Modellierung von Hydrosystemen - SoSe 2022

BHYWI-22-B2-T14: Finite-Differenzen-Methode: Implizit

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

Zeitplan: Modellierung von Hydrosystemen: Zweiter Block (B2)

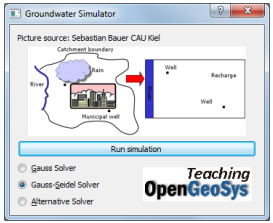
Sommersemester 2022: BHYWI-22-B2

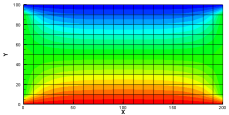
Datum	B2	Thema	Format
27.05.2022	B2-T1.0	Einführung in die Veranstaltung (B2) (Kolditz)	Online
27.05.2022	B2-T1.1	Hydromechanik und Numerische Methoden (Kolditz)	Online
27.05.2022	B2-T1.2	Grundwasserhydraulik und Prinzipbeispiel (Kolditz)	Online
03.06.2022	B2-T3.1	Stofftransport in Hydrosystemen (Shao)	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

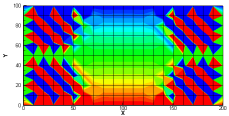
TECHNISCHE UNIVERSITÄT DRESDEN

Konzept

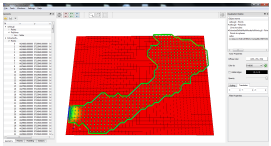
explizite FDM

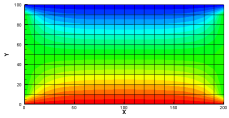


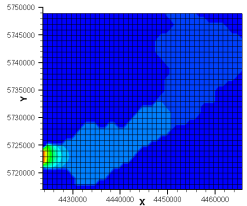




implizite FDM

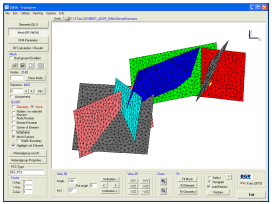


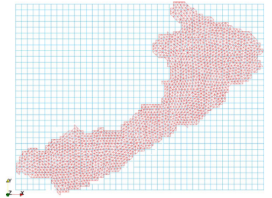




\- ...

