BHYWI-22: Semester-Fahrplan: 2018

Vorlesungen & Übungen

Datum	V	Thema
		Grundlagen
13.04.2018	1	
20.04.2018	3	Hydrologische Modellierung: Wasserquantität [Agnes' Lecture]
27.04.2018	5	Hydromechanik / Numerische Methoden: Wiederholung
		Numerische Methoden
04.05.2018	7	Einzugsgebiet: Übung
11.05.2018	9	Numerik: Finite-Differenzen-Verfahren (2D): Vorlesung
18.05.2018	11	Numerik: Finite-Differenzen-Verfahren (2D): Übung
25.05.2018	--	Pfingsten
01.06.2018	13	Software: Objekt-Orientierte FDM
08.06.2018	15	Numerik: Implementierung von Anfangs- und Randbedingungen
15.06.2018	17	Selke-Modell: Übung
22.06.2018	19	*VISLAB Exkursion*
29.06.2018	21	Numerik: Finite-Elemente-Verfahren
		Datenbasierte Methoden
06.07.2018	23	Hydrologeologische Modellierung: Datenbasierte Verfahren I
13.07.2018	25	Hydrologeologische Modellierung: Datenbasierte Verfahren II
20.07.2018	27	Klausurvorbereitung

Modellierung von Hydrosystemen

"Numerische und daten-basierte Methoden"

BHYWI-22-09 @ 2018

Finite-Differenzen-Methode

Olaf Kolditz

*Helmholtz Centre for Environmental Research – UFZ

[1]Technische Universität Dresden – TUDD

[2]Centre for Advanced Water Research – CAWR

11.05./18.05.2018 - Dresden

Fahrplan für heute ...

- Übersicht Übungen
- Grundlagen - GWE
- Grundlagen - TSE
- Übungen zur expliziten FDM (page 12) [BHYWI-22-E2]

- TSE double-check
- Übung zur expliziten FDM [BHYWI-22-E2]
- Übung zu OOP-FDM [BHYWI-22-E4]

TECHNISCHE UNIVERSITÄT DRESDEN

- PDE

$$S\frac{\partial h}{\partial t} - \frac{\partial}{\partial x}\left(K_x\frac{\partial h}{\partial x}\right) - \frac{\partial}{\partial y}\left(K_y\frac{\partial h}{\partial y}\right) = Q \tag{1}$$

in time ($\Delta t = t^{n+1} - t^n$)

$$ {}_j^{n+1} = \sum_{m=0}^{\infty} \frac{\Delta t^m}{m!} \left[\frac{\partial^m}{\partial t^m} \right]_j^n \tag{2} $$

in space ($\Delta x = x_{i+1}^n - x_i^n$)

$$ {}_{i+1}^{n} = \sum_{m=0}^{\infty} \frac{\Delta x^m}{m!} \left[\frac{\partial^m}{\partial x^m} \right]_i^n \tag{3} $$

in space ($\Delta y = y_{j+1}^n - y_j^n$)

$$ {}_{j+1}^{n} = \sum_{m=0}^{\infty} \frac{\Delta y^m}{m!} \left[\frac{\partial^m}{\partial y^m} \right]_j^n \tag{4} $$

- Zeitableitung

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

- in space

$$\left[\frac{\partial^2}{\partial x^2}\right]_{i,j}^n = \frac{{}_{i+1,j}^n - 2{}_{i,j}^n + {}_{i-1,j}^n}{\Delta x^2} - \frac{\Delta x^2}{12}\left[\frac{\partial^4}{\partial x^4}\right]_{i,j}^n - \dots \qquad (6)$$

$$\left[\frac{\partial^2}{\partial y^2}\right]_{i,j}^n = \frac{{}_{i,j+1}^n - 2{}_{i.j}^n + {}_{i,j-1}^n}{\Delta y^2} - \frac{\Delta y^2}{12}\left[\frac{\partial^4}{\partial y^4}\right]_{i,j}^n - \dots \qquad (7)$$

$$
\begin{aligned}
&S_{i,j}\frac{u_{i,j}^{n+1}-u_{i,j}^{n}}{\Delta t} \\
-K_{i,j}^{x}\frac{u_{i+1,j}^{n}-2u_{i,j}^{n}+u_{i-1,j}^{n}}{\Delta x^2} \quad - \quad &K_{i,j}^{y}\frac{u_{i,j+1}^{n}-2u_{i,j}^{n}+u_{i,j-1}^{n}}{\Delta y^2} = Q_{i,j}
\end{aligned}
\tag{8}
$$

