Modellierung von Hydrosystemen

"Numerische und daten-basierte Methoden"

BHYWI-22-15 @ 2018

implizite Finite-Differenzen-Methode

Selke-Modell

Olaf Kolditz

*Helmholtz Centre for Environmental Research – UFZ

[1]Technische Universität Dresden – TUDD

[2]Centre for Advanced Water Research – CAWR

06.07.2018 - Dresden

TECHNISCHE UNIVERSITÄT DRESDEN

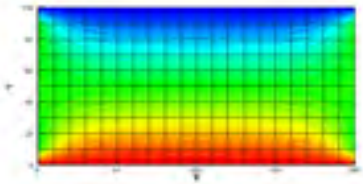
Fahrplan ...

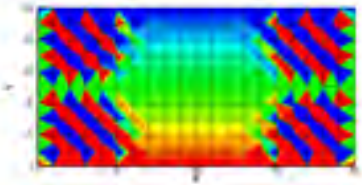
- (void FDM::OutputResultsVTK(int t))
- implizite FDM (2D)
- Gleichungssysteme

TECHNISCHE UNIVERSITÄT DRESDEN

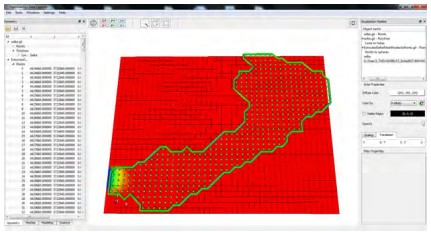
Konzept

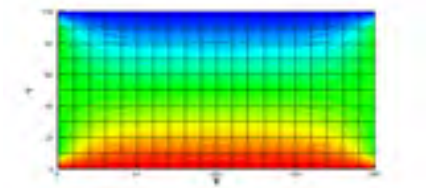
explizite FDM

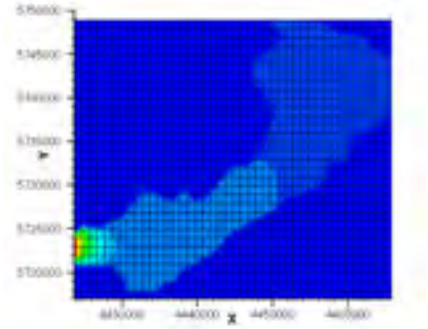




implizite FDM

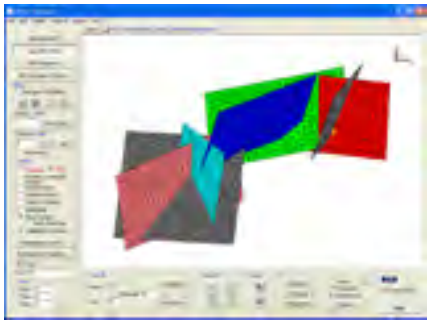






...

