DEPARTMENT OF CIVIL AND ENVIRONMENTAL ENGINEERING
THE NORWEGIAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

A THREE-DIMENSIONAL NUMERICAL MODEL FOR SIMULATION OF SEDIMENT MOVEMENTS IN WATER INTAKES WITH MULTIBLOCK OPTION

# SSIIM 2 wx

# User's Manual

BY NILS REIDAR B. OLSEN

21. NOVEMBER 2023

# Foreword and history

The SSII program was developed in 1990-91 during the work with my dr. ing. degree at the Division of Hydraulic Engineering at the Norwegian Institute of Technology. SSII is an abbreviation for Sediment Simulation In Intakes. The program was originally built around the CFD program SPIDER, made by Prof. M. Melaaen during the work on his dr. ing. degree in 1989-90. SPIDER solves a flow problem for a general three-dimensional geometry. SSII was made up of sediment calculation routines for 3D solution of the convection-diffusion equation for the sediments, communications with SPIDER and a graphical user interface made in OS/2.

The main motivation for making SSII was the difficulty to simulate fine sediments in physical models. The fine sediments, often under 0.2 mm, are important for wear on turbines. It was also an advantage to be able to simulate other problems as for example sediment filling of reservoirs and channels.

After finishing my dissertation in 1991, I wanted to improve the CFD program. A disadvantage with the SSII and the SPIDER programs for practical situations was that a structured grid was used, and it was only possible to have one block for an outblocked region. A natural improvement was a multi-block model with general outblocking possibilities. This meant considerable changes in SPIDER. Instead, a new water flow module for multi-block calculation was made. This model was added to SSII, and the resulting model was called SSIIM.

SSIIM, version 1.0, was uploaded on the Internet 17th of June 1993. Version 1.1 had some bug-corrections and some improvements in the water flow calculation for multiple blocks. Version 1.1 was uploaded on the net 18th of October 1993. In the fall of 1993 version 1.2 was made, with an improved user interface, some additional tools and a revised manual. It was uploaded on the net 22nd of December 1993. It was also distributed by diskette to selected water institutions in January 1994. Version 1.3 included several bug-fixes, improved sediment calculation and improved graphics. This was uploaded on the net 5th of April 1994. Version 1.4 also included transient calculations of water flow, free surface and sediment transport, water quality, a 2D depth-averaged water flow module and improved graphics.

SSIIM 1 uses a structured grid, which makes it difficult to model very complex geometries. In the summer of 1997 the main modules of SSIIM 2.0 was made, with an unstructured and nested grid. This was tested and enhanced during the following two years of my post-doc study in Bristol and Trondheim.

In the summer/fall 1999 both versions of SSIIM were ported from OS/2 to Windows. Version 1 was split into three executables: The *GridEditor*, the OpenGL 3D graphics and the main program. SSIIM 2 had one executable, with the *GridEditor* incorporated, and no OpenGL graphics. The OpenGL graphics was only supported on Windows NT, and omitting it in SSIIM meant that the program could be run on Windows 95 and Windows 98. The main advantages of the Windows version were more computers could run the program and the Windows version ran roughly twice as fast as the OS/2 version, probably due to the faster compiler. In SSIIM 1.1 for Windows, the *GridEditor* was again included into the main program. The OpenGL graphics was a separate program: si3dview. In 2001, algorithms were made that printed out files that could be read directly by the Tecplot program. The si3dview program is therefore

not planned to be updated. In the spring 2007, algorithms for writing input files for the ParaView program was added. The ParaView program is similar to Tecplot, but it is freeware. Also, algorithms to write an OpenFOAM mesh was added in the spring 2007. OpenFOAM is a general-purpose CFD program that is also freeware, with available source code and many more turbulence models than SSIIM. During some period of time, the OpenFOAM files produced by SSIIM were not compatible with the newest OpenFOAM version. However, as of 2023, updated SSIIM algorithms were again able to make OpenFOAM grid files.

In the spring 2001, the Windows versions of SSIIM was made with DLL libraries. DLL is an abbreviation for Dynamic Link Libraries. DLLs containing numerical algorithms for sediment transport and flow resistance from vegetation were made. The DLLs may be further developed by cooperating research groups. In 2005, the source code for the beddll.dll file was made available on the web. The DLL files are discontinued in the SSIIM 2 versions made after 2018.

A native Linux version of SSIIM 1 without user interface was made in 2005, and made available on our web pages. The source code for this version is almost identical to the Windows source code for the computational part.

In the summer 2007, work was started on parallel versions of the SSIIM programs. Implementation was done by using OpenMP. This enabled the utilization of multi-core capabilities of the emerging processors. It also enabled the use of the program on high-performance clusters with shared memory nodes. Initial work on an MPI version of SSIIM 2 was started in the fall 2012. This work was discontinued, as it became too complicated to maintain.

The development of the SSIIM 1 program was discontinued in 2016.

From 2010, the nested grid option in SSIIM 2 was further developed in connection with sediment transport computations. The main application was local scour. In 2013, algorithms to compute geotechnical sand slides were coded in SSIIM 2. These algorithms were further expanded in 2018. New algorithms for the free surface was developed in 2014 and implemented in SSIIM 2. Algorithms in SSIIM 2 for dredging of a reservoir/river was made in 2017, including the use of the “bagger” file.

In 2020, algorithms were made for multiple sediment layers. A new grid algorithm was made in 2022, with considerable simplifications and options with more layers along the bed and non-uniform distribution of the grid cells in the vertical direction. The new grid algorithm meant that complex grids could be written to OpenFOAM blockMeshDict files, giving the possibility to make a grid with SSIIM 2 and use it with OpenFOAM later. Also, combinations of grids that follow the bed and grids with horizontal grid lines were implemented. This was believed to improve computations of turbidity currents, as a more dense grid with hex-cells could be made close to the bed.

In 2022, the algorithms computing drag from vegetation was extended to compute immersed solid objects covering multiple cells. This enabled modelling of complex geometries that did not fit the computational grid.

In 2017, the computer I used to program SSIIM broke down after 10 years of service. I wanted to move to more open source tools, so I switched the compiler and the IDE to GNU C++ and CodeBlocks with wxWidgets for the graphics. This meant rewriting the user interface of the program. This was

considerable work, but it also meant the possibility to improve the SSIIM graphics. The new version has multiple windows that be shown at the same time, similar to the earlier OS/2 version of SSIIM. The new computer was 64 bits, meaning the 32 bits SSIIM versions were discontiuned. The use of CodeBlocks for making the user interface also meant it was possible to port the whole program with the user inteface to Linux. Initial tests were done using Debian, but the problem was that the Debian version of SSIIM 2 was not compatible with Ubuntu. On the other hand, the Windows version running Wine was compatible with both Ubuntu and Debian.

The Windows/wxWidgets version of SSIIM was slower than the original Windows versions. However, the Linux version without user interface had a similar computational time as the original Windows version. Two versions of SSIIM 2 was therefore made: A Windows/wxWidgets version and a Linux version without user interface. The Linux version was successfully tested on Ubuntu and Debian. The SSIIM versions with wx in the name was Windows versions with a graphical user interface, and the versions with an l in their name was a native Linux version without user interface. For example ssiim2wx532.exe and ssiim2l532.exe.

Over many years I have been working on web pages for my research on CFD using SSIIM. The address of the page is http://folk.ntnu.no/nilsol/cfd. In the last couple of years, I have also given information about some cases at the web pages http://www.pvv.ntnu.no/~nilsol. The pvv.org address will be active after I retire from NTNU, as I have a lifelong membership at pvv. The web pages at http://folk.ntnu.no/nilsol may be removed after I retire.

I would like to thank all the people who have provided me with insight into the various problems I have encountered in the development of this program. I have benefited greatly from the knowledge of Prof. Morten Melaaen at Telemark Institute of Technology, in the science of computational fluid dynamics. In the topics of hydraulics and sedimentation engineering I have learned from Prof. Dagfinn Lysne at Division of Hydraulic and Environmental Engineering at the Norwegian University of Science and Technology, and from Prof. Pierre Julien, Prof. Johannes Gessler, Prof. Ellen Wohl and Prof. Bogusz Bienkjewicz at Colorado State University. Torulf Tjomsland and Gosta Kjellberg from the Norwegian Institute of Water Research helped me with the biochemical models together with Glen George at the Institute of Freshwater Ecology, UK, Sally Heslop at University of Bristol and Richard Hedger at University of Edinburgh. Also thanks to Prof. Steve Chapra for his advice and excellent book on water quality modelling. Knut Alfredsen at the Norwegian University of Science and Technology helped me with software and hardware problems during the work with my dissertation, making the SSII model. Vijaya K. Singh at IBM Canada helped me with the C compiler. Dave Zenz and Suzy Deffeyes at IBM Visual Systems helped me with the OpenGL graphics. I would also like to thank the following people for helping me test the program: Morten Skoglund, Oscar Jimenez, Aslak Løvoll, Lars Abrahamsen, Siri Stokseth, J. Chandrashekhar, Knut Alfredsen, Hild Andreassen, Hilde Marie Kjellesvig, Md. Mahbubur Rahman, Tuva Cathrine Daae, Anne Sintic, Atle Harby, Amirul Islam Khan, Noor Quasim Khan, Chris Bowles, Catherine A. M. E. Wilson, Per-Ludvig Bjerke, Yaw Okyere, S. M. A. Azim, Ishfaq Ahmed, Josip Jugovic, Sebastian Palt, Thorsten Stosser, Richard Hedger, Roland Parrot, Koen Blanchaert, Istiarto Istiarto, Ahmed Siyam, Tom Bryant, Peter Borsanyi, Hans-Petter Fjeldstad, Chris Whitlow, Doug Booker, Mohammad Irfan, Pravin Raj Aryal, Michael Abebe Haile, Harsha Suriyaarachchi, Tim Fischer-Antze, Lars Jensen, Susanne Krüger, Sabine Sultzer, Felix Hermann, Beate Kohler, Faruk Bhuiyan, Philip Soar, Georg Premstaller, Michael Tritthart, Dania Huggins, Juan Carlos Atoche, Ricardo Mantilla, Nils Ruther, Ingerid Pegg, Oral Yagci, Sandor Baranya, Nasib Man Pradan, Jagadishwar Man Singh, Åsta Gurandsrud, Jerome Molliex, Morten Stickler, Jorn Wildhagen,

