

Code avalanches coulantes AVAC : utilisation du module r.avac de GRASS GIS

Décembre 2020

Vincent Bain¹

¹Toraval France, 2838 route des Annuits, F-73400 Ugine

1 Préambule

Le code AVAC permet d'effectuer des simulations numériques d'écoulement d'avalanches coulantes. Il s'appuie sur la bibliothèque **CLAWPACK** et sa librairie **GEOCLAW**. Afin de faciliter la préparation des données sources du calcul et l'affichage des résultats, nous proposons de recourir à un module additionnel du système d'information géographique GRASS GIS (GRASS AddOn) nommé **r.avac**. Ce document fournit les instructions permettant sa prise en main rapide.

Le module fait appel à plusieurs programmes qu'on trouve couramment dans les dépôts des principales distributions GNU/Linux. Il n'a pas été testé pour les systèmes Mac OS et Windows. La solution la plus simple pour s'affranchir des problèmes de portabilité est probablement de travailler sur un poste tournant sous GNU/Linux, physique ou virtualisé (p. ex. Oracle VM ou VirtualBox).

2 Prérequis

On considère que l'utilisateur a suivi les instructions d'installation du code AVAC, c'est-à-dire qu'il dispose d'un environnement Clawpack fonctionnel, notamment :

- la variable d'environnement \$CLAW pointe sur le répertoire d'installation de Clawpack ;
- le fichier `qinit_module.f90` situé dans le répertoire `$CLAW/geoclaw/src/2d/shallow` a été correctement modifié (voir documentation AVAC).

Il faut installer les programmes GNU/parallel et OpenMP afin de bénéficier de la parallélisation des tâches.

Enfin le module **r.avac** requiert les fonctionnalités Temporal de GRASS GIS, donc il convient d'utiliser une version de GRASS supérieure à 7.0

3 Installation

Cloner le dépôt GitHub [ravac](#)

```
git clone https://github.com/xyleme/avac  
cd ravac
```

Le répertoire `ravac` comporte trois éléments :

- un répertoire nommé `avac`, qu'il faut déplacer dans le répertoire `examples/` de `geoclaw` :

```
mv avac/ $AVAC/geoclaw/examples/
```

- le fichier exécutable `r.avac`, qu'il faut déplacer dans un répertoire reconnu par GRASS GIS comme source d'AddOn. Le module dans sa version actuelle ne figure pas dans les dépôts officiels de GRASS GIS, il ne peut donc pas être installé depuis l'utilitaire `g.extension`.

Plutôt que de placer `r.avac` dans le répertoire dédié (`~/.grass7/addons/bin/`) on préfère le placer dans un répertoire distinct `/usr/local/grass-addons/`:

```
$ mkdir /usr/local/grass-addons/  
$ mv r.avac /usr/local/grass-addons/
```

puis déclarer la variable d'environnement correspondante dans le fichiers `.bashrc` (si le shell du système est bash), comme suit :

```
export GRASS_ADDON_PATH=/usr/local/grass-addons/
```

- le fichier `boussolenc.gpkg` qui contient un jeu de données cartographiques permettant d'effectuer une première exécution du code.

4 Préparation des données

Comme dit en préambule, la vocation essentielle de cet utilitaire est de simplifier la mise en œuvre du code AVAC en procurant à l'utilisateur des outils simples de préparation des données sources et de définition des paramètres du modèle. GRASS GIS fournit un cadre et des outils adéquats. L'utilisateur rompu aux SIG et en particulier à GRASS peut se reporter directement à la section suivante. Nous détaillons ici la marche à suivre pour préparer une première exécution du code.

4.1 Location et Mapset de travail

GRASS GIS s'exécute au sein d'une structure de fichiers déterminant un système de coordonnées cartographique de référence (répertoire dénommé *Location*) et des jeux de données (sous-répertoires *Mapsets*). Considérons que nous travaillons sur un versant situé dans les Alpes françaises, par conséquent nous nous plaçons dans le système de coordonnées légal Lambert 93, en créant une localisation L93. Lorsqu'on lance le programme en mode graphique pour la première fois, une suite d'étapes guide l'utilisateur dans la détermination d'un système de référence (voir figure 1). Le programme suggère de créer un mapset qui par défaut porte le nom de l'utilisateur.

