

Modellierung Hydrosysteme: Finite-Differenzen-Methode (FDM)

Prof. Dr.-Ing. habil. Olaf Kolditz

¹Helmholtz Centre for Environmental Research – UFZ, Leipzig

²Technische Universität Dresden – TUD, Dresden

Dresden, 10. Juni 2016

Vorlesungsplan SoSe 2016: Hydrosystemanalyse

#	Datum	Vorlesung	Übung	Skript
1	15.04.16	Einführung, Systemanalyse		
2	15.04.16	Grundwasserhydraulik: Einzugsgebiet Bilanzierung, Vorlesung		1.1+2
3	23.04.16	Grundwasserhydraulik: Einzugsgebiet Bilanzierung, Übung		1.3
7	20.05.16	Grundwasserhydraulik: Finite-Differenzen-Verfahren, Rechteckaquifer		
11	10.06.16	Grundwasserhydraulik: Finite-Differenzen-Verfahren, Selke	BHYWI-22-E3	1.4, 1.5
13	17.06.16	Grundwasserhydraulik: Finite-Differenzen-Verfahren, OOP, VTK, iFDM	BHYWI-22-E4	1.5, 1.6
15	26.06.16	Grundwasserhydraulik: implizites Finite-Differenzen-Verfahren, Randbedingungen	BHYWI-22-E15	1.5
20	03.07.16	Grundwasserhydraulik: Finite-Elemente-Verfahren		
23	10.07.16	UFZ-Exkursion: VISLAB		
32	17.07.16	FEM 2-D, Klausurvorbereitung		

Fahrplan für heute ...

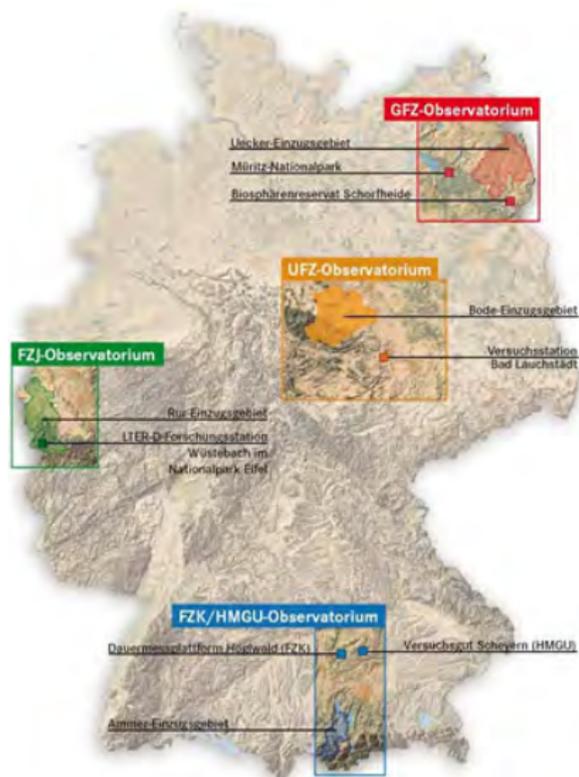
- ▶ Belege (Noten und fehlende ...)
- ▶ Vorstellen der Case Study: Selke Catchment (Bode)
- ▶ OGSDataExplorer
- ▶ Aktive und Inaktive Knoten
- ▶ FDM - erstmal QAD Programmierung (USA3)

Belege (fehlende)

3948820, 3992355, 3857261, 3948787, 3948849, 3965563,
3748848, 4000448, 3959987, 4024305, 3952636, 3765740,
3963344, 3956377,

Bester Beleg: 4008007

TERENO - Hydrologische Observatorien



- ▶ TERENO Projekt
- ▶ Steffen Zacharias
- ▶ Ute Wollschläger
- ▶ VISLab (Karsten Rink)

