Hydroinformatik - SoSe 2025

UW-BHW-414-J: Finite-Differenzen-Methode I

Prof. Dr.-Ing. habil. Olaf Kolditz

[1]Helmholtz Centre for Environmental Research – UFZ, Leipzig

[2]Technische Universität Dresden – TUD, Dresden

[3]Center for Advanced Water Research – CAWR

[4]TUBAF-UFZ Center for Environmental Geosciences – C-EGS, Freiberg / Leipzig

Dresden, 27.06.2025

TECHNISCHE UNIVERSITÄT DRESDEN

Zeitplan: Hydroinformatik I+II

Sommersemester 2025

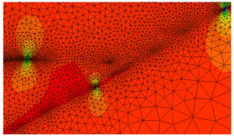
Stand: 08.05.2025

Nr.	KW	Datum	ID	Thema
01	15	11.04.2025	UW-BHW-414-A	Einführung, Werkzeuge#1, Hello World
03	17	25.04.2025	UW-BHW-414-B	Umweltinformatik, Werkzeuge#2 (git), Datentypen
05	18	02.05.2025	UW-BHW-414-C	Selbststudium
07	19	09.05.2025	UW-BHW-414-D	Objekt-Orientierte Programmierung: C++, Klassen
09	20	16.05.2025	UW-BHW-414-E	Python
11	21	23.05.2025	UW-BHW-414-F	Modellierung, Digitalisierung, Wasser 4.0
13	22	30.05.2025	UW-BHW-414-G	KI, Maschinelles Lernen, Neuronale Netzwerke
15	23	06.06.2025	UW-BHW-414-H	Kontinuumsmechanik, Hydromechanik
05	18	13.06.2025		Vorlesungsfreie Woche
17	25	20.06.2025	UW-BHW-414-I	Differentialgleichungen, Näherungsverfahren
19	26	27.06.2025	UW-BHW-414-J	Finite-Differenzen, explizite Verfahren
21	27	04.07.2025	UW-BHW-414-K	Finite-Differenzen, implizite Verfahren
24	28	11.07.2025	UW-BHW-414-L	Gerinnehydraulik, Grundwasserhydraulik
25	29	18.07.2025	UW-BHW-414-M	Zusammenfassung, Klausurvorbereitung

TECHNISCHE UNIVERSITÄT DRESDEN

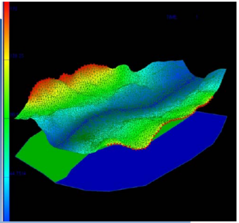
Konzept

$$\frac{d\psi}{dt} = \frac{\partial\psi}{\partial t} + \mathbf{v}^E \nabla\psi$$

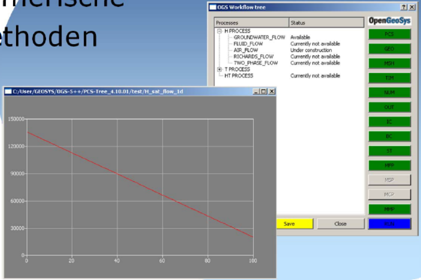


Basics
Mechanik

Anwendung



Numerische
Methoden



Programmierung
Visual C++

Prozessverständnis

Fahrplan für heute

0 Jupyter installation: Probleme? ($\Rightarrow$ 5)

---

1 Grundlagen der Finite Differenzen Methode

2 Approximation methods (FDM, FEM)

3 Finite difference method – FDM (Ch. 3)

4 Taylor series expansion

5 Derivatives

6 Diffusion equation

---

7 Implizites FDM Verfahren

TECHNISCHE UNIVERSITÄT DRESDEN

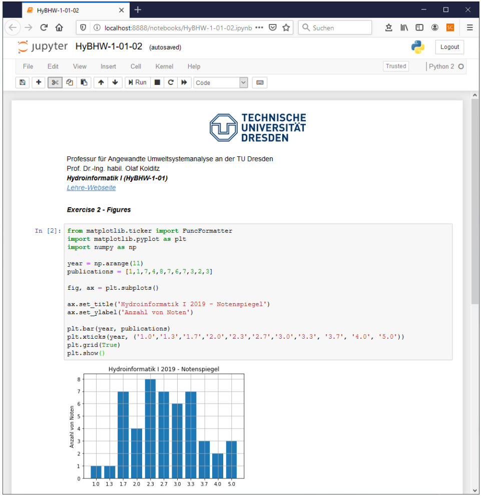
Jupyter

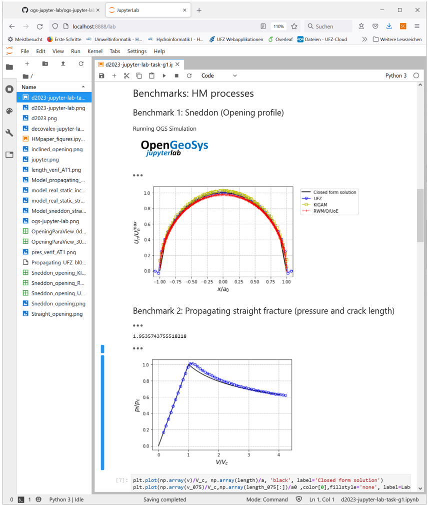
- Jupyter Notebook
- Jupyter Lab
- Browser-basiert

