TatooineMesher: Anisotropic interpolation from 1D cross-sections and 2D channel mesher

Luc Duron, François-Xavier Cierco
Hydraulic Engineering Department
Compagnie Nationale du Rhône
Lyon, France
l.duron@cnr.tm.fr

Khaled Saad
Numerical modelling and high performance computing
University of Bordeaux
Bordeaux, France

***Abstract* — A set of scripts named TatooineMesher were developed at CNR to provide tools for the modeller to mesh rivers and interpolate bathymetry from discrete cross-section data for TELEMAC-2D calculations. The principles are also applied to the visualization of 1D model results (including MASCARET) as the surface reconstruction problem from discrete data is the same. The scripts made available for the community, are written in the Python programming language and rely on many standard scientific libraries.**

The meshing and interpolation processes are described separately as they are implemented as such for the sake of versatility. The meshing part consist of a regular mesher (structured or not) following constraint lines and including many parameters to control density and positions of nodes. Different anisotropic interpolation methods (following the flow direction) to interpolate data between discrete cross-sections are available to the modeler.

I. CONTEXT

A. Aims of the developed scripts

The developed scripts can be applied to the following framework:

- pre-treatment for 1D models: interpolate intermediate cross-sections,
- post-treatment of 1D model: visualize results in 2D, especially in the framework of a coupling with TELEMAC-2D,
- pre-treatment for TELEMAC-2D river model: interpolate bathymetry and/or mesh river bed.

B. Description of meshing issues

Hydraulic numeric modelling aims to estimate in every point of time and space the hydraulic variables such as: water discharge, water heights and velocities. The horizontal spatial discretization is done on a mesh (or grid) which consists of a set of triangular elements connected together. The mesh quality is crucial as the accuracy (including the diffusion terms) and the computation time are directly affected. Therefore, the definition of the mesh is adapted by the modeller to its needs to have a satisfactory compromise between accuracy and computation cost.

The main issues to mesh a river are the integration of longitudinal constraint lines, lateral discretization of dykes and integration of structures (such as bridge piers and groynes). TatooineMesher generates a mesh (possibly not structured) with regular element sizes which are controlled by the user through some files (for polylines) and options (at least longitudinal and lateral target edge lengths).

C. Description of interpolation issues

Hydraulic numerical simulation requires a large amount of topographic data to build an accurate digital elevation model. The bathymetry being often more complex to measure on permanently inundated river beds, long river reaches are usually only surveyed at some regular longitudinal position and discrete 1D cross-sections of the river channel are obtained.

With discrete cross-sections data, measurement points are not regularly spaced and the resolution of the cross-section profiles is generally much higher in the lateral than in the longitudinal direction of the river. That is why isotropic interpolation methods should not be applied and methods considering the anisotropy of the bathymetry were investigated.

The interpolation being crucial to estimate bathymetry between cross-sections, several 1D and 2D interpolation methods (linear, cubic…) were implemented within TatooineMesher and compared (see part IV.B).

D. Overview of interpolator/mesher existing tools

In framework of 2D hydraulic modelling a large range of tools are available to mesh and/or interpolate a domain. In the TELEMAC community the commonly adopted software are: BlueKenue, Salome-Hydro (SMESH), SMS Mesh Module, Janet and GMSH [1].

Although they are often fast to compute and easy to use (they often include a GUI), they are often affected by some limitations such as:

- for those providing a channel mesher (e.g. BlueKenue, Salome-Hydro):

- mesh cannot be controlled by constraint lines,
- mesh is fully structured and not suited in case of longitudinal variation of the channel width,

- definition of constraint lines is restricted (number or proximity limitations, they must cross all cross-sections...),
- the interpolator is absent or is based on isotropic approaches.

In the context of surface reconstruction, a constraint line is a longitudinal polyline defining a change in the bathymetry. This notion is used in TatooineMesher to constraint element edges to pass along these lines and to guide the interpolation.

TatooineMesher aims to fill the lack of functionalities of the available software. Moreover, in the investigated tools, the channel interpolator is often too simple, and the mesh generation cannot scripted (except for GMSH). For all these reasons, TatooineMesher was developed as a standalone tool and cover different needs (see part I.A).

II. Mesh Generation

A. General mesh specifications

The mesh generation process has to follow some rules in order to be optimized for further computations and to correspond to the modeller needs. Usual mesh specifications are listed below (not-exhaustive, for more details on mesh quality see [2]):

- nodes preferably located along cross-sections (to limit interpolation),
- nodes preferably located along constraint lines,
- elements edges do not cross constraint lines,
- element size and number of nodes controlled by the user (to optimize the computation time),
- elements are equilateral or possibly elongated along flow direction,
- the transition of element size is progressive.

B. Steps of the channel mesh generator

The mesh generation is splitted into the 4 main following steps.

