

Documentation of *FullSWOF_2D* v1.07.00 (2016-03-14)

Christian LAGUERRE, Christian.Laguerre@math.cnrs.fr

2016-02-10

Contents

1	Presentation of the <i>FullSWOF_2D</i> software	2
2	Software distribution	2
2.1	How to download <i>FullSWOF_2D</i> ?	2
2.2	License	2
2.3	Installation	2
2.4	Check for proper functioning	3
3	Input and output values	3
3.1	<i>Inputs</i> directory	3
3.1.1	Space and time scales	3
3.1.2	Boundary conditions	4
3.1.3	Friction law	4
3.1.4	Numerical scheme	5
3.1.5	Infiltration	5
3.1.6	Topography	6
3.1.7	Initial water height and velocity	6
3.1.8	Rain	8
3.1.9	Name of output directory	9
3.1.10	Format of the output files	9
3.1.11	Comments	10
3.1.12	Advised values	10
3.2	<i>Outputs</i> directory	10
4	For developers	11
4.1	Debugging	11
4.2	Check for performances	11
4.3	Doxygen	12
5	Validation	13
5.1	Steady-state solutions	13
5.1.1	Emerged bump at rest	13
5.1.2	Fluvial Mac Donald test with rain and Darcy-Weisbach friction coefficient	14
5.1.3	Torrential Mac Donald test with rain and Darcy-Weisbach friction coefficient	14
5.1.4	Mac Donald test with smooth transition and shock, with Manning friction coefficient	14
5.1.5	Mac Donald pseudo-2D solutions	15
5.2	Transitory solutions	16
5.2.1	Dam break on a dry domain without friction	16
5.2.2	Thacker test case with planar surface in paraboloid	17

1 Presentation of the *FullSWOF_2D* software

The name *FullSWOF_2D* stands for “Full Shallow Water equations for Overland Flow in two dimensions of space”. In this software, the Shallow Water (or Saint-Venant) equations are solved using finite volumes and numerical methods especially chosen for hydrodynamic purposes (transitions between wet and dry areas, small water heights, steady-state preservation ...). A graphic user interface is available at <https://sourcesup.renater.fr/projects/fullswof-ui/>.

For explanations concerning the numerical schemes and approximations, the reader is referred to Delestre et al. [2014], Delestre [2010], Delestre et al. [2009] and Delestre and James [2009]. For a precise description of the structure of the software in several classes, see the Doxygen file (refman.pdf in the *doc* directory) and Delestre [2008]. The structure of the source code is designed to make future evolutions easy: for example, a new friction law can easily be added in the libfriction library, by creating a new friction file.

If you plan to change the code of *FullSWOF_2D*, see Section 4 for explanations on how to use the benchmarks. Doing so, you should pay attention to the license (section 2.2)

2 Software distribution

2.1 How to download *FullSWOF_2D*?

The *FullSWOF_2D* software can be downloaded on the website <https://sourcesup.renater.fr/projects/fullswof-2d/>.

2.2 License

This software is distributed under CeCILL-V2 (GPL compatible) free software license. So, you are authorized to use the Software, without any limitation as to its fields of application.

If you make changes to *FullSWOF_2D* code, you are welcome to **contribute your changes to the main repository**, directly through the website (<https://sourcesup.renater.fr/projects/fullswof-2d/>) or by contacting its main developers (fullswof.contact@listes.univ-orleans.fr). You may prefer to distribute yourself the *Modified Software*. In such a case, we ask you to **change its name** in order to avoid confusion between your software and the original one. In such a case, pay attention to the text that follows.

The license authorizes you to distribute the *Modified Software*, in source code or object code form, provided that said distribution complies with all the provisions of the *Agreement* and is accompanied by:

- a copy of the Agreement,
- a notice relating to the limitation of both the Licensor’s warranty and liability,

and that, in the event that only the object code of the *Modified Software is redistributed*, you allows future users *access to the full source code of the Modified Software by indicating how to access it*, it being understood that the additional cost of acquiring the source code shall not exceed the cost of transferring the data.

For further explanation about this free software license, you should read the following links:

- http://www.cecill.info/licences/Licence_CeCILL_V2-fr.html (in French)
- http://www.cecill.info/licences/Licence_CeCILL_V2-en.html (in English)

2.3 Installation

Remark 1 *To windows’ users: please, look at the application note entitled “Using Cygwin to compile and run FullSWOF_1D, FullSWOF_2D or SWASHES under windows”.*

First unzip the archive of the software. When you are in the *FullSWOF_2D* directory, write the following lines:

```
make cleanall
```

```
make
```

For the first tests, you can use the *Exp01* directory, where you will find examples of inputs files:

```
cd Exp01
```

```
../bin/FullSWOF_2D
```

2.4 Check for proper functioning

FullSWOF_2D comes with a set of test cases used for benchmarking (see section 5). Each test case has its own directory, which initially contains:

- the analytic solution (file *analytic.dat*),
- the *FullSWOF_2D* input parameters (Inputs directory)
- the benchmark outcome as computed by the developers (files *comp_STANDARD_32bits.dat* and *comp_STANDARD_64bits.dat*).

Once the software is installed on your computer, it is worth checking its proper functioning. For this, simply run the command `make benchref32` or `make benchref64` (depending if your operating system is 32 or 64 bits). This will first compute all the test cases (the results of the computation will be stored in the *Outputs_REFERENCES* directories). Then, the differences with the analytic solutions will be computed and, finally, it will be checked if they are differences between your run and the one of the developers. It is expected no such difference will be found: hence, for each test case, you should get the diagnosis “Results are identical.”. Otherwise, please, contact the developers at fullswof.contact@listes.univ-orleans.fr.

3 Input and output values

Remark 2 *FullSWOF_UI*, a graphic user interface dedicated to *FullSWOF_2D*, is available at <https://sourcesup.renater.fr/projects/fullswof-ui/>.

When launched, *FullSWOF_2D* expects two subdirectories: one for the inputs, one for the outputs. In the following sections, the notation $\langle x \rangle$ stands for the tag corresponding to the x variable, whereas the square brackets $[\cdot]$ give the unit of the variable.

3.1 Inputs directory

You can set the values of most of parameters in the *parameters.txt* file, located in the *Inputs* directory (the values we advise to use for overland flow can be found in Section 3.1.12).

3.1.1 Space and time scales

First, you have to specify the **number of grid cells** $\langle N_{xcell} \rangle$ and $\langle N_{ycell} \rangle$ (in space) and the **length of the domain** along x and y ($\langle L \rangle$ [m] and $\langle l \rangle$ [m], respectively). For the time, you should set the value of the **duration of the simulation** $\langle T \rangle$ [s] and the **number of times saved** $\langle nbtimes \rangle$ (that is the number of pictures you will save to see the evolution as a movie).

If you do not want to save pictures, you can set $\langle nbtimes \rangle$ to 0.

You can run *FullSWOF_2D* either with a **constant CFL** $\langle cflfix \rangle$ or with **constant time step** $\langle dtfix \rangle$. The type of time constraint is defined by $\langle scheme_type \rangle$. You can choose either:

- case 1: fixed CFL value. You only have to specify the value of $\langle cflfix \rangle$. Usually, a CFL value equal to 0.4 is suitable.
- case 2: fixed time step (in seconds). In this case, you have to specify the values of both $\langle dtfix \rangle$ and $\langle cflfix \rangle$.

