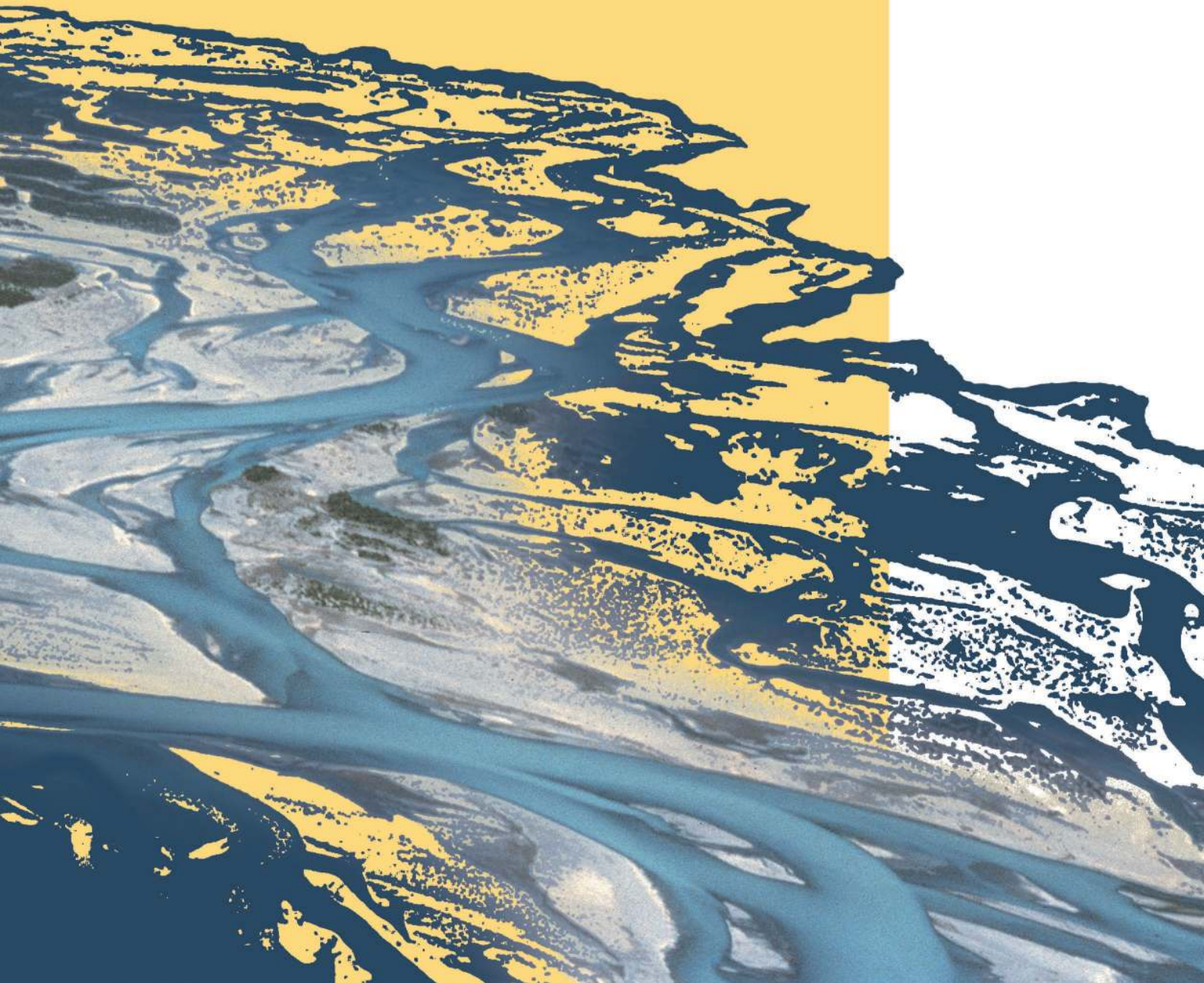
SYSTEM MANUALS

of BASEMENT



U

USER MANUAL

of BASEMENT

Contents

Preamble

VERSION 2.8.2

January, 2022

Credits

Project Team

Software Development, Documentation and Test (alphabetical)

M. Bürgler, MSc. ETH Environmental Eng.
F. Caponi, MSc. Environmental Eng.
Dr. D. Conde, MSc. Civil Eng.
E. Gerke, MSc. ETH Civil Eng.
S. Kammerer, MSc. ETH Environmental Eng.
Dr.techn. M. Weberndorfer, MSc.

Scientific Board

Prof. Dr. R. Boes, Director VAW, Member of Project Board
Dr. A. Siviglia, MSc, Scientific Adivisor
Dr. D. Vanzo, MSc. Environmental Eng., Scientific Adivisor
Dr. D. Vetsch, Dipl. Ing. ETH, Project Director

Former Project Members

See https://www.basement.ethz.ch/people

Cover Page Art Design

W. Thürig

Commissioned and co-financed by

Swiss Federal Office for the Environment (FOEN)

Contact

website: http://www.basement.ethz.ch
user forum: http://people.ee.ethz.ch/~basement/forum

Citation Advice

For System Manuals:

Vetsch D., Siviglia A., Bürgler M., Caponi F., Ehrbar D., Facchini M., Faeh R., Farshi D., Gerber M., Gerke E., Kammerer S., Koch A., Mueller R., Peter S., Rousselot P., Vanzo D., Veprek R., Volz C., Vonwiller L., Weberndorfer M. 2022. System Manuals of BASEMENT, Version 2.8.2 Laboratory of Hydraulics, Glaciology and Hydrology (VAW). ETH Zurich. Available from http://www.basement.ethz.ch. [date of access].

For Website:

BASEMENT – Basic Simulation Environment for Computation of Environmental Flow and Natural Hazard Simulation, 2022. http://www.basement.ethz.ch

For Software:

BASEMENT – Basic Simulation Environment for Computation of Environmental Flow and Natural Hazard Simulation. Version 2.8.2 © ETH Zurich, VAW, 2006-2022.

License

BASEMENT SOFTWARE LICENSE

between

ETH

Rämistrasse 101

8092 Zürich

Represented by Prof. Dr. Robert Boes

VAW

(Licensor)

and

Licensee

1. Definition of the Software

The Software system BASEMENT is composed of the executable (binary) file BASEMENT and its documentation files (System Manuals), together herein after referred to as "Software". Not included is the source code.

Its purpose is the simulation of water flow, sediment and pollutant transport and according interaction in consideration of movable boundaries and morphological changes.

2. License of ETH

ETH hereby grants a single, non-exclusive, world-wide, royalty-free license to use Software to the licensee subject to all the terms and conditions of this Agreement.

3. The scope of the license

a. Use

The licensee may use the Software:

- according to the intended purpose of the Software as defined in provision 1
- by the licensee and his employees
- for commercial and non-commercial purposes

The generation of essential temporary backups is allowed.

b. Reproduction / Modification

Neither reproduction (other than plain backup copies) nor modification is permitted with the following exceptions:

Decoding according to article 21 URG [Bundesgesetz über das Urheberrecht, SR 231.1)

If the licensee intends to access the program with other interoperative programs according to article 21 URG, he is to contact licensor explaining his requirement.
If the licensor neither provides according support for the interoperative programs nor makes

the necessary source code available within 30 days, licensee is entitled, after reminding the licensor once, to obtain the information for the above mentioned intentions by source code generation through decompilation.

c. Adaptation

On his own risk, the licensee has the right to parameterize the Software or to access the Software with interoperable programs within the aforementioned scope of the licence.

d. Distribution of Software to sub licensees

Licensee may transfer this Software in its original form to sub licensees. Sub licensees have to agree to all terms and conditions of this Agreement. It is prohibited to impose any further restrictions on the sub licensees' exercise of the rights granted herein.

No fees may be charged for use, reproduction, modification or distribution of this Software, neither in unmodified nor incorporated forms, with the exception of a fee for the physical act of transferring a copy or for an additional warranty protection.

4. Obligations of licensee

a. Copyright Notice

Software as well as interactively generated output must conspicuously and appropriately quote the following copyright notices:

Copyright by ETH Zurich / Laboratory of Hydraulics, Glaciology and Hydrology (VAW), 2006-2018

5. Intellectual property and other rights

The licensee obtains all rights granted in this Agreement and retains all rights to results from the use of the Software.

Ownership, intellectual property rights and all other rights in and to the Software shall remain with ETH (licensor).

6. Installation, maintenance, support, upgrades or new releases

a. Installation

The licensee may download the Software from the web page http://www.basement.ethz.ch or access it from the distributed CD.

b. Maintenance, support, upgrades or new releases

ETH doesn't have any obligation of maintenance, support, upgrades or new releases, and disclaims all costs associated with service, repair or correction.

7. Warranty

ETH does not make any warranty concerning the:

- warranty of merchantability, satisfactory quality and fitness for a particular purpose
- warranty of accuracy of results, of the quality and performance of the Software;
- warranty of noninfringement of intellectual property rights of third parties.

8. Liability

ETH disclaims all liabilities. ETH shall not have any liability for any direct or indirect damage except for the provisions of the applicable law (article 100 OR [Schweizerisches Obligationenrecht]).

9. Termination

This Agreement may be terminated by ETH at any time, in case of a fundamental breach of the provisions of this Agreement by the licensee.

10. No transfer of rights and duties

Rights and duties derived from this Agreement shall not be transferred to third parties without the written acceptance of the licensor. In particular, the Software cannot be sold, licensed or rented out to third parties by the licensee.

11. No implied grant of rights

The parties shall not infer from this Agreement any other rights, including licenses, than those that are explicitly stated herein.

12. Severability

If any provisions of this Agreement will become invalid or unenforceable, such invalidity or enforceability shall not affect the other provisions of Agreement. These shall remain in full force and effect, provided that the basic intent of the parties is preserved. The parties will in good faith negotiate substitute provisions to replace invalid or unenforceable provisions which reflect the original intentions of the parties as closely as possible and maintain the economic balance between the parties.

13. Applicable law

This Agreement as well as any and all matters arising out of it shall exclusively be governed by and interpreted in accordance with the laws of , excluding its principles of conflict of laws.

14. Jurisdiction

If any dispute, controversy or difference arises between the Parties in connection with this Agreement, the parties shall first attempt to settle it amicably.
Should settlement not be achieved, the Courts of Zurich-City shall have exclusive jurisdiction.
This provision shall only apply to licenses between ETH and foreign licensees

By using this software you indicate your acceptance.

(License version: 2018-05-31)

THIRD PARTY SOFTWARE

BASEMENT uses third party software. For instance, the BASEMENT executable directly links the following external libraries:

- CGNS
- HDF5
- Qt5 (non-cluster version only)
- Qwt (non-cluster version only)
- Shapelib
- TecIO
- VTK (non-cluster version only)

The libraries (and their dependencies) are included in the BASEMENT distribution if they are not provided by the operating system.

Please refer to `ThirdPartySoftwareLicenses.txt` in the distribution and/or the operating system documentation for the third party software licenses and copyright notices. The external libraries for Windows 10 have been built using `vcpkg` version 2020.07 (`HDF5` was compiled without `szip`).

1

Basic Simulation Environment

1.1 Introduction

The software system „BASEMENT“(basic-simulation-environment) shall provide a flexible and functional environment for numerical simulation of alpine rivers and sediment transport involved. The numerical models for the computation of one- and two dimensional flows with moving boundaries and appropriate models for bed load as well as suspended load are forming the core of the software system. Two of the main project tasks were the renewal and further development of the existing 1-D and 2-D models (”Floris“,”2dMB“). The one-dimensional model complies with the upper bound of the considered spatial scale (maximum idealization and slightest resolution of spatial processes) and is meant to provide appropriate boundary conditions for the 2-D and 3-D models.

The main focus of conception and development was the stability of the numerical models, the flexibility of the computational grid and the combination and efficiency of the method of calculation (problem dependent equations, coupling of models, parallelization).

The development process was orientated at the concepts of object orientation, to assure transparency, documentation and flexibility of the software system as far as possible. Future developments, applications related to practice and scientific projects shall build upon the environment of BASEMENT to ensure sustainability.

1.2 General Use

1.2.1 Problem Description

In connection with watercourses and river areas, increasingly complex problems have to be addressed. The estimation of floods, the more frequent occurrence of restoration projects or the study of naturally shaped watercourses implicate the examination of larger regions - also outside of the actual waterway - and a more manifold shape of the channels. The simple formulas for the calculation of flow behaviour used in the past showed in several

cases to be insufficient to obtain the desired information. The extent of the considered areas makes the application of hydraulic models in a laboratory - usually employed for difficult cases - impossible or too expensive. So, the numerical simulation of flow behaviour is in many cases the most obvious solution. However, existing programs have still some weak points. Some are limited in their capabilities (e.g. only steady flow, no sediment transport and one dimension only) or may lack in user support caused in incompleteness of documentation or training of users. Furthermore, inherent numerical problems request certain expertise to be overcome. In addition, the preparation of the input data and the processing of the results to a shape, which facilitates the interpretation, are often very laborious.

The aim of the software system BASEMENT, in terms of its free availability and its accompanying scholar programs, is to enable a broader range of people to skilfully process river modelling projects in a justifiable amount of time.

1.2.2 Product Delineation and Employment Domains

1.2.2.1 Product Delineation

BASEMENT is a river engineering tool, which supports the engineer in the solution of tasks in the domain of river area modelling. The program permits reliable computations based on state of the art numerical tools, constant onward development and successive realisation of case studies.

Unlike currently used programs for the simulation of a specific flow behaviour, BASEMENT intends the arrangement of many different problem types with one single tool to gain an integrated understanding for the initial position, the solution process and its results.

1.2.2.2 Employment Domains

The aim of BASEMENT is to permit the solution of as many problems as possible in the domain of river engineering, especially in cases for which the traditional dimensioning tools are insufficient and studies including physical hydraulic models are not possible or too expensive. Typical employment domains are:

- Several problems in relation with the sediment transport of water courses, for instance the future development of deltas and alluvial fans, the long term evolution of the bottom of channels, or the aggradations of storage spaces and the consequences of their scavenging;
- River engineering enterprises, which imply the modification of the channel geometry, as this can be the case for example for revitalisations or protection measures, where the consequences of the interventions have to be evaluated;
- Identification and quantification of dangers for the development of danger maps or of protection and emergency measures, considering the flow behaviour and sediment deposition both inside and outside of the main channel, as well as erosion danger, and consequences of debris flows and dam breaks.

1.2.3 Processed Data Types and Capabilities

1.2.3.1 Processed Data Types

The raw data can be divided into three groups:

- **topographic data**: particularly elevation models and cross sections
- **hydrologic data**: time series of flow discharge, water levels or concentration of suspended sediments, velocity profiles;
- **granulometric data**: grain size distributions from water-, sediment- or line samples.

1.2.3.2 Capabilities

BASEMENT has the following fundamental capabilities:

- Simulation of flow behaviour under steady and unsteady conditions in a channel as well as its transition;
- Simulation of sediment transport (both bed load and suspended load) under steady and unsteady conditions in a channel with arbitrary geometry;
- Simulation of erosion and deposition;
- Choose between different approaches (e.g. choice of problem matched solver-algorithms);

1.3 System Overview

At the current stage of development, the software system consists of the numerical subsystems and the different interfaces to the infrastructural software, such as pre- and post processors. The core of BASEMENT consists of the numerical solution algorithms comprised in the appropriate modules. Pre- and post-processing can be performed with independent products using a well defined common interface. The flexible software design enables a future adoption to a common database for input- and output data (Figure 1.1).

Numerical Subsystems

The core of the software system consists of the numerical subsystems, which actually are:

The one dimensional numerical tool named BASEchain enables the simulation of river reaches (based on cross sections) with respect to sediment transport. Arbitrary coupling with the 2-D tool is possible.

The two dimensional numerical tool named BASEplane enables the simulation

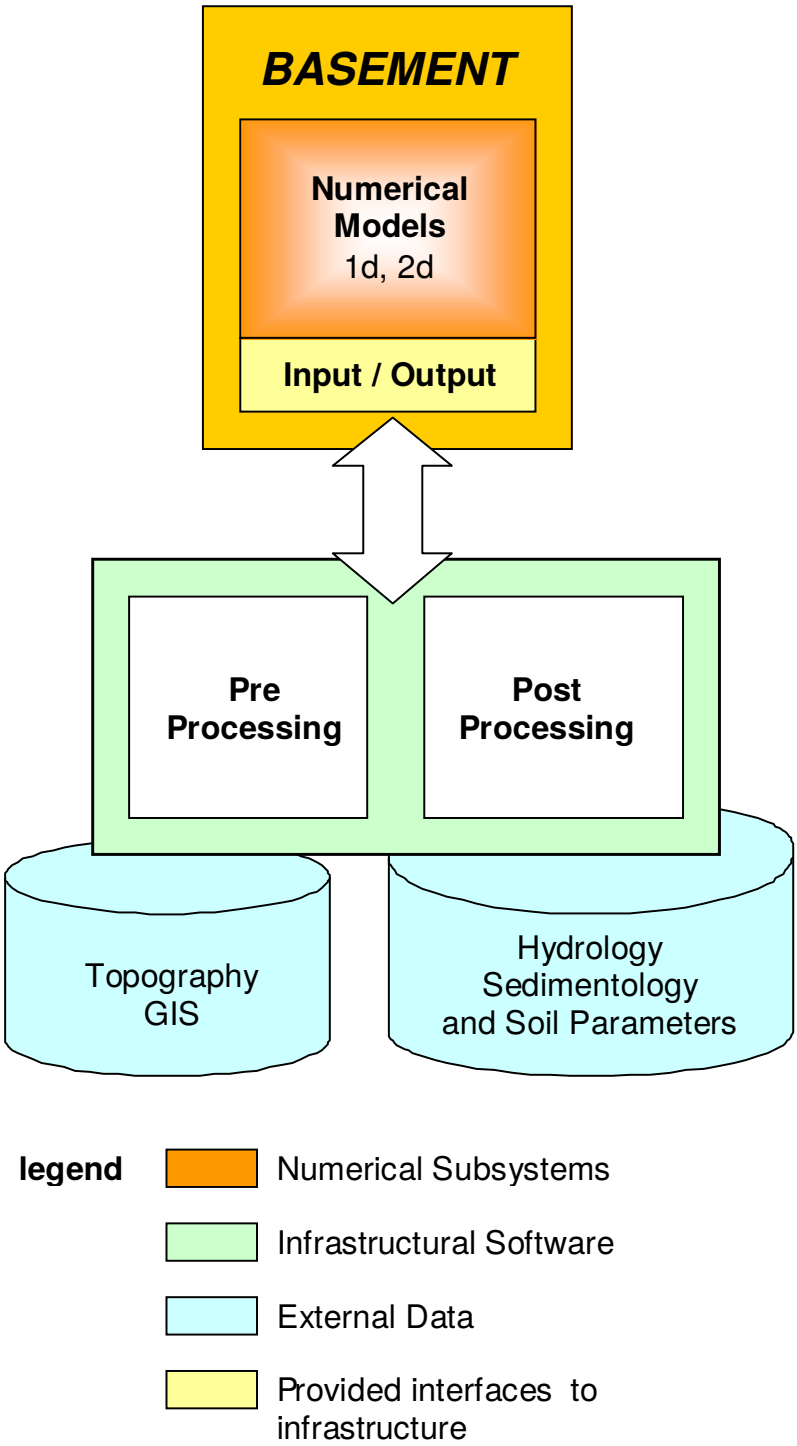


Figure 1.1 *System Overview*

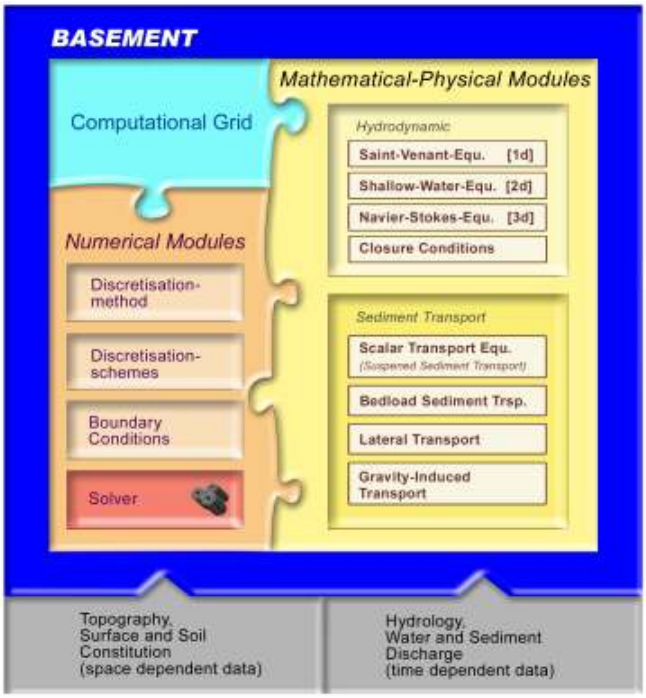


Figure 1.2 *Modules and their Components*

of river reaches as well as flood plains (bases on a digital terrain model) with respect to sediment transport. Arbitrary coupling with the 1-D tool is possible.

The three dimensional numerical tool named BASEsub is meant for the simulation of local subsurface flow fields (based on spatial geometry). The model is coupled with the BASEplane surface flow module.

For a list of implemented features of each model, please refer to the actual release notes.

1.4 Components

To reveal the "black box" of the numerical models, Figure 1.2 gives a graphical insight. The simulation tools of BASEMENT can be subdivided into three different parts:

- the mathematical-physical modules consisting of the governing flow equations
- the computational grid representing the discrete form of the topography
- and the numerical modules with their methods for solving the equations

In the next few chapters an overview of these modules is given. A more detailed description can be found in the reference manual.

1.4.1 Mathematical – Physical – Modules

The behaviour of the fluid hydraulics can be explained with physical models, namely the conservation of mass and momentum. Theoretically, it is possible to resolve the mathematical problem up to small scale phenomena like turbulence structures. In a natural problem however, it is mostly impossible to determine all boundary- and the exact initial conditions. Furthermore, the computational time needed to solve the full equation system is increasing very fast with higher spatial and temporal resolution. Therefore, dependent on the problem, simplified mathematical models are used.

In three dimensions, the flow and pressure distribution are completely described by the Navier-Stokes equations. These equations can only be solved numerically, as analytical solutions exist only for some strongly simplified problems. The 3-dimensional approach is only suitable for local problems, where turbulence phenomena and flow in all directions are essential for the results, e.g. the flow around bridge piers.

Assuming a static pressure distribution and neglecting the vertical flow components, the Navier-Stokes equations simplify to the 2-dimensional shallow water equations. This set of equations provides accurate results for the behaviour of water level and velocities in a plane. Turbulence effects cannot be resolved anymore but are accounted for by an artificial friction factor in the closure condition, which establishes a relation between flow velocity and shear stress. The shallow water equations are used for 2-dimensional flows like dam breaks, curved flow etc.

Reducing the spatial dimension once more results in the 1-D Saint-Venant equations. The main outputs of these equations are the water level and mean velocity in flow direction. This method is still in use for computing large river systems.

The computation of sediment transport is mathematically not as well developed as the hydrodynamic part. Theoretically, the movement of every single stone within the sediment could be computed by solving its equation of motion. However, this approach is yet numerically too expensive. Therefore, sediment transport and behaviour of the riverbed are computed using empirical formulas developed by river engineers. The computation of the sediment flux is physically not really correct, but proved to be accurate enough for a broad range of sediment transport problems. Usually, sediment transport occurs in the main flow direction. More sophisticated models consider also lateral phenomena within a curved flow.

Very small grain sizes are treated as suspended sediment load. Their behaviour can be computed by a physically scalar transport equation.

1.4.2 Computational Grid

1.4.2.1 The Meta Model

An important aspect of every computational task is the grid generation where the real world topography data is transformed into an internal computational grid on which the governing equations are solved. Independent of the discretization method, the construction of the computational grid has great impact on the accuracy of the results and on the computational time needed for the simulation or the numerical time step, respectively. Generally, a suitable mesh is dense at regions, where strong changes in the flow occur and

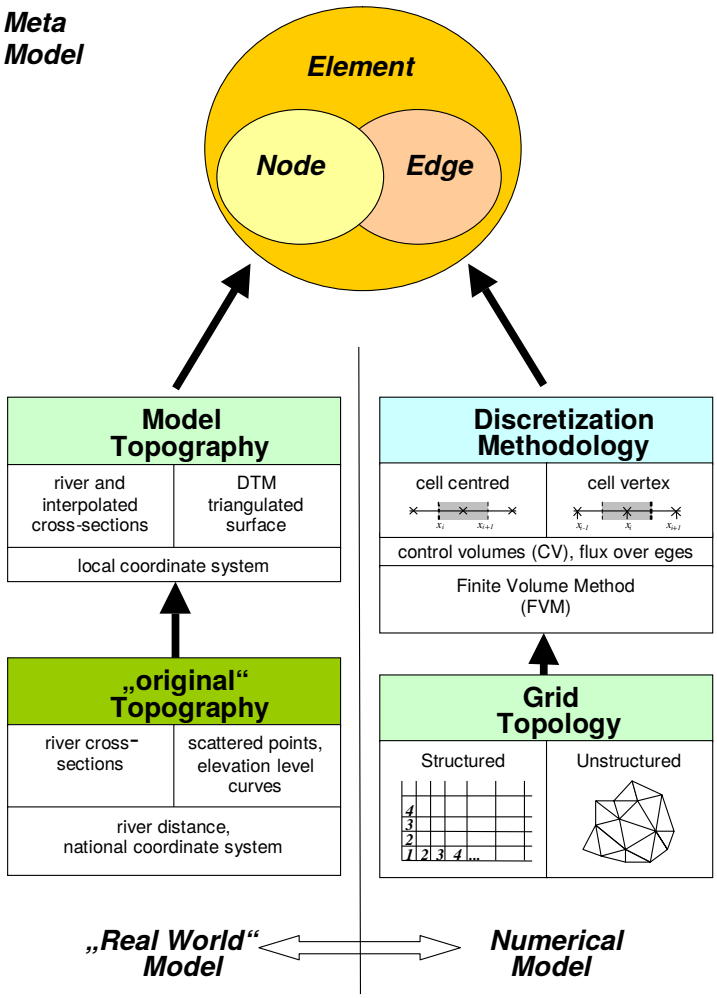


Figure 1.3 *The Meta Model: fusion of "real world" data with abstract numerical considerations*

coarse in regions of lower interest. Additionally, the grid cells should not underlie strong deformations.

Usually, the raw real world data comes in form of river cross sections or geographic terrain models e.g. from a GIS. This elevation information has to be mapped onto a suitable mesh.

There are two types of computational grids: structured and unstructured ones. Structured grids consist of quadrilaterals and can be mapped onto a Cartesian domain. They allow for simple data structures and efficient algorithms. The mesh generation is relatively simple and can even be done manually. However, structured meshes are somehow unhandy for the representation of arbitrary topography data. Unstructured grids are mostly composed of triangles and cannot be mapped onto Cartesian meshes. They usually need more complicated data structures but are highly flexible for automatic mesh generation in complex geometries. An unstructured grid is the most general case of a grid based discretization and is perfectly suitable for object oriented modelling. BASEMENT is built on unstructured grids.

The computational grid consists in general of cells (control volumes). In the software model, the mesh is based on three different objects:

- [node] the nodes – mass free points in relation to a coordinate system;
- [edge] the edges, which are defined by two nodes and define the place of information flux between two elements in Finite Volume Methods;
- [element] the elements, which are defined by several nodes and define the place of the physical variables, e.g. cell centered methods.

This data structure allows for similar treatment of 1-D and 2-D methods and schemes.

As the difficulty of mesh generation occurs in many different computational tasks, a broad range of different triangulation techniques or mesh refinement methods can be found in the literature. Some of them are specially designed for a certain discretization scheme but can also be used elsewhere. As there is nothing such as an ideal or perfect mesh, the user is recommended to produce different grids and compare their behaviour to find the best solution. There are commercial tools available which can be used for the grid generation, e.g. SMS. However, the BASEMENT standard for grid representation (see Reference manual) also allows for self created meshes.

1.4.2.2 BASEchain : one dimensional model

In one dimension, an element consists of two nodes with known cross-section. With a cell-centred discretization, all variables – velocity, flow depth and cross-section geometry - are defined at the location of the nodes. The midpoint of the connecting line between two nodes defines the common edge of the two elements. The more nodes are known, the better the representation of the real world data, especially at regions with strongly curved watercourse.

1.4.2.3 BASEplane : two dimensional model

In two dimensions, an element consists of three nodes with a known ground elevation. Usually, this real world height information is not given exactly at the desired node

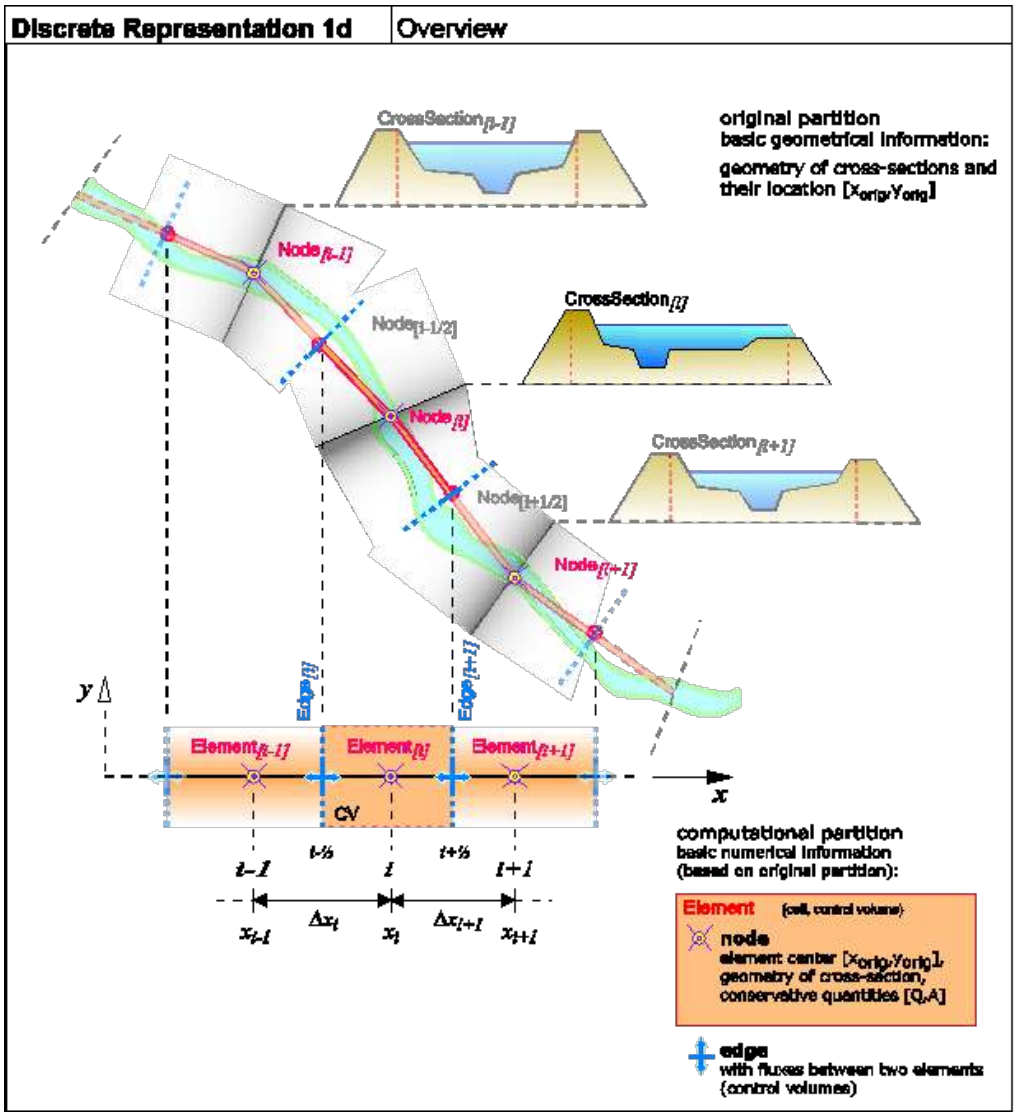


Figure 1.4 *Discrete Representation of the Topography within BASEchain*

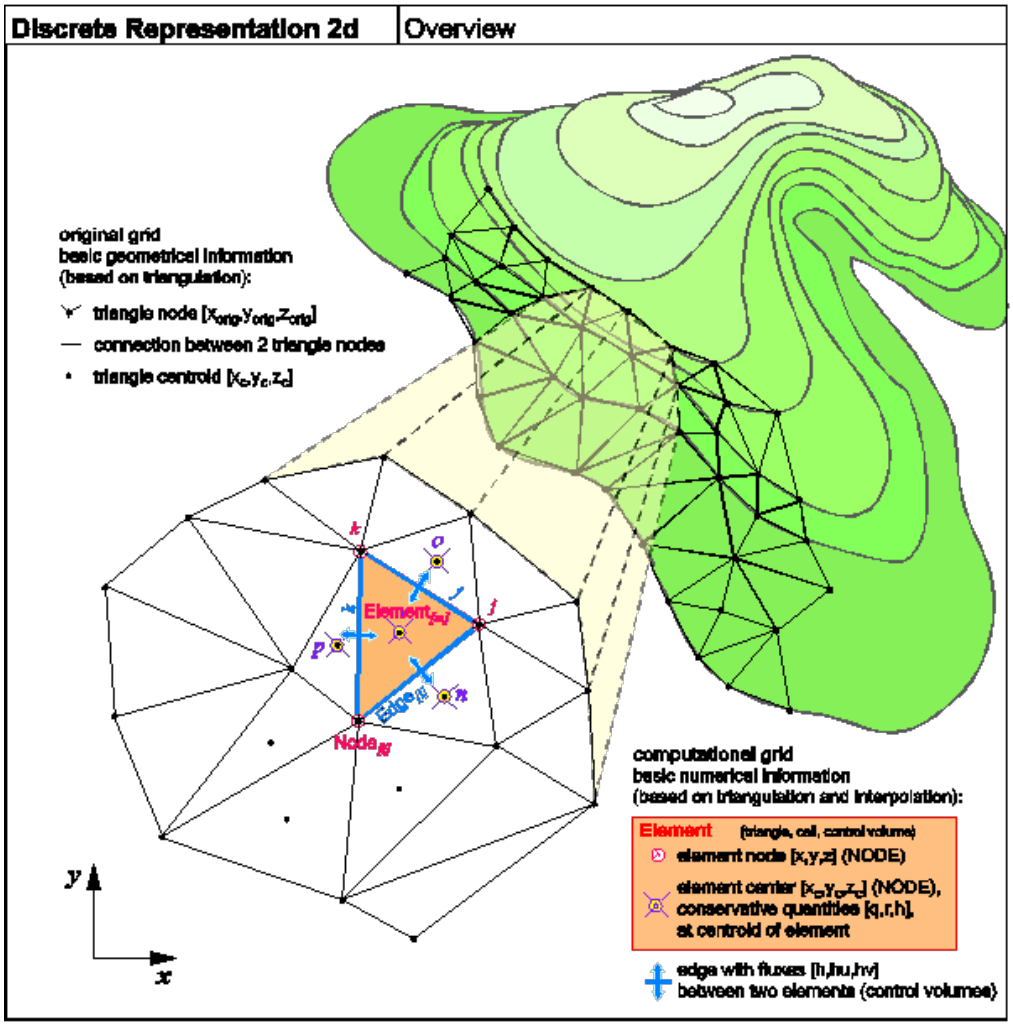


Figure 1.5 *Discrete Representation of the Topography within BASEplane*

coordinates and therefore has to be interpolated. The primary variables are defined somewhere inside the element, e.g. the balance point. The fluxes between two elements are defined at their corresponding edges.