Step 1: Order cross-sections

1.1. Cross-sections are ordered from upstream to downstream. They are located with their curvilinear abscissa along the hydraulic axis,

1.2. Cross-sections which do not cross the hydraulic axis are ignored.

Step 2: Intersect cross-sections and constraint lines

The following steps describe how the domain delimited by two consecutive cross-sections and two constraint lines is meshed, in a so-called submesh (as see in Figure 1).

Step 3: Generate nodes for each submesh

3.1. A set of intermediate cross-sections are defined in accordance to the longitudinal space step (see Figure 1a),

3.2. Application of an affine transformation to ensure that X and Y coordinates of nodes follow constraint lines (see Figure 1b).

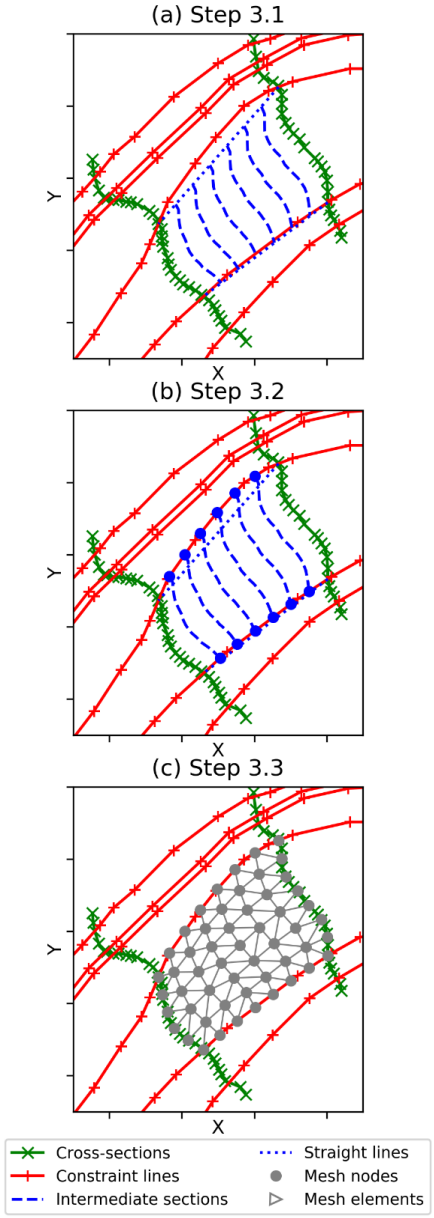

Figure 1 : Illustrations of the mesh generation steps for a single submodel

3.3. Lateral sampling of each intermediate cross-sections in accordance to mesh parameters (lateral space step or number of nodes), see Figure 1c.

Step 4: Triangulate over the whole domain

A constrained Delaunay triangulation is performed under the following constraints:

- coordinates of vertices/nodes are imposed (defined at step 3.1),

- segments along constraint lines and cross-sections are specified as hard lines.

The generated mesh operation leads to a "planar straight-line graph".

C. Overview of main features to control the mesh

C.1) Choice of the interpolation method of the coordinates of the constraint lines

If no constraint lines file is specified, TatooineMesher considers the banks of the river (right and left) as two constraint lines (Figure 2a). On the other hand, user defined constraint lines, which can intersect only a subset of cross-sections, are used to delimit the meshed domain (Figure 2b).

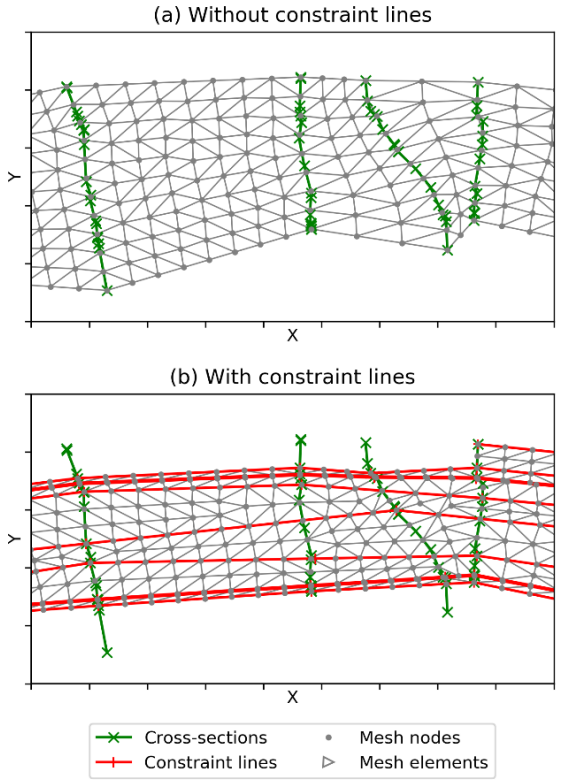

Figure 2 : Meshes generated without constraint lines (a) or with multiple constraint lines (b)

In addition to the vertices defining the constraint lines, the interpolation method between them can be computed with 2 different methods:

- linear interpolation (Figure 3a),
- Cubic Hermite Spline (CHS) interpolation (Figure 3b).