Indeed, it is always necessary to set a CFL value even if you choose the fixed time step because *FullSWOF* verifies at each loop in time that the time step is not greater than this CFL value in order to respect the stability condition.

3.1.2 Boundary conditions

For some boundary conditions, you may have to specify the discharge and the water height on the boundary. You must use the following principles:

Imposed discharges <left_imp_discharge>, <right_imp_discharge>, <bottom_imp_discharge> and <top_imp_discharge> [m^3/s] correspond to the discharges at the left, right, bottom and top boundaries, respectively. At the left (*i.e.* $x = 0$) and bottom (*i.e.* $y = 0$) boundaries, if you want an incoming flow, you have to impose a positive discharge whereas for an outgoing flow you must impose a *negative* discharge.

At the opposite boundaries (*i.e.* $x = x_{max}$ and $y = y_{max}$), impose a *negative* discharge for the inflow and a positive discharge for the outflow;

Imposed water heights <left_imp_h>, <right_imp_h>, <bottom_imp_h> and <top_imp_h> [m] correspond to the water heights at the left, right, bottom and top boundaries respectively.

You must choose the **left**, **right**, **top** and **bottom boundary conditions** (<Lbound>, <Rbound>, <Tbound> and <Bbound>):

- the case 1 corresponds to the imposed height boundary condition (based on the modified method of characteristics). **Imposed discharge and water height** have to be specified. Depending on the regime (sub/super-critical) and on the boundary (inflow/outflow), the value of the imposed discharge may not be used by *FullSWOF_2D*;
- the case 2 is for the wall condition;
- the case 3 is for the Neumann boundary condition, which means that the normal derivatives of the water height and velocity are null. Physically, this case represents an open boundary;
- the case 4 stands for the periodic case, in which the outflow on one boundary is considered as the inflow on the other boundary. In this case, at each time step, the incoming flow is set equal to the outgoing flow;
- in case 5, the value of the discharge is imposed. You must specify the **value of flow discharge** and the **specific water height**, but the latter will be considered only in supercritical cases.

3.1.3 Friction law

You have to impose the **friction law** <fric>. You can use either:

- case 0 to run *FullSWOF_2D* with no friction.
- case 1 to use the Manning law; in this case, the friction coefficient is the Manning friction coefficient [$\text{m}^{-1/3}\text{s}$].
- case 2 to choose the Darcy-Weisbach law; for this choice, the friction coefficient is the Darcy-Weisbach friction coefficient [dimensionless].
- case 3 to use the laminar law; in this case, the friction coefficient [m^2/s] is the laminar friction coefficient, equal to the triple of the kinematic viscosity ν .

If you choose case 1, case 2 or case 3, the value of the friction coefficient must be imposed either by using a file or by giving a value, according to the parameter <fric_init>:

- case 1 to load the friction coefficient from a file. The file name is specified by the parameter <fric_NF>. This file must be in the *Inputs* directory. It should be in ASCII, in the format “ $x \ y \ f$ ” in which x represents the coordinate along x , y represents the coordinate along y and f represents the friction coefficient.
- case 2 to use a constant friction coefficient over the whole domain. This value is set by the parameter <friccoef>.

Remark 3 For a spatially variable friction coefficient, if the variation is too important (in particular if a discontinuity occurs), numerical errors such as spurious peaks may appear. A possible way to overcome this problem is to implement a well-balanced scheme for the friction source term, see for example Bouchut [2004].

3.1.4 Numerical scheme

The next six parameters are related to the numerical scheme. You have to choose the:

Numerical flux <flux>. You can use either:

- case 1: Rusanov flux,
- case 2: HLL (Harten, Lax and Van Leer) flux,
- case 3: HLL2 flux, that is another way of programming HLL (unusual formulation but quicker resolution),
- case 4: HLLC (Harten, Lax and Van Leer with Contact Surface) flux,
- case 5: HLLC2 flux, that is another way of programming HLLC (unusual formulation but quicker resolution).

Remark 4 *The HLLC approximate Riemann solver is a modification of the basic HLL scheme to account for the influence of intermediate waves. Indeed, the HLLC solver restores the missing contact and shear waves in the HLL scheme.*

Order of the scheme<order>. You can use either:

- case 1: order 1,
- case 2: order 2.

Linear reconstruction <rec>. This parameter will play a role only for a scheme of order 2. It can be:

- case 1: MUSCL,
- case 2: ENO,
- case 3: a modified ENO.

<armortENO>. This parameter must be specified when you are using linear reconstructions ENO or modified ENO. Its value must be between 0 and 1 (usually set to 0.25). If you take 0, it will be equivalent to choose the MUSCL reconstruction; if <armortENO> is set equal to 1, the reconstruction will be exactly ENO or modified ENO, depending on your previous choice.

<modifENO>. For the modified ENO reconstruction, you have to set the <modifENO> parameter between 0 and 1, usually taken equal to 0.9.

Slope limiter <lim>. This parameter will play a role only for a scheme of order 2. You can use either:

- case 1: the classical minmod slope limiter.
- case 2: the more complicated expression of Van Albada.
- case 3: Van Leer's reconstruction.

3.1.5 Infiltration

In this version of *FullSWOF_2D*, a modified bi-layer Green-Ampt is the only infiltration model (see Esteves et al. [2000]). So, you have the choice between:

- case 0 to run *FullSWOF_2D* without infiltration.
- case 1: a modified bi-layer Green-Ampt infiltration model.

For each infiltration parameter you can either initialize it from a file or give constant value.

If you choose to parameterize the infiltration using files, you can include a spatial variability of the infiltration parameters. For this, you have to provide ASCII files in the format “*x y w*” and these files must be in the *Inputs* directory.

The modified bi-layer Green-Ampt infiltration model assumes the soil to be represented by two layers:

The first layer (top of the surface) represents a crust of **thickness <zcrust>** [m] and **hydraulic conductivity <Kc>** [m/s].

If you want to initialize these two parameters from a file, write the **name of the thickness file** in <zcrust_NF> and/or the **name of your hydraulic conductivity file** in <Kc_NF>.

The **second layer** has a **saturated hydraulic conductivity** $\langle Ks \rangle$ [m/s].

If you want to initialize this parameter from a file, write the **name of your saturated hydraulic conductivity file** in $\langle Ks_NF \rangle$.

The other parameters (common values for the two layers) are:

- the **saturated moisture content** θ_s $\langle dtheta \rangle$ [dimensionless] between 0 (dry) and 1 (fully saturated).

If you want to initialize this parameter from a file, write the **name of your saturated moisture content file** in $\langle dtheta_NF \rangle$.

- the **load pressure** ψ $\langle Psi \rangle$ [m].

If you want to initialize this parameter from a file, write the **name of your load pressure file** in $\langle Psi_NF \rangle$.

- the **maximum infiltration rate** i_{max} $\langle imax \rangle$ [m/s]. In the standard Green-Ampt model, at $t = 0$ s, if the cumulative volume of water infiltrated is zero, the infiltration capacity is infinite. To avoid this phenomena, the Green-Ampt model has been modified to be able to impose a maximum infiltration rate i_{max} .

If you want to initialize this parameter from a file, write the **name of maximum infiltration rate file** in $\langle imax_NF \rangle$.