Jupyter

- "The Jupyter Notebook · The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, ..."
- Webseite: `https://jupyter.org/`
- Vorteil: funktioniert auf allen Rechnern
- ... ein Teil unserer (neuen) Übungen machen wir mit Jupyter Notebooks (>> Demo)

Jupyter: Example

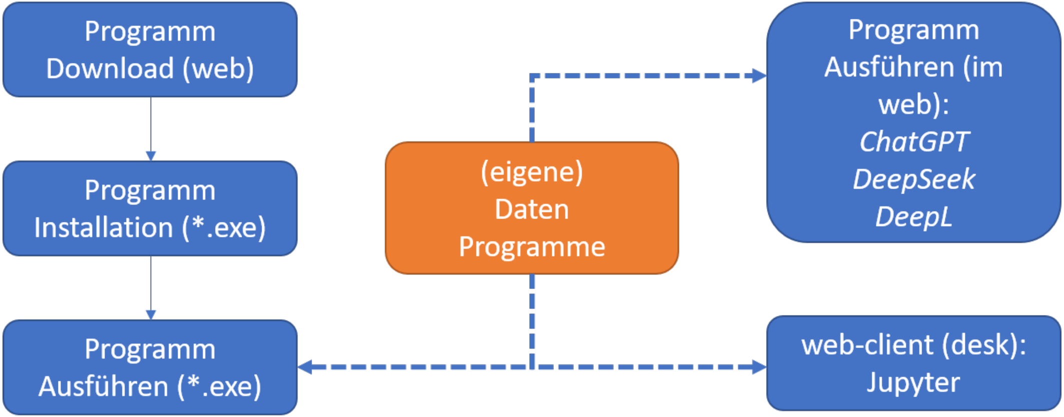




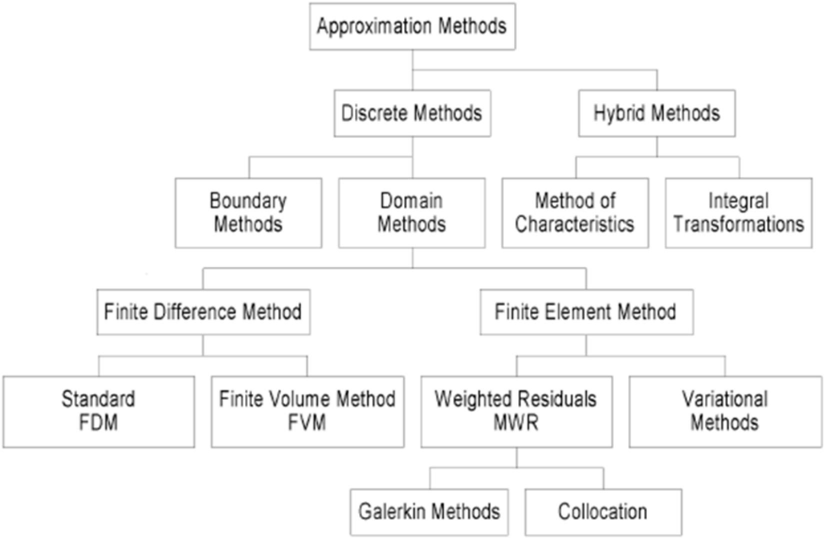
Jupyter: Installation

`https://jupyter.org/install`

Desk und Web-Anwendungen



Näherungsverfahren



FDM Anwendungen - MODFLOW

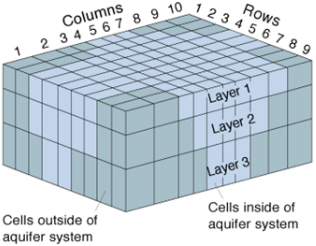


Figure 2. Example of model grid for simulating three-dimensional ground-water flow.

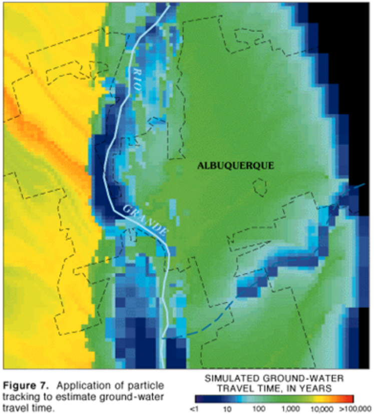
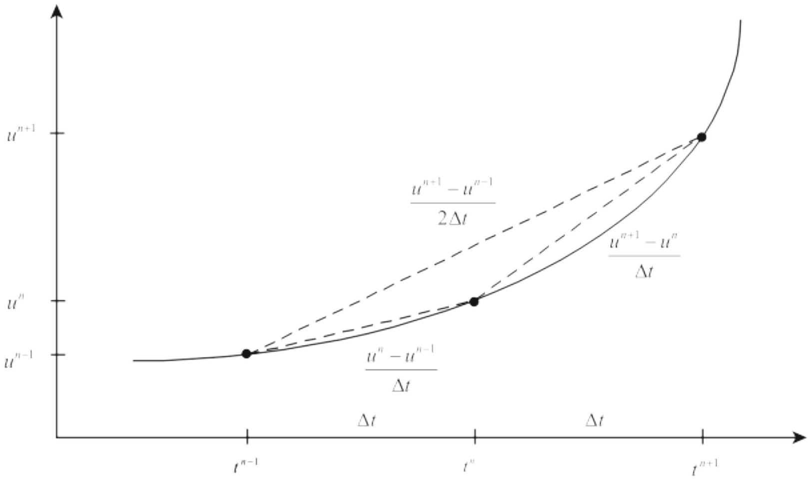