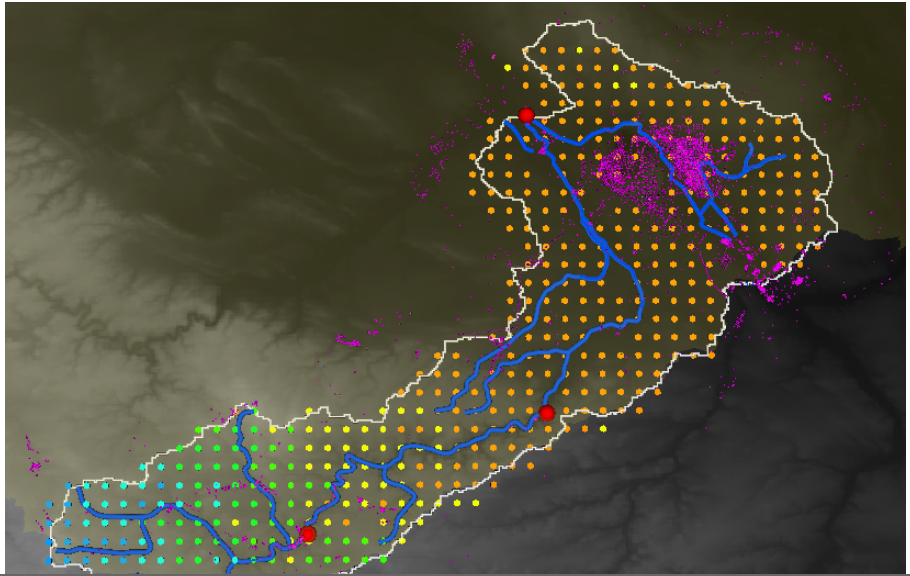
FEM



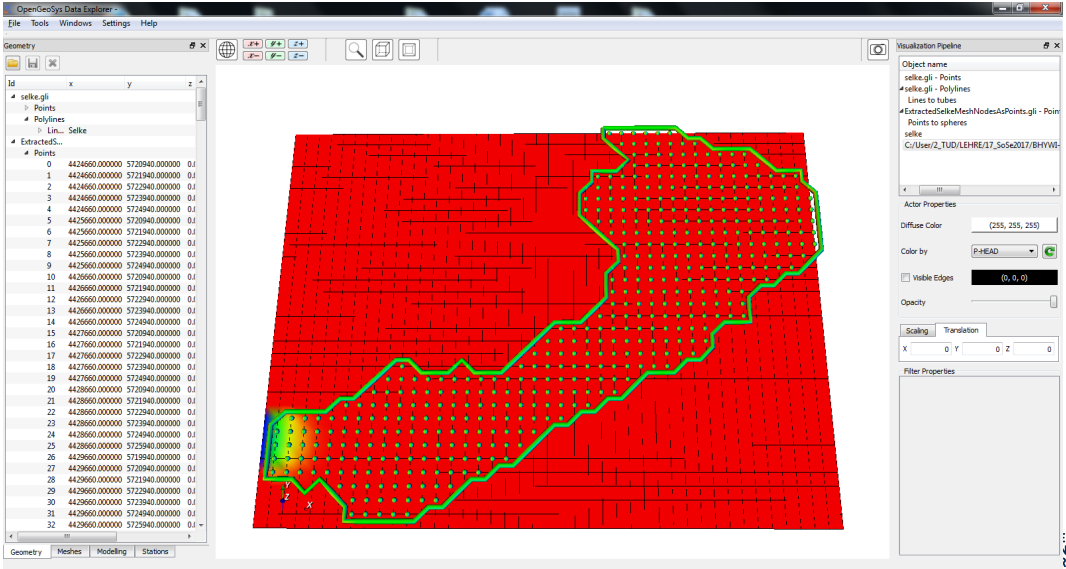


TECHNISCHE UNIVERSITÄT DRESDEN

Selke Einzugsgebiet



Selke Einzugsgebiet



2D implizite FDM

siehe auch Abschn. 4.2 Hydroinformatik II

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

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

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

2D implizite FDM

- Differenzen-Schema

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

2D implizite FDM

- Gleichungssystem

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

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).

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).

2D implizite FDM - Gleichnungssystem

$$\underbrace{\begin{bmatrix} 1+4Ne & -Ne & & & & & & & \\ -Ne & \dots & \dots & & & & & & \\ & \dots & \dots & \dots & & & & & \\ & & & -Ne & 1+4Ne & & & & \\ & & & & & 1+4Ne & -Ne & & \\ & & & & & & \dots & \dots & \dots \\ & & & & & & & -Ne & 1+4Ne \end{bmatrix}}_{\mathbf{A}}$$

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

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. Die wesentlichen Unterschiede der impliziten zur expliziten FDM sind, dass wir ein Gleichungssystem aufbauen und lösen müssen.

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

2D implizite FDM

Dennoch können wir erstaunlich viel wiederverwenden, bis auf

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

Listing 2: Rechenschema

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.

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

```
1  void FDM::AssembleEquationSystem()
2  {
3    // Matrix entries
4    for(i=0;i<IJ;i++)
5    {
6      vecb[i] = u[i];
7      for(j=0;j<IJ;j++)
8      {
9        matrix[i*IJ+j] = 0.0;
10       if(i==j)
11         matrix[i*IJ+j] = 1. + 4.*Ne;
12       else if(abs((i-j))==1)
13         matrix[i*IJ+j] = - Ne;
14     }
15   }
16   // Incorporate boundary conditions
17   IncorporateBoundaryConditions();
18   // Matrix output
19   WriteEquationSystem();
20 }
```

Listing 3: Gleichungssystem aufbauen

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.

