Hydroinformatik II: Finite Differenzen Methode

[1]Helmholtz Centre for Environmental Research – UFZ, Leipzig
[2]Technische Universität Dresden – TUD, Dresden

Dresden, 12. Juni 2015

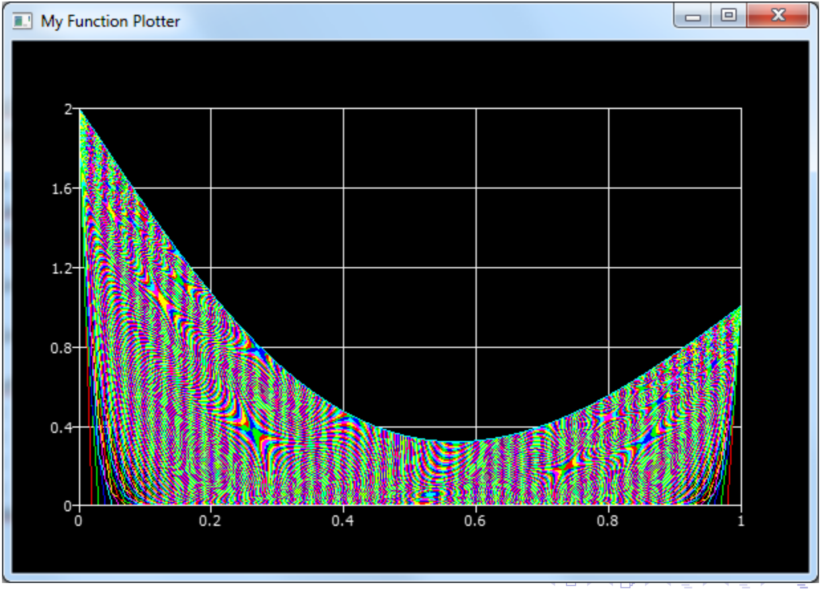
Vorlesungsplan Hydroinformatik II SoSe 2015

#	Datum	Thema
01	17.04.2015	Einführung, Grundlagen: Kontinuumsmechanik
02	24.04.2015	Grundlagen: Kontinuumsmechanik/Hydromechanik
-	01.05.2015	Maifeiertag
03	08.05.2015	HW: Einführung in Qt (Installation)
04	15.05.2015	Grundlagen: Partielle Differentialgleichungen / TEX
05	22.05.2015	Grundlagen: Numerische Methoden
-	29.05.2015	Pfingsten
06	05.06.2016	Numerik: (exp) Finite Differenzen Methode
07	12.06.2015	Numerik: (imp) Finite Differenzen Methode
08	19.06.2015	Numerik: Zusammenfassung FDM
09	26.06.2015	Gerinnehydraulik: Theorie - Grundlagen
10	03.07.2015	Gerinnehydraulik: Saint-Venant Gleichung ($\Longrightarrow$ HSA)
11	10.07.2015	Gerinnehydraulik: Programmierung, Übung 1
12	17.07.2015	Gerinnehydraulik: Programmierung, Übung 2
13	17.07.2015	Kurs-Zusammenfassung und Abschluss

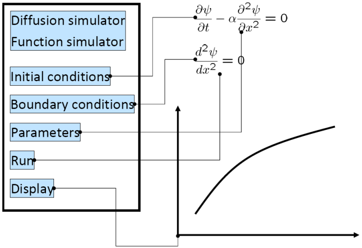
Fahrplan für heute ...
Implicit FDM for diffusion equation (section 4.2)

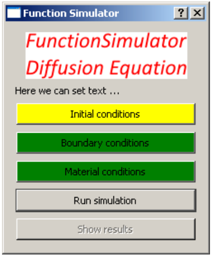
1. L5: Implicit FDM for diffusion equation
2. E9: Implementation of iFDM
3. Qt Basics

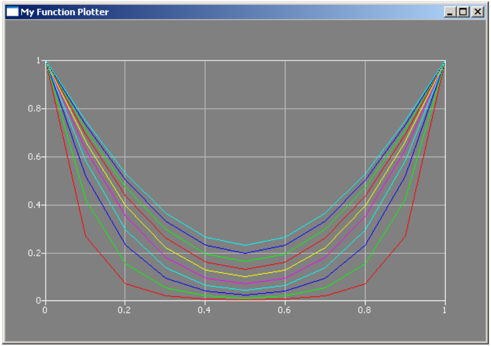
Letzte Vorlesung: explizite FDM



Ziel der Vorlesung







Implicite FDM - Theory #1 (Skript 4.2)

- PDE for diffusion processes

$$\frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} = 0 \quad (1)$$

- Time discretization

$$\left[\frac{\partial u}{\partial t}\right]_j^n \approx \frac{u_j^{n+1} - u_j^n}{\Delta t} \quad (2)$$

