OpenFLUID

Software Environment
for Modelling Fluxes in Landscapes

mhydas

Quick tutorial

Geo-MHYDAS v12.01

LISAH
Laboratoire d'étude des Interactions Sol - Agrosystème - Hydrosystème

*Rabotin, M.,
and the modelling
group at LISAH*

Contents

Foreword **5**

1 General princips **7**

2 Copying LOCATION example in GRASS GIS and data visualisation **9**
2.1 Opening *vcatch* location . . . 9
2.2 Data visualisation . . . 10

3 Processing geographical objects **13**
3.1 Dissolving small area features on field layer : m.dispolyg . . . 13
3.2 Dissolving small linear features on reach layer : m.disline . . . 14
3.3 Splitting lines on reach2 layer : m.segline . . . 14
3.4 Snapping reach3 layer on field2 boundaries : m.snaplp . . . 14
3.5 Verifying network geometry : m.network . . . 15

4 Creating hydrological units by selective overlay **17**
4.1 Segmentation procedure : m.seg . . . 17
4.2 Dissolving sliver areal features on segmented1 layer : m.sliverpolygseg . . . 17
4.3 Dissolving small areal features on segmented2 layer : m.dispolygseg . . . 18
4.4 Patching columns from input field2 to segmented3 layer : m.colseg . . . 18
4.5 Extracting linear units from segmented3 layer : m.extractlineseg . . . 18
4.6 Patching columns from input reach to reach_seg layer : m.colseg . . . 18

5 Building oriented topologies of hydrological units **19**
5.1 Topology calcul for reach_seg2 : m.toporeach . . . 19
5.2 Topology calcul for segmented4 : m.toposu . . . 19
5.3 Create fluidx files for OpenFluid 1.6 : m.definput . . . 20

Foreword

This quick tutorial will help you to prepare your geographical data for the Digital Landscape Representation (DLR) for the model MHYDAS running with the platform OpenFLUID 1.6.x with Geo-MHYDAS 12.01.

It will not explain the concept behind the software, nor the scientific approaches in landscape fluxes modelling.

For more informations, please visit OpenFLUID website

Please read also : Lagacherie, P., Rabotin, M., Colin, F., Moussa, R. and Voltz, M., 2010. Geo-MHYDAS: A landscape discretization tool for distributed hydrological modeling of cultivated areas, Computers and Geosciences, 36, p.1021 - 1032

The user is supposed to be an experienced user of GRASS GIS software.

Typographic conventions

The "to note" informations are emphasized like this:

"to note" information with blue background...

The source code examples are emphasized like this:

```
Source code, with grey background and fixed size font
```

The "warning" informations are emphasized like this:

"warning" informations with red background...

Chapter 1

General princips

The whole Digital Landscape Representation (DLR) procedures can be divided in three steps:

- processing of the geographical objects
- creating hydrological units by selective overlays
- building oriented topologies of hydrological units

These segmentation procedures are made for MHYDAS model running with the OpenFluid platform 1.6 or superior. Ubuntu operating system and GRASS GIS Software version 6.3 or superior are needed (for information concerning GRASS installation, please read README file in the Geo-MHYDAS installation package).

A virtual catchment is used to illustrate a whole segmentation process. The following geographical object datas used are:

- dem.asc : a Digital Elevation Model (DEM)
- soil.shp : a simplified soil map
- field.shp : the delineation of the fields included in the catchment
- reach.shp : the delineation of the reach network of the catchment

These layers are available in *shapes* directory and can be visualised into your favorite GIS viewer.

Chapter 2

Copying LOCATION example in GRASS GIS and data visualisation

To avoid importing data into GRASS, a GRASS location example with the whole data needed has already been created. You can find the *vcatch* location into the LOCATION directory

The default directory for GRASS database is supposed to be /home/$USER/grassdata. If not yet create, please create this directory.

The example LOCATION *vcatch* must be unzipped and paste in /home/$USER/grassdata.

The different data layers contained in this LOCATION are :

- dem : DEM raster
- reach : reach network vector layer
- field : fields vector layer
- soil : simplified soil vector layer

2.1 Opening vcatch location

The first step is the opening of the location "vcatch" in GRASS
Launch GRASS in a terminal.

```
grass
```

A new interface appears.