1.5 Simulation Procedure

The procedure to simulate a concrete problem setup is not unique. BASEMENT is coded using an object-oriented design which allows for flexibility and interchange ability concerning different application problems. The possible combinations are manifold. On the one hand, the governing equations may change dependent on simplifications or extensions of certain terms, use of sediment transport or pure hydraulics, etc. On the other hand, there are miscellaneous numerical methods, e.g. for time integration (implicit, semi-implicit, explicit) or computation of spatial fluxes. Therefore, the main variables of interest differ from one problem to the other. It is of great importance, to plan carefully each simulation approach to a certain problem. The most difficult and time-consuming part is not the simulation itself but the acquisition of all needed data (topography, boundary- and initial conditions) and a proper setup of this data. This section describes the main activities performed to execute a simulation with BASEMENT in a very general case. In most problems, only a part of them are being used.

1.5.1 Flow of main activities

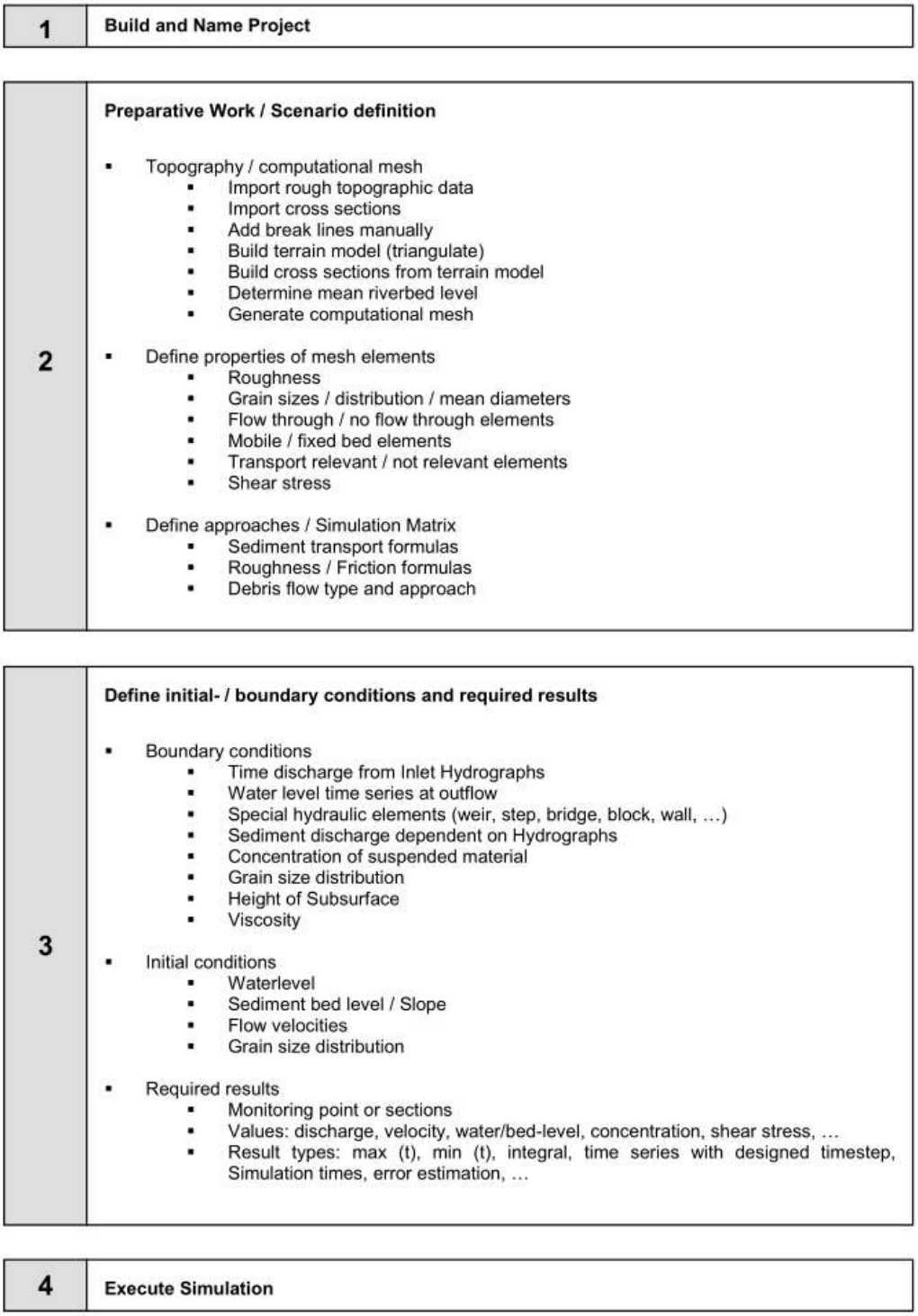


Figure 1.6 *Flow of main activities, part I*

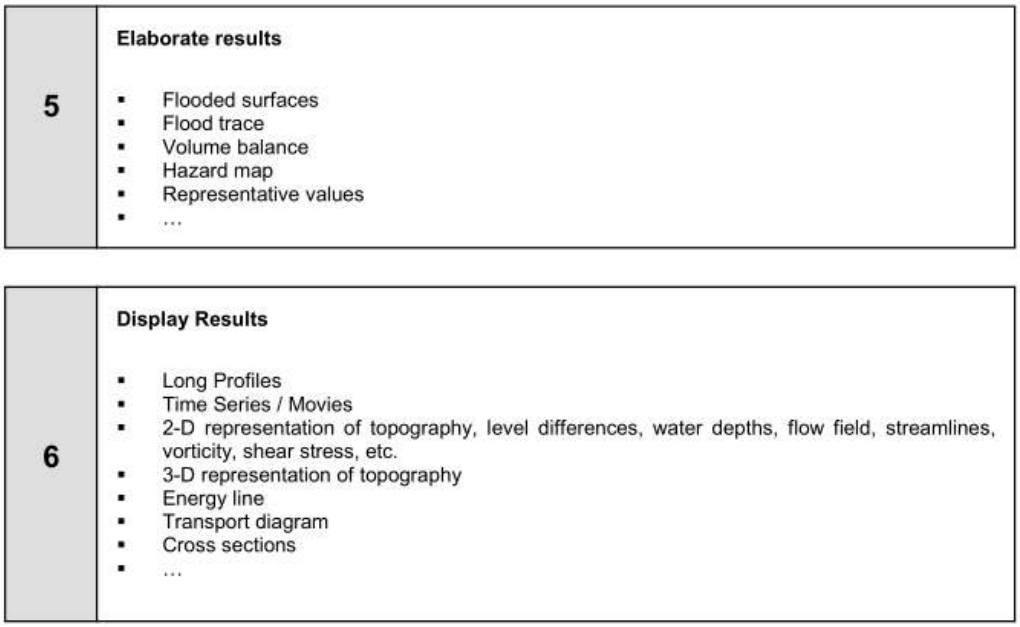


Figure 1.7 *Flow of main activities, part II*

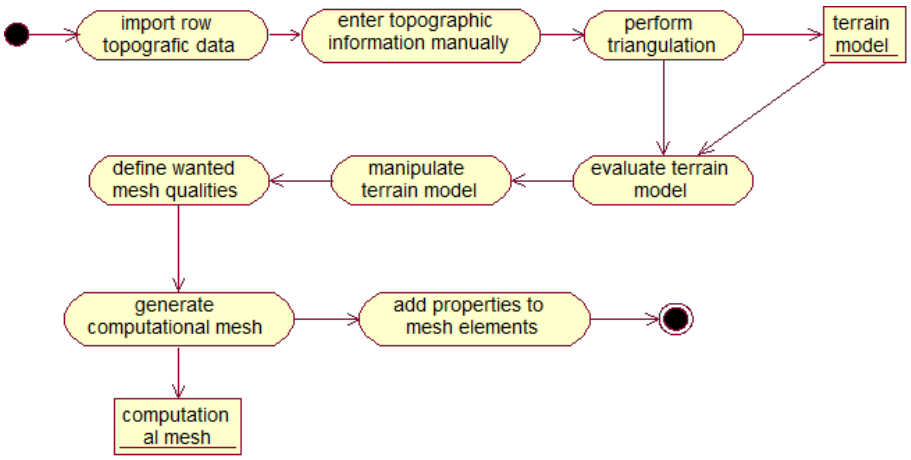


Figure 1.8 *Activity diagram generate computational mesh*

1.5.2 Scenario Examples

Depending on the chosen scenario and on the available boundary conditions or i.e. topography data, the approach for a successful simulation differs from case to case. As there are different ways to reach a certain target, the following activity diagrams just present possibilities but not a strict guideline.

An important part is always the grid generation. Usually, the raw topography data needs a lot of treatments (manual correction, interpolation, adjustment of single elements, etc.) until a suitable computational mesh can be generated. Although programs for grid generation like SMS provide some powerful tools to manipulate mesh transformations, the user still has to retain an overview over the required steps leading to the final computational mesh.

The following activity diagrams show a possible procedure for different project scenarios.

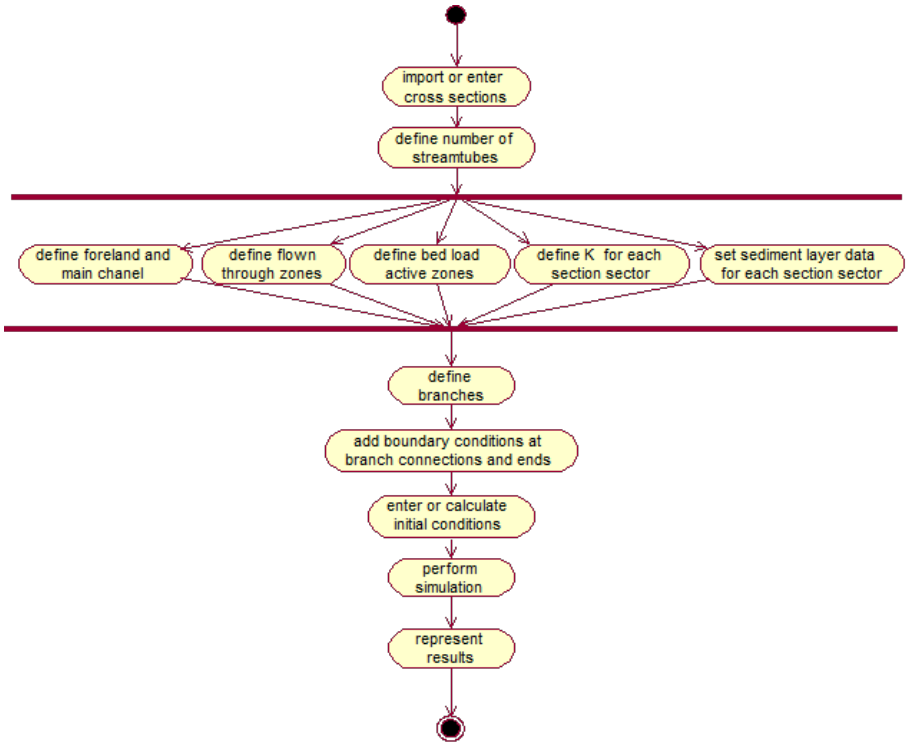


Figure 1.9 *Activity diagram Sediment balance in a river*

1.5.2.1 Sediment balance in a river 1-D

1.5.2.2 Flood 1-D + 2-D

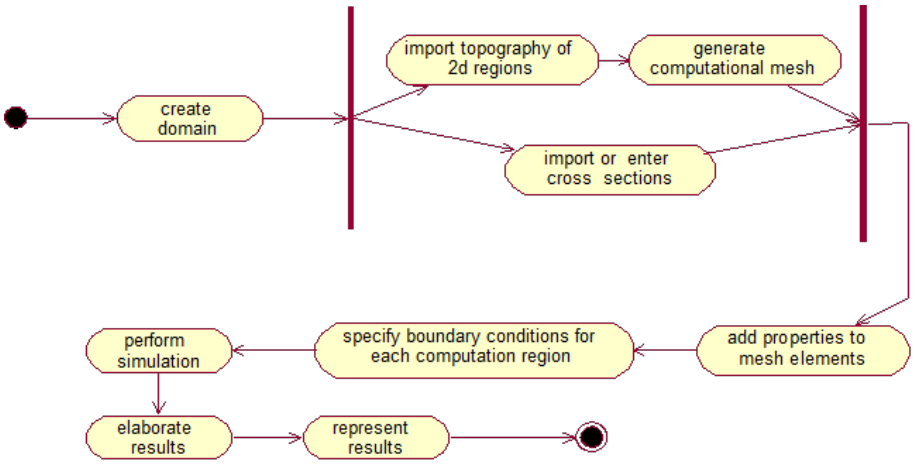


Figure 1.10 *Activity diagram Flood 1-D + 2-D*

2

Simulation procedure

2.1 Run BASEMENT with graphical user interface (GUI)

The start and executing of the BASEMENT software is described in the part “Introduction and Installation” of this manual. Further details concerning the GUI of BASEMENT are explained in Section 4.1.

2.2 Run BASEMENT on Microsoft Windows

When running BASEMENT under Microsoft Windows operating system, the easiest way to start a simulation is by double clicking on the command file ending with “.bmc”. Otherwise the program can be executed choosing the “Run. . . ” command for the Windows “Start”-menu or by double–clicking the executable file in Windows Explorer. After running, BASEMENT will open the graphical user interface where the command file can be loaded and the simulation can be started.

BASEMENT creates an initialization file in the user’s HOME-directory ‘bm.ini’, which stores the present work directory and scenario name to ease the input procedure for repeated simulations of the same scenario.

According to the existence of a main control block, either BASECHAIN_1D or BASEPLANE_2D, the appropriate simulation will be carried out.

2.3 Run BASEMENT on Linux

BASEMENT runs as a console application without screen graphics output. On LINUX you open a console and type ‘BASEMENT_vX.Y’ (replace X.Y with current version number) to start the executable (if no environment variables have been set, change into your ‘bin’ directory of the installation path). After running, BASEMENT will open the graphical user interface where the command file can be loaded and the simulation can be started.

BASEMENT creates an initialization file (as a hidden file) in the user's HOME-directory '.bm.ini', which stores the present work directory and scenario name to ease the input procedure for repeated simulations of the same scenario.

According to the existence of a main control block, either BASECHAIN_1D or BASEPLANE_2D, the appropriate simulation will be carried out.

2.4 Run BASEMENT in batch mode

Executing a simulation with BASEMENT normally opens the graphical user interface (GUI) and requires some input from the user, e.g. to select the model data and to confirm warnings generated by the program at the start and during run-time. But BASEMENT can optionally be started without any graphical interaction and without user input. This feature is especially useful if one or several models shall be run automatically via batch or script file.

But be aware that executing in batch mode requires special attention, since significant warnings may be suppressed without being noticed! It is recommended to study the generated 'log-file' after the simulation to check the program output for warnings which may have been generated during run time.

Executing in batch mode can be specified at the program start of BASEMENT using program arguments. The following list of program arguments is supported at the moment and can be specified in any order.

Table 2.1 *List of program arguments.*

Command line arguments	Description
- b	BASEMENT is run in batch mode without manual user input.
- f filename	The file flag '-f' and the space separated 'filename' argument specify the model filename which shall be executed. The filename must be the full path including the name of the *.bmc-file. (Please note: no empty spaces are allowed to be part of the filename!)
- version	Display the version number of the BASEMENT executable.
- h	This help flag '-h' displays all available command line arguments.
- doc	This generates a reference documentation of all blocks and parameters in .html format.
-log [1-4]	The number [1-4] defines the level of log output generated by BASEMENT.

Example:

Type the following line to execute the model 'Scenario1.bmc' in batch mode without user input.

```
BASEMENT_vX.X.exe -f d:\data\Scenario1.bmc -b
```

2.5 Restart the simulation in BASEplane from a given solution

In 2D simulations with BASEplane an improved and enhanced method for the restart from existing solutions from old simulation runs was implemented. Such a 'restart-file' contains all relevant information and data which is needed for the continuation of an old simulation and therefore often needs a lot of disk storage. These data are now stored in a binary format to reduce the needs for disk storage and to obtain smaller files. For this purpose a standardized CFD format was chosen (CGNS – CFD General Notation System, www.cgns.org). This standardized data format additionally can be edited and visualized by different programs and simplifies data exchange between different programs. (A simple Data Viewer for CGNS files is the adfviewer which can be found on www.cgns.org). The restart from an old solution is possible for the hydraulic computations, bed load transport computations and suspended load computations.

Using this new file format the possibilities to continue a simulation from a given solution were enhanced. It is possible to continue a simulation not only from the last point in time of the old simulation run, but also from different times of the old simulation. This can be very helpful especially for large simulation runs with long durations. For example, if there was an error in the inflow hydrograph at a point in time, than the simulation can be restarted shortly before the time when the error occurred with a corrected hydrograph and without having to repeat the whole simulation from beginning.

3

Preprocessing

3.1 General

3.1.1 Requirement

The main purpose of the pre-processing activities is to define the project and the different scenarios, to prepare the available (topographic) input data and to put it in the format needed by the computational module as shown in the overview in Section 1.1. Additionally, the choice of computational approaches and boundary/initial conditions has to be made.

3.1.2 Project / Scenarios

A project is defined for one region which is to be analyzed. Within a project, one or more scenarios can be embedded. To manage a project, it is therefore necessary to define which scenarios, files, and other elements belong to it. This includes the information where all these elements are stored and how they are connected to each other.

For the same project, several scenarios can be created. For each of them a simulation is executed. The different scenarios can vary regarding the computational mesh, other input data, the used approaches and the boundary conditions. In addition, type, location and time of the results to save have to be specified. The simulation can be done in 1, 2 or 3 dimensional computation or as a combination of them. The phenomena to be considered have to be chosen as well, as there are at the moment: dam break, bed load, suspended load, debris flow, mobile bed, erosion, collapse, pollutants etc.

3.1.3 Input data

As mentioned in the introduction, there are three main types of data to be provided for a simulation: topography, hydrology and sediment data. All data has to be transformed in a certain way to satisfy the input specifications of the main computation program. The precise specifications are available in the reference manual.

3.1.3.1 Topography / Computational mesh

The most important, and most laborious to achieve, input is the retrieval and setup of the computational mesh. It is based on the real world topographic data. At the end of the pre-processing task, a grid in a defined shape and format is available for the simulation module.

This mesh can be generated in different ways. The topographical raw data may come from a cluster of points described by three dimensional coordinates, digitized contour lines, break line polygons or cross sections and probably is furnished in different file formats, which have to be interpreted and transformed. For a 2-D simulation, in a first step a TIN (Triangulated Irregular Network) has to be generated. Then this mesh has to be modified and refined in order to satisfy the special mesh qualities needed for stability of computation. For the 1-D model it might be necessary to interpolate cross sections or deduce cross sections from a DEM (digital elevation model).

3.1.3.2 Hydrologic data

At all boundaries of the mesh, hydrologic boundary conditions such as hydrographs or time series of water levels have to be provided. In the case of a simulation with sediment transport, also a sediment concentration in the water might be needed.

The choice of boundary data has to be made with care, considering the type of event being simulated. The data might be created especially for the wanted simulation hypothesis or be adapted from existing measured or statistical data. Some times, special manipulation of the data turns out to be necessary, for instance to omit discharges not relevant for sediment transport. Finally, again, the hydrologic data has to be put in the format wanted by the program.

3.1.3.3 Sediment data

Sediment data primarily comes from water sediment samples or surface samples like the line method. Possibly, a grading curve still has to be built from this raw data. Then the number of desired grain classes for the simulation has to be chosen and the characteristic grain classes need to be identified. Based on the grain distribution, other values like roughness or angle of rest may have to be calculated.

3.1.4 Boundary conditions

The boundary conditions define quantities at the margin of the computational region during the whole simulation time. These are typically hydrographs at the inflow boundary and water levels at the outlet boundary. In general, different special conditions can be applied by defining for instance the water level of a lake, weirs, steps, etc.

3.1.5 Source terms and local properties

For every mesh element, several characteristics can be specified, for example not flown through cells, mobile bed, bed load potential, k-value, roughness, shear stress or others.

Sources and sinks of water and/or sediments might be added within the computational region.

3.2 Model input data

3.2.1 Topographic data sources

The topographic raw data in the form of digital terrain information builds the fundamentals for grid generation and further numerical simulations using the BASEMENT software. In the case of 1-D simulation, the raw data consists of recordings of river cross-sections. In a 2-D or 3-D model, the raw data is built from point clouds, height contour lines or a digital terrain model (DTM).

Dependent on the assignment and its requirements, most of the raw data usually gets collected by experts. However, there are some extensive topographic models with different quality available, see e.g. the cross-sections database from the Federal Office for the Environment (BAFU) and high resolution terrain models from “swisstopo” or “Swissphoto AG”.

The next sections provide a short overview of different methods and data sources of data sources for topography and also hydrology and sediment. Most data sources in the next few sections are somehow related to Switzerland. Other regions and countries provide similar data – depending on the actual law.

3.2.1.1 Terrestrial data or surveying with differential GPS

Surveying or special engineering agencies provide terrestrial terrain data or terrain data obtained by differential GPS (DGPS) of the desired quality. A pure terrestrial recording is more expensive because at least two workers are needed in contrast to a DPGS scan, which can be done by one employee only.

The survey with GPS needs a certain visibility concerning the GPS satellites. The accuracy depends on the exactness of positioning for the reference point but usually lies in the range of a few centimetres.

The terrain data is delivered in different formats as e.g. xyz-coordinates or GEOBAU standard. Be sure to specify your claims on quality of the data as resolution, typical terrain deformations but also the desired data format when asking for a tender offer.

3.2.1.2 Official topographic survey

According to a decision of the Swiss federal assembly (“Einführung der amtlichen Vermessung” AV93), the bureau of land charge register provides digital cadastral data almost over the whole country (ca 80%) and data from the country’s information system (ca 30%).

The cadastral information is just of secondary use as the parcels are all flat. However, the ground plans of certain constructions and buildings may be helpful.

The data from the country’s information system can be used (with some additional work) for the definition of boundary conditions, e.g. data about floor cover types.

The data format used for the AV93 act is called INTERLIS (Data exchange mechanism for "Landes-InformationsSysteme").

3.2.1.3 Laser scanning

Essential factors for an accurate triangulation concerning topographic raw material are accuracy of position and height as well as the density of the measured points. Of special interest are therefore height model data taken by Airborne Laser Scanning (LIDAR) as e.g. provided by "swisstopo". This method allows for a good data base within forests. Only in dense coniferous forest, the determination of the terrain model tends to be impossible. Of course, as with all measurements from the air, neither ground nor surface levels from water bodies can be obtained.

As the surface model generated from the LIDAR method distinguishes between terrain, buildings and vegetation, it provides a good starting point for further processing towards a numerical grid.

The federal office for topography ("swisstopo") has commissioned Swissphoto AG to collect height data all over Switzerland for the identification of agricultural areas. However, this project is restricted to areas below 2000 m asl. For higher located zones (which could be relevant for debris flows), the data could be produced with reasonable costs. Until end of 2007 this project shall be finished.

"swisstopo" delivers their terrain data in two formats:

- DTM-AV raw: a point cloud with averagely 1 point per 2 m^2
- DTM-AV grid2: an interpolated 2x2m mesh

The accuracy of position and height is about +/- 0.5 m. Additionally, "swisstopo" also works with 1x1m meshes which reach an accuracy below +/- 0.3 m.

For large areas and a high density of grid points, the amount of data is beyond the resources even of nowadays computing power. Therefore, there exists a variety of algorithms which eliminate all unnecessary points in a certain area that have no influence on the shape of the triangulation. This may reduce the amount of data by 10 – 60 %.

3.2.1.4 Sources and quality of data

One main factor for the quality of the results is the accuracy of the original topographic data. The accuracy varies depending on the objective of the recording, as well as on the recording methods. The accuracies of the most diffused methods are listed in Table 3.1.

Table 3.1 *Accuracy of current measurement methods*

	Situation accuracy	Height accuracy	Observations
Terrestrial	+/- 2 cm (1)	+/- 5 cm (1)	
Sonic depth finder	+/- 5-10 cm (1)	> 10 cm (1)	

	Situation accuracy	Height accuracy	Observations
GPS	+/- 1-3 cm (1) +/- 1 cm (4)	+/- 5 cm (1)	At least 4 satellites
Orthophoto		+/- 2-3 cm (1) 0.2 – 0.3 ‰ of flying altitude (3)	3 cm with photo scale 1:2000
LiDAR		+/-20-30 cm (2) open lea +/- 15-20 cm forest insecure (3)	
Radar (InSAR)		open lea +/- 20 cm (3)	

(1) *Gewässergeometrie, Landesanstalt für Umweltschutz Baden-Württemberg, Karlsruhe 1999.*

(2) *Swissphoto AG (verbal).*

(3) *Leitfaden Qualitätssicherung Photogrammetrie und DTM-Generierung, Konferenz der Kantonalen Vermessungsämter, 2000.*

(4) *GPS global positioning System, "swisstopo".*

The products which are directly available in Switzerland and their accuracies are listed in the following tables.

Table 3.2 *Products offered by "swisstopo"*

Product	Density	Source	Format	Accuracy
DHM25 Base Model DTM	35-1600 point /km^2	1:25000 map Vectorized contour lines and contour lines in lakes, lake perimeters and spot heights, main alpine break lines (photogrammetric data)	Arc View Shape- Files, DXF,GEN, BMBLT	Mean error 1.5 to 3 meters depending on the regions
DHM25 Matrix Model DTM	1600 point/km^2	Interpolation of the Base Model on a 25x25m grid	MMBLT, MMBL, AIGRID, XYZ, DXF, VRML	

Product	Density	Source	Format	Accuracy
(1)DTM-AV raw (2)DTM-AV grid2	1 point for 2 m^2	Based on official measurement with Lidar Airborne Laser Scanning(1) 2x2m grid interpolated from DTM-AV raw (2)	ASCII , Interlis, others possible if requested	Height accuracy +/- 50 cm
(1)DSM-AV raw (2)DSM-AV grid2	1 point for 2 m^2	With buildings and vegetation (1) 2x2m grid interpolated from DSM-AV raw (2)	ASCII , Interlis, others possible if requested	Height accuracy +/- 50 cm +/- 150 cm for vegeta- tion

Table 3.3 *Products offered by Swissphoto AG*

Product	Density	Source	Format	Accuracy
DSM	10x10m, 20x20m, 50x50m	Photogrammetric interpretation of aerial photos from 1995/96	ASCII xyz or GRID	Height accuracy: 3-5m midland and 7-10 mountains
DEM LIDAR (on commission)	31000/(250x160m), spacing 1-1.3m	Lidar Airborne Laser Scanning	ASCII xyz, ASCII ArcInfo, ...	Height accuracy 30 cm

3.2.2 River related data sources

The following list provides some more Swiss data sources for real world problem modelling and boundary conditions.

River cross-sections: Primary use for 1-D models; also qualified for 2-D models to improve topography data within a rivers area. Terrestrial mapping, optimal quality and format according to the cross-section database of the Federal Office for Environment FOEN (formerly FOWG / bafu.admin.ch).

Terrain topography: Basic dataset for simulations with spatial (2- or 3-D) models. Optimal quality and format (ASCII x y z) as swisstopo DTM-AV / DOM-AV (New swisswide high resolution DTM/DOM based on laser scanning). (http://www.swisstopo.ch/de/digital/dom.htm)

Surface texture and special buildings: Declaration of roughness coefficients, porosity and erosion resistance of surfaces; Optimal format INTERLIS-1, as e.g. DOM-AV or floor cover plans AV93 (generated by the cadastral register on demand of the cantons).

Sedimentological /geological data: Surface characterization of the riverbed (mostly using line samples), of the erosion capable underground (volume sampling or geological drilling) and of the flow induced transported material (grain size distribution from sieve analysis).

Possible data sources: geology of cantons, FOEN (formerly BWG), Nagra, in situ suspended load and bed load measurements.

The estimation of sedimentologic data and its analysis is surely the most time consuming part. At certain water bodies, a continuous monitoring of suspended load exist. However, the granulometric size distribution is seldom measured. Therefore, the bed load near the surface but also the suspended load often have to be estimated under certain assumptions or they will have to be measured.

Hydrological data: Temporal or stationary boundary conditions for the numerical model, e.g. inflow discharges, water levels and local sources or sinks.

Possible data sources: Hydrological data from the federal measuring facilities, specific rainfall-discharge models (e.g. IFU/ETHZ), retention models.