Case Study: Bode Einzugsgebiet



Abbildung: Digitales Geländemodell (DEM) des Bode-Einzugsgebietes

Case Study: Bode Einzugsgebiet

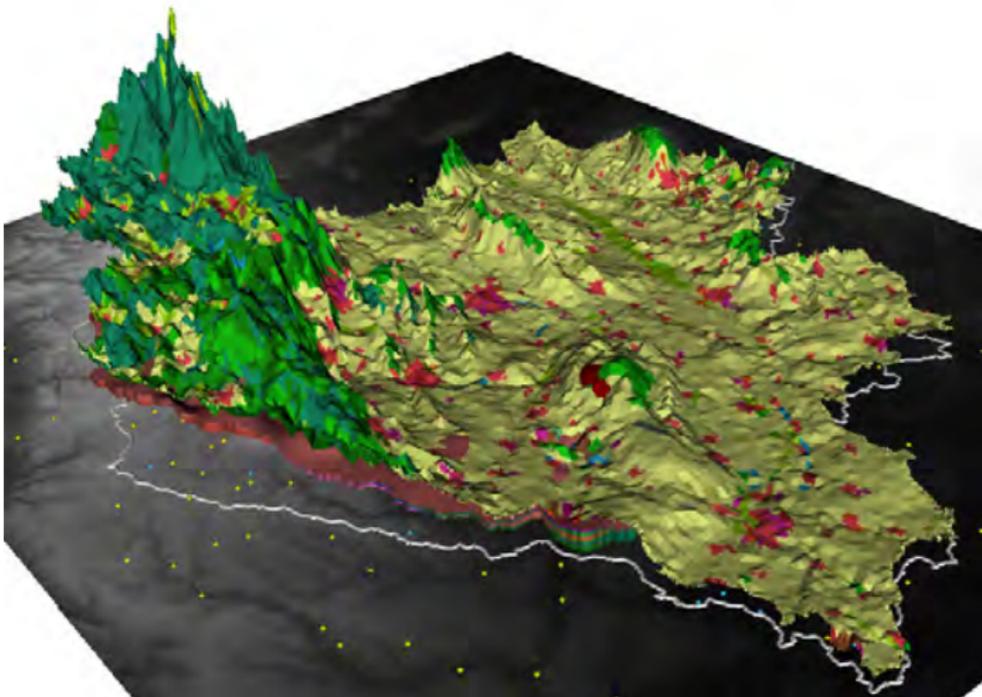


Abbildung: DEM mit Landnutzung

Case Study: Bode Einzugsgebiet

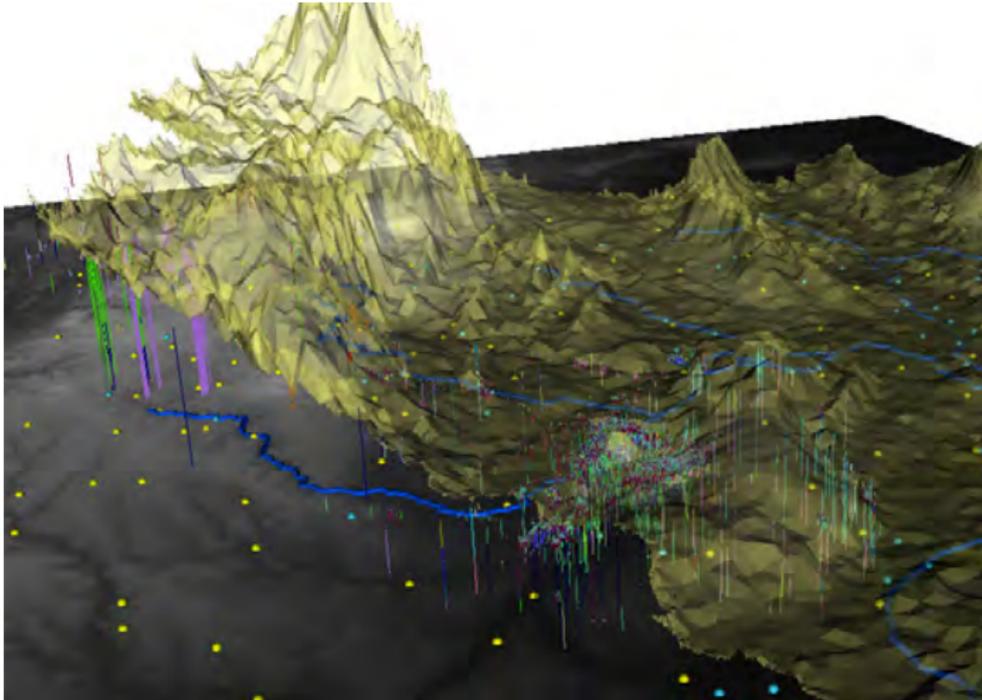


Abbildung: DEM mit geologischen Daten