Cette étape peut aussi être effectuée à la console avec la commande suivante :

```
grass79 -c EPSG:2154 L93
```

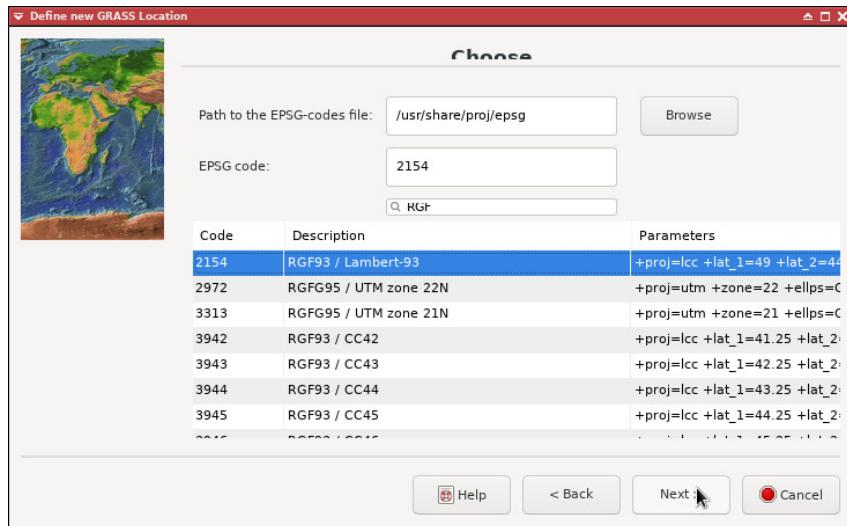


Figure 1 : La procédure de création de localisation de GRASS GIS invite l'utilisateur à préciser un système de coordonnées de référence.

GRASS GIS permet de gérer des données matricielles (raster), des données vectorielles (vector), des données attributaires. Le module `r.avac` fait appel à ces trois types de données.

4.2 Données raster : importation de données topographiques

Le fichier `boussolenc.gpkg` est un geopackage dont les données sont référencées dans le système Lambert 93. Il comporte un modèle numérique de terrain nommé `topo`, et une carte de relief ombré nommée `ombrage` que l'on va importer dans notre localisation en passant les commandes suivantes au terminal :

```
r.in.gdal input=boussolenc.gpkg gdal_doo='table=topo_2m' output=topo
r.in.gdal input=boussolenc.gpkg gdal_doo='table=ombrage' output=ombrage
```

Dans la fenêtre *Layer Manager* l'utilisateur peut ajouter ces cartes à la vue (voir figure 2).

Pour alimenter AVAC, `r.avac` transforme cette carte raster, qu'elle adressera au code sous la forme d'un fichier nommé `topo.asc`. La résolution et l'extension géographique du fichier sont contrôlées par l'utilisateur lors de la définition de la région de travail.

4.3 Notion de région

Une notion centrale de GRASS GIS est ce qu'on appelle la *région*. Il s'agit d'un ensemble de valeurs numériques indiquant au programme la position, l'étendue, et la résolution d'une grille qui discrétise la zone de travail. Par défaut dans une localisation vierge, cette région est définie par une seule cellule située à l'origine du repère cartésien de la projection courante. **Si l'on veut que GRASS traite nos données correctement il importe avant tout d'indiquer les paramètres de région correspondant à notre zone de travail.** Dans le cas présent il suffit de lancer la commande suivante pour faire coïncider la région avec l'extension et la résolution du modèle numérique de terrain importé à l'étape précédente :