The CHS interpolation method is interesting to smooth constraint lines [3]. Moreover, if the modeller does not provide any constraint line, the river will certainly be less distorted with this interpolation (river width is more conservative).

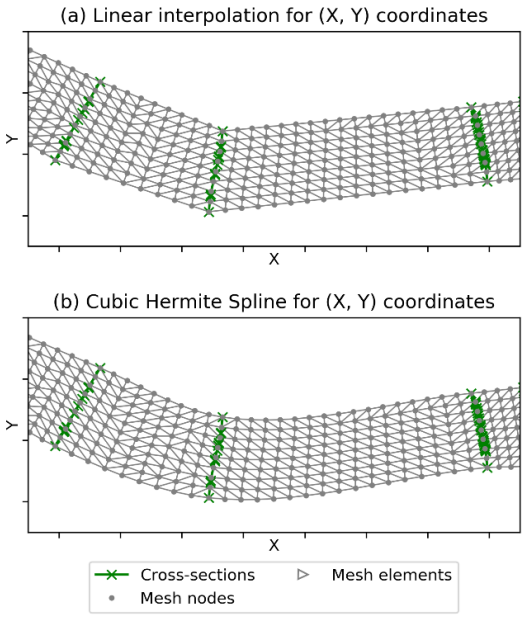

Figure 3 : Linear interpolation for X and Y coordinates of constraint lines (a) or cubic spline interpolation (b)

C.3) Option for planar projection of cross-sections

TatooineMesher has an option to make the squiggly cross-sections straight through a planar projection of cross-sections points on the line connecting left and right banks (see Figure 4).

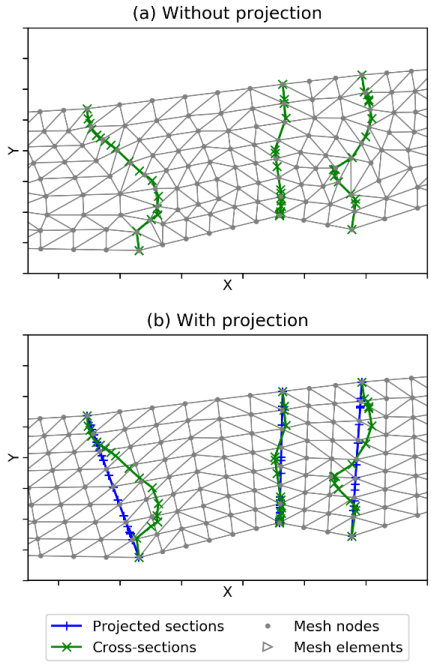

Figure 4 : Meshes generated without (a) or with (b) projection

This option makes the elements mesh adjacent to the cross-sections more organized. The initial bathymetry points are shifted but this can avoid elements to be oriented in the transversal direction.

C.4) Lateral discretization

The lateral discretization of the mesh can be specified either through a constant space step (Figure 5a) or through a number of nodes along these cross-sections (Figure 5b). The first option is highly recommended if the river width varies, and this will generate an unstructured mesh in the lateral direction.

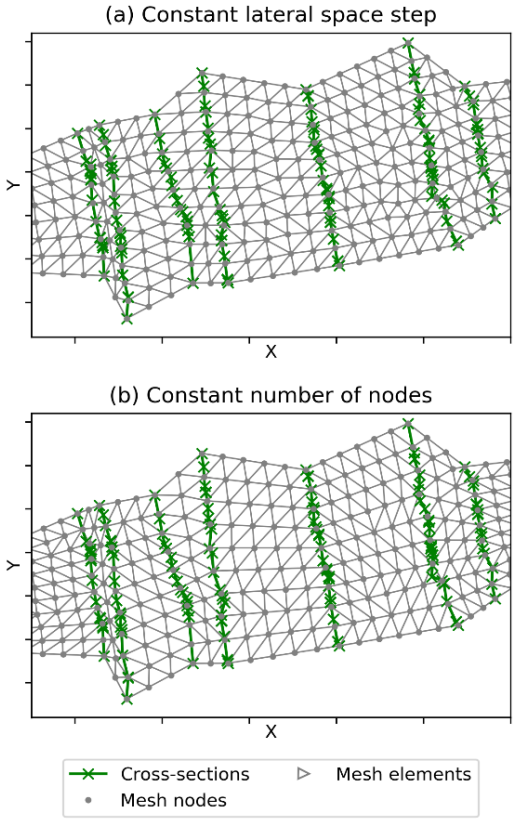

Figure 5 : Lateral discretization of the mesh with a constant lateral space step (a) or a constant number of nodes (b)

C.5) Longitudinal discretization

The longitudinal discretization of the mesh is controlled by a space step which can be different from the lateral one (elements can be elongated if necessary).