$$
\begin{aligned}
u_{i,j}^{n+1} &= u_{i,j}^{n} \\
&+ \frac{K_{i,j}^{x}}{S_{i,j}}\frac{\Delta t}{\Delta x^2}u_{i+1,j}^{n}-2u_{i,j}^{n}+u_{i-1,j}^{n} \\
&+ \frac{K_{i,j}^{y}}{S_{i,j}}\frac{\Delta t}{\Delta y^2}u_{i,j+1}^{n}-2u_{i,j}^{n}+u_{i,j-1}^{n} \\
&+ \frac{Q_{i,j}}{S_{i,j}}
\end{aligned}
\tag{9}
$$

Grundlagen - FDM

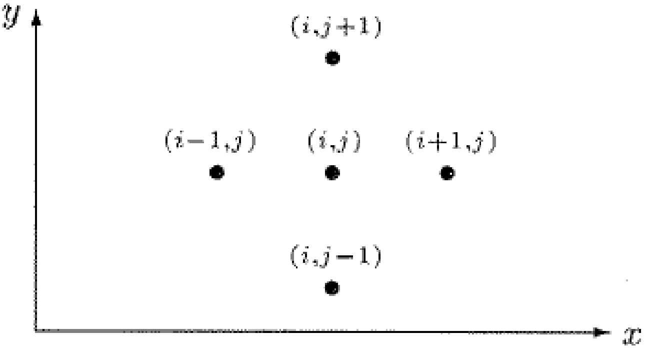


Figure: 5-Punkte-Stern (Knabner und Angermann 2000)

- Datenstrukturen

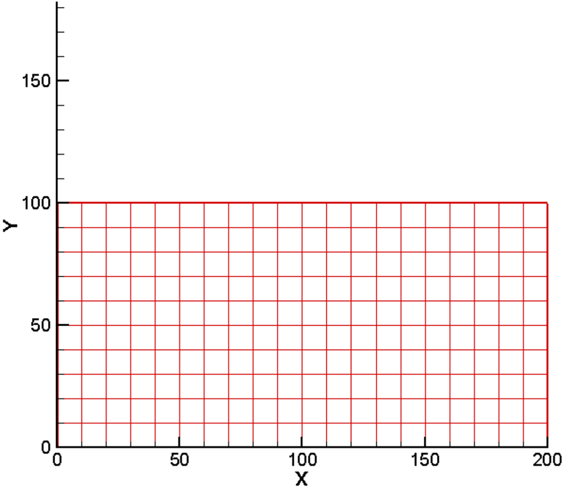
Table:

Feldgröße	u
Physikalische Parameter	S, K, Q
Numerische Parameter	$\Delta t, \Delta x, \Delta y$

Die Minimal-Datenstrukturen für die Programmierung der Gleichung (9) sind damit:

```
std::vector<float>u_new;
std::vector<float>u_old;
float S0,Kf,Q;
float dx,dy,dt;
```

Übung - BHYWI-22-E2 - QAD



Um dieses Gitter "abtasten" zu können, schreiben wir folgende doppelte Schleife.

```
for(j=0;j<jy;j++)
{
  nn = j*ix;
  for( i=0;i<ix;i++)
  {
    n = nn+i;
    u_new[n] = u[n] \
             + Kf/S0*dt/dx2 * (u[n+1]-2*u[n]+u[n-1]) \
             + Kf/S0*dt/dy2 * (u[(j+1)*ix+i]-2*u[n]+u[(j-1)*ix+i]) \
             + Q/S0;
  }
}
```

Um dieses Gitter "abtasten" zu können, schreiben wir folgende doppelte Schleife.

```
for(int j=0;j<jy;j++)
{
  nn = j*ix;
  for(int i=0;i<ix;i++)
  {
    n = nn+i;
    if(IsBCNode(n,bc_nodes))
      continue;
    u_new[n] = u[n] \
             + Kf/S0*dt/dx2 * (u[n+1]-2*u[n]+u[n-1]) \
             + Kf/S0*dt/dy2 * (u[(j+1)*ix+i]-2*u[n]+u[(j-1)*ix+i]) \
             + Q/S0;
  }
}
```

TECHNISCHE UNIVERSITÄT DRESDEN

Übung - BHYWI-22-E2 - QAD

Dabei ist j der Laufindex über die y Richtung und i der Laufindex über die x Richtung. Ganz wichtig ist natürlich, den Speicher für die Vektoren bereitzustellen, bevor es los geht.

```
u.resize(ix*jy);
u_new.resize(ix*jy);
```

- Welche Rolle spielen `ix` und `jy` bei der Speicherreservierung?