- Forward time / centered space

$$\left[\frac{\partial^2 u}{\partial x^2}\right]_j^{n+1} \approx \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{\Delta x^2} \quad (3)$$

- (Current time / centered space)

$$\left[\frac{\partial^2 u}{\partial x^2}\right]_j^n \approx \frac{u_{j-1}^n - 2u_j^n + u_{j+1}^n}{\Delta x^2} \quad (4)$$

Implicite FDM - Theory #2 (Skript 4.2)

- Substitute into PDE

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} - \alpha \frac{u_{j-1}^{n+1} - 2u_j^{n+1} + u_{j+1}^{n+1}}{\Delta x^2} = 0 \tag{5}$$

- Algebraic equation (index notation)

$$\frac{\alpha \Delta t}{\Delta x^2} \left(-u_{j-1}^{n+1} + 2u_j^{n+1} - u_{j+1}^{n+1}\right) + u_j^{n+1} = u_j^n \tag{6}$$

- Algebraic equation (matrix notation)

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{7}$$

- Explain steps with black board

Implicite FDM - Theory #3 (Skript 4.2)

- Algebraic equation (matrix notation)

$$\mathbf{Ax} = \mathbf{b} \tag{8}$$

- Algebraic equation (index notation)

$$Ne\left(-u_{j-1}^{n+1} + 2u_j^{n+1} - u_{j+1}^{n+1}\right) + u_j^{n+1} = u_j^n \tag{9}$$

- Let's take a closer look ...

$$\underbrace{\begin{bmatrix} 1+2Ne & -Ne & & & \\ -Ne & \dots & \dots & & \\ & \dots & \dots & \dots & \\ & & \dots & \dots & -Ne \\ & & & -Ne & 1+2Ne \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} u_0 \\ u_1 \\ \dots \\ u_{n-1} \\ u_n \end{bmatrix}}_{\mathbf{x}} = \underbrace{\begin{bmatrix} b_0 \\ b_1 \\ \dots \\ b_{n-1} \\ b_n \end{bmatrix}}_{\mathbf{b}}$$

Implementation #1

- Data structures (as usual ...)

```
Dialog::Dialog(QWidget *parent) : QDialog(parent)
{
  matrix = new double[n*n];
  vecb = new double[n];
  vecx = new double[n];
}

Dialog::~Dialog()
{
  delete [] matrix;
  delete [] vecb;
  delete [] vecx;
}
```

Implementation #2

- Functions (a pain in the neck ...)

```
AssembleEquationSystem();
Gauss(matrix,vecb,vecx,n);

void Dialog::AssembleEquationSystem()
{...
  int i,j;
  // Matrix entries
  for(i=0;i<n;i++)
  {
    vecb[i] = u_old[i]; // RHS Vektor
    for(j=0;j<n;j++)
    {
      matrix[i*n+j] = 0.0;
      if(i==j) // Hauptdiagonale
        matrix[i*n+j] = 1. + 2.*Ne;
      else if(abs((i-j))==1) // Nebendiagonalen
        matrix[i*n+j] = - Ne;
    }
  }
...}
```

Implementation #3

- Boundary conditions - concept

$$\mathbf{Ax} = \mathbf{b} \tag{11}$$

$$[1\,0\,0\ldots0] \begin{bmatrix} u_0 \\ u_1 \\ \ldots \\ u_n \end{bmatrix} = \begin{bmatrix} u_0 \\ 0 \\ \ldots \\ 0 \end{bmatrix} \tag{12}$$

Implementation #4

- Boundary conditions - implementation

```
void Dialog::AssembleEquationSystem()
{...
  // Treat boundary conditions
  for(i=0;i<n;i++)
    for(j=0;j<n;j++)
    {
      if(i==0||i==n-1)
        matrix[i*n+j] = 0.0;
    }
  for(i=0;i<n;i++)
  {
    if(i!=0&&i!=n-1)
      continue;
    for(j=0;j<n;j++)
    {
       if(i==j)
         matrix[i*n+j] = 1.0;
       else
         matrix[i*n+j] = 0.0;
    }
  }
}
```

Solving EQS - How to ... the magic Gauss function

- 1

$$a_{11}u_1 + a_{12}u_2 = b_1 \quad (13)$$

$$a_{21}u_1 + a_{22}u_2 = b_2 \quad (14)$$

- 2

$$a_{21}\frac{a_{11}}{a_{21}}u_1 + a_{22}\frac{a_{11}}{a_{21}}u_2 = \frac{a_{11}}{a_{21}}b_2 \quad (15)$$

- 3

$$\left(\frac{a_{22}a_{11}}{a_{21}} - a_{12}\right)u_2 = \frac{a_{11}}{a_{21}}b_2 - b_1 \quad (16)$$