An additional option is provided to the modeller to specify how intermediate points are defined. Two approaches are available:

- number of intermediate cross-sections is determined for each bed (or submesh) individually, leading to an unstructured mesh (Figure 6a),
- or is unique per profile (Figure 6b).

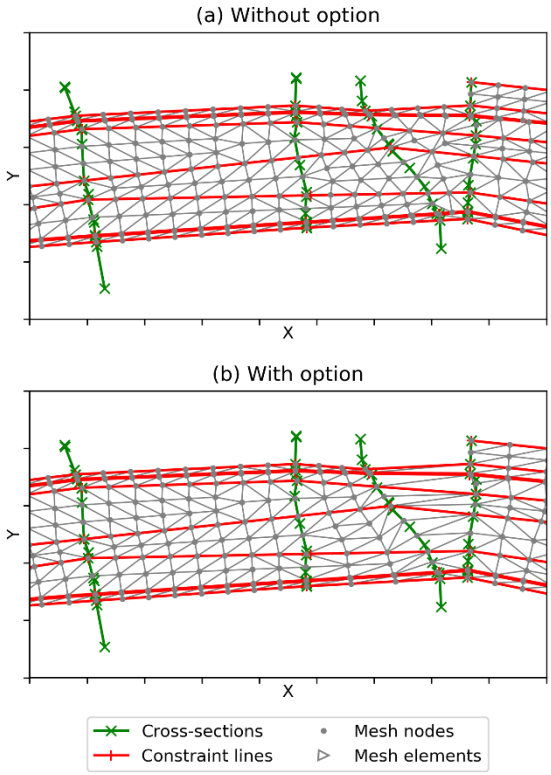

Figure 6 :Number of intermediate lines between two consecutive cross-sections different for each submesh (a) or identically laterally (b)

III. INTERPOLATION

After having determined the nodes localization (during the mesh generation, see part II), the bathymetry elevation (or any variable defined along each cross-section) can be interpolated at their positions. This section presents the importance of the choice of spatial interpolation methods in interpolating river bathymetry.

A. Isotropic interpolation methods not suitable

Some standard isotropic interpolation methods were tested to analyse their suitability to reconstruct surface from discrete cross-sections. Three interpolation methods already implemented in many GIS software are compared in Figure 7. The input data set consisting of cross-sections is represented in black and the reference (measured) surface corresponds to the left subplot.

This analysis shows that standard isotropic methods have major limitations/issues:

- interpolated bottom can be discontinuous,
- complex geometry might lead to inconsistencies, especially in curvatures or in case of non-straight cross-sections.

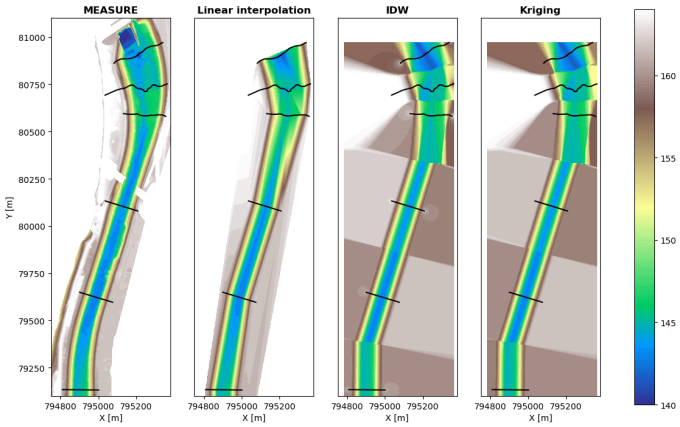

Figure 7 : Bathymetry measured (left) compared to interpolated bathymetry from elevation along cross-sections (3 interpolation methods: Linear, IDW (Inverse distance weighting) and Kriging)

B. *Flow-oriented coordinate system*

The mesh built with TatooineMesher has a set of nodes regularly distributed in every submeshes. Each node can be identified with its X and Y coordinates in the Geographic coordinate system (e.g. Figure 1) or with the couple of coordinates (x_t, x_l) per submesh. These latter coordinates (ranging from 0 to 1) correspond respectively to the lateral and longitudinal dimensionless curvilinear distances (see Figure 8). The corresponding coordinate system is called "flow-oriented" and enables to take into account the anisotropy of the bathymetry (the 2 axes are orthogonal).