For larger water bodies within Switzerland (observed by the FOEN), time series and hydrographs are available online. Smaller watercourses are often not covered and need either a hydrologic model or new measuring/monitoring to determine the discharge.

3.2.3 Characteristic quantities of riverbed

The numerical models used for the computation of hydraulic behaviour are always declared to be either 1-D (flow in direction of main flux / x-axis) and/or 2-D (horizontal, depth averaged flow field). However, the fundamental terrain topography is always three-dimensional. The spatial discretization of the transport equations is based in 1-D on cross-sections or in 2-D on an unstructured grid of mostly triangles. The vertical component consists of different layers separated by a planar joint face for both, 1-D and 2-D simulations.

The characteristic values are indicated relative to a joint surface (e.g. riverbed level) or to a layer (e.g. granulometric composition of bed material in layer S1). Their properties are represented in the balance point of the corresponding joint plane. This results in a simplified model of the layering for a computational cell. The characteristic data used in the model to describe the state and conditions of the surface are defined as followed:

Position:

$$\text{Soil - properties} = \begin{cases} z_R: \text{ bottom elevation (bed level)} \\ z_{Sub}: \text{ elevation level of the lower, closing joint plane of the corresponding sublayer } Sub_i \ (i \in 1...n, \ with \text{ n = maximum number of Sublayers}) \\ z_0: \text{ elevation level of the lower joint face of sublayer Sn, which finishes the model downwards, respectively describes the position of the non erodable underground.} \end{cases}$$

(all variables are measured in [m.a.s.l.] or in [m] relative to a userdefined reference system)

Attributes:

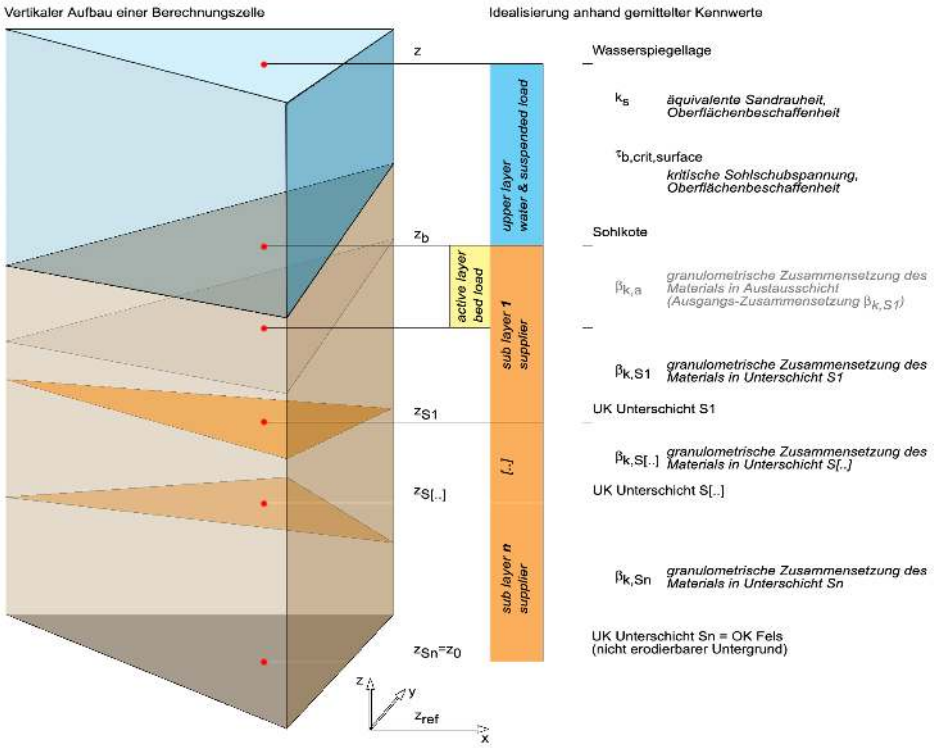


Figure 3.1 *Visualization of the vertical setup for a computational cell within the models.*

$$\text{Vegetation}-\text{ground covering} = \begin{cases} k_s\ [m]: \text{ equivalent sand roughness based on Nikuradse: description of the surface roughness (e.g. grass, sealed plane, etc.). (default value or derived by grain distribution of the first sublayer)} \\ \tau_{B,crit}\ [N/m^2]: \text{ critical value for begin of sediment movement at the surface, in the sense of a rised erosion constancy, e.g. by natural cover (default values)} \end{cases}$$

$$\text{Soil–properties} = \begin{cases} \beta_{g,Sub_i}\ [-]: \text{ mass fraction of the } g^{th} \text{ grain class relative to the total mass of sublayer (default values according to the grain size distribution)} \\ \beta_g\ [-]: \text{ mass fraction of the } g^{th} \text{ grain class relative to the total mass of the active layer (possible default values according to line samples if available)} \end{cases}$$

The listed attributes are mostly not completely determinable directly. For example, the equivalent sand roughness k_s based on Nikuradse is an experimentally obtained value for the flow resistance of different kind of surfaces. In the case of a more complex soil structure, k_s may consist of several components. Its actual value has to be determined by a calibration of the numerical model. The same holds true for the critical value $\tau_{B,crit}$ which defines the beginning of movement at the soil surface. This value is affected by the character of the bed surface like natural cover or sealing (at farmlands). Without special conditions, the critical value can be estimated via the grain size distribution.

The mass fraction β_{g,Sub_i} of grain class g in sublayer Sub_i has to be derived from an experimental grain size distribution using a sieve analysis. The material to be sieved may

come from a near surface sample take (e.g. by daggering) or a drill hole. The granulometric composition directly at the surface can be estimated by a line sample. Accordingly, the corresponding mass fractions β_g in the active layer can be determined.

3.2.4 Processing the raw data

It was shown in the previous sections that the numerical model needs some specific characteristic values which usually are not given directly by in situ measurements. Often, some experience in handling computational simulation models is necessary to reproduce the natural conditions in an adequate way and obtain realistic results.

3.2.4.1 Topography

The topographic raw data for 1-D or 2-D simulations comes in the form of cross-sections or by a digital terrain model (DTM). For a 1-D model, cross sections are mostly of accurate quality and need just few corrections and adaptations. However, most DTM's serving as a base for a 2-D model need an elaborate revision because of the surface triangulation:

- the resolution of the DTM and the computational grid are not congruent (e.g. areas of lower interest shall have a coarse grid to reduce computational effort – the DTM usually has a uniform point density)
- the course of water bodies is not continuous or cross sections are not complete (e.g. data only available up to the water level)
- passages are not omitted (e.g. bridges)
- modelling and representation of buildings is not a priori known (e.g. height of buildings, retention volume, flow resistance)
- relevance of waste edges and artefacts are costly to determine (e.g. small walls, temporary dumpsites, hedgerows)
- etc.

The choice of software resources for processing the topographic raw data is huge, but often, these programs are somehow limited to specific applications and data formats. It is therefore recommended to specify the desired data format and resolution in advance to avoid unnecessary repetitions.

More detailed information about grid generation is given in the next chapter.

3.2.4.2 Hydrology

The hydrologic raw data for discharge and water levels usually has a non uniform resolution in time. Often, the measurements are indexed with date and time. For the simulation module, the time series have to be converted to a system with standard units (e.g. seconds). In an easy case, the converted data can directly be used as a boundary condition. The simulation module will then interpolate the desired values to the actual computational time.

Further modifications may be necessary if e.g. just some time slots have to be simulated or if the main processes for a phenomenon like sediment transport occur at no more than a certain water discharge.

3.2.4.3 Grain size distribution

As mentioned earlier, the granulometric composition of the erodable material gets measured by a line sample or a sieve analysis and results in a grain size distribution (grain diameter versus weight percentages relative to the total sample weight). For the simulation, this distribution has to be discretized and desired grain classes have to be determined. Each grain class is defined by its medium diameter and has a percentage. The choice of the numbers of grain classes and thus the resolution of the material composition depends on the size of the problem and the computing power available. The classification of the grain diameters which represent a grain class strongly affects the transport behaviour of the whole material and the efficiency of the transport model.

It is recommended to run several simulations with one but also with more grain classes and to compare them.

3.2.5 GIS Interface

A geographic information system (GIS) as assistance for a numerical simulation model shall provide several data in best possible spatial and temporal resolution but not necessarily in a model inherent format. The data structures described in the following are just proposals. For all of them, one has to decide a priori whether the raw data shall be saved per se or in a processed form (raw data and pre-GIS-processed data).

3.2.5.1 Administration of geographical data

Generally, one distinguishes two kind of data types: matrix data (based on a uniform matrix, e.g. air taken pictures) and vector data (e.g. property borderlines).

Vector data serves for the visualization of the geometry and uses different data structures. Objects with simple geometry (see “OpenGIS Simple Feature Specifications”) are e.g. points, lines and polygons. For complex geometries (see “OpenGIS Geography Markup Language (GML)”), additional information as topology and other attributes (e.g. time) are required.

A further possibility to structure geographical data in a GIS is a splitting on different topic layers. Digital terrain models with height information are represented in a GIS either as TIN (triangulated irregular network) or as a GRID (structured mesh). With regard to numerical simulation models, the topology of a TIN or a GRID corresponds to unstructured resp. uniform computational grids.

3.2.5.2 Plane assignment for point data

In a GIS, real, area-related conditions are given in an idealized way. On the one hand, there is a limited number of attributes and on the other hand the values over the surface are averaged. Partially, the data base is just point wise available and just sparsely against resolution and process scales of numerical simulation models, e.g. at geological

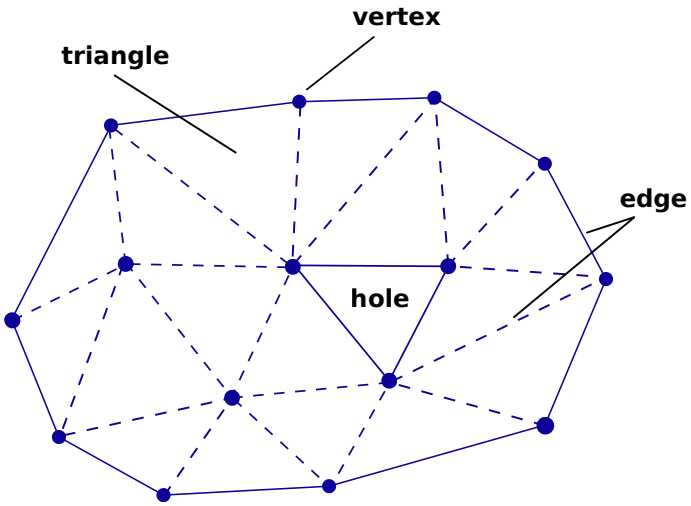


Figure 3.2 *Unstructured grid: Triangulation*

probe holes. Therefore, the coarse values have to be extrapolated on the corresponding surfaces of the numerical grid. The determination of the corresponding surface, or i.e. the surrounding polygon, needs different geometric constructs as e.g. Dirichlet Tesselation (Voronoi-Polygons). Additionally, one should consider related information from superior scales (e.g. geological maps with layering data and downcasts).

3.2.5.3 Interoperability and standard interfaces

To assure interoperability – meaning system independent communication – between different information systems, several norms for a standardized data representation exist. Examples are the "OpenGIS Geography Markup Language (GML)" or the norms used in Switzerland: INTERLIS (SN 612 030 / 612 031) and GEOBAU (SN 612 020).

3.3 Grid generation

3.3.1 Introduction

The numerical methods used for the approximation of differential equations needed in flow simulations are based on a discretization of the domain in simple small shapes. The complex of these elements forms a mesh. In BASEMENT, the finite volume method, being particularly valuable for fluid dynamics, is used.

To describe a topographic surface, the terrain data is often triangulated to allow a perspective representation of the topography. The result is a piece-wise linear interpolation of the surface. This triangulation can be used directly as mesh for simulations, as it permits to have the original data in the vertices of the initial grid and no interpolation is necessary. However, most of the time, the mesh has to be transformed somehow. Regions of high interest need some mesh refinement for higher accuracy and areas of lower interest are often coarsened to save computing power.

In numerical simulations, two types of meshes are used: structured and unstructured grids. Structured grids are simpler to use, but need an interpolation of the data to determine the

values at the desired vertex position if the input is an irregular point cloud. In addition, they are not sufficiently flexible to fit complex geometries.

Among unstructured meshes, the triangulated irregular networks (TIN) are most convenient because they fit the irregular distribution and complex geometry of the topography best. Furthermore, they allow for a rapid transition from small to large elements and the insertion and conservation of break lines. BASEMENT is built on unstructured grids.

The computational mesh consists of vertices, edges which connect the vertices and elements (control volumes) which are bounded by the edges.

The triangulation of terrain data is a special case of mesh generation, as the vertices do not only have a position in the coordinate system but also elevation information. This is a so-called 2,5 dimensional problem.

3.3.2 Topographical input data

The original terrain data typically has a very irregular distribution and a locally variable density. Data can be available in different shapes:

- A cloud of points (x, y, z) e.g. digital elevation model;
- Polygons e.g. digitalized contour lines;
- PSLG (planar straight line graph) e.g. a DEM with addition of break lines. This is the most general case, in which the input is a set of vertices and non-crossing line segments that must be conserved in the triangulation.

Although the raw input data, e.g. from a point cloud, could be directly used as a grid, usually further transformations are performed to gain a suitable computational mesh. A suitable mesh is mainly defined by its quality.

3.3.3 Mesh quality

3.3.3.1 Shape of mesh elements

The shape of the elements of the meshes has an important effect on the applicability of numerical methods. Speed (due to convergence time), accuracy and stability of a simulation depend strongly on the quality of the employed mesh. Therefore, it is important to produce the best possible triangulation for the application.

Size, shape and number of the elements play an important role for the quality of the mesh. Possible characterizations of a triangulation, which can be optimized depending on the application, are:

- minimal angle
- maximal angle
- maximum edge length
- total edge length
- maximum height

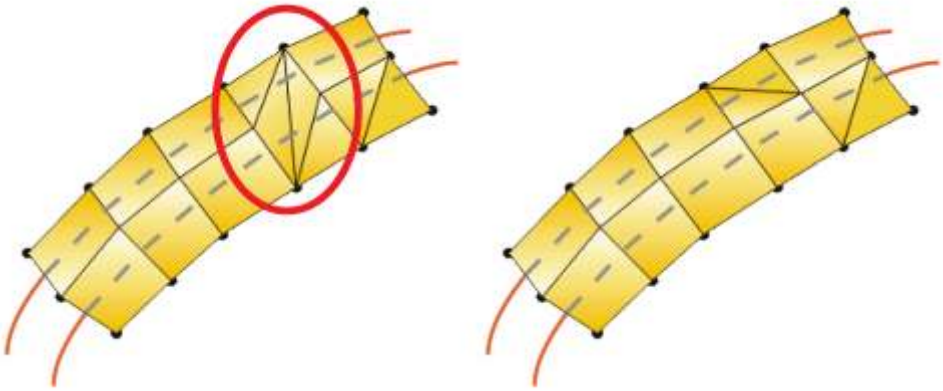


Figure 3.3 *Left: ambiguous quadrilateral element with false break line; right: correct discretisation of the dyke crest.*

- area of the triangle
- aspect ratio: ratio of the length of the longest side to the height (definition after Bern and Eppstein (1995), other similar definitions existent)
- aspect ratio: ratio of the circumference and the in-circle of a triangle
- roughness of the piecewise linear interpolation of a 2,5 D problem (The roughness of an interpolated surface can be measured e.g. by the Sobolev semi-norm)

The criteria which determine the quality of a mesh differ depending on the application (respectively partial differential equation) and the numerical method implemented. Also, the possible optimization depends on the constraints given by the original data. It is recommended to avoid L-shaped overall areas (non convex hull) as such problems often lead to numerical instabilities in the concave corner.

In general, a height aspect ratio, very small angles and especially very large angles are considered as bad, as they lead to a poor numerical condition of matrices and increase the approximation error, which arises with the element size in general, as well. Big differences of size between neighbouring cell elements can have a negative influence on the numeric simulation too.

3.3.3.2 Ambiguous gradients

A further mesh quality criterion specific to meshes consisting of quadrilateral elements, or meshes with mixed element types, is the problem of ambiguous gradients. Quadrilateral elements are defined by four nodal points. These nodes ideally have elevations which guarantee that the four nodes lie within a plane. But in case of strongly varying terrain topographies, a bad placement of quadrilateral elements can lead to situations where this is not the case. Such elements are deformed and have an ambiguous gradient. Figure 3.3 illustrates a quadrilateral element with ambiguous gradient which is situated across a river dyke.

In such situations a splitting of the quadrilateral element into two triangles becomes necessary. The selection of the correct break line within the quadrilateral element (here, along the dyke crest) must usually be done manually according to local topography. To

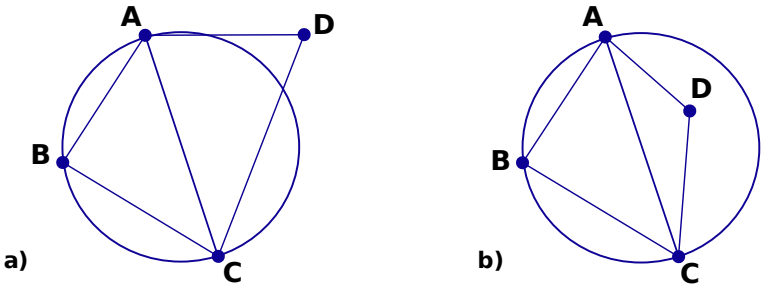


Figure 3.4 *a) Empty circle criterion satisfied. b) Empty circle criterion not satisfied*

facilitate this task, most grid generating tools (e.g. SMS) offer special features for detecting quadrilateral elements with ambiguous gradients in the mesh.

3.3.4 Issues on triangulation

Mesh triangulation and grid refinement play an important role in almost every numerical simulation. Therefore, a lot of different techniques have been developed to achieve suitable computational meshes. The user is basically free to use any tool or method which generates a mesh of accurate quality out of the raw data. This section shall give a short overview on some popular triangulation methods.

BASEMENT does not provide an automated routine for mesh generation. Therefore the plugin BASEmesh for the free and open source geographic information system (GIS) software *Quantum GIS* (QGIS) was developed (see Section 3.3.5).

3.3.4.1 Delaunay triangulation and constrained Delaunay triangulation

The Delaunay triangulation is one of the most employed triangulation methods because it optimizes several quality criteria: It maximizes the minimal angle and minimizes the maximum containment circle radius, the maximum enclosing circle radius and the roughness of a piecewise-linear interpolation. It also provides good results regarding the minimization of the maximum angle, but it does not find a global optimum in this case.

A Delaunay triangulation has the following properties. It

- is the straight line dual of the Voronoi diagram;
- is unique;
- respects the circumcircle criterion;

The circumcircle criterion is respected if the circumcircle of every interior triangle does not contain other points.

This corresponds to say that if ABC + CDA $< 180°$ the empty circle criterion is satisfied.

- respects the edge circle property: for each edge exists some point-free circle which passes through the end points;
- respects the neighbour property: an edge formed by joining a vertex to its nearest neighbour is an edge of the triangulation.

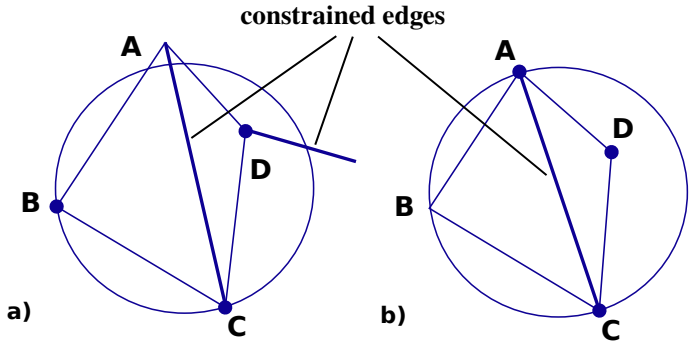


Figure 3.5 *a) edge circle criterion. b) empty circle criterion*

3.3.4.1.1 Constrained Delaunay triangulation (CDT)

The terrain data is sometimes provided in the shape of a PSLG as it contains break lines which must be conserved as edges in the triangulation. In this case, the constrained Delaunay triangulation can be used.

For the definition of a constrained Delaunay triangulation the notion of visibility of a point is needed: In a PSLG domain P a point D is visible to a point C if the open line segment CD lies within the domain and does not intersect any edges or vertices of P. Point D is visible to CB if it is visible to some point on CB.

For the CDT the edge circle and the empty circle criterion are modified as follows:

- Edge circle: for each edge a circle passing through its end-points containing no other point of the domain visible to the edge exists;
- Empty circle: the circumcircle of every triangle contains no points visible to points inside of the triangle.

3.3.4.1.2 MinMax triangulation

The MinMax triangulation minimizes the maximum angle. It has proven to be very useful in CFD (Barth, 1994).

3.3.4.1.3 Data dependent triangulation

A data dependent triangulation can be used for a 2.5 d problem. Its aim is to find the best triangulation under data dependent constraints, by minimizing a local cost function of a piece-wise interpolation. Two examples of local cost functions are described in (Barth, 1994).

3.3.4.1.4 Steiner triangulation

A Steiner triangulation is a triangulation in which extra points are added to the original data to improve the quality of the mesh. The additional points are called Steiner points. The number of Steiner points must be limited, limiting also the quality of the mesh.

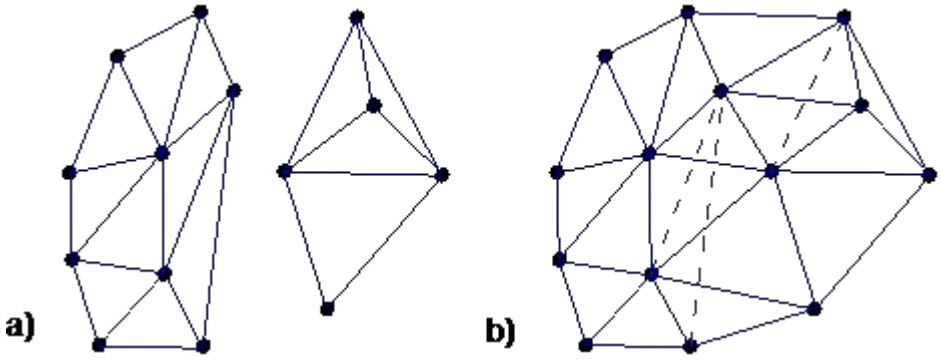


Figure 3.6 *a) triangulated half planes. b) merged triangulation*

3.3.4.1.5 Non obtuse triangulation

One of the most interesting types of Steiner triangulations is the triangulation without obtuse angles. It is a Delaunay triangulation or constrained Delaunay triangulation and also minimizes the maximum angle and maximizes the minimum height (Bern and Eppstein, 1995).

3.3.4.2 Algorithms

3.3.4.2.1 Sweep-line algorithm

The sweep-line algorithm can be used to perform a first triangulation without any quality criteria. It adds the points by x-coordinate order and then connects them to all visible points of the convex hull of the existing triangulation.

3.3.4.2.2 Divide and conquer

For the divide and conquer algorithm the point set is divided in two half planes along the x axis. Then each half plane is triangulated recursively, and finally the two planes are merged.

3.3.4.2.3 Incremental Insertion algorithms

The incremental insertion algorithms successively insert new points to an existing Delaunay triangulation. These algorithms have a worst case running time of (n^2) if the points are badly ordered. But in practice it is near to 0(nlogn) for Green-Sibson.

Watson (or incremental delete and build)

In this algorithm, after the insertion of the point, all the triangles containing the edge q are searched. Then the edges of these triangles visible to q are deleted and the new edge is connected with the vertices of the originated polygon creating new edges.

Green – Sibson

After its insertion, the point q is connected to the vertices of the triangle that contains q.

Then all suspect edges have to be swapped to satisfy the empty circle criterion if we want to obtain a Delaunay triangulation. Other edge swapping criteria can be used, for instance the minimization of the maximum angle.

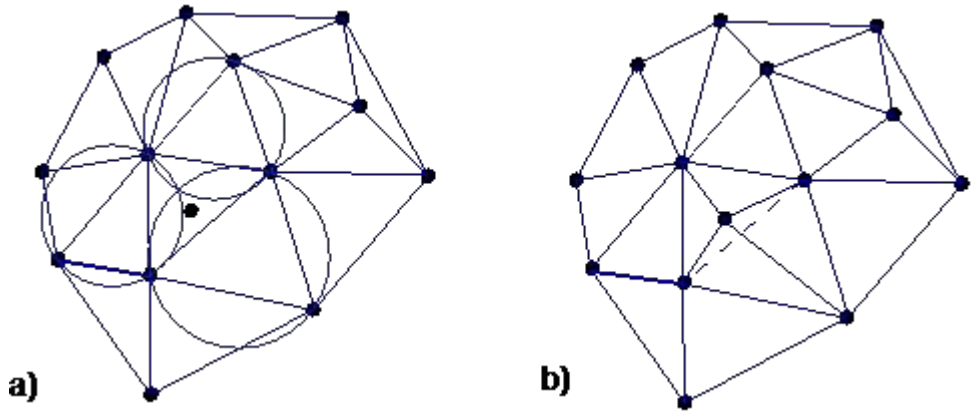


Figure 3.7 *a) point insertion and determination of affected triangles. b) new triangulation*

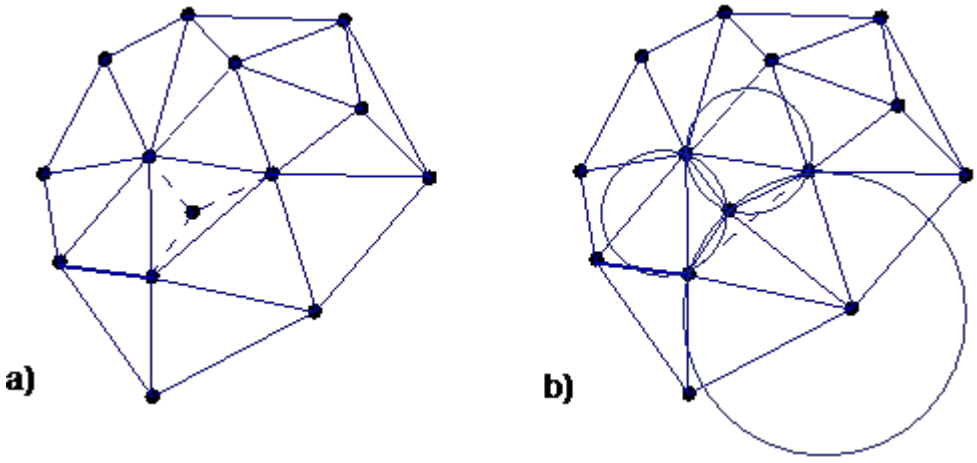


Figure 3.8 *a) insertion of the new point b) edge swapping based on empty circle criterion*

3.3.4.3 Edge flipping algorithm

The edge flipping algorithm is based on the local optimization of an initial triangulation. For each quadrilateral formed by a convex pair of triangles the diagonal is chosen with regard to a local optimization criterion.

Possible optimization criteria:

- Empty circle criterion: a Delaunay triangulation or constrained Delaunay triangulation is obtained. In this case a global optimum is reached;
- Minimize the maximum internal angle for both triangles: gives a MinMax triangulation only locally optimized;
- Optimize a local cost function: leads to data dependent triangulations (only locally optimal triangulation) (Barth (1994)).

Other properties that can be optimized with edge flipping are for instance vertex degree or total edge length, but a global optimum is not guaranteed. If the algorithm is used to obtain a CDT, the constrained edges simply are not tested because they can not be swapped.

3.3.4.3.1 Edge insertion:

This algorithm solves the problem of minimizing the maximum angle in time $O(n^2 \log(n))$ exactly. Like the edge flipping algorithm, it starts from an arbitrary initial triangulation. It incidentally inserts candidate edges on a vertex of a worst triangle. The crossed edges are removed, and the remaining regions are re-triangulated. If the triangulation gets worse the added edge is rejected. This algorithm can also be used to find an interpolating surface with minimal slope in time $O(n^3)$ (Bern and Plassmann (2000)).

3.3.4.3.2 Blue, red and green refinement

This is another popular method for mesh refinement. Basically, an element with an aspect ratio out of bound gets divided into two elements by inserting a new vertex on the midpoint of the longest edge. This is the so-called green refinement. The new vertex has to be connected on both sides of the edge with the opposing vertices of the neighbouring elements.

If one of the two resulting triangles still has a bad aspect ratio, the element gets divided again in the same manner resulting in three new elements. This is called the blue refinement. It is the goal of the blue and green refinement to halve the longest side of the original triangle and whereas degeneration of the mesh in repeated mesh refinement steps is prevented. Finally, the red refinement connects all midpoints of the original edges which leads to four new elements.

3.3.5 Use of QGIS plugin BASEmesh for grid generation

In order to provide a free and open source solution for the creation of computational meshes (Pre-Processing) and to visualize simulation results (Post-Processing) the plugin BASEmesh for the free and open source geographic information system (GIS) software QGIS was developed.

We hope to enforce this powerful free GIS-solution in future to allow for a workflow which is completely based on non-commercial software. A further advantage of this workflow is that the involved programs are portable and are available for Windows-OS as well as Linux-OS.

3.3.5.1 Introduction

The plugin BASEmesh is a tool for creating computational meshes, based on the advanced mesh generator TRIANGLE by Jonathan R. Shewchuk. Furthermore, features for loading and editing existing meshes are provided. At the moment, BASEmesh only supports the automated creation of meshes consisting of triangular elements. Meshes containing quadrilateral elements, however, can be imported, manually edited and exported.

Beside 2D meshes, the plugin is also able to convert 1D meshes of different formats into each other (at the moment conversions are possible between BASEchain and HEC-RAS). The meshes created by BASEmesh can directly be used for computations with the basic simulation environment software BASEMENT. Post-processing, i.e. the visualization of simulation results, is not directly supported by BASEmesh. Still, there are various options for displaying and processing BASEMENT results in QGIS, which is briefly explained in Section 3.3.5.9 and detailed in the Post-Processing tutorial.

The manual at hand introduces the features of BASEmesh and the basics of its usage. No information is given about the usage of GIS software in general. Some basic knowledge in applying GIS software might be acquired using QGIS' excellent documentation. To get in touch with BASEmesh and QGIS fast and easy, a step by step tutorial is provided as well and can be downloaded as separate document. This manual provides the reader with an overview of the capabilities of BASEmesh, the installation procedure, and a short description of the BASEmesh components that enable creating, importing, editing, and exporting computational meshes.

3.3.5.1.1 Installation

BASEmesh is at present available on a specific Plugin repository which has to be connected manually in the QGIS plugin manager. In contrast to other plugins, it is not available via the official QGIS plugin repository which is set as default in every QGIS installation.

To install BASEmesh, follow these steps:

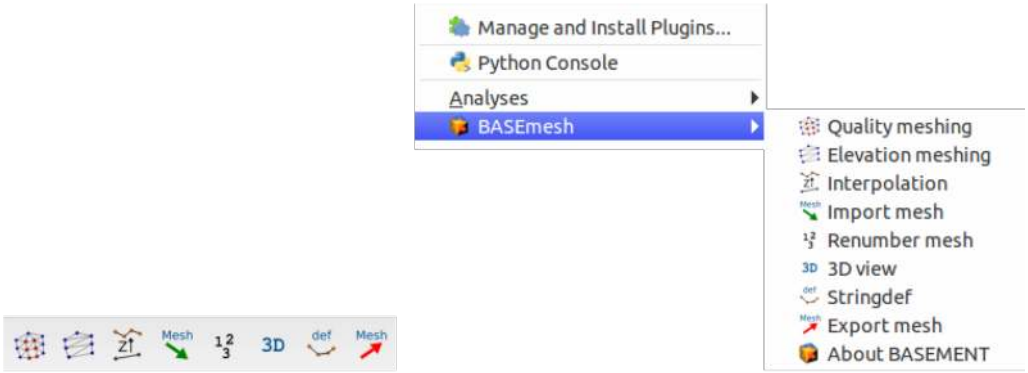
(1) Start QGIS

(2) Load the QGIS plugin manager by choosing *Manage and Install Plugins...* in the menu *Plugins* in the QGIS main toolbar

(3) Go to *Settings* (you should now see the connection to the official QGIS-plugin repository)

(4) Click on *Add...* and give a name, e.g. 'BASEmesh repository'

(5) Enter the repository address: http://people.ee.ethz.ch/~basement/qgis_plugins/qgis_plugins.xml (do not copy paste this address, because it might include line breaks)



(a) Access via toolbar (b) Access via menu *Plugins*

Figure 3.9 *BASEmesh representation in QGIS after successful installation*

(6) Press *OK*

(7) The additional repository should now be visible (make sure that the *Status* is *connected*)

(8) Go to *All* in the menu of the plugin manager and search for 'BASEmesh'

(9) Choose the BASEmesh plugin (if several are available, choose the one with the highest version number) and press *Install plugin*

If everything worked, you should be able to access BASEmesh via the toolbar and via the menu *Plugins* (see Figure 3.9).