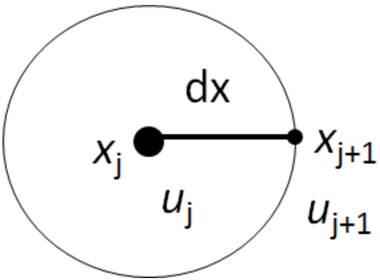
Figure 7. Application of particle tracking to estimate ground-water travel time.

http://water.usgs.gov/pubs/FS/FS-121-97/images/fig7.gif

Ableitungen



Taylor-Reihe



in time

$$u_j^{n+1} = \sum_{m=0}^{\infty} \frac{\Delta t^m}{m!} \left[\frac{\partial^m u}{\partial t^m} \right]_j^n \qquad (1)$$

$$\Delta t = t^{n+1} - t^n$$

in space

$$u_{j+1}^{n} = \sum_{m=0}^{\infty} \frac{\Delta x^m}{m!} \left[\frac{\partial^m u}{\partial x^m} \right]_j^n \qquad (2)$$

$$\Delta x = x_{j+1} - x_j$$

TECHNISCHE UNIVERSITÄT DRESDEN

Trunkation

$$u_j^{n+1} = u_j^n + \Delta t \left[\frac{\partial u}{\partial t}\right]_j^n + \frac{\Delta t^2}{2}\left[\frac{\partial^2 u}{\partial t^2}\right]_j^n + 0(\Delta t^3) \tag{3}$$

$$u_{j+1}^{n} = u_j^n + \Delta x \left[\frac{\partial u}{\partial x}\right]_j^n + \frac{\Delta x^2}{2}\left[\frac{\partial^2 u}{\partial x^2}\right]_j^n + 0(\Delta x^3) \tag{4}$$

siehe auch Trunkationsfehler (letzte Vorlesung)

1. Ableitung

$$\left[\frac{\partial u}{\partial t}\right]_j^n = \frac{u_j^{n+1} - u_j^n}{\Delta t} - \frac{\Delta t}{2}\left[\frac{\partial^2 u}{\partial t^2}\right]_j^n + 0(\Delta t^2) \tag{5}$$

$$\left[\frac{\partial u}{\partial x}\right]_j^n = \frac{u_{j+1}^{n} - u_j^n}{\Delta x} - \frac{\Delta x}{2}\left[\frac{\partial^2 u}{\partial x^2}\right]_j^n + 0(\Delta x^2) \tag{6}$$

Differenzen-Schemata

Forward difference approximation

$$\left[\frac{\partial u}{\partial x}\right]_j^n = \frac{u_{j+1}^n - u_j^n}{\Delta x} + 0(\Delta x) \tag{7}$$

Backward difference approximation

$$\left[\frac{\partial u}{\partial x}\right]_j^n = \frac{u_j^n - u_{j-1}^n}{\Delta x} + 0(\Delta x) \tag{8}$$

Central difference approximation

$$\left[\frac{\partial u}{\partial x}\right]_j^n = \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x} + 0(\Delta x^2) \tag{9}$$

TECHNISCHE UNIVERSITÄT DRESDEN

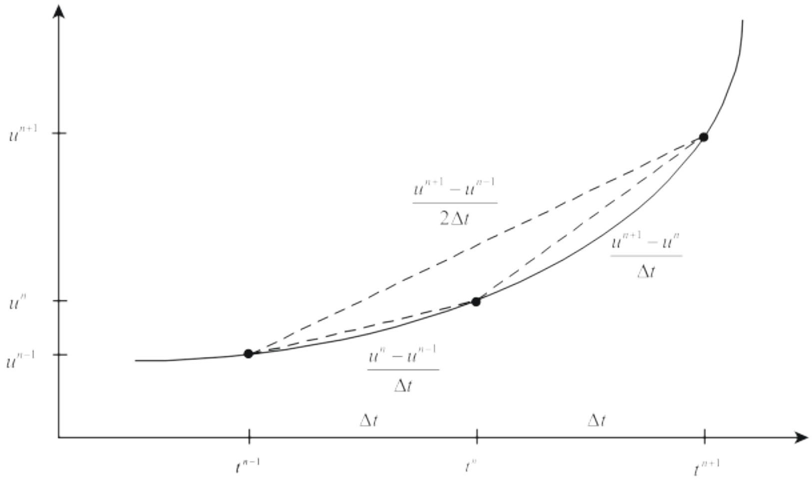
Zentrale Differenzen

$$\begin{aligned} u^n_{j+1} &= u^n_j + \Delta x \left[\frac{\partial u}{\partial x}\right]^n_j + \frac{\Delta x^2}{2}\left[\frac{\partial^2 u}{\partial x^2}\right]^n_j + 0(\Delta x^3) \\ u^n_{j-1} &= u^n_j - \Delta x \left[\frac{\partial u}{\partial x}\right]^n_j + \frac{\Delta x^2}{2}\left[\frac{\partial^2 u}{\partial x^2}\right]^n_j - 0(\Delta x^3) \end{aligned} \tag{10}$$

