Goethe-Center for Scientific Computing (G-CSC)
Goethe-Universität Frankfurt am Main

Modelling and Simulation I

(Practical SIM1, WS 2016/17)

M. Hoffer, Dr. S. Reiter, Dr. A. Vogel

Aufgabenblatt 8 (Abgabe: Mo., 16.1.2016, 10h)

Bitte beachten Sie die Hinweise zur Installation von **ug4** auf der Vorlesungshomepage.
Für Hinweise im Umgang mit der Skriptsprache *Lua* beachten Sie bitte das nach der Installation von 'Examples' auf Ihrem Rechner verfügbare Skript `ug4/apps/Examples/lua-programming.lua`.

Hinweis: Um Konflikte mit der Versionskontrolle zu vermeiden sollten Sie heruntergeladene Skripte zunächst kopieren und dann die kopierte Version editieren. Für jede Aufgabe sollten Sie dabei ein separates Skript erstellen. Sollte es bei Aktualisierungen zu Problemen kommen, können Sie die problematischen Skripte löschen und nochmals aktualisieren. Stellen Sie dann sicher, zuvor eine Sicherheitskopie des betroffenen Skripts erstellt zu haben.

Bitte aktualisieren Sie Ihre ug4 Version vor dem Bearbeiten der Aufgaben, indem Sie in Ihrem ug4 Ordner oder einem Unterordner folgendes ausführen:

```
ughub gitpull
```

Bitte schicken Sie erstellte Skripte, Bilder und Ausgaben an

`practical.sim1@gcsc.uni-frankfurt.de`

Aufgabe 1 (Simulation 16P)
Mittels einer Simulation soll die Ausbreitung einer giftigen Substanz in einem Fluss untersucht werden. Als mathematisches Modell soll dazu folgende Konvektions-Diffusionsgleichung mit Reaktionsterm genutzt werden:

$$\frac{\partial c}{\partial t} = \nabla(D\nabla c - vc) - rc,$$

mit Diffusionskonstante $D \in \mathbb{R}$, Geschwindigkeitsfeld $v : \mathbb{R}^3 \to \mathbb{R}^3$ und Reaktionsrate $r \in \mathbb{R}$.

Die giftige Substanz fließe dabei über ein Abwasserrohr am Ufer in den Fluss und löse sich im Wasser mit einer Halbwertszeit von $T_{\frac{1}{2}} := 60s$ auf.

- Der betrachtete Flussausschnitt hat eine Breite von 10m (Ufer zu Ufer), eine Tiefe von 3m sowie eine Länge von 100m. Der Fluss fließe dabei von West nach Ost.
- Das Zentrum des Abwasserrohrs ist 10m Flussabwärts vom westlichen Einflussrand entfernt. Das Rohr treffe direkt unterhalb der Oberfläche in den Fluss. Der Einfachheit halber habe das Abwasserrohr einen quadratischen Querschnitt (Breite und Höhe 1m) und rage nicht in den Fluss hinein.
- Die Fließgeschwindigkeit des Flusses sei für dieses Beispiel an der Oberfläche entlang der mittleren Längsachse $0.2\frac{m}{s}$ und nehme linear zu den Ufern sowie zum Grund des Flusses auf $0\frac{m}{s}$ ab.
- Die Diffusionskonstante des Wassers betrage für dieses Beispiel $D = 0.01\frac{m^2}{s}$.
- Die Substanz fließe mit $10\frac{g}{m^2 s}$ in den Fluss.
- Am westlichen Einflussrand wird die Konzentration als 0 angenommen und soll entsprechend festgesetzt werden.

Erstellen Sie ein Konvektions-Diffusions basiertes Skript um den stabilen Zustand des Systems zu berechnen. Gehen Sie dazu von dem in Blatt 7 Aufgabe 4 erstellten Konvektions-Diffusions Skript aus.
Ersetzen Sie in `solverDesc` die Zeile

```
smoother = ''jac'',
```

durch

```
smoother = ''gs'',
```

(ca Zeile 117), da dies für die Konvergenz des linearen Lösers in dieser Anwendung benötigt wird.

Die Halbwertszeit einer Substanz kann über den *'reaction_rate'* Term der Konvektions-Diffusionsgleichung realisiert werden:

```
elemDisc:set_reaction_rate(r)
```

wobei r $(\frac{1}{s})$ den Faktor angibt, mit dem die Konzentration der Substanz pro Sekunde abnimmt.