FEM



Selke Einzugsgebiet

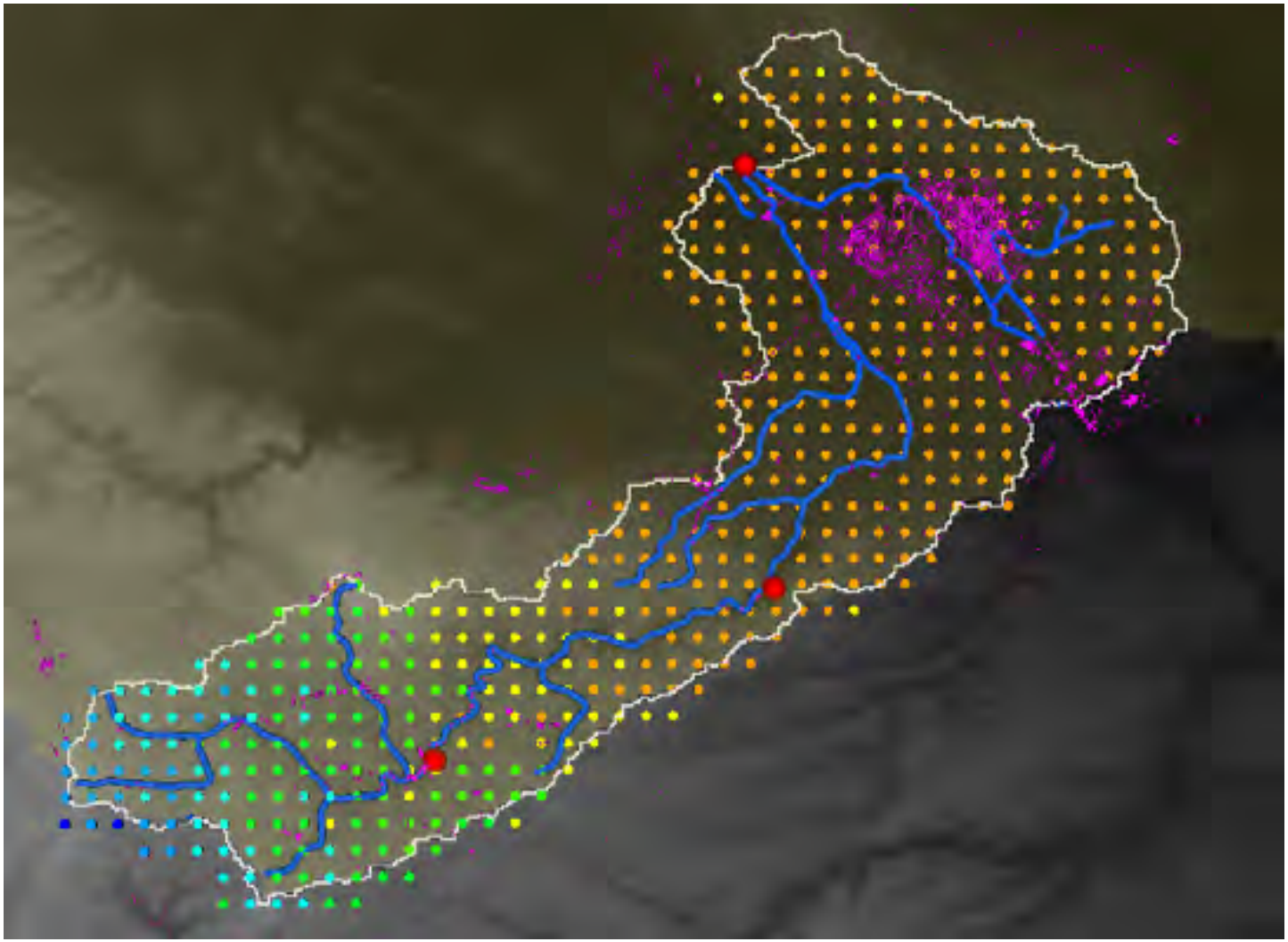
