theta = 0$ und das entsprechende Phasenraumdiagramm (links).

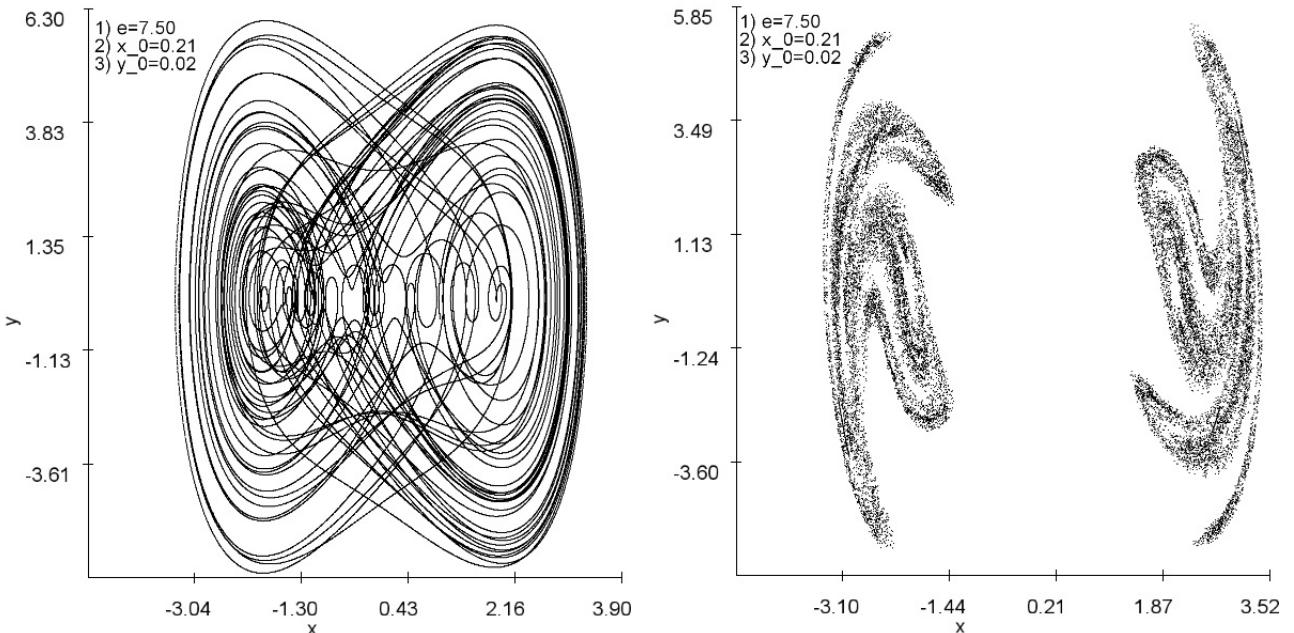


Abbildung 15: Phasenraumdiagramm und Poincaré-Schnitt des Duffing Oszillators für $\epsilon = 7.5$, $x = 0.21, y = 0.02$, $\lambda = 0.04$, $\beta = 1$, $\theta = 1$.

4 LDR-Oszillator

Im folgenden untersuchen wir einen realen nichtlinearen Schwingkreis. Dieser wurde realisiert indem bei einem LCR-Schwingkreis der Kondensator durch eine Diode ausgetauscht wurde (Abbildung 16)

4.1 Theorie

Zunächst lässt sich ein linearer Schwingkreis der aus einem Widerstand, einem Kondensator und einer Spule besteht über die Spannungen an den einzelnen Bauteilen beschreiben

$$V_g = V_c + V_l + V_r \text{ (Kirschoff'sches Gesetz)}$$

Daraus folgt mit $V_g = V_s \cos \omega t$ die Differentialgleichung

$$\ddot{Q}L + \dot{Q}R + \frac{Q}{C} = V_s \cos \omega t$$

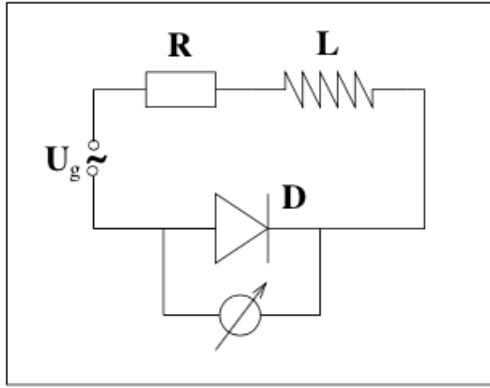


Abbildung 16: Schaltkizze für nichtlinearen Schwingkreis. Quelle: Versuchsanleitung, S. 43, Uni Hamburg

welche mit dem angetriebenen, gedämpften Oszillatoren vergleichbar ist. Dabei ist Q die Ladung, L die Induktivität der Spule, R der Widerstand, C die Kapazität des Kondensators V_s , die angelegte Spannung und ω die Frequenz der angelegten Wechselspannung.

Wird nun der Kondensator durch eine Diode ausgetauscht erhalten wir, wie bei dem Duffing-Oszillatoren eine Gleichung die nicht mehr analytisch lösbar ist. Während beim linearen Schwingkreis $I = \frac{dQ}{dt}$ gilt lässt sich die Diode als parallel geschalteter Kondensator und Widerstand mit

$$I = I_f(1 - \exp(-\frac{V_d}{V_t})) + \frac{dQ}{dt} \quad (39)$$

beschreiben, wobei I_f und V_t Konstanten sind. Dabei beschreibt der erste Summand den Strom durch den Widerstand und der zweite die Änderung der Ladung aufgrund der Diodenkapazität. Dies führt bei einer Kapazität in Durchlassrichtung von $C = C_f \exp(-\frac{V_d}{V_t})$ zu

$$\frac{dV_d}{dt} = \frac{I - I_f(1 - \exp(-\frac{V_d}{V_t}))}{C_f \exp(-\frac{V_d}{V_t})} \quad (40)$$

und bei einer Kapazität in Sperrrichtung von $C = C_r(1 + \frac{V_d}{\phi})^{-\gamma}$ zu

$$\frac{dV_d}{dt} = \frac{I - I_f(1 - \exp(-\frac{V_d}{V_t}))}{C_r(1 + \frac{V_d}{\phi})^{\gamma}} \quad (41)$$

wobei ϕ und γ Konstanten sind. Außerdem gilt

$$\frac{dI}{dt} = \frac{V_s \cos \theta - V_d - RI}{L} \quad (42)$$

$$\frac{d\theta}{dt} = \omega \quad (43)$$

Diese Gleichungen lassen sich numerisch lösen.