Rami Malki,  Aravind Kumar Agrawal, Juan Martin Viscardi, Robert Feurich, Nadine Kilsby, Hans Bihs, Peggy Zinke, Zafer Defne, Desislava Balzhieva, Dejana Djordjevic, Annette Schulte-Rentrop, Clemens Dorfmann, Mostafa Jalali, Ursula Stephan, Christine Sindelar, Christoph Ortner, Gudrun Hillebrand, Stefan Haun, Lisa Emilie Hoven, Marc Roberts, Irina Klassen, Gabriele Harb, Laura Nardi, Sigurd Løvfall, Markus Noack, Christian Svensson, Halvor Kjærås, Sandeep Bomminayuni, Qing Zhang, Yannick Baulig, Francisco Nunez-Gonzalez, Vahid Shoarinezhad, Dirk Schäfer and Subhojit Kadia. Also thanks to Richard Hibbert, David Seed, Richard May, Luca Barone, Isabelle Lavedrine, Norbert Jamot and Fabio Spaliviero at HR Wallingford Ltd., U.K. for their input and evaluation reports on SSIIM. Special thanks to Prof. Wolfgang Rodi, Prof. Gerhard Jirka, Prof. Roger Falconer, Prof. Thorsten Stosser, Dr. Catherine Wilson, Dr. Jan Wissink, Dr. Dominic von Terzi, Dr. Manuel Garcia-Villalba and Dr. Clemens Braun for advice on numerical algorithms and other assistance for the work during my sabbatical stay in Cardiff and Karlsruhe 2005/2006. I am also greatful for all the assistance from staff and fellow researchers at the Bundesanstalt fur Gewasserkunde in Koblenz, Germany, during my sabbaticals in 2012/2013 and 2019/2020: Dr. Stefan Vollmer, Dr. Gudrun Hillebrand, Dr. Irina Klassen, Marc Roberts, Dr. Marcus Noack and Christian Svensson.

Trondheim, 21. November 2023

Nils Reidar Boe Olsen

# Table of content

# Chapter 1. Introduction

## 1.1 Disclaimer and legal matters

I disclaim all warranties with regard to this software and the information in this document, whether expressed or implied, including without limitation, warranties of fitness and merchantability. In no event shall I or my employer be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortuous action, arising out of or in connection with the use or performance of this software. The program contains a number of bugs. It is not recommended that the program be used for solving a problem whose incorrect solution could lead to injury to a person, loss of property or economical losses. If you do use the program in such a manner, it is at your own risk. It is necessary to know that to understand and interpret the program results properly it is required that the user have knowledge and experience in computational fluid dynamics and hydraulic engineering.

If results from SSIIM are used in a scientific publication, references to earlier SSIIM work should be given so that the reader will understand that the SSIIM program has been used.

Provided the user complies with the above statements, the program can be used freely. The program can be distributed freely on condition that an unchanged copy of this manual is distributed with the program.

Nils Reidar B. Olsen

## 1.2 Limitations of the program and known bugs

Some of the limitations of the program are listed below.

* The program neglects non-orthogonal diffusive terms.
* The grid lines in the vertical direction have to be exactly vertical.
* Kinematic viscosity of the fluid is equivalent to water at 20 degrees Centigrade. This is hard-coded and can not be changed.

In computer science, a very well tested program still contains about one bug pr. 2000 lines of source code. The SSIIM programs contains over 100 000 lines of source code, and several modules have not been much tested. Also, combinations of modules may not have been tested at all. It is therefore likely that there are a number of bugs in the program. The user is advised to take this into consideration when evaluating the results of the program.

If the user publish results where the SSIIM model has been used, the user should include in the

publication a statement that says that the SSIIM model has been used.

Provided the user complies with the above statements, the program can be used freely. The program can be distributed freely on condition that this disclaimer and an unchanged copy of the User's Manual is distributed with the program.

Some problems are also described in more detail in Chapter 2.15.

## 1.3 Model purpose

SSIIM is an abbreviation for Sediment Simulation In Intakes with Multiblock option. The program is made for use in River/Environmental/Hydraulic/Sedimentation Engineering. Initially, the main motivation for creating the program was to simulate the sediment movements in general river/channel geometries. This has shown to be difficult to do in physical model studies for fine sediments. Later, the use of the program has been extended to other hydraulic engineering topics, for example spillway modelling, head loss in tunnels, stage-discharge relationships in rivers and turbidity currents. However, the main focus of the program is to model sediment transport in rivers, reservoirs and around hydrualic structures.

The main strength of SSIIM compared to general-purpose CFD programs is the capability of modelling sediment transport with moveable bed in a complex geometry. This includes a number of algorithms for different sediment processesizes, including sorting, bed load and suspended load, bed forms and effects of sloping beds. The latest modules for wetting and drying in the unstructured grid further enables complex geomorphological modelling.

Over the years, SSIIM has also been used for habitat studies in rivers, mainly for salmon. Free-flowing algae has also been modelled, as a part of extending the model for use in water quality engineering. The latest version also includes plastic particle modelling. However, the main focus of my research is on sediment transport.

The program is made for teaching and research purposes. It is not as well tested as commercial CFD programs, meaning it will have more bugs and be less reliable.

## 1.4 Model overview

The SSIIM program computes the water velocities and sediment transport in rivers, channels and reservoirs. The Navier-Stokes equations are solved with the k-e turbulence model on a three-dimensional almost general non-orthogonal grid. The velocities are used when solving the convection-diffusion equations for different sediment sizes. This gives trap efficiency and sediment deposition pattern. Bed changes over time can be computed, together with movement of the free water surface.

As with other CFD programs, SSIIM is divided into three parts: A pre-processor, a solver and a post-processor. The pre-processor includes tools to generate input data, including the computational grid. There is an interactive graphical *GridEditor* with elliptic and transfinite interpolation. The grid can be generated on the basis of measured geometry data.

The user interface of the program can present velocity vectors and scalar variables in a two-dimensional view of the three-dimensional grid, in plan view, a cross-section or a longitudinal profile. It is possible to export results to ParaView for post-processing. Multiple vtk files can be made during the computation, which can be used to make animations of the results.

In the new SSIIM 2 versions made by the wxWidgets, multiple windows can be opened simultaneously, showing for example a plan view of the geometry and a cross-section at the same time. Also, the grid can be edited while the program is running. Although care must be taken to do this.

The results can be viewed in the SSIIM 2 graphics during the solution of the equations. Also, print-out of result files can be done from the menu.

New users are recommended to read Chapter 2.1 which gives more details and advice. It is also recommended to try the tutorials described in Chapter 2.

**Post-processing**

The most convenient way to see the results is in the SSIIM 2 graphics. The grid can be seen in a plan view (Map graphics), longitudinal profile or cross-section. Unlike most other CFD program, the SSIIM 2 graphics shows the results simultaneously with the computation of the variables. Also, different variables can be chosen from the menu, for example velocity vectors, water depth etc. SSIIM 2 will also write the velocity field to a file called *result*, which can be read back to the program. Using SSIIM 2 without a user interface, the *result* file can be copied to a Windows computer and read by a SSIIM 2 version that has a user interface. For sediment computations, SSIIM 2 can produce a file called *bedres*, which contains the bed levels. This file can be read back by the program in a similar way as the result file.

More advanced post-processing packages are also supported: ParaView. The program can show 2D or 3D views of the grid and the variables. They can also be used for making animations. Input files for these two programs can be written from the user interface of the program or automatically for time-dependent computations. The ParaView program is freeware, and can be downloaded from the Internet. It is very easy to install.

Note that all SSIIM input files and results files are ASCII files. Files can be generated with a Linux version of SSIIM, and processed by a Windows version of ParaView. Or the other way around.

# Chapter 2. Startup using SSIIM 2

## 2.1 Advice for new users

Before you use SSIIM 2, it is necessary to have a minumum knowledge about hydraulics, fluid mechanics, sediment transport and numerical modelling. It is recommended that courses are taken in these subjects at university level. Then it is advisable to start with reading this manual.

A CFD computation generally consists of three steps:

1. Pre-processing
2. Computations
3. Post-processing

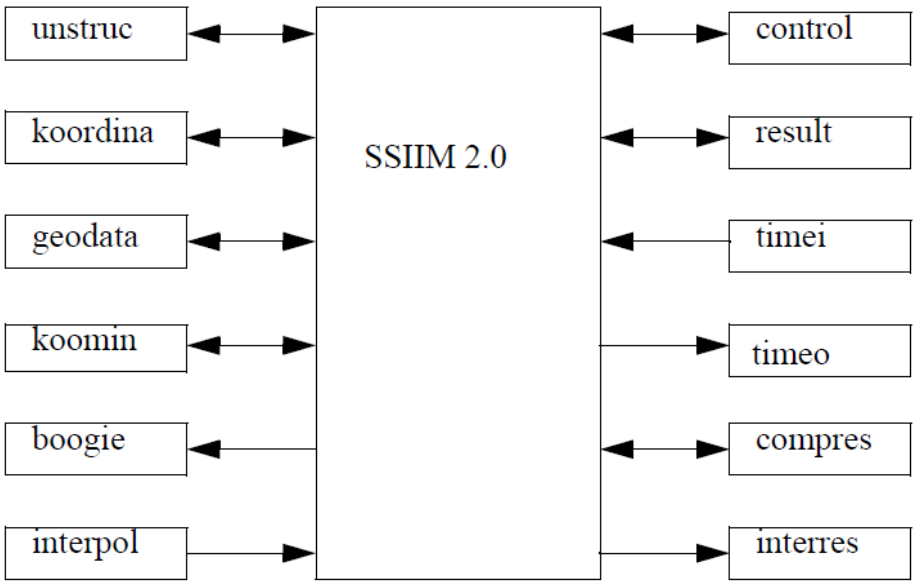
Pre-processing is generation of grid and input files. The SSIIM 2 model contains a graphical grid generator. Post-processing is viewing the results of the model. There are some graphics to do this in the SSIIM 2 model. But it is also possible to use the post-processing program ParaView. An introduction to the pre and post processing in SSIIM 2 is given by the tutorials in Chapter 2.3 to 2.7.

The computations itself can be calculations of water velocity calculations, sediment flow, bed level changes, water level changes and/or water quality. Each calculation is done by a module in SSIIM 2. Some of the modules can be started from the program menu. The different the calculations can be combined, for example the water flow, sediment flow and bed changes computations. This is further described in Chapter 2.2.

An advice for the first-time user is to run the tutorials described later in Chapter 2 and some of the example cases. Try to modify some of the parameters and run it again. Often the user wants to simulate a particular case. It is then advisable to try to find a similar example case and modify this step by step.

