Modellierung von Hydrosystemen - SoSe 2022

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

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

[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

[4]Bundesanstalt für Geowissenschaften und Rohstoffe – BGR, Hannover / Berlin

Dresden, 01.07.2022

TECHNISCHE UNIVERSITÄT DRESDEN

Zeitplan: Modellierung von Hydrosystemen: Zweiter Block (B2)

Sommersemester 2022: BHYWI-22-B2

Datum	B2	Thema	Format
27.05.2022	B2-T1.0	Einführung in die Veranstaltung (B2) (Kolditz)	Online
27.05.2022	B2-T1.1	Hydromechanik und Numerische Methoden (Kolditz)	Online
27.05.2022	B2-T1.2	Grundwasserhydraulik und Prinzipbeispiel (Kolditz)	Online
03.06.2022	B2-T3.1	Stofftransport in Hydrosystemen (Shao)	HSZ/403
03.06.2022	B2-T3.2	Stofftransport in Hydrosystemen (Shao)	HSZ/403
10.06.2022	–	Vorlesungsfrei (Pfingsten)	
17.06.2022	B2-T2.1	Regionale Grundwassersysteme (Nixdorf)	HSZ/403
17.06.2022	B2-T2.2	Regionale Grundwassersysteme (Nixdorf)	HSZ/403
17.06.2022	B2-T2.3	Regionale Grundwassersysteme (Nixdorf): Übung	HSZ/403
24.06.2022	B2-T4.1	Virtuelle VISLAB Tour - Vorlesung (Rink/Bilke)	Online
24.06.2022	B2-T4.2	Virtuelle VISLAB Tour - Demo (Rink/Bilke)	Online
01.07.2022	B2-T1.3	Finite-Differenzen-Methode: Explizit (Kolditz)	HSZ/403
01.07.2022	B2-T1.4	Finite-Differenzen-Methode: Implizit (Kolditz)	HSZ/403
01.07.2022	B2-T1.5	Finite-Differenzen-Methode: Übungen (Kolditz)	HSZ/403
08.07.2022	B2-T3.3	Stofftransport in Hydrosystemen (Shao)	GER/38
08.07.2022	B2-T3.4	Stofftransport in Hydrosystemen (Shao)	GER/38
15.07.2022	B2-T1.6	Zusammenfassung der Veranstaltung (Hartmann/Kolditz)	HSZ/403
15.07.2022	B2-T1.7	Vorbereitung Klausur (Hartmann/Kolditz)	HSZ/403

TECHNISCHE UNIVERSITÄT DRESDEN

Übungen

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

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

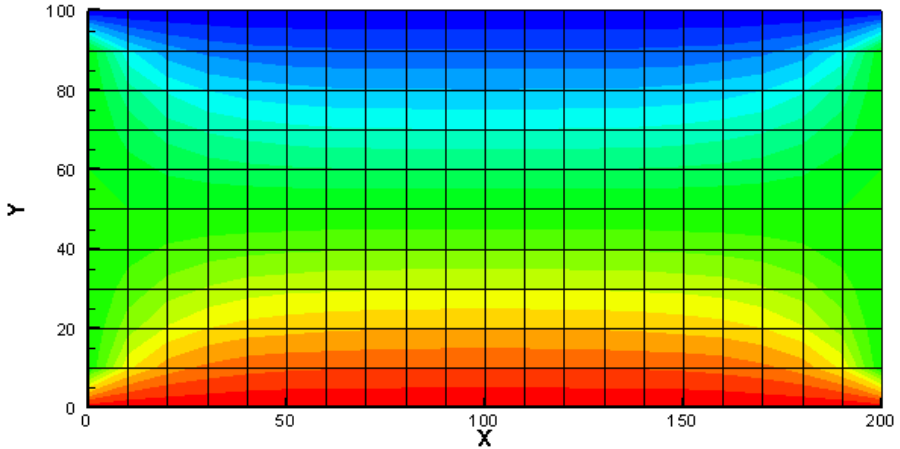
TECHNISCHE UNIVERSITÄT DRESDEN

- 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++

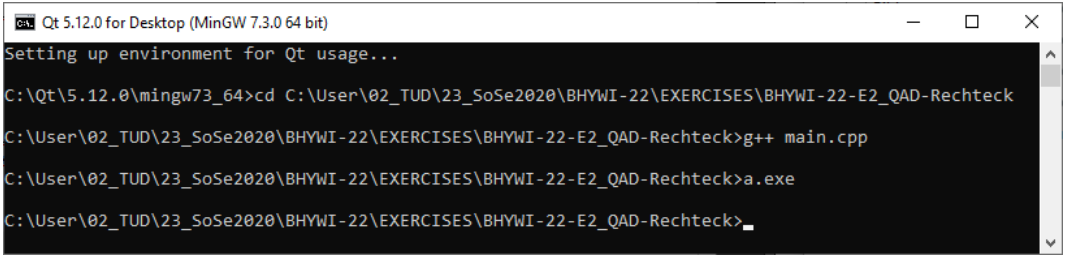
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?

