TU Berlin – FG Angewandte Mathematik
Prof. Dr. Gabriele Steidl
Dr. Johannes Hertrich, Jonas Bresch, Viktor Stein

Numerische Mathematik I
Wintersemester 2023/24

Programmieraufgabe 3 - Das Newton-Verfahren

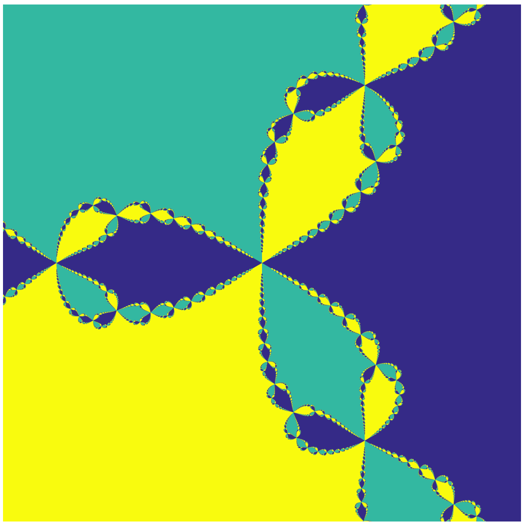
In dieser Programmieraufgabe werdet ihr das Newton-Verfahren zur Bestimmung von Nullstellen und Minimierern implementieren.

1. Schreibe eine Funktion `newton(F, dF, x0, delta, epsilon, maxIter)`, welche das Newton-Verfahren für die Funktion $F\colon \mathbb{R}^n \to \mathbb{R}^n$ (`F`), deren Ableitung J_F (`dF`) und den Startwert $x^{(0)}$ (`x0`) ausführt. Die Parameter δ (`delta`) und ε (`epsilon`) sind die Genauigkeiten für die Abbruchkriterien ($\|F(x^{(k)})\|_2 < \delta$ und $\|x^{(k+1)} - x^{(k)}\|_2 < \varepsilon$) und `maxIter` die Anzahl der maximalen Iterationen. Implementiere die letzten drei Parameter optional mit Standardwerten $\delta = \varepsilon = 10^{-4}$ und maximal 100 Iterationen.

2. Verwende das implementierte Newton-Verfahren, um die Nullstellen der Funktion
$$f\colon \mathbb{R} \to \mathbb{R}, \qquad x \mapsto x^3 - 2x$$
zu bestimmen. Verwende hierfür die Startwerte $x^{(0)} = 1/10$, $x^{(0)} = 2$ und $x^{(0)} = -2$, die Toleranzen $\delta = \varepsilon = 10^{-10}$ und maximal 50 Iterationen. Stelle die Funktion f auf dem Interval $[-21/10, 22/10]$ grafisch dar und markiere die berechneten Nullstellen.

3. Verwende das in 1. implementierte Newton-Verfahren, um die Nullstellen der Funktion
$$F\colon \mathbb{R}^2 \to \mathbb{R}^2, \qquad x \mapsto \begin{pmatrix} x_1^2 + x_2^2 - 6x_1 \\ \frac{3}{4}\, e^{-x_1} - x_2 \end{pmatrix}$$
zu bestimmen. Verwende hierfür den Startwert $x^{(0)} = (2/25, 7/10)^\mathsf{T}$. Gib das Ergebnis auf den Bildschirm aus.