Central difference approximation

$$\left[\frac{\partial u}{\partial x}\right]^n_j = \frac{u^n_{j+1} - u^n_{j-1}}{2\Delta x} + 0(\Delta x^2) \tag{11}$$

Ableitungen



TECHNISCHE UNIVERSITÄT DRESDEN

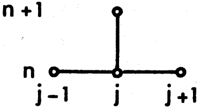
2. Ableitung

$$\left[\frac{\partial^2 u}{\partial x^2}\right]_j^n \approx \frac{1}{\Delta x}\left(\left[\frac{\partial u}{\partial x}\right]_{j+1}^n - \left[\frac{\partial u}{\partial x}\right]_j^n\right) \approx \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} \tag{12}$$

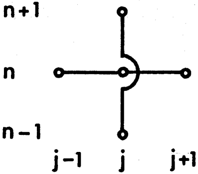
$$\left[\frac{\partial^2 u}{\partial x^2}\right]_j^n = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} + \frac{\Delta x^2}{12}\left[\frac{\partial^4 u}{\partial x^4}\right]_j^n + \dots \tag{13}$$

TECHNISCHE UNIVERSITÄT DRESDEN

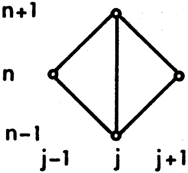
Übersicht Differenzenverfahren



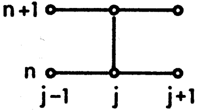
FTCS



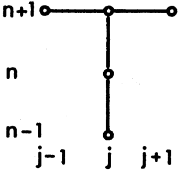
Richardson



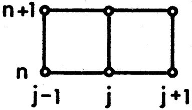
Du Fort-Frankel



Crank-Nicolson



3LFI



Linear F.E.M./
Crank-Nicolson

TECHNISCHE UNIVERSITÄT DRESDEN

Diffusionsgleichung

$$\frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} = 0 \tag{14}$$

TECHNISCHE UNIVERSITÄT DRESDEN

Analytical solution for diffusion equation (Skript 5.2.2)

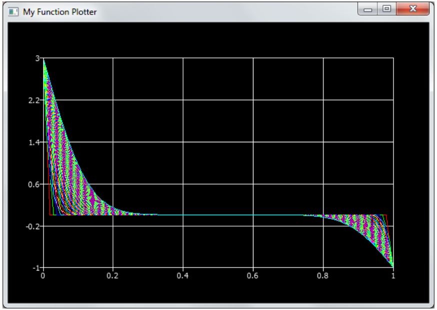
- Diffusion equation

$$\frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} = 0 \qquad (15)$$

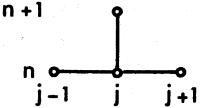
- Analytical solution

  Einschränkungen: Nur für symmetrische Randbedingungen

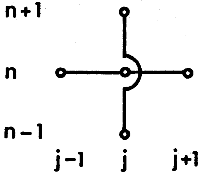
- K: validity



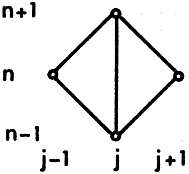
$\Rightarrow$ Übung



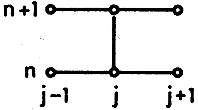
FTCS



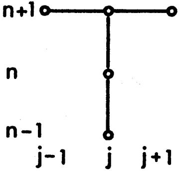
Richardson



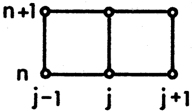
Du Fort-Frankel



Crank-Nicolson



3LFI



Linear F.E.M./
Crank-Nicolson

TECHNISCHE UNIVERSITÄT DRESDEN

Explizite FDM - FTCS Verfahren (Skript 3.2.2/4.1)

- PDE for diffusion processes

$$\frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} = 0 \tag{16}$$

- forward time / centered space

$$\left[\frac{\partial u}{\partial t}\right]_j^n \approx \frac{u_j^{n+1} - u_j^n}{\Delta t} \quad \left[\frac{\partial^2 u}{\partial x^2}\right]_j^n \approx \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{\Delta x^2} \tag{17}$$

- substitute

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} - \alpha \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{\Delta x^2} = 0 \tag{18}$$

- FTCS scheme for diffusion equations

$$u_j^{n+1} = u_j^n + \frac{\alpha \Delta t}{\Delta x^2}(u_{j-1}^n - 2u_j^n + u_{j+1}^n)\,, \; Ne = \frac{\alpha \Delta t}{\Delta x^2} \tag{19}$$

TECHNISCHE UNIVERSITÄT DRESDEN

Eigenschaften numerischer Verfahren

Analysis of approximation schemes consists of three steps:

- Develop the **algebraic scheme**,
- Check **consistency** of the algebraic approximate equation,
- Investigate **stability** behavior of the scheme.