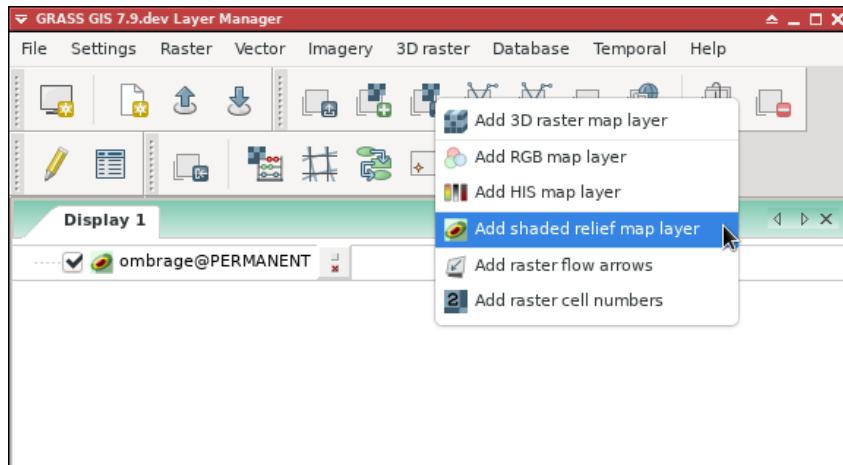


Figure 2 : Le menu *Add various raster map layers >Add shaded relief map* permet de combiner le relief ombré et le fichier topo.

```
g.region -p rast=topo
```

L'argument `-p` force l'affichage des nouveaux paramètres de région en sortie.

La fenêtre d'affichage des cartes permet de visualiser la région courante (voir figure 3). On peut aussi définir soi-même l'extension de la région de façon interactive dans la fenêtre d'affichage des cartes (icône Various zoom options >set computational region extent interactively).

Enfin est possible, et souhaitable, de fixer la résolution à laquelle nous souhaitons travailler. La commande précédente nous apprend que le pas d'échantillonnage de la topographie que nous avons importée est de 2 m ; on peut décider de vouloir travailler à un pas de 5 m, en donnant l'instruction suivante :

```
g.region -pa res=5
```

L'argument `-a` force la définition d'une région dont les dimensions sont des multiples de la résolution demandée.

La version actuelle de `r.avac` gère de façon rudimentaire la fonctionnalité AMR (adaptive mesh refinement) ; pour indiquer à `Geoclaw` de travailler à un niveau de grille supérieur sur une portion de la zone de calcul, on peut préciser au module (option `amr`) une zone rectangulaire qu'on aura préalablement déterminée dans `GRASS` au moyen d'une carte vecteur comportant un polygone dont l'identifiant `cat` porte la valeur du facteur `amr` souhaité (actuellement le code ne prend en charge que le niveau 2). Une manière commode de procéder est la suivante :

- définir comme précédemment l'extension d'une région de façon interactive dans la fenêtre d'affichage des cartes ;
- créer une carte vectorielle comportant un rectangle aux dimensions de la région ainsi définie, avec la commande

```
v.in.region cat=2 out=amr2 --o
```