4. Verwende das implementierte Newton-Verfahren, um die Gleichung
$$z^3 = 1$$
mit $z \in \mathbb{C}$ zu lösen. Das Ergebnis, eine der dritten Einheitswurzeln, hängt maßgeblich vom gewählten Startwert ab. Um das Konvergenzverhalten des Newton-Verfahrens zu untersuchen, diskretisiere die Menge $B_\infty = \{x + iy : -1 \le x, y \le 1\}$ in x- und y-Richtung jeweils durch 512 äquidistante Punkte. Verwende diese als Startwerte für das Newton-Verfahren mit $\delta = \varepsilon = 10^{-5}$ und maximal 15 Iterationen. Erzeuge aus den Ergebnissen ein Farbbild, wobei alle Gitterpunkte, die gegen dieselbe Einheitswurzel konvergieren, die gleiche Farbe haben. Mit deutlich mehr als 512^2 Gitterpunkten wird das Ergebnis so aussehen:

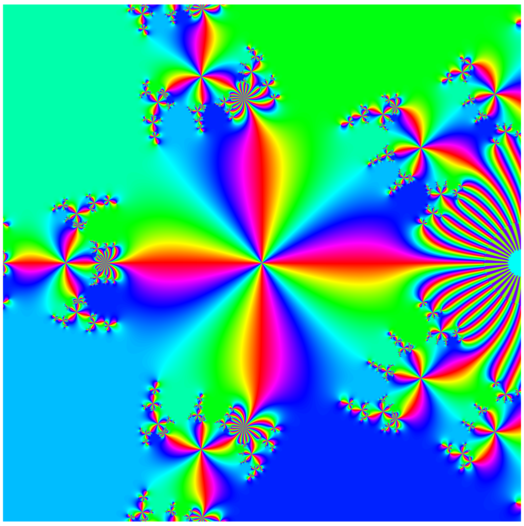
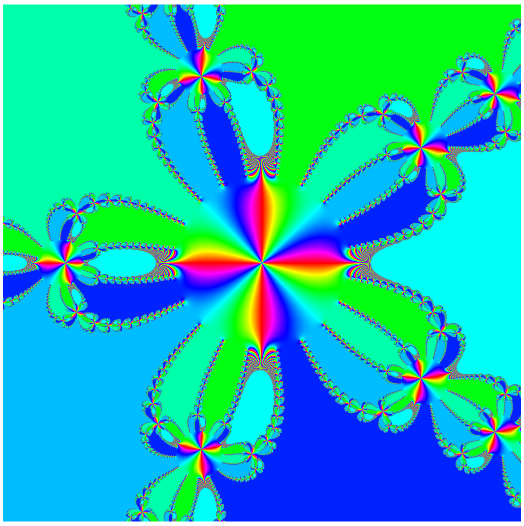


Es kann hier Startwerte geben, für welche das Newton-Verfahren nicht konvergiert.

5. Verwende das implementierte Newton-Verfahren, um die Gleichung

$$z^5 = 1,$$

mit $z \in \mathbb{C}$ zu lösen. Wähle als Startwerte wieder die Gitterpunkte des diskretisierten Balls B_∞. Für die restlichen Parameter verwende $\delta = \varepsilon = 10^{-14}$ und starte das Newton-Verfahren einmal mit `maxIter = 5` und `maxIter = 15`. Erzeuge für beide Experimente aus den Ergebnissen jeweils ein Farbbild, wobei der Farbwert die Phase[1] des Ergebnisses repräsentiert. Mit deutlich mehr als 512^2 Gitterpunkten wird das Ergebnis so aussehen:

Es kann hier Startwerte geben, für welche das Newton-Verfahren nicht konvergiert.