Choose *vcatch* location and *PERMANENT* mapset, and click on **Start GRASS** button to enter into the mapset.

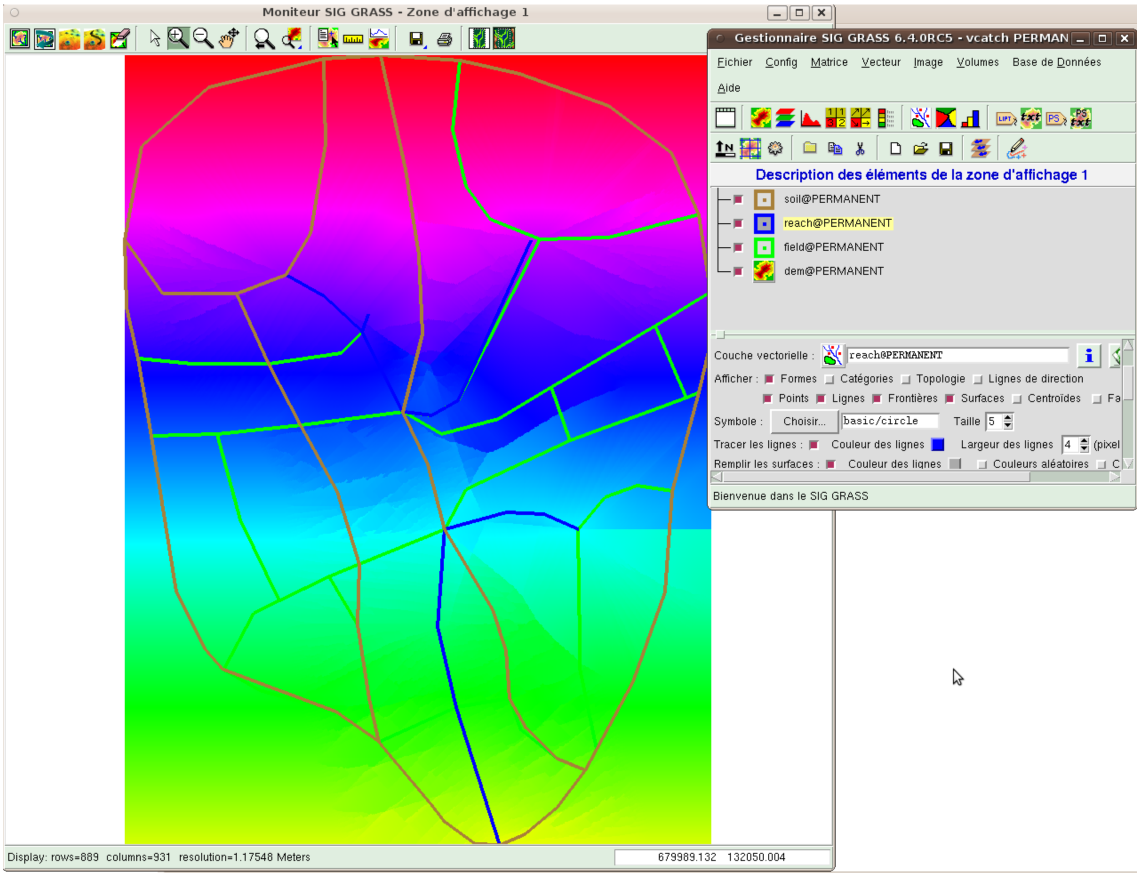
> In this manual, we will use the *gis.m* interface to visualise the data (the new python interface can be use in replacement)

2.2 Data visualisation

To open the *gis.m* interface, write :

```
gis.m
```

Add the different data layers in gis.m to visualise them :