4.2 Numerische Berechnungen

Unter Verwendung der Parameter:

R	100Ω	Widerstand
L	$2367 \cdot 10^{-6} H$	Spule
C_r	$82 pF$	Konstante proportional zur Kapazität der Diode in Sperrrichtung
C_f	$56 \cdot 10^{-6} pF$	Konstante proportional zur Kapazität der Diode in Durchlassrichtung
I_f	$2,8 pA$	Konstante proportional zum Strom
γ	$0,44$	Konstante
ϕ	$0,6V$	Konstante
V_t	$34 mV$	Konstante

haben wir für unterschiedliche Anregungsspannungen V_s numerisch die Gleichungen aus 4.1 gelöst. Dabei sind wir davon ausgegangen, dass für $V_d > -0.6V$ die Diode sperrt. Um sinnvolle Lösungen zu erhalten muss für die Schrittweite $h \geq 10^{-8}$ gelten. Diese oszillierenden Lösungen für den Strom und die Spannung über der Diode sind in Abbildung 17 dargestellt. Strom und Spannung oszillieren mit gleicher Phase und pendeln sich nach gewisser Zeit unabhängig von den Anfangsbedingungen ein.

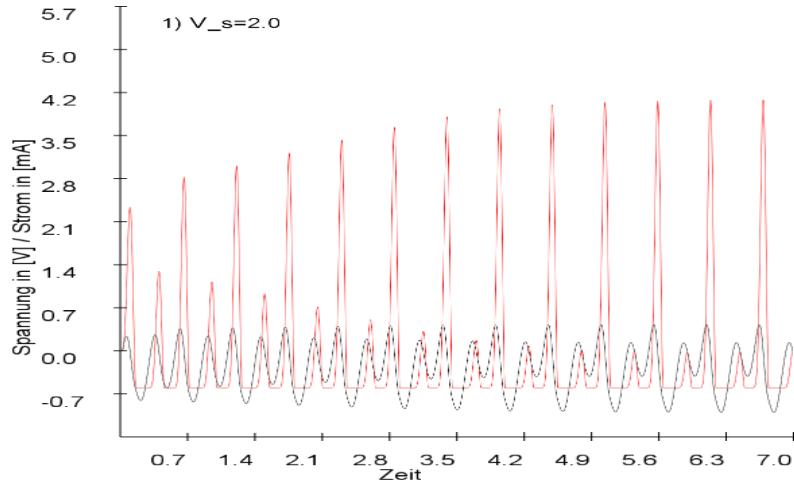


Abbildung 17: Oszillation von Strom (schwarz) und Spannung (rot) bei $V_s = 2V$ mit Runge-Kutta-Verfahren 4ter Ordnung

4.2.1 Phasenraumdiagramme

Bei einer Anregungsspannung im Grenzfall $V_s < 0.001V$ erhalten wir die Lösungen des angetriebenen gedämpften harmonischen Oszillators, denn bei $V_s \approx 0$ verschwindet der nicht-linearen Term in den Gleichungen.

Erhöhen wir die Spannung $V_s > 1V$, so finden wir weitere Attraktoren (Abbildung 18). Dabei ist bei $V_s = 1V$ (links) die Trajektorie wohl definiert, während bei $V_s = 1.2V$ (Mitte) sie sich langsam aufspaltet und bei $V_s = 1.42V$ (rechts) auf 2 Pfaden durch den Phasenraum verläuft. Je höher V_s

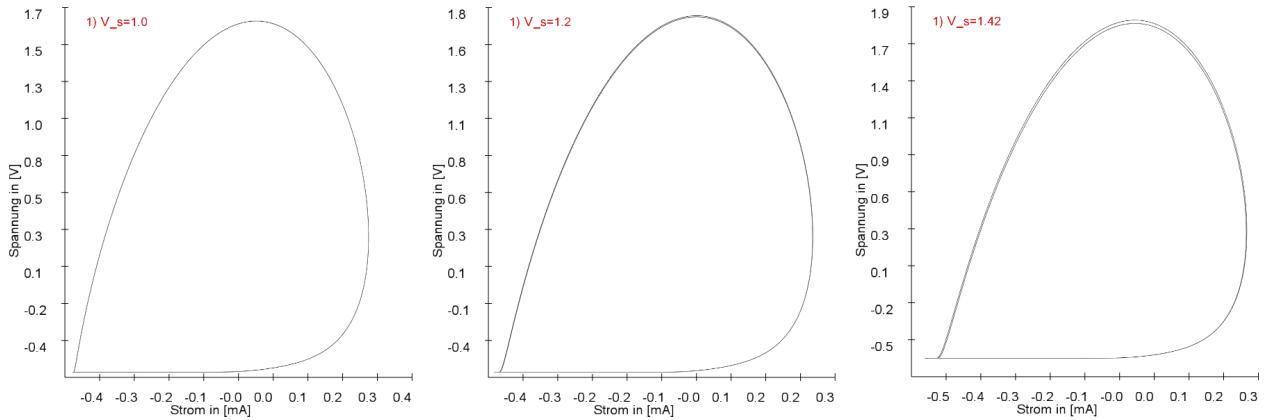


Abbildung 18: Phasenraumdiagramme mit Runge-Kutta 4ter Ordnung für $2 \cdot 10^4$ Zeitschritte nach einer Einschwingzeit von $3 \cdot 10^5$ Zeitschritten und einer Schrittweite von $h = 10^{-9}$. Links: $V_s = 1V$, Mitte: $V_s = 1.2V$. Rechts: $V_s = 1.42V$.

wird, desto stärker gehen die Trajektorien auseinander bis sie bei ca. $V_s = 20V$ ein Maximum erreichen (Abbildung 19 links) und anschließend wieder zusammen gehen und chaotisches Verhalten zeigen (Abbildung 19 Mitte und rechts).