Selke Einzugsgebiet

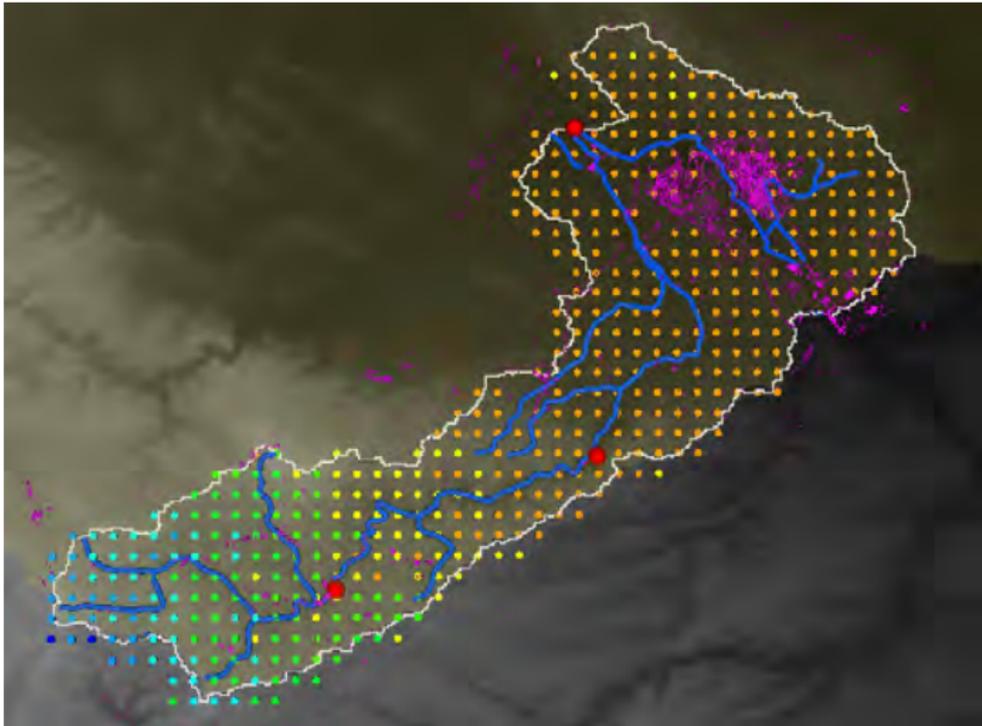
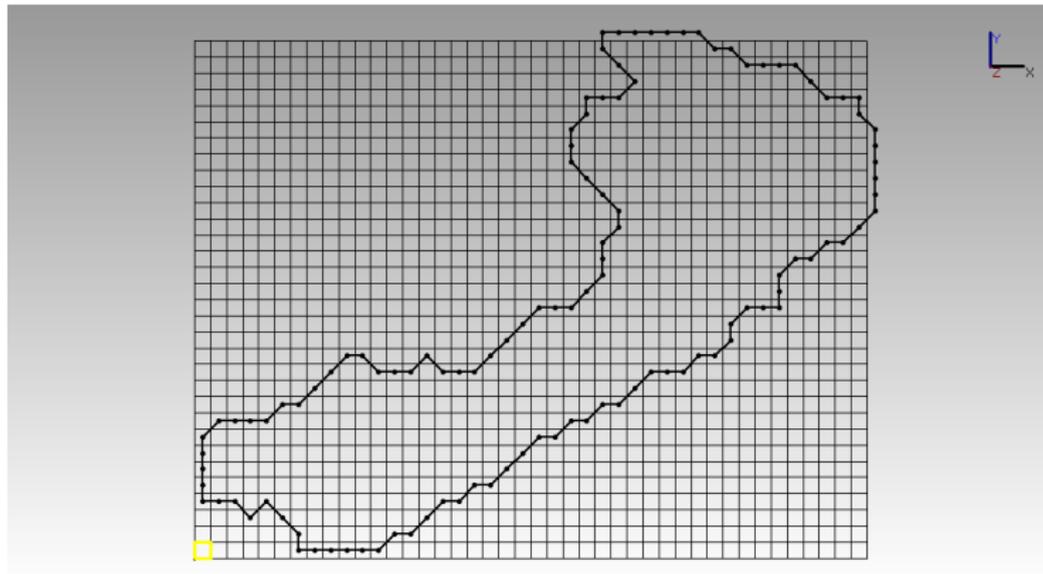


Abbildung: Untersuchungsgebiet - Selke

Selke Einzugsgebiet

Die Abb. zeigt uns eine mögliche Approximation des Selke-Einzugsgebietes mit einer relativ geringen Anzahl von FD Zellen ($32 \times 42 = 1344$).