The next problem is then often computational speed and convergence. The user is recommended to read Chapter 2.12 for advice on these issues.

The SSIIM 2 program has many input and output files, some of which are shown in the figure to the right. One of the most important files is the *boogie* file, which is produced by SSIIM 2 and



contains error messages and other results. SSIIM 2 includes some controls for input and checking of intermediate results. If any of these controls finds something wrong, an error message is written to the *boogie* file, and the program terminates. Therefore, if the program suddenly stops, and the main user interface disappears, check the *boogie* file for possible error messages. Also, some warning messages may be written to the *boogie* file if particular problems occur. Chapter 2.16 gives more advice for finding bugs in the program.

The SSIIM 2 program is made up of a number of modules that reads files, makes grids, calculates unknown parameters, write result files etc. Some of the modules are started automatically, and some need to be started by the user. The user has two options on how to start the modules:

1. Use the menu in the graphics user interface
2. Specify which modules are to be run in an input file.

Option 1 is described in Chapter 4. Option 2 requires that a file called *control* exist in the directory the program is run from, and that this file contains a data set called F 2. The *control* file can contain a number of data sets, most often identified with a capital letter and a number. Then data is given after the identificators. The F 2 data set contains a number of capital letters. For each letter, a computing module of SSIIM 2 is started. See Chapter 5.3 for more details about the data sets in the *control* file.

Some of the SSIIM 2 versions do not have a user interface. Then option 2 is the only possibility to start modules of the program. The Windows versions of SSIIM 2 has user interfaces. It consist of a main window that shows text with output from the computations. At the top, there is a menu. The menu can be used to for example start computaitons, read/write files, change the view of the graphics or the parameters shown in the graphics. The first time user is recommended to start the program and try the different options to get to know the program.

The hardware requirements for running the program are sufficient amount of RAM in the computer and the speed of the computer. In the beginning of the boogie file it is printed how much RAM the program allocates for the arrays. This can be added to the RAM requirement for the program itself, about 4 MB, plus what the operating system requires. An estimate for the amount of RAM is thereby obtained. The harddisk is used as extra memory if there is not sufficient RAM. The penalty is that the program runs very much slower. This situation can be detected by observing if the system swaps to the harddisk while running only the SSIIM 2 program. The water flow module may take up to several days to converge for some cases, even when there is enough RAM.

When you have gotten results from SSIIM 2, you need to interpret these. You should then think about:

- Possibilities of bugs in the program making errors
- Previous cases where the results have been compared with measurements
- Numerical errors, like false diffusion, grid independence, etc.
- Accuracy of input data and boundary conditions

Knowledge and experience in computational fluid dynamics and hydraulic engineering are essential for the assessment of the validity and accuracy of the results. Chapter 2.13 gives further assistance for interpreting the results.

If you want to make a scientific publication with the SSIIM 2 results, you have to compare the results with measured values, from the field or the laboratory. This enables an assessment of how input values affect the final results. Without this test, the scientific quality of the work is most often not of sufficient quality to make an article publishable.

## 2.2 Making the grid

A CFD program computes the detailed flow field in a complex geometry by first dividing the water volume into cells. Then a set of equations are solved for each cell, producing a flow field or for example particle concentrations. The combination of all the cells is called a grid.

When SSIIM 2 is started, it automatically searches for the *control* file. If the file is found, it will use the data from this file. The file includes information about the maximum size of the grid. The data sets in the control file giving the size is the *G 1* and the *F 64* data sett. The grid and information about inflow/outflow of water is given in a file called *unstruc*. This file is not automatically read. The user needs to make the program read the file by using the menu or the *F 2* data set in the *control* file. The parameter used in the *F 2* data set is *U*, meaning the letters *F 2 U* has to be given in the *control* file. Of course, the grid has to be made first. This is described in the tutorials.

Also note that the *control* file is only read once: when the program starts. If the user changes something in the *control* file, the program has to be restarted for this change to have any effect.

The best grid is made of mostly hexaheadral cells. These cells have six sides, like a dice. The hex-grid will give better accuracy and stability than a grid made up of fewer or more sides. For modelling very complex geometries, it is not possible to us only hex cells. Then some cells with fewer sides are required. However, the number of such cells should be kept as small as possible.

SSIIM 2 includes a module called the *GridEditor*. It helps the user to make a grid using a graphics inteface. Grid lines can be moved with the mouse, and key coordinates can be given in a dialog box. Also, some graphical support for measured geometry data is given. The *GridEditor* is described in more detail in Chapter 3.2. Also, the tutorials in Chapter 6 includes information about how it is used.

Geometries that can alternatively be described with a formula, for example boundaries of straight lines or part of circles, can be made with a spreadsheet. The coordinates along the boundary are written to a file called *koosurf*, which is read by SSIIM. The procedure is described in Chapter 6.4, where flow in a 90 degree bend is computed.

In general, it is recommended to start a CFD progject computing a coarse grid and then make a finer grid afterwards. A fine grid may have very long computational times. Using a coarse grid first will enable the user to solve initial problems with short computational times. When the case is running well, a finer grid can be made.

## 2.3 Computing water velocities and turbulence

The water velocities are computed solving the Navier-Stokes equations for turbulent flow, using the k-ε model. A simpler turbulence model can be used. This is specified on the *F 24* data set of the *control* file.The SIMPLE method is used to compute the pressure field. The power-law scheme or the second-order upwind scheme is used in the discretization of the convective terms. This is determined by the values on the *K 6* data set in the *control* file.

The computation of the water velocities can be started from the menu of SSIIM 2 with the option *Compute->Waterflow*. Remember that the grid information has to be read in first. An alternative is to use a *W* on the *F 2* data set in the *control* file. Since the grid has to be read first, the data set will be *F 2 UW*.

If the user want to start the computation with a previously computed water flow field stored in the *result* file, this file can be read first. The *File* menu can be used, or an *R* can be used in the *F 2* data set. The data set will then be *F 2 URW*.

The Navier-Stokes equations are solved with an iterative method. Many times it may be difficult to get the solution to converge. If such problems occur, the first thing to do is to lower the relaxation coefficients on the *K 3* data set in the control file. A shorter time step (*F 33* data set) may also give a more stable solution. The cause of the instabilities may be a strange grid. For more information about solving stability problems, please see Chapter 4.2.

## 2.4 Post-processing (graphics output)

The most convenient way to see the results is in the SSIIM 2 graphics. The grid can be seen in a plan view, longitudinal profile or cross-section. Unlike most other CFD program, the SSIIM 2 graphics shows the results simultaneously with the computation of the variables. Also, different variables can be chosen from the menu, for example velocity vectors, water depth etc. SSIIM 2 can write the velocity field to a file called *result*, which can be read back to the program. Using SSIIM 2 without a user interface, the *result* file can be copied to a Windows computer and read by a SSIIM 2 version that has a user interface. For sediment computations, SSIIM 2 can produce a file called *bedres*, which contains the bed levels. This file can be read back by the program in a similar way as the result file. For time-dependent computations, a series of *result* and/or *bedre*s files can be written. The increment for when these times should be written is given on the *P 10* data set. The *alldata* file is a combination of grid, velocity, sediment concentrations and bed grain size distributions. This can also be written and read by the program.

A more advanced post-processing package is also supported: ParaView. The program can show 2D or 3D views of the grid and the variables. It can also be used for making animations. Input files for the program can be written from the user interface of the program or automatically for time-dependent computations. Use the *F 48* or the *F 329* data set to specify the type of file to be written. The ParaView program is freeware, and can be downloaded from the Internet. It is very easy to install.

Note that all SSIIM input files and results files are ASCII files. Files can be generated with a Linux version of SSIIM, and processed by a Windows version of ParaView. Or the other way around.

## 2.5 Using free surface algorithms

When solving the shallow-water equations, the free surface is found as a part of the solution. The solution of the Navier-Stokes equations does not automatically involve finding the location of the free water surface. The default algorithms in SSIIM 2 assumes a fixed water surface with no friction (zero gradient). Iterative methods are required to find the location of the free surface, if it is unknown. A number of such algorithms are implemented in SSIIM 2 (Olsen, 2015)

The default free surface in SSIIM 2 is flat and completely horizontal. The water level is given on the *W 1* data set in the *control* file. If the water level in the geomery is non-horizontal, but known, it is possible to give it in a spreadsheet, producing the *koosurf* file. This will be the most stable solution.

The default algorithm in SSIIM neglects the transient term. To include this in the calculations the *F 33* data set in the *control* file is used. The time step and number of inner iterations are given on this data set. For transient calculations it is possible to give the water levels and discharges as input time series. The *timei* file is then used. For further description of this file, see Chapter 5.14.

The gravity term is not included in the standard algorithms. It is only invoked in some of the free-surface calculations, for modeling spillways and flood waves with steep fronts. The *F 36* data set is used to include the gravity term.

The free surface is computed using a fixed-lid approach, with zero gradients for all variables. The location of the fixed lid and its movement as a function of time and the water flow field can be computed by one of four different algorithms:

1. Algebraic pressure and Bernoulli algorithm
2. Differential pressure and Bernoulli algorithm
3. Gravity and control volume algorithm
4. Diffusive wave equation

**Pressure and Bernoulli algorithms**

This algorithm can be used for both steady and unsteady computations. The algorithm is based on the computed pressure field. It uses the Bernoulli equation along the water surface to compute the water surface location, based on one fixed point in the grid that does not move. The location of this point is given on the *G 6* data set, both for steady and unsteady computations.

For a steady computation, the algorithm is invoked by using a small number for the second integer on the *K 1* data set. This integer gives the number of iterations between each time the water surface is updated. For an unsteady computation, *F 36 2* is specified in the *control* file, together with the time step on the F 33 data set. Using this algorithm, it is possible to use the *timei* file to compute a location of the water surface that varies over time.

The algorithm is fairly stable, so that it can also be used in connection with computation of sediment transport and bed changes. However, it will not work for higher Froude numbers.

The algorithms have been further developed, and the newest version, IPDA (Implicit Pressure Difference with Adaptive grid) is invoked with the *F 36 7* option. Further details are given by Olsen (2015).

**Diffusive wave equations (IDWA)**

This algorithm s used similar to the Pressure and Bernoulli algorithm, but the *F 36 9* is specified in the *control* file, together with *F 323 1*. The method is more stable than the pressure and Bernoulli algorithm, but probably not as accurate for complex water surfaces. The frition loss is computed from the Manning-Strickler value on the *W 1* data set, and not from the energy loss computed by the turbulence model. However, if the water surface is close to horizontal, which it usually is, then the accuracy may be of sufficient quality. Further details are given by Olsen (2015).

