

# Modellierung von Hydrosystemen - SoSe 2024

## BHYWI-22-B2-05-T15a: Finite-Differenzen-Methode: Übungen

Olaf Kolditz, Lars Bilke, Karsten Rink, Haibing Shao, Erik Nixdorf

<sup>1</sup>Helmholtz Centre for Environmental Research – UFZ, Leipzig

<sup>2</sup>Technische Universität Dresden – TUD, Dresden

<sup>3</sup>Center for Advanced Water Research – CAWR

<sup>4</sup>TUBAF-UFZ Center for Environmental Geosciences – C-EGS, Freiberg / Leipzig

<sup>4</sup>Bundesanstalt für Geowissenschaften und Rohstoffe – BGR, Hannover / Cottbus

Dresden, 21.06.2024

# Zeitplan: Modellierung von Hydrosystemen: Zweiter Block (B2)

Sommersemester 2024: BHYWI-22-B2

Datum	B2	Thema	Format
12.04.2024	B2-T1.0	Einführung in die Veranstaltung (B2) (Kolditz)	HSZ/401
21.06.2024	B2-T1.1	Hydromechanik und Numerische Methoden (Kolditz)	HSZ/403
21.06.2024	B2-T1.2	Grundwasserhydraulik und Prinzipbeispiel (Kolditz)	HSZ/403
21.06.2024	B2-T1.3	Finite-Differenzen-Methode: Explizit (Kolditz)	HSZ/403
21.06.2024	B2-T1.4	Finite-Differenzen-Methode: Implizit (Kolditz)	HSZ/403
28.06.2024	B2-T4.1	Virtuelle VISLAB Tour - Vorlesung (Rink/Bilke)	Online
28.06.2024	B2-T4.2	Virtuelle VISLAB Tour - Demo (Rink/Bilke)	Online
05.07.2024	B2-T3.1	Stofftransport in Hydrosystemen (Shao/Chen)	HSZ/403
05.07.2024	B2-T3.2	Stofftransport in Hydrosystemen (Shao/Chen)	HSZ/403
05.07.2024	B2-T3.3	Stofftransport in Hydrosystemen (Shao/Chen)	HSZ/403
12.07.2024	B2-T2.1	Regionale Grundwassersysteme (Nixdorf)	HSZ/403
12.07.2024	B2-T2.2	Regionale Grundwassersysteme (Nixdorf)	HSZ/403
12.07.2024	B2-T2.3	Regionale Grundwassersysteme (Nixdorf): Übung	HSZ/403
19.07.2024	B2-T1.6	Zusammenfassung der Veranstaltung Numerik (Kolditz)	HSZ/403
19.07.2024	B2-T1.7	Zusammenfassung der Veranstaltung (Hartmann/Kolditz)	HSZ/403
19.07.2024	B2-T1.8	Vorbereitung Klausur (Hartmann/Kolditz)	HSZ/403

# Übungen

- Werkzeuge
- FDM Rechteckaquifer (explizites und implizites Verfahren)
  - ParaView
  - Python
- FDM Selke-Einzugsgebiet

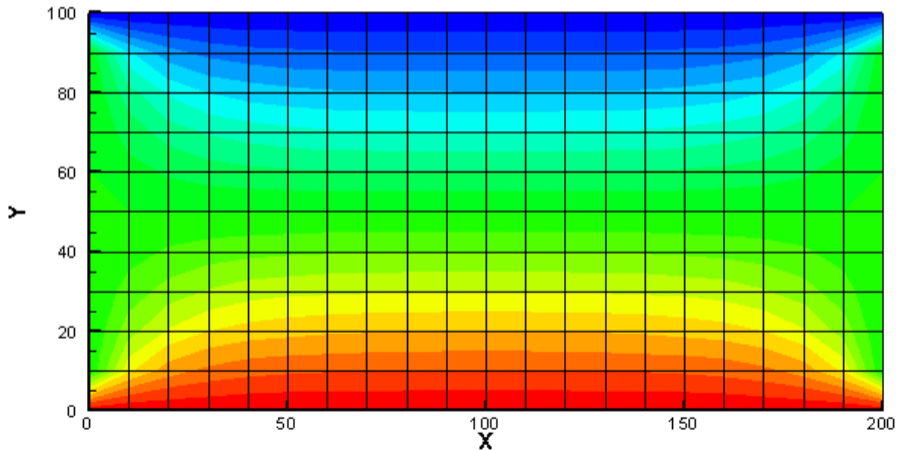
GitHub-Repository für die Übungen:  
<https://github.com/OlafKolditz/HYDROSYSTEMS>