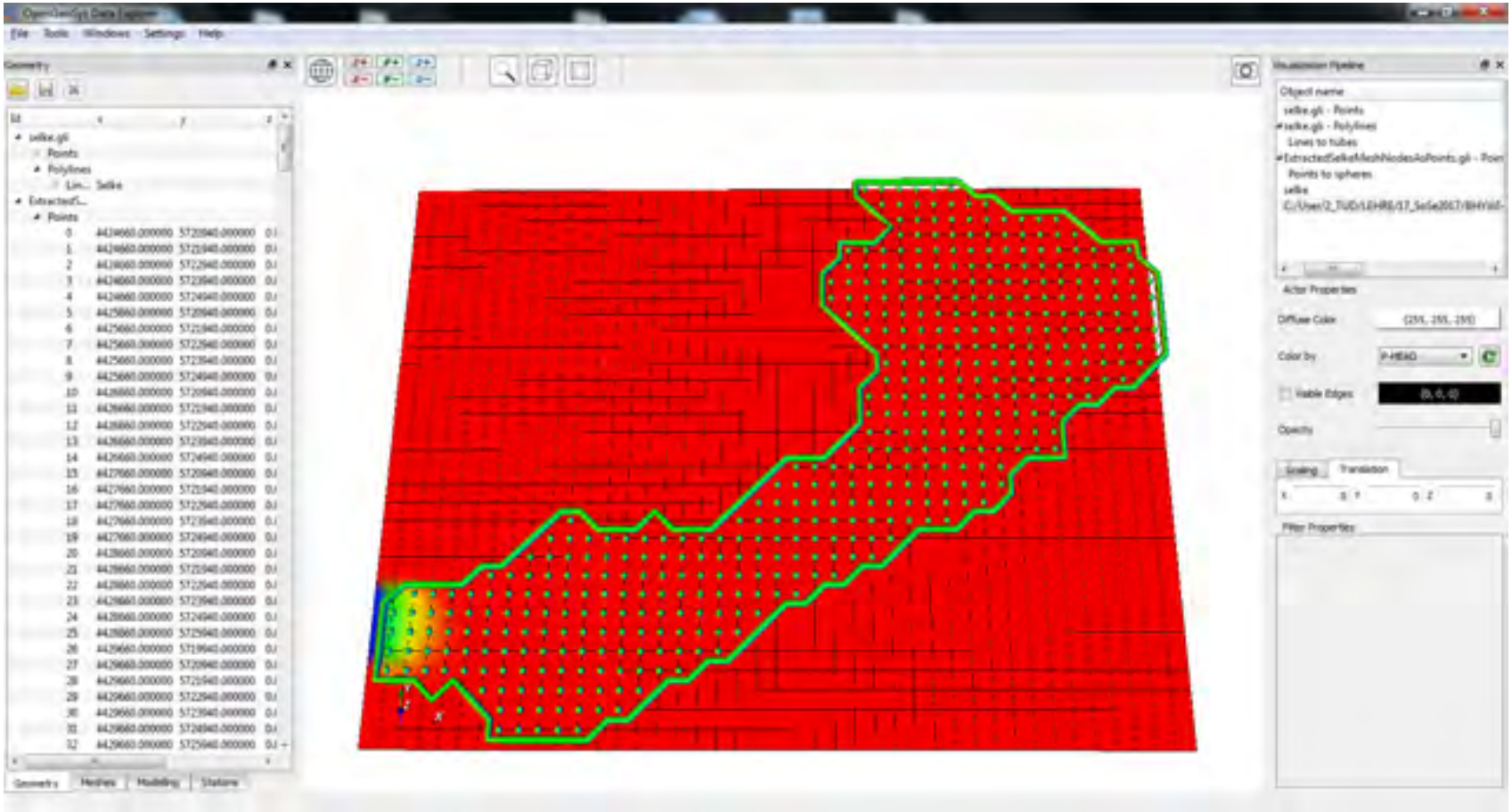