- ▶ Aus wie vielen FD Knoten besteht das FD Mesh?



Selke Einzugsgebiet

Wie bekommen wir aus unserem regelmäßigen rechteckigen Raster ein eher unregelmäßiges Catchment herausgeschnitten? Der Trick besteht darin, einzelne Zellen zu deaktivieren. Das klingt schon wieder nach Arbeit, ist aber machbar, dafür gibt's die nächste Übung (GW2). Die geometrische Analyse mit OGS liefert uns zunächst eine Liste von Gitterpunkten die ausserhalb des Catchments liegen (siehe Übung GW2):

- ▶ ExtractedSelkeMeshIDs.txt
- ▶ selke.gli

Diese Files können wir mal mit dem OGS-DatenExplorer (ogs-gui.exe) laden.

OpenGeoSys - Datenexplorer (OGS-DE)

- ▶ Download von der Lehre-Seite
- ▶ Manual
- ▶ Übung

OGS-DE: Selke Einzugsgebiet

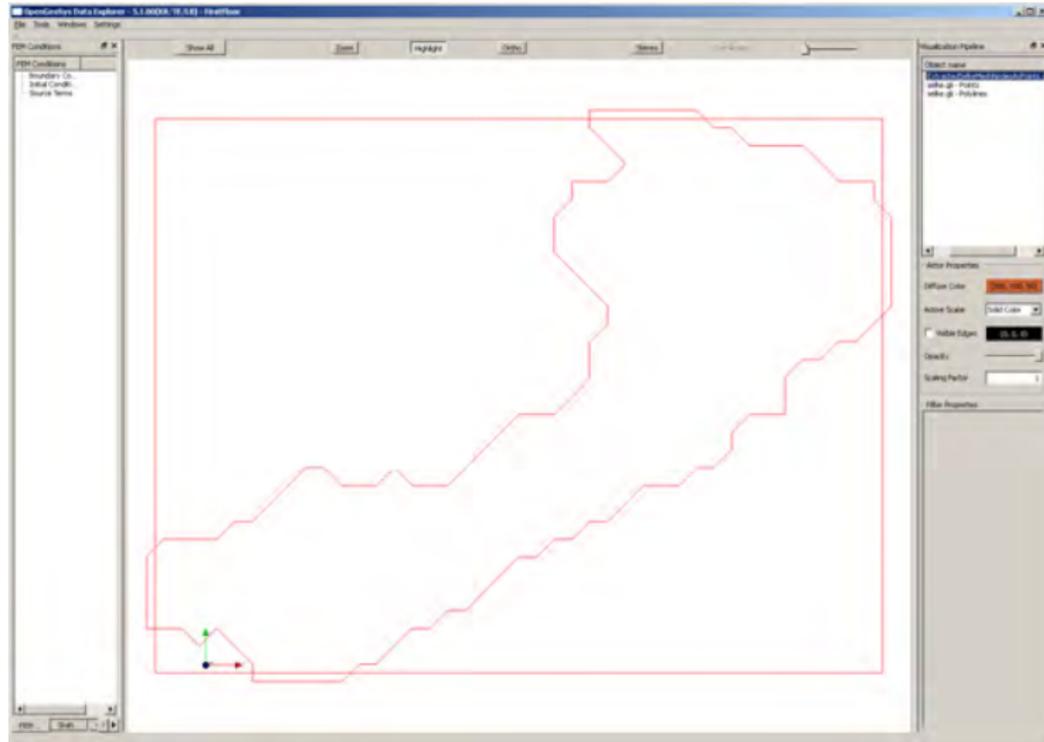


Abbildung: Das sind zwar die Daten, sieht aber noch nach nix aus ...