Eigenschaften numerischer Verfahren

Analysis of approximation schemes consists of three steps:

- Develop the **algebraic scheme**,

$$u_j^{n+1} = u_j^n + \frac{\alpha \Delta t}{\Delta x^2}\left(u_{j-1}^n - 2u_j^n + u_{j+1}^n\right) \tag{20}$$

- Check **consistency** of the algebraic approximate equation,

$$\lim_{\Delta t, \Delta x \to 0} |\hat{L}(u_j^n) - L(u[t_n, x_j])| = 0 \tag{21}$$

- Investigate **stability** behavior of the scheme.

$$Ne = \frac{\alpha \Delta t}{\Delta x^2} \leqslant 1/2 \tag{22}$$

Lösung des FTCS Schemas

Algebraische Schema

$$u_j^{n+1} = u_j^n + \frac{\alpha \Delta t}{\Delta x^2}(u_{j-1}^n - 2u_j^n + u_{j+1}^n) \tag{23}$$

Resultierendes Gleichungssystem

$$\mathbf{u}^{n+1} = \mathbf{A}\mathbf{u}^n \qquad , \qquad n = 0, 1, 2, \ldots \tag{24}$$

$$\mathbf{A} = \begin{bmatrix} 1 - 2\frac{\alpha \Delta t}{\Delta x^2} & \frac{\alpha \Delta t}{\Delta x^2} & & & \\ \frac{\alpha \Delta t}{\Delta x^2} & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \frac{\alpha \Delta t}{\Delta x^2} \\ & & & \frac{\alpha \Delta t}{\Delta x^2} & 1 - 2\frac{\alpha \Delta t}{\Delta x^2} \end{bmatrix}, \mathbf{u}^n = \begin{bmatrix} u_2^n \\ u_3^n \\ \ldots \\ u_{np-2}^n \\ u_{np-1}^n \end{bmatrix}$$

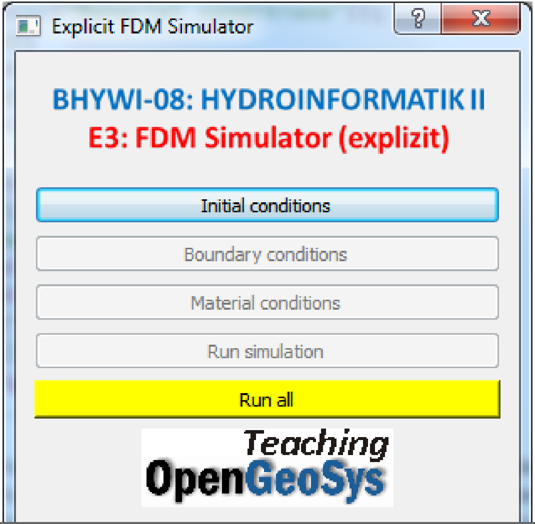
Übungen

- (Qt Übung (alt): BHYWI-08-03-E)
- C++ Übung: EX08-fdm-explicit-C++
- Python Übung: EX08-fdm-explicit-python

TECHNISCHE UNIVERSITÄT DRESDEN

FDM Übung: Explizit Qt

Progamm-Dialog



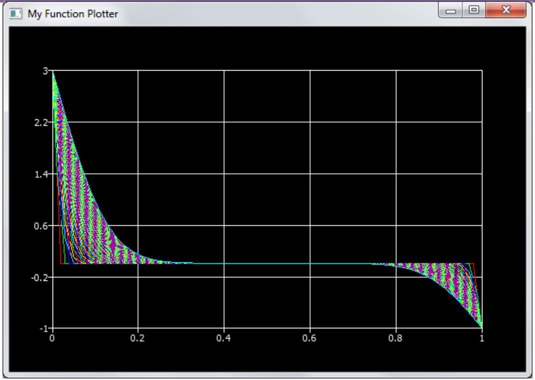
Dialog-Klasse: Konstruktor
`Dialog::Dialog`

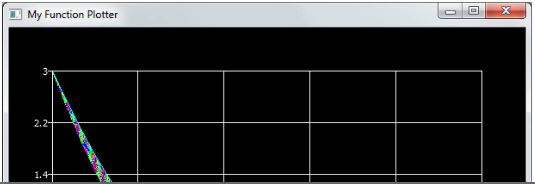
1. Elemente
2. Connects
3. Layout
4. Datenstrukturen (Speicherreservierung)