**Gravity and control volume algorithm (CGA)**

This algorithm is used to compute the movement of a free surface with steep gradients. It is invoked by using the F 36 1 option. The algorithm includes the gravity term in the Navier-Stokes equations. A time step has to be specified by the F 33 data set. The basis of the algorithm is to use the continuity equation instead of the SIMPLE algorithm to compute the changes in the water surface. The algorithm is very unstable, and a very short time step has to be used for stability reasons. This algorithm is only used for cases with very steep water surface gradients, for example computation of coefficient of discharge for a spillway or a flood wave with a steep front.

In 2009, this algorithm was improved a bit for the SSIIM 2 version, giving a more stable solution. The new algorithm is invoked by *F 36 15* and called CGA (Continuity and Gravity on an Adaptive grid).

## 2.6 Computing suspended sediment transport

Sediment transport is traditionally divided in bedload and suspended load. The suspended load is calculated with the convection-diffusion equation for the sediment concentration, *c* (volume fraction in SSIIM):

$$\frac{\partial c}{\partial t}+U_j\frac{\partial c}{\partial x_j}+w\frac{\partial c}{\partial x_3}=\frac{\partial}{\partial x_j}(\Gamma\frac{\partial c}{\partial x_j})+S \qquad (2.6.1)$$

The fall velocity of the sediment particles is denoted w. The diffusion coefficient, Γ, is computed from the eddy-viscosity, $\nu_T$, in the k-ε model:

$$\Gamma=\frac{\nu_T}{Sc} \qquad (2.6.2)$$

*Sc* is the Schmidt number, set to 1.0 as default. A different value can be given on the *F 12* data set in the *control* file.

*S* is a source term giving the pick-up rate of sediments due to erosion. The default formula is van Rijns (1984a) formula for suspended sediments. A number of other formulas can be used. This is chosen on the *F 84* data set. The empirical parameters van Rijns equation (0.015, 1.5 and 0.3) may be changed by using the *F 6* data set in the *control* file.

The procedure to take the sediment resuspension into account can be to specify a concentration in the bed cell, or use a pick-up rate as the source term in Eq. 2.6.1. The choice is specified on the *F 37* data set. *F 37 1* is using a specified concentration, and *F 37 2* is using a pick-up rate. The two options usually give very similar results.

The decrease in critical shear stress for the sediment particles as a function of the sloping can be bed can be taken into account by a number of formulas. They are given on the *F 182* data set. This process is particularly important when computing local scour (Bihs and Olsen, 2011)

A sediment computation requires information about the sediments on data sets in the *control* file. SSIIM 2 calculates sediment transport by size fractions. In the *control* file, each fraction is specified on an *S* data set, where the characteristic diameter and fall velocity are given. This data set must be present in the control file when calculating sediment transport. The number of sediment sizes is given on the *G 1* data set. The *S* data sets gives sediment size and fall velocity and the *I* data sets give inflow of sediments in kg/s. The *N* data sets gives information about different initial grain size distributions of the bed material and the *B* data sets tells where in the geometry the different distributions are. These data sets must be given with correct parameters.

The steady sediment flow is computed by the menu or the letter *S* in the *F 2* data set. An initialization of the sediment flow (giving values to the concentration in all cells based on the Hunter-Rouse distribution) is done by using the *I* letter on the *F 2* data set. The steady sediment flow will use a previously computed water flow field. In other words, the **water flow has to be computed first, and then the sediments**. On the *F 2* data set this will be *F 2 UWIS* for SSIIM 2. However, usually the water flow computation takes much longer than the sediment computation. Often, one might want to do several sediment computations with for example different transport formulas. Then one can do the water flow computation first, and store the flow field in the *result* file. Instead of recomputing the water flow field for each sediment computation, one can read the *result* file by giving an *R* on the *F 2* data set. This will then result in the following data sets: *F 2 URIS* for SSIIM 2.

*Note that it is also technically possible to <u>not</u> compute the water flow field before the sediment computation. This will of course give a completely wrong result, as the sediment computations are based on a known water flow field.*

The procedure described above do not compute any bed level changes. Information about the computations of bed level changes are given in Chapter 2.8 and 2.9.

For SSIIM 2, the inflowing sediment concentration should be given in the *timei* file.

Specification of initial sediment fractions on the bed is done by using *N* and *B* data sets in the *control*

file. The *N* data sets specify a number of sediment mixes. The distribution of the mixes in the various parts of the bed is given on the *B* data sets. In SSIIM 2, the initial spatial variation of the grain size distribution on the bed is given in the *fracres* file.

## 2.7 Computing concentration of plastic particles

A computation of plastic particles solves the same convection-diffusion equation as for normal suspended silt/sand sediments (Eq. 2.6.1). The main difference is that the plastic particles often have a positive rise velocity in the vertical direction. Also, there is most often no significant deposition on the bed, meaning no changes in the grid is needed. This makes the computation of plastic particles much less complicated than sedments with a sink velocity.

The input parameters for the plastic particles is the same as for the normal sediments. The boundary conditions at the water surface and at the bed needs to be modified, and also negative fall velocities has to be allowed. This is done by including an *F 417 1* data set in the *control* file. A negative fall velocity needs to be used on the S data set. Note that the solver is the same for the plastic particles and the normal sediments, so it is possible to compute both at the same time with multiple sediment sizes. The plastig partices will then have a negative number on the *S* data sets, and the normal sediments will have a positive value. As long as the plastic particles have a positive rise velocity. If the plastic particles are sinking, just like the sand/silt, then a positive value on the *S* data set is given.

Plastic partices with a positive rise velocity will often accumulate near the water surface. The vertical distribution of the particles will be affected by the turbulence. The free water surfac will dampen the turbulence, and this effect may be important. In SSIIM 2, the effect can be modelled by using the *F 185* data set in the *control* file.

## 2.8 Computing bed level changes

A time-dependent computation of water and sediment transport require more data sets in the *control* file. First, the same data sets describing the sediments as given in Chapter 2.6 has to be used. Also a time step has to be given, on the *F 33* data set. Then the *F 37* data set has to be used, typically with an integer 1, 2 or 3. If a free water surface is to be computed, the *F 36* data set also has to be used, with the integer 2, for example. The computation itself is started by the letter *S* on the *F 2* data set. The *F 2* data set might therefore be *F 2 URIS* for SSIIM 2. Starting with a previously computed flow field is optional, and so is the initialization of the sediment concentrations. The data set might therefore be *F 2 S* or *F 2 US*.

Any variation over time in the boundary condition of the water level, water discharge or sediment inflow must be given in the *timei* file.

The thickness of the bed sediment layer can be given on the F 41 data set, if it is uniform over the whole geometry. If the thickness varies, the *koomin* file must be used. The *koomin* file gives the location of the non-erodable bed.

## 2.9 Lateral grid movements

Cases with wetting and drying are complicated, but can be computed with SSIIM 2. The main principle is to make a depth-averaged 2D grid with SSIIM 2 that covers all areas, both wetted and dry. This grid is then stored in the *unstruc* file, which is used at startup of the computations. The *unstruc* file must not be overwritten later, when the grid shrinks. The file will contain information about the bed levels in areas that can dry up and later become wetted. If the water level is to be changed during the computation, the reference cell number on the *G 6* data set has to be taken from this initial grid.

Since the *unstruc* file has to cover all areas that can be wetted, the water level has to be high during the generation of the initial grid. If this is the physical situation we want to model, this is fine. For example, a drawdown of the water level in a reservoir. Algorithms has to be invoked that moves the water level down (*G 6* data set + *timei* file). However, sometimes the user want to start the computations with a lower water level. An example is the computation of the formation of a meandering channel. The initial channel will then move sideways in areas that were not wet at the start of the computation. To solve the problem, the following procedure can be used:

When the *unstruc* file is written, also a file called *koordina.t* is written automatically. This file is similar to the SSIIM 1 *koordina* file, but it contains both the bed and the water level. The water level is the last floating point on each line. The file can be edited with a spreadsheet and a user-specified water level can be given in this file. The file is then renamed *koordina* without extension, and placed in the same directory as the *unstruc* file. During startup, when SSIIM 2 reads the *unstruc* file, it will search for the *koordina* file and if it is found, it will automatically use the *koordina* water levels as initial values. The grid in the *unstruc* file must then be regenerated, which can be done automatically at startup by using an *F 112 1* data set in the *control* file. It is very important that a new *unstruc* file is not written from the program after this procedure, as the information about the initially dried up areas may then be lost.

The *control* file must contain the same information about the sediments as for the steady computations. Also, time-varying parameters must be given in the timei file.

A couple of more advice for these type of computations: Use the following data sets in the *control* file: *F 94* control the criteria for if a cell is wetted or dry, or if one or more cells are generated in the vertical direction. Use *F 201 1* to get smoother sides, *F 113 7* to avoid unphysical velocities in partially dry cells. Also, it is advisable to use the multi-grid solver to improve convergence. This is specified on the *F 168* and *K 5* data sets. If the program crashes, look at advice in Chapter 2.12. Chapter 2.11 gives more details about lateral grid movements.

This chapter describes the method to make a dynamic grid that can move in the lateral direction. Such a grid is used to model the wetting and drying process taking place in a meandering channel, or when modeling reservoir flushing.

Some details of the grid generation algorithms are described by Olsen (2003). The general principle is to first generate a 2D depth-averaged grid over all the areas where the river may flow. This is done by

using a water level that is higher than the bed everywhere. This 2D grid is then used as a basis for the 3D grid that is used to compute the water and sediment flow. The 2D grid stores information about the bed level location and sediment data. Also read Chapter 3.2 and 4.1, where more information is given.

The grid is stored in the *unstruc* file, which can be used when starting the program. This *unstruc* file can be generated in different ways, depending on the geometry. For a complex natural geometry, the *GridEditor* in SSIIM 2 can be used, with input from a *geodata* file. This is described previously in this manual. Note that a high water level has to be used, that covers both the wetted and dry areas. The second method to generate the *unstruc* file can be used if a regular geometry is to be modelled. Then a *koordina* file can be made in a spreadsheet, and this can be transformed to an *unstruc* file using the method described in Appendix I.

The *unstruc* file is usually very large. If this is a problem, then it is advisable to use as few cells in the vertical direction as possible when making the file. The number of cells in the vertical direction can then be increased later by adjusting the *G 1* and *G 3* data sets the *control* file.

The *unstruc* file has to contain the water discharges at the correct locations for inflow and outflow. This is specified in the *Discharge Editor*.