Figure: Untersuchungsgebiet - Selke

TECHNISCHE UNIVERSITÄT DRESDEN

Selke Einzugsgebiet



2D implizite FDM

siehe auch Abschn. 4.2 Hydroinformatik II

- Auswertung der Ableitungen zum neuen Zeitpunkt t^{n+1}

$$\left[\frac{\partial^2 h}{\partial x^2}\right]_{i,j}^{n+1} \approx \frac{h_{i-1,j}^{n+1} - 2h_{i,j}^{n+1} + h_{i+1,j}^{n+1}}{\Delta x^2} \tag{1}$$

$$\left[\frac{\partial^2 u}{\partial y^2}\right]_{i,j}^{n+1} \approx \frac{h_{i,j-1}^{n+1} - 2h_{i,j}^{n+1} + h_{i,j+1}^{n+1}}{\Delta y^2} \tag{2}$$

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

- Differenzen-Schema

$$S_{i,j}\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t} \tag{3}$$
$$-K_{i,j}^{x}\frac{u_{i+1,j}^{n+1}-2u_{i,j}^{n+1}+u_{i-1,j}^{n+1}}{\Delta x^2} \quad - \quad K_{i,j}^{y}\frac{u_{i,j+1}^{n+1}-2u_{i,j}^{n+1}+u_{i,j-1}^{n+1}}{\Delta y^2} = Q_{i,j}$$

2D implizite FDM

- Gleichungssystem

$$
\begin{aligned}
& \left(\frac{S}{\Delta t} + 2\frac{K^x}{\Delta x^2} + 2\frac{K^y}{\Delta y^2}\right) u_{i,j}^{n+1} \\
- & \left(\frac{K^x}{\Delta x^2}\right)(u_{i-1,j}^{n+1} + u_{i+1,j}^{n+1}) - \left(\frac{K^y}{\Delta y^2}\right)(u_{i,j-1}^{n+1} + u_{i,j+1}^{n+1}) \\
= & \frac{S}{\Delta t} u_{i,j}^{n} + Q_{i,j} \qquad (4)
\end{aligned}
$$

2D implizite FDM

Wir vereinfachen die Gleichung (4), indem wir für den Moment annehmen, dass $K^x = K^y = K$ (Isotropie) und $\Delta x = \Delta y = \Delta l$ (gleichförmige Diskretisierung). Die Multiplikation mit $\Delta t / S$ ergibt dann folgende Beziehung.

$$\begin{aligned} & \left(1 + 4\frac{K\Delta t}{S\Delta l^2}\right) u_{i,j}^{n+1} \\ - & \left(\frac{K\Delta t}{S\Delta l^2}\right) (u_{i-1,j}^{n+1} + u_{i+1,j}^{n+1} + u_{i,j-1}^{n+1} + u_{i,j+1}^{n+1}) \\ = & \; u_{i,j}^{n} + \frac{\Delta t}{S} Q_{i,j} \end{aligned} \tag{5}$$

K : Vergleichen Sie die Beziehung (5) mit der Gleichung (4.10, Skript Hydroinformatik II).

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

Der Ausdruck $K/S = \alpha$ entspricht dem Diffusivitätskoeffizienten (Überprüfen sie dies anhand der Einheiten). Damit ist die Neumann-Zahl

$$Ne = \frac{K}{S}\frac{\Delta t}{\Delta l^2} \tag{6}$$

Nun versuchen wir anhand der Gleichung (5) die Struktur des zu lösenden Gleichungssystems zu beschreiben. Wir gehen wieder ganz genau so vor wie bei der 1D Diffusionsgleichung im Abschn. 4.2 (Hydroinformatik II).

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM - Gleichnungssystem

$$
\underbrace{\begin{bmatrix}
1+4Ne & -Ne & & & & & & & \\
-Ne & \dots & \dots & & & & & & \\
 & \dots & \dots & \dots & & & & & \\
 & & -Ne & 1+4Ne & & & & & \\
 & & & & 1+4Ne & -Ne & & & \\
 & & & & \dots & \dots & \dots & & \\
 & & & & & -Ne & 1+4Ne &
\end{bmatrix}}_{\mathbf{A}}
\underbrace{\begin{bmatrix} u_{0,0}^{n+1} \\ u_{1,0}^{n+1} \\ \dots \\ u_{I-1,0}^{n+1} \\ u_{0,1}^{n+1} \\ \dots \\ u_{I-1,J-1}^{n+1} \end{bmatrix}}_{\mathbf{x}}
=
\underbrace{\begin{bmatrix} u_{0,0}^{n} + b_{0,0} \\ u_{1,0}^{n} + b_{1,0} \\ \dots \\ u_{I-1,0}^{n} + b_{I-1,0} \\ u_{0,1}^{n} + b_{0,1} \\ \dots \\ u_{I-1,J-1}^{n} + b_{I-1,J-1} \end{bmatrix}}_{\mathbf{b}}
\tag{7}
$$

2D implizite FDM

Auch was die Programmierung betrifft, können wir auf unsere Erfahrungen in Hydroinformatik II aufbauen. Es gibt praktisch keinen Unterschied, ob wir es mit einem 1D oder 2D Problem zu tun haben. Wir müssen lediglich aufpassen, dass wir die Indizes richtig zählen.