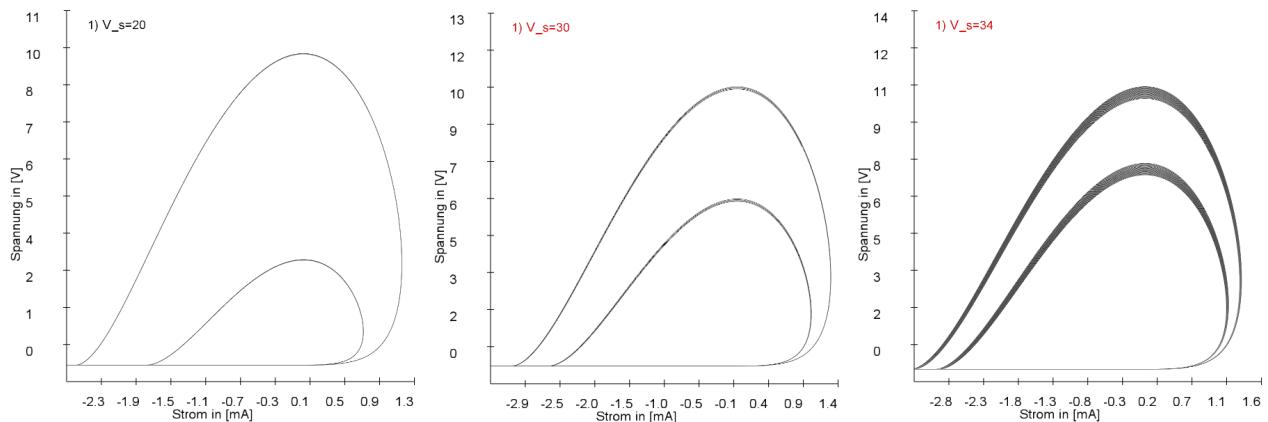


Abbildung 19: Phasenraumdiagramme mit Runge-Kutta 4ter Ordnung nach einer Einschwingzeit von $3 \cdot 10^5$ Zeitschritten und einer Schrittweite von $h = 10^{-9}$. Links: $V_s = 20V$ für $2 \cdot 10^4$ Zeitschritte, Mitte: $V_s = 30V$ für $2 \cdot 10^4$ Zeitschritte. Rechts: $V_s = 34V$ für $1.2 \cdot 10^5$ Zeitschritte

Nach Änderung des Zeitschrittes h auf $h = 10^{-8}$ haben wir sehr unterschiedliche Beobachtungen gemacht. Beispielsweise erhalten wir bei $V_s = 9.1300024986267072V$ einen eindeutigen Attraktor (Abbildung 20 links) und Chaos bei einem Wert $10^{-15}V$ größer (rechts). Unter Verwendung des Euler-Cauchy-Verfahrens beobachten wir teilweise ein anderes Verhalten. So können wir im Bereich von $V_s = 3V$ bis $V_s = 4.7V$ mehrere Periodenverdopplungen beobachten (Abbildung 21) bis hin zum Chaos.

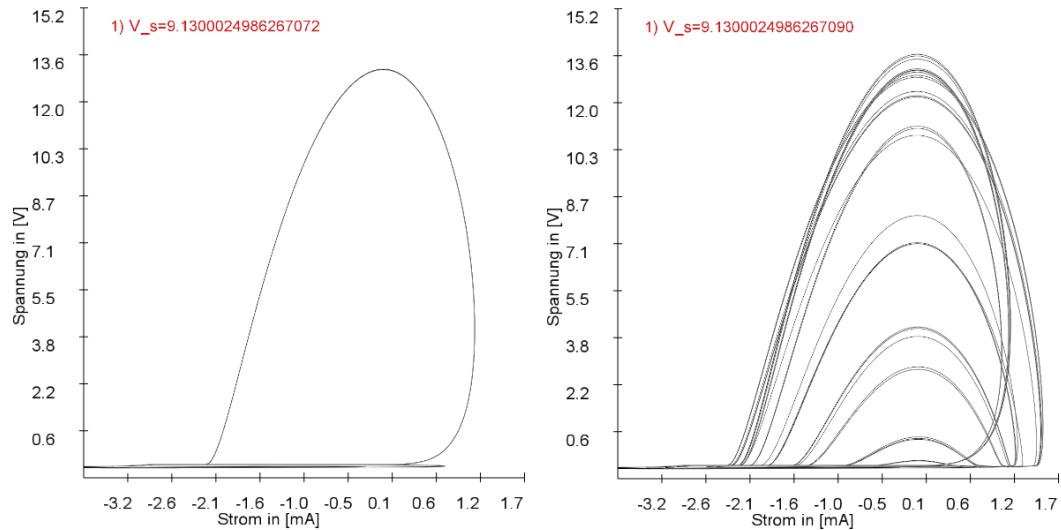


Abbildung 20: Phasenraumdiagramme mit Runge-Kutta 4ter Ordnung nach einer Einschwingzeit von $3 \cdot 10^5$ Zeitschritten für weitere $5 \cdot 10^4$ Zeitschritte bei einer Schrittweite von $h = 10^{-8}$. Links: Wohldefinierter Attraktor. Rechts: Chaos