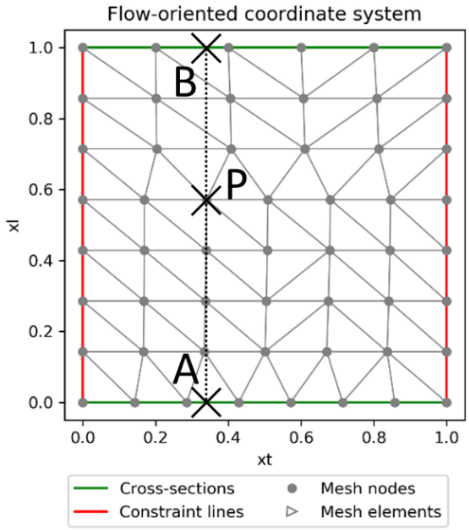

Figure 8 : Single submesh displayed in the flow-oriented coordinate system

C. *Consecutive 1D interpolators (lateral then longitudinal)*

P being a node between 2 consecutive cross-sections (with indices i and $i+1$). The bathymetry of P (denoted Z_P) can be interpolated from its position (x_t, x_l) in the flow-oriented coordinate system. The bathymetry can be determined (for each bed individually) with 2 consecutive 1D interpolations as presented hereafter.

1. Two lateral interpolations to have Z_A and Z_B: $Z_A = Z_i(x_t)$ and $Z_B = Z_{i+1}(x_t)$
2. Longitudinal interpolation between these 2 values: $Z_P = Z(x_t, x_l) = (1 - x_l)\, Z_A + x_l\, Z_B$

In the previous equations, $Z_i(x_t)$ corresponds to the bottom elevation at x_t position along section i. The points A and B are respectively located along sections i and $i+1$ (thus their coordinates are $(x_t, 0)$ and $(x_t, 1)$ in the flow-oriented coordinate system).

The two distinct interpolations (lateral and longitudinal) can possibly be based on different methods. In the context of surface reconstruction from cross-sectional data, the linear interpolation is widely used (see [3] and [4]). In this section, only the choice of the lateral interpolation is analysed. A more complete comparison is presented in part IV.B.

The methods tested and implemented in TatooineMesher are:

1. Linear interpolation (polynomial of degree 1);
2. Spline interpolation (polynomial of degree 3) with different assumptions at boundaries:
    a. Cubic spline,
    b. Akima,
    c. Piecewise Cubic Hermite Interpolating Polynomial (PCHIP).

A comparison of the interpolation methods on a simple cross-section is shown in the figure below. All the different methods ensure that the interpolated values pass through initial points.

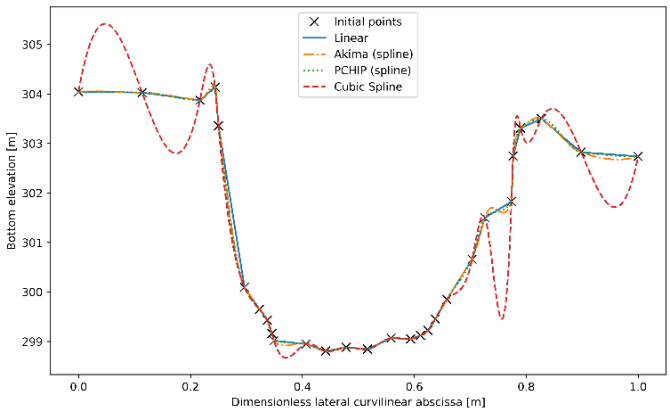

Figure 9 : Comparison of lateral interpolation methods for a cross-section

D. *Global 2D interpolators*

A more complex approach is to perform a 2D interpolation for the whole domain in a global flow-oriented coordinates, which means that the y-axis corresponds to non-dimensionless values (x_l being interpreted a sequence ranging from 1 to the number of cross-sections). This method is not compatible if the number of constraint lines changes over the longitudinal direction (x_t would not be consistent). The bilinear and bicubic methods are implemented in TatooineMesher.

IV. Results and Discussions on Test Cases

A. Mesh generation on "L'Étournel" site

A limited domain on the Upper Rhône River (upstream Génissiat dam) called L'Étournel is chosen to compare meshes generated with TatooineMesher with different space discretization options. This simple data set, presented in Figure 10, includes 25 cross-sections intersected by at most 5 constraint lines.

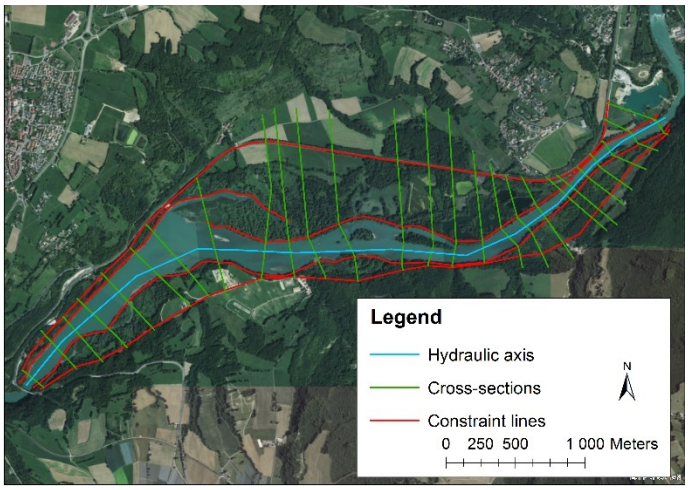

Figure 10 : Geometrical data used to mesh "L'Étournel" site

The different space discretization choices compared are presented in the Table 1.