It is recommended to use the *F 102 1* data set in the *control* file to make the grid cells along the boundary change in shape to improve the smoothness of the banks.

When the *unstruc* file is made, its water level is often higher than what the user wants in the start of the computation. This is especially the case when modelling wetting/drying situations. To specify that a lower water level is used at the start of the computation, two steps have to be taken:

1. Make a *koordine* file containing the desired initial water level
2. Add the *F 112 1* data set to the *control* file

This *koordina* file has to have the same grid coordinates for the *x*, *y* and bed levels as in the initial *unstruc* file, but it also has to contain the water levels. A recommended procedure is to use the file *koordina.t*, which is written automatically when the initial *unstruc* file is written from SSIIM 2. The *koordina.t* file has the correct values for *x*, *y* and the *bed levels*. However, the water level is the same as in the *unstruc* file. The *koordina.t* file can be imported into a spreadsheet and the values for the water level can be changed to the desired initial values. Then it can be written to the working directory and named *koordina*, without an extension.

Note that the *F 112 1* option in the control file only works with the original *unstruc* file and the modified *koordina* file. If later during the computation a new unstruc file is written from the menu, then this will not be compatible with the modified *koordina* file. Writing a new *unstruc* file during the computation will overwrite the original *unstruc* file. In other words, keep a safe copy of the original *unstruc* file and the modified *koordina* file. To reduce the occurrence of this common mistake, the *unstruc* file will get an extension .dry if it is written when some part of the geometry has dried up.

The lateral movement of the grid is due to fluctuations in the bed and water levels. This means that one or both the *F 36* and *F 37* data sets have to be used in the *control* file. Also, the *timei* file has to be used, where time-variations in water level, water discharge and sediment inflow can be given.

As the computation proceeds over time, intermittent results may often be required. This can be done by using the *P 10* option in the *control* file. A number of *bedres* and *result* files are produced, which can be read by SSIIM 2 at a later time after the program has finished. This is done by removing the extensions of the files at the selected iteration and reading the results from the menu of SSIIM 2. Using the *F 389* data set in the *control* file, similar writing of the *alldata* files is invoked.

## 2.10 Computing water quality parameters

The water quality algorithms in SSIIM 2 was made in 1998 and has not been used or tested afterwards. The risk of bugs or other problems is therefore high. The text below is from the Users Manual made in 1998.

The water quality is calculated with a convection-diffusion equation for the concentration, $c$, of each water quality constituent. The difference from computing sediment concentration is the addition of extra source and sink terms, due to fluxes at the water surface and chemical and biological reactions. The source (sink=negative source) terms in the equations have been modelled in a number of different ways by various researchers and computer programs. A goal in the implementation of water quality in SSIIM was to give the user a flexibility on how to model each term. The user must therefore not only specify various reaction constants, but also specify each term in the source equation. This involves some more work than other models when making the input files, but it gives better flexibility for the user to make new terms and models for the various situations.

The user first have to decide how many constituents are to be simulated. The user have to number each variable from 0 to as many variables as is used. Presently, the maximum number of variables are 20. The number of variables have to be given on the *F 50* data set. The names of each variable is given in the *control* file on the *Q 0* data set. Note that only lower case letters have to be used, and a maximum of 40 characters. For each equation, the user have to specify the source terms in the equations. This is done on the *Q* data set in the *control* file. There are a number of various *Q* data sets, from *Q 1, Q 2* ... etc. The various data sets are described in Chapter 5.3.9. Each data set gives a number of integers and floats. The first integer is always the index for the equation.

The temperature is defined as a water quality constituent. If the temperature influence on the water flow is to be modeled, as in stratified flow, the temperature has to be variable no. 0. Then also the data sets *F 40*, *F 62* and *F 67* may have to be used. This is only implemented for SSIIM 2.

Each source term in a convection-diffusion equation for a water quality parameter has its own *Q* data set. For example, if temperature, oxygen and nitrogen is simulated, this gives three variables. The following *Q 0* data sets can then be used:

*Q 0 0* temperature
*Q 0 1* oxygen
*Q 0 2* nitrogen

The source terms then have to be specified. For simplicity, let us say that the temperature source is 0.1 times the oxygen concentration and the oxygen source is 0.3 times the temperature. The nitrogen source is 2.1 times the oxygen concentration multiplied with the temperature minus 0.4 times the nitrogen concentration. This is then given as:

*Q 1002 0 1.0 18.0*
*Q 1 2 1 0.3*
*Q 2 3 1 2 2.1*
*Q 1 3 2 -0.4*

No more than 40 variables can be simulated, and it is not allowed to have more than 200 *Q* data sets in the *control* file.

The above example shows the flexibility of the program to incorporate new models for the various reactions. It also shows that the user is able to make models that make little sense from a biological/chemical point of view, and the program will allow this calculation to be carried out. The user must therefore have a good knowledge of water quality processes to give reasonable input data and to obtain reasonable results. Care must also be taken not to mistype numbers on the *Q* data sets.

**Initial values**

The initial values for water quality parameters when the calculation starts is given on the *G 41* data set. The data set specifies a vertical distribution of a variable. Horizontal homogeneity is then assumed for the initial values. For the temperature, it is also possible to use the *G 40* data set instead.

**Inflow of water quality constituents**

To have an inflow of nutrients or algae, the following is done:

In the discharge editor, a water discharge is given in different groups. Up to nine groups can be used. The water discharge and the group index is stored in the *unstruc* file.

The inflow concentration of nutrients is coded in the *control* file, on the *M* data set. The data set takes two integers and one float. The first integer is the same integer used to index the water discharge group in the unstruc file. The second integer is the number of the water quality constituent, similar to the *Q* data set. The float is the concentration of the nutrient. The data set can also be used to specify inflow of algae.

It is possible to specify a number of M data sets, according to the number of inflow groups and water quality constituents. If a nutrient has zero concentration at the inflow, it is not necessary to specify an M data set for it. It is not necessary to specify M data sets for the outflow.

**Time-dependent variation in input parameters**

To specify time-dependent inflow of nutrients, the timei file has to be used. Also, it is necessary to specify an *M* data set in the *control* file for each of the varying inflow concentrations. But instead of using the *I* or *J* data set in the *timei* file, an *M* data set is used. On this data set, the time is given first,

similar to the *I* data set. Then a float is given, which is a variable time step. The variable time step is not coded yet, but it has but be given in the data set. Then the three floats for the wind is given, if wind is specified in the *control* file. Then up to 20 inflow concentrations are given. Note that the depth and water discharges from the I data set is not given. The number of inflow concentrations is the same as the number of *M* data sets in the control file. Irradiance read as the last number of the line, if specified by the *Q* data sets in the control file.

Note that the order of the *M* data sets in the control file is not important, but the order of the data on the *M* data in the *timei* file has to be the same as the order of the *M* data sets in the *control* file.

**Example:**

***control*** file:

..

*Q 0 0 cyanobacteria*
*Q 0 1 phosphorous*
*Q 0 2 nitrogen*
*Q 0 3 algae_density*
*Q 0 4 light_history*

*fall velocity for algae:*
*algae_diameter form_resistance temperature*
*Q 131 0 3 0.0001 1.0 20.0*

*algae density:*
*a b c1 c2 c3 ks_irradiance min max*
*Q 282 3 0 4 0.0086 0.69 0.0022 0.00000028 0.0003833 25.0 0.7 1.3*

*light history:*
*a b averaging_period*
*Q 132 4 0 0.0086 0.69 86400.0*

*algae growth:*
*a b c1 c2 growth_phos+nit ks_phos ks_nit temp_const*
*Q 281 1 2 3 0.0086 0.69 1.18 1.0 0.82 0.0009 0.042 1.106*

*phosphorous depletion:*
*a b c1 c2 growth_phos+nit ks_phos ks_nit temp_const nutrient_fraction*
*Q 291 1 0 2 0.0086 0.69 1.18 1.0 0.82 0.0009 0.042 1.106 0.02*

*nitrogen depletion:*
*a b c1 c2 growth_phos+nit ks_nit ks_phos temp_const nutrient_fraction*
*Q 291 2 0 1 0.0086 0.69 1.18 1.0 0.82 0.0042 0.0009 1.106 0.14*

*q 42 0 0.0086 0.69 1.18 1.0 concentration, kc_(attenuation), c0, c1*

*M 1 0 0.001 inflow concentration of cyanobacteria*
*M 1 1 0.0003 inflow concentration of phosphorous*
*M 1 2 0.002 inflow concentration of nitrogen*

***timei*** file:

time timestep windspeed winddir_x winddir_y inflow0 inflow1 inflow2 irradiance

*M 0.0 0.0 1.0 1.0 0.0 0.001 0.0003 0.002 0.0*
*M 1000.0 0.0 1.0 1.0 0.0 0.001 0.0003 0.002 10.0*

**Summary of special *Q* data sets:**

The table below summarizes special *Q* data sets, including their creation date.

*Q 1102*: oxygen reaeration, 18/7-98
*Q 1060*: surface temperature flux, 17/7-98
*Q 2030*: resuspensjon, 17/7-98
*Q 2149*: SOD 30/7-98
*Q 11*: general fall velocity
*Q 42*: algae growth, Loch Leven, diatoms, 3/4-98
*Q 113*: time history of irradiance, 20/7-98
*Q 120*: WQRRS data set: algae
*Q 121*: predatation 17/7-98
*Q 122*: Pathogen, light, 20/7-98
*Q 123*: sorption fall velocity 17/7-98
*Q 131*: algae fall velocity, 20/6-98, const. temperature
*Q 132*: light history, set, 8/7-98
*Q 133*: light history, passive scalar, 10/7-98
*Q 151*: specific algae density, SCUM, one light att. coeff.
*Q 152*: algae growth, Leven, diatoms, 3/2-98
*Q 161*: specific algae density, SCUM, two param. light att., 30/6-98
*Q 162*: algae growth, Leven, diatoms, 3/2-98
*Q 221*: algae fall velocity, SCUM, 30/6-98, variable temperature
*Q 230*: WQRRS data set: algae
*Q 252*: algae growth, 20/7-98, multiple algae, one nutrient
*Q 261*: nutrient depletion 20/7-98
*Q 272*: diatom growth, 26/2-98, diatoms, silica, .
*Q 281*: algae growth, cyanobacteria, 25/6-98
*Q 282*: specific algae density, SCUM, 8/7-98, light history
*Q 291*: nutrient depletion, 25/6-98, one algae, two nutrients
*Q 361*: algae growth, cyanobacteria, 16/7-98, multiple algae, two nutrients
*Q 371*: nutrient depletion, 16/7-98, multiple algae, two nutrients

**Modelling free-flowing algae**

Free-flowing algae has two characteristics which needs special care in modelling. The first characteristic is its growth process. This is often both dependent on light and nutrients. The nutrients are often phosphorous, but nitrogen and silica may also be important for some species and situations.