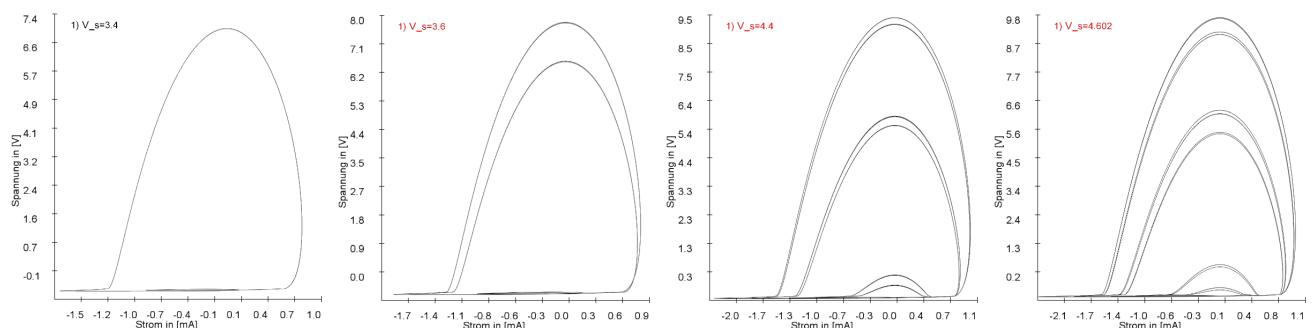


Abbildung 21: Phasenraumdiagramme mit Euler-Cauchy-Verfahren nach Einschwingzeit von $2.5 \cdot 10^6$ Zeitschritten für weitere 10^5 Zeitschritte. Von links nach rechts: 1. $V_s = 3.4V$, 2. $V_s = 3.6V$, 3. $V_s = 4.4V$, 4. $V_s = 4.602V$

4.2.2 Poincaré-Schnitt

Im chaotischen Fall können wir wie beim Duffing-Oszillator seltsame Attraktoren finden (vgl. Abbildung 22). Dabei fällt auf, dass die Trajektorie nur bis $V_d \approx 1.4V$ die Ebene bei $\theta = 0$ schneiden. Daraus schließen wir, dass die Trajektorie in einer gewissen Periodizität zwischen kleinen V_d und großen V_d oszilliert.

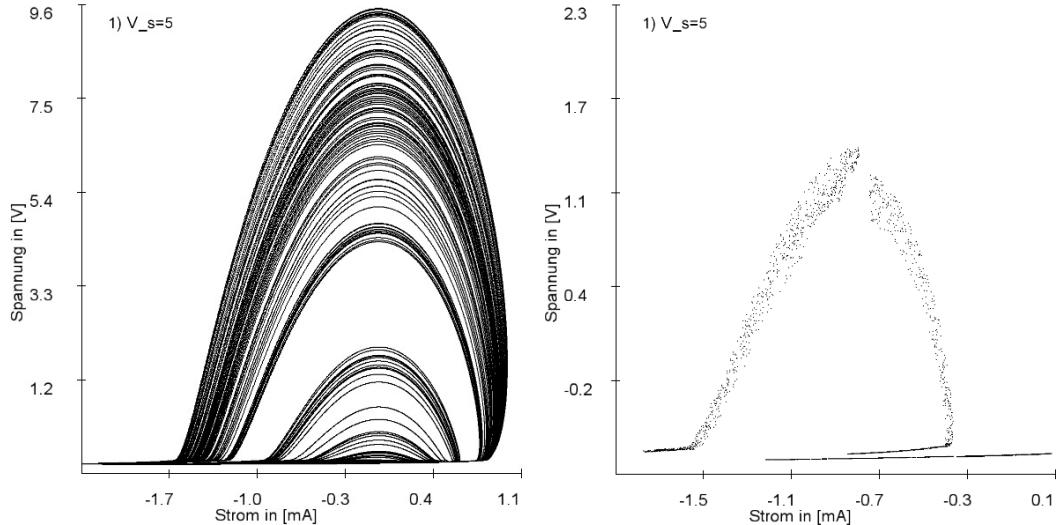


Abbildung 22: $5 \cdot 10^5$ Zeitschritte nach einer Einschwingzeit von $\cdot 10^5$ Zeitschritten bei $V_s = 5V$. Links: Phasenraumdiagramm. Rechts: Poincaré-Schnitt durch $\theta = 0$ Ebene

4.2.3 Bifurkationsdiagramm-Diagramm

Um den Bereich von Chaos und Ordnung zu visualisieren, haben wir im folgenden separat für Strom und Spannung das Bifurkationsdiagramm geplottet. Dabei ist der variierende Parameter die Eingangsspannung V_s . Bei $V_s \approx 1.4$ beobachten wir sowohl beim Strom als auch bei der Spannung

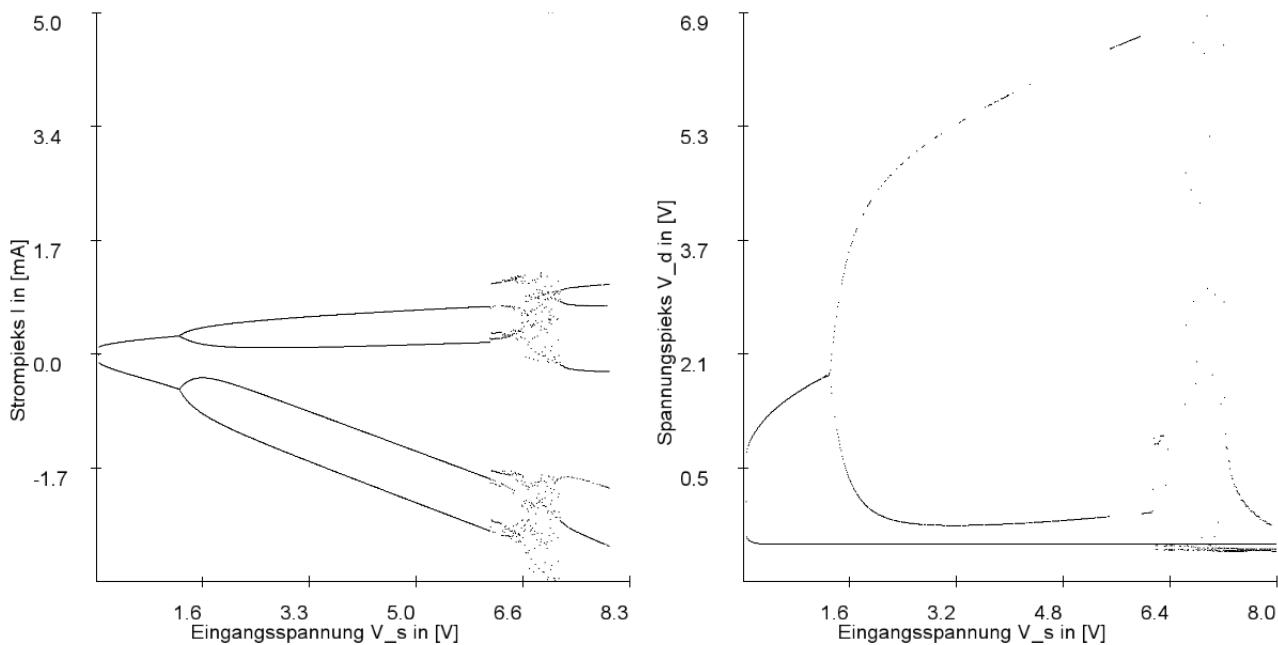


Abbildung 23: Bifurkationsdiagramm mit $4 \cdot 10^5$ Iterationen pro V_s . Von $V_s = 0$ bis $V_s = 8$ in 0.01 Schritten. Links: Strom. Rechts: Spannung