Moniteur SIG GRASS - Zone d'affichage 1
Gestionnaire SIG GRASS 6.4.0RC5 - vcatch PERMAN
Fichier Config Matrice Vecteur Image Volumes Base de Données
Aide
Description des éléments de la zone d'affichage 1
soil@PERMANENT
reach@PERMANENT
field@PERMANENT
dem@PERMANENT
Couche vectorielle : reach@PERMANENT
Afficher : Formes Catégories Topologie Lignes de direction
Points Lignes Frontières Surfaces Centroïdes
Symbole : Choisir... basic/circle Taille 5
Tracer les lignes : Couleur des lignes Largeur des lignes 4 (pixel
Remplir les surfaces : Couleur des lignes Couleurs aléatoires
Bienvenue dans le SIG GRASS
Display: rows=889 columns=931 resolution=1.17548 Meters
679989.132 132050.004

Chapter 3

Processing geographical objects

The first step of the DLR process is the processing of the geographical objects. The user is able to create or modify datas. 7 algorithms are availables on this step :

- *m.disline* : dissolving small linear entities
- *m.dispolyg* : dissolving small areal entities
- *m.douglas* : smoothing areal entities (Douglas Peucker algorithm)
- *m.network* : hydrological network verification (rooted tree)
- *m.sbw* : subwatershed calculation program with hydrological network influence
- *m.segline* : splitting linear entities by points
- *m.snaplp* : snapping linear entities on areal entities

Only the *m.network* is compulsory to ensure that the reach network is correct.

In this manual, we will not use the *m.sbw* and *m.douglas* scripts

After each procedure, you can visualise data layers on *gis.m*

3.1 Dissolving small area features on field layer : m.dispolyg

We want to have only fields with an area superior at 7000 m^2, all the fields under this threshold must be dissolved. Use *m.dispolyg* command with flag -i to print statistics and exit:

```
m.dispolyg -i input=field output=field2 unit=me area=7000
```

The algorithm will dissolve 1 feature on 18 total features. Write again this command without flag -i to launch the algorithm

```
m.dispolyg input=field output=field2 unit=me area=7000
```

This new layer *field2* has now 17 features.

3.2 Dissolving small linear features on reach layer : m.disline

For the reach network, we want to have only reach with 30 m length at minima. Use *m.disline* command with flag -i to print statistics and exit:

```
m.disline -i input=reach output=reach2 unit=me length=30
```

The algorithm will dissolve 2 features on 8 total features. Write again this command without flag -i to launch the algorithm

```
m.disline input=reach output=reach2 unit=me length=30
```

This new layer *reach2* has now 6 features.

3.3 Splitting lines on reach2 layer : m.segline

For the reach2 layer, we want to split one reach with a specific point. Point coordinates will be provided by an ascii file called *point*

```
echo "679310 132649" > ~/point
```

The algorithm *m.segline* is used to split lines :

```
m.segline input=reach2 output=reach3 file=~/point snap=1
```

This new layer *reach3* has 7 features.
We update the column of the new segment of *reach3*

```
v.db.select map=reach3
v.db.update map=reach3 column=idreach value=7 where="cat=7"
v.db.update map=reach3 column=width value=0.7 where="cat=7"
v.db.update map=reach3 column=Ks value=0.0005 where="cat=7"
v.db.select map=reach3
```

3.4 Snapping reach3 layer on field2 boundaries : m.snaplp

To make coincide reach geometry with field geometry, *m.snaplp* algorithm is used to snap reach segments on field boundaries. We suppose that a threshold of 15 meters is sufficient.

```
m.snaplp input=reach3 output=reach4 polygon=field2 snap=15
```

3.5 Verifying network geometry : m.network

To ensure that reach network has a rooted tree structure, launch *m.network* command

```
m.network -c input=reach4 output=reach5
```

Chapter 4

Creating hydrological units by selective overlay

This second step is the creation of the hydrological units by selective overlays. The following algorithms are available :

- *m.colseg* : patch desired columns from the input polygon vector to the segmented output vector
- *m.dispolygseg* : dissolving small areal entities (with hierarchical level) after segmentation
- *m.extractlineseg* : linear units extraction after segmentation
- *m.seg* : segmentation procedure
- *m.sliverpolygseg* : dissolving sliver entities with hierarchical level after segmentation (use Gravelius Index)