- ▶ Editor (Notepad++)
  - ▶ Compiler (Qt5/MinGW73)
  - ▶ Workflows (Python, (Jupyter))
  - ▶ Ergebnisdarstellung
- 
- ▶ Python
  - ▶ ParaView
  - ▶ OpenGeoSys DataExplorer (VISLAB Vorlesung letzte Woche)

# FDM Übung: Rechteckaquifer (BHYWI-22-E2\_FDM-Rechteck)

## Aufgabenstellung



# FDM Übung: Rechteckaquifer

Rechenprogramm: main.cpp

```
C:\User\02_TUD\23_SoSe2020\BHYWI-22\EXERCISES\BHYWI-22-E2_QAD-Rechteck\main.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
main.cpp x
91 out_file << "VARIABLES = X, Y, H" << std::endl;
92 for(int t=0;t<tn;t++)
93 {
94     start = clock();
95     //-----
96     //time step
97     for(int j=0;j<jy;j++)
98     {
99         nn = j*ix;
100        for(int i=0;i<ix;i++)
101        {
102            n = nn+i;
103            if(IsBCNode(n,bc_nodes))
104                continue;
105            //if(IsNodeInactive(n,nodes_inactive))
106                //continue;
107            u_new[n] = u[n] \
108                + Rf/s0*dt/dx2 * (u[n+1]-2*u[n]+u[n-1]) \
109                + Rf/s0*dt/dy2 * (u[(j+1)*ix+i]-2*u[n]+u[(j-1)*ix+i]) \
110                + Q/s0;
111        }
112    }
113    //-----
114    end = clock();
115    // check steady state
116    //-----
117    //save time step
118    for(int j=0;j<jy;j++)
```

# FDM Übung: Rechteckaquifer

Rechenprogramm: main.cpp Kompilieren und Ausführen

```
Qt 5.12.0 for Desktop (MinGW 7.3.0 64 bit)
Setting up environment for Qt usage...
C:\Qt\5.12.0\mingw73_64>cd C:\User\02_TUD\23_SoSe2020\BHYWI-22\EXERCISES\BHYWI-22-E2_QAD-Rechteck
C:\User\02_TUD\23_SoSe2020\BHYWI-22\EXERCISES\BHYWI-22-E2_QAD-Rechteck>g++ main.cpp
C:\User\02_TUD\23_SoSe2020\BHYWI-22\EXERCISES\BHYWI-22-E2_QAD-Rechteck>a.exe
C:\User\02_TUD\23_SoSe2020\BHYWI-22\EXERCISES\BHYWI-22-E2_QAD-Rechteck>_
```

- ▶ Compiler (Qt5/MinGW73): `g++ main.cpp`
- ▶ Programm ausführen: `a.exe`
- ▶ Ergebnisse: `out.dat`

# ParaView

- BHYWI-22-E2\_FDM-explizit-Rechteck-paraview

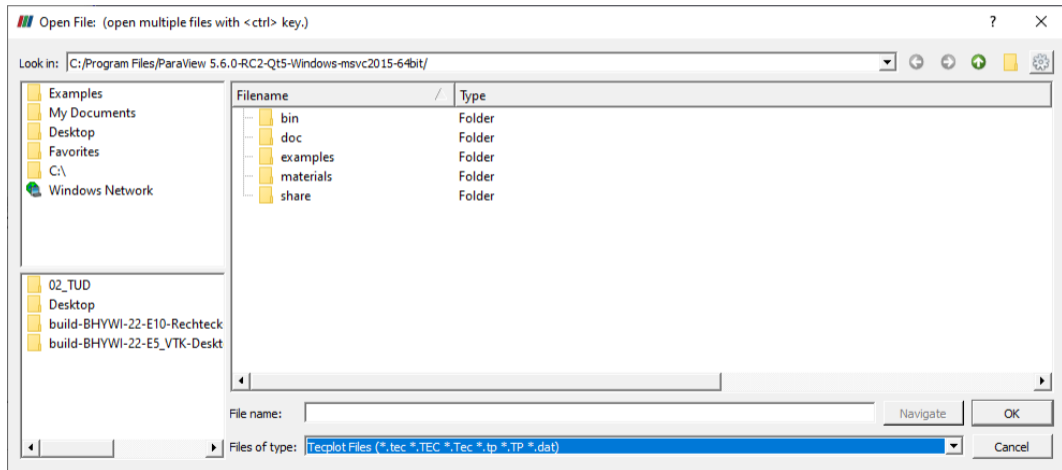
GitHub-Repository für die Übungen:

<https://github.com/OlafKolditz/HYDROSYSTEMS>



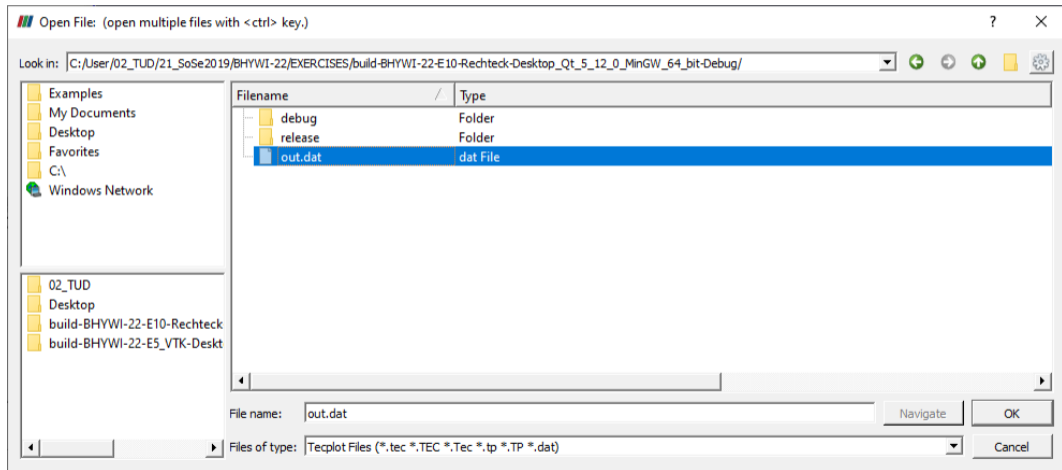
# FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



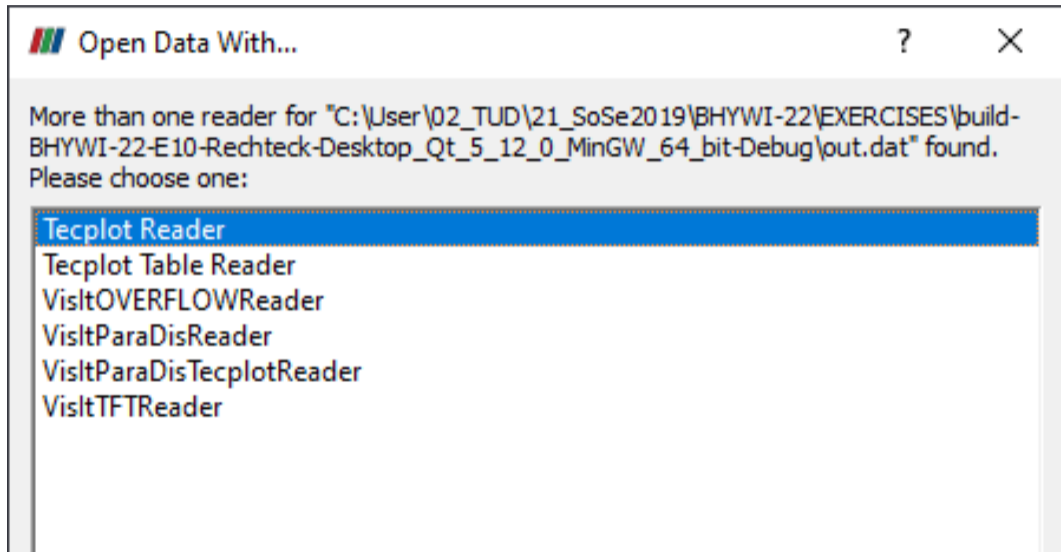
# FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



