

# Woche 8

Freitag, 28. April 2023 11:14

Was kommt noch?

Nullstellensuche

steife ODE's

Matrizenzerlegungen

Ausgleichsrechnung  
 $\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_2$



Eigenwerte

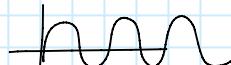
## Nullstellensuche

Ziel der ÜS: existierende Verfahren kennen und Konvergenz in diesem Zusammenhang

Allgemein: geg.:  $D \subseteq \mathbb{R}^n$ ,  $F: D \rightarrow \mathbb{R}^n$

ges.:  $x^* \in D$  s.d.  $F(x^*) = 0$

Ist das  $x^*$  eindeutig? i.A nicht!



## Wieso wichtig?

1) Gleichungssysteme können immer als Nullstellenproblem geschrieben werden:

$$\begin{cases} x_1^2 + \sin(x_2) = 3 \\ \sqrt{x_2^2 + 3} = e^{x_1} \end{cases} \quad \rightarrow \quad \begin{pmatrix} x_1^2 + \sin(x_2) - 3 \\ \sqrt{x_2^2 + 3} - e^{x_1} \end{pmatrix} = 0$$

$\Rightarrow$  Lösung der Gleichung ist  $x^* = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$  s.d.  $F(x^*) = 0$

2) implizite ODE Solver müssen in jedem Schritt ein Gleichungssystem lösen

$\hookrightarrow$  effizientere Verfahren als fsolve notwendig

**Ansatz:** Iterative Verfahren von Startwert  $x_0$  aus (Wahl von  $x_0$  wichtig!)

Wir wollen also ein Verfahren  $\varphi: D \rightarrow D$  finden, s.d.  $\lim_{n \rightarrow \infty} \varphi^n(x_0) = x^*$   
mit  $F(x^*) = 0$ .

Wie klassifiziert man solche Verfahren? Konvergenzverhalten

D.h. "wie schnell" geht  $x_n \xrightarrow{\varphi} x^*$

## Def.: Fehler

Sei  $\|\cdot\|: \mathbb{R}^n \rightarrow \mathbb{R}_{>0}$  eine Norm und  $x^* \in \mathbb{R}^n$  s.d.  $F(x^*) = 0$  die gesuchte Lösung.

Dann definiere  $e_n := \|x_n - x^*\|$  als Fehler.

Für uns:  $\|\cdot\| = \|\cdot\|_2$  (np.linalg.norm(x, axis=1)) <sup>Falls array als input</sup>

## Def.: Konvergenz

Haben zwei Typen:

- Konvergenzordnung  $p$
- Konvergenzrate  $L$  (für  $p=1$ )

**Konvergenzordnung  $p$ :**  $\exists c > 0$  s.d.  $e_{k+1} \leq c \cdot e_k^p$

Wie in python bestimmen?  $\tilde{p} = \text{np.log}(e[2:] / e[1:-1]) / \text{np.log}(e[1:-1] / e[:-2])$

wobei  $e = (e_0, \dots, e_N)$  der Fehlerarray und wähle für  $p$  das letzte sinnvolle Resultat im Array  $\tilde{p}$

**Konvergenzrate  $L$  ( $p=1$ ):**  $e_{k+1} < c \cdot e_k \Rightarrow L = c$

$L = \text{np.exp}(\text{np.polyfit}(k, \text{np.log}(e), 1)[0])$  - ↳ Ggf. nicht über alle  $e_k$ 's

passen, sondern nur für genügend grosse  $k$ 's, z.B.  $k > n: e[n:], k[n:]$

Jetzt können wir also unsere Verfahren vergleichen. Aber wann bricht ein Verfahren ab?