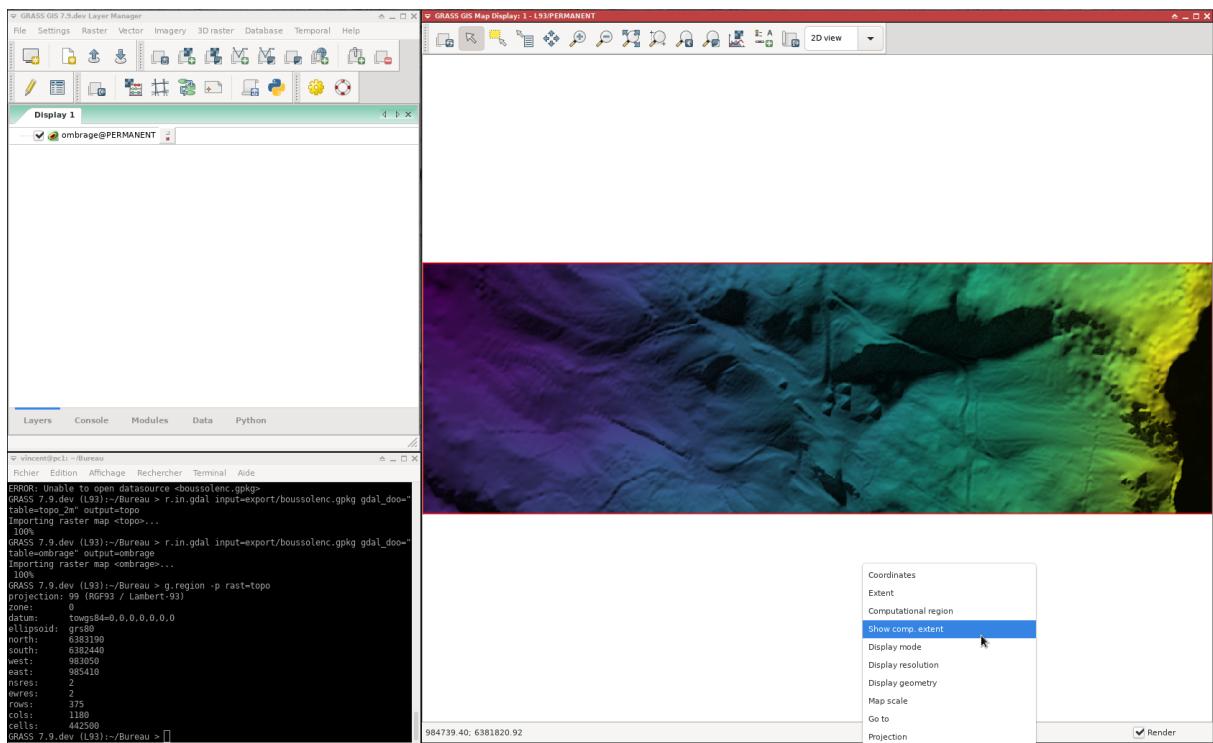


Figure 3 : affichage en relief ombré du modèle numérique de terrain. La fenêtre *map display* permet d'afficher la région courante sous la forme d'un cadre rouge.

4.4 Données vecteur : importation et création de données vectorielles

Le fichier `boussolenc.gpkg` comporte une couche vectorielle de courbes de niveau qui peut être importée comme suit :

```
v.in.ogr in=boussolenc.gpkg layer='cbn' out=cbn
```

Superposée à l'affichage des cartes raster précédentes, elle offre une meilleure visibilité du relief de notre jeu de données exemple.

Le code AVAC a besoin de connaître la localisation des zones de départ des avalanches, ainsi que la hauteur de neige mobilisée pour chacune. Le module `r.avac` récupère cette donnée initiale sous la forme d'une carte vecteur que nous nommerons `ZA`, comportant autant de polygones qu'il y a de zones de départ ; la hauteur de neige dans chacun des panneaux est portée par un attribut que nous nommons `h`.

Le fichier `boussolenc.gpkg` comporte un exemple de couche vectorielle nommé `ZA` répondant au format attendu par `r.avac`. Pour l'importer dans GRASS, exécuter la commande suivante :

```
v.in.ogr in=boussolenc.gpkg layer='ZA' out=ZA
```

Les commandes suivantes permettent de s'assurer que le champ `h` est bien présent dans la table attributaire de la couche `ZA` :

```
v.db.connect ZA -p
```

db.columns ZA

L'utilisateur se reportera à la documentation de GRASS GIS pour une présentation du modèle vectoriel¹ et attributaire², et l'utilisation des outils d'édition³. (voir figure 4)