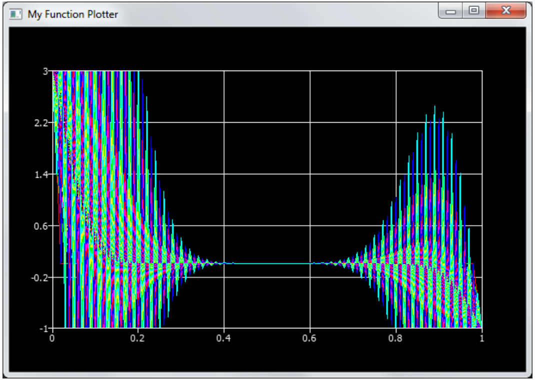
TECHNISCHE UNIVERSITÄT DRESDEN

FDM Übung: Explizit Qt

Ergebnisse



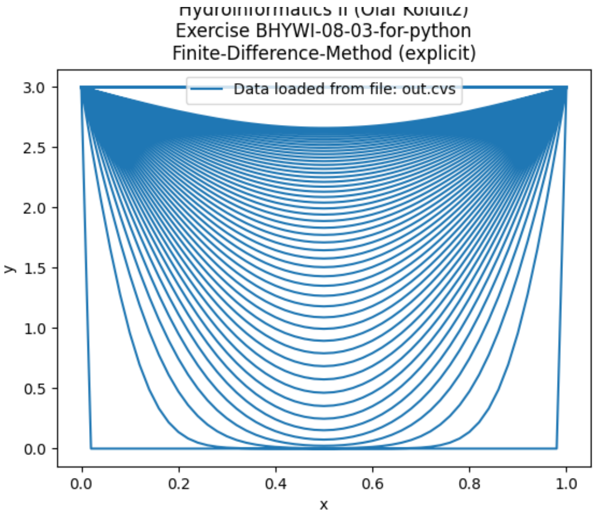




Übung: EX08 (C++): explizite FDM

```
echo Compilation
g++ main.cpp
echo Execution
a.exe
echo Ploting
data_from_file.py
echo End
```

Listing: Skript: C++ Berechnung > Python Grafik



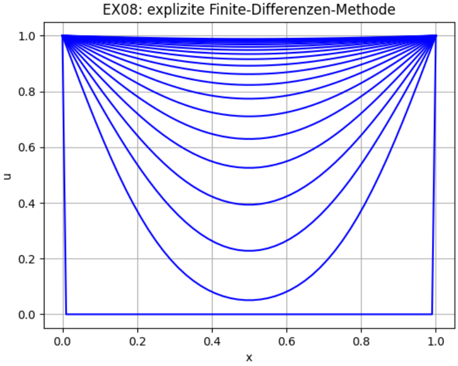
Übung: EX08 (Python): explizite FDM

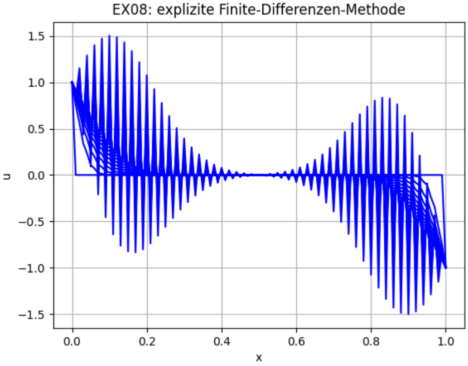
```python
import math
import numpy as np
import matplotlib.pylab as plt
#data structures
##physical parameter
alpha = 1.0
##numerical parameters (discretization)
nx = 10
x = np.zeros(nx+1)
dx = 1./nx
t = [0.01]
nt = 10 #wieviele Zeitschritte bis zum stationaeren
     Zustand
dt = 0.5 * dx*dx / alpha
Ne = alpha * dt / (dx*dx)
##field function
u = np.zeros(nx+1)
uo = np.zeros(nx+1)
#initial condition
u_ic = 0.
for i in range(nx+1):
  x[i] = 0
  u[i] = 0
  uo[i] = 0
#boundary conditions
u_bc_l = 1.
u_bc_r = 1.
u[0] = uo[0] = u_bc_l
u[nx] = uo[nx] = u_bc_r
#initial state
```

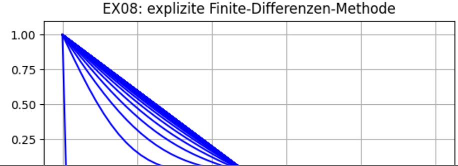
```python
...
#initial state
for i in range(0,nx+1):
    x[i] = (float(i)/float(nx))
plt.plot(x,u,color='blue')
#fdm-explicit
for n in range(1,nt):
  for i in range(1,nx):
    u[i] = uo[i] + Ne *(uo[i-1] - 2*uo[i] + uo[i+1])
  plt.plot(x,u,color='blue')
  for i in range(1,nx):
    uo[i] = u[i]
#plots
plt.title('EX08: explizite Finite-Differenzen-Methode')
plt.xlabel('x')
plt.ylabel('u')
plt.axis('tight')
plt.grid()
plt.savefig("fdm-explicit.png")
plt.show()
```