Wir benutzen die Grundstruktur des objekt-orientierten Programms für das explizite FD Verfahren (Abschn. **??**). Die wesentlichen Unterschiede der impliziten zur expliziten FDM sind, dass wir ein Gleichungssystem aufbauen und lösen müssen.

TECHNISCHE UNIVERSITÄT DRESDEN

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;
}
```

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

Dennoch können wir erstaunlich viel wiederverwenden, bis auf

```
fdm->AssembleEquationSystem();
Gauss(fdm->matrix,fdm->vecb,fdm->vecx,fdm->IJ);
```

Der Gleichungslöser `Gauss` ist übrigens genau der gleiche, den wir schon für die Lösung des impliziten FD Verfahrens für die Diffusionsgleichung in Hydroinformatik II benutzt haben.

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

Der Reihe nach. Die Assemblierfunktion soll das Gleichungssystem (7) aufbauen. Vom Prinzip her das Gleiche wie beim 1D FD Verfahren:

- Die Hauptdiagonale bekommt den Wert $1 + 4Ne$,
- die Nebendiagonalen haben den Wert $-Ne$.

Dies lässt sich programmtechnisch recht einfach bewerkstelligen (sie erinnern sich, wie wir in einer Doppelschleife, die Hauptdiagonale herausfinden können)

2D implizite FDM

```
void FDM::AssembleEquationSystem()
{
  // Matrix entries
  for(i=0;i<IJ;i++)
  {
    vecb[i] = u[i];
    for(j=0;j<IJ;j++)
    {
      matrix[i*IJ+j] = 0.0;
      if(i==j)
        matrix[i*IJ+j] = 1. + 4.*Ne;
      else if(abs((i-j))==1)
        matrix[i*IJ+j] = - Ne;
    }
  }
  // Incorporate boundary conditions
  IncorporateBoundaryConditions();
  // Matrix output
  WriteEquationSystem();
}
```

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

Um die Assemblierfunktion zu testen bauen wir uns ein ganz einfaches Beispiel bestehend aus nur 9 Knoten (Abb. 2) - je einfacher, desto besser.

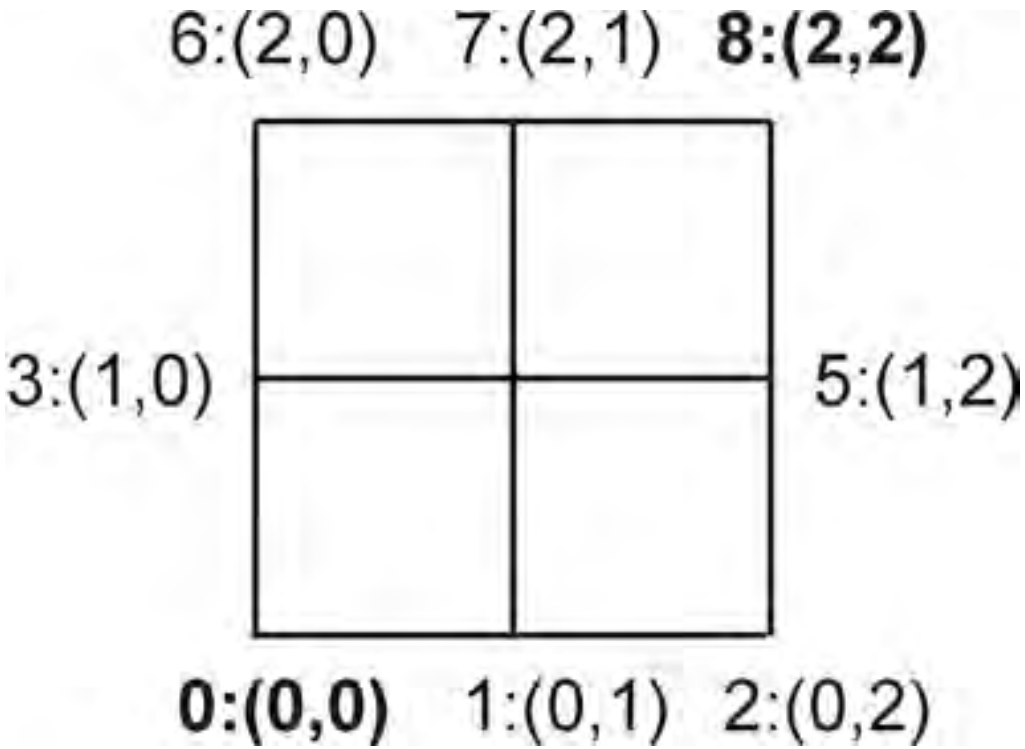


Figure: Testbeispiel

2D implizite FDM

Mit Hilfe der nützlichen Funktion `WriteEquationSystem()`können wir das Gleichungssystem in eine Datei schreiben, das Ergebnis passt.