3.3.5.1.2 Components of BASEmesh

The philosophy of creating a computational mesh within BASEmesh is as follows (see Figure 3.10):

Generate a quality mesh Here you produce the actual computational grid ('quality mesh') with all its attributes you define, i.e. domain area, element sizes, mesh holes, breaklines or enforced vertices. This mesh does not contain any topographic information. The quality of this mesh influences the results of your numerical analysis, e.g. stability, computation time, accuracy etc.

Provide an elevation model The topography of your computational domain is usually described by points in a text file containing the (x,y,z)-coordinates or by a raster dataset. In case of xyz-data the elevation model has to be formulated as triangulated irregular network (TIN), called *elevation mesh* within BASEmesh.

Interpolation The topographical information contained in the elevation model will be interpolated on the quality mesh, i.e. an elevation value is assigned to each node of the computational grid.

In addition to mesh creation the possibility of manipulating an existing mesh is provided as well in BASEmesh (see Figure 3.10). Starting from a mesh you have created previously or you have imported to QGIS, you can edit the following:

Material indices This attribute is used in BASEMENT to distinguish between different materials which can be used to define friction, soil composition etc.

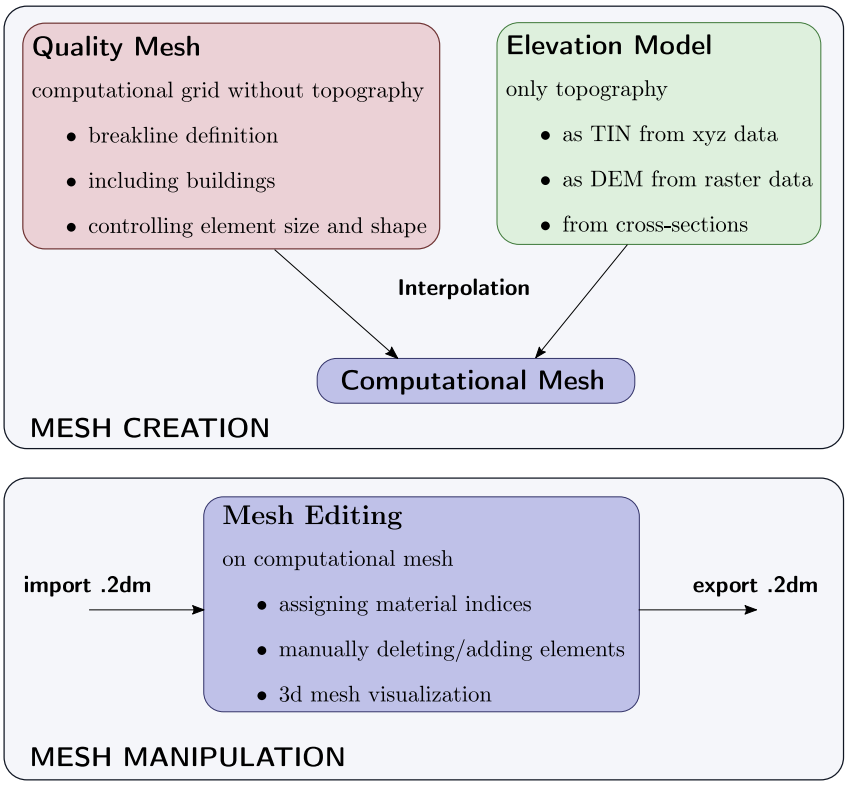
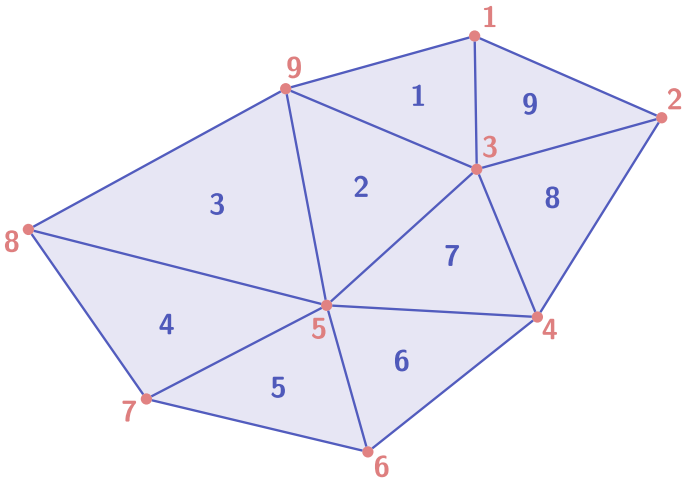


Figure 3.10 *Pre-processing components of BASEmesh*

Combine meshes Multiple meshes can be combined to one big mesh, by carefully applying manual meshing.

Renumber your mesh After applying manual changes to your mesh, you need to make sure that the internal representation of the mesh is correct. To this end BASEmesh provides a tool called *Renumber Mesh*, where all the manual changes are incorporated into a valid mesh structure (see Section 3.3.5.1.3).

Find *stringdef* nodes After changes of the computational mesh the node IDs may change as well, and all *stringdef* definitions of BASEMENT have to be updated. To this end BASEmseh provides a tool called *Stringdef*.



Node 4	
NODE_ID	4
X	635000
Y	278000
Z	486.5

Element 3	
ELEMENT_ID	3
NODE1	8
NODE2	5
NODE3	9
NODE4	-1
MATID	1

Figure 3.11 *Example of internal mesh representation with nodes as point layer (red) and elements as polygon layer (blue).*

3.3.5.1.3 Mesh respresentation in BASEmesh

Meshes generated by BASEmesh follow a specific structure. When manually editing a mesh it is of high importance to maintain this structure, otherwise it is not possible to export the mesh to a 2dm - grid which can be used for computational purposes. The mesh structure of BASEmesh is defined as follows (see Figure 3.13):

- In BASEmesh, meshes are represented by two vector layers: (1) '*meshName*_nodes' containing only the nodes of the mesh and (2) '*meshName*_elements' containing only the elements of the mesh.
- '*meshName*_nodes' contains point features (2-dimensional geome). The attributes for each feature are *NODE_ID*, *X*, *Y*, and *Z*. Please note thate the elevation information is only contained in the attribute table of the vector file.
- '*meshName*_elements' contains polygon features (2-dimensional). The attributes for each feature are *ELEMENT_ID*, *NODE1*, *NODE2*, *NODE3*, *NODE4*, and *MATID*. The attributes *NODEx* stand for the *NODE_ID*'s within the layer '*meshName*_nodes' and define the mesh connectivity.
- The *MATID* attribute is used to set material indices for each element within the mesh. BASEMENT uses these indices to set up zones with different characteristics in the mesh, e.g. to set different friction values in the river bed and in the flood plain of the domain.
- In case of triangular elements the attribute *NODE4* is -1. In case of quadrilateral elements, however, the *NODE4* attribute has to be specified.

Within QGIS, it is possible to create elements without defining the attributes *NODEx*. In this case, the tool *Renumber Mesh* must be used to update the connectivity of the mesh before exporting to a 2dm - grid.

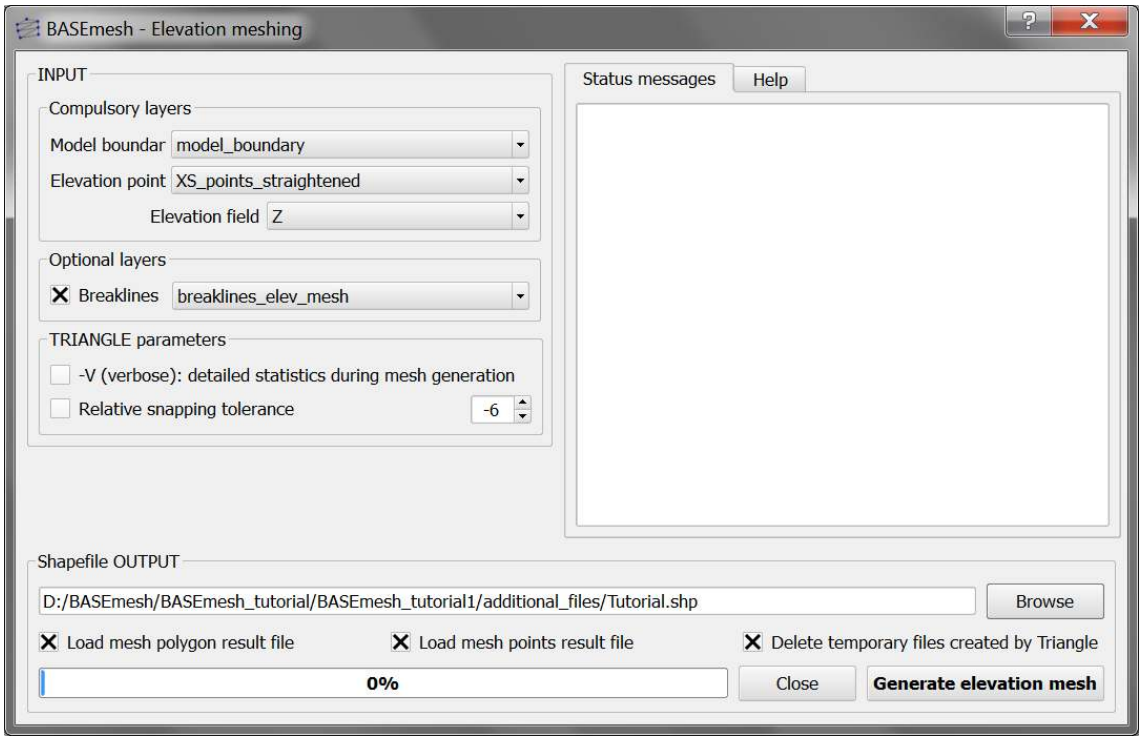


Figure 3.12 *User interface for the creation of elevation meshes.*

3.3.5.2 Elevation Meshing

The elevation mesh consists of triangulated elevation points and corresponding breaklines. It represents the topography of the computational domain and organizes the elevation data into a geometric structure which can be used for future elevation interpolations. The elevation mesh needs to be distinguished from the quality mesh, since it does not consider any quality aspects of the mesh elements and therefore should not be used for computations. The following data sources are combined:

1. *Elevation points* with x,y,z-coordinates. These elevation points may be taken from DTMs, cross section profiles, field measurements and other data sources. As input, all elevation points need to be combined into one single point shapefile.

2. *Breaklines* defining distinct interruptions of the the surface slopes (e.g. dyke crests or river side walls). Breaklines are given as line shapefile and connect elevation points with each other. During mesh generation, no triangles crossing the breaklines will be created. Be aware that all end points or vertices of a breakline must have corresponding elevation points!

3. *Model boundary* defining the extents of the elevation mesh. It must be given as polygon shapefile, which connects elevation points with each other. Be aware that all vertices of the model boundary polygon must have corresponding elevation points!

The elevation mesh is generated automatically with the program Triangle, without any parameters to be set.

3.3.5.3 Quality Meshing

The quality mesh defines the computational elements and nodes as they are used for the numerical computations. In contrast to DTM-models or TIN of elevation data, it has high demands concering the quality of the elements (element size, element angles, shape, etc.), which directly influence the quality, stability and performance of the computations. It is important to have smooth transitions between elements with different sizes and to prevent distorted elements with small angles or very small element sizes. In general, a high-quality mesh is one which is nice to look at! The quality mesh contains all geometric information (x-y plane), but has no topographical information (z plane). Quality mesh generation involves mainly two steps:

1. Specification of all geometric information about the computational domain:
   - *Model boundary*: extent of the computational domain.
   - *Break lines*: distinct interruptions of the the surface slopes (dyke crest, river side walls, ...) which shall be preserved in the computational mesh.
   - *Holes*: parts within the mesh which are excluded from modelling (e.g. buildings). These parts are defined by special points (layer region_points) surrounded by breaklines.
   - *Vertices*: enforced geometric points in the mesh (e.g. measurement points).
2. Generation of triangles respecting specific quality criteria using the free program TRIANGLE. Parameters of major importance are:
   - *Maximum area* constraints: definition of the mesh density using *maximum area* contraints for the triangular mesh elements. The *maximum area* is defined as attribut in the layer region_points and holds for a specific *Region* surrounded by breaklines.
   - *Dividing constraints*: With this attribut in the layer *Breaklines* one can enforce a certain number of mesh elements along a breakline. This is of major importance for the use of inner boundaries in BASEMENT, where an equal number of mesh elements at the uptream and downstream interface is required.
   - *Minimum triangle angle*: no elements with angles smaller than the minimum angle specified are generated (smaller angles lead to less elements, while larger angles lead to more elements).
   - *Relative snapping tolerance*: defines, how far two point coordinates may be located apart to still be considered at the same location. The default value is 10E-6. Increasing this tolerance can help to avoid problems due to improper snapping of vertices (line or polygon features) and points in OGIS.

Since the quality mesh has no topographical information, it is usually necessary to interpolate data from an elevation model on the mesh nodes before using it for computations.

3.3.5.4 Interpolation of Elevation Data

An important task in mesh generation is the interpolation of elevation data onto the nodes of the quality mesh. As a result, the final computational mesh is obtained, which then can

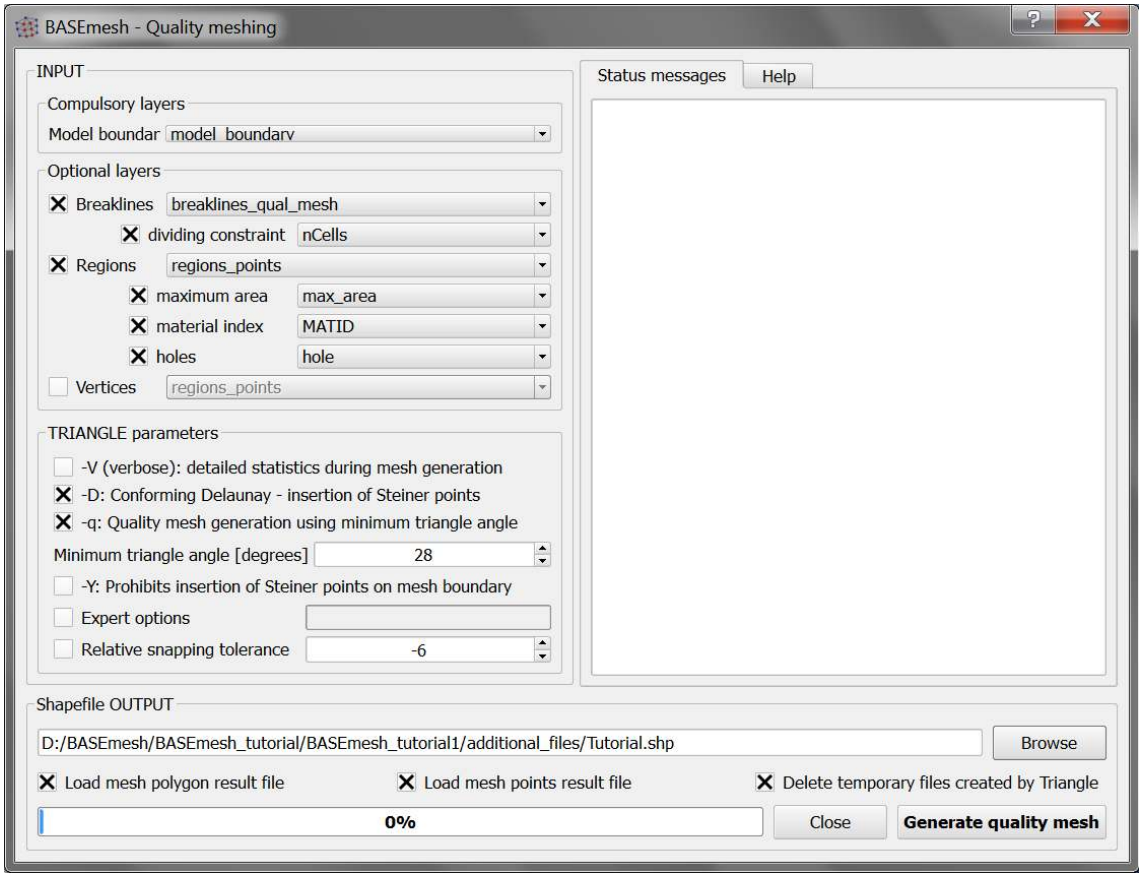


Figure 3.13 *User interface for the creation of quality meshes.*

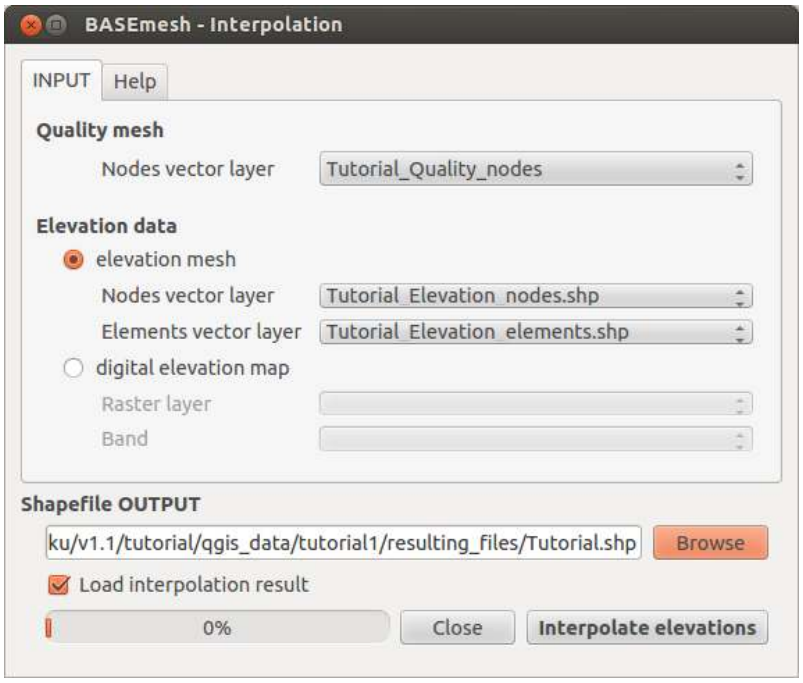


Figure 3.14 *User interface for the interpolation of topographical data onto quality meshes.*

be exported and used for simulations. The elevation data serving as input can be provided in two different elevation model types:

1. *Elevation mesh* triangulated from pointwise elevation data (TIN). The routine identifies the coordinates of each quality mesh node and determines any underlying elevation mesh element. Two methods are used for data interpolation:

   a) If an underlying elevation mesh element is found, the elevation of the quality mesh node is interpolated at its x-y-coordinates. This is the normal case, since the elevation mesh usually covers the whole computational domain. Nodes interpolated with this method are marked by a 1 in the element - field of the node attribute table. If the quality mesh node is located at the exact coordinates of an elevation mesh node, its height value is preserved exactly.

   b) If no underlying elevation mesh element is found, the quality node elevation is set to that of the nearest node of the elevation mesh. This is the case if quality mesh nodes lie outside the domain covered by the elevation mesh or when holes are present in the elevation mesh. It may lead to incorrect quality mesh ndoe elevations. Hence, it is recommended to choose a bigger domain for elevation meshing than for quality meshing. Nodes interpolated this way are marked by a 0 in the element - field of the result attribute table and are named ‘with special treatment’ in the QGIS status messages.

2. *Digital elevation map* as raster data which contains the topography as DTM. The raster elevation data is directly mapped on the computational mesh nodes without interpolation. If no corresponding raster cell is found, the elevation is set to ‘-999’.

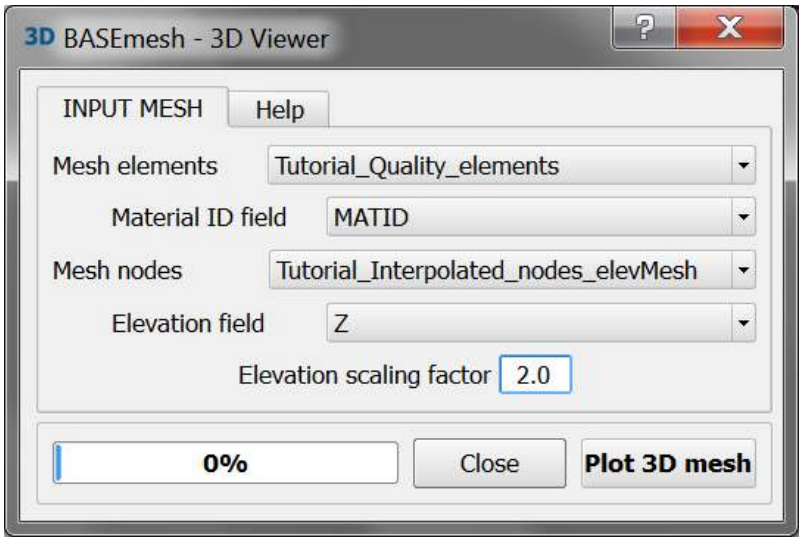


Figure 3.15 *User interface for 3D view.*

3.3.5.5 Visualizing meshes in 3D

During the process of mesh generation you can have a look at your mesh using the 3D viewer. The intention is to only have a quick glance to the third dimension of your mesh to detect potential errors or shortcomings in your mesh visually.

The 3D view tool requires the specification of a polygon shapefile with mesh elements and a point shapefile with the corresponding mesh nodes. The mesh nodes need to have elevation attributes and the mesh elements need to have material indices specified in their attribute table. Changing the *Elevation scaling factor* will stretch (>1.0) or squeeze (<1.0) the third dimension of your mesh view.

The window popping up shows your mesh (if only parts of your mesh is visible, zoom out). You can change the view as follows:

- zooming in and out: scroll using the wheel of your mouse
- shifting: drag and drop using the wheel button of your mouse
- rotating: drag and drop using left-click of your mouse (the orbit centre is located in the centre of your mesh)

3.3.5.6 Renumbering Meshes

An unstructered mesh generally consists of a list of nodes, a list of elements and the mesh connectivity logic behind. The elements are formed by three or four nodes and dispose of a certain orientation (here: clockwise orientation). If a mesh is generated via BASEmesh using *quality meshing* or *elevation meshing*, then the mesh connectivity is always in a valid state. However, if the mesh is modified afterwards, e.g. by deleting or adding elements/nodes or swapping edges, then the mesh connectivity becomes invalid. Therefore, after modifications, the mesh connectivity has to be updated. This task is performed using the *Renumber Mesh* tool:

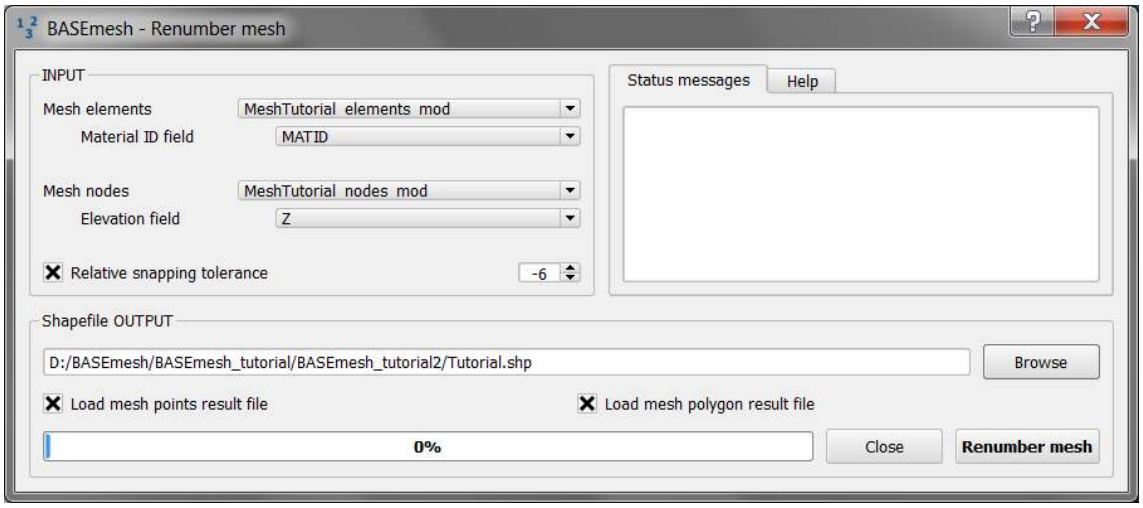


Figure 3.16 *User interface for renumbering meshes.*

1. The mesh connectivity is checked and updated. For modified or added elements, the corresponding node ID-numbers are searched by comparisons of x-y coordinates. Furthermore, the clockwise element orientation is established by re-ordering the element nodes if necessary.

2. Nodes and elements are renumbered in a way that their ID-numbers begin ascending from '1' without gaps.

Renumbering requires a mesh as input data, i.e. a polygon shapefile containing the mesh elements and a point shapefile containing the mesh nodes. In addition, all elements must have material indices specified in their attribute table and all nodes must have elevation attributes. After renumbering, the mesh is in a valid state and can be exported for computations.

3.3.5.7 Finding Node IDs for *stringdef*

In BASEMENT a list of node IDs is defined as *stringdef*. They can be used to define a boundary condition or an output along these nodes. The IDs correspond to the node IDs of the computational mesh. As soon as there are changes in this mesh, or only in its connectivity, the IDs will change as well and the list of node IDs inside of BASEMENT have to be updated. This task is automated with the tool *Stringdef* of BASEmesh:

1. For each line feature with a non-empty stringdef-field, all nodes that are located exactly on that line, are listed in a text file (BASEMENT-like stringdef block). The content of the stringdef-field represents the stringdef name.

2. The stringdef line features are favorably included in the layer where the breaklines of the quality mesh are defined. To distinguish between regular breaklines and stringdef lines, the stringdef-field can be used (empty vs. non-empty).

3. The upstream direction of the generated stringdefs is *right*. Therefore the line feature has to be drawn from the left riverbank to the right riverbank.

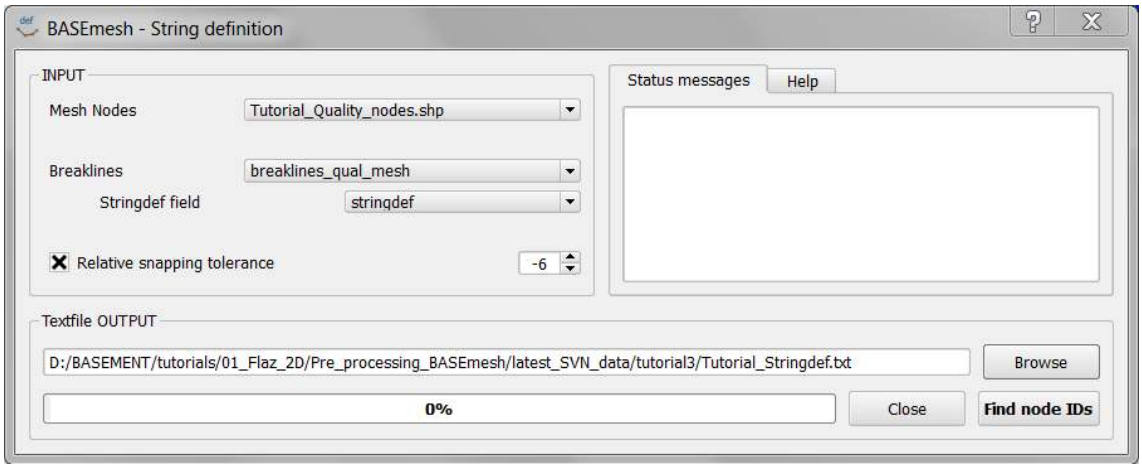


Figure 3.17 *User interface for Stringdef.*

The Stringdef tool requires a point layer (i.e. the computational nodes) and a line layer (i.e. the breaklines with at least one stringdef line feature) as input. The resulting list(s) of node IDs is written as text file output that is user defined.

3.3.5.8 Import and Export of mesh - files

2D mesh files:

The *.2dm*-format is an ASCII data format for unstructured meshes specified for the Mesh Module of software SMS (www.aquaveo.com) . Due to its simplicity, this data format is used by a variety of surface water modelling programs, as e.g. BASEMENT. A *.2dm*-file contains the node list with all coordinates and the elements with the mesh connectivity.

The main task of BASEmesh is the generation of a computational mesh in *.2dm*-format, which can be directly used for numerical simulations with BASEMENT. Therefore it is necessary to export the generated mesh to the .2dm-format. Furthermore, it is also often useful to visualize an existing *.2dm*-mesh, to change its material indices, to extract node or element numbers or to modify other mesh properties. For these purposes, BASEmesh provides options for importing and exporting *.2dm* - meshes.

1D mesh files:

Beside 2D meshes, BASEmesh is also able to import and export 1D grid files. These file formats can be converted into each other. At the moment, two file formats are supported:

- BASEchain geometry files (*.bmg*), and
- HEC-RAS geometry files (*.g01*).

These 1D functions are useful for visualization purposes of 1D grids, as for example the lateral coupling of 1D and 2D models. Furthermore, the conversion utility allows for using the HEC-RAS software for pre-processing of 1D geometry files (e.g. extracting cross sections from digital elevation models (DEM) or interpolations of cross sections).

Import mesh

After specification of the locations of the input and output files, the 2D / 1D mesh-file can be imported into QGIS. The mesh data are stored in shape-format within QGIS.

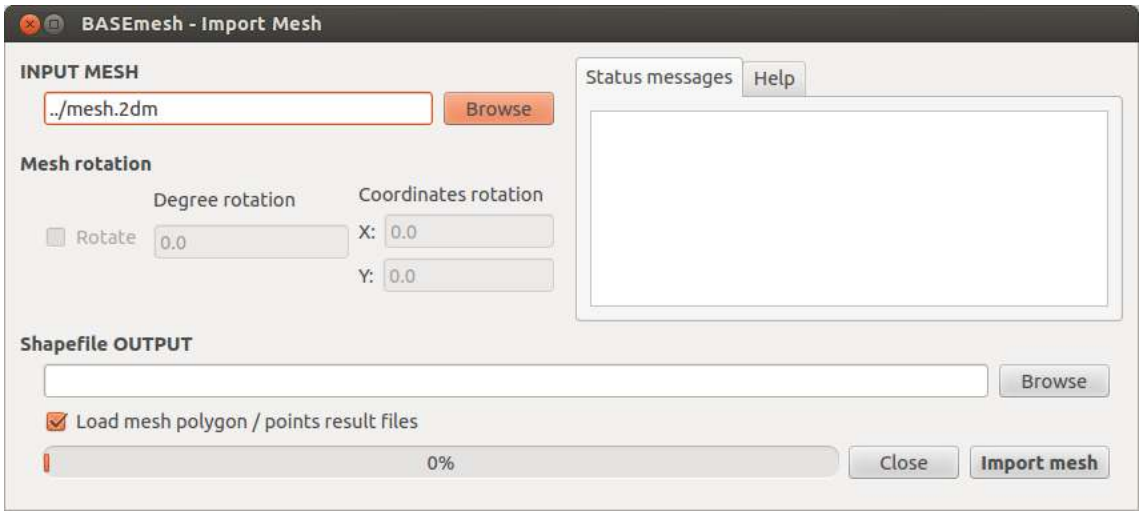


Figure 3.18 *Importing an existing mesh from a .2dm-file.*

- 2D: Two shapefiles are generated with (a) element data (suffix: '_elements.shp') and (b) node data ('_nodes.shp'), containing the mesh connectivity information and material indices.
- 1D: Two shapefiles are generate with (a) cross section data (suffix: '_crossSections.shp') and (b) point data ('_points.shp'), containing the cross section data and the cross section points.