main.cpp

```
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                + Kf/S0*dt/dx2 * (u[n+1]-2*u[n]+u[n-1]) \
109                + Kf/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



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

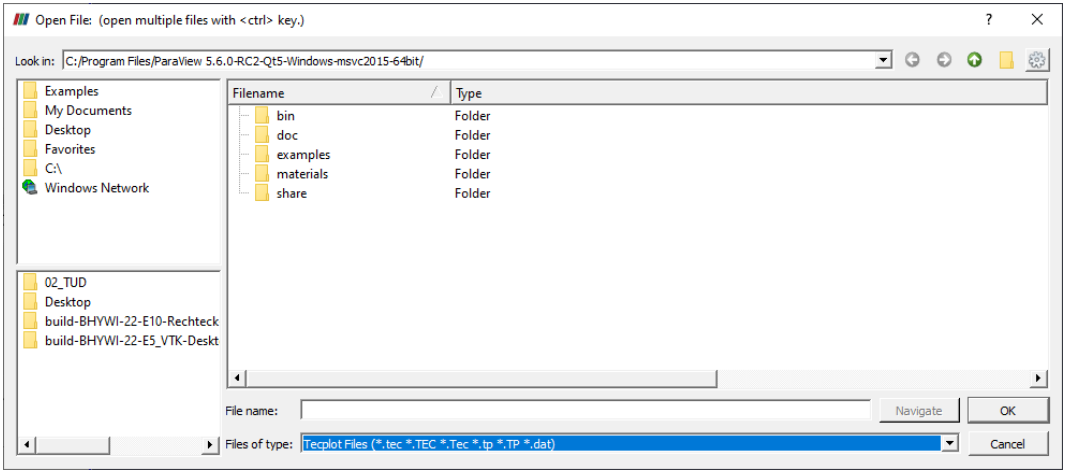
ParaView

- BHYWI-22-E2_FDM-explizit-Rechteck-paraview

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