The other main algae modelling characteristic is its fall/rise velocity. This is due to changes in its buoyancy. The buoyancy is often dependent on other parameters, for example irradiance.

A number of special data sets are made for different species of algae. Some of these can be used in combination with each other. Also, they can be used in combination with the other water quality data sets.

Two different groups of data sets are used, depending on whether there are multiple species of algae or only one single species contributing to the light shading. If there is only one species of algae, the light shading parameters are give in the growth data set. These types of data sets are described first.

Calculation of spatial distribution of algae includes modelling of the algal vertical rise/fall velocity. The rise/fall velocity is dependent on the algal density, algal diameter and a form factor. The algal density is dependent on the irradiance and the light extinction in the water body. Because the algal density depends on the values of the previous time step, it has to be modelled as a separate parameter. This means that modelling algae concentration necessitates an extra variable.

Some special *Q* data sets can be used to model algae settling and production. For example, *Q 151* is used to calculate the density as a function of the irradiance, *I,* light extinction coefficient, *k*, and a number of coefficients in the density formula. The time-dependent irradiance is given in the *timei* file. The amount of irradiance, *I*, decrease with the factor *f*, as a function of the water depth and the algae concentration. A formula for f is given by Olsen et al (2000), eq. 8. The computation of the change in the algae density is also described by Olsen et al (2000), together with the determination of the rise/fall velocity of the algae.

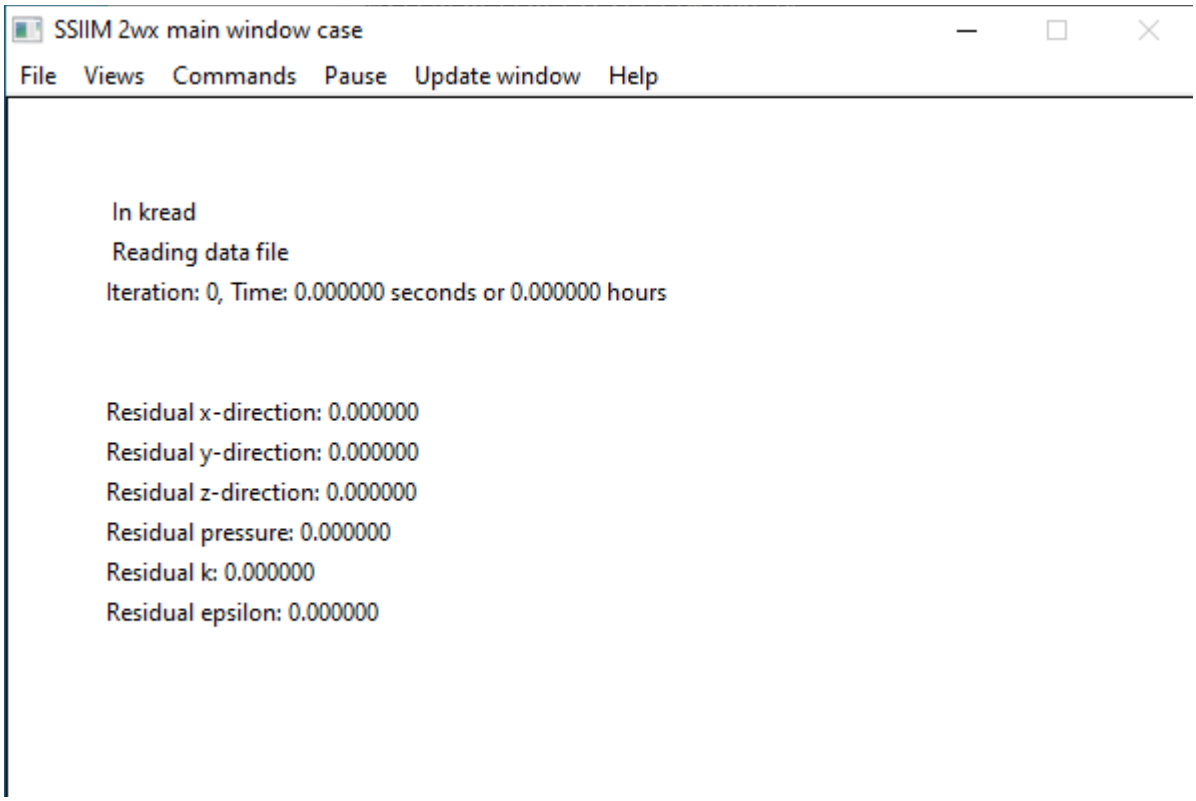
The *Q 42* data set models algal production. The computation of the algae growth rate is described by Olsen et al (2000), Eq. 10 and 11. The two last parameters on the *Q 42* data set are the a and b coefficients in Eq. 10 (Olsen et al, 2000).

# Chapter 3. User interface

There are in principle two versions of SSIIM 2 - one with a graphical user interface and one without. The current chapter describes the version with the user interface. The file name of this version includes the letters "wx", as the C++ graphics library wxWidgets has been used.

## 3.1 The main user interface

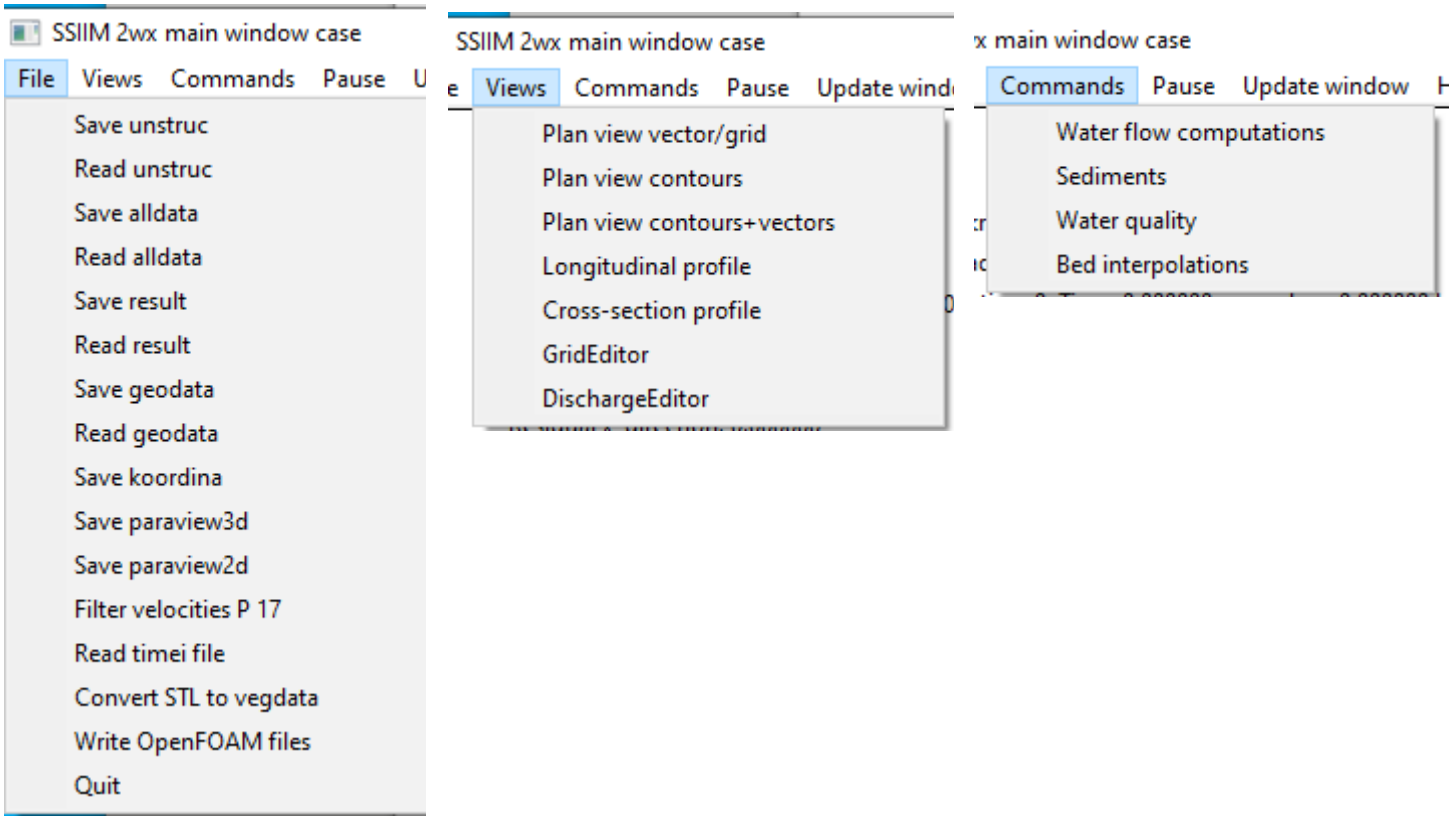
The main user interface appears once the program is started. It consist of a window with some text inside and it also has a menu. The menu options can be used to start computations, write files or show the results in separate graphics windows.




There are some text lines in the main window that give information about residuals for the iterative solutions of hte Navier-Stokes equations with a turbulence model. There is also some information about which computational module is running and other relevant information.

The menu has six choises. The File option invokes functions that reads or writes different types of files. The files are described in detail in Chapter 5. The unstruc file contains the main grid and the inflow/outflow regions. The *geodata* file contains measured points of the river bed with (x,y,z) coordinates. The paraview files contain graphics that can be read by the paraView program, which is a

powerful visualization program that is freeware. The option "Convert STL to vegdata" is described in Chapter 4.10. And the OpenFOAM files are described in Chapter 4.11.




The next option in the main menu is "Views". Choosing one of these options opens a new window on the screen. This is different from the old version of SSIIM 2, where there was only one window. The plan view windows shows the grid from above. These windows were earlier called "Map". There are also two windows showing a longitudinal and cross-sectional profile. Note that multiple windows of the same type can be opened at the same time.

The *GridEditor* is a graphical tool that helps make the computational grid. It is further described in Chapter 3.2. The *DischargeEditor* is also a graphical program. It helps the user to specify locations inflow/outflow in the grid. It is described in Chapter 3.4.