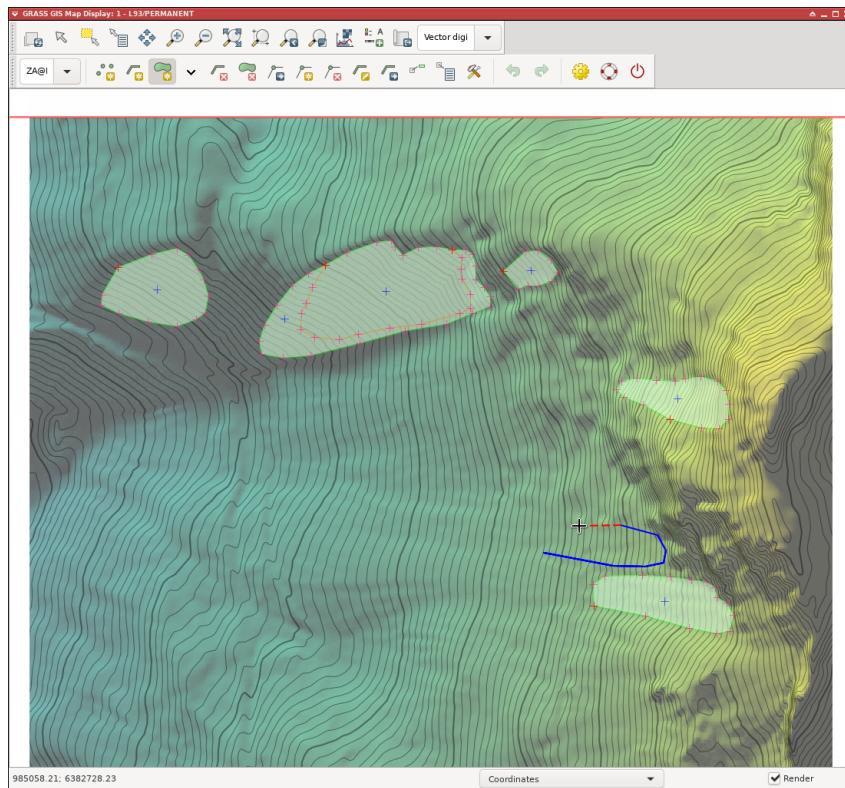


Figure 4 : La couche vectorielle ZA peut être éditée pour modifier, ajouter, supprimer des zones de départ.

Pour alimenter AVAC, r.avac discrétise cette carte vectorielle, qu'elle adressera au code sous la forme d'un fichier nommé `initials.xyz`.

4.5 Fonctionnalités de r.avac

Loi rhéologique Le code AVAC offre le choix entre deux lois de comportement de la neige. L'expression des frottements peut suivre :

- une loi de *Voellmy* (à deux paramètres μ, ξ), ou
- une loi de *Coulomb* (à un paramètre μ).

L'argument `rheology` de `r.avac` permet de choisir l'une ou l'autre de ces lois :

- `rheology=0` pour Voellmy, choix par défaut ;
- `rheology=1` pour Coulomb.

Le choix `rheology=1` rend inopérant le paramètre ξ .

1. <https://grass.osgeo.org/grass76/manuals/vectorintro.html>
2. https://grasswiki.osgeo.org/wiki/Vector_Database_Management
3. <https://grass.osgeo.org/grass76/manuals/wxGUI.vdigit.html>

Hauteurs de neige Dans sa dernière version r.avac procède à une modulation de la hauteur de neige fournie par le paramètre depth ou l’attribut h, en fonction de la raideur de la pente en chaque point de la zone de départ. Le module utilise une variation de la formule de Quervain :

$$h(z) = h_0(z) \frac{\sin \theta_{ref} - \nu \cos \theta_{ref}}{\sin \theta - \nu \cos \theta},$$

où h_0 est le cumul de neige à plat à une altitude z donnée, θ_{ref} est un angle de référence (pente pour laquelle le facteur de correction serait 1), θ l’angle de pente du panneau à l’altitude z , et ν un coefficient empirique. Les valeurs par défaut $\nu = 0,2$ et $\theta_{ref} = 30^\circ$ peuvent être modulées directement dans le code source de r.avac.