eine Periodenverdopplung. Dies passt mit unserer Beobachtung des Attraktors bei $V_s = 1.42$ (Abbildung 18 rechts) zusammen. Auch das Chaotische Verhalten bei $V_s = 7$ (Abbildung 28 links) finden wir im Bifurkationsdiagramm wieder.

4.3 Versuchsdurchführung

4.3.1 Versuchsaufbau

Für die Messung an einem realen LDR-Schwingkreis haben wir zunächst einen Messaufbau wie in Abbildung 24 (links) verwendet. Mit den Eingängen A und B des Oszilloskop konnten wir Strom und Spannung über der Diode einzeln und als Phasenraumdiagramm darstellen. Ferner haben wir einen Gate-Delay Generator verwendet der Abbildung 24 (rechts) zusehen ist. Die Parameter

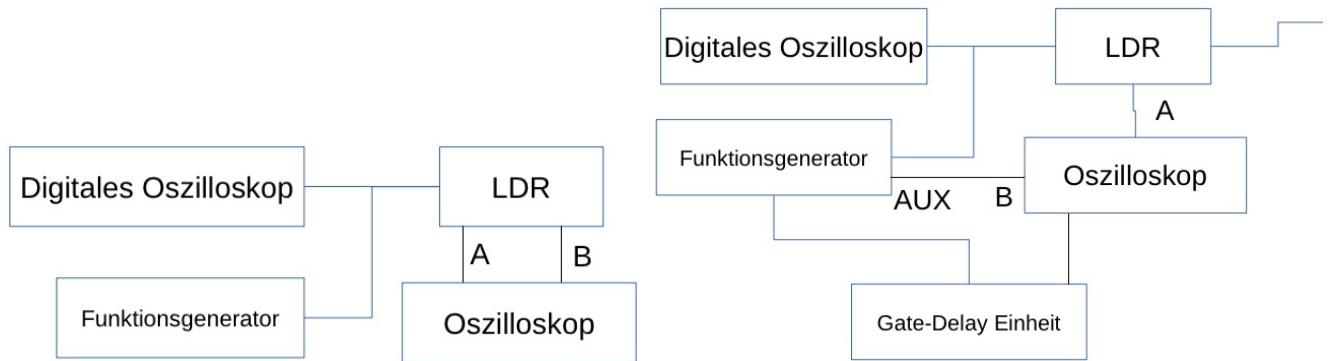


Abbildung 24: Messaufbau zur Visualisierung des LDR-Schwingkreises. Oszilloskop links zur Überprüfung der Eingangsspannung und Frequenz. Links: Strom und Spannung über der Diode werden dargestellt auf Oszilloskop über Eingänge A und B. Rechts: Gate-Delay Einheit verzögert Signal in Abhängigkeit der jetzt varierenden Eingangsspannung V_s zur Darstellung des Stroms bzw. der Spannung in Abhängigkeit von V_s

des Schwingkreises unterscheiden sich von den Parametern, die bei der numerischen Berechnung verwendet wurden. So hatten wir eine gesamt Induktivität von $L \approx 370 \cdot 10^{-6} H$ was bei einer Resonanzfrequenz von $w \approx 800 kHz$ zu $C_r \approx 4.2 \cdot 10^{-9} F$ führt.

4.3.2 Oszillation und Phasenraumdiagramm

Zunächst haben wir bei einer Eingangsspannung von $V_s = 0.128V$ die Resonanzfrequenz über die Betrachtung des Phasenraumdiagrammes auf $\omega = 800kHz$ bestimmt. Der aufgebaute Schaltkreis hatte nicht die selbe Konfiguration wie unsere numerischen Modelle, dennoch erhalten wir einen sehr ähnlichen Verlauf. In Abbildung 25 sind die Oszillationen von Strom und Spannung (links) und das entsprechende Phasenraumdiagramm dargestellt (rechts) welches dem numerischen Ergebnis in 21 (links) sehr ähnlich sieht.

4.4 Bifurkationsdiagramm

Beim erzeugen des Bifurkationsdiagrammes haben wir einen anderen Schaltkreis verwendet, dessen Resonanzfrequenz wir auf $\omega = 325kHz$ bestimmt haben. Der Versuchsaufbau wurde nun mit einer Gate-Delay Einheit aufgebaut (Abbildung 24 rechts). Wir modellierten über den Aux-Out des Funktionsgenerators eine Sägezahnspannung welche später den Definitionsbereich des Parameters V_s darstellt. Zunächst ist dies nicht gelungen. Mit Hilfe des digitalen Oszilloskopes konnten wir dann aber den Verlauf so modellieren, dass $V_s = 0V$ linear auf $V_s = V_{Ausgang}$ ansteigt