```
2 -0.25 0 0 0 0 0 0 0
-0.25 2 -0.25 0 0 0 0 0 0
0 -0.25 2 -0.25 0 0 0 0 0
0 0 -0.25 2 -0.25 0 0 0 0
0 0 0 -0.25 2 -0.25 0 0 0
0 0 0 0 -0.25 2 -0.25 0 0
0 0 0 0 0 -0.25 2 -0.25 0
0 0 0 0 0 0 -0.25 2 -0.25
0 0 0 0 0 0 0 -0.25 2
```

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

Etwas kniffliger is es mit dem Einbauen der Randbedingungen. Wir erinnern uns, der Trick war eine Manipulation der Matrix und des RHS (right-hand-side) vectors, um den vorgegebenen Wert der Randbedingung zu erzwingen. Der Code zeigt das Beispiel für den Einbau einer Randbedingung im Knoten (also oben rechts in unseren kleinen Testbeispiel).

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

```
void FDM::IncorporateBoundaryConditions()
{
  size_t i_bc;
  int i_row, k;
  for(i_bc=0;i_bc<bc_nodes.size();i_bc++)
  {
    i_row = bc_nodes[i_bc];
    // Null off-diangonal entries of the related row and columns
    // Apply contribution to RHS by BC
    for(j=0;j<IJ;j++)
    {
      if(i_row == j)
        continue; // do not touch diagonals
      matrix[i_row*(IJ)+j] = 0.0; // NUll row
      k = j*(IJ)+i_row;
      // Apply contribution to RHS by BC
      vecb[j] -= matrix[k]*u[i_row];
      matrix[k] = 0.0; // Null column
    }
    // Apply Dirichlet BC
    vecb[i_row] = u[i_row]*matrix[i_row*(IJ)+i_row];
  }
}
```

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

Wir schreiben wieder das Gleichungssystem mit `WriteEquationSystem()`in eine Datei und schauen uns jeden Schritt genau an.

- Diagonalelemente werden nicht angefasst.
- Reihe zu Null setzen

```
2 0 0 0 0 0 0 0 0          b: 1
-0.25 2 -0.25 0 0 0 0 0 0 b: 0
0 -0.25 2 -0.25 0 0 0 0 0 b: 0
0 0 -0.25 2 -0.25 0 0 0 0 b: 0
0 0 0 -0.25 2 -0.25 0 0 0 b: 0
0 0 0 0 -0.25 2 -0.25 0 0 b: 0
0 0 0 0 0 -0.25 2 -0.25 0 b: 0
0 0 0 0 0 0 -0.25 2 -0.25 b: 0
0 0 0 0 0 0 0 0 2          b: -1
```

2D implizite FDM

- Rechte Seite manipulieren