Eine Einflussrandbedingung können sie im Skript wie folgt erstellen:

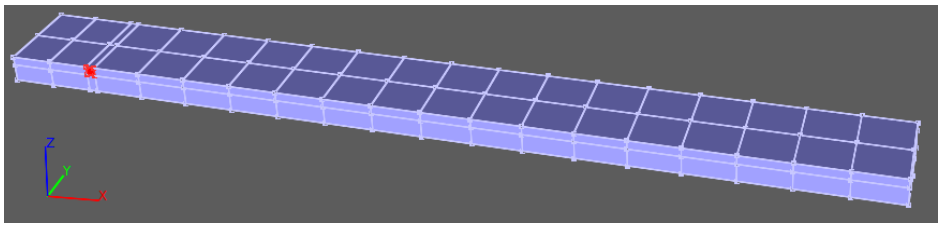
```
neumannBnd = NeumannBoundary(''c'')
neumannBnd:add(v, boundarySubset, innerSubset)
domainDisc:add(neumannBnd)
```

wobei `v` eine Zahl sei, die die Größe des Flusses angebe, `boundarySubset` den Namen des Rand-Subsets angebe, für das der Einfluss definiert wird und `innerSubset` der Name des inneren Subsets sei, in das die Substanz hinein fließt. Beachten Sie, dass Normalen in diesem Zusammenhang stets nach außen zeigen, also vom 'innerSubset' weg.

Das Geschwindigkeitsfeld kann in 3d analog zu Blatt 7 über eine Callbackfunktion mit folgender Signatur implementiert werden:

```
function CustomVelocity(x, y, z) ... end
```

Das Gitter auf dem Sie Ihre Simulation ausführen könnte in etwa wie folgt aussehen (reines Hexaedergitter mittels Extrusion erstellt):



Bitte passen Sie im Simulationsskript die `requiredSubsets` entsprechend den von Ihnen gewählten Subsetnamen an (ca Zeile 50).

Sollte der Lösungsvorgang zu lange dauern, können Sie die Zahl an Verfeinerungen über das Parameter '-numRefs' zwischenzeitlich herunter setzen. Um eine höhere Genauigkeit zu erreichen sollten Sie die Zahl an Verfeinerungen dann später wieder hoch setzen (empfohlen ca 3 bis 4).
Um eine höhere Ausführungsgeschwindigkeit zu erreichen sollten Sie außerdem das Speichern der Matrix sowie der rechten Seite deaktivieren.

Aufgabe 2 (Visualisierung 4P)
Visualisieren Sie Ihr Ergebnis mit ParaView (www.paraview.org). Nutzen Sie das *Contour*-Tool, z.B. mit 50 Contour-Flächen und logarithmischer Skalierung. Unter 'File-SaveScreenshot' können Sie die Visualisierung speichern.

Erstellen Sie außerdem ein Schaubild der Konzentration der giftigen Substanz entlang einer Linie die vom westlichen bis zum östlichen Ende des betrachteten Flussabschnitts in 2m Abstand vom Südufer und 1m unter der Oberfläche verläuft. Nutzen Sie dazu das *Plot Over Line* Tool in ParaView.

In welchem Bereich (Abstände zum Westlichen Einflussrand) ist der Fluss als vergiftet anzusehen, wenn eine Konzentration von $0.5\frac{g}{m^3}$ als giftig angesehen wird?

Nutzen Sie das Tool *IntegrateVariables* um die Gesamtmenge der im Fluss befindlichen Substanz zu berechnen.

Bitte beachten Sie, dass unterschiedliche Tools in ParaView nicht immer gut miteinander harmonieren oder dass unerwünschte Nebeneffekte auftreten können. Entfernen Sie in diesem Fall jeweils nicht genutzte Tools aus der Visualisierungskette.

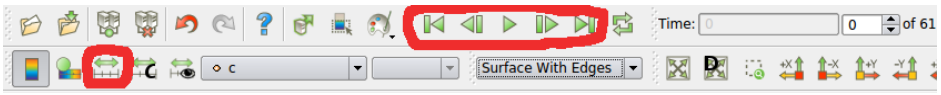
Aufgabe 3 (Freiwilliger Zusatz)
Ersetzen Sie in Ihrem Simulationsskript die Zeilen

```
solver:init(A, u)
solver:apply(u, b)
```

durch

```
startTime = 0
endTime = 1500
dt = 25
util.SolveLinearTimeProblem(u, domainDisc, solver, VTKOutput(),
          ''sol_river'', ''ImplEuler'', 1, startTime, endTime, dt);
```

Anstatt der Lösung im Gleichgewichtszustand zu suchen wird nun mittels des impliziten Eulerverfahrens eine zeitabhängige Simulation durchgeführt. Der aktuelle Zustand nach jedem Zeitschritt wird dabei in Dateien "sol_river_..." geschrieben. Laden Sie diese Dateien in ParaView (wählen Sie die Datei mit '.pvd' Endung) und Visualisieren Sie diese. Nutzen Sie dabei die Werkzeuge zum Abspielen von Animationen (oben mitte) sowie das Werkzeug 'Rescale to Data Range' (links).



Mittels 'File-SaveAnimation' können Sie Ihre Animation als Video speichern.