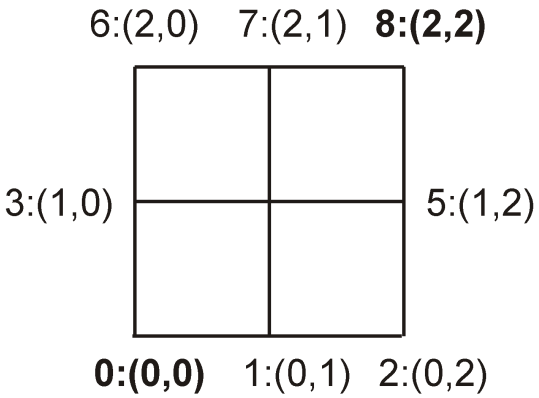


Fig.: Testbeispiel

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
```

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).

2D implizite FDM

```
1  void FDM::IncorporateBoundaryConditions()
2  {
3    size_t i_bc;
4    int i_row, k;
5    for(i_bc=0;i_bc<bc_nodes.size();i_bc++)
6    {
7      i_row = bc_nodes[i_bc];
8      // Null off-diangonal entries of the related row and columns
9      // Apply contribution to RHS by BC
10     for(j=0;j<IJ;j++)
11     {
12       if(i_row == j)
13         continue; // do not touch diagonals
14       matrix[i_row*(IJ)+j] = 0.0; // NUll row
15       k = j*(IJ)+i_row;
16       // Apply contribution to RHS by BC
17       vecb[j] -= matrix[k]*u[i_row];
18       matrix[k] = 0.0; // Null column
19     }
20     // Apply Dirichlet BC
21     vecb[i_row] = u[i_row]*matrix[i_row*(IJ)+i_row];
22   }
23 }
```

Listing 4: Randbedingungen einbauen (1. Art - Dirichlet)

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 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 0 2        b: -1
```

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 0 2        b:-2
```

2D implizite FDM

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

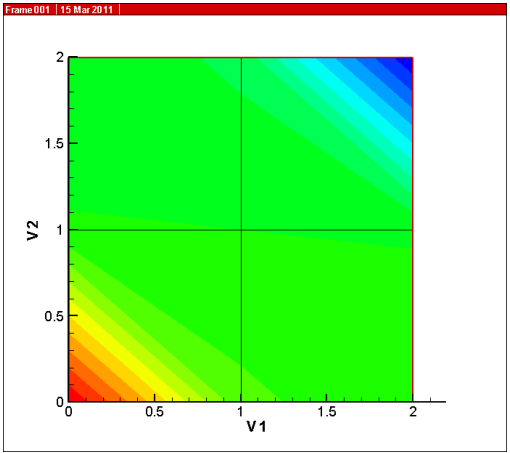
$$
\begin{aligned}
2h_1^{n+1} &= 2h_1^n \\
2h_2^{n+1} - 0.25h_3^{n+1} &= h_2^n + 0.25h_1^n \\
-0.25h_2 + 2h_3 - 0.5h_4 &= 0 \\
-0.25h_3 + 2h_4 - 0.5h_5 &= 0 \\
-0.25h_4 + 2h_5 - 0.5h_6 &= 0 \\
-0.25h_5 + 2h_6 - 0.5h_7 &= 0 \\
-0.25h_6 + 2h_7 - 0.5h_8 &= 0 \\
-0.25h_7 + 2h_8 &= -0.25 \\
2h_9^{n+1} &=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

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
```

2D implizite FDM



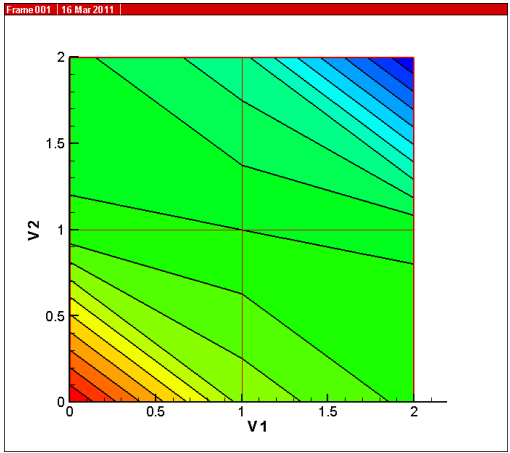


Fig.: Ergebnisse des impliziten FD Verfahrens für t=0.25, 10 sec