3.1.6 Topography

The type of input is defined by $\langle topo \rangle$ which can take the following values:

- 1 to load the topography from a file. You have to enter the **topography** using an ASCII file you previously generated in the format “ $x y z$ ”. Write the **name of your topography file** in $\langle topo_NF \rangle$. This file must be in the *Inputs* directory.
- 2 to use a flat topography with $z = 0$;
- 3 to have the Thacker’s paraboloid defined by $z = -0.1 * (1 - \sqrt{(x - x_m)^2 + (y - y_m)^2})$ where (x_m, y_m) is the center point of the computational domain. This is used for the benchmark shown in section 5.2.2 and detailed in [Delestre et al., 2013, § 4.2.2].

Remark 5 FullSWOF_2D considers values (such as h, u, v, z) constant on a given cell, and the constant is given in the center of the cell, see figures 1 and 2.

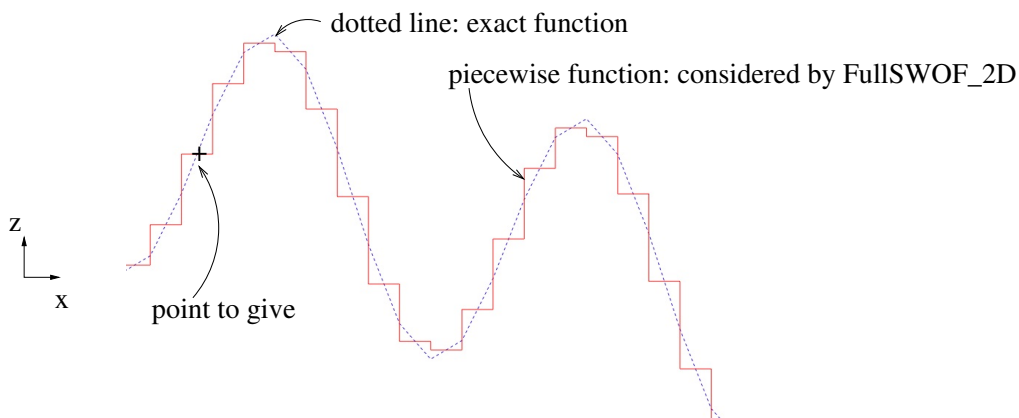


Figure 1: Piecewise approximation of 2D curves in *FullSWOF_2D*

3.1.7 Initial water height and velocity

You can impose the initial water height and velocity using $\langle huv_init \rangle$:

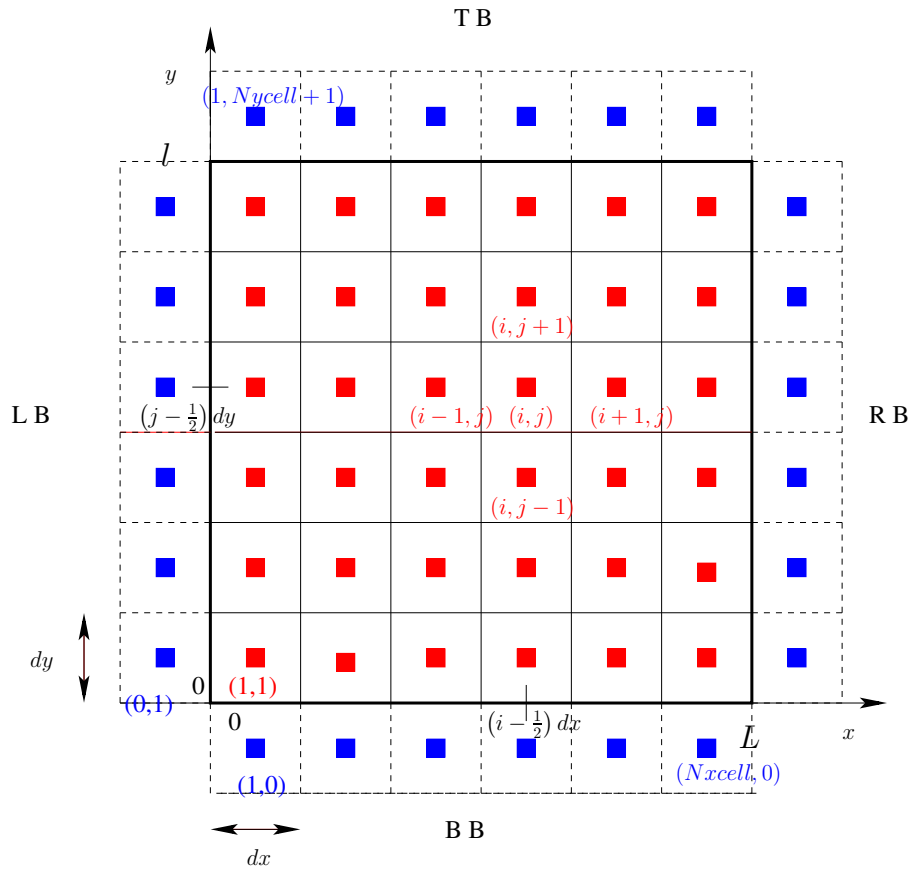


Figure 2: Mesh *FullSWOF_2D*;

LB= Left Boundary, RB= Right Boundary, BB= Bottom Boundary, TB= Top Boundary.

The computational domain is represented by the cells with the red center and the boundary corresponds to the cells with blue center.

- case 1 to load the initialization of variables h [m], u [m/s] (velocity in the x -direction) and v [m/s] (velocity in the y -direction) from a file. This file must be in ASCII and follow the format “ $x y h u v$ ”. The **name of the file** `<huv_NF>` should be specified and this file must be in the *Inputs* directory.
- case 2 to have $h = 0$ m, $u = 0$ m/s and $v = 0$ m/s.
- case 3 to have the initialization for the paraboloid used as a benchmark in section 5.2.2 and detailed in [Delestre et al., 2013, § 4.2.2].
You should choose the Thacker’s topography for this case.
- case 4 to have $h = 0.005$ m in a disk centered at $(L/2, l/2)$ (which is the middle of the computation domain $[0; L] \times [0; l]$) with a radius $L/10$. $h = 0$ m outside the disk, and $u = v = 0$ m/s on the whole domain. This case is used to simulate the dry radial dam break shown as a benchmark in section 5.2.1 (Figure 3) and detailed in [Delestre et al., 2013, § 4.1.2)].
You should choose the flat topography for this case.
- case 5 to have $h = 0.005$ m in a disk centered at $(L/2, l/2)$ (which is the middle of the computational domain $[0; L] \times [0; l]$) with a radius $L/10$. $h = 0.001$ m outside the disk, and $u = v = 0$ m/s on the whole domain. This case is used to simulate a wet radial dam (see Figure 4, and [Delestre et al., 2013, § 4.1.1]).
You should choose the flat topography for this case.

