

INPUT : $A \in \mathbb{R}^{n \times n}$ (invertierbar)
 $b \in \mathbb{R}^n$

OUTPUT : $x \in \mathbb{R}^n$ mit $Ax = b$

ALGORITHMUS: rel. Richardson:
 $x^{k+1} = x^k - \Theta(Ax^k - b)$

↳ PARAMETER: conf.py

- A : Matrix
- b : Vektor
- x^0 : Startvektor
- Θ : Schrittweite
- tol : Fehlertoleranz
- $maxiter$: max. Iterationszahl
- (*Lösung: für callback?)

↳ WAS MUSS IMPLEMENTIERT WERDEN?

Wir gehen lediglich von Basistypen aus.

• FORMAT FÜR MATRIZEN & Vektoren

- Vektoren: List()
- Matrix: CSR-Klasse

• LA-ROUTINEN

- Skalarmultiplikation: float x List
- Addition: list + list
- Euklidische Norm: $(float^2 + \dots + float^2)^{1/2}$
- Matrix-Vektor Produkt: CSR x list

• rel. Richardson

• ERGEBNISSE AUSWERTEN

- Tabellen speichern
- Grafiken erstellen

helpus.py

laLib.py

iterSolver.py

SINNVOLE MODULARISIERUNG: [siehe Kursnotizen]

- Metadaten
- Inhaltliche Bausteine
- Medien
- Literatur
- LaTeX Konfiguration
- ...

CODE

Code

- src
 - laLib.py
 - iterSolver.py
 - helpus.py
- examples
 - confEx1.py
 - confText1.py
- output
 - table.tex/mid
 - plot.png
- main.py
- README.md
- requirements.txt
- .git
- docs
- .idea
- .venv

← VERSIONSKONTROLLE

← DOCUMENTATION

← IDE Project

← Virtual Python environment

- Grafiken
- Tabellen

↑
 [kontinuierliche
 Integration von
 Code & Text]

TEXT

- text

- macros

- paper

- content

- ...

- main.article.tex

- meta.tex

- media

...