```
2 0 0 0 0 0 0 0 0          b: 1
-0.25 2 -0.25 0 0 0 0 0 0 b: 0.25
0 -0.25 2 -0.25 0 0 0 0 0 b: 0
0 0 -0.25 2 -0.25 0 0 0 0 b: 0
0 0 0 -0.25 2 -0.25 0 0 0 b: 0
0 0 0 0 -0.25 2 -0.25 0 0 b: 0
0 0 0 0 0 -0.25 2 -0.25 0 b: 0
0 0 0 0 0 0 -0.25 2 -0.25 b: -0.25
0 0 0 0 0 0 0 0 2          b: -1
```

2D implizite FDM

- Spalte Null setzen

```
2 0 0 0 0 0 0 0 0          b: 1
0 2 -0.25 0 0 0 0 0 0      b: 0.25
0 -0.25 2 -0.25 0 0 0 0 0 b: 0
0 0 -0.25 2 -0.25 0 0 0 0 b: 0
0 0 0 -0.25 2 -0.25 0 0 0 b: 0
0 0 0 0 -0.25 2 -0.25 0 0 b: 0
0 0 0 0 0 -0.25 2 -0.25 0 b: 0
0 0 0 0 0 0 -0.25 2 0      b: -0.25
0 0 0 0 0 0 0 0 2          b: -1
```

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM

- Neumann Randbedingungen setzen

```
2 0 0 0 0 0 0 0 0          b:2
0 2 -0.25 0 0 0 0 0 0      b:0.25
0 -0.25 2 -0.25 0 0 0 0 0 b:0
0 0 -0.25 2 -0.25 0 0 0 0 b:0
0 0 0 -0.25 2 -0.25 0 0 0 b:0
0 0 0 0 -0.25 2 -0.25 0 0 b:0
0 0 0 0 0 -0.25 2 -0.25 0 b:0
0 0 0 0 0 0 -0.25 2 0      b:-0.25
0 0 0 0 0 0 0 0 2          b:-2
```

2D implizite FDM

Schließlich ergibt sich folgendes Gleichungssystem zur Lösung durch das Gauss-Verfahren noch mal richtig aufgeschrieben.

$$
\begin{aligned}
2h_1^{n+1} &= 2h_1^n \\
2h_2^{n+1} - 0.25h_3^{n+1} &= h_2^n + 0.25h_1^n \\
-0.25h_2 + 2h_3 - 0.5h_4 &= 0 \\
-0.25h_3 + 2h_4 - 0.5h_5 &= 0 \\
-0.25h_4 + 2h_5 - 0.5h_6 &= 0 \\
-0.25h_5 + 2h_6 - 0.5h_7 &= 0 \\
-0.25h_6 + 2h_7 - 0.5h_8 &= 0 \\
-0.25h_7 + 2h_8 &= -0.25 \\
2h_9^{n+1} &= h_9^n - 2
\end{aligned}
\tag{8}
$$

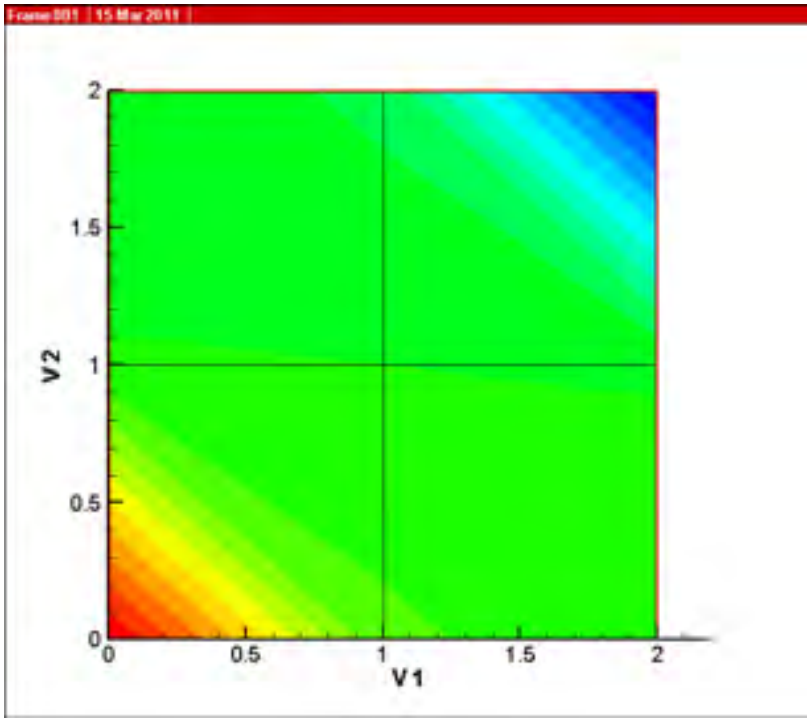
2D implizite FDM

Das geschriebene Ergebnisfile sieht dann folgendermaßen aus.