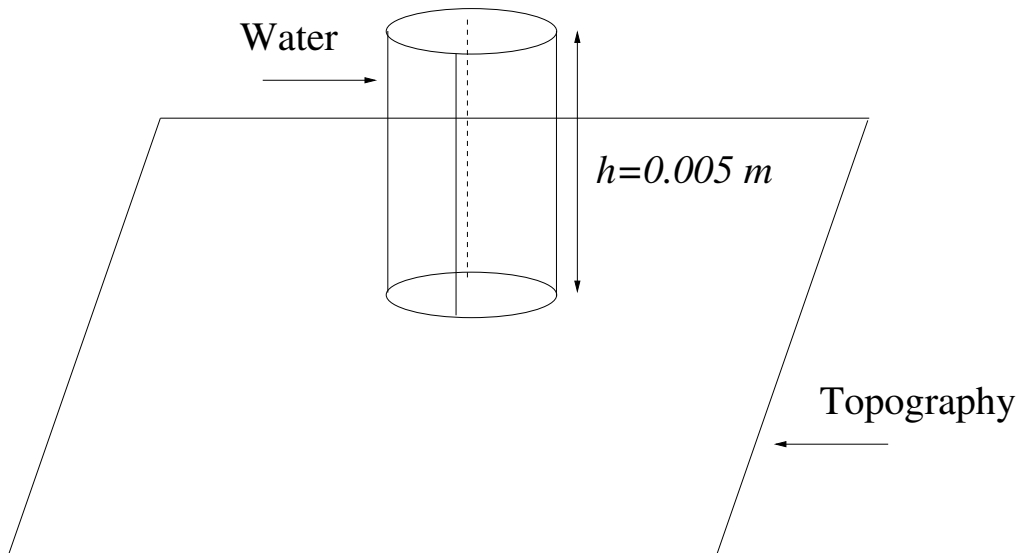


Figure 3: Initialization of h , u and v to simulate a dry radial dam: $h = 0$ m or $h = 0.005$ m, $u = 0$ m/s, $v = 0$ m/s.

3.1.8 Rain

For the **rain** `<rain>`, you can use:

- case 0: **No Rain.**
- case 1: An ASCII file. The **name of your rain file** `<rain_NF>` should be set and this file must be in the *Inputs* directory.

The file must contain two columns:

- the first one represents the time [s].
- the second one the rain intensity [m/s].

Each column must contain at least one value.

In the first column, the first value must be 0 (initial time). If the file contains one value by column (namely 0 in the first column and any value in the second column), the rain intensity is

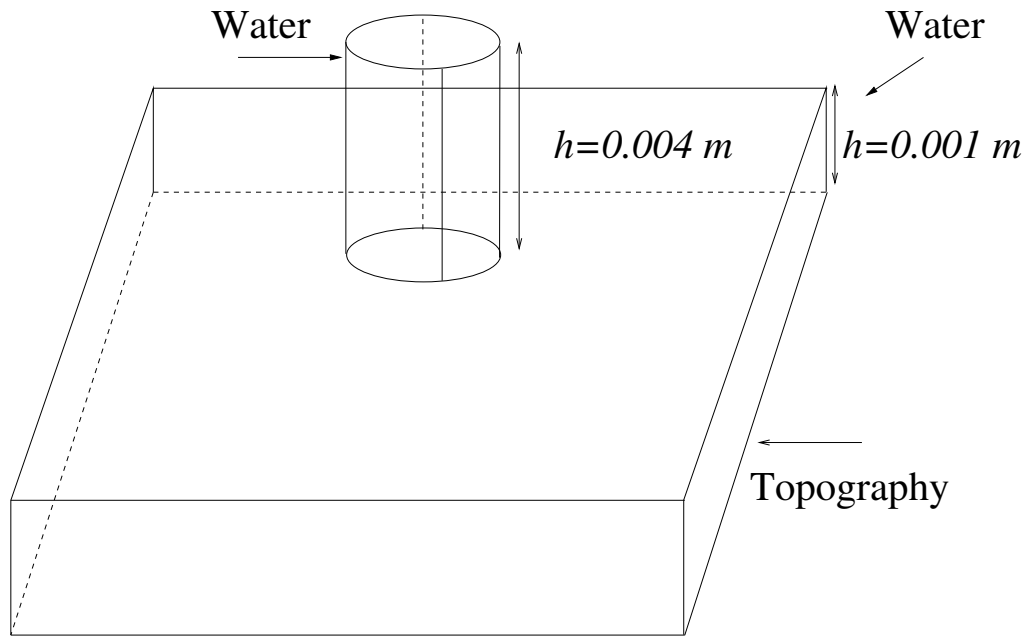


Figure 4: Initialization of h , u and v to simulate a wet radial dam: $h = 0.001$ m or $h = 0.005$ m, $u = 0$ m/s, $v = 0$ m/s.

constant during the run.

If the file contains several lines with different values, that means the rain intensity will be changed during the simulation. For example, let us consider three values for the time $(0, t_1, t_2)$ and the corresponding three values for the rain intensity (a_0, a_1, a_2) (written into the second column). Let us consider $Rain(t)$ the intensity of the rain at the time t , we will get the values:

$$Rain(t) = \begin{cases} a_0 & \text{for } 0 \leq t < t_1, \\ a_1 & \text{for } t_1 \leq t < t_2, \\ a_2 & \text{for } t_2 \leq t. \end{cases}$$

Be careful: the intensity may not change exactly at $t = t_1$ but at $t > t_1$. Indeed, whether using a constant CFL or a constant time step, t_1 might not be reached exactly. We can have $t < t_1$ for a given computing time, and the next computing $t + dt$ may be greater than t_1 . So, $t = t_1$ may not exist and the change in rain intensity will take place at $t + dt > t_1$.

- case 2: the rain intensity is constant during the run. The value is equals to 10^{-5} m/s that is equivalent to 36 mm/h.

Remark 6 You can create another case from, for example, the function in the `rain_generated.cpp` file (see the directory: `Sources/librain_infiltration/`).

3.1.9 Name of output directory

The default name of the output directory is `Outputs`. However, you can add a **suffix** to this name (`<suffix_o>`). This is especially useful if you are running several tests.

3.1.10 Format of the output files

You have the choice between two possibilities to save the evolution of the computed values:

- case 1: gnuplot format is designed for the gnuplot software¹. This ASCII file can be used to draw easily the output using gnuplot software.

¹<http://www.gnuplot.info>

- case 2: vtk format can be used for the display of the computed values with Paraview software². It is an ASCII file too.

3.1.11 Comments

You can also add comments after the input value of each parameter. For example:

```
Time of simulation <T>:: 0.001 # try 0.01 next time
```

3.1.12 Advised values

For overland flow, several tests have been performed, in particular on the numerical scheme Delestre [2010]. Consequently, we advise the user to choose by default the following numerical parameters:

- Choice of type of scheme (1=fixed cfl 2=fixed dt) <scheme_type>:: 1
- Value of the cfl <cflfix>:: 0.4
- Numerical flux (1=Rus 2=HLL 3=HLL2 4=HLLC 5=HLLC2) <flux>:: 5
- Order of the scheme <order>:: 2
- Reconstruction (1=MUSCL 2=ENO 3=ENOmod) <rec>:: 1
- Limiter (1=Minmod 2=VanAlbada 3=VanLeer) <lim>:: 1

If you choose the ENO or the modified ENO reconstruction, you should use:

- AmortENO <amortENO>:: 0.25
- ModifENO <modifENO>:: 0.9

3.2 Outputs directory

The results are saved in the *Outputs* directory. When starting the program, the following files are saved:

parameters.dat contains the parameters used by *FullSWOF_2D*, under the same format as the input file. In this file, the value of parameters will be empty or equal to zero if the user did not fill them in parameters.txt file.

huz_initial.dat contains the initial conditions (water height, velocity, topography and free surface).

During the computation, one file is modified in order to save several time steps:

huz_evolution.dat contains the evolution (in time) of the main variables on each cell (water height, velocity, free surface, topography, the Euclidean norm of the velocity, Froude number, discharge along x , discharge along y , the Euclidean norm of the discharge). There are <nbtimes> time steps saved.

If you set <nbtimes> to 0, *FullSWOF_2D* will not create the **huz_evolution.dat** file.