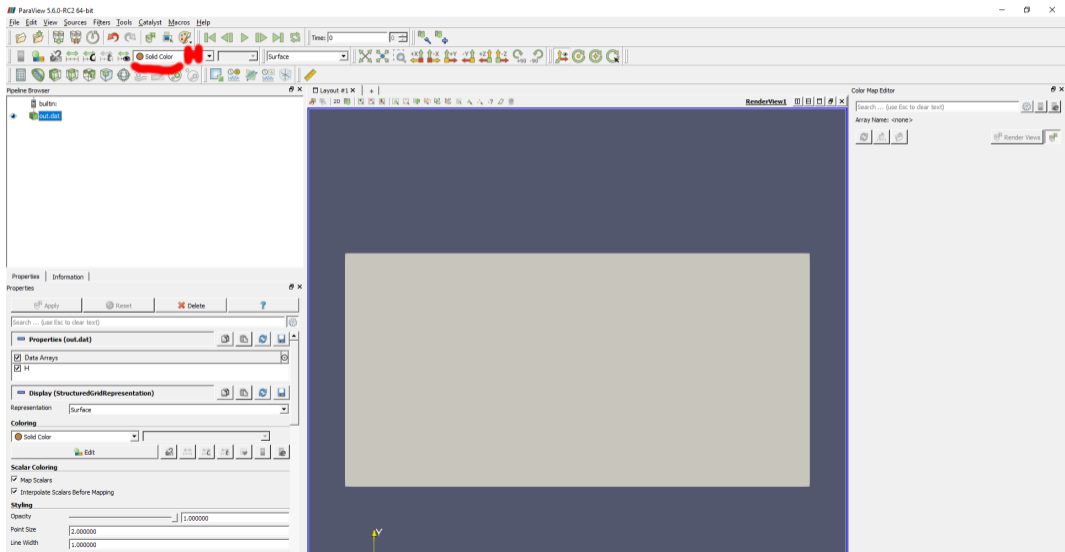
# FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



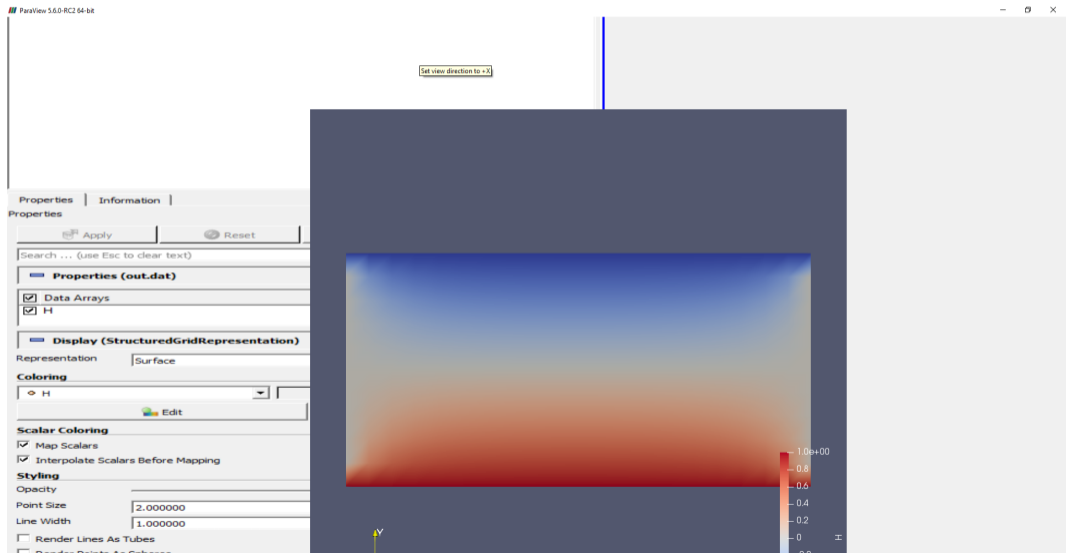
# FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



# FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



# Python

- BHYWI-22-E2\_FDM-explizit-Rechteck-python

GitHub-Repository für die Übungen:

<https://github.com/OlafKolditz/HYDROSYSTEMS>

# Python

- BHYWI-22-E3\_FDM-implizit-Rechteck-python

GitHub-Repository für die Übungen:

<https://github.com/OlafKolditz/HYDROSYSTEMS>

# 2D implizite FDM - die main function

```
1 #include <iostream>
2 #include "fdm.h"
3 #include <time.h>
4 extern void Gauss(double*,double*,double*,int);
5 int main(int argc, char *argv[])
6 {
7     //-----
8     FDM* fdm = new FDM();
9     fdm->SetInitialConditions();
10    fdm->SetBoundaryConditions();
11    //-----
12    int tn = 2;
13    for(int t=0;t<tn;t++)
14    {
15        fdm->AssembleEquationSystem();
16        Gauss(fdm->matrix,fdm->vecb,fdm->vecx,fdm->IJ);
17        fdm->SaveTimeStep();
18        fdm->OutputResults(t);
19    }
20    //-----
21    fdm->out_file.close();
22    return 0;
23 }
```

Listing 1: OOP main function



# Python

- BHYWI-22-E5\_FDM-Selke-python

GitHub-Repository für die Übungen:

<https://github.com/OlafKolditz/HYDROSYSTEMS>

# 2D explizite FDM - die main function

```
1 #include <iostream>
2 #include <time.h>
3 #include "fdm.h"
4 int main(int argc, char *argv[])
5 {
6     clock_t start, end;
7     start = clock();
8     //-----
9     FDM* fdm = new FDM();
10    fdm->SetActiveNodes();
11    fdm->SetInactiveNodes();
12    fdm->SetInitialConditions();
13    fdm->SetBoundaryConditions();
14    //-----
15    int tn = 10;
16    for(int t=0;t<tn;t++)
17    {
18        fdm->RunTimeStep();
19        fdm->SaveTimeStep();
20        if(t==tn-1)
21            fdm->OutputResults(t);
22        //    if((t%10)==0)
23        //        fdm->OutputResultsVTK(t);
24    }
25    //-----
26    end = clock();
27    return 0;
28 }
```

Listing 2: OOP main function

```
1 #include <iostream>
2 #include <time.h>
3 #include "fdm.h"
4 int main(int argc, char *argv[])
5 {
6     //-----
7     FDM* fdm = new FDM();
8     fdm->SetActiveNodes();
9     fdm->SetInactiveNodes();
10    fdm->SetInitialConditions();
11    fdm->SetBoundaryConditions();
12    //-----
13    int tn = 1000;
14    for(int t=0;t<tn;t++)
15    {
16        fdm->RunTimeStep();
17        fdm->SaveTimeStep();
18        if(t==tn-1)
19            fdm->OutputResults(t);
20    }
21    //-----
22    return 0;
23 }
```

Listing 3: Explizites Verfahren

```
1 #include <iostream>
2 #include "fdm.h"
3 #include <time.h>
4 extern void Gauss(double*,double*,double*,int);
5 int main(int argc, char *argv[])
6 {
7     //-----
8     FDM* fdm = new FDM();
9     fdm->SetInitialConditions();
10    fdm->SetBoundaryConditions();
11    //-----
12    int tn = 2;
13    for(int t=0;t<tn;t++)
14    {
15        fdm->AssembleEquationSystem();
16        Gauss(fdm->matrix,fdm->vecb,fdm->vecx,fdm->IJ);
17        fdm->SaveTimeStep();
18        fdm->OutputResults(t);
19    }
20    //-----
21    fdm->out_file.close();
22    return 0;
23 }
```

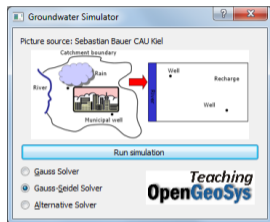
Listing 4: Implizites Verfahren

# Numerische Verfahren

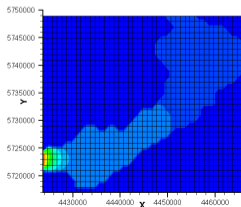
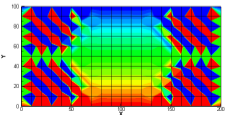
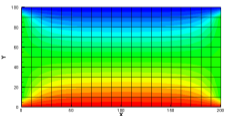
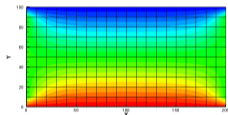
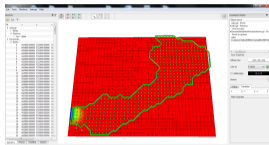
- ...

# Übersicht: Numerische Verfahren

## explizite FDM



## implizite FDM



- Pro / Cons
- FDM: einfache Implementierung, starre Geometrien
- FEM: schwieriger zu implementieren (heute), flexible Geometrien

## FEM