Export mesh

- 2D: Export into the 2D mesh-format requires the specification of a polygon shapefile with mesh elements and a point shapefile with the corresponding mesh nodes. The mesh elements need to have material indices specified in their attribute table and the mesh nodes need to have elevation attributes.
- 1D: Export into the 1D mesh-formats requires the specification of the file format and the cross section shapefile.

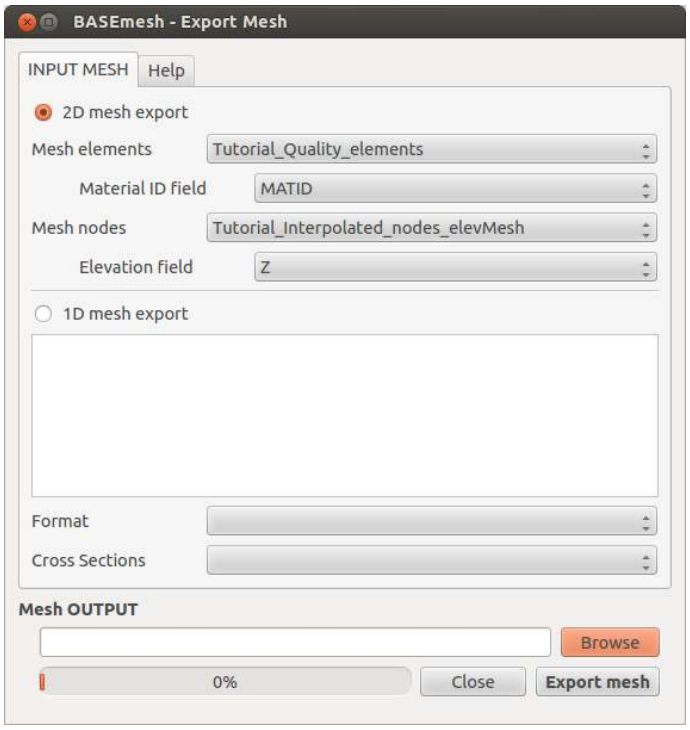


Figure 3.19 *Exporting a mesh from QGIS to file.*

3.3.5.9 Visualization of Simulation Results using QGIS (Post-Processing)

3.3.5.9.1 Visualization with the Crayfish plugin

Fast and easy visualizations of time dependent BASEMENT simulation results can be achieved using the Crayfish plugin developed by Lutra Consulting. The plugin is based on a fast engine which can deal with large and time dependent data sets in an efficient way. The plugin allows for plotting simulations results as adjustable color maps and offers additional visualization options like velocity vectors. Furthermore time dependent 2D results can be exported as animation.

How to do the Post-Processing using the Crayfish plugin is explained in detail in the Post-Processing tutorial.

3.3.5.9.2 Visualization with QGIS operations

The BASEMENT simulation results can also be visualized using QGIS core functions, mainly via two approaches:

1. **Shapefiles:** BASEMENT has the option to write results files in the shape-format with each time step as column in the attribute table. These data files can be directly loaded into QGIS and visualized/colorized. A disadvantage of this visualization method is, however, that there is no smooth color transition between the discrete elements.

2. **Raster data:** Output data of BASEMENT (either as shapefiles or as ASCII-x-y-z data files) can be rasterized in QGIS. Hereby, the pointwise data values are interpolated on a raster. To rasterize the pointwise data values you can use the QGIS

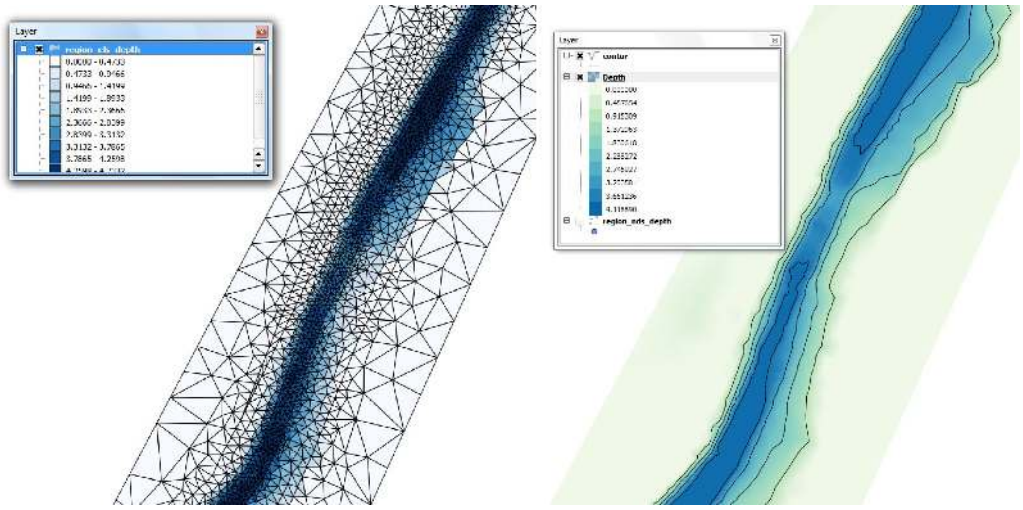


Figure 3.20 *Screenshot of data visualization using shapefiles (left) and raster data (right).*

option *Raster* → *Interpolation*, which offers multiple interpolation algorithms, like TIN or inverse distance weighting. To colorize the raster data, open the properties dialog and select the option singleband pseudocolor and choose your preferred color map. Raster data are generally nice to work with, since GIS-applications have many and powerful tools to cope with raster data in various ways, e.g. to create contour lines or cross section profiles.

It is important to note that output can be generated as 'element - centered' data (in the middle of elements) and as 'node - centered' data (at nodes). Most data values are computed on the elements in BASEMENT. Therefore 'element - centered' data usually equals exactly the computed values, whereas 'node - centered' data is interpolated onto the node coordinates.

Post-processing with Shapefiles or Raster data is usually more time consuming than using the Crayfish plugin, especially if time dependent data shall be visualized. However, it may be a good choice especially if high-quality plots and graphics of a specific simulation timestep are needed and the powerful mapping tools of QGIS shall be used.

4

Model Setup

4.1 General

Since Version 2.0, BASEMENT is not a pure console application anymore. The Graphic User Interface (GUI) enhances the usability, but it is still possible to work with BASEMENT on a console.

It is now possible to define the whole Command File using the GUI. This avoids Syntax Errors within the Command File compared to the versions where the Command File was created manually. Furthermore, the *Command File Editor* validates the current state of the Input File and indicates whether there remain some Errors or Warnings.

Grid generation for 1-D simulations has never been easier. The *1-D Grid File Editor* allows the definition of cross sections and provides a graphical visualisation of the whole river reach and also every single cross section. There are many additional tools to work on cross sections which facilitate the setup of a 1-D geometry.

The intention of this document is to help the user working with the Graphic User Interface. The common basic behaviour is explained in Section 4.2. The Use of the Command File Editor is described in Section 4.3 and the Explanation of the 1-D Grid File Editor can be found in Section 4.4. In Section 4.5, interactive Visualisation and Manipulation tools during the runtime of a simulation are shown.

In some situations it may be desired to run the program without GUI and without user interaction, e.g. to execute several program runs via batch script over night or when running the program on a high-performance machine. Further details concerning the batch execution mode are explained in the Section 1.1.

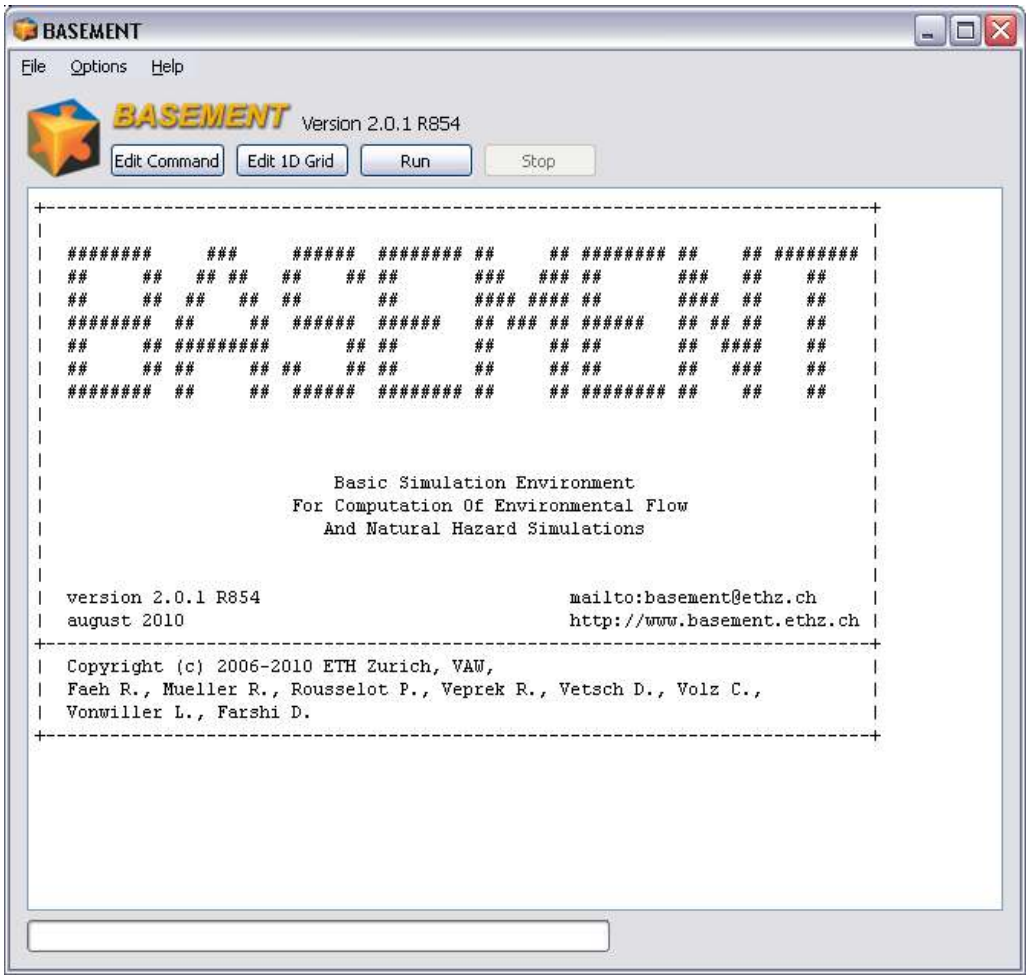


Figure 4.1 *BASEMENT Main Window with menu bar, action buttons, splash screen and a progress bar.*

4.2 The Graphical User Interface (GUI)

4.2.1 The BASEMENT Main Window

Starting BASEMENT will first produce a Graphical User Interface as shown in Figure 4.1. On the top of the screen, a menu bar consisting of the menus *File*, *Options* and *Help* is available. Below the menu bar, near the BASEMENT logo, there are four action buttons named *Edit Command*, *Edit* 1D *Grid*, *Run* and *Stop*. The main window displays a splash screen indicating the current version of the Software. A yet empty progress bar is located below the main window.

The menu File allows for opening either an existing command file (Open Command) or an existing 1 D Grid (Open 1D Grid) within the BASEMENT Editor. Selecting Quit from the File menu will exit the Application.

The menu Options hosts the setting for the Log level. Different Log levels are available:

- Log level 4 will show all possible Log output,
- Log level 3 will suppress debug messages,

- Log level 2 will print warnings and errors only,
- Log level 1 will show error messages only.

The Log output is divided into two tabs, one to show general Log information and debug messages, another one to print warnings and errors. In case of an error the simulations stops and BASEMENT exits. The error messages can then be found in the error-Log file.

The splash screen can be shown anytime again by selecting About BASEMENT from the *Help* menu.

The action buttons have the following behaviour:

- *Edit Command* opens the currently active Command File within the Command File Editor. If no Command File has been opened yet, the Editor will be empty. A description of the Command File Editor is given in Section 4.3.
- *Edit 1D Grid* opens the currently active 1D Grid definition within the 1-D Grid File Editor. If no 1-D Grid has been opened yet, the Editor will be empty. A description of the 1-D Grid File Editor is given in Section 4.4.
- *Run* is only active when a Command File is active in the background. Pressing this button will start a simulation according to the settings in the Command File.
- *Stop* is only active if a simulation is currently being carried out. Pressing this button will stop the current simulation.

4.2.2 General Topics of Editing Files

4.2.2.1 BASEMENT Editor

Both, the Command File Editor and the 1-D Grid File Editor share a common composition. The Window is subdivided into three parts: the Input Structure to the left, the Main View to the right and the Validation Messages at the lower right corner (Figure 4.2).

The Input Structure shows the tree of all Blocks defined and allows the selection of a specific Block. The Main View shows all relevant Information for the currently selected Block like available Sub-Blocks/Tags and the values for all chosen Tags. Graphical visualization of e.g. a Cross-section or a time series can also be seen.

The Validation Messages indicate by their Color whether the Input for the selected Block contains any Errors or Warnings.

4.2.2.2 Edit Blocks and Tags

4.2.2.2.1 Add Blocks

Select a Block from the Input Structure. Use the drop down menu near the *Add Block* button in the Main View to see all available Blocks. Select one of them and press the *Add Block* button to add a new Block.

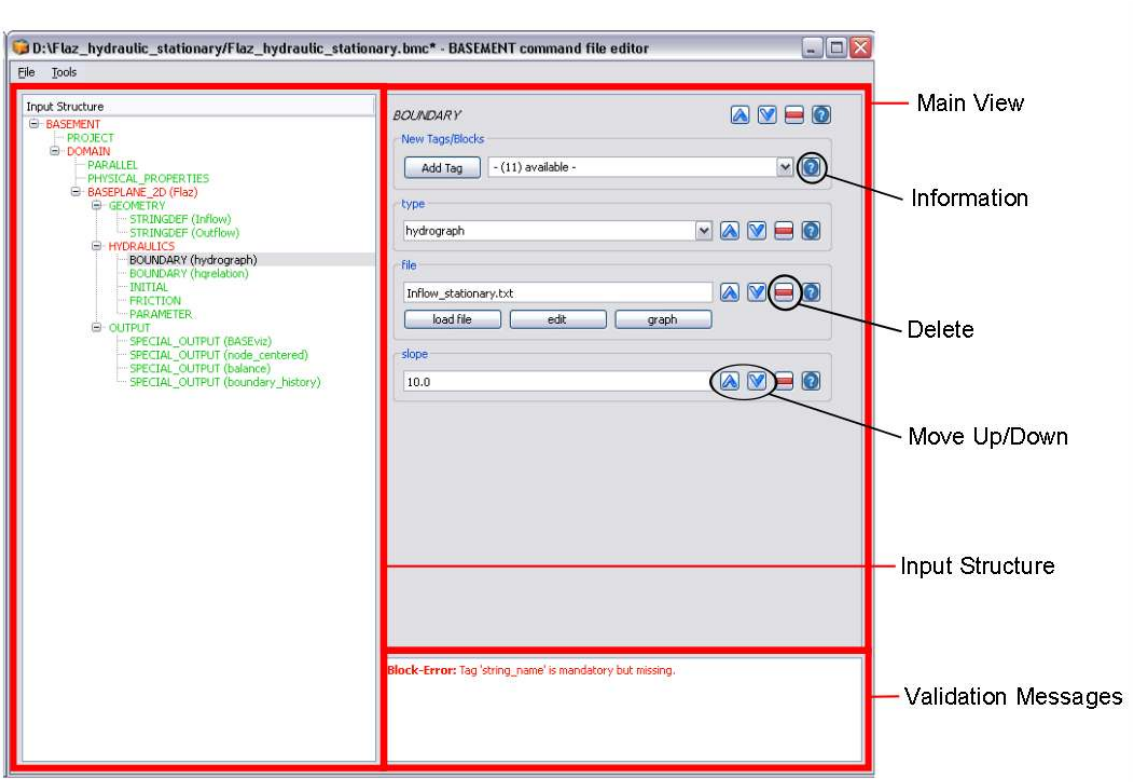


Figure 4.2 *Main Structure the BASEMENT Command File Editor.*

4.2.2.2.2 Define Tags

Select a Block from the Input Structure. Use the drop down menu near the *Add Tag* button in the main view to see all available Tags. Select one of them and press the *Add Tag* button to add a new tag. The Tag will appear in the main view and needs to be given a precise value. Depending on the Tag, a value can be chosen from a drop down menu or has to be defined by the user.

4.2.2.2.3 Remove Blocks or Tags

To remove a Block or a Tag, use the Delete button as indicated in Figure 4.2. For a block, this is located at the top of the main view near the Blocks name. For a Tag, the button is to the right of the precise value of a Tag. There will be a confirmation message whether a Block or Tag really shall be removed.

4.2.2.3 Automatic Validation

Validation Messages indicate immediately whether the current choice of Blocks and Tags is valid (Figure 4.2). In case of errors, the Block and its parent Blocks in the Input Structure are red. In case of warnings, the Block and its parent Blocks in the Input Structure are brown. Blocks without warnings or errors are green.

Selecting a red or brown Block in the Input Structure will show all errors and warnings as Validation Messages.

An Input without red messages has no errors and can be used for simulation.

4.2.2.3.1 Mandatory Blocks and Tags

Certain Blocks and Tags may be mandatory to define. If a mandatory Block or Tag is missing, an error message like *Block Error: Tag "XY" is mandatory but missing* is displayed until the Block or Tag is defined.

4.2.2.3.2 Default Values

Some Tags have default values. They are usually not mandatory to define. Defining a Tag with an associated default value will show the specific Tag value in the Input Field for the Tag where it can also be changed to a user defined value.

4.2.2.4 Use the Built-in Help Function

Every Block and Tag is documented within the Reference Manual R IV. This same information is also available within the BASEMENT Editor. Clicking on the Information button with a question mark (Figure 4.2) will produce a pop up window with a description of the usage and some examples. The Information button for a block is located at the top of the main view near the Blocks name. For a Tag, the button is to the right of the precise value of a Tag. The documentation for Tags also includes information about whether it is mandatory, default values, value range, etc.

4.3 Edit Command File

4.3.1 The "BASEMENT Command File Editor"

From the BASEMENT main window (Figure 4.1) the user gets via the *Edit Command* button to the *Command File Editor* which is shown in Figure 4.3. The Input Structure (as it is built up with Blocks) is on the left-hand side. The Main View on the right-hand side gives more details of the selected Block with the corresponding Tags and the possibility to add Tags and/or Blocks (see Section 4.2).

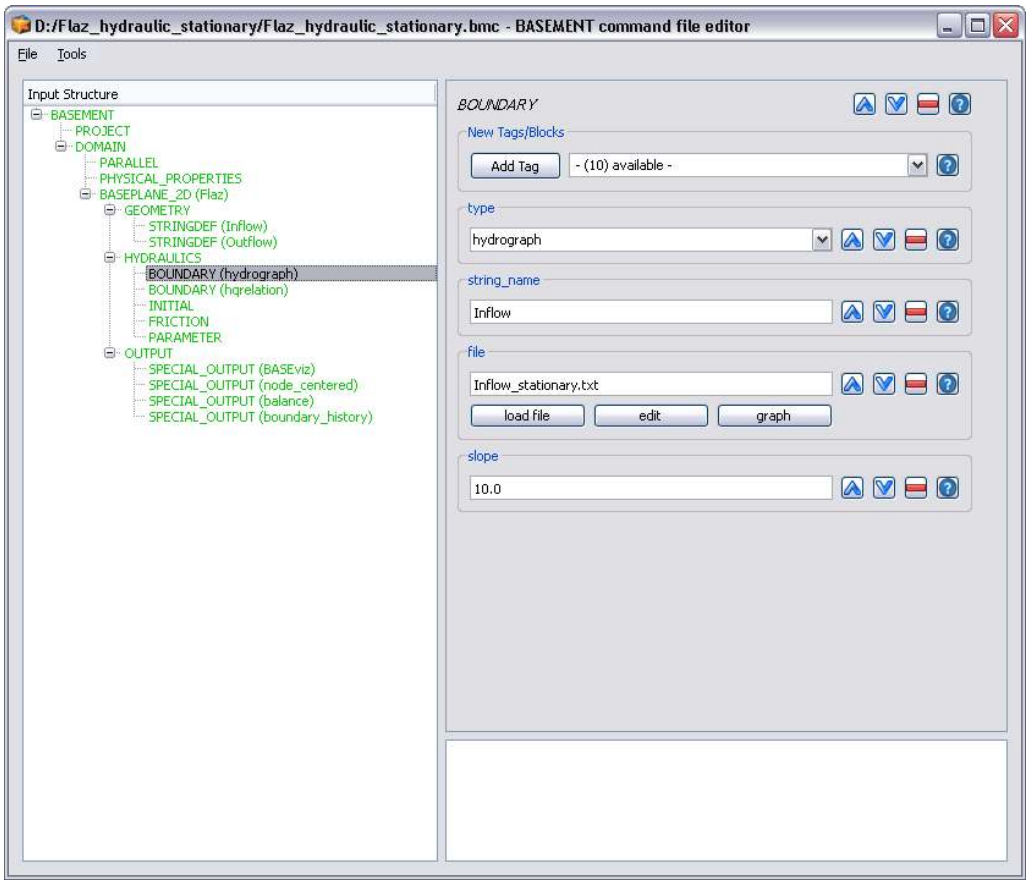


Figure 4.3 *BASEMENT Command File Editor*

4.3.2 Create New or Edit Existing Command File

A new Command File is created form the menu bar in the *Command File Editor*. Choose *File* on the menu bar and select *New*. The whole command file consisting of Blocks and Tags can be built up from scratch. For further details on how to add Blocks and define Tags see Section 4.2.2.2 in this manual.

If you want to open an existing command file you can do this from the *Command File Editor* or directly from the BASEMENT *Main Window*. From the *Command File Editor* choose *File* on the menu bar and then select *Open*. Then you browse and choose your file as usually. Don't forget to save your changes made in the command file before you run a simulation.

4.3.3 Tools

4.3.3.1 Edit Raw

Another way to edit a command file is to edit basically the command file in a raw text mode. For this purpose choose *Tools* on the menu bar and select *Edit Raw*. The Editor window will pop up as shown in Figure 4.4. Long-standing users of BASEMENT will recognize the input structure of the former command file. In the lower part of the window the Input is validated and possible parse errors are indicated. For the sake of completeness it is mentioned here that the command file can still be built up and edited with a simple text editor.

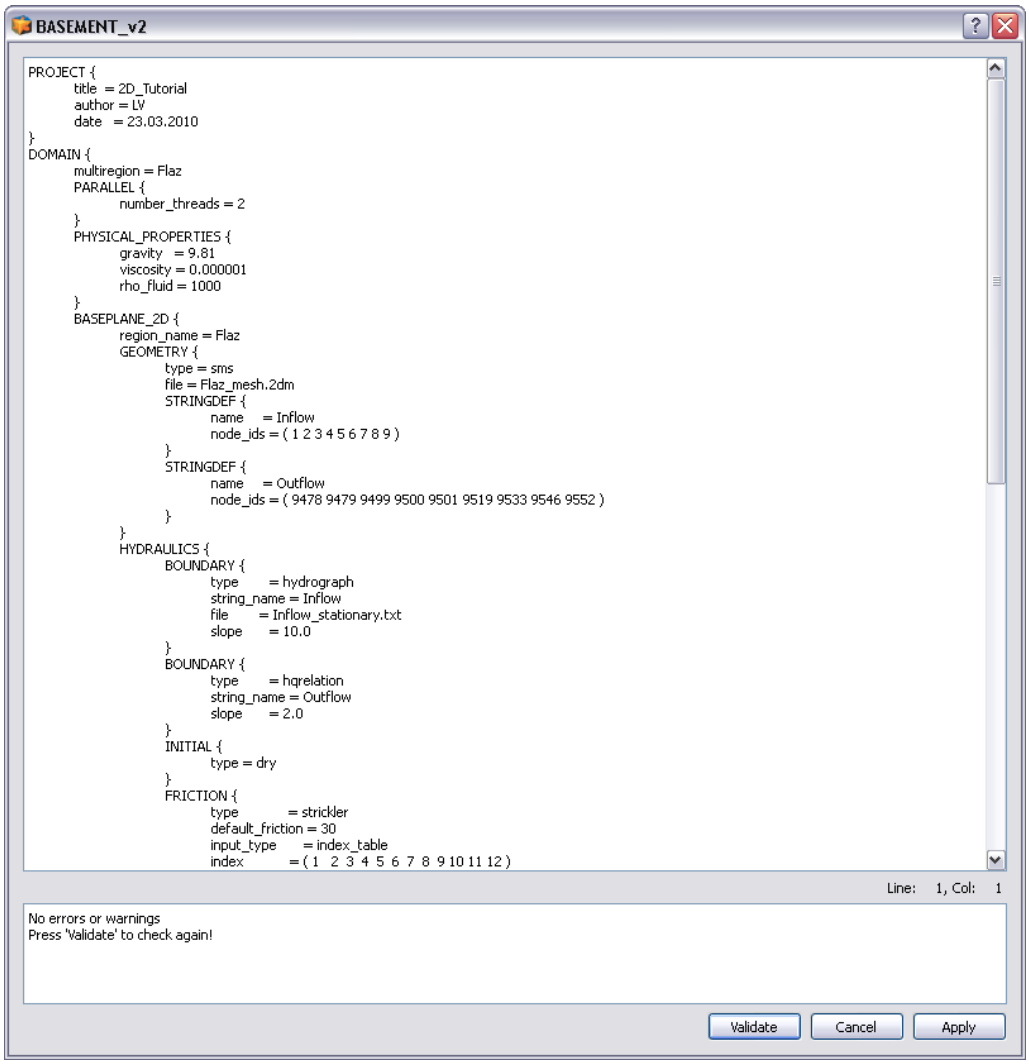


Figure 4.4 *Command File Editor: Raw Edit Window.*

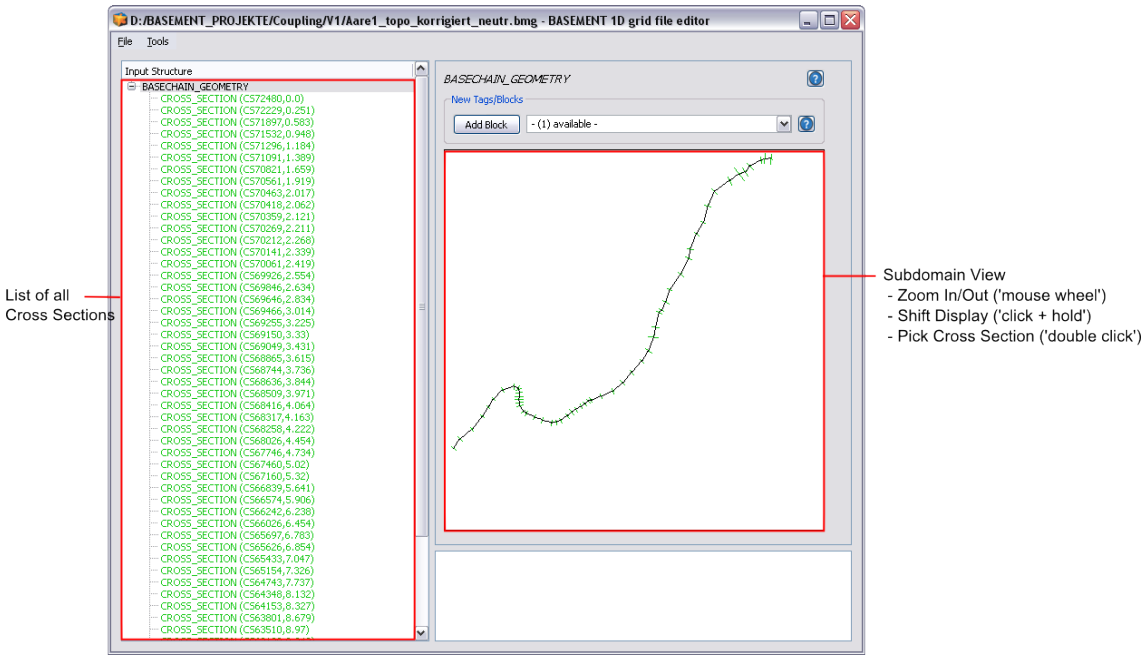


Figure 4.5 *Grid File Editor – Subdomain View*

4.3.3.2 Expand All

The command Expand All (Ctrl+E) opens the whole input structure on the left-hand-side of the Command File Editor. It provides a faster access to a single block or tag.

4.4 Edit 1-D Grid

4.4.1 The "BASEMENT 1-D Grid File Editor"

When the Grid File Editor is opened a new window pops up (see Figure 4.5). In the left part of the window a list of all cross sections is shown where the cross sections can be seen and selected. In the right part of the window a visualization of the whole subdomain with all cross sections is drawn and new cross sections can be created with the *Add Block* option. The subdomain view allows zooming and shifting of the display and the selection of a specific cross section by double clicking. If a cross section is selected then the view changes to the cross section view.

The real (usually curved) shape of the stream can only be illustrated if all cross sections are geo-referenced and if all corresponding data is set (the *orientation_angle* and the *left_point_global_coordinates* must be set for each cross section). If these data are not given than the cross sections are drawn along a straight line.

4.4.2 Create New or Edit Existing Geometry File

If a specific cross section is selected or a new cross section is created, than a profile view of the selected cross section is shown (see Figure 4.6). With this visualization of the profile one can easily check for input errors in the geometrical definition of the cross section profile.

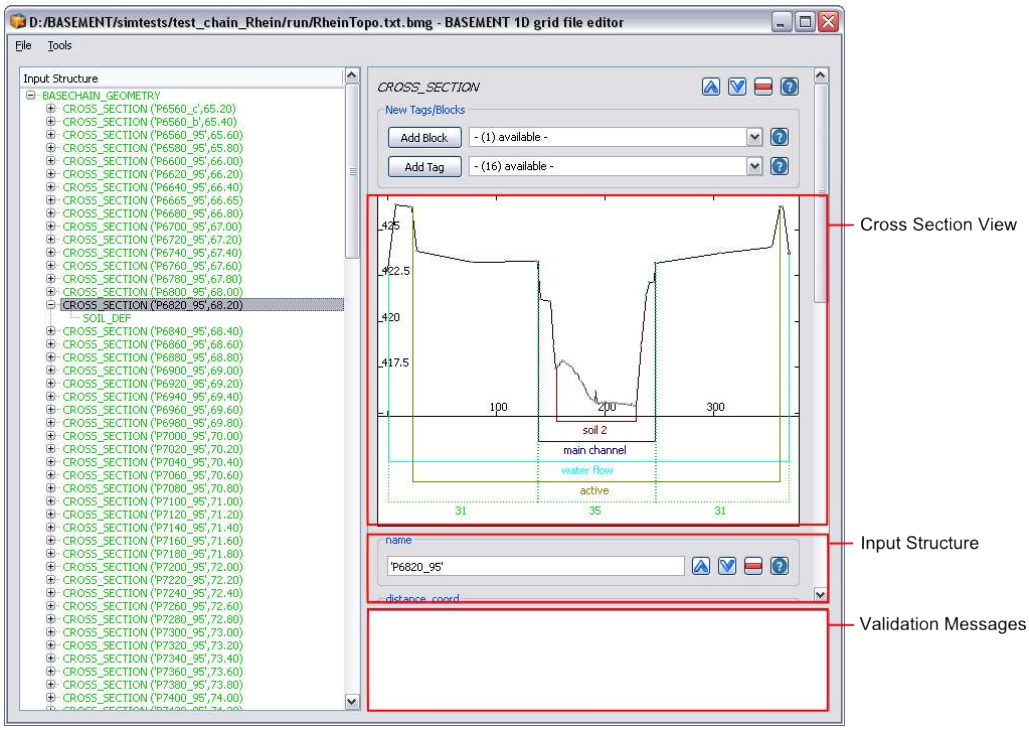


Figure 4.6 *Grid File Editor – Cross Section View*

Furthermore, the most important cross section parameters are indicated visually with different colours, like e.g. the definition of the main channel, the range of the soils, the friction parameters, the cross section fixpoints, etc. Again, one can visually check if these parameters are set up correctly and thus easily detect type errors. New input tags can be added and the validation message box shows warnings or errors if some problematic inputs have been made.