If the software did not stop before (meaning, in case of running with a fixed dt, the CFL maximum value has been reached, which implies that you must decrease the time step or increase the space step), the computation is done until the final time, and two other files are created:

huz_final.dat contains the main variables on each cell (water height, velocity, free surface, topography, the Euclidean norm of the velocity, Froude number, discharge along x , discharge along y , the Euclidean norm of the discharge) at the final time.

results.dat contains a mass balance to check the scheme conservation at the final time. More precisely, you get:

- Infiltrated volume [m^3] is the total infiltrated water volume cumulated during the simulation.
- Stream volume [m^3] is the total water volume staying at the final time above topography.
- Complete volume (Inf+Stream) [m^3] is the sum of infiltrated volume and stream volume.

²<http://www.paraview.org>

- Volume of the rain [m^3] is the total rain volume cumulated during the simulation.
- Outflow volume [m^3] is the sum of the cumulated outflow volume at all boundaries.
- Duration of the computation [s].
- Number of iterations in the algorithm [dimensionless].
- Mean Froude number [dimensionless] in space at the final time.

Finally, if you ran *FullSWOF_2D* in debug mode (see Section 4.1), four other files are saved:

check_vol.dat contains the cumulated volumes [m^3] at each time step. The first column is the time and the others are: the overland flow volume (Vol_of_tot), the infiltrated volume (Vol_inf_tot), the rain volume (Vol_rain_tot) and the balance of the input and output volumes at the boundaries (Vol_bound_tot).

Recall that the second order is computed thanks to a prediction-correction method. Hence, when the code is run at the second order, two fluxes are used to perform one time step. In this case, the two values printed at t^n are the cumulated volumes associated with the two fluxes used to compute the values at t^{n+1} .

boundaries_flux.dat contains the cumulated fluxes [m^2] for each boundary and at each time step.

When the code is run at the second order, two fluxes are used to perform one time step. In this case, the two values printed at t^n are the cumulated fluxes associated with the two fluxes used to compute the values at t^{n+1} .

flux_boundaries_LR.dat contains the fluxes [m^2/s] calculated at each cell of the left and right boundaries. When the code is run at the second order, the two values printed at t^n are the two fluxes used to compute the values at t^{n+1} .

flux_boundaries_BT.dat contains the fluxes [m^2/s] calculated at each cell of the bottom and top boundaries. Here again, when the code is run at the second order, the two values printed at t^n are the two fluxes used to compute the values at t^{n+1} .

4 For developers

4.1 Debugging

If you make some changes in *FullSWOF_2D*, you will need to debug your code. The default configuration is the release mode. To change it into debug mode, you must set the **DEBUG** value to **yes** in the **make_config** file. In that case, additional output files will be created to help you in your tests (see Section 3.2).

4.2 Check for performances

FullSWOF_2D comes with a set of test cases used for benchmarking (see section 5). Each test case has its own directory, which should contains:

- the analytic solution (file analytic.dat),
- the *FullSWOF_2D* input parameters (Inputs directory),
- the benchmark outcome as computed by the developers (files comp_STANDARD_32bits.dat and comp_STANDARD_64bits.dat),
- the benchmark outcome as computed on your computer after installation (file comp_REFERENCES.dat) — see section 2.4.

After you modify *FullSWOF_2D* code, you may want to check if the performances are the same, have been degraded (a bug?), or have been improved, compared to the capabilities of the software after installation. For this, simply run the command **make bench32** or **make bench64** (depending if your operating system is 32 or 64 bits). This will first compute all the test cases (the results of the computation will be stored in the Outputs directories). Then, the differences with the analytic solutions will be computed and, finally it will be checked if they are differences between your current run and the run upon installation.

Remark 7 *If, at some point, you do not want to compare the performances with the run upon installation anymore, you should delete the files `comp_REFERENCES.dat` and the directories `Outputs_REFERENCES`. Then, run `make benchref32` or `make benchref64` (depending if your operating system is 32 or 64 bits). The newly created `comp_REFERENCES.dat` file and `Outputs_REFERENCES` directories will be then used as references when using `make bench32` or `make bench64`.*

If there is no difference, you should get the diagnosis “Results are identical.” for each test case.

If differences are reported for one or more test cases, you may first want to look at the files `diff_REF_USER.dat` of the relevant test cases). After the header (lines starting with the # character), are listed a set of values:

The first column identifies the variable. DhSI stands for “Differences in water height in international system unit” (meter in the case of water height) while Dh% stands for “Differences in water height expressed as a percentage” (taking as a reference the case `Outputs_REFERENCES`). Differences in velocity starts with “Du” and differences in water flux starts with “Dq”.

The second column identifies the statistics for each variable. First is given the number of differences that cannot be computed (“`nbdiff==NaN`”), probably because it involves a division by zero. Then the number of differences equal to zero, larger than zero and smaller than zero. Following are the minimum, maximum, mean and median values, and finally the L^1 , L^2 and L^∞ norms.

The third column shows the absolute differences.

The fourth column shows the relative differences (as a percentage), taking as a reference the case `Outputs_REFERENCES`.

So, you should be able to identify if the differences concern numerous values (or just a few), and if they are about water height, velocity or flux. At this stage in the diagnosis, it may be time to dig further by comparing the content of the `Outputs` and `Outputs_REFERENCES` directories:

- First, check for uncalled-for differences in the `parameter.dat` files.
- Check that the initial data are identical (files `huz_initial.dat`).
- Locate the differences in the final results by comparing the files `huz_final.dat`.
- Eventually, compare the time evolution of these differences (files `huz_evolution.dat`).

Based on this, you should be able to evaluate if the software results are as accurate, more accurate or less accurate than before. Since each test case addresses specific flow conditions (see section 5), you should be able to build a rationale about which part of the simulation has been impacted, and, if required, which part of the source code is involved.

4.3 Doxygen

You may wish to add some functionalities to *FullSWOF_2D* to suits your needs. Always comment the files, at the beginning of the file, using Doxygen syntax (<http://www.doxygen.org>). Then, you will be able to create the Doxygen documentation of the whole code.

HTML documentation. In order to generate the Doxygen html file, the `Doxygen_config_file_html` file is saved in the `doc` directory. To run Doxygen, from the *FullSWOF_2D* directory, use the command:

```
doxygen doc/Doxygen_config_file_html
```

Warning: Graphviz (<http://www.graphviz.org/>) must be in your PATH to generate HTML diagrams. If not, change the `HAVE_DOT` parameter of the `Doxygen_config_file_html` file. In the `doc/html/` directory, `index.html` is created.

PDF documentation. To generate the Doxygen PDF file (using \LaTeX), you must use the `Doxygen_config_file_latex` file and compile the .tex file:

```
doxygen doc/Doxygen_config_file_latex
cd doc/latex
```

make

In the *doc/latex* directory, **refman.pdf** is created.

Your version of Doxygen should be 1.8.7 or greater.

Remark 8 *To simplify these operations, you can run the script `UpdateDateVersion` located in the `bin` folder.*

5 Validation

This software has been validated using several analytic solutions and benchmarks from the literature, gathered in Delestre et al. [2013] and in the SWASHES software (<https://sourcesup.renater.fr/projects/swashes/>). Some of them are already configured in the **Benchmarks** directory.

In this documentation, we recall the main characteristics of these tests, and give the results of *FullSWOF_2D*.