Mesh ID	*Lateral discretization*	*Option to have laterally the same number of nodes per submesh (see II.C.5)*	*Longitudinal discretization*
1	ΔXt=20m	Without option	ΔXl=20m
2	ΔXt=20m	Without option	ΔXl=30m
3	ΔXt=20m	With option	ΔXl=20m
4	30 nodes	-	ΔXl=20m

Table 1 : Mesh generation parameters tested on L'Étournel site

The generated mesh characteristics are compared in Figure 11 with multiple probability density functions:

1. elements area
2. edge ratio per element
3. minimal angle per element.

The first barplot shows that the generated meshes have different number of nodes. The elongation of elements along the longitudinal direction (mesh 1 vs 2 which have a ratio of 1.5) is very efficient to minimize the number of elements (and indirectly the time step if CFL is fixed), and of course the minimal angles are getting slightly worse.

The quality of mesh n°4 is the worst, the element shape being highly heterogenous because the river reach width varies, while the number of nodes remain constant transversally. Consequently, the number of elements with an angle less than 30° is high.

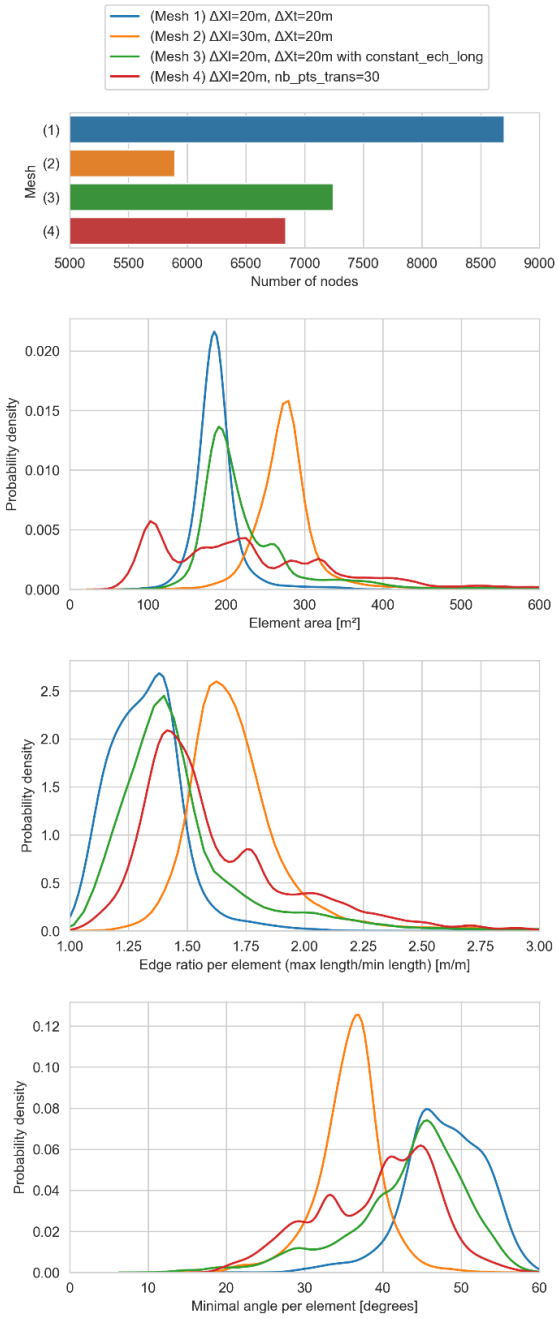

Figure 11 : Statistics on generated meshes

B. Interpolation methods benchmark on a part of the Rhône river

A validation test case was used to compare a reference surface to the interpolated surfaces (with methods presented in parts III.C and III.D). The reference surface was measured by LIDAR and multi-beams to get a very fine dataset of 3D points.

The test case is in the region of Vaugris, which is located along the Rhône river. The river reach considered is 10km long and the river width is about 200m. A set of 27 cross-sections are positioned regularly at each 400m (see Figure

12). The cross-sections are straight and composed of 20 points. Two lateral constraint lines are used to define the river bed to be meshed.