```
ZONE T="0.25", I=3, J=3, DATAPACKING=POINT
0 0 1
1 0 0.127016
2 0 0.016129
0 1 0.00201613
1 1 0
2 1 -0.00201613
0 2 -0.016129
1 2 -0.127016
2 2 -1
ZONE T="100.", I=3, J=3, DATAPACKING=POINT
0 0 1
1 0 0.267857
2 0 0.0714286
0 1 0.0178571
1 1 1.80718e-19
2 1 -0.0178571
0 2 -0.0714286
1 2 -0.267857
2 2 -1
```

TECHNISCHE UNIVERSITÄT DRESDEN

2D implizite FDM



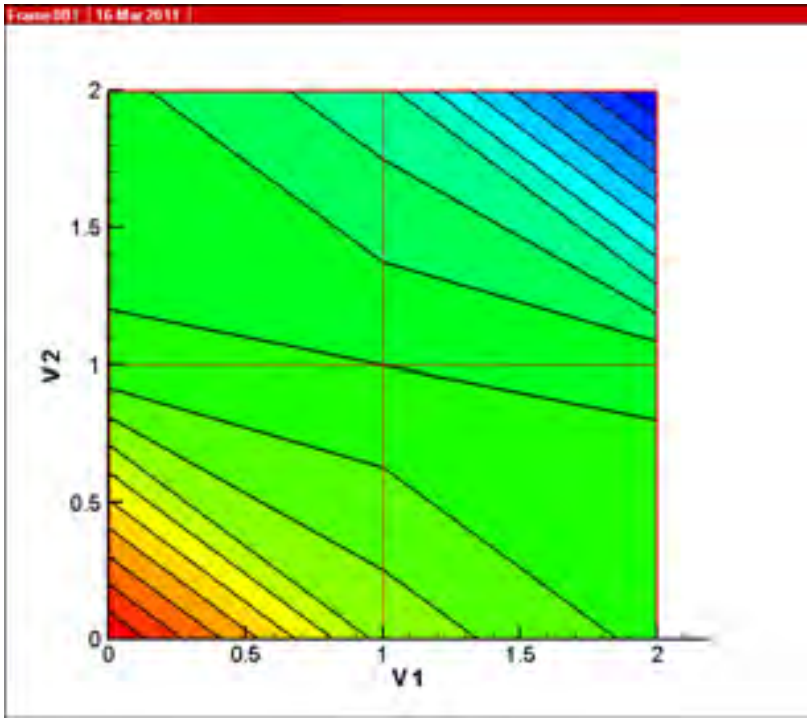


Figure: Ergebnisse des impliziten FD Verfahrens für t=0.25, 10 sec

Übung GW4 Der Quelltext für diese Übung befindet sich in `EXERCISES\GW4`.

TECHNISCHE UNIVERSITÄT DRESDEN