Institut für Theoretische Physik
der Universität zu Köln

Prof. Dr. Simon Trebst
Michael Becker, Peter Bröcker

Computerphysik

Übungsblatt 1

SS 2014

Website: http://www.thp.uni-koeln.de/trebst/Lectures/2014-CompPhys.shtml
Abgabedatum: Montag, 14. April 2014 vor Beginn der Vorlesung

Auf diesem ersten Übungsblatt wollen wir die in der Vorlesung besprochenen **iterativen Verfahren** anhand erster einfacher Beispiele ein wenig vertiefen.

4. Plotten mit Python

Programmiertechniken

Daten lassen sich am Computer auf verschiedenste Weise **graphisch darstellen** – Liniengraphiken und bitmap-basierte Darstellungen in 2D Plots oder etwa Visualisierungen dreidimensionaler Datensets, welche sich sogar relativ einfach aus verschiedenen Perspektiven betrachten lassen. Auch die zugrunde liegenden Daten können sich dabei in ihrer Form sehr voneinander unterscheiden – sie können etwa als zu plottende Funktionen in einer geschlossenen analytischen Form vorliegen oder als Set separater Datenpunkte.

In Python bietet das Paket **matplotlib** umfangreiche Funktionen an, ebensolche Visualisierungen umzusetzen und in verschiedensten Datei-Formaten abzuspeichern. Folgen Sie dem obigen Link auf die Webseite des Pakets und machen Sie sich damit vertraut.

Mittels des folgenden Codeausschnitts wollen wir das Erstellen eines einfachen Plots einüben. Studieren Sie ihn und beschreiben Sie, was die einzelnen Befehle bewirken.

```
import matplotlib.pylab as plt
import numpy as np

plt.figure(figsize=(10,6))

for d in [4, 20, 500]:
    x_values = np.linspace(0, 4, d)
    y_values = [ np.sqrt(x) for x in x_values ]
    plt.plot(x_values, y_values, label=str(d))

plt.xlabel("x");
plt.ylabel("y");
plt.legend()

plt.savefig("test.pdf")
plt.show()
```

Nun wollen wir uns **mehrdimensionalen Abbildungen** zuwenden. Auch hier haben Sie die Wahl zwischen verschiedenen Darstellungstypen, die je nach Art der darzustellenden Daten unterschiedlich gut geeignet sein können. Wie in der vorherigen Aufgabe haben wir einen fertigen Code zusammengestellt, mit dem Sie experimentieren können. Versuchen Sie etwa, die Maschenweite des Gitternetzes zu verändern. Wie verändern sich die Variablen *X, Y* durch den Befehl *meshgrid*? Finden Sie anhand der matplotlib Dokumentation heraus, wie sie die Achsen des ersten Bildes so anpassen können, dass die korrekten Einheiten gezeigt werden. Im Code sind schon Hinweise darauf enthalten, wie man zwei Graphen in einem Bild darstellt. Vereinen Sie die beiden Konturplots und speichern Sie das resultierende Bild als PDF-Datei.

5. Gumowski-Mira Attraktor

5 Punkte

Schreiben Sie ein Programm zur Berechnung der folgenden Sequenz von Koordinaten (x_n, y_n)

$$\begin{aligned} x_{n+1} &= y_n + a \cdot \left(1 + b \cdot y_n^2\right) \cdot y_n + f(x_n) \\ y_{n+1} &= -x_n + f(x_{n+1}) \end{aligned}$$

$$f(x) = c \cdot x + \frac{2 \cdot (1-c) \cdot x^2}{1+x^2},$$

wobei die Parameter a, b, c wie unten spezifiziert vorgegeben seien.

Stellen Sie die Sequenz (x_n, y_n) *graphisch* dar, indem Sie für jedes Koordinatenpaar einen Punkt an entsprechender Stelle setzen. Sie werden damit den sogenannten **Gumowski-Mira Attraktor** visualisieren.

Plotten Sie die ersten 100.000 Punkte der Sequenz für

$$\begin{aligned} a &= 0.026 \\ b &= -0.034 \\ c &= -0.802095 \end{aligned}$$

mit $(x_0, y_0) = (0.1, 0.1)$ im Koordinatenbereich $-18 \leq x \leq 22$ und $-14 \leq y \leq 12$.

Plotten Sie die ersten 100.000 Punkte der Sequenz für

$$\begin{aligned} a &= -0.05 \\ b &= 0.005 \\ c &= -0.495 \end{aligned}$$

mit $(x_0, y_0) = (0.085, 0.085)$ im Koordinatenbereich $-12 \leq x \leq 16$ und $-15 \leq y \leq 8$.