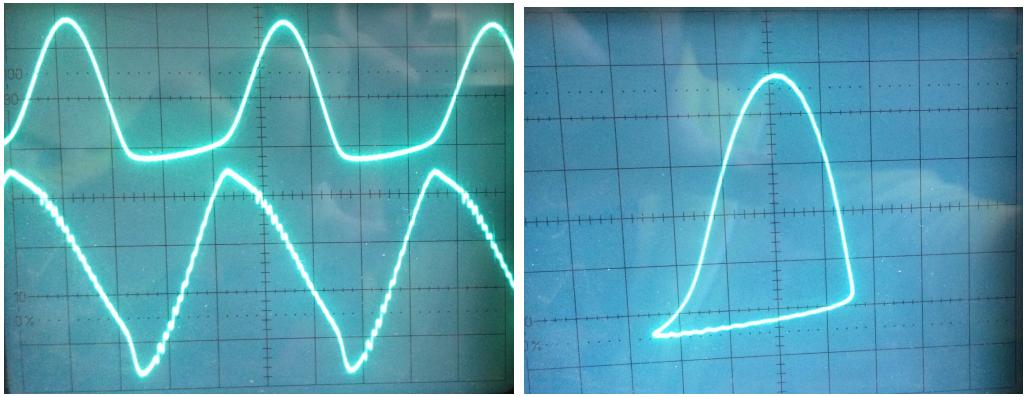


Abbildung 25: Eingangsspannung $V_s = 5V$ und Resonanzfrequenz von $\omega = 800kHz$. Links: Zeitlicher Verlauf von Spannung (oben) und Strom (unten). Rechts: Phasenraumdiagramm

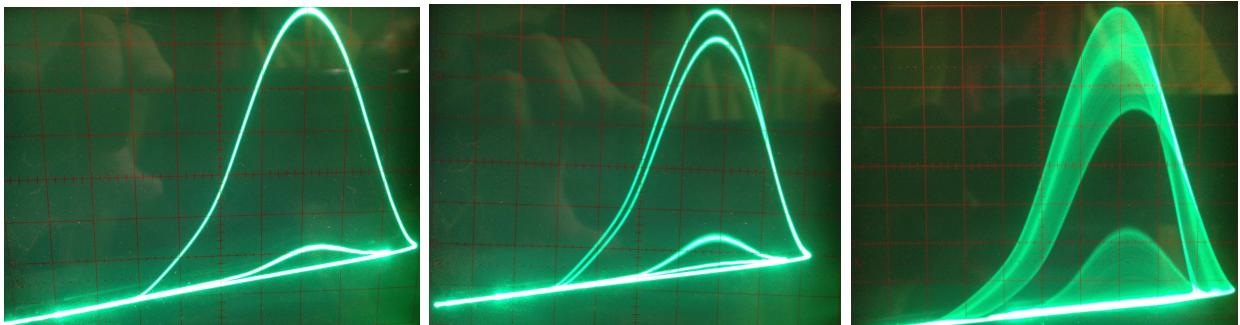


Abbildung 26: Phasenraumdiagramm bei einer Resonanzfrequenz von $\omega = 800kHz$. Links: Eingangsspannung $V_s = 134V$. Mitte: Eingangsspannung $V_s = 112V$. Links: Eingangsspannung $V_s = 29.6V$

und periodisch wiederholt wird. Abbildung 27 verdeutlicht die Auswirkung auf das Bifurkationsdiagramm durch unterschiedliche Spannungsverläufe. Wir stellten fest, dass das Bifurkationsdiagramm am besten am Oszilloskop zu sehen war, wenn wir den Definitionsbereich von oben nach unten ($V_s = V_{Ausgang}$ nach $V_s = 0V$) eingestellt haben (Abbildung 27 rechts). In dieser Einstellung gab es am wenigsten Artefakte aufgrund der Ungenauigkeit des Funktionsgenerators. Wir erzeugten das Bifurkationsdiagramm zusätzlich mit einer Dreiecksspannung (Abbildung 27 Mitte). Wie zu erwarten ergab sich so eine Überlagerung aus den beiden Sägezahnspannungen.

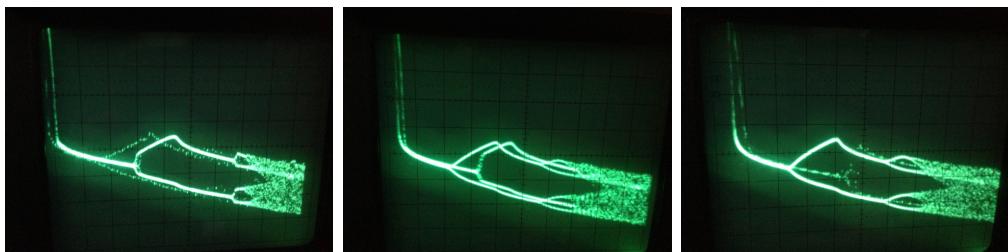


Abbildung 27: Bifurkationsdiagramme mit verschiedenen Modulationen des Parameters V_s : Links Sägezahnspannung V_s läuft von $V_s = 0V$ bis $V_s = V_{Ausgang}$. Mitte: Dreiecksspannung, rechts: Sägezahnspannung V_s läuft von $V_s = V_{Ausgang}$ bis $V_s = 0V$

Wir fragten uns, wieso es zu einer Spitze im oberen Ast (Abbildung 28 links) nach dem ersten Bifurkationspunkt kommt. Dabei zeigte sich, dass die Frequenz nicht genau auf $\omega = 325kHz$ eingestellt war. Als wir dies korrigierten erhielten wir ein glatteres Bifurkationsdiagramm (Abbildung 28 rechts)

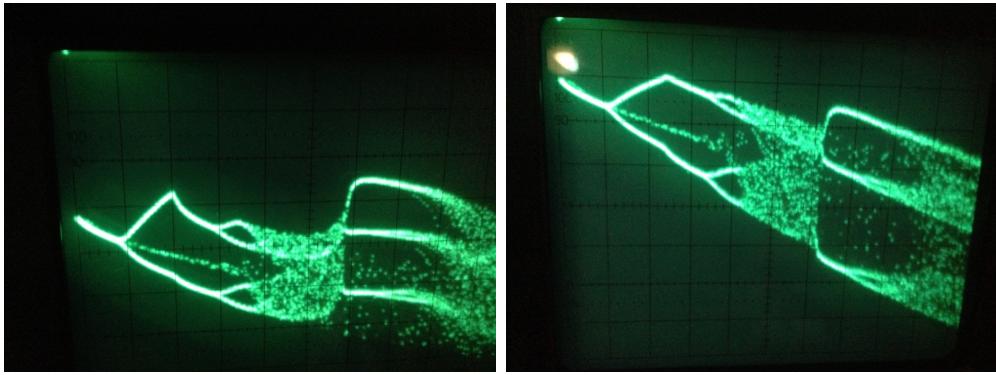


Abbildung 28: Bifurkationsdiagramm mit variierendem Parameter V_s . Links: Verlauf der Spannung V_d . Rechts: Verlauf des Stroms I