5.1 Steady-state solutions

In this section, we focus on a family of steady-state solutions, that are solutions satisfying:

$$\partial_t h = \partial_t u = \partial_t v = 0.$$

5.1.1 Emerged bump at rest

Here we present a steady-state case (see [Delestre et al., 2013, § 3.1.2]) with a flat topography at the boundaries, no rain, no friction and no diffusion. Initial conditions satisfy the hydrostatic equilibrium

$$h + z = Cst \text{ and } q = 0 \text{ m}^2/\text{s}. \quad (1)$$

This solution tests the preservation of steady state and the boundary condition treatment.

In the following case, we choose a domain of length $L = 25$ m with a topography given by:

$$z(x, y) = \max\left(0, 0.2 - 0.05(x - 10)^2 - 0.1(y - 10)^2\right)$$

The water height is smaller than the topography height in order to have emergence of some parts of the bump. In such a configuration, starting from the steady state, the velocity must be null and the water surface should stay flat (see Figure 5).

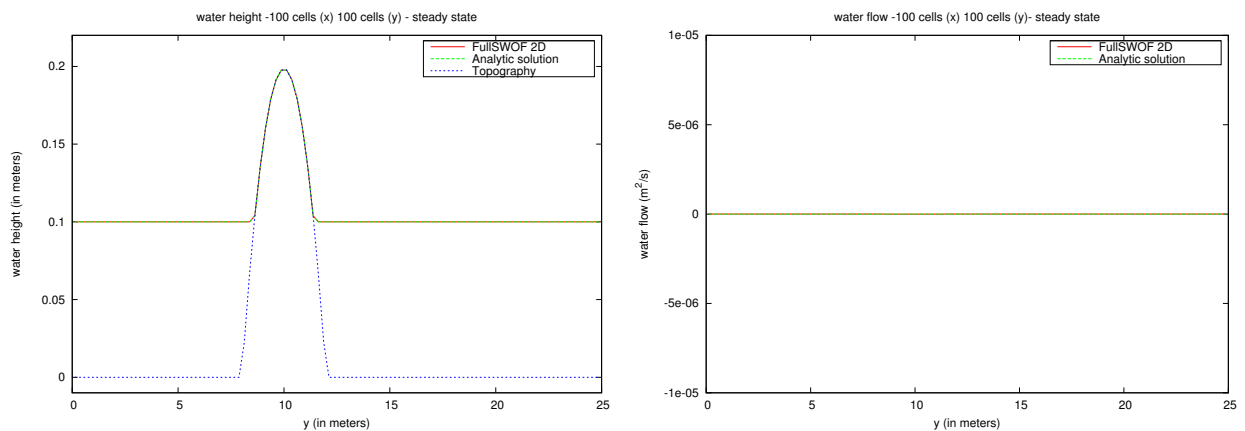


Figure 5: Emerged bump at rest

5.1.2 Fluvial Mac Donald test with rain and Darcy-Weisbach friction coefficient

For a 1000 m long channel, we consider a flow which is fluvial on the whole domain (see Figure 6 and [Delestre et al., 2013, § 3.3.1]). Thus we impose a constant discharge ($q = 1 \text{ m}^2/\text{s}$) at the inflow and a constant water height at the outflow.

Initially, the channel is dry and we choose Darcy-Weisbach friction law with $f = 0.093$. We also impose a constant rain intensity ($R_0 = 0.001 \text{ m/s}$) during the simulation.

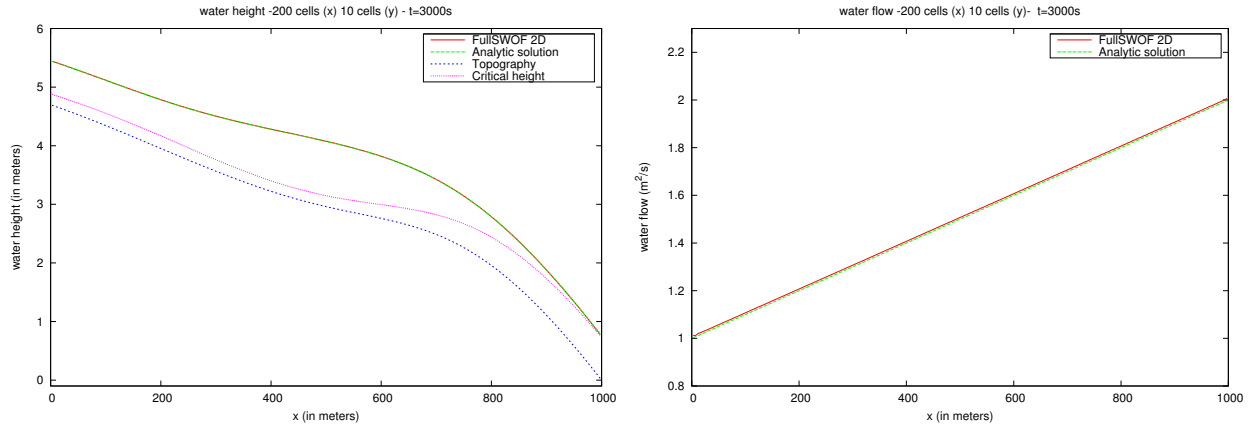


Figure 6: Fluvial Mac Donald Rain test with Darcy-Weisbach friction coefficient

5.1.3 Torrential Mac Donald test with rain and Darcy-Weisbach friction coefficient

The channel length remains unchanged (1000 m), but, as the flow is supercritical, we consider constant discharge ($q = 2.5 \text{ m}^2/\text{s}$) and water height at the inflow and a free outflow (see Figure 7 and [Delestre et al., 2013, § 3.3.1]). At initial time, the channel is dry and we choose Darcy-Weisbach friction law with $f = 0.065$.

In this benchmark, there is no rain until 1500 s and after the rain intensity is 0.001 m/s until the end.

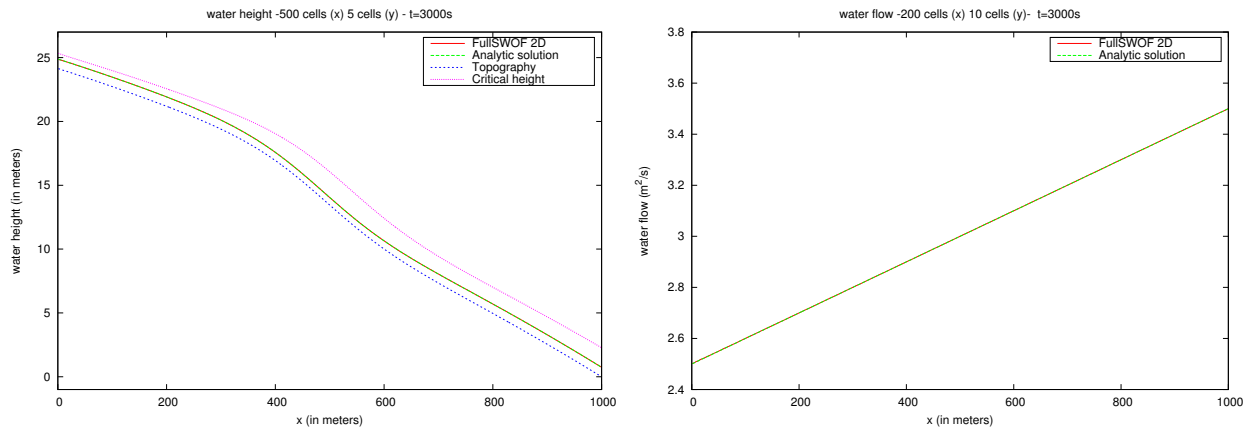


Figure 7: Torrential Mac Donald: Rain test with Darcy-Weisbach friction coefficient

5.1.4 Mac Donald test with smooth transition and shock, with Manning friction coefficient

The length of the channel is 100 m and the discharge at steady state is $q = 2 \text{ m}^2/\text{s}$. The flow is fluvial both upstream and downstream (see Figure 8 and [Delestre et al., 2013, § 3.2.2]). In this case, the

Manning friction coefficient is $n = 0.0328 \text{ m}^{-1/3}\text{s}$, the inflow is subcritical, becomes supercritical via a sonic point, and, through a shock (located at $x = 200/3 \approx 66.67 \text{ m}$), becomes subcritical again.