4.1 Segmentation procedure : m.seg

This algorithm will create the hydrological units layer by overlaying the input layers with a hierarchical order. For this example, the following hierarchical order is : 1) landscape use, 2) soil data and 3) reach network. Creation of the *segmented1* layer.

```
m.seg input=field2,soil,reach5 output=segmented1 snap=0.2
id=idfield,idsoil,idreach
```

4.2 Dissolving sliver areal features on segmented1 layer : m.sliverpolygseg

This algorithm has same concept than *m.dispolyg* but it takes care about the hierarchical order of the features when dissolving them and works with feature forms (sliver forms). We choose an index equal at 1.8.

```
m.sliverpolygseg -i input=segmented1 output=segmented2 index=1.8 unit=me
```

This algorithm will dissolve 2 polygon on 25 total polygons
Launch the algorithm

```
m.sliverpolygseg input=segmented1 output=segmented2 index=1.8 unit=me
```

4.3 Dissolving small areal features on segmented2 layer : m.dispolygseg

This algorithm has same concept than *m.dispolyg* but it takes care about the hierarchical order of the features when dissolving them. We choose a minimum area of 5000 m^2.

```
m.dispolygseg -i input=segmented2 output=segmented3 unit=me area=5000
```

This algorithm will dissolve 1 polygon on 23 total polygons. Launch the algorithm

```
m.dispolygseg input=segmented2 output=segmented3 unit=me area=5000
```

4.4 Patching columns from input field2 to segmented3 layer : m.colseg

We want to add the "num_field" and "Ks" columns from the *field2* layer to the segmented3 layer

```
m.colseg input=field2 segmented=segmented3 output=segmented4
columns=Ks,num_field val=5
```

4.5 Extracting linear units from segmented3 layer : m.extractlineseg

This algorithm will extract the linear units from the segmented layer *segmented3* and the reach network *reach5*.

```
m.extractlineseg input=reach5 polygon=segmented3 output=reach_seg
```

4.6 Patching columns from input reach to reach_seg layer : m.colseg

We want to add the "width" and "Ks" columns from the *reach* layer to the reach_seg layer

```
m.colseg input=reach segmented=reach_seg output=reach_seg2 columns=width,Ks
val=5
```

Chapter 5

Building oriented topologies of hydrological units

This third step will calculate the RS and SU topologies for MHYDAS model. The several procedures are :

- *m.toporeach* : Topology calcul for reach segments RS
- *m.toposu* : Topology calcul for surface units SU
- *m.definput* : Create fluidx files for OpenFluid Engine 1.6

5.1 Topology calcul for reach_seg2 : m.toporeach

This algorithm will calcul the topology for reach_seg2 units We will calcul the reach_seg2 topology with "the most common" options; for more informations about these options, please read html help page for *m.toporeach* command

```
m.toporeach -c input=reach_seg2 output=reach_seg3 id=SELF_ID dem=dem outlet=1
```

You can visualize the topology information into the attribut table of reach_seg3

```
v.db.select map=reach_seg3
```

5.2 Topology calcul for segmented4 : m.toposu

We will calcul the segmented4 topology with "the most common" options; for more informations about these options, please read html help page for *m.toposu* command

```
m.toposu -s -l -d input=segmented4 output=segmented5 dem=dem hydro=reach_seg3
id=SELF_ID idhydro=SELF_ID pohydro=PCSS_ORD distreach=1
```

You can visualize the topology information into the attribut table of segmented5

```
v.db.select map=segmented5
```

5.3 Create fluidx files for OpenFluid 1.6 : m.definput

If you are using OpenFluid version 1.6 or superior, the xml files are replaced by fluidx files. Using *m.definput* algorithm allows you to create these fluidx files. For more informations about theses files, please read the OpenFLUID software documentation. after calculating topologies with *m.toporeach* or *m.toposu* algorithms
Creating the fluidx files for *reach_seg3* layer:

```
m.definput input=reach_seg3 directory=/home/$user type=RS column=width,Ks
```

The fluidx files are created on */home/* directory.

Creating the fluidx files for *segmented5* layer:

```
m.definput input=segmented5 directory=/home/$user type=SU column=Ks,num_field
```

The fluidx files are created on */home/* directory.