Cette version de r.avac permet également de définir plus finement la hauteur de neige dans la zone de départ : il est possible d’indiquer un gradient hypsométrique de hauteur de neige grâce à l’argument grad, qui se décline sous la forme d’un triplet (hauteur,gradient,altitude). Par exemple

```
grad=1.8,0.03,1850
```

signifie que la hauteur de neige vaut 180 cm à 1850 m d’altitude, et le gradient hypsométrique est de 3 cm par tranche de 100 m de dénivellation. Lorsque cette option est donnée à r.avac, la valeur calculée pour chaque maille de la grille écrase la valeur fournie par l’attribut h de la carte ZA.

5 Exécution

5.1 Syntaxe

La commande r.avac peut être lancée depuis le terminal en mode textuel, mais pour se familiariser avec la commande il est commode d’appeler l’interface graphique du module. Pour cela taper simplement dans le terminal :

```
r.avac --ui
```

Il suffit alors de renseigner correctement les arguments obligatoires figurant dans l’onglet *Required*. L’onglet *Optional* comporte des arguments dont les valeurs sont renseignées par défaut, notamment les paramètres de calage du modèle AVAC (voir documentation AVAC).

L'argument `--h` renvoie la syntaxe complète de la commande :

```
r.avac --h
```

Usage:

```
r.avac [-mrc] simul=name dem=name za=name [depth=name] [amr=name]
[dt=string] [t=string] [rheology=string] [rho=string] [xi=string]
[mu=string] [grad=string] [vthres=string] [bslope=string]
[--overwrite] [--help] [--verbose] [--quiet] [--ui]
```

Flags:

- m compute max(h) and max(p) maps throughout simul strds
- r reset existing simul strds
- c run geoclaw 'make clobber'
- a run r.avac in amnesiac mode (make sure -m flag is on)

Parameters:

simul	output simulation STRDS
dem	input elevation raster map
za	input starting area vector map
depth	depth column for starting area
amr	input AMR zone map
dt	time increment default: 20
t	simulation duration default: 100
rheology	type of rheology default: 0
rho	snow density default: 300
xi	Voellmy xi default: 1500
mu	Voellmy mu default: 0.15
grad	depth hypsometric gradient
vthres	velocity threshold default: 0.05
bslope	beta slope default: 1.1

Avec un peu d'habitude, l'utilisateur lancera `r.avac` depuis la console GRASS. Par exemple :

```
r.avac -mr simul=sim2 rheology=0 dem=topo za=ZA dt=1 t=40 \
mu=0.2 xi=800 rho=300 grad=1.7,0.03,1850 amr=amr2 vthres=0.2
```

L'intérêt d'invoquer la ligne de commande plutôt que l'interface graphique est que l'on peut facilement faire plusieurs itérations du code en changeant un seul paramètre, et ce en profitant du rappel de l'historique des commandes au prompt GRASS.

5.2 Fonctionnement

Lorsque la commande est lancée, le script se charge d'alimenter le code AVAC avec les données topographiques (topo.asc), les conditions initiales (initials.xyz), l'extension géographique du calcul (conforme aux paramètres de région fixés précédemment), les paramètres du calcul (voellmy.data, AddSetrun.py, et setrun.py).

Le calcul effectué, r.avac traite les fichiers `fort.qxxxx` produits par AVAC afin d'importer dans GRASS les paramètres $h(t)$ et $p(t) = \frac{1}{2}\varrho u^2$, où ϱ et u sont respectivement la densité de la neige et la vitesse. Suit un travail d'assemblage des multiples grilles générées par le code pour obtenir une série de rasters décrivant l'écoulement au pas de temps `dt` indiqué en entrée. En reprenant l'exemple de commande précédent, les rasters `h.sim2_xxxx` et `p.sim2_xxxx` sont incorporés dans deux séries temporelles (spatio-temporal raster dataset, strds), l'une nommée `sim2_h` décrivant la hauteur d'écoulement (exprimée en m), l'autre nommée `sim2_p` décrivant la pression au sein de l'écoulement (exprimée en kPa).

Si l'utilisateur souhaite interrompre l'exécution du code, il peut le faire en mode graphique en appuyant sur le bouton Stop ; en mode console c'est le raccourci `<ctrl+c>` qui envoie le signal d'interruption. Le script effectue des opérations de nettoyage avant de quitter (purge de certains paramètres, suppression de fichiers temporaires).