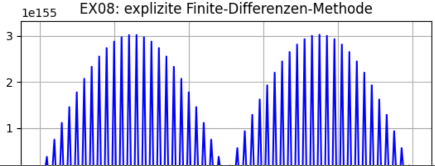
Listing: …

Übung: EX08 (Python): explizite FDM

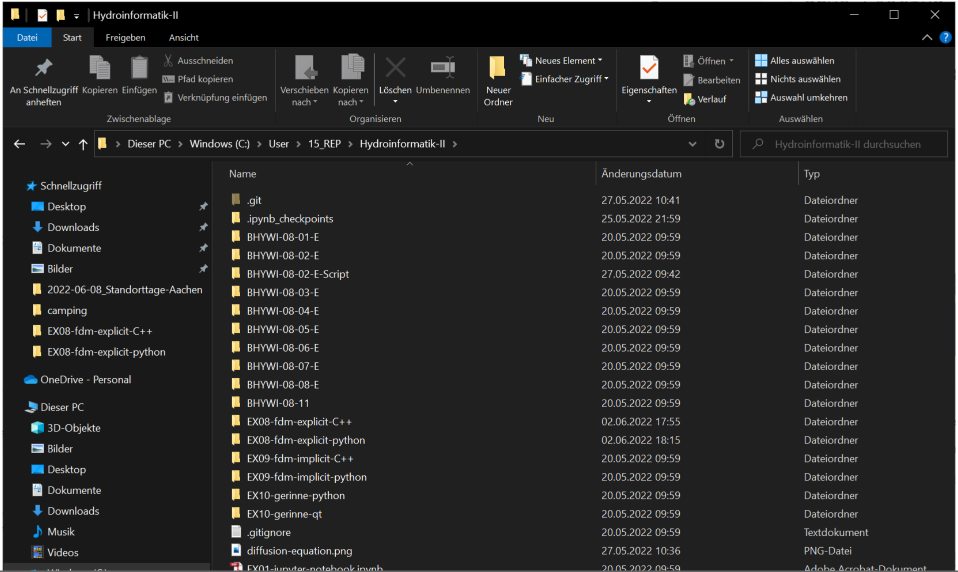








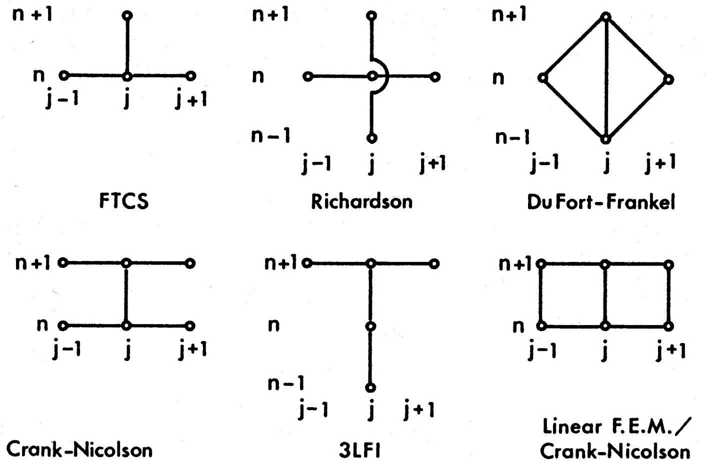
Hydroinformatik II Übungen - Repository



TECHNISCHE UNIVERSITÄT DRESDEN

Alternative Verfahren

Explizite und implizite Differenzenverfahren



Implizites Differenzenverfahren: Next Lecture

Algebraische Schema:

$$\left[\frac{\partial^2 u}{\partial x^2}\right]_j^{n+1} \approx \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{\Delta x^2} \tag{26}$$

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} - \alpha \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{\Delta x^2} = 0 \tag{27}$$

$$\frac{\alpha \Delta t}{\Delta x^2}\left(-u_{j-1}^{n+1} + 2u_j^{n+1} - u_{j+1}^{n+1}\right) + u_j^{n+1} = u_j^n \tag{28}$$

TECHNISCHE UNIVERSITÄT DRESDEN

Farbcode

Vorlesung

Übung

Hausaufgabe

Klausur

Hausaufgaben

Hausaufgaben: Hydroinformatik

Hydroinformatik II - HyBHW-1-02

1. Skalarprodukt: Schreiben sie das Skalarprodukt $\nabla \cdot \mathbf{v}$ in Komponentenschreibweise.
2. Mechanik: Was ist $\mathbf{v} \cdot \nabla\psi$? Physikalische Bedeutung des Terms
3. Mechanik: Was ist Φ^ψ? Physikalische Bedeutung des Terms
4. Hydromechanik: Komponentenschreibweise $\nabla \cdot (\mathbf{v}\psi)$
5. Hydromechanik: Komponentenschreibweise $\nabla \cdot (\mathbf{D}^\psi \nabla\psi)$
6. Analytik: Prüfen Sie die Gültigkeit einer der Lösungen für die Diffusionsgleichung: (20), (21), (22) (siehe Vorlesung 5 $>>$ HyBHW-S2-01-V05).
7. Analytik: Stellen Sie die ausgewählte analytische Lösung für die 1-D parabolische Differentialgleichung unter Verwend der Übung EX06-parabolische-gleichung-1D.py dar. Ergänzen Sie Ihren Namen oder Matrikelnummer mit dem Befehl `plt.title("Name oder Matrikelnummer")`.
8. Numerik: Darstellung der numerischen Lösung (explizite FDM) für die 1-D parabolische Differentialgleichung (EX08-fdm-explicit-python). Produzieren Sie eine stabile und instabile Lösung.
9. Numerik: Darstellung der numerischen Lösung (implizite FDM) für die 1-D parabolische Differentialgleichung (EX09-fdm-implicit-python). Produzieren Sie die stationäre Lösung.