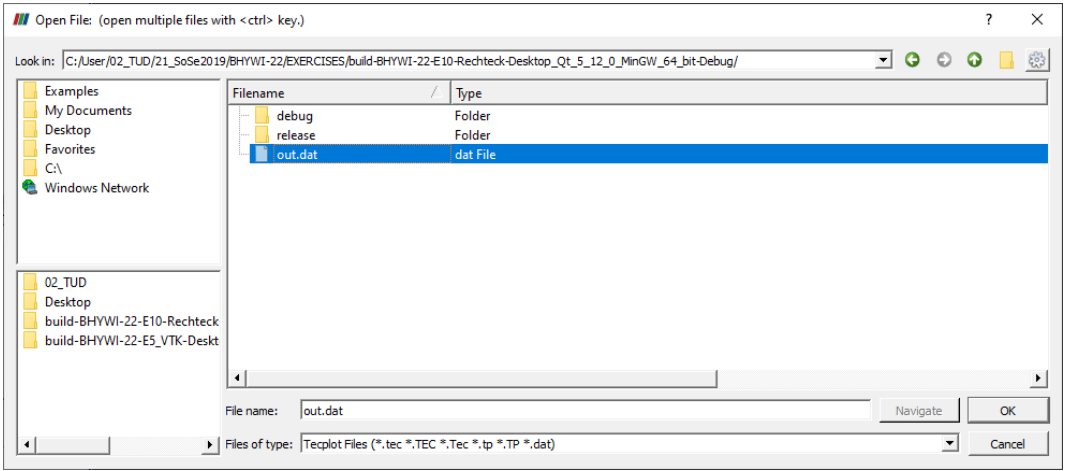
FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



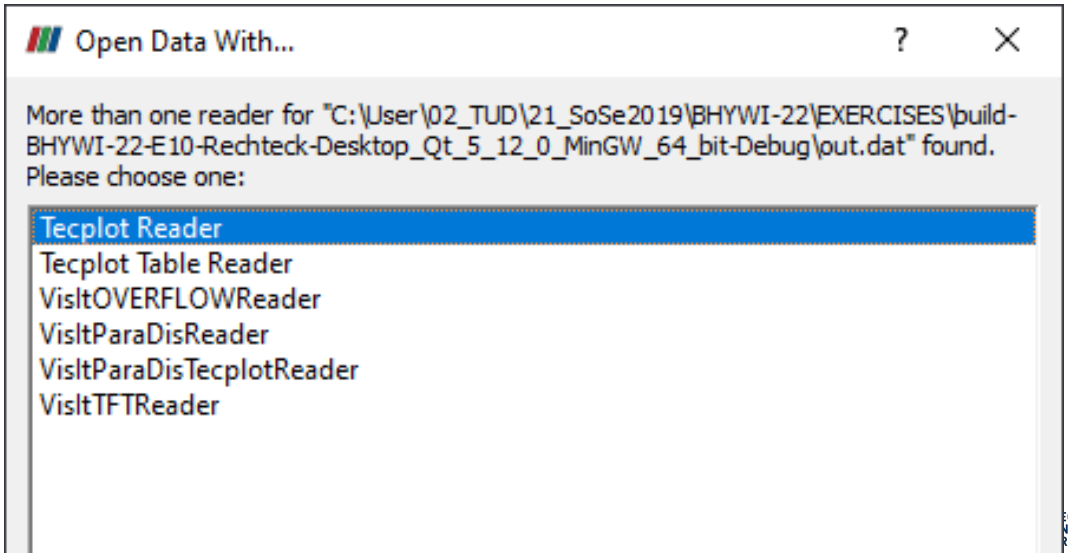
FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



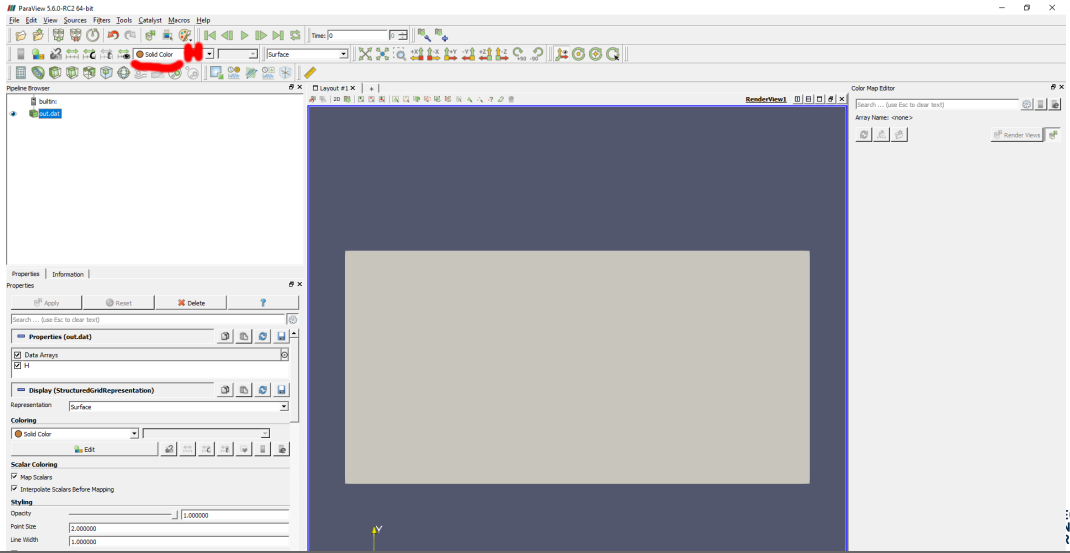
FDM Übung: Rechteckaquifer

Ergebnisse darstellen mit ParaView



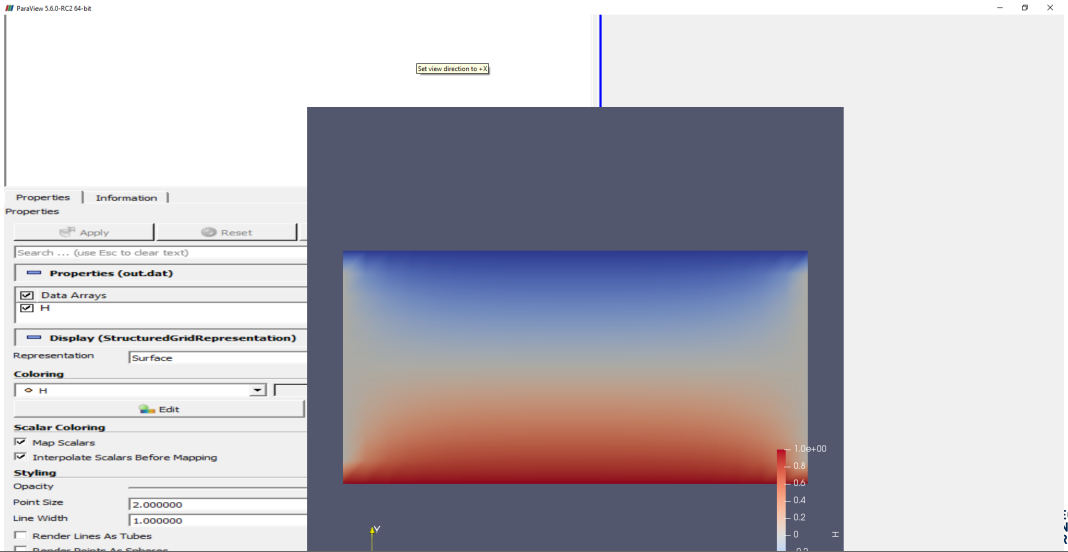
FDM Übung: Rechteckaquifer

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