Natürlich müssen auch die Parameter vor der Berechnung initialisiert werden

```
ix = 21;
jy = 11;
dx = 10.;
dy = 10.;
dt = 0.25e2;
S0 = 1e-5;
Kf = 1e-5;
Q  = 0.;
u0 = 0.;
```

- Welche Einheiten haben die einzelnen Parameter?

Übung - BHYWI-22-E2 - QAD

Das mit den Anfangsbedingungen ist eine einfache Sache. Mit der Doppelschleife über alle Knoten, können wir sehr einfach einen Wert `u0` als Anfangsbedingung überall zuweisen.

```
for(int i=0;i<ix;i++)
  for(int j=0;j<jy;j++)
  {
    u[j*(ix+1)] = u0;
    u_new[j*(ix+1)] = u0;
  }
}
```

Mit den Randbedingungen ist es etwas kniffliger ...

```
//top and bottom
int l;
for(int i=0;i<ix;i++)
{
  bc_nodes.push_back(i); u[i] = u_top  u_new[i] = u_top;
  l = ix*(jy-1)+i;
  bc_nodes.push_back(l); u[l] = u_bottom;  u_new[l] = u_bottom;
}
//left and right side
for(int j=1;j<jy-1;j++)
{
  l = ix*j;
  bc_nodes.push_back(l); u[l] = u_left;  u_new[l] = u_left;
  l = ix*j+ix-1;
  bc_nodes.push_back(l); u[l] = u_right;  u_new[l] = u_right;
}
```

Sie sehen, dass wir für die Zuweisung der Randbedingungen eine neue Datenstruktur eingeführt haben.

```
std::vector<float>u_bc;
```

Das Einbauen der Randbedingungen integrieren wir direkt in die Doppelschleife zur Berechnung der Knotenwerte. Dabei kommt eine neue Funktion `IsBCNode` ins Spiel, die wir uns gleich noch näher anschauen. `IsBCNode` soll eigentlich nichts anderes machen, als beim Auftreten einer Randbedingung nichts zu tun (i.e. `continue`). Randbedingungswerte sind gesetzt, müssen also nicht gerechnet werden.

```
for(int j=0;j<jy;j++)
{
  nn = j*ix;
  for(int i=0;i<ix;i++)
  {
    n = nn+i;
    if(IsBCNode(n,bc_nodes))
      continue;
    ...
  }
}
```

Wie funktioniert nun IsBCNode?

```
bool IsBCNode(int n,std::vector<int>bc_nodes)
{
  bool is_node_bc = false;
  for(int k=0;k<(size_t)bc_nodes.size();k++)
  {
    if(n==bc_nodes[k])
    {
      is_node_bc = true;
      return is_node_bc;
    }
  }
  return is_node_bc;
}
```

Übung - BHYWI-22-E2 - QAD

Struktur der Funktion:

- Rückgabewert: logischer Wert wahr oder falsch
- Parameter: aktueller Gitterpunkt und Randbedingungsknotenvektor

Die Funktion überprüft, ob der Gitterpunkt n ein Randbedingungsknoten ist und gibt den entsprechenden logischen Wert zurück.

TECHNISCHE UNIVERSITÄT DRESDEN

Das Ergebnis der finite Differenzen Simulation sehen wir in der Abb.

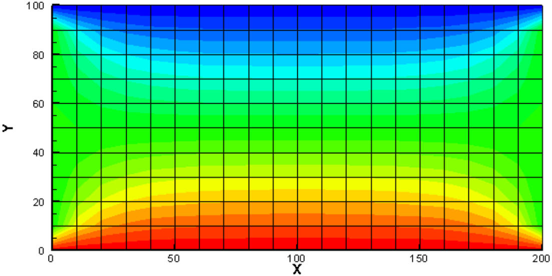


Figure: Berechnete Druckverteilung im Rechteck-Aquifer nach 100 Zeitschritten $\Delta t = 25$ sec

Übung - BHYWI-22-E2 - QAD

Jetzt werden wir mutig und vergrößern mal den Zeitschritt, sagen wir mal verdoppeln: $\Delta t = 50$ sec. Das Maleur sehen wir in der Abb. Was ist hier los?