## Abruchkriterien:

a) Anzahl Iterationen

b)  $\|x_{k+1} - x_k\| < \varepsilon$

c)  $\|x_{k+1} - x_k\| \leq \varepsilon \|x_k\|$

d) Residuum  $\|F(x_k)\| \leq \varepsilon$  Nicht gut



Man verwendet (a) + (c) um Genauigkeit für die Lösung zu gewährleisten und um Terminierung zu gewährleisten

## Verfahren:

1) Fixpunktiteration:  $F(x^*) = 0 \Rightarrow \varphi(x^*) = x^*$

Idee: Rufe  $\varphi$  iterativ auf und hoffe auf Konvergenz bei  $x^*$

Schreibe  $F$  um zu Fixpunktproblem und rufe  $\varphi$  iterativ auf, d.h.  $x_{k+1} = \varphi(x_k)$

(Banachscher Fixpunktsatz) s.d.  $\lim_{k \rightarrow \infty} x_k = x^*$

Bsp.:  $F(x) = xe^x - 1 \stackrel{!}{=} 0$

a)  $F(x) = 0 \Leftrightarrow x = e^{-x} \Rightarrow \varphi_1(x) = e^{-x}$

nicht alle Möglichkeiten konvergieren!

b) "  $x+1-xe^{-x} = x \Rightarrow \varphi_2(x) = x+1-xe^{-x}$

Lemma: Sei  $\varphi: \mathbb{R}^n \rightarrow \mathbb{R}^n$  mit  $\varphi(x^*) = x^*$  stetig diff'bar, s.d.

$\|\nabla \varphi(x^*)\| < 1$ , dann haben wir lokal (aka. Startwert schlaw gewählt) min.

lineare Konvergenz

a)  $|\varphi'_1(0,6)| = e^{-0,6} < 1 \Rightarrow$  Konvergenz

b)  $|\varphi'_2(0,6)| = |1 - e^{0,6} - 0,6 \cdot e^{0,6}| > 1 \Rightarrow$  Divergenz

Jede Nullstellengleichung  $f(x) = 0$  lässt sich in eine Fixpunktgleichung  $\varphi(x) = x$  umschreiben, indem man z.B.  $\varphi(x) = x + f(x)$  setzt

## 2) Bisektionsverfahren:

Idee: ZWS für stetige Funktionen:  $F: [a,b] \rightarrow \mathbb{R}$  mit  $F(a) < 0$

und  $F(b) > 0$ , halbiere Intervall immer wieder aufs neue und

tausche  $m = \frac{a+b}{2}$  mit  $a$  oder  $b$

Pro: Benötigen keinen guten Startwert

Con: lineare Konvergenz, nur für 1D Funktionen

Python: `scipy.optimize.bisect(F, a, b)`

## 3) Newton-Verfahren:

Idee: Versuche nicht  $F(x) = 0$  zu lösen, sondern linearisiere  $F$  um  $x$  und löse das einfachere Problem  $\tilde{F}(x) = 0$  für  $\tilde{F}$  linear  $\approx F$

1D:  $\tilde{F}(x) := F(x_k) + F'(x_k)(x - x_k)$  Tangente an  $F$  bei  $x_k$

$$\tilde{F}(x) = 0 \Leftrightarrow F(x_k) + F'(x_k)(x - x_k) = 0$$

$$\Leftrightarrow x = x_k - \underbrace{\frac{F(x_k)}{F'(x_k)}}_{=: S} \text{ Newton Korrektur}$$

Iterativ:  $x_{k+1} := x_k - \underbrace{\frac{F(x_k)}{F'(x_k)}}_{=: S}$

Wie das effizient tun?

ND:  $x_{k+1} = x_k - \underbrace{DF(x_k)^{-1}}_{A} \underbrace{F(x_k)}_{x} \underbrace{- b}_{b}$

Da Inversenbildung teuer ist, berechne nie  $DF(x_k)^{-1}$ , sondern löse LGS

$$S = DF(x_k)^{-1} F(x_k) \Leftrightarrow \underbrace{DF(x_k)}_{A} \underbrace{S}_{x} = \underbrace{F(x_k)}_{b}$$

Python:  $s = \text{np.linalg.solve}(DF(x_k), F(x_k))$

konvergiert lokal mit Ordnung  $p=2$

Ableitung  $DF$  muss bekannt sein, sonst:

1D: Sekantenverfahren

ND: Broyden-Verfahren

um das Konvergenzgebiet zu vergrößern, kann das gedämpfte Newton-Verfahren verwendet werden