```
#include <iostream>
#include "fdm.h"
#include <time.h>
extern void Gauss(double*,double*,double*,int);
int main(int argc, char *argv[])
{
  //-----------------------------------------------
  FDM* fdm = new FDM();
  fdm->SetInitialConditions();
  fdm->SetBoundaryConditions();
  //-----------------------------------------------
  int tn = 2;
  for(int t=0;t<tn;t++)
  {
    fdm->AssembleEquationSystem();
    Gauss(fdm->matrix,fdm->vecb,fdm->vecx,fdm->IJ);
    fdm->SaveTimeStep();
    fdm->OutputResults(t);
  }
  //-----------------------------------------------
  fdm->out_file.close();
  return 0;
}
```

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

```cpp
#include <iostream>
#include <time.h>
#include "fdm.h"
int main(int argc, char *argv[])
{
  clock_t start, end;
  start = clock();
  //-----------------------------------------------
  FDM* fdm = new FDM();
  fdm->SetActiveNodes();
  fdm->SetInactiveNodes();
  fdm->SetInitialConditions();
  fdm->SetBoundaryConditions();
  //-----------------------------------------------
  int tn = 10;
  for(int t=0;t<tn;t++)
  {
    fdm->RunTimeStep();
    fdm->SaveTimeStep();
  if(t==tn-1)
    fdm->OutputResults(t);
//    if((t%10)==0)
//      fdm->OutputResultsVTK(t);
  }
  //-----------------------------------------------
  end = clock();
  return 0;
}
```

Listing 2: OOP main function

FDM Verfahren - OOP

```cpp
#include <iostream>
#include <time.h>
#include "fdm.h"
int main(int argc, char *argv[])
{
  //-----------------------------------------------
  FDM* fdm = new FDM();
  fdm->SetActiveNodes();
  fdm->SetInactiveNodes();
  fdm->SetInitialConditions();
  fdm->SetBoundaryConditions();
  //-----------------------------------------------
  int tn = 1000;
  for(int t=0;t<tn;t++)
  {
    fdm->RunTimeStep();
    fdm->SaveTimeStep();
  if(t==tn-1)
    fdm->OutputResults(t);
  }
  //-----------------------------------------------
  return 0;
}
```

Listing 3: Explizites Verfahren

```cpp
#include <iostream>
#include "fdm.h"
#include <time.h>
extern void Gauss(double*,double*,double*,int);
int main(int argc, char *argv[])
{
  //-----------------------------------------------
  FDM* fdm = new FDM();
  fdm->SetInitialConditions();
  fdm->SetBoundaryConditions();
  //-----------------------------------------------
  int tn = 2;
  for(int t=0;t<tn;t++)
  {
    fdm->AssembleEquationSystem();
    Gauss(fdm->matrix,fdm->vecb,fdm->vecx,fdm->IJ);
    fdm->SaveTimeStep();
    fdm->OutputResults(t);
  }
  //-----------------------------------------------
  fdm->out_file.close();
  return 0;
}
```

Listing 4: Implizites Verfahren

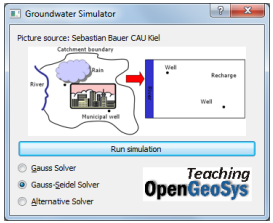
TECHNISCHE UNIVERSITÄT DRESDEN

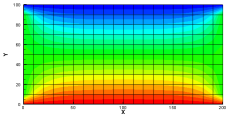
Numerische Verfahren

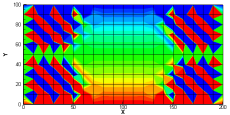
- ...

Übersicht: Numerische Verfahren

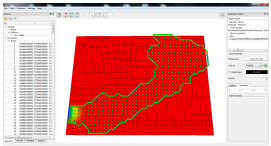
explizite FDM

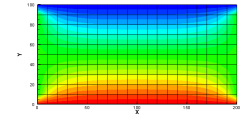


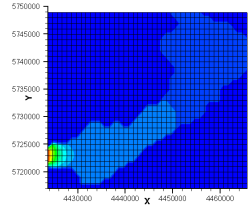




implizite FDM







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

FEM