6 Visualiser, exporter

L'utilisateur peut visualiser le résultat de la simulation en lançant le module interactif `g.gui.animation`.

On peut aisément modifier la charte colorimétrique associée à chaque série. Par exemple si on souhaite appliquer la charte des pressions utilisée dans les documents Toraval, qui est consignée dans le fichier `p_toraval.file` joint à notre archive :

```
t.rast.colors input=sim5_h rules=[/path_to/]p_toraval.file
```

Le module `g.gui.animation` propose l'export d'animations sous plusieurs formes. Lorsque le nombre de cartes de la série est grand (typiquement un millier d'itérations), il n'est plus possible de visualiser ni d'exporter la série. L'utilisateur peut alors choisir d'exporter une série d'images au format png en vue d'un post-traitement avec d'autres outils, comme par exemple l'utilitaire `ffmpeg` :

```
g.list rast | grep h.sim2 | parallel r.out.png in={} out={}
ffmpeg -f image2 -r 12 -i "h.sim2_0\%03d.png" -vcodec png out_h.avi
```

```
g.list rast | grep p.sim2 | parallel r.out.png in={} out={}
ffmpeg -f image2 -r 12 -i "p.sim2_0\%03d.png" -vcodec png out_p.avi
```

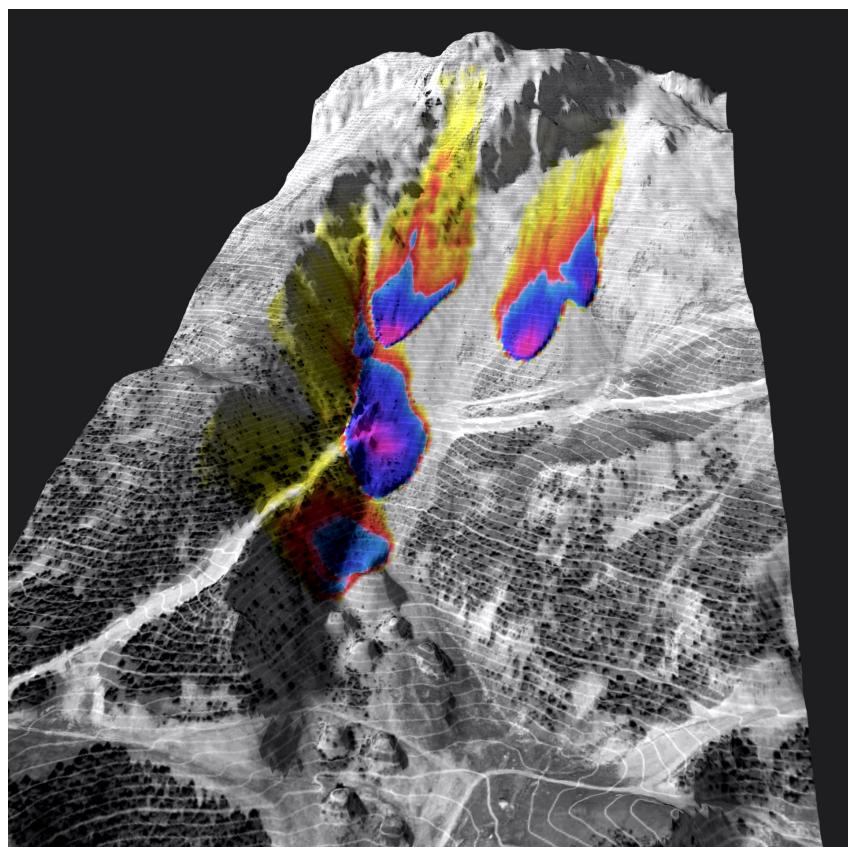


Figure 5 : Exemple de post-traitement réalisé avec Blender pour produire une animation combinant hauteur d'écoulement (déplacement) et pression (couleur). Voir la vidéo à l'adresse <http://telec.toraval.fr/boussolenc.avi>