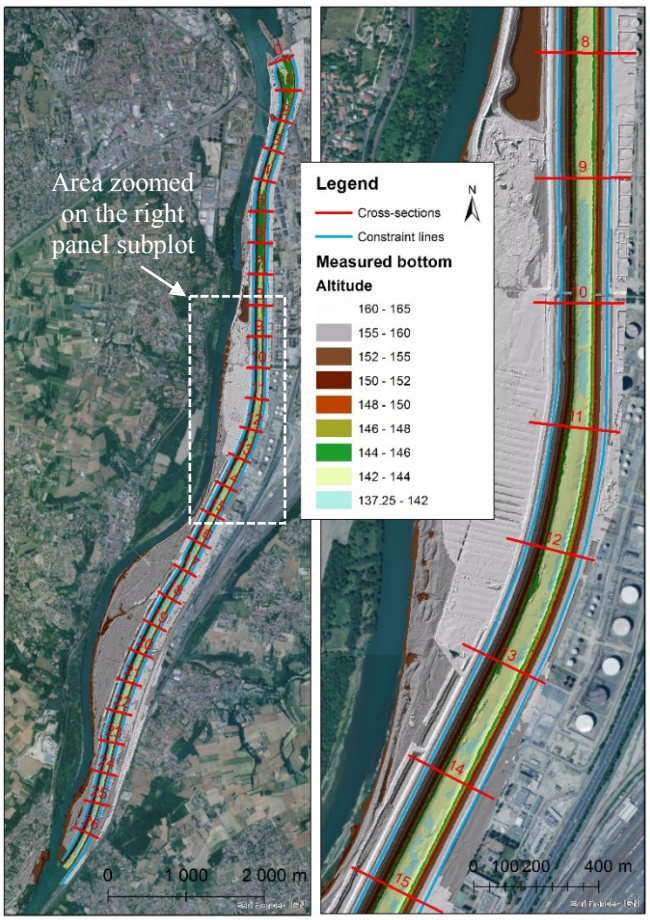

Figure 12 : Map of the "Vaugris" validation test case with the reference surface and input data (cross-sections and constraint lines)

This validation test case was used to compare the different interpolation methods with all other parameters and input data being equal. Six meshes were generated with strictly the same characteristics (space step are ΔXt=6.5m and ΔXl=10m, 50k elements) but with values at nodes depending on the interpolation method used. Two 2D global interpolation methods and four 1D lateral interpolation methods (in combination with a linear longitudinal interpolation) are compared.

As expected, differences between computed surfaces and the reference surface are spatially varying and are larger far from input data than in the vicinity of cross-sections data points. The performance of the different interpolation methods is assessed with the calculation of some statistical criteria:

- MSD = Mean Standard Deviation (closest to 0 is better),
- MAD = Mean Absolute Deviation (lowest is better),
- RMSD = Root-Mean Square Deviation (lowest is better).

These criteria are computed with PyTelTools [5] on the domain meshed by removing some specific local zones (vicinity of bridges not represented by cross-sections) to compute reliable differences. Results are plotted in the Figure 13 for the 6 interpolation methods. The generated surfaces have almost a null bias (less than 3cm) and the absolute differences are on average around 48cm. The lateral 1D interpolation methods play a role in the interpolation because cross-section resolution is not very fine (with "only" 20 points) and the Akima spline interpolation performs well. Therefore, this latter lateral interpolation is recommended if cross-sections are not finely discretized, but if not, linear interpolation remains a very efficient and robust method.

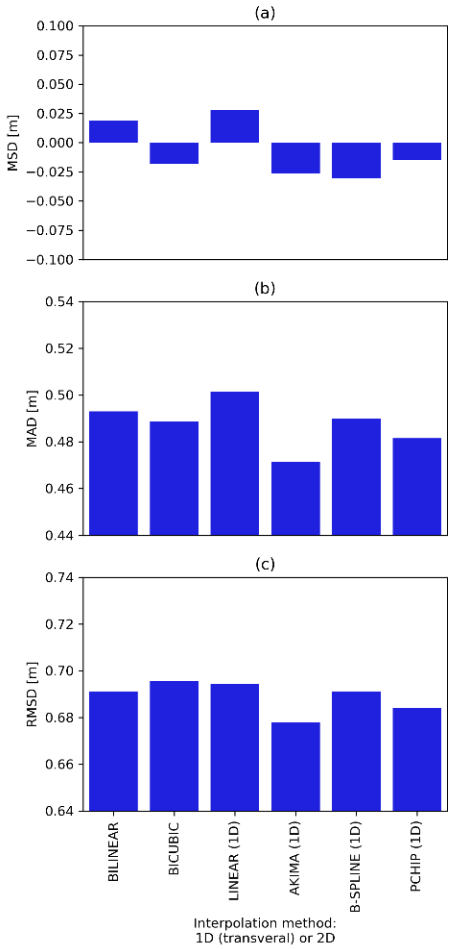

Figure 13 : Criteria computed to compare interpolation methods (a=MSD, b=MAD, c=RMSD)

V. MAIN APPLICATIONS OF TATOOINEMESHER

The three main applications of TatooineMesher (see part I.A) correspond to different scripts which are described in this section.

A. Densify profiles (densify_profiles.py)

This tool is used to refine 1D cross-sections for a further 1D calculation, no mesh being generated.

B. Interpolator and mesher (interpolator_and_mesher.py)

This script is dedicated to interpolation and/or channel meshing for 2D models.