OGS-DE: Selke Einzugsgebiet

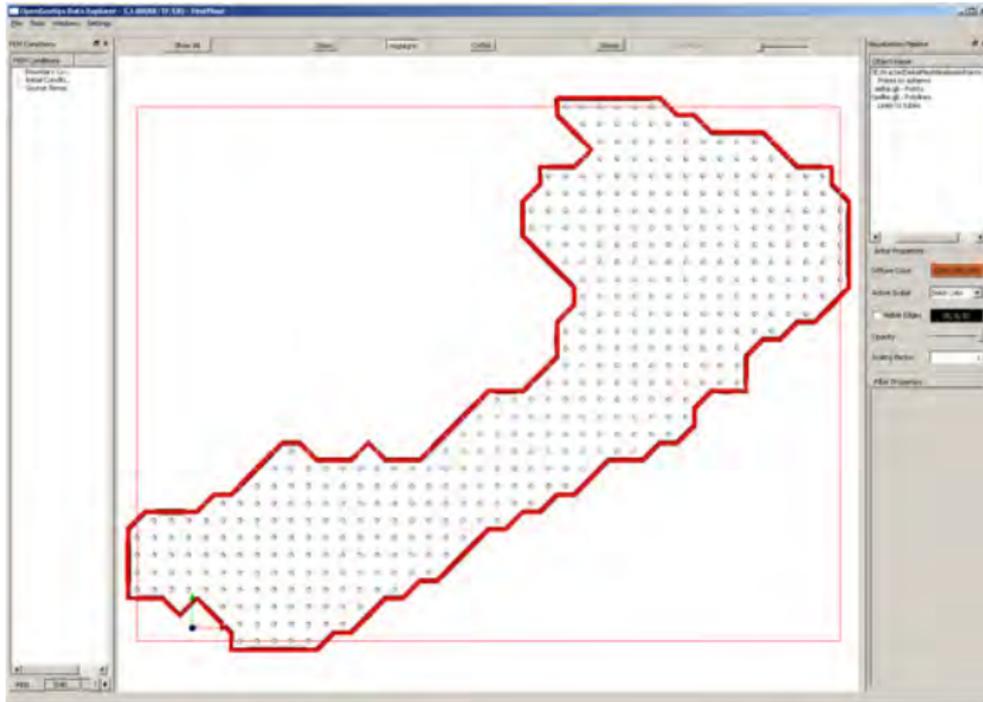
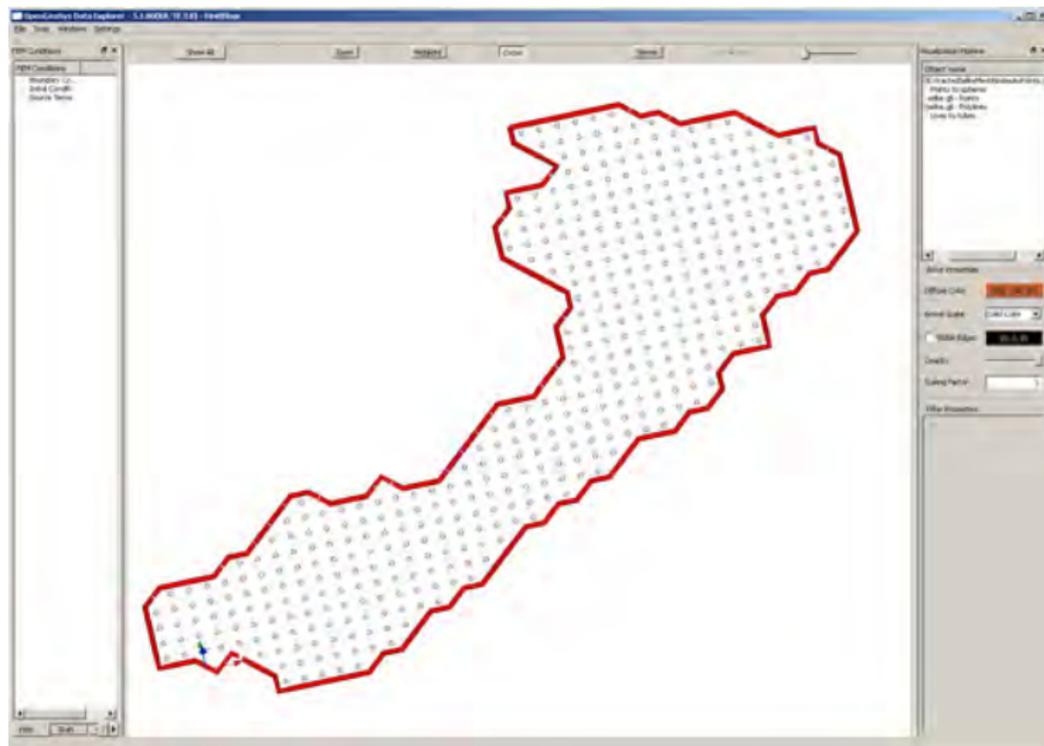