4.5 Vermessung der Bifurkationspunkte

Zuletzt haben wir noch die Bifurkationspunkte vermessen. Dazu haben wir die Amplitude der Sägezahnspannung minimiert, so dass $V_s = V_{\text{Ausgang}}$ konstant war. Bei geringer Spannung wurde auf dem Oszilloskop nur ein Punkt abgebildet. Beim durchlaufen der Spannung entstanden so sprungartig zwei Punkte und schließlich 4 Punkte. Die jeweiligen $V_{\text{Bifurkation},i}$ konnten wir somit als Bifurkationspunkte identifizieren:

$$V_{\text{Bifurkation},1} = 6.6V \pm 0.2V$$

$$V_{\text{Bifurkation},2} = 13.6V \pm 0.2V$$

In Abbildung 29 haben wir noch ein mal mit der Sägezahnspannung die Bifurkationspunkte vergrößert aufgenommen. Es gelang uns nicht den Bifurkationspunkt $V_{\text{Bifurkation},3}$ zu bestimmen weshalb wir die Feigenbaumkonstante nicht aus dem Experiment ableiten konnten.

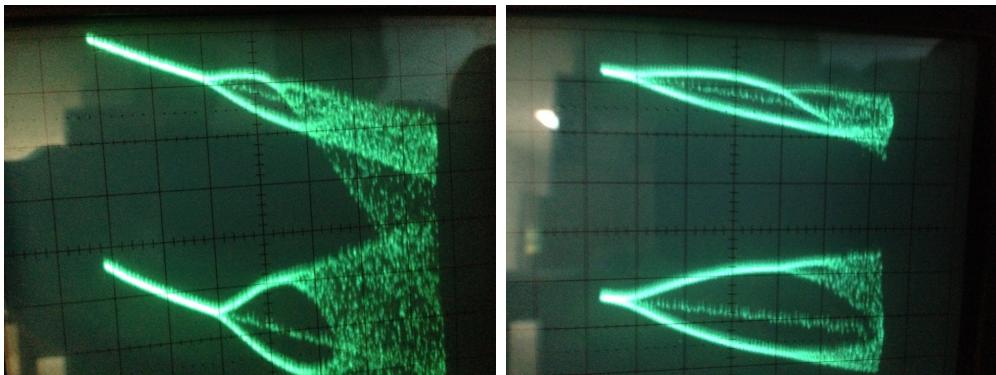


Abbildung 29: Bifurkationsdiagramm des Stroms I . Vergrößerung kurz vor dem chaotischen Bereich.

4.6 Auswertung

Bereits bei dem Verlauf der Spannung V_d (vgl. Abbildung 25 links, oberer Verlauf) erkennt man die Ähnlichkeit zu den numerischen Berechnungen (Abbildung 17 rot). Gleiches gilt für die Phasenraumdiagramme bei denen auf gleiche Art und Weise Periodenverdopplung zu beobachten ist. Auch der berechnete seltsame Attraktor (Abbildung 22 links) hat die selbe Form wie der Attraktor in Abbildung 26 (rechts).

Die gemessenen Bifurkationsdiagrammen ähneln den berechneten Bifurkationsdiagrammen. Unterschiede sind einerseits damit begründet, dass bei den Berechnungen nicht die exakt gleichen

Parameter wie bei der Versuchsdurchführung verwendet wurden und andererseits, dass die Berechnungen nur auf einem vereinfachten Modell basieren. So wurde die Diode durch einen Kondensator und einen Widerstand, die Parallel geschaltet wurden, modelliert. Auch wird ein möglicher Durchbruchbereich in Sperrrichtung nicht berücksichtigt. Ebenfalls ignoriert werden Temperaturabhängigkeiten. Trotz diesem modellartigen Ansatz finden wir es erstaunlich wie gut es möglich ist dieses System zu beschreiben und wie groß die Ähnlichkeit zum experimentellen Versuch ist.

5 Zusammenfassung

In dem Versuch haben wir einen Einblick in die Welt des deterministischen Chaos erhalten. Anfangen mit der Dreiecksgleichung und der logistischen Abbildung konnten wir bereits mit sehr einfachen Gleichungen Charakteristiken von Chaos beschreiben. Bei komplexeren Problemen wie dem Duffing-Oszillatator wurde der Aspekt der Anfangsbedingungen deutlich, welche eine Form des Chaos beschreibt. Bei der Untersuchung des LDR-Schwingkreis war es dann möglich das Chaos in einem realen System zu beobachten. Dabei waren zwar deutliche Parallelen zu den numerischen Berechnungen ersichtlich aber es wurde auch klar, dass es sich dabei nur um Modelle handelt und zusätzliche Effekte nicht berücksichtigt werden.

5.1 Software

Wir haben für alle Grafiken einen eigenen Plotter genutzt. Diesen haben wir im Ramen des Versuches entwickelt. Dabei zeigten sich viele Aspekte welche wir in der nächsten Version des Plotters verbessern müssen.

Für das generieren des Plottes haben wir verschiedene Wege ausprobiert:

1. OpenGL Shader. Die VBO's werden in Python initialisiert (x,y). Der Vertex- oder Fragments- shader bildet diese Punkte dann direkt ab. Mittels einer Translations- und Skalierungsmaatrix können beim bewegen durch die Plotplane die (x,y) so transformiert werden, dass beim zoomen und beim translatieren im Plot stets gleich viele Punkte abgebildet werden.
2. OpenCL Kernel. Die VBO's werden via OpenCL mit Daten gefüllt.
3. Direktes eingeben von Plotdaten, (z.B. Textfile, Fortran, Python).

Wir waren von dem Potential der ersten Variante begeistert. Beispielsweise konnte mit einem Fragmentshader Plot sehr einfach eine Abbildung im komplexen Raum farbig visualisiert werden.