C. *Visualize 1D model results (mesh_crue10_run.py and mesh_mascaret_run.py)*

This last tool is based on the same principle of *interpolator_and_mesher.py* but input data correspond to a 1D model (composed of a set of branches with its associated cross-sections) and output file contains 2D temporal generated surfaces for every variable. The following data can be meshed and visualised:

- model geometry: bathymetry and friction coefficient spatialization,
- 1D hydraulic results: a set of steady states or a transient simulation. Variables might be 1D (do not change laterally, see Figure 14) or 2D as seen in Figure 15.

Figure 14 : Visualization of a 1D variable (Froude number) at a given time

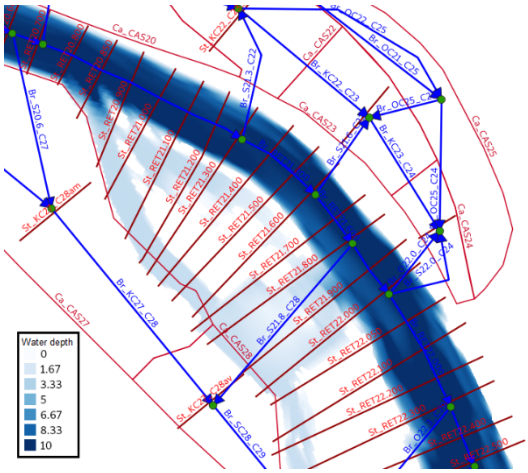

Figure 15 : Visualization of a 2D variable (water depth) at a given time

VI. How to use the scripts

A. *Installation and requirements*

TatooineMesher is released under the GPLv3 license. The source code was developed in Python version 3 and relies on several Python scientific libraries:

- *NumPy*: numerical library for large multidimensional arrays, and high-level mathematical routines [6],
- *SciPy*: used for several interpolation methods [7],
- *triangle*: simple mesh generator proving a triangulation [8].

In addition to these standard packages, external packages (available on https://github.com/CNR-Engineering) are used to parse files:

- *PyTelTools*: to write mesh files from a given triangulation [5],
- *Crue10_tools*: to read 1D geometry and results files from Crue and Mascaret (based on new *postel* module).

B. *Command line interface scripts*

The developed scripts can be run through a command line with the relevant arguments. The help message can be displayed with *-h* argument. For more details, see online at https://github.com/CNR-Engineering/TatooineMesher/wiki.

C. *File formats used in TatooineMesher*

Input data files can be provided in different formats and should fulfil some conditions. The Table 2 summarized them for the 3 types of input file.

File containing	Supported formats	Expected data
Cross-sections	shp, i3s	3D lines in arbitrary order but described longitudinal in the same direction (right to left bank, or the opposite)
Hydraulic axis	shp, i2s	A single 2D line oriented from upstream to downstream
Constraint lines (optional)	shp, i2s	At least 2 lines, not intersecting them and oriented in the same direction as the hydraulic axis
1D model	Crue10, Mascaret	Geometry and hydraulic results at sections
Output mesh	slf, t3s, LandXML	Mesh which can contain multiple temporal frame and variables

Table 2 : Characteristics of data input and output files of TatooineMesher

VII. Conclusion and Perspectives

TatooineMesher is a set of scripts released by CNR to provide efficient tools for 1D and 2D river modelling engineering with MASCARET and TELEMAC-2D. A channel mesher and an anisotropic interpolator were developed to reconstruct a 2D continuous surface from data at discrete cross-sections. A more complex mesher (e.g. GMSH) could be combined with TatooineMesher to mesh the floodplain or to provide an alternative mesher for river bed (to finely mesh obstacles and to adapt node density spatially). This will probably be investigated in the future.

References

[1] C. Geuzaine and J.-F. Remacle, “Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities”. International Journal for Numerical Methods in Engineering 79(11), pp. 1309-1331, 2009

[2] U. H. Merkel. “Generation and Quality of Unstructured Meshes”, Telemac User Conference 2013, Karlsruhe (Germany)

[3] D. Caviedes-Voullième et al., “Reconstruction of 2D river beds by appropriate interpolation of 1D cross-sectional information for flood simulation”, Environmental Modelling & Software 61, 2014

[4] B. Schäppi et al., “Integrating river cross section measurements with digital terrain models for improved flow modelling applications”, Computers & Geosciences, 2009

[5] L. Duron, Y. Wang, “PyTelTools: Python scripts and GUI to automate Telemac post-processing tasks”, 2017, Graz (Austria)

[6] T. E, Oliphant. A guide to NumPy, USA: Trelgol Publishing, 2006

[7] E. Jones, T. Oliphant, P. Peterson, et al., SciPy: Open Source Scientific Tools for Python, 2001, http://www.scipy.org/ [Online; accessed 2019-08-09].

[8] J. R. Shewchuk, “Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator”, in Applied Computational Geometry: Towards Geometric Engineering, May 1996.