Abbildung: OGS hat ein paar nette VTK-Filter, um geometrische Objekte herauszuheben

OGS-DE: Selke Einzugsgebiet



Selke Einzugsgebiet Modellieren

Das ist zwar alles schön und gut, was wir aber brauchen sind die Knoten ausserhalb des Catchments, damit wir diese für das FD Verfahren deaktivieren können (Tafelbild). Also müssen wir doch selber ran. Unser Plan ist wie folgt:

1. Aktive Knoten lesen und speichern.
2. Aktive Knoten sortieren (Gruß an Hydroinformatik I - Hantieren mit Listen)
3. (das Zwischenergebnis zur Sicherheit mal rausschreiben)
4. Alle Knoten rausfischen, die NICHT aktiv sind.
5. Dabei kommt eine neue Hilfs-Funktion `NodeInList` ins Spiel (siehe unten).
6. Wir überzeugen uns vom Ergebnis (File schreiben) ...
7. ... und natürlich graphisch, wozu haben wir denn Visual C++ gelernt!

Selke Einzugsgebiet Modellieren

- ▶ ActiveNodes.txt

173

216

259

302

345

174

217

260

303

346

175

218

261

304

347

...

Selke Einzugsgebiet Modellieren

- ▶ Aktive Knoten lesen und speichern.
- ▶ Aktive Knoten sortieren (Gruß an Hydroinformatik I - Hantieren mit Listen)

```
std::list<int>nodes_active;
std::ifstream active_nodes_file;
active_nodes_file.open("ActiveNodes.txt");
int na;
while(!active_nodes_file.eof())
{
    active_nodes_file >> na;
    nodes_active.push_back(na);
}
nodes_active.sort();
```

Selke Einzugsgebiet Modellieren

- ▶ Das Zwischenergebnis zur Sicherheit mal rausschreiben

```
std::ofstream active_nodes_file_test;
active_nodes_file_test.open("ActiveNodesSorted.txt");
list<int>::const_iterator p = nodes_active.begin();
while(p!=nodes_active.end())
{
    active_nodes_file_test << *p << endl;
    ++p;
}
active_nodes_file_test.close();
```

Selke Einzugsgebiet Modellieren

- ▶ ActiveNodesSorted.txt

50

51

52

53

54

93

94

95

96

97

98

...

- ▶ E: Überzeugen sie sich, ob unser Unterfangen erfolgreich war, indem sie die Elementanzahl der beiden Listen bestimmen und rausschreiben.

Selke Einzugsgebiet Modellieren

- ▶ Alle Knoten rausfischen, die NICHT aktiv sind.
- ▶ Dabei kommt eine neue Hilfs-Funktion `NodeInList` ins Spiel (siehe unten).

```
for(j=0;j<jy;j++)
{
    nn = j*ix;
    for( i=0;i<ix;i++)
    {
        n = nn+i;
        if(!NodeInList(n,nodes_active))
            nodes_inactive.push_back(n);
    }
}
```

Selke Einzugsgebiet Modellieren

Die nützliche Hilfs-Funktion, die alle Knoten raussucht, die NICHT in nodes_activestehen.

```
bool NodeInList(int n, std::list<int>nodes_active)
{
    list<int>::const_iterator p = nodes_active.begin();
    while(p!=nodes_active.end())
    {
        if(n==*p)
            return true;
        ++p;
    }
    return false;
}
```

Selke Einzugsgebiet Modellieren

- ▶ Wir überzeugen uns vom Ergebnis (File schreiben) ...

```
std::ofstream inactive_nodes_file;
inactive_nodes_file.open("InactiveNodes.txt");
for(i=0;i<nodes_inactive.size();i++)
{
    inactive_nodes_file << nodes_inactive[i] << endl;
}
inactive_nodes_file.close();
```

OGS-DE: Selke Einzugsgebiet

- ▶ ... und natürlich graphisch, wozu haben wir denn Visual C++ gelernt!