The "Commands" menu option can be used to start a computation of water flow velocities, sediments or water quality. It can also be used to interpolated the bed levels in the grid, from the *geodata* file.

The "Pause" option is used for debugging. It stops the computation at specified points in the program, so it is possible to see how the variables look at this particular location. This is then seen in the graphics menu. Or it can be written to the paraView files.

The "Update window" option can be used to refresh the text in the main window. New in the SSIIM wx version is that a timer can be started, updating the window automatically at certain time intervals. The intervals are chosen in the sub-menu.

The "Help" menu is not implemented. It is recommended to look in this Users Manual instead.

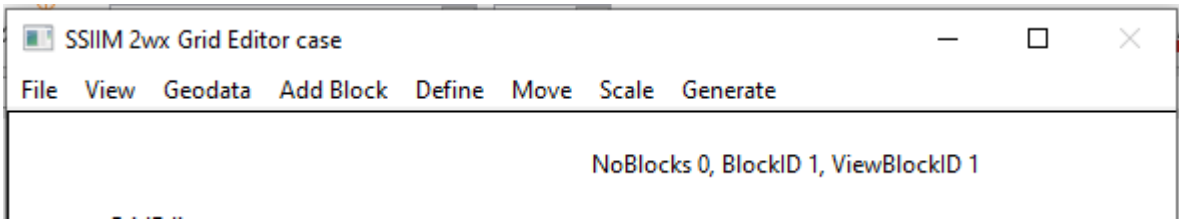
## 3.2 The *GridEditor*

The *GridEditor* is invoked from the *View* option in the main menu. The grid is seen from above, in plan view.

The types of modelled geometries can basically be divided in two: Man-made channels and flumes have fairly regular geometries. The other type is the more complex natural rivers and lakes. A natrual geometry usually has scanned data from a laser scanner or other type of instruments/maps.

### 3.3.1 The menu

The main menu of the *GridEditor* is made up of several options with corresponding sub-menus. The main menu is given in the figure below.




***File***

The File option in the menu only has two sub-options: Quit the GridEditor and write fixedpoints. The fixedpoints are grid line intersections that do not move. This is explained further below. The fixedpoints are written to a file and can be copied into the control file so that they are stored in a restart. Otherwise they are lost when the program stops.

***View***

Very fine grid will take a long time to display in the window. The default is therefore only to show the outline of the grid. But if the user wants to display all the grid cells or indexes/cell numbers, this can be done in this menu. Also, the user can choose which block is displayed. Or all blocks. SSIIM 2 can have many structured blocks that are joined together. The method to make more blocks is described in Chapter 6.2.

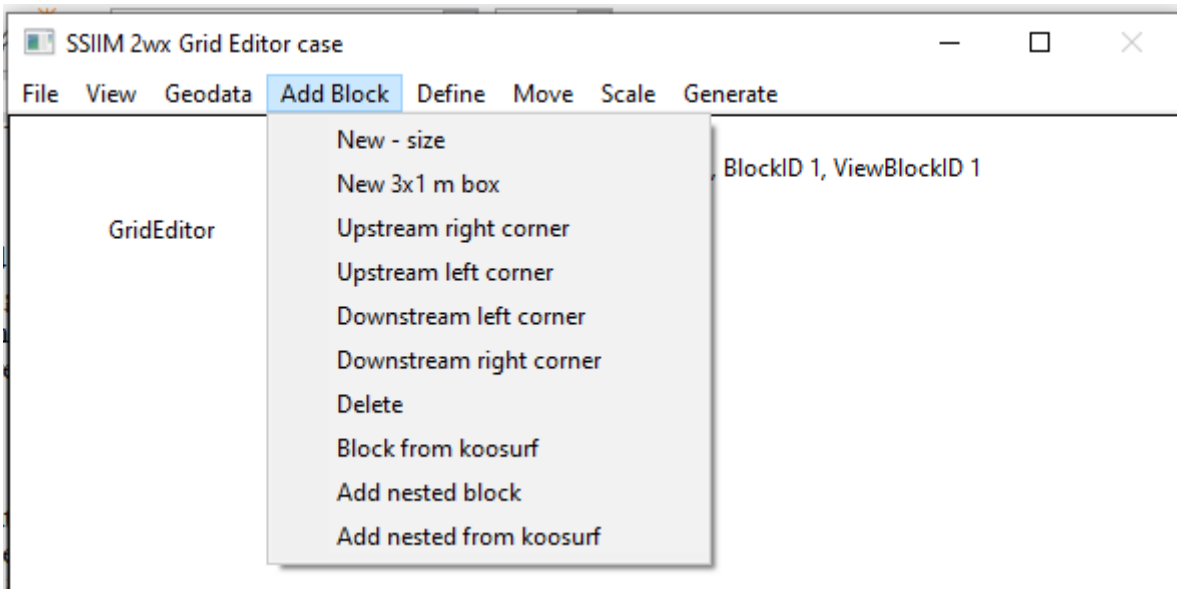
***Geodata***

A grid of a natural river often has a complex shape. When making the grid graphically, some of this information needs to be displayed in the window. The *geodata* file contains a large number of

measured (x,y,z) points in the geometry. They can be displayed by invoking options in this menu. If there are many million geodata points, it can take a long time to display all in the GridEditor window. A solution is to only display each 10th or 100th point. This is an option in the menu.

The geodata points are given different colour depending on their vertical level. The colour scale can be modified in the menu. If a large number of geodata points are displayed, the GridEditor window will have a colour map that can be used in reports etc. The colour scaling is specified in *Scale* menu option.

***Add Block***



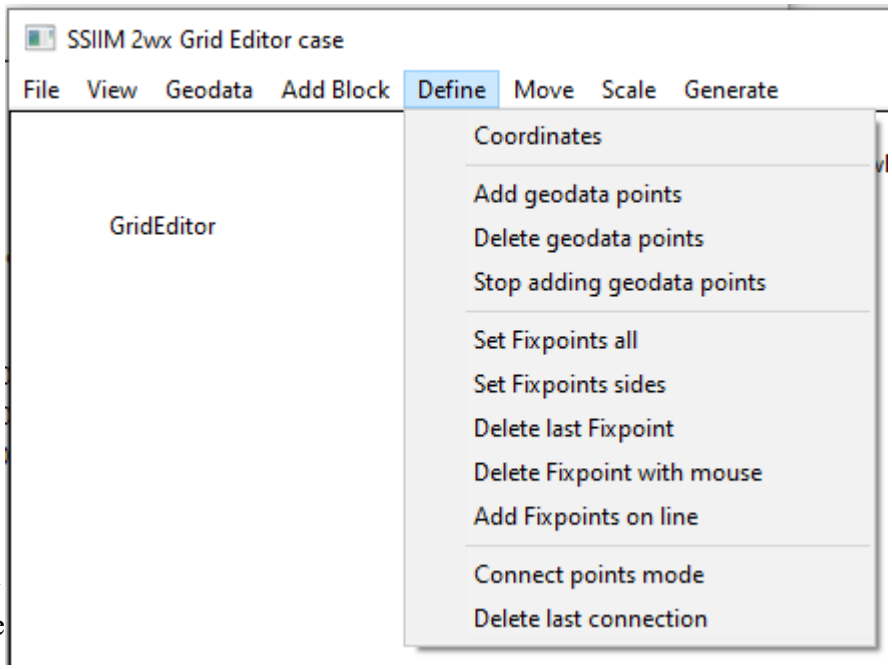

The *Add Block* menu option is used to make grid blocks of the geometry. The blocks are structured grids with a given number of cross-sections and points in the cross-section. These two numbers are given from the menu suboption New - size. Also, the water level location is given. After these numbers are given, a default orthogonal grid is made. The user can then define the four corners of the block. This can be done graphically with the submenu options that include the word "corner". The tutorial in Chapter 6.3 describes this in more detail.

An alternative to this procedur is to add a block from the *koosurf* file. Then the *koosurf* file has to exist in the working directory. The size of the block will then be taken from the *koosurf* file. The "*Delete*" option will delete the last added block.

Blocks can also be nested. The last two meny options give the size for a nested block, or a nested block from a koosurf file. This is further described in Chapter 4.7.

***Define***

This option is used for defining different parameters. The parameters are often connected to grid intersection points. The point that was last activated by the mouse is used as default.




The first option is *Coordinates*. This gives a dialog box where the user can give numerical values for *x,y* and *z* for a grid intersection. Note that the z value is for the bed.

The second option is *Set FixedPoint*. This invokes a mode where the user can define certain points which will not be moved by the interpolation, called *FixedPoints*. In *Set FixedPoint* mode it is not possible to move the grid points with the mouse. When the user clicks on a grid intersection, a blue star emerges on the intersection as a sign that this is chosen. Up to 19 999 *FixedPoints* can be chosen. To verify that this mode is present, the letters "Point mode, 0" is shown on the lower part of the EditWindow when the *Set FixedPoint* option is chosen. The integer shows how many points you have chosen. To return to the normal mode, choose *Define* and *Set FixedPoint again*. It is verified that the normal mode is set because the text "Point mode" disappears. In the normal mode the user can move all points including the *FixedPoints*.

Add *FixedPoints* on a line will specify a number of FixedPoints between the two last generated *FixedPoints*.

The Connect points mode is used to connect two points in two different grids. This procedure is used to "glue" two blocks together. This is showed in Chapter 3.3.2.

***Move/Scale***

The option Move is used to move the plot upwards, downwards or sideways. The arrow keys can be used instead of the menu. The option Scale is used to enlarge, shrink or distort the plot. The keys <Page Up> and <Page Down> can be used for scaling. By holding down the <ALT> button while scaling/moving, then the changes are smaller.

***Generate***

The first choice in the pull-down menu is *Boundary*. This choice interpolates linearly along the four border lines of the grid. Note that the *z* values of the bed are also interpolated. This will create a rectangle unless a *fixedpoint* has been defined on the border. Then the interpolation will be between the corners and the *fixedpoints*.

The second choice is *Elliptic*. This starts the elliptic grid generator. Note that this will not change the bed levels.

The third choice is TransfiniteI. This is transfinite interpolation in the streamwise direction. The z values will be interpolated. IMPORTANT: In this mode the *fixedpoints* will also be moved.

The fourth choice is *TransfiniteJ*. This is the same as *TransfiniteI*, except it is in the lateral direction.