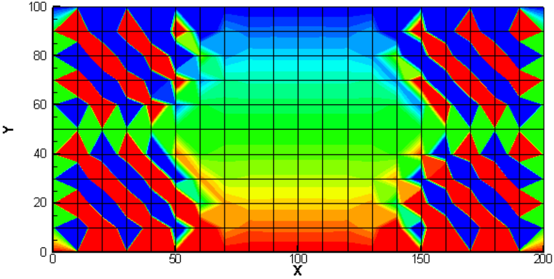


Figure: Berechnete Druckverteilung im Rechteck-Aquifer nach 100 Zeitschritten $\Delta t = 50$ sec

Übung - BHYWI-22-E2 - QAD

Wir erinnern uns noch dunkel daran, dass der Preis für das einfache explizite FDM ein strenges Stabilitätskriterium war (siehe Hydroinformatik, Teil II, Abschn. 3.2.2 und Abschn. 4.1). Dabei muss die Neumann-Zahl kleiner einhalb sein.

$$Ne = \alpha \frac{\Delta t}{\Delta x^2} \leq \frac{1}{2} \tag{10}$$

TECHNISCHE UNIVERSITÄT DRESDEN

Übung - BHYWI-22-E2 - QAD

Prima, aber was ist jetzt α und warum steht nur Δx und nicht auch Δy in der Gleichung? Zur bestimmung des α schreiben wir die Grundwassergleichung in eine Diffusionsgleichung wie folgt um.

$$\frac{\partial h}{\partial t} = \frac{K_x}{S}\frac{\partial^2 h}{\partial x^2} + \frac{K_y}{S}\frac{\partial^2 h}{\partial y^2} + \frac{Q}{S} \tag{11}$$

Übung - BHYWI-22-E2 - QAD

Wir sehen, dass es eigentlich zwei α-s gibt, für jede Richtung eins.

$$\begin{aligned} \alpha_x &= \frac{K_x}{S} \\ \alpha_y &= \frac{K_y}{S} \end{aligned} \tag{12}$$

- Welche Einheit hat unser Grundwasser-α ?

TECHNISCHE UNIVERSITÄT DRESDEN

Übung - BHYWI-22-E2 - QAD

Der richtige Zeitschritt für unser explizites FD Verfahren ergibt sich somit zu:

$$\Delta t \leq \frac{\min(\Delta x^2, \Delta y^2)}{2\alpha} \tag{13}$$

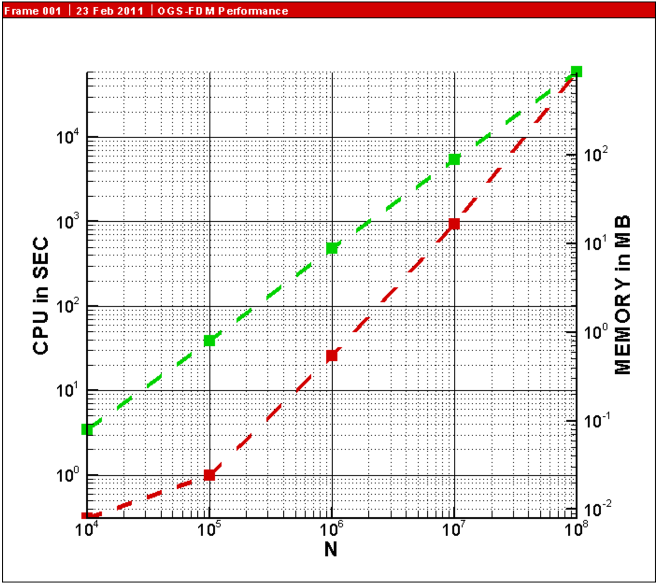
TECHNISCHE UNIVERSITÄT DRESDEN

Die numerischen und hydraulischen Parameter sind in der nachfolgenden Tabelle **??** zu finden. Glücklicherweise sind die Ortdiskretisierungen und die hydraulischen Leitfähigkeiten in beide Koordinatenrichtungen gleich (isotropes Problem).

Übung - BHYWI-22-E2 - QAD

$$\Delta t \leq \frac{100m^2}{2 \times 1m^2/s} = 50s \quad (14)$$

Übung - BHYWI-22-E2 - QAD



Übung - BHYWI-22-E2 - QAD

Wie stellen wir eine Zeitmessung in einem Programm an.

```
clock_t start, end; // Definitionen
...
start = clock();    // Beginn Zeitmessung
...
end = clock();      // Ende Zeitmessung
...
time= (end-start)/ (double)(CLOCKS_PER_SEC); // Differenz
```

Übung USA2 Der Quelltext für diese Übung befindet sich in `EXERCISES\USA2`.

TECHNISCHE UNIVERSITÄT DRESDEN