- 4

$$u_2 = \frac{\frac{a_{11}}{a_{21}}b_2 - b_1}{\frac{a_{22}a_{11}}{a_{21}} - a_{12}} \quad (17)$$

Implementation #5

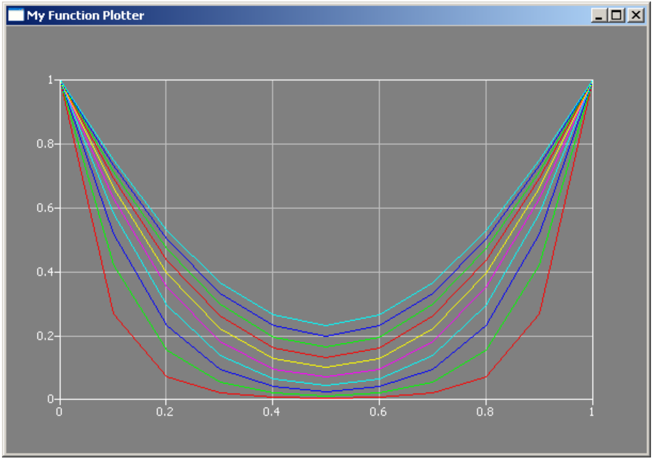
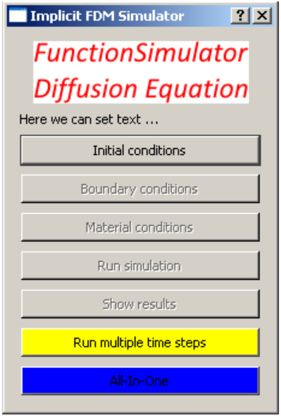


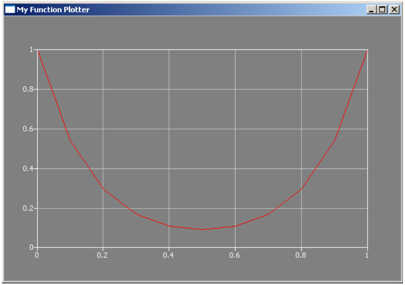
Abbildung: Zeitliche Entwicklung des Diffusionsprofils - implizites Verfahren (Wahoo...)

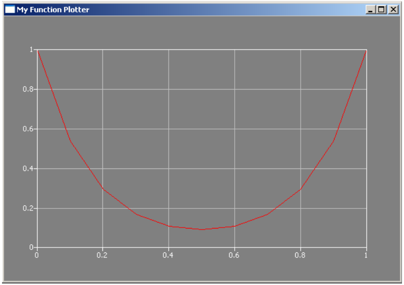
Implementation #6: Run multiple time steps

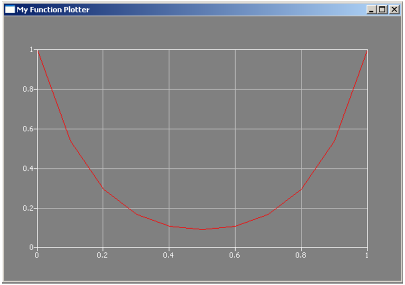


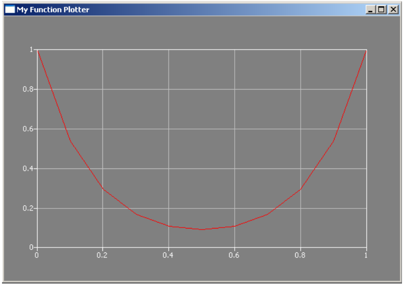

```
void Dialog::on_pushButtonMTS_clicked()
{
  //run time step
  on_pushButtonRUN_clicked();
  //show results
  on_pushButtonSHO_clicked();
  //prepare next time steps
  for(int i=0;i<n;i++)
  {
    u_old[i] = u_new[i];
  }
}
```

Implementation #6: Run multiple time steps









C++Basics (Star Wars ...)

- ... just a star (*)

```
void Dialog::on_pushButtonSHO_clicked()
{
  Plotter *plotter = new Plotter;
  ...
  plotter->show();
}
```

- ... for better plotting (in your life)

```
void Dialog::SHOBetter()
{
  ...
  plotterAIO->show();
}
```

C++Basics (... the difference (the whole story))

- Plotter declaration

```
class Dialog : public QDialog
{...
private:
  Plotter *plotterAIO;
}
```

- Plotter declaration ... otherwise

```
Dialog::Dialog(QWidget *parent) : QDialog(parent)
{...
  plotterAIO = new Plotter;
}
```

- Finally ready to use ...

```
void Dialog::SHOBetter()
{...
  plotterAIO->show();
}
```

Qt Basics - E10

- QLineEdit