Hausaufgaben: Hydroinformatik

Beispiel: Aufgaben 7-9

- zum Internet-Repository gehen (Webseite)
- Python-File editieren (Matrikel-Nummer oder Name)
- Programme zum Rechnen und Darstellen ausführen
- Ergebnis (Abbildung) in die Hausaufgaben einfügen

TECHNISCHE UNIVERSITÄT DRESDEN

git | GitHub Praxis

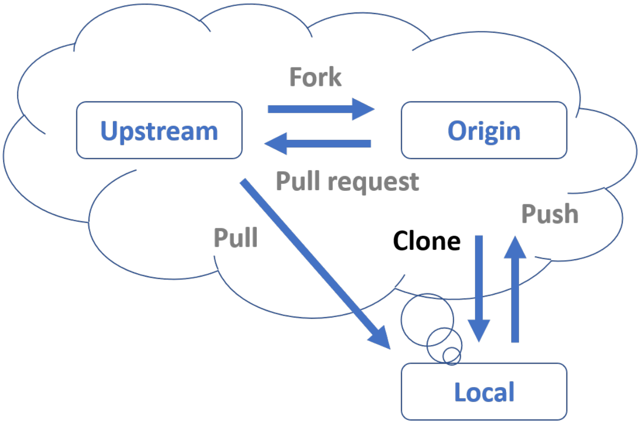
TECHNISCHE UNIVERSITÄT DRESDEN

GitHub

Übung: git: Installieren

1. Teaching Tutorial: Ausführlichere Beschreibung zum Nachlesen (Kapitel 1.1)
2. git Installation: Webpage
3. git Version

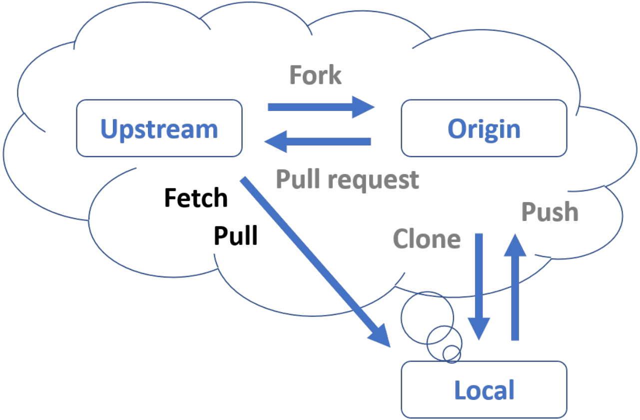
```
git -v
```



1. Quelle auswählen: HYDROINFORMATIK-I
2. Code: Pfad für’s Clonen kopieren
3. geeignetes Verzeichnis auf eigenem Rechner auswählen (root)
4. Quelle clonen:

```
git clone https://github.com/OlafKolditz/HYDROINFORMATIK-I.git
```

Übung: git: Repo auschecken



1. ins richtige Verzeichnis gehen (HYDROINFORMATIK-I)
2. fetch : Repo anfragen: Gibt's was Neues?
3. pull : Änderungen in lokalem Repo runterladen

```
git fetch --all
git pull
```

GitHub

Arbeiten mit verschiedenen Repos

Dozent: OlafKolditz

- Template für das Jupyter Notebook
- Repo auschecken (git clone / git pull) in das lokale Repo auf eigenem Rechner kopieren
- oder einzelne Datei herunterladen
- Datei in eigenes lokales Repo übernehmen $\Rightarrow$

Mein Repo (KolditzTUDD)

- $\Rightarrow$ Jupyter Notebook lokal auf eigenem Rechner bearbeiten
- Eigenes Internet Repo aktualisieren: git push
- ... Wiederholen (hierfür mit branching arbeiten)
- wenn das Notebook fertig ist, dann einen merge request in das Dozenten Notebook anmelden

TECHNISCHE UNIVERSITÄT DRESDEN

GitHub

Arbeiten mit eigenem Repo

- ⇒ Jupyter Notebook lokal auf eigenem Rechner bearbeiten
- Eigenes Internet Repo aktualisieren: git push
- … Wiederholen (hierfür mit branching arbeiten)
- (wenn das Notebook fertig ist, dann einen merge request in das Dozenten Notebook anmelden)

```
//eigenes Repo lokal aktualisieren
git fetch --all
git pull
//branch - Name 'devs' kann frei gewaehlt werden
git checkout -b devs
git branch
//neue Datei hinzufuegen
git add hydroinformatik-jupyter-notebook.ipynb
//commit
git commit -m "Add mein ipynb zum Internet Repo"
//push
git push origin devs
//status pruefen
git status
```