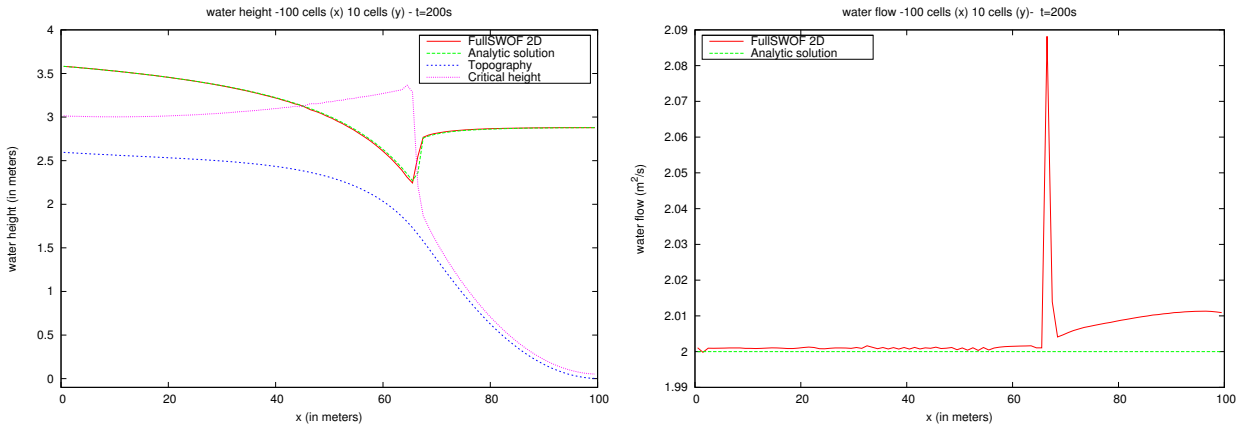


Figure 8: Mac Donald test with smooth transition and shock, with Manning friction coefficient

5.1.5 Mac Donald pseudo-2D solutions

In this section, we give several analytic solutions for the pseudo-2D Shallow-Water system (see [Delestre et al., 2013, § 3.5]). This system can be considered as an intermediate between the one-dimensional and the two-dimensional models. More precisely, these equations model a flow in a rectilinear three-dimensional channel with the quantities averaged not only on the vertical direction but also on the width of the channel.

We consider two cases for non-prismatic channels. Each channel is determined through the definition of the bottom width B (as a function of the space variable x) and the slope of the boundary Z . The bed slope is an explicit function of the water height.

Torrential Mac Donald pseudo-2D test with Manning friction coefficient In this case, the flow ($q = 20 \text{ m}^3\text{s}^{-1}$) and the water height are set at the inflow whereas the outflow is free (see Figure 9 and [Delestre et al., 2013, § 3.5.2]).

The channel is initially dry and the flow is supercritical during the simulation.

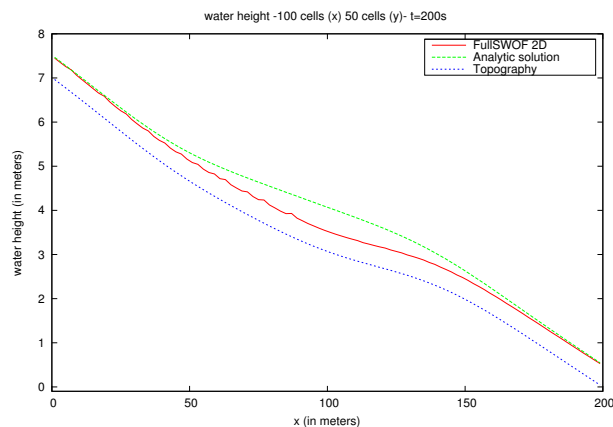


Figure 9: Torrential Mac Donald pseudo 2D test with Manning friction coefficient

Fluvial Mac Donald pseudo-2D test with Manning friction coefficient Now, the length of the domain is $L = 400 \text{ m}$, the boundaries of the channel are given by B_2 and the cross sections are isoscele trapezoids (see Figure 10 and [Delestre et al., 2013, § 3.5.5]).

In this case, the flow is imposed at the inflow ($q = 20 \text{ m}^3\text{s}^{-1}$) and the water height is prescribed at the outflow (see Figure 11). The flow is subcritical along the whole channel.

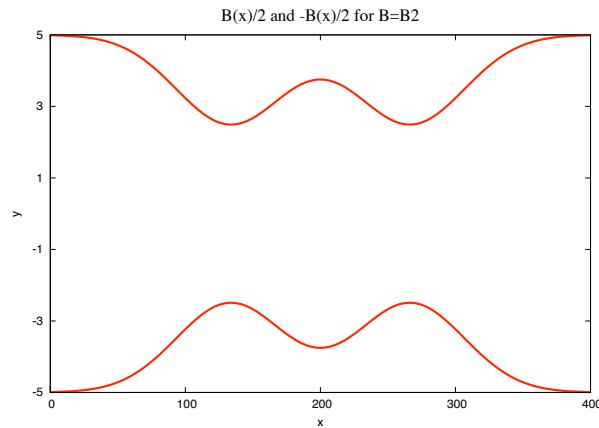


Figure 10: Function B2 for the shape of the channel in the fluvial Mac Donald pseudo-2D test.

The channel is initially dry, with a little puddle downstream.

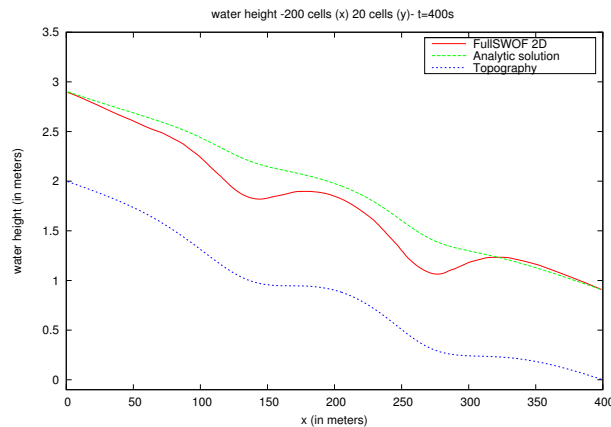


Figure 11: Fluvial Mac Donald pseudo 2D test with Manning friction coefficient

5.2 Transitory solutions

In the previous section, we gave steady-state solutions of increasing difficulties. These solutions can be used to check if the numerical methods are able to keep/catch steady-state flows. But even if the initial condition differs from the expected steady-state, we do not have information about the transitory behavior. Thus, in this section, we give some transitory solutions that may improve the validation of the numerical methods. Moreover, as these cases have wet/dry transitions, one can check the ability of the schemes to capture the evolution of these fronts (*e.g.* some methods may fail and give negative water height). At last, we give a periodic transitory solution in order to check whether the schemes are numerically diffusive or not.