For details on how to set up a new cross section and for information about the various cross section parameters see the 1-D tutorial, hydrodynamics and sediment transport at the river Thur.

4.4.3 Tools

BASEMENT supports several tools which support the user in creating and modifying the 1D grid file. In the following sections some information about the usage and the methods of these tools are given. Please be aware that some of the offered tools are still in a beta-status.

4.4.3.1 Edit Raw

Another way to edit a grid file is to edit the grid file in raw text mode. For this purpose choose *Tools* on the menu bar and select *Edit Raw*. The Editor window will pop up as shown in Figure 4.7 and Figure 4.4. In the lower part of the window the Input is validated and possible parse errors are indicated. For the sake of completeness it is mentioned here that the grid file can still be built up and edited with a simple text editor.

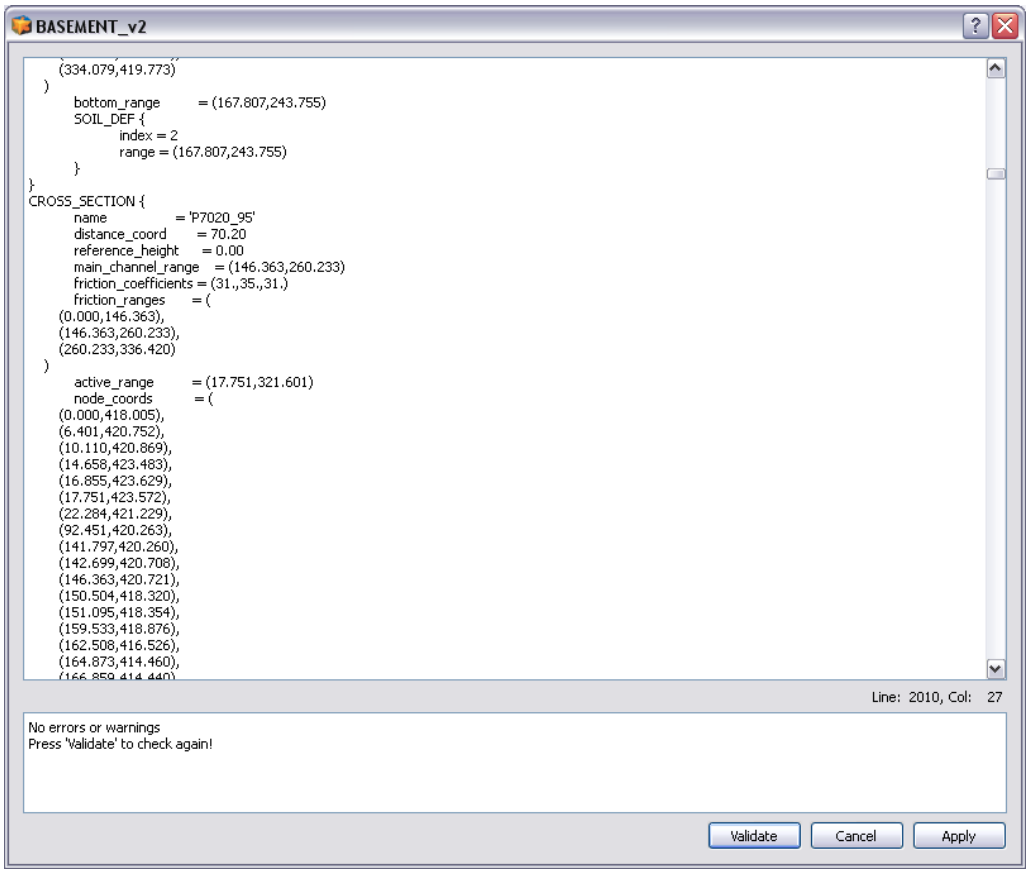


Figure 4.7 *Grid File Editor: Raw Edit Window*

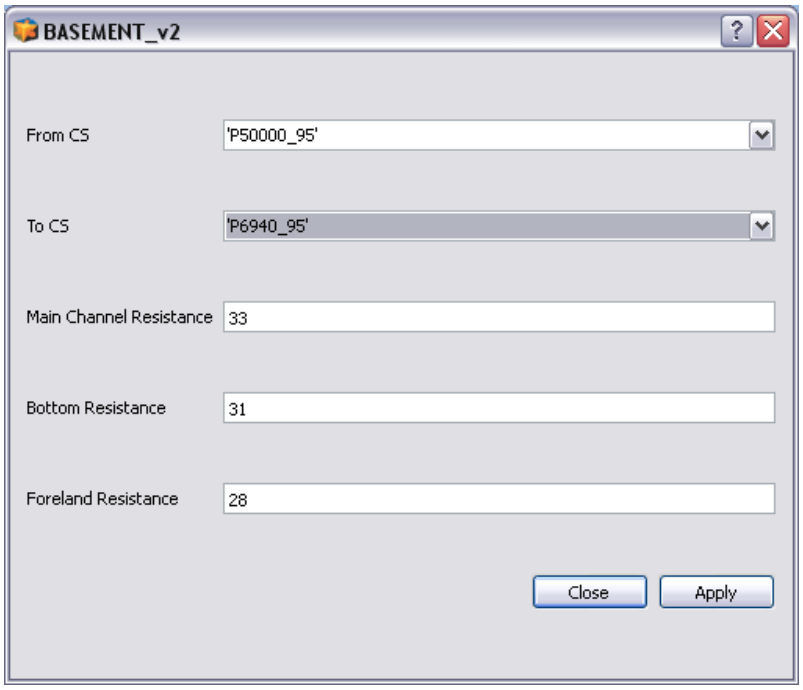


Figure 4.8 *Friction assignment*

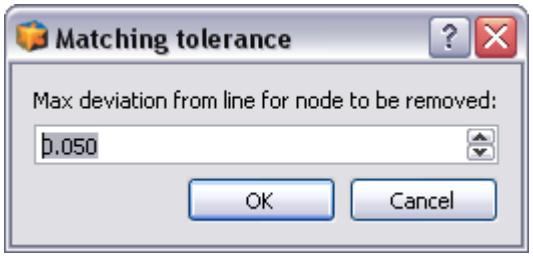


Figure 4.9 *Node removal dialog for setting up the tolerance*

4.4.3.2 Friction

With this tool the friction value of the main channel, the forelands and the bottom can be assigned to a range of consecutive cross sections. The names of the first and the last cross section of the range have to be given.

4.4.3.3 Remove Nodes

Cross sections often consist of a large number of nodes and slices which consumes significant computational performance. If multiple nodes lie on a straight line within the cross section there is redundant information present which can be removed without reducing the accuracy of the computations. Therefore identifying and removing these redundant nodes is a frequent and recommended task before running the simulations. BASEMENT offers a tool which performs this task automatically without need for costly manual operations.

To access this tool open the *Tools* menu and click on the *Remove nodes option*. The following dialog opens:

In this dialog you can define the maximum tolerance by which a node may deviate from the straight line of its two neighbours. If the node lies within this tolerance the node removal is

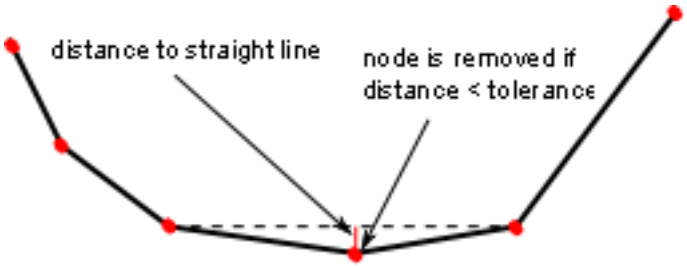


Figure 4.10 *Schematic sketch of node removal*

applied. If you set the tolerance to small values only nodes are removed which are almost exactly situated on the line. If you increase the tolerance more nodes will be removed but some more information about the cross section profile may get lost. Usually one should try different tolerances until the best compromise between computational performance and accuracy is found. Also, the algorithm can be applied multiple times.

The applied algorithm loops all nodes of the cross sections and checks if a node is situated on a straight line between its two neighboured nodes (considering the given tolerance). If this is the case the node is removed from the profile (see Figure 4.10). Nodes which are used as fixpoint or which are explicitly referenced by a slice_index range are excluded from the algorithm and cannot be removed automatically.

Guess Active Range

This tool provides a definition of the active range where it is not yet defined. For this purpose the lowest point of the cross section is searched and then the highest points to the left and the right of it (usually the dam crests) are set as limits of the active range. The definition of the active range can always be changed manually by the user in the interface. As an example on how the active range is set see Figure 4.11.

4.4.3.4 Guess Fixpoints

This tool provides the definition of some fix points, which are needed for a correct interpolation of new cross sections between existing cross sections. So this should be done before an interpolation is executed. The fix points are displayed in red. The points which are automatically set as fix points are:

- The limits of soils
- The limits of the main channel
- The limits of the active range
- The midpoint of the main channel.

It is recommended to check the points visually and add other important points, especially on the break lines. For the interpolation all involved cross sections must have the same number of fix points.

4.4.3.5 Interpolation

First of all, before an interpolation of 1-D cross sections can be performed some information is needed about the spatial alignment of all cross sections in the x-y plane.

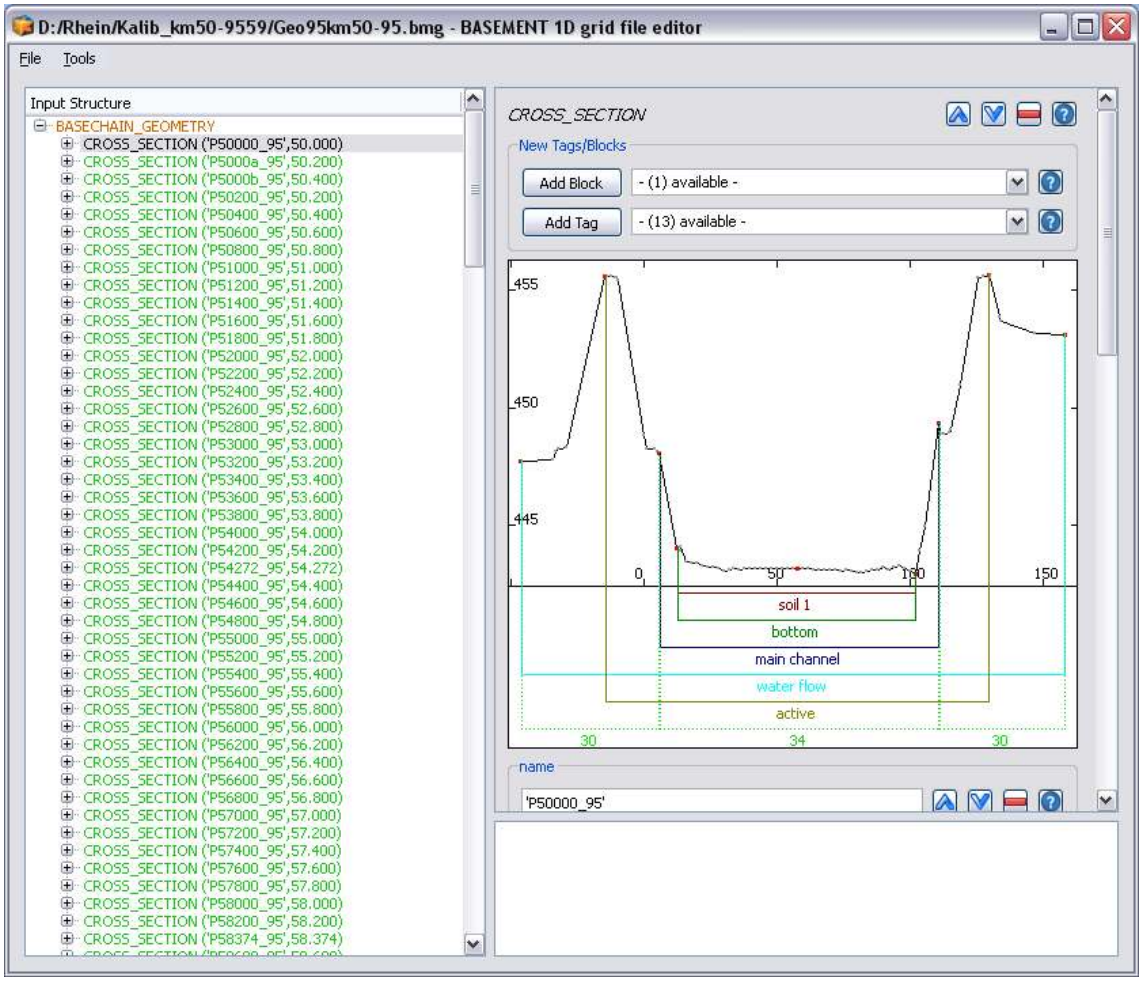


Figure 4.11 *In red the guessed fix points for cross section interpolation*

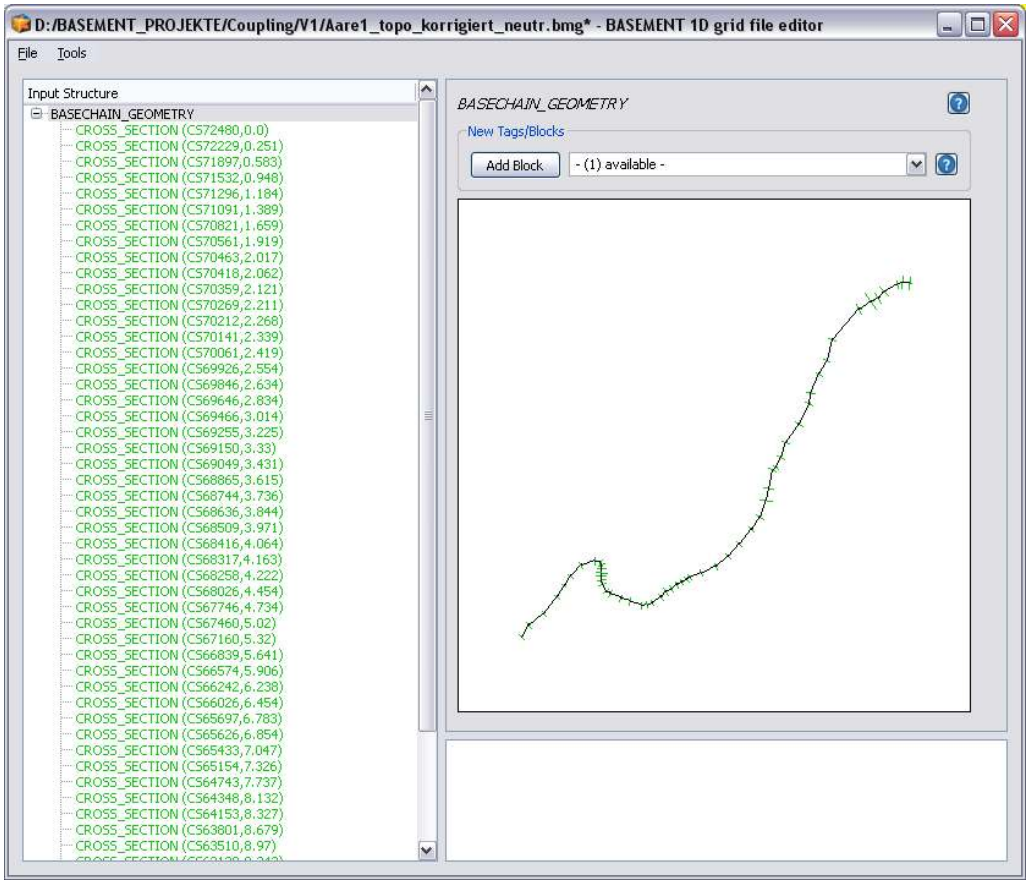


Figure 4.12 *Curved alignment of the cross sections.*

There are two main tags which determine the spatial orientation of the cross section which are crucial for the interpolation algorithm. The *orientation_angle* provides the information which is needed for the orientation of the cross sections. This is the angle between the normal vector of the cross section and the vector in x-direction (1,0). Consequently, in a fully straight channel in x-direction all cross sections would have the angle 0°. If the orientation angle is not given it is set to this value. The other essential tag is the *left_point_global_coordinates*. This parameter sets the global (real world) x,y,z-coordinates of the outer left point of the cross section. This parameter in combination with the orientation angle delivers all needed information about the spatial configuration of the cross section. If this coordinates are not given the value of *distance_coord* is used for x and the elevation of the first point on the left for z. y is set to negative distance of the first point on the left from the middle of the cross section. If both parameters are set for all cross sections, one can see the curved alignment of the stream in the right-hand visualization (see Figure 4.12).

The algorithm of the cross section interpolation is briefly sketched in the following. To grasp the meaning of the different parameters it is helpful to understand the basics of this interpolation algorithm.

This interpolation algorithm bases on the creation of spline curves (a spline is a special polynomial function which is often used for smooth interpolations between given points). For each fixpoint of the cross sections such a spline curve is determined which connects all the corresponding fixpoints with each other in a smooth way. Therefore every cross section must have the same number of fix points. Furthermore, the spline curves have the special

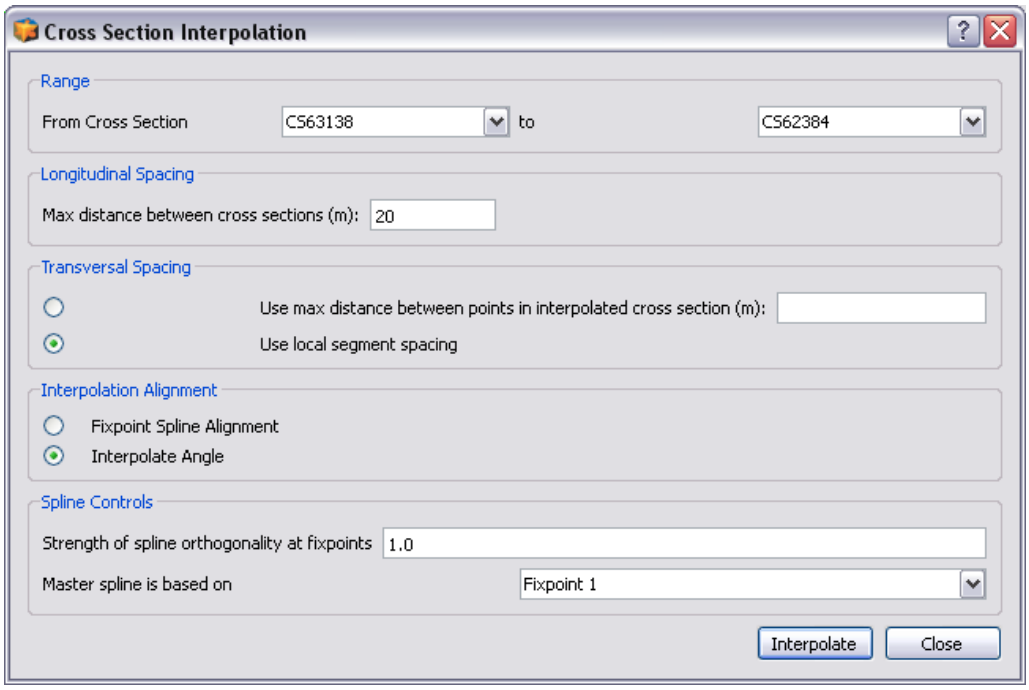


Figure 4.13 *Setup dialog for the cross section interpolation*

property that they are aligned orthogonal to each cross section profile.

After the spline curves have been calculated, the positions of the new interpolated cross sections are determined in given intervals along the spline curves. As soon as these positions are known, the cross sections are created orthogonal to the tangent direction of the master spline. To determine the fixpoints of the new cross sections the intersections of this orthogonal cross section line with all spline curves (of the other fixpoints) are calculated. Finally the new cross section points are determined in between the fixpoints in a given transversal distance interval. The elevations of the cross section points are finally determined using a weighting procedure between the elevations of the left and the right cross section.

In Figure 4.13 the setup dialog for the interpolation is shown. The different parameters are explained briefly in the following. By clicking on *Interpolate* the interpolation of the cross sections finally starts if all data is available.

First of all the *Range* of the interpolation must be defined. This is done by specifying two subsequent cross sections which are chosen from a list of all existing cross sections in the drop down menus.

Another important parameter is the *Longitudinal spacing* which determines the resolution of the interpolated grid. Enter the maximum distance between two interpolated cross sections in [m]. If you choose a small value than many cross sections in small distances will be generated, if you choose a large value only few cross sections in large distances will be generated. The optimal choice depends on the type of simulation.

Furthermore, also the *Transversal spacing* can be specified. It determines the spacing of the points in the newly generated cross section profile. There are two possible choices here two determine the transversal spacing. You can either explicitly specify the maximum distance in [m] using the first option. Alternatively, by using the second option, the distance is chosen automatically from the left and right cross sections by the interpolation algorithm

(local segment spacing).

The alignment of the new cross sections in the x-y plane can be determined with two different methods in the *Interpolation Alignment section*. The *Fixpoint Spline Alignment* means that the cross sections are always oriented orthogonal to the master spine's tangent direction. On the other hand, using the option *Interpolate Angle* the orientation of the new cross section is determined by interpolation of the orientation angles of the left and the right cross sections. This latter option is recommended in strongly curved streams in order to prevent overlapping cross sections.

Finally in the *Spline controls* section some parameters of the spline calculations can be adapted to special needs. The *Strength of spline orthogonality* parameter determines if the spline always must be completely orthogonal to the cross sections or not. In strongly curved streams some relaxation from strict orthogonality (different from 1.0) may lead to nicer shaped spline curves. Some variations and iterative testing with this parameter may improve the interpolation result in such situations. And finally also the fixpoint which determines the master spline can be chosen. The master spline thereby is the spline which determines the orientation of the interpolated cross sections.

Please note: In order to generate a 2D mesh from a given 1D mesh, this interpolation option can be very helpful in combination with the *Export DTM* option.

4.4.3.6 Export DTM for BASEplane

This tool enables the user to convert a 1-D BASECHAIN_GEOMETRY (Figure 4.14) into a digital terrain model (DTM) for further processing in SMS and BASEplane. The main application for this tool is to be found in combination with the Interpolation tool (see chapter 0): In a first step the cross sections are interpolated with the Interpolation tool in order to get a smooth river topography. In a next step the DTM is exported with the *Export DTM for BASEplane* tool (on the menu bar choose *Tools*). The generated DTM can be imported in SMS. Although the file is of .2dm type it can be easily converted into scatter points (DTM) in SMS. Then it can be used for the interpolation of the elevation information on any computational mesh.

Basically a computational mesh can be obtained directly from the *Export DTM for BASEplane* tool, if the interpolated cross sections are chosen in a close and optimal distance to each other. Nevertheless it is suggested to generate the mesh properly in SMS and to consider the generated DTM just as a terrain model from which to get the elevation information.

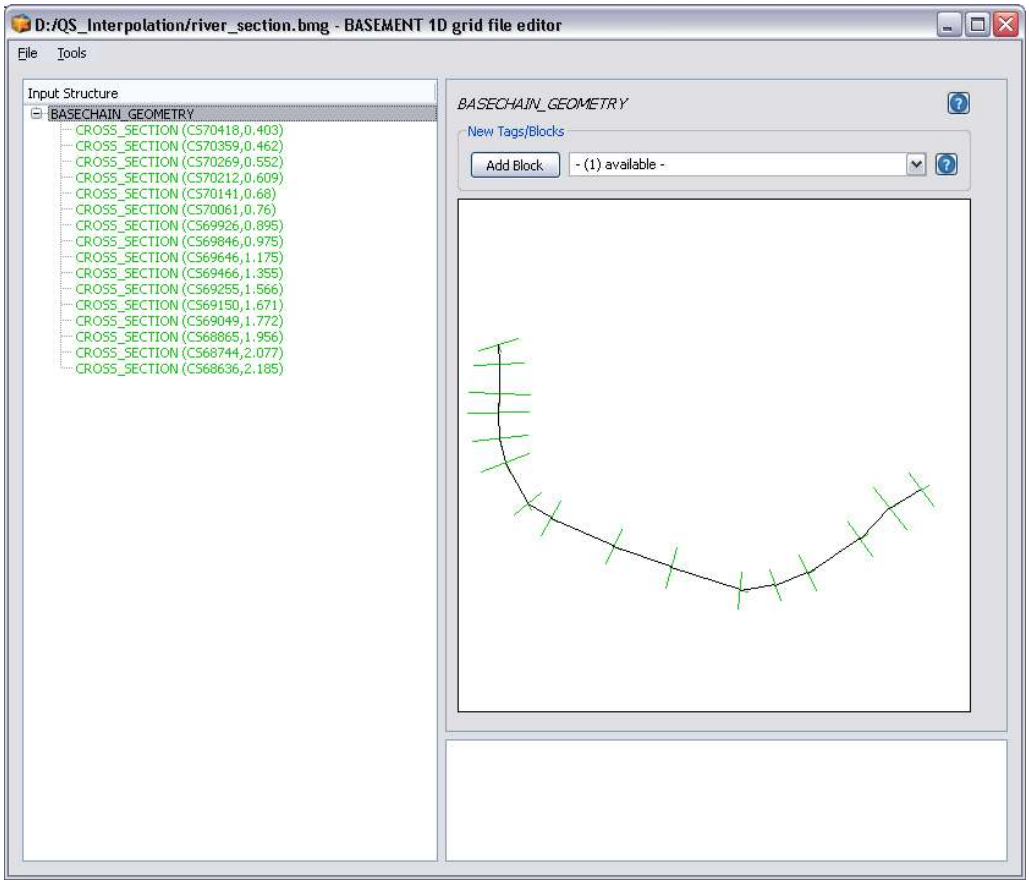


Figure 4.14 *GUI of the BASEMENT 1D grid file editor. The 1-D BASECHAIN_GEOMETRY can be exported with Export DTM for BASEplane under the menu bar Tools.*

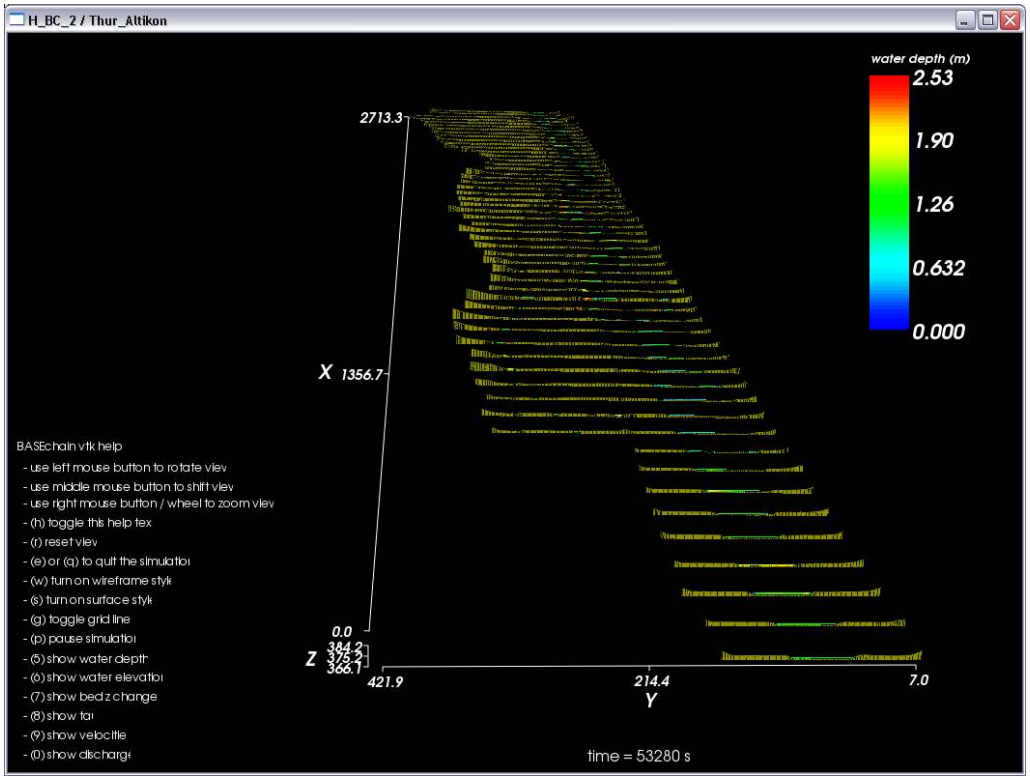


Figure 4.15 *Visualization of BASEchain with BASEviz*

4.5 Built-In GUI Tools

In this section some built-in BUI tools are explained and information about the usage is provided. Built-in GUI Tools will pop up if a certain tag is activated by the user.

4.5.1 Interactive Visualization during run time using BASEviz

BASEviz is a small and lightweight visualization tool which can be used to visualize simulation results during run time. To activate the visualization tool, a *SPECIAL_OUTPUT* block of 'BASEviz' type must be created within the parent OUTPUT block. Then the BASEviz window will appear automatically by starting the simulation. The output can be visualized interactively using the mouse and keyboard keys according to the legend shown in the BASEviz window (see Figure 4.15 and Figure 4.16). The view can be changed and the displayed variables can be selected. This visualization tool allows to easily check for a correct simulation setup and to stop a simulation run if some evident problems arise. Furthermore, it is possible to dump the rendered images from the visualization window in a JPEG image for a given time interval.

- BASEviz for 1-D simulations with BASEchain:

  All 1-D cross sections with its multiple slices are plotted one after the other along the x-axis. The water elevation is plotted within each cross section slice according to its present value.

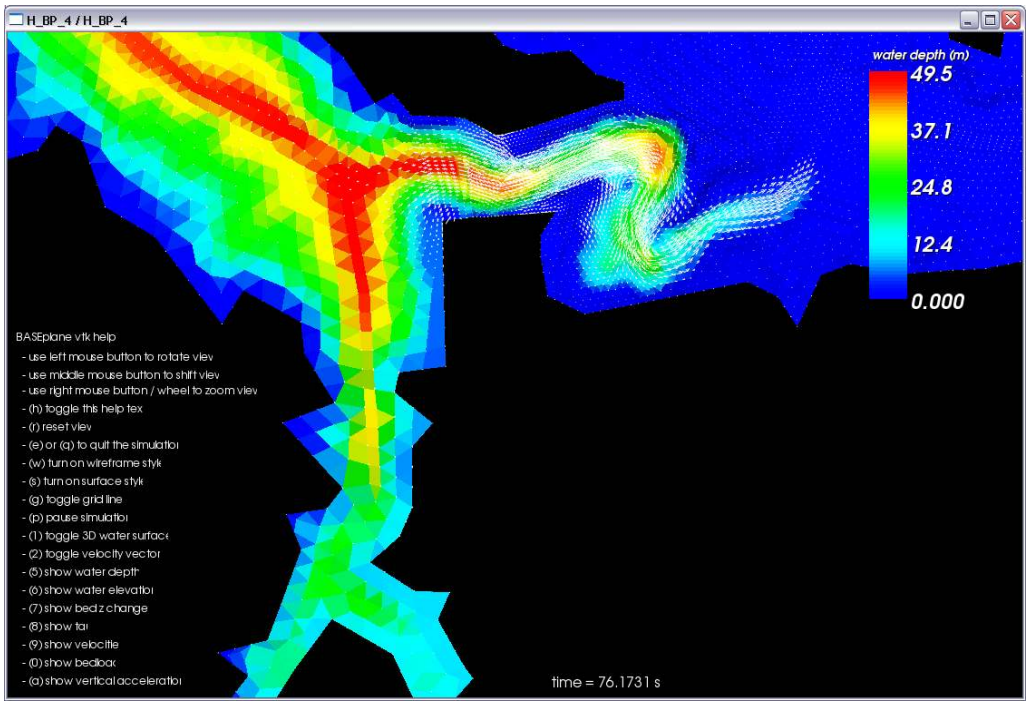


Figure 4.16 *Visualization of BASEplain with BASEviz*

- BASEviz for 2-D simulations with BASEplane:

  The unstructured 2-D mesh is plotted in combination with a contour plot of a chosen output flow variable. Optionally, velocity vectors can be added to the data visualization.

BASEviz is based on the visualization libraries of the Visualization ToolKit (VTK,http://www.vtk.org) which makes use of OpenGL for rendering.