6. Verwende das implementierte Newton-Verfahren aus dem ersten Aufgabenteil, um die Funktion

$$F\colon \mathbb{R}^2 \to \mathbb{R}, \qquad x \mapsto (x_1+1)^4 + (x_2-1)^4$$

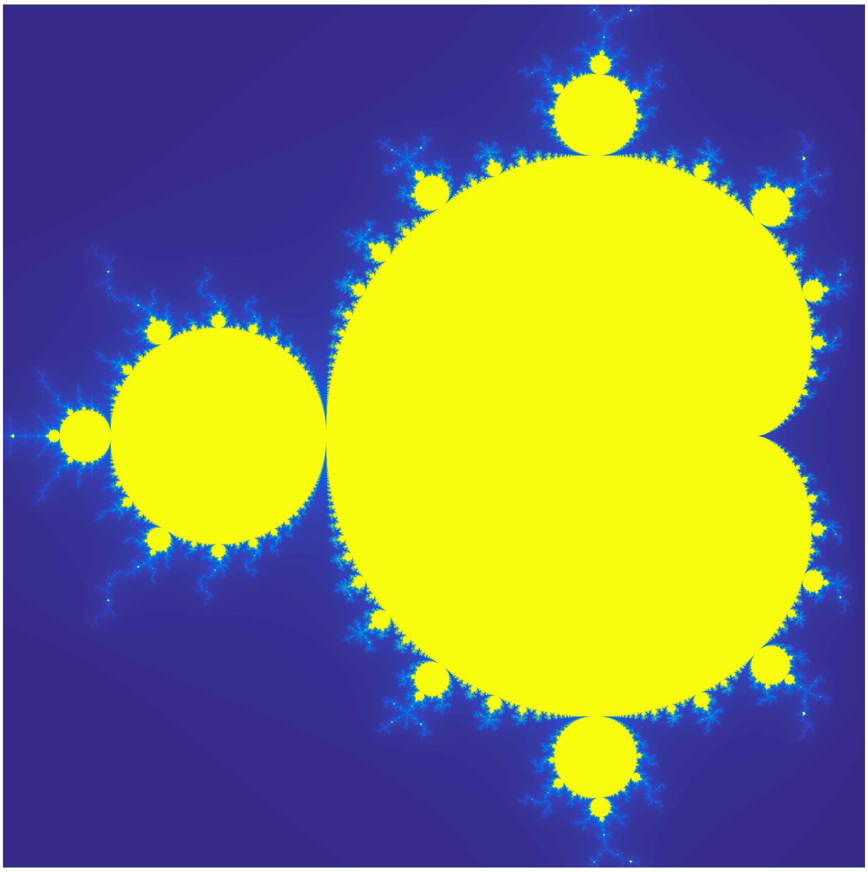
[1] Die Phase einer komplexen Zahl $z \in \mathbb{C}$ ist der Winkel $\varphi \in [-\pi, \pi)$ in der Polardarstellung $z = re^{i\varphi}$ mit $r \geq 0$.

zu minimieren. Verwende hierfür den Startwert $x^{(0)} = {}^{11}\!/_{10}(-1,1)^{\mathsf{T}}$ und gebe das Ergebnis auf den Bildschirm aus.

7. Betrachte die Iteration

$$z^{(k+1)} = (z^{(k)})^2 + c \qquad \text{mit} \qquad z^{(0)} = c$$

für ein $c \in \mathbb{C}$. Als Startwerte verwende die Gitterpunkte $M := \{x + iy \in \mathbb{C} : -1.5 \leq x \leq 0.5, -1 \leq y \leq 1\}$, wobei in x- und in y-Richtung jeweils mit 1024 äquidistanten Punkten diskretisiert wird. In manchen Punkten konvergiert diese Folge, in anderen divergiert sie. Um dies grafisch darzustellen, bestimme für jeden Gitterpunkt c die Anzahl der Iterierten $z^{(k)}$, deren Betrag kleiner oder gleich 2 ist. Führe für jeden Punkt maximal 256 Iterationen aus. Mit deutlich mehr als 1024^2 Gitterpunkten wird das Ergebnis so aussehen:



Die gelben Punkte bilden die Mandelbrot-Menge, ein Fraktal, welches im allgemeinen Sprachgebrauch auch „Apfelmännchen" gennant wird.

Verwende außer `skimage`, `numpy`, `scipy` und `matplotlib` keine weiteren Pakete. Kommentiere den Quellcode! Füge deiner Abgabe ein Hauptprogramm (Python-Skript) bei, welches eigenständig alle Ausgaben erzeugt. Beschrifte die Ausgaben direkt oder erläutere in einer Readme-Datei in welcher Reihenfolge die Ausgaben erfolgen.