Es zeigte sich, dass die MacBooks mit denen wir arbeiten (MacBook Pro (Retina, 13-inch, Late 2013)), keine double-precision GPUs verbaut haben. Deshalb waren alle GPU Berechnungen limitiert in der Genauigkeit (Abbildung 30). Mit einigen Tricks (Renormierung) konnten wir die Ergebnisse zwar verbessern, aber beispielsweise lässt sich die Feigenbaumkonstante trotzdem nur ungenau auf unserer GPU bestimmen.

Nach dem Versuch wurde das Github repository des Versuches nicht mehr weiterentwickelt. Ein neues Repository wo das Feedback welches wir gewinnen konnte einfießt wurde erstellt [3].

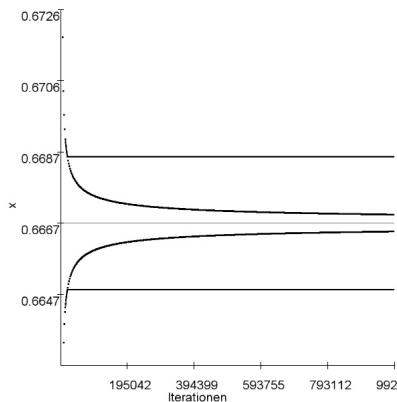


Abbildung 30: Beispiel für schlechte Konvergenz auf der GPU verglichen mit der CPU Berechnung.

6 Appendix

6.1 Runge-Kutta 4ter Ordnung in 2 Dimensionen

Hängt f von mehr als nur x und t ab, so lassen sich weitere Parameter ebenfalls mit der selben Idee variieren. Gegeben seien folgende Differentialgleichungen:

$$\frac{dx}{dt} = f(x, y, t)$$

$$\frac{dy}{dt} = g(x, y, t)$$

Dann definieren wir im Sinne des Runge-Kutta Verfahrens:

$$k_1 = hf(x_n, y_n, t)$$

$$l_1 = hg(x_n, y_n, t)$$

$$k_2 = hf\left(x_n + h\frac{k_1}{2}, y_n + h\frac{l_1}{2}, t + \frac{h}{2}\right)$$

$$l_2 = hg\left(x_n + h\frac{k_1}{2}, y_n + h\frac{l_1}{2}, t + \frac{h}{2}\right)$$

$$k_3 = hf\left(x_n + h\frac{k_2}{2}, y_n + h\frac{l_2}{2}, t + \frac{h}{2}\right)$$

$$l_3 = hg\left(x_n + h\frac{k_2}{2}, y_n + h\frac{l_2}{2}, t + \frac{h}{2}\right)$$

$$k_4 = hf(x_n + hk_3, y_n + hl_3, t + h)$$

$$l_4 = hg(x_n + hk_3, y_n + hl_3, t + h)$$

Und schließlich:

$$y_{n+1} = y_n + \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}$$

$$x_{n+1} = x_n + \frac{l_1 + 2l_2 + 2l_3 + l_4}{6}$$

6.2 Algorithmus zum bestimmen des Bifurkationspunktes

Sourcecode: prak/konvergenz.py

```
import numpy

N_POINTS = 10000
N_ITER = 15
X0      = 0.8
A       = 0.5
f       = lambda r, x: r*x*(1-x)
EPS     = 0.00000001
DEPTH   = 8
R_MIN   = 0.0
R_MAX   = 4.0

data_f   = numpy.zeros(N_POINTS)
data_df  = numpy.zeros(N_POINTS)
tmp_data = numpy.zeros(N_ITER)
tmp_bifurk = numpy.zeros(DEPTH)

r_min = R_MIN
r_max = R_MAX
r_len = r_max-r_min
eps = EPS

for j in range(0, DEPTH):
    r = r_min
    eps *= -1
    dr = r_len/N_POINTS
    for k in range(0, N_POINTS):
        tmp_data[0] = X0
        for i in range(1, N_ITER):
            tmp_data[i] = A*(f(r, tmp_data[i-1])+tmp_data[i-1])
        data_f[k] = tmp_data[N_ITER-1]
        data_df[k] = numpy.abs((f(r, data_f[k])-f(r, data_f[k]+eps))/eps)
        if k > 0 and (data_df[k] > 1 and data_df[k-1] < 1 or data_df[k-1] >
                       1 and data_df[k] < 1):
            tmp_bifurk[j] = r+eps/2
            r_len /= 10
            r_min = max(r_min, tmp_bifurk[j]-r_len/2)
            r_max = min(r_max, tmp_bifurk[j]+r_len/2)
            break
        r += dr

bifurk = (tmp_bifurk[DEPTH-1]+tmp_bifurk[DEPTH-2])/2
print(bifurk) # 2.9999999322
```

6.3 Lyapunov renormiert - OpenGL GLSL Quellcode

```
float g(float r, float x) {
    return r * x * (1-x);
}
vec4 f(vec4 x) {
    float x0 = 0.4;
    float eps = 0.0001;
    float n = 10000;
    float summe = x0;
    for (int i=1; i < n; i++) {
        x0 = g(x.x, x0);
        summe += log(abs(g(x.x, x0+eps)-g(x.x, x0))/eps);
    }
    return vec4(x.x, summe/n, 0, 0.5);
}
```

Literatur

- [1] Physikalisches Praktikum für Fortgeschrittene Universität Hamburg. *Nichtlineare Dynamik und Chaos*. URL: <http://www3.physnet.uni-hamburg.de/ex/html/fprakt/dwn/chaos/chaos1.pdf>.
- [2] OESIS. *A006890 Decimal expansion of Feigenbaum bifurcation velocity*. @ONLINE. URL: <https://oeis.org/A006890>.
- [3] Weiterführende GitHub repository des Plotters. *opengl plot prototype*. URL: https://github.com/keksnico/opengl_plot_prototype.git.
- [4] Explizites Euler-Verfahren Wikipedia. *Explizites Euler-Verfahren* @ONLINE. URL: https://de.wikipedia.org/wiki/Explizites_Euler-Verfahren.
- [5] Runge-Kutta-Verfahren Wikipedia. *Runge-Kutta-Verfahren* @ONLINE. URL: <https://de.wikipedia.org/wiki/Runge-Kutta-Verfahren>.