4.5.2 Manual Controller Interface (HID)

In order to create a controller, a new CONTROLLER block is generated in the DOMAIN block. The HID controller provides an interface for the manual operation (Figure 4.17). The control window will pop up automatically after starting the simulation with the start button. In the CONTROLLER block several manipulated variables and controlled variables can be defined. The manipulated variables will appear on the left hand side of the controller interface, whereas the monitored variables will show up on the right hand side (Figure 4.17). In this example two manipulated variables (height of a weir and height of a gate) and two monitored variables (discharge over the weir and through the gate) are selected. With the cursor the slide can be moved within the predefined range of the manipulated variable. Additionally the target value can be entered directly into the white text box (Target). As the simulation proceeds the impact on the monitored variable is visualized on the chart on the right hand side. Additionally the output and the impact of your control measures can be visualised with BASEviz (Section 4.5.1).

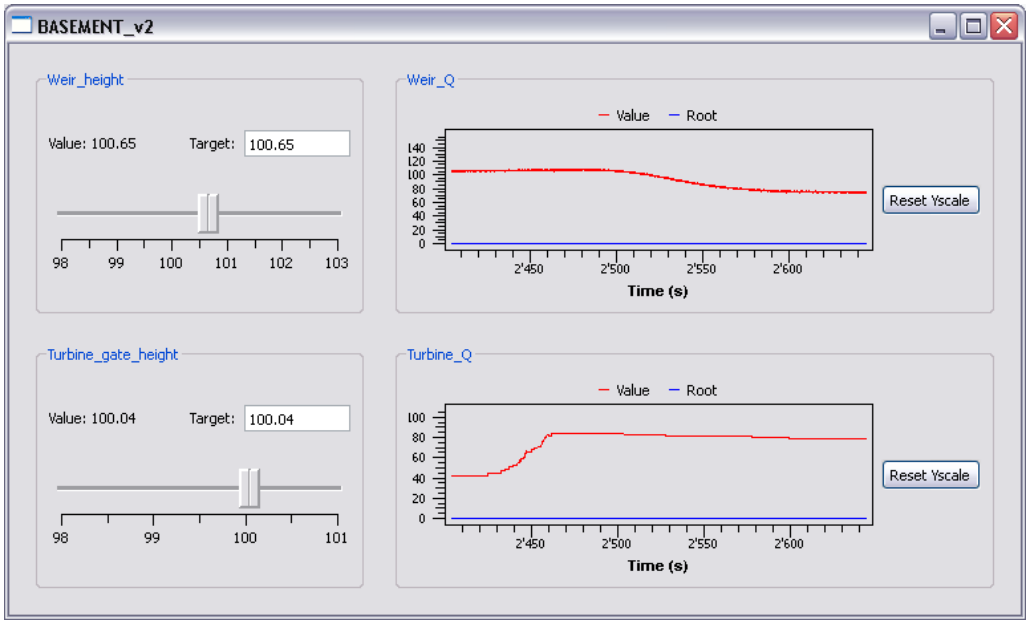


Figure 4.17 *Interface for the manual control and monitoring of the selected variables.*

5

Advanced features

5.1 Parallelization

5.1.1 Overview

5.1.1.1 Needs for parallel computing

Parallel computing is currently getting increasingly important for numeric scientific and engineering applications and this trend will become even more significant in the near future. Where parallel computers in the past were mainly limited affordable and manageable by large research institutions, today nearly all new available computers provide capacities for parallel computing. The famous law of Moore, which predicts an increase in computer power about a factor 2 every two years, is expected to be valid also in the near future due to parallel computing (Manferdelli et al., 2008).

In parallel computing the increase of execution speed is mainly achieved by sharing the overall work load between several processors instead of further accelerating single processors by an increase of tact rates. These changes in computer hardware architectures raise the needs for new parallel programming concepts. Programs originally developed and optimized for execution on a single processor cannot automatically benefit from the availability of multiple processors. Specific software techniques and algorithms must be developed and applied to exploit the additional performance provided by parallel hardware. The software BASEMENT therefore needed to be adapted and optimized for the use of parallel computers.

Some typical hydrodynamic simulation scenarios with high demands on performance are

- multidimensional flow simulations with large computational domains consisting of a very large number of elements, as well as simulations of large river networks,
- simulations of river flow over long time periods (especially with regard to morphological changes) and
- real-time simulations of river flow and flood wave propagation.

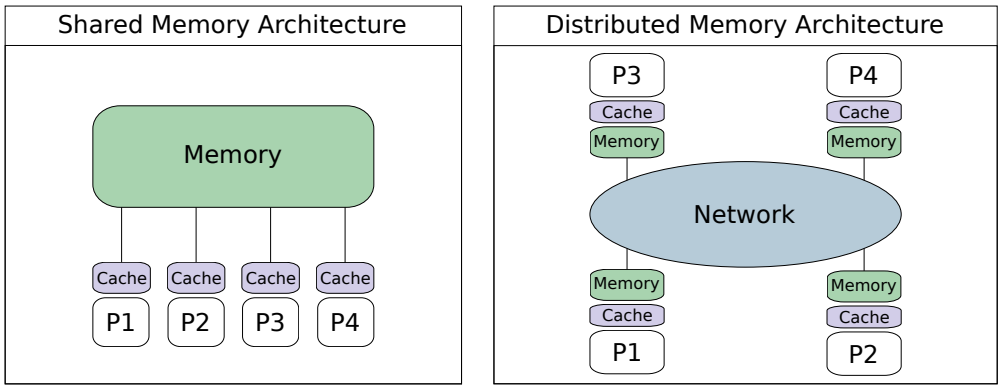


Figure 5.1 *Shared memory architectures (left) and distributed memory architectures (right) with four cores*

5.1.1.2 Parallel computer architectures

Types of parallel computing can be differentiated in many aspects, concerning hardware, software concepts and various other criteria. Frequently, parallel computers are classified in two groups, which differ in the way memory is organized between the multiple processors.

In so called "distributed memory" architectures each core possesses its own memory unit, which cannot be accessed by any of the other cores. Distributed memory systems usually consist of large computer clusters which are connected via a network (right side of Figure 5.1). Such a parallelization concept enables the use of very large numbers of processors and is employed in high performance computing (HPC). But software programming, maintenance and debugging for distributed memory systems generally is very time-consuming and costly. Complex message passing interfaces via network are necessary for data exchange and synchronizations between the processors. Out of these reasons distributed memory systems, or combinations of distributed and shared memory systems, so called hybrid systems, are not further discussed here.

On the other hand, "shared memory" architectures have only one single memory unit which is shared by multiple cores (Figure 5.1, left side). The parallelization of the software BASEMENT is implemented for such shared memory architectures. This concept enables fast and easy communication between all cores through global access to memory. Furthermore, making benefit of parallel computing often is significantly easier and less time-consuming, especially for already existing sequential applications.

Such parallel computers with shared memory, often termed multi-core machines, are highly available today at low costs. The general trend in processor development is from multi core to many core systems with up to even more than hundred cores in cc-NUMA systems. As a consequence, a major drawback of shared memory parallelization, namely not being portable to distributed memory systems, seems acceptable for an application like BASEMENT which is not focused on high performance computing.

5.1.2 Parallelization issues on shared memory systems

5.1.2.1 Levels of parallelization

Parallelization in shared memory systems is based on multi-threading concepts. Threads are a way for a program to divide its work load into several independently running parts

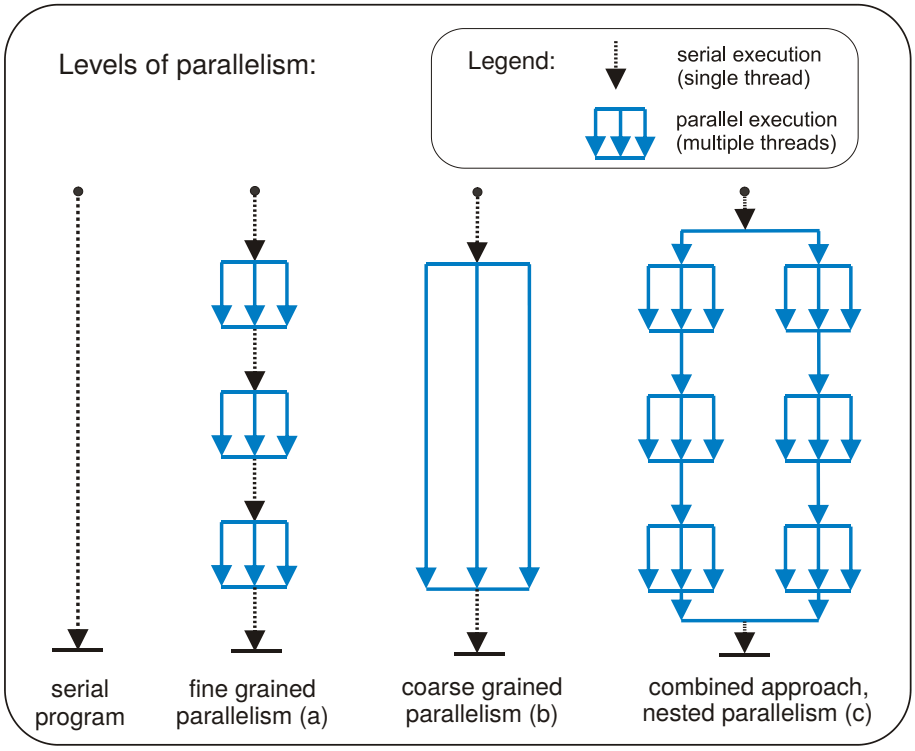


Figure 5.2 *Illustration of different levels of parallelism*

which are finally mapped on the available cores. Threads can be started ("forked") and terminated ("joined") by an application in order to exploit parallelism provided by multiple cores.

The distribution of work load on a series of threads can be made on a small scale loop level ("fine grained parallelism") or on a larger scale procedural or module level ("coarse grained parallelism"). See Figure 5.2 for illustration of these different concepts. Parallelizing on each of these levels has certain advantages and disadvantages. At the moment BASEMENT mainly implements a fine grained parallelism concept. In principle, both concepts can also be combined (nested parallelism).

Numerical applications like BASEMENT spend most of their execution time in rather small parts of the code. These parts are the main calculation loops which iterate over all elements and edges of the computational grid. Fine grained or loop level parallelization aims to exploit parallelism by parallelizing these time consuming loops. If the serial running program arrives at such a loop, its iterations are divided among a certain number of threads and are executed in parallel. Afterwards, a synchronization of the threads is performed so that the program can continue its serial execution (see Figure 5.2). This parallelization approach usually requires no crucial changes to the source code and is especially suitable for parallelizing already existing serial programs. All time consuming loops can be parallelized step by step, thereby incrementally increasing the parallel performance of the program. Due to the small changes in the source code the robustness of a tested serial program is maintained.

Whereas most loops can easily be parallelized, some loops contain flow dependences where the calculation results depend on an ordered succession of all iterations, which is generally not maintained during parallel execution. Such flow dependences can often be resolved by reorganising some parts of the loop. Details on locating and removing flow dependences in

loops can be found in literature (e.g. Chandra et al. (2001)).

5.1.2.2 Factors influencing parallel performance

Parallel performance is often measured as speedup S(n), which is defined as the ratio between "serial execution time" and "parallel execution time" and states how much faster the parallel application runs compared to the serial one.

The theoretical maximum achievable speedup is limited to the number of cores n. The actual speedup is determined by several factors whose influences largely depend on size and type of the simulation. Furthermore the scalability indicates wether the speedup increases linearly with increasing numbers of processors or at a slower rate. The key factors influencing the speedup and according parallel programming aspects are briefly discussed in the following section.

- **Parallel overhead**

  Parallel overhead is created at many locations and times in the program. Threads must be forked and joined, loop iterations must be divided among threads and threads must be synchronized. In the worst case, the parallel overhead can even exceed the performance gains from parallel execution. This is especially a problem in case of small loops with small workload, e.g. in case of initialisation loops. Such loops are not suited for parallelization and should better be executed in serial or, if possible, should be integrated into larger loops.

  For simulations with loops of rather small workload but very high number of time steps, it is important to prevent parallel overhead from frequently joining and forking threads. This was achieved in Basement by integrating the whole time loop into a single parallel region.

  The significance of parallel overhead in a parallel application depends largely on the computational costs of the simulation domain. Simulations with high computational costs, e.g. simulations with large numbers of elements, are less negatively affected.

- **Coverage**

  In order to obtain a good parallel scalability it is important to parallelize large portions of the code resulting in a so called high "coverage". A high coverage is of importance, because the negative impacts of remaining serial parts in the code increase as more and more cores are employed. The achievable speedup finally becomes limited by these serial parts (see "Amdahl's law", Chandra et al. (2001)).

- **Load balance**

  The overall execution time of a parallel loop is determined by the thread with the slowest execution time. If the work load is not properly balanced between the multiple threads performance will suffer. A good approach for parallelizing a loop often is a simple static scheduling, which means that all iterations are divided in parts of equal size before the threads start execution. But in case that the computational costs of the iterations differ largely, a static scheduling may not be optimal. E.g., in 2-D simulations with dry and wet regions, the flow equations must only be solved entirely for the wet or transitional elements but not for the dry elements.

Beside the problem of moving boundaries of the wetted domains, it is also important to note that hydrodynamic models may lead to imbalanced load distributions in case of local, highly unsteady processes like wave propagations.

The load balance can sometimes be optimized in such cases by the use of dynamic scheduling, which divides the work load dynamically among the threads. Threads with computational cheap iterations will dynamically receive additional iterations, thereby taking load from threads with costly iterations. Dynamic scheduling can improve performance in case of load imbalance, but leads also to an increased overhead and suffers from bad data locality. Consequently, the decision between static or dynamic balancing is largely problem dependent.

- **Synchronization**

  Synchronization between threads can become time consuming if some threads must wait for other threads to finish execution. Quite similar, in some cases threads must be prevented from mutual accessing the same memory locations to prevent read/write conflicts and data races. Such parts of the code must be protected by setting mutual exclusions using locks. These code parts can only be accessed by one thread a time and may cause other threads having to wait until access is granted.

  Such bottlenecks caused by synchronisations and mutual exclusions should be avoided and minimized as far as possible. They can sometimes be prevented by reorganising parts of the algorithms. Prevention of memory conflicts can sometimes also be achieved by privatization of the problematic shared variables, i.e. global variables are replaced by thread private variables.

- **Locality**

  When parallelizing a program attention should be paid on data locality aspects. Modern cache based processors optimize execution speed by caching data which speeds up data load and store cycles. Caches are very efficient if a processor can use the cached data (the local data) for many operations instead of having to reload them from memory or from caches of other processors. In general, better data locality is guaranteed if the work load is distributed statically among the threads instead of using a dynamic schedule.

  Additional problems limiting the parallel performance may arise when OpenMP parallelized programs are executed on cc-NUMA architectures with large number of cores due to general aspects of the underlying hardware architecture (Chapman et al. (2008)).

5.1.3 Parallelization with OpenMP

5.1.3.1 Overview

As mentioned before, parallel programming for shared memory systems bases on multi-threading software concepts. Thread programming can be done on a rather complex and time consuming low level by explicitly specifying and controlling all threading options. In recent years high level concepts for thread based parallelization were developed which ease parallelization and do not require low level thread programming skills. Among these high level concepts it can be differentiated between implicit parallelism using parallel programming languages or automatic parallelization by compilers and explicit

parallelism where the developer controls the parallelism with support of parallel libraries and application programmer interfaces (APIs).

In cooperation with leading software and computer enterprises a parallel API called OpenMP ("Open Multi Processing") was developed which is today the de facto standard for parallelizing scientific and engineering applications on shared memory systems. OpenMP is used for the parallelization of BASEMENT.

5.1.3.2 Characteristics of OpenMP

OpenMP consists of a set of compiler directives and library functions. With these compiler directives the developer describes the parallelism in the code. Therefore a compiler is needed which supports these OpenMP directives. OpenMP is currently supported by many C/C++ and Fortran compilers on a variety of platforms. In the table below some aspects pro and contra parallelization with OpenMP are listed (see also Kuhn et al. (2000) for a comparison of OpenMP and threading in C/C++).

Table 5.1 *OpenMP parallelization – pro and contra.*

Pro	Contra
First successes in parallelization are relatively easy to achieve. Compiler directives reduce the needed programming efforts and implementation details are left to the compiler.	No full control over the implementation details is possible. Bugs in libraries can be difficult to isolate.
Rather small increases in the size of the source code due to the use of compiler directives. Good readability of the code is maintained and the algorithms are not buried by large blocks of added parallelization instructions.	Compiler generated code portions and library calls can complicate debugging of the parallel program.
OpenMP is standardized and portable on different platforms.	OpenMP does not support the whole whole range of threading concepts (e.g. no support for semaphores for synchronization).
The source code can be compiled as serial program without support of OpenMP (compiler directives are treated as comments). This may ease debugging of new implemented application features.	

5.1.3.3 Parallel directives in OpenMP

The basic parallelization construct in OpenMP is the so called "parallel region". The source code placed within a parallel region is executed in parallel on a number of threads. At the end of a parallel region an implicit barrier is automatically set to synchronize the threads.

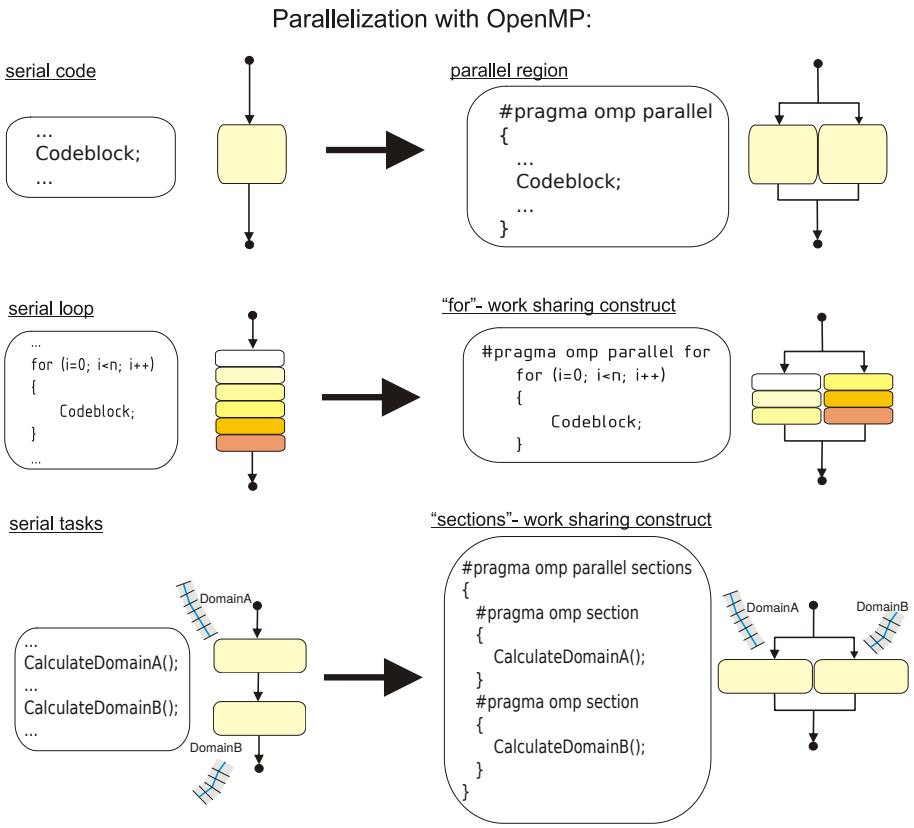


Figure 5.3 *Compiler directives for basic parallelization constructs in OpenMP*

To ease the distribution of work load among the threads of a parallel region, OpenMP supports work sharing constructs which automate such tasks, see Figure 5.3. With optional parameter clauses it is possible for the user to define the behaviour of these compiler directives in more detail. A complete OpenMP reference is available online at http://www.openmp.org ("Online reference of openmp"). For special needs it is also possible to share the work load among the threads fully flexible by individually addressing each thread.

In OpenMP several compiler directives are supported for synchronization issues which cover the most frequent tasks. Such constructs are barriers, locks, critical sections and atomic constructs. Beside these standard constructs, fully flexible synchronisation operations can be implemented for special tasks using global flags in memory. The choice of the best suited constructs for synchronisation and mutual exclusion is problem depended and involves differing efforts in parallel programming.

Some general parameters controlling the parallelism can be set in OpenMP at the program start or during runtime with library calls or by setting environment variables. Such parameters include, for example, the number of parallel threads or the choice between static or dynamic scheduling. This enables the flexibility to let the user determine the number of threads used for parallelization or to optimize the parallel speedup by varying the type of schedule.

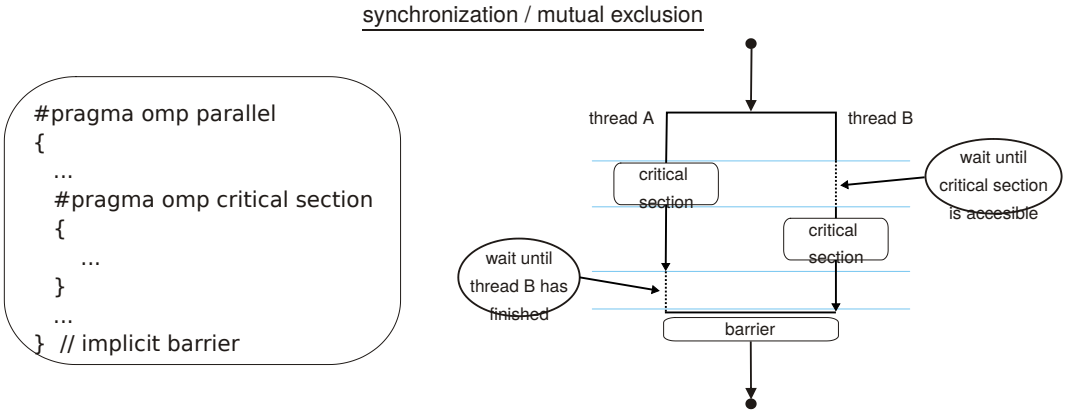


Figure 5.4 *Critical sections and barriers for synchronization operations in OpenMP*

5.2 Model Coupling

5.2.1 Introduction

In addition to the simulation of single sub-domains using BASEchain (1-D) or BASEplane (2-D), the software BASEMENT also provides the possibility to connect sub-domains for combined numerical simulations. Such coupled simulations can range from simple configurations up to simulations of river networks with integrated river junctions / bifurcations or integrated 1-D/2-D modelling. In Figure 5.5 a river network of multiple sub-domains with several coupling interfaces is illustrated. The coupling mechanisms thereby allow to couple hydrodynamic simulations as well as morphological simulations with sediment transport and suspended load.

Some typical applications of coupled simulations are:

- A step wise modeling approach to the overall problem using smaller parts of the whole domain. This approach has the advantages of reduced complexity and reduced execution and calibration times. Also extensions of existing and calibrated models can be easily made with coupled simulations without the need to redesign the existing models.
- Simulations with hydraulic structures (like weirs or gates) within the domain of interest can be realized by using multiple sub-domains, which are coupled via these hydraulic structures.
- Coupled simulations can be helpful for mixed-dimensional modeling approaches, e.g. for cases where large scale 1-D simulations shall be combined with detailed modeling of local areas in 2-D. Thereby, the advantage of efficient and robust modeling in 1-D is combined with the capability to simulate 2-D flow characteristics. Also, the required efforts for data acquisition and data preparation can be minimized using mixed-dimensional modeling approaches.

5.2.2 Coupling Types

The implemented coupling types are briefly sketched below.

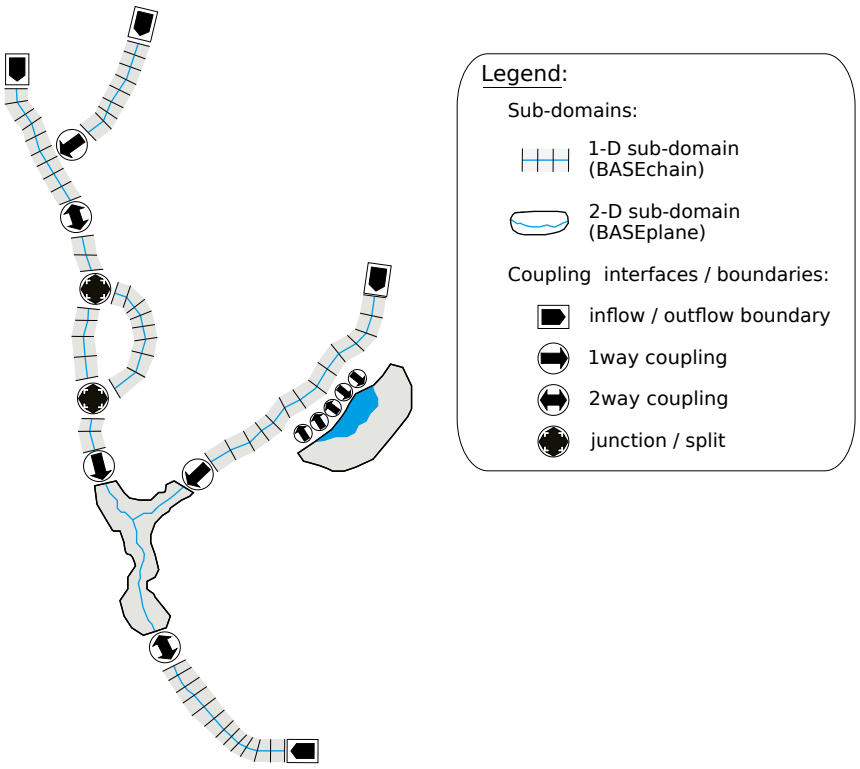


Figure 5.5 *River network with multiple BASEchain (1-D) and BASEplane (2-D) sub-domains and several coupling interfaces.*

- **Sequestial, Riemann (1D)**

  Single sub-domains can be combined sequentially via coupling interfaces at the upstream or downstream boundaries (Figure 5.6). This can also be done for sub-domains with mixed dimensionalities (1-D / 2-D, 2-D / 1-D), see Figure 5.7.

  These **sequential** coupling types can be used to combine sub-domains over their boundary conditions or external sources. For example, a weir outflow boundary can be combined with an input hydrograph of a downstream boundary.

  Beside sequential couplings, also so called **Riemann** couplings can be used (at the moment only 1D) which set a Riemann solver between the sub-domains and allow flow in any direction. This coupling type requires special 'connection' boundaries at the coupling interfaces.

- **Junctions / Bifurcations / ConfluenceWSE (1D)**

  Coupling interfaces for river **junctions** or river **bifurcations** allow a simplified modelling of conjunctions of river branches within a 1-D river network (Figure 5.8).

  Beside these both coupling types also a **confluenceWSE** coupling can be used which tries to establish a common water surface elevation (WSE) at the confluence point (at the moment only 1D). This coupling allows flow in any direction and requires special 'connection' boundaries at the coupling interfaces.

- **Lateral coupling.**

  For integrated 1-D and 2-D modelling, a 1-D sub-domain can be coupled laterally with a 2-D sub-domain. The coupling takes place along the river channel via multiple

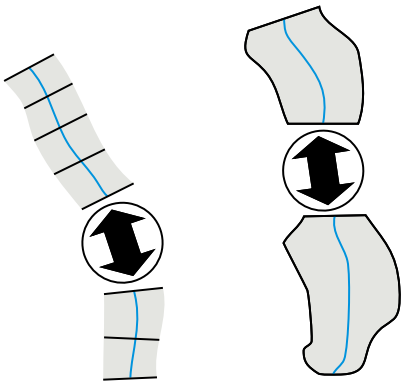


Figure 5.6 *1-D / 1-D coupling (left), 2-D / 2-D coupling (right)*

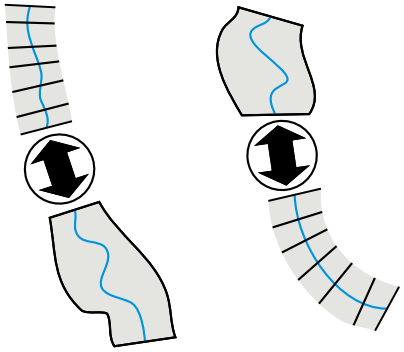


Figure 5.7 *1-D / 2-D coupling (left), 2-D / 1-D coupling (right)*

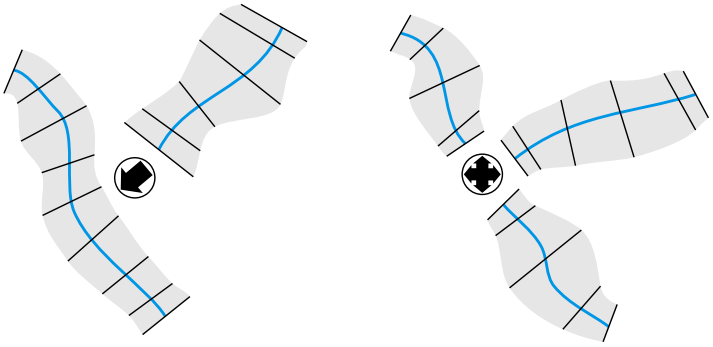


Figure 5.8 *junction /bifurcation / confluence WSE (1-D)*

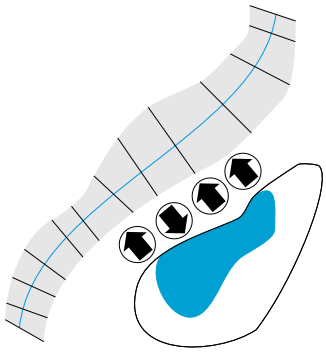


Figure 5.9 *lateral coupling*

coupling interfaces which connect cross sections (1-D) with corresponding mesh elements (2-D), see Figure 5.9.

5.2.3 Coupling Mechanisms

5.2.3.1 Explicit coupling of sub-domains

Coupling of sub-domains is implemented as an explicit coupling approach, which means that data is exchanged explicitly between the sub-domains at certain time intervals. This approach is simpler to implement than an implicit approach, especially regarding the coupling of sub-domains with mixed dimensionalities. However, in comparison to an implicit coupling approach, special care must be taken to achieve robust and stable combined simulations.

5.2.3.2 One-way coupling and two-way coupling

A simple way to couple two sub-domains is to exchange data only in one direction from upstream to downstream. Such a situation is termed as 1-way coupling from here on. It has the advantage that the upstream sub-domain can run independently from the downstream sub-domain and the flow variables are passed over at some time intervals to the downstream sub-domain. But being a one-directional coupling, no information from downstream can travel upstream. Therefore, this type of coupling is restricted to cases where no backwater effects from downstream take place or such influences can be neglected.

In contrast, a two-way coupling enables mutual interactions between the sub-domains by providing mutual data exchange. In two-way coupled sub-domains, backwater effects from downstream can influence the upstream sub-domain. Instead of executing the sub-domains sequentially from upstream to downstream direction, here the sub-domains are executed simultaneously. The two-way coupling approach has the difficulty that no unique flow variables are present at the coupling cross sections, as water levels from upstream and downstream direction may differ for a given time. In principle, iterations between the sub-domains are required and must be performed until the differences of the variables at the coupling cross section do no longer change within subsequent iteration steps. Although a rather small number of iterations has to be expected (as reported by Miglio et al. (2005)), these iterations lead to large additional computational efforts. Therefore these iterations are not performed here. As the time steps in the explicit approach are usually very small, the

differences between the upstream and downstream variables are rather small and iterations may be neglected without substantial loss of accuracy. But in cases of crucial and abrupt changes in the flow variables, oscillations may result.

A combination of both concepts can be used in the coupled river simulation. One-way coupled sub-domains are executed sequentially from upstream to downstream direction, whereas two-way coupled sub-domains are treated as being a single sub-domain within the execution sequence.

5.2.4 Definitions of Exchange Conditions

5.2.4.1 General remarks

Data can be exchanged between the sub-domains by coupling interfaces using boundary conditions and source terms. The following table shows the exchange variables grouped by the direction of the exchange.