5.2.1 Dam break on a dry domain without friction

In this section, we are interested in a dam break solution on a flat topography namely Ritter's solution (see [Delestre et al., 2013, § 4.1.2]). The initial condition for this configuration is the following Riemann problem

$$h(x, y) = \begin{cases} 0.005 & \text{for } 0 \text{ m} \leq y \leq L - x, \\ 0 & \text{for } y > L - x, \end{cases}$$

with $u(x, y) = v(x, y) = 0$ m/s. The comparison between the solution given by *FullSWOF_2D* and analytic solution is shown in Figure 12.

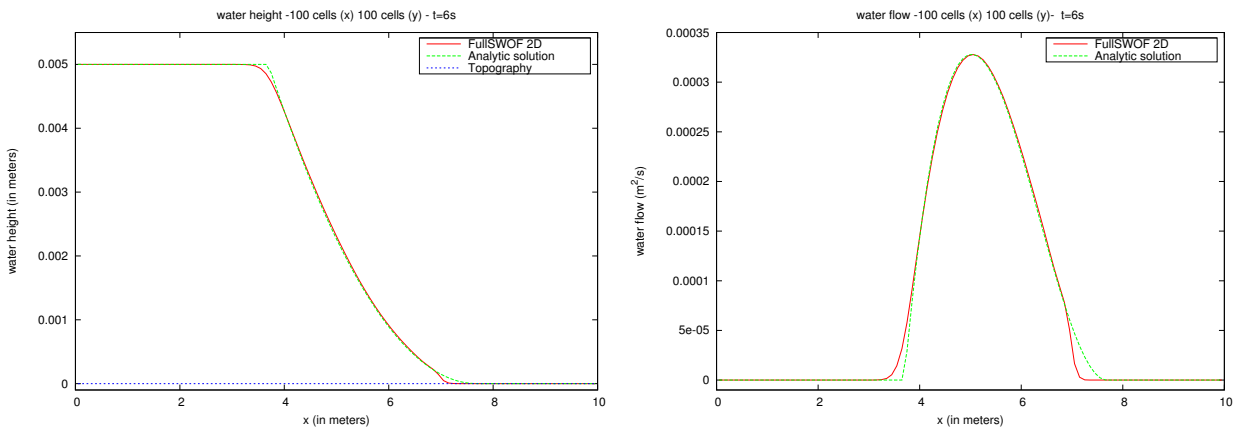


Figure 12: Dam break dry without friction

5.2.2 Thacker test case with planar surface in paraboloid

For this Thacker 2D case, the moving shoreline is a circle and the topography is a paraboloid of revolution. The free surface has a periodic motion and remains planar in time (see [Delestre et al., 2013, § 4.2.2]). To visualize this case, one can think of a glass with some liquid in rotation inside (see Figure 13).

Here again, the analytic solution at $t = 0$ s is taken as initial condition.

This is an analytic solution with a variable slope (in space) for which the wet/dry transitions are moving. It also tests the ability of the schemes to simulate flow with comings and goings and, as the water height is periodic in time, the numerical diffusion of the scheme.

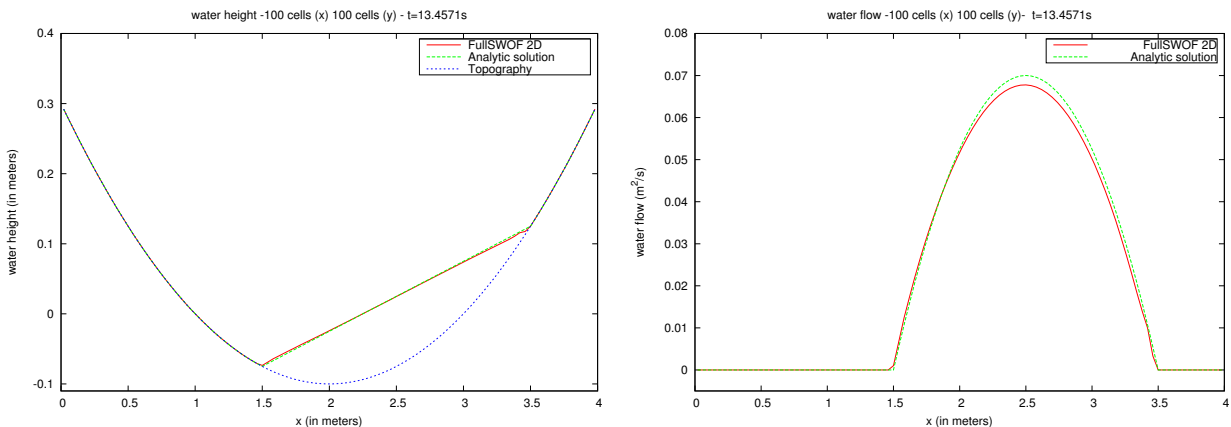


Figure 13: Thacker test case with planar surface in paraboloid

References

F. Bouchut. *Nonlinear stability of finite volume methods for hyperbolic conservation laws, and well-balanced schemes for sources*, volume 2/2004. Birkhäuser Basel, 2004. doi: 10.1007/b93802.

Olivier Delestre. *Ecriture d'un code C++ pour la simulation en hydrologie*. Master's thesis, Université d'Orléans, Orléans, France, September 2008. <http://dumas.ccsd.cnrs.fr/dumas-00446163/en/>.

- Olivier Delestre. *Simulation du ruissellement d'eau de pluie sur des surfaces agricoles*. PhD thesis, Université d'Orléans, Orléans, France, July 2010. <http://tel.archives-ouvertes.fr/tel-00531377/en/>.
- Olivier Delestre and François James. Simulation of rainfall events and overland flow. In *Proceedings of X International Conference Zaragoza-Pau on Applied Mathematics and Statistics, Jaca, Spain*, Monografías Matemáticas García de Galdeano, 2009. <http://hal.archives-ouvertes.fr/hal-00426694/en/>.
- Olivier Delestre, Stéphane Cordier, François James, and Frédéric Darboux. Simulation of rain-water overland-flow. In E. Tadmor, J.-G. Liu, and A. Tzavaras, editors, *Proceedings of the 12th International Conference on Hyperbolic Problems*, volume 67 of *Proceedings of Symposia in Applied Mathematics*, pages 537–546, University of Maryland, College Park (USA), 2009. Amer. Math. Soc.
- Olivier Delestre, Carine Lucas, Pierre-Antoine Ksinant, Frédéric Darboux, Christian Laguerre, Thi Ngoc Tuoi Vo, François James, and Stéphane Cordier. SWASHES: a compilation of shallow water analytic solutions for hydraulic and environmental studies. *International Journal for Numerical Methods in Fluids*, 72(3):269–300, 2013. ISSN 1097-0363. doi: 10.1002/fld.3741. URL <http://hal.archives-ouvertes.fr/hal-00628246>.
- Olivier Delestre, Frédéric Darboux, Francois James, Carine Lucas, Christian Laguerre, and Stephane Cordier. FullSWOF: A free software package for the simulation of shallow water flows. 30 pages, January 2014. URL <http://hal.archives-ouvertes.fr/hal-00932234>.
- Michel Esteves, Xavier Faucher, Sylvie Galle, and Michel Vaucelin. Overland flow and infiltration modelling for small plots during unsteady rain: numerical results versus observed values. *Journal of Hydrology*, 228:265–282, 2000.