6. Brüchige Bögen

5 Punkte

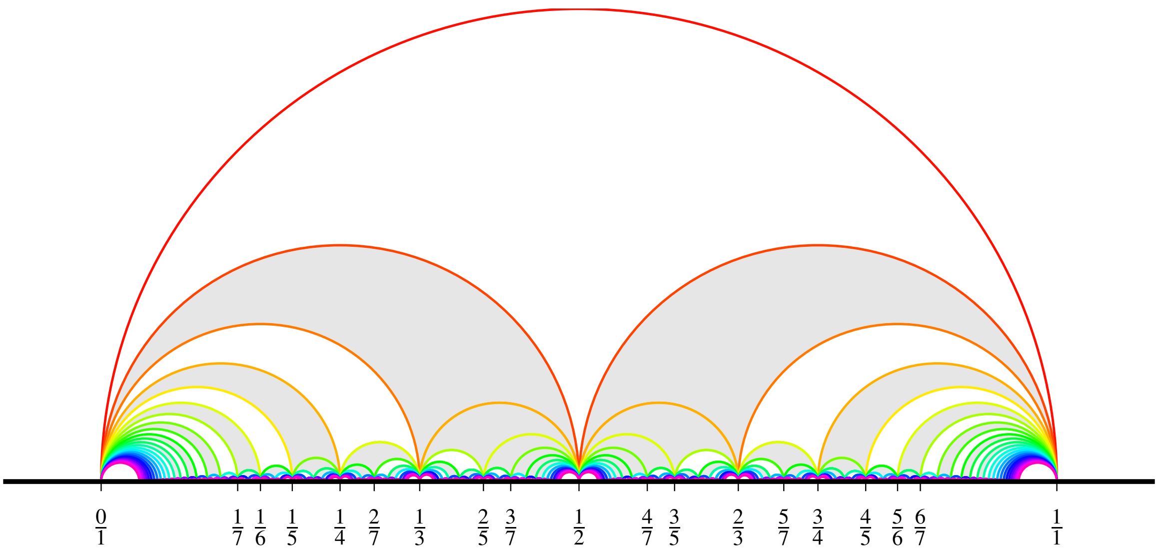
Schreiben Sie ein Programm zur Konstruktion der **Farey-Folge** in beliebiger Ordnung.

Die Farey-Folge n-ter Ordnung ist die Folge aller vollständig gekürzten Brüche p_i/q_i zwischen 0 und 1, deren Nenner q maximal n ist. Die ersten beiden Glieder der Farey-Folge sind $\{\frac{0}{1}, \frac{1}{1}\}$ für $n = 1$. John Farey, nach welchem die Folge später benannt wurde, hat die Vermutung aufgestellt, dass man die zusätzlichen Glieder der Folge $n+1$-ter Ordnung aus der n-ter Ordnung berechnen kann, indem man den Ausdruck

$$\frac{p_1 + p_2}{q_1 + q_2}$$

aus zwei benachbarten Folgengliedern der Folge n-ter Ordnung auswertet. Bildet man den obigen Ausdruck für die beiden ersten Glieder der Folge, so erhält man mit $\frac{0+1}{1+1} = \frac{1}{2}$ den neu hinzukommenden Term der Folge zweiter Ordnung. Wie sieht die Farey-Folge in sechster Ordnung aus?

Die Folge lässt sich außerdem sehr schön visualisieren, etwa wie folgt:



Untersuchen Sie, wie diese Darstellung erstellt wurde, und geben Sie einen Algorithmus zur Darstellung der sich ergebenden Bögen an.

Aufgaben

1. Schreiben Sie eine Funktion, die die **Farey-Folge** mit einer beliebigen Ordnung n ausgibt. Der Bruch soll dabei als Paar (p, q) dargestellt sein.
2. (optional) Erstellen Sie eine Illustration der Farey-Bögen ähnlich dem oben dargestellten.

7. Zellulärer Automat

optionale Aufgabe – 6 Punkte
Abgabe am Dienstag, 22. April

In dieser *optionalen Aufgabe* wollen wir einen **zellulären Automaten** betrachten, dessen Zellen durch drei Parameter $a, b, c \in [0, 1]$ beschrieben seien, welche die Konzentration dreier Spezies a, b, c beschreiben sollen. Anfänglich seien diese für jede Zelle unabhängig auf drei zufällige Werte in $[0, 1]$ gesetzt.

Wir iterieren nun den Zustand dieser Zellen, indem wir folgende Regel iterativ anwenden:

- Bestimme für jede Zelle die *durchschnittlichen* Werte der Konzentrationen $\bar{a}, \bar{b}, \bar{c}$ in der 3×3 Nachbarschaft der Zelle (wobei die Zelle selbst mit eingeschlossen sei).
- Der neue Zustand der Zelle sei gegeben durch

$$\begin{aligned} a &= \min\big(1.0\,,\, \bar{a} + \bar{a} \cdot (\bar{b} - \bar{c})\big) \\ b &= \min\big(1.0\,,\, \bar{b} + \bar{b} \cdot (\bar{c} - \bar{a})\big) \\ c &= \min\big(1.0\,,\, \bar{c} + \bar{c} \cdot (\bar{a} - \bar{b})\big) \end{aligned}$$

Beachten Sie dabei, dass alle Zellen *gleichzeitig* in den neuen Zustand übergehen.

Schreiben Sie ein Programm, welchen den oben beschriebenen zellulären Automaten umsetzt. Visualisieren Sie den Zustand der Zellen, indem Sie eine der drei Konzentrationen, etwa a, farbkodiert darstellen. Iterieren Sie den zellulären Automaten ausgehend von einer zufälligen Anfangsverteilung. Unabhängig von dieser Startkonfiguration werden sich ab ca. 80 Iterationen regelmäasige Strukturen ausbilden. Beschreiben Sie diese.

Tip: Um Ihr Program zunächst zu debuggen, starten Sie mit einem kleinen Gitter von etwa 128×128 Zellen (nicht viel kleiner). Wenn Ihnen die visualisierten Strukturen zusagen, erhöhen Sie die Auflösung und simulieren einen erheblich grösseren Automaten.