Table 5.2 *Possible exchange conditions between sub-domains.*

direction of exchange	type of coupling	exchange variables
in downstream direction	boundary conditions & sources	Q : discharge $q_{b,g}$: bed load C_g : concentration
in upstream direction	boundary conditions & sources	z_s : water surface elevation

In order to enable simple, flexible and efficient coupled simulations, some assumptions are made here:

- It is assumed that flow directions at the coupling interfaces of the river network are known a priori and do not change during the simulation (with the exception of special coupling types, like the lateral coupling).
- The cross sections (1-D) or mesh elements (2-D) of the coupling interfaces should ideally be located at the same or nearby locations and have the same geometries. This is necessary to reduce possible errors around the coupling interfaces due to the disregard flow taking place in between and to avoid discontinuities due to abrupt changes in the geometries.
- It is assumed that the flow is orthogonal over the boundaries, i.e. the directional x- and y flow components in 2-D are not exchanged separately.
- In 2-D coupling only summarized or averaged data are exchanged, instead of exchanging data separately for each edge or element. This approach simplifies the coupling setup since no restrictions are set regarding the geometries and number of cells at the boundaries or sources.

5.2.4.2 Exchange conditions for mixed-dimensional sub-domains

Exchange via boundary/sources:

Table 5.3 *Exchange conditions for mixed-dimensional coupling (i=index of 2-D edge or 2-D element)*

Echange variable	Exchange equations	Nr of exchange terms
Discharge	$1D \rightarrow 2D$: $q_i^{2D} = \omega_i Q^{1D}$ (ω_i=area/length weighting or conveyance weighting) $2D \rightarrow 1D$: $Q^{1D} = \sum_i^n q_i^{2D}$	1
Water surface	$1D \rightarrow 2D$: $z_{S,i}^{2D} = z_S^{1D}$ $2D \rightarrow 1D$: $z_S^{1D} = \frac{1}{n} \sum_i^n z_{S,i}^{2D}$	1
Bed load	$1D \rightarrow 2D$: $q_{b,g,i}^{2D} = \omega_i q_{b,g}^{1D}$ (ω_i=area/length weighting) $2D \rightarrow 1D$: $q_{b,g}^{1D} = \sum_i^n q_{b,g,i}^{2D}$	g=1..ng
Suspended load	not available yet in 2D	g=1..ng

5.2.4.3 Exchange conditions for river junctions in 1-D river networks

Within a river network locations are encountered where river branches flow together or where a river bifurcates into several branches. The flow characteristics at such conjunctions generally are multidimensional. Therefore the preferable modelling approach to achieve a good accuracy is to simulate a 2-D sub-domain. But if such a situation shall be modelled with 1-D sub-domains than special coupling concepts are required.

Two different approaches are implemented in BASEMENT (see Figure 5.10). These approaches allow no more than three sub-domains being part of a junction. If a larger numbers of river branches are to be modelled, they must be approximated by multiple junctions, placed in small distances.

Following the first approach a junction can be regarded as region where three different river branches meet and mutually exchange data (a). A control volume is defined to which mass and momentum conservation principles can be applied. A simple approach is here balancing discharges and assuming equal water surface elevations along the junction.

Table 5.4 *Exchange conditions for river junctions*

	Exchange conditions	Nr. of equations
Discharge	$Q_{up1} + Q_{up2} = Q_{down}$	1
Bed load	$Q_{up1,bed,g} + Q_{up2,bed,g} = Q_{down,bed,g}$	g=1..ng
Suspension	$Q_{up1} C_{up1,g} + Q_{up2} C_{up2,g} = Q_{down} C_{down,g}$	g=1..ng

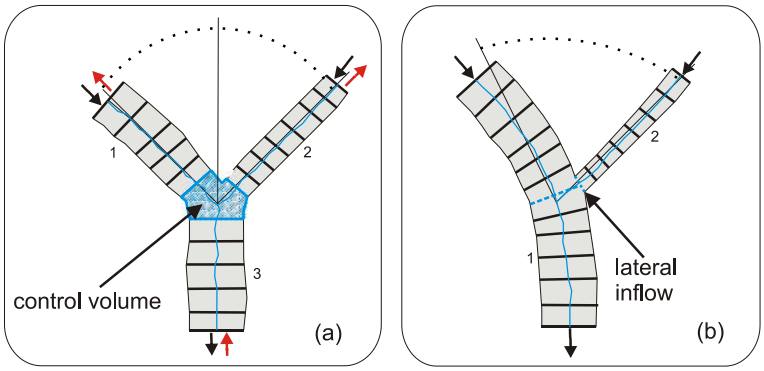


Figure 5.10 *Modeling of a river junction with two different approaches (black arrows indicate a confluence of river branches, red arrows a bifurcation): (a) = river junction with 3 different river branches, (b) = junction as a lateral inflow of a tributary.*

The second approach is to regard the junction as a lateral inflow of a tributary into a river at a specified location (b). The discharge (and sediment) is passed from the tributary to the river as lateral inflow via source term. Additionally, the water level at the inflow cross section can be passed in return to the tributary. Despite its simplicity this approach can be suited well to simulate simple river junctions in 1-D.

5.2.4.4 Exchange conditions for river bifurcations in 1-D river networks

In case of modeling a river branch which bifurcates into two branches, the upstream discharge (and sediment) must be distributed among the two downstream sub-domains. The distribution factor ϕ among the downstream sub-domains has to be chosen according the local conditions. The downstream water elevations of the two downstream sub-domains are averaged and then passed in upstream direction.

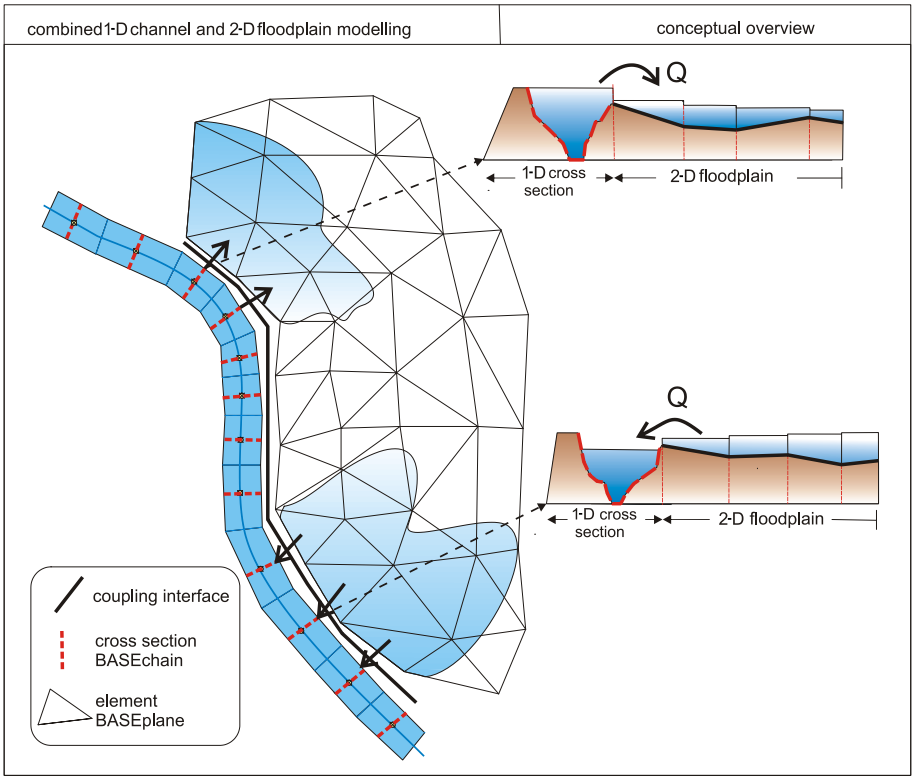


Figure 5.11 *Conceptual overview of combined 1-D river flow and 2-D floodplain modeling*

Table 5.5 *Exchange conditions for river bifurcations*

	Exchange conditions	Nr. of equations
Discharge	$Q_{up} = \phi Q_{down1} + (1-\phi) Q_{down2}$	1
Bed load	$Q_{up,b,g} = \phi Q_{down1,b,g} + (1-\phi) Q_{down2,b,g}$	g=1..ng
Suspension	$Q_{up} C_g = \phi Q_{down1} C_{down1,g} + (1-\phi) Q_{down2,g} C_{down2,g}$	g=1..ng

5.2.4.5 Exchange conditions for combined 1-D and 2-D modelling

The combined 1-D river flow and 2-D floodplain modelling bases mainly on the approach presented by Beffa (2002). A conceptual overview is given in Figure 5.11 which illustrates river cross sections of the BASEchain sub-domain and the 2-D mesh of a floodplain modelled with a BASEplane sub-domain.

The coupling interfaces between the sub-domains are implemented as one-way couplings via source terms. As a consequence, only discharges are exchanged between the sub-domains. The exchange between the sub-domains is calculated as weir flow over the dykes of the 1-D cross section or as weir flow over the edges of 2-D sub-domain. The weir level is chosen as the higher elevation of the dyke or the corresponding edge. As weir width in the weir formula the length of the 2-D boundary edge is taken. Exchange of discharge is possible in both directions, either from the river into the floodplains or backwards depending on the water elevations in the 1-D cross section and the corresponding 2-D element.

To enable a flexible coupling approach it is possible to connect a 1-D cross section with multiple 2-D elements (1:n-relation). For each 2-Element, in contrast, only one connection to a 1-D cross section is possible (1:1-relation). The coupling interfaces are defined using a list, from which the connections are automatically extracted and generated during a pre-processing step.

Table 5.6 *Exchange conditions for lateral coupling*

	Exchange conditions	Nr. of equations
Discharge	$1D \rightarrow 2D$: $Q = \delta\ \mu\frac{2}{3}b_{weir}\sqrt{2g}h^{3/2}$ if $(z_{S,1D} \geq z_{S,2D})$ (side weir, δ= side weir reduction factor)	1
	$2D \rightarrow 1D$: $Q = \mu\frac{2}{3}b_{weir}\sqrt{2g}h^{3/2}$ if $(z_{S,1D} < z_{S,2D})$ (weir overfall over edge)	

The 1-D dyke crest elevation, where the water overtops, as well as the 1-D water surface elevation are interpolated between the cross section at the location of the 2-D edge. This procedure shall increase the accuracy of the lateral exchange modeling. It is assumed hereby, that the dyke-crest elevation and the water surface elevation vary linearly between two adjacent 1-D cross-sections.

5.2.4.6 Data exchange for morphological simulations with multiple grain classes

In morphological, coupled simulations the possibility exists that sub-domains can have differing grain compositions. The handling of data exchange for such cases is not trivial and unclear. But such situations may arise in coupled large-scale simulations where grain classes get finer along course of the river. A flexible approach is adopted here which allows the usage of differing compositions as well as different numbers of grain classes of the sub-domains.

For data exchange the bed loads of each grain class are mapped on the grain classes of the receiving sub-domain. The mapping is achieved by three successive steps as illustrated in Figure 5.12. The sediment mass balance is thereby fulfilled.

5.2.5 Synchronization Concept

5.2.5.1 General remarks on synchronization

For the coupling of sub-domains a synchronization mechanism must be implemented which directs the execution of the sub-domains and controls the data exchanges at the appropriate times. The type and complexity of the synchronization effort thereby generally depends on the degree of spatial and temporal compatibility of the sub-domains. Especially in case of combined 1-D and 2-D simulations the spatial extends and time step sizes can vary considerable.

Mainly two different coupling concepts are often encountered for the selection of the time step sizes of the sub-domains.

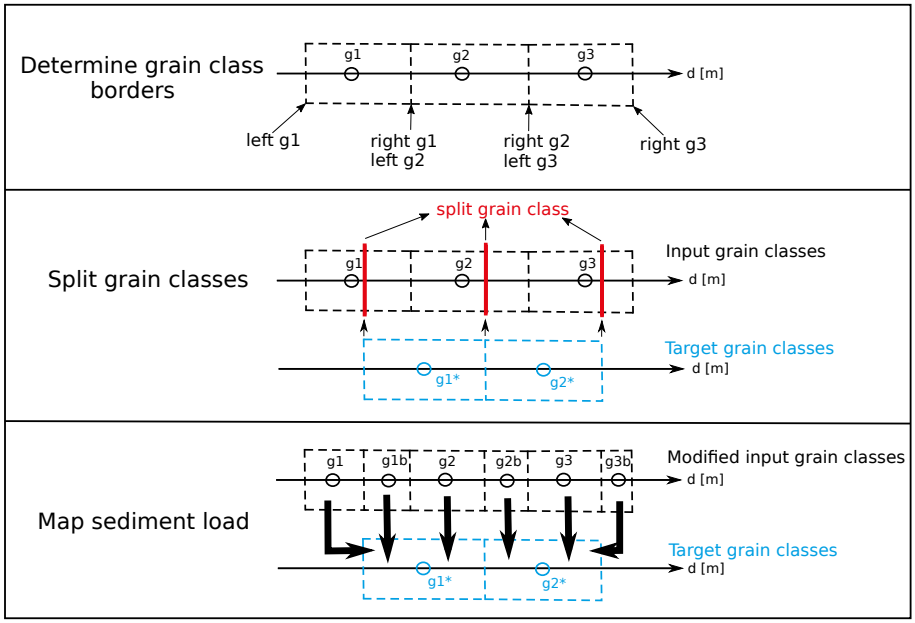


Figure 5.12 *Mapping of grain compositions from one sub-domain to another*

- All sub-domains are executed in a synchronous manner with an equal time step size. To guarantee stable execution the chosen time step size ("global time step") is set to the minimum time step size of all sub-domains, which is determined by stability conditions (CFL criterion). But due to the fact that the sub-domain time step sizes can vary considerable, such a restriction on the minimum time step size can lead to inefficient small time step sizes resulting in large computational efforts.
- In contrast, all sub-domains can be executed asynchronous with different time step sizes ("local time steps"), which are chosen according the sub-domain's optimal time step size. This approach does not suffer the computational inefficiency due to small time step sizes. But generally more synchronization efforts are required and data exchange between the sub-domains requires interpolations and can become cumbersome especially for complex interfaces like junctions or bifurcations.

Here, another approach is selected, a local-time stepping approach, lying in between these concepts and combining efficiency and simplicity.

5.2.5.2 "Local time stepping" approach

This approach bases on the method of local time stepping (LTS) as presented by Osher and Sanders (1983) and Sanders (2008). But in contrast to these methods, LTS is applied here to whole sub-domains instead of single grid elements. Different local time step sizes are allowed for the sub-domains instead of using one global time step for all sub-domains. This enables efficient computations by preventing very small time steps of single sub-domains to dominate the time step sizes of the other sub-domains. But restrictions are set for the time step sizes in a way to ensure that the sub-domains always reach common time levels. At these common time levels data can be exchanged easily without the need for interpolations.

Hierarchical levels L are introduced and attributed to each sub-domain. These levels categorize the sub-domains into groups of common time step sizes. These levels are thereby

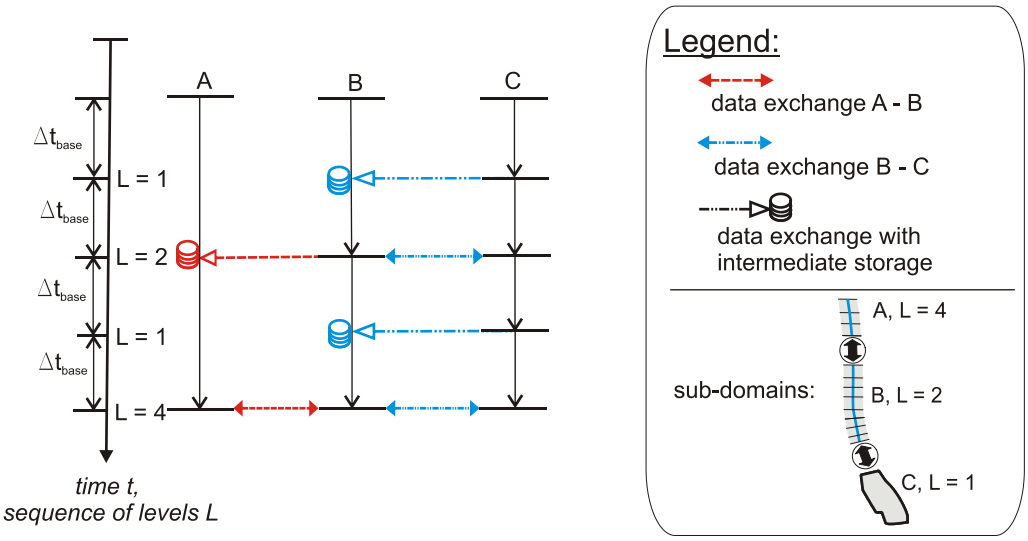


Figure 5.13 *LTS-synchronization for 3 sub-domains with different time step sizes. The sub-domain C with the smallest time step size determines the base time step. Sub-domains A and B run for multiples of 4 and 2 of the base time step size.*

chosen as power-of-two multiples of the base time step size Δt_{base} . This base time step is selected as the minimum time step size of all coupled sub-domains. The attribution of levels L to a sub-domain i depends on the relation of its present time step size to the base time step size and is determined as:

$$2^k \leq \frac{\Delta t_i}{\Delta t_{base}} < 2^{k+1} \rightarrow L_i = 2^k, k = 0..n$$

where L_i is the level attributed to the sub-domain i , k indicates the level and n is the number of levels. Each sub-domain determines its own local time step size as its level L_i multiplied with the base time step size $\Delta t_i = L_i \Delta t_{base}$.

The execution of the sub-domains takes place in loops over level sequences. One loop sequence of the LTS synchronization is sketched in Figure 5.13 for three sub-domains with different time step sizes ($\Delta t_A > \Delta t_B > \Delta t_C$) and levels L_i.

For example, in case that the maximum level of a sub-domain is 8, a level sequence of $m =$ [1,2,1,4,1,2,8] is executed, where m equals the present level of the loop. Each sub-domain is executed only if its level L_i is smaller or equal to the present level m. These sub-domains are then advanced for a time step size of $\Delta t_i = L_i \Delta t_{base}$. Data exchange between adjacent sub-domains takes place only when the sub-domains have reached a common level. If adjacent sub-domains have different time levels then the exchanged data must be stored intermediately to guarantee conservation principles. The data is finally passed over when the sub-domains reach a common time level. After the end of the loop of the level sequence, all sub-domains have been executed at least once and have finally have reached a common final time $t_{new} = t_{old} + n\Delta t_{base}$. From this starting point the levels L_i are assigned again to the sub-domains and the procedure is repeated.

The selection of the base time level is done at the beginning of each level loop. To account for the possibility that the minimum time step could change during the loop iterations, due to changed flow conditions, the base time level can be reduced by a factor $F \leq 1$ for stability reasons.

5.2.6 External Coupling

5.2.6.1 Introduction

The term "external coupling" means the coupling between the program BASEMENT and an external program. This may be e.g. a rainfall-run off model which delivers input data for a river reach or it may be a standalone groundwater model which makes use of the stream water elevations computed by BASEMENT. As described in the model coupling section, one can also distinguish here between one-way coupling and two-way coupling.

One-way coupling

Two different scenarios can be distinguished here:

- An external Program may receive data which is sent by BASEMENT. Therefore the external program must be defined as an external sub-domain in the command file. In the OUTPUT block an output must be defined and connected with this external sub-domain. While executing, BASEMENT sends data as soon as it is computed using the output routines to the specified external sub-domain. The external program must fetch the data using TCP/IP routines and must take care of the synchronization, i.e. it must always wait until new data is available.
- Another scenario is to send input data to BASEMENT. Thereby the external program again has to be defined as an external sub-domain in the coupling process and it must be connected with other sub-domains using boundary conditions. Then, the external program can send its data in XML format to BASEMENT using TCP/IP routines. BASEMENT takes care of the synchronization within the coupling process and always waits until new data is available before executing.

Two-way coupling

It is possible to couple an external program with BASEMENT with mutual data exchange. Again, the external program must be defined as an external sub-domain in the coupling process. Furthermore, the external program must implement a synchronization mechanism in order to check if the needed data is available. Differing from previous versions, BAEMENT now does NOT apply the local-time-stepping algorithm (LTS) to the external coupling. Data exchange takes place, when either BASEMENT or the external program is ahead in time.

For example:

- The external program is executed until it is ahead in time compared to BASEMENT. Then it has to send its current time and input data to BASEMENT. Afterwards, it waits and checks for incoming data over TCP/IP.
- BASEMENT waits until the external program is ahead in time. While waiting, it checks for incoming data over TC/IP. If the incoming data shows that the external program is ahead in time, then BASEMENT runs as long as its current time is behind the time of the external program. Afterwards, if BASEMENT run-time exceeds the current time of the external program, it sends its current time and data to the external program.

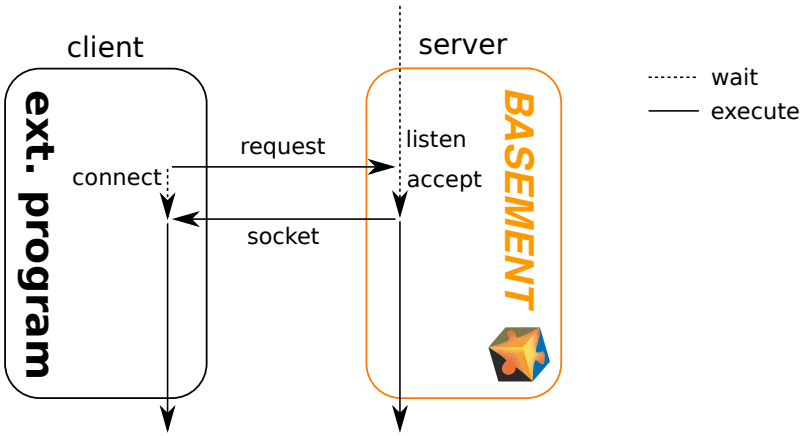


Figure 5.14 *Connection request from external program (client) to BASEMENT (server)*

Note

Please be aware that the external coupling approach is still in an experimental stage. In addition, the usage of this coupling requires programming efforts and knowledge in TCP/IP programing and XML parsing. External coupling may require the implementation of special boundary conditions in the BASEMENT model. For example, coupling with a groundwater model requires leakage boundaries for water exchange to be set. If you want to make use of such a coupling type, you may contact the developer team regarding the implementation of appropriate boundary conditions in the model.

5.2.6.2 Data exchange over TCP/IP

The data exchange between BASEMENT and the external program takes place using TCP/IP communication. This has the advantages that it is generally faster than communications via files and enables the coupling between different computers via intra- or internet, even using different operating systems.

Create connection

The communication requires an IP-address and a port-number as identifier and takes place using TCP-sockets. Usually BASEMENT runs as the server application and must be started first. Then, it waits for an incoming connection request. After the incoming request, a connection is established with the external program and the connection information is sent to the external program (socket descriptor). In case of multiple external programs, BASEMENT waits until all connections are established before it starts the computations.

Data packet

The data is wrapped in “data packets” using the common XML-format, whereby the data values and several additional attributes must be specified. All communications between the programs take place by sending data packets. Therefore, also additional information, like e.g. the time, or the time step size, must be included in the data packet. It is also possible to send or receive multiple data packets for different data types or boundaries. The XML-tag has the following structure

```
<Data attribute1="..." attribute2="...">...<\Data>
```

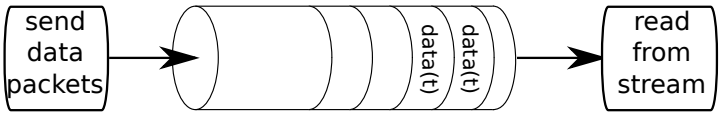


Figure 5.15 *Sending and receiving data from socket stream (FIFO pipe)*

The data values within the XML tag can either be written as ascii or binary data. If ascii format is used, a semicolon separates multiple data values from each other, e.g.

```
<Data>10.0;10.0;10.0</Data>
```

The following attributes can be set:

Attribute	Values	Obligatory
size	number of data values	Yes
time	time of data values in sec	Yes
type	[Q,h,v,...] type of data	Yes
encoding	[ascii, binary]	No (default=ascii)
byte_order	[little_endian]	No
boundary	name of boundary condition of data values	Only if data is sent to BASEMENT
timestep	current time step of model in sec	Only for local time stepping

Data communication

The data communication via sockets can be compared to data exchange via file-streams. The data packets are inserted into a pipe and the other side of the connection reads the contents after the FIFO concept (First In First Out). The receiving part of the connection must parse the contents of the pipe and extract the data packets. The time attribute of the data packets indicate the time level of the other program required for the synchronization. If the time levels of the data packets are behind the program's time or if no data is in the pipe, than the program must wait and continuously check for incoming data.

5.3 Flow Control in River Systems

5.3.1 Introduction

The flow characteristics of a river system are not only governed by the character of a channel, the morphology and topography, but also by regulations for hydropower stations and lakes. Such regulations commonly demand that a certain water level is maintained or

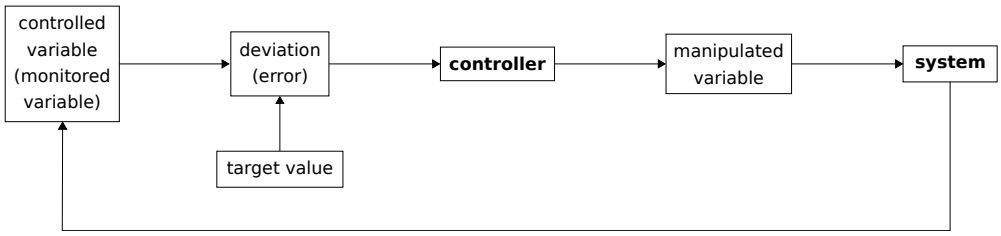


Figure 5.16 *Basic Control Cycle*

impose certain limits on the maximum discharge. The exertion of control structures has commonly a significant direct impact on certain river sections or even on the whole river system. The setting of the control structures over time cannot be defined in advance, but depends on the reaction to a change of the whole river system.

The numerical simulation of regulations is very helpful to properly judge such river systems, as it allows assessing and optimizing the effect of individual regulations of control structures on the whole system. This is of great importance as efficient flood control demands an optimal use of existing retention structures.

Therefore, the automatic steering of control structures has been added to BASEMENT, covering 1-D and 2-D simulations as well. The chosen approach allows the simultaneous combination of different controlled and manipulated variables. Controlled variables can be either water surface elevations or discharges. Here not only fixed values can be defined, but also series in time or values depending on the current flow in the river system. As manipulated variable, settings of weir or gates and an abstract outflow hydrograph has been implemented.

Within the present implementation, the determination of the control structure settings have been strongly abstracted, which allow a very flexible integration of further controlling algorithms in the future. As reference, a classical Proportional-Integral-Derivative (PID) controller has been implemented. By combining various control and manipulated variables within a single controller, BASEMENT now offers the possibility to simulate complex series of weirs over coupled regions.

5.3.2 Concept of Flow Control

There are many cases where the behaviour of boundary conditions such as weirs or gates depends on the actual state of the river system and cannot be described by a simple time-dependent boundary setting. An example would be adjusting the weir height in order to maintain a specific water level in front of the weir. This process is commonly denoted as controlling. The basic controller has a controlled variable, such as water surface elevation, which is desired to be kept at a certain level, i.e. its target value. The deviation from the current value of the controlled variable and its target value, also denoted as error, is then fed into the controller. The controller reads the deviation and calculates the new value for its manipulated variable. An example for a manipulated variable would be a weir height. The new value for the manipulated variable is then fed into the system, i.e. the hydraulic simulation, which finally affects the controlled variable.

In Basement, the system is represented by the simulation, the controller is a mathematical function $\boldsymbol{f}(.)$, determining the values of the manipulated variables $\boldsymbol{u}(t)$ from the values of

the monitored variables $\boldsymbol{m}(t)$. This can be expressed using the following mathematical expression:

$$\boldsymbol{u}(t) = \boldsymbol{f}(\boldsymbol{m}(t))$$

Logically, there can be multiple monitored and manipulated variables.

5.3.2.1 Monitored Variable

A monitored variable is defined by

$$m_i(t) = \nu_i(t - \tau_i) - \nu_{target,i}$$

Here, ν_i can be either a water surface elevation, measured on a specific cross-section (1-D) or element (2-D), or a water flow over a cross section (1-D) or a STRINGDEF (2-D). $\nu_{target,i}$ describes the target value, or in case of a *feed forward* controller, the equilibrium state. τ_i is a delay time controlling when the information of the measured variable is fed into the controller.

5.3.2.2 Manipulated Variable

A *manipulated variable* refers to a boundary condition and can be a weir height, a gate level or an outflow (in case of 1-D hydrographs as downstream boundary).

5.3.3 Controller Types

5.3.3.1 PID-Controller

One possible approach to describe the mathematical function $\boldsymbol{f}(.)$ is a PID (proportional-integral-derivative) controller. This type of controller relates a monitored variable to a manipulated variable by three additive controller elements:

$$u_i(t) = \underbrace{K_{p,ij} m_j(t)}_{P-Element} + \underbrace{\int_0^t K_{I,ij} m_j(t')dt'}_{I-Element} + \underbrace{K_{D,ij} \frac{d}{dt} m_j(t)}_{D-Element}$$

Internally, the PID-controller is implemented in its differential form (i.e. the change of is calculated in each time step). The three required variables are K_P, K_I and K_D. The correct definition of these variables is very crucial to the proper operation of the controller. Which values should be used is highly dependent on the system and therefore requires some care and experience.

The P-element represents an adjustment proportional to the deviation and therefore only limits the deviation, but does not bring the system back into the state where no deviation exists. For this reason, the I-Element integrates the deviation and consequently, the system can be forced into its equilibrium state. If the response of the I-Element is too strong compared to the P-Element, the system oscillates. If the values of both P and I elements are too small, the reaction of the system is very slow or even too weak to re-establish the

given targets. The D-Element depends on the change of the monitored variable and is used to quickly adapt the manipulated variables in case of a fast change.

More on the determination of correct PID coefficients can be found in the article by Fäh and Kühne (1987). Recommendations on how to choose the coefficients are given in the integrated help of the software.

6

References

Barth, T.J. (1994). Aspects of Unstructured Grids and Finite Volume Solvers for the Euler and Navier-Stokes Equations. von Karman Institute Lecture Series for Fluid Dynamics (1994-05).

Beffa, C. (2002). Integration ein- und zweidimensionaler Modelle zur hydrodynamischen Simulation von Gewässersystemen. *Int. Symposium Moderne Methoden und Konzepte im Wasserbau*, ETH Zürich.

Bern, M. and Eppstein, D. (1995). Mesh generation and optimal triangulation. *Computing in euclidean geometry*, Du and Hwang, F. eds., 47–123. *World Scientific*, Singapore.

Bern, M. and Plassmann, P. (2000). Mesh Generation. *Handbook of Computational Geometry*, Sack, J. and Urrutia, J. eds., *Elsevier Science*, Amsterdam, The Netherlands.

Chandra, R., Dagum, L., Kohr, D., Maydan, D., McDonald, J. and Menon, R. (2001). Parallel Programming in OpenMP. *Morgan Kaufmann Publishers Inc*, San Francisco, CA, USA.

Chapman, B., Jost, G. and van der Pas, R. (2008). Using OpenMP - Portable Shared Memory Parallel Programming. *The MIT Press*, Cambridge, Massachusetts.

Fäh, R. and Kühne, A. (1987). Numerische Simulation automatischer Stauregelungen bei Laufwasserkraftwerken. (p. 9. Heft 5/6, Ed.) *Wasser, Energie, Luft*, 79(5/6): 93–98.

Kuhn, B., Petersen, P. and O'Toole, E. (2000). OpenMP vs. threading in C/C++. *In European Workshop on OpenMP Concurrency: Pract. Exper*, No. 12: 1165–1176. *John Wiley & Sons, Ltd.*, New Jersey.

Manferdelli, J., Govindaraju, N. and Crall, C. (2008). Challenges and Opportunities in Many-Core Computing. *Proceedings of the IEEE.*, No. 96: 808–815.

Miglio, E., Perotto, S. and Saleri, F. (2005). Model coupling techniques for free surface flow problems: Part I. *Nonlinear Analysis: Theory, Methods & Applications*, 63: 1885–1896.

Online reference of openmp.

Sanders, B.F. (2008). Integration of a shallow water model with a local time step. *Journal of Hydraulic Research*, 46(4): 466–475.