The second last choice, Bed levels/Bed interpolations, generates *z* values for the bed surface of the grid. The *z* values are interpolated from a set of geometrical data read from the *geodata* file. If there is no *geodata* file present, an error message is given. The interpolation routine goes through all the grid points (*i,j*), and finds the closest points in the *geodata* file in all four quadrants where the grid intersection (*i,j*) is the centre of the coordinate system. Then a linear interpolation from these four points is made. If one of the points in the *geodata* file is closer than 5 cm from the grid point, this *z* value is chosen and no interpolation is done. The outcome of the interpolation is logged to the file boogie.bed. If the interpolation routine is unsuccessful in finding the point, the *z* value is set to zero.

The last choise, *3D Grid/Implementation*, is used after having changed the *z* values on any of the coordinates. The *GridEditor* only moves the grid in the layer bordering the bed. So if any of the grid points have been moved in the vertical direction, the water level and the grid points above the bed needs to be recalculated. The new grid is computed by using the *3D Grid* option.

### 3.3.2 Grid generation for SSIIM 2

**Moing grid intersection points graphically**

In the *GridEditor*, it is possible to move individual grid line itersections with the mouse. In the old ssiim2wn version this was done by clicking on an intersection with the left mouse button, and sliding it to the new location. In the new ssiim2wx version, this is done by first clicking on the point with the left mouse button and then clicking on the new location with the right mouse button pressed. This has the advantage that if you are not happy with the new location, you can just click on a better location with the right mouse button clicked, without first re-clicking on the orginal position.

Also note that if you want to specify an exact coordinate position with numbers, you can click on the *Define->Coordinate* menu option and give in the numbes in the dialog box.

**2D and 3D grids**

SSIIM 2 uses two grids: a two-dimensional depth-averaged structured grid and another three-dimensional unstructured grid. The two grids cover the same area. The 2D grid is fixed in space over time and contain both the wetted and dry cells. The indexing of the 2D grid is similar to SSIIM 1. The 3D grid may change over time as the water and bed elevations change. The total number of cells in the 3D grid will therefore also change, and also the indexing of the cells.

For a real-world case where a complex water body is simulated, it is necessary to start with a *geodata* file, giving the coordinates of for example contour lines of the bed levels of the lake.
The grid generation procedure then starts with reading this file from the *GridEditor*. The individual points will then be shown in the main window, with different colours according to the vertical elevation.

It may not be necessary to use a *geodata* file for demonstration and teaching purposes, or if the geometry is fairly simple, for example a channel.

The next step is to start making a number of blocks in the *GridEditor*. Each block is made up of a structured grid, which is 2D depth-averaged. The blocks are glued together to form an unstructured grid, covering the whole water body. The blocks can have different number of cells, both related to other blocks and the two horizontal directions.

**Wetting/drying**

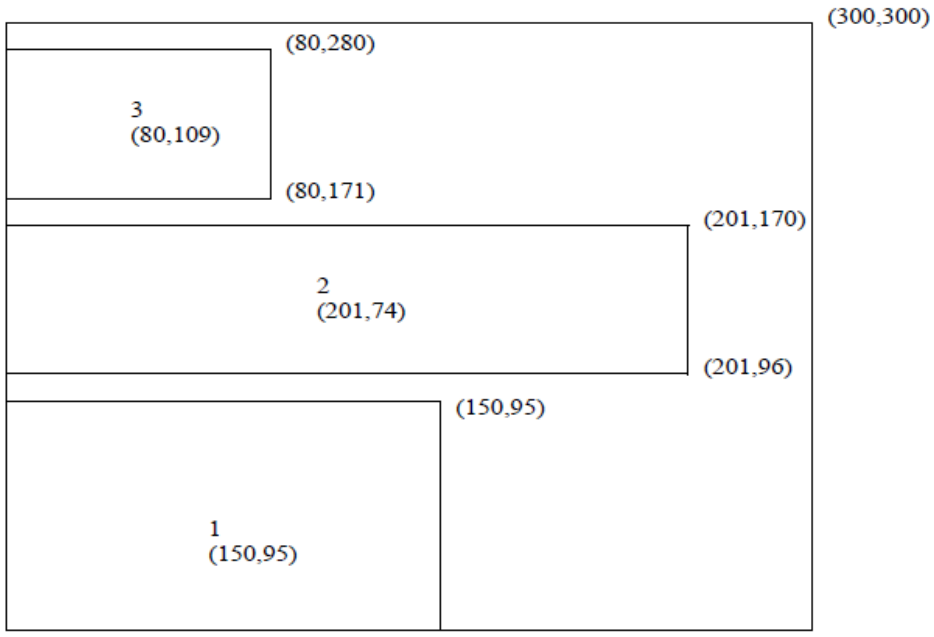
With the most recent wetting/drying algorithm, it may be most convenient to make a single 2D block of the geometry. The wetting/drying algorithm can then be used to make the complex boundary.

**Final 3D grid**

When the plan view of the grid looks ok., the 3D grid can be generated. This grid is based on the 3D grid, but has a varying number of grid cells in the vertical direction. The 3D grid is generated in the *GridEditor*, from the menu options *Generate -> 3D grid*. Afterwards, it is advisable to save the grid from the main SSIIM menu, by writing to the *unstruc* file.

**Multiple blocks in SSIIM 2**

SSIIM 2 has the option of using multiple blocks. The blocks can be glued together. Each block is stored in one large block. This is seen in the figure to the right. The large block has by default 300 x 300 cells dimension. This can be changed with specifying different parameters on the *G 1* data set.



The figure above shows the *i*-direction (along the main flow direction) to the left and *j*-direction (lateral) upwards. Let us say that the first block has 150 cells in the *i*-direction and 95 in the *j*-direction. Then the second block has 74 cells in the *j*-direction. This means that the second block will be placed at *j*-values between 96 and 170 in the initial main block. It also means that there is only 300-170 = 130 more *j*-spaces in the main block. If many blocks are created, they may fill up the main block in the *j*-direction. This needs to be kept in mind, and the *G 1* data set must be adjusted accordingly.

**Details of grid generation**

For real-world cases, it is important to make a hand-drawn sketch of the geometry and the grid blocks before making the grid on the PC, to determine the approximate size and number of blocks. Note the limit of number of blocks is currently 19, but the graphics is not made for that many blocks. It is possible to add blocks to an already existing grid, so usually it is advisable to start with a simpler grid with a couple of blocks covering the main part of the water body, and add more detail afterwards. Also note that it is not possible to remove already added blocks, so it may be advisable to save files with the simplified grid.

The blocks are added by choosing the *Add block* in the *Block* menu. After making the choice, the user clicks with the mouse on the four corners of the block. The four points must be added in the clockwise direction, starting with the south-west corner, then the north-west corner, then the north-east corner and finally the south-east corner. Note that the directions north-south-east-west does not have to follow the true directions according to the map, but it has to be consistent relative to the rest of the grid. After the four clicks are made, a rectangle is seen on the screen. The user then have to choose the number of grid cells in the block. This is done by choosing the *Size block* in the *Block* menu, and giving the numbers in the dialog box. The grid lines in the block then emerge. The user can now move the grid lines of the block to better fit the boundary or other characteristics of the desired grid.

A new block is added by the repeating the same procedure, starting with *Add block*. Up to 19 blocks can be used.

The next step is to glue blocks together. The connection is actually made in the three-dimensional generation procedure. The algorithm checks if two grid points are very close to each other. If two grid lines at borders of two grids are very close, then the connection is made. Note that the ends of the grid lines also have to be located at the same place, so that only one cell from one block connects to one cell at another block. It is not possible to have two cells from one block connecting to one cell in another block. Then the connection is not made, and a wall is used instead.

Before the 2nd block is made, one should think about the orientation of the blocks. The new block will similarly to the first block have a south/west/east/north side. When the blocks are glued together, the north side of one block should be glued to a south side of the other block, for example. Or the east side of one block should be glued to the west side of another block. Thereby a consistency of the direction of the total grid is maintained. All the blocks should have the same north/east/west/south direction.

To make the connection it is therefore necessary to locate grid corners from two blocks at the same place. This is done by choosing *Connect points mode* in the *Block* menu. In this mode, it is possible to click on one grid intersection with the left mouse button and then with the right mouse button click on the grid intersection in another block. The two grid intersections are then located at the same place.

When this is done, the point is locked, and will not move later, similar to a *fixedpoint*. This is seen in Figure 4.3.2.1. Fig. A shows two blocks that are to be connected. Fig. B shows the connection of two border grid points

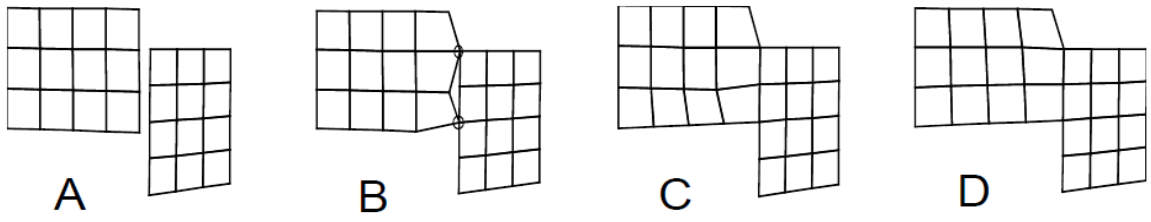
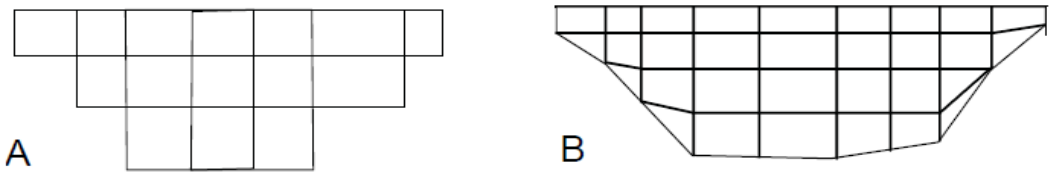



**Fig. 3.3.2.1 Procedure to glue blocks together.**

It is not necessary to connect all the points along a connection, only the end points. Then select the block which has non-stright line between the two connected points. Use the *Generate -> Boundary* from the menu make this line straight. Then the intermediate points on the connection line will be located at the same place. This is done in Fig. 4.3.2.1 C above. Note that it is only possible to edit one block at a time. The user choose which block to edit by the menu option *Block - Choose no*.

After the blocks are made and glued together, it is usually necessary to modify each block boundary according to the points in the *geodata* file, and also possibly to generate the internal points in the block with the elliptic or transfinite grid generator. This is shown in Fig D above. The user can click with the left mouse button on a grid intersection and move it to a new location with the right mouse button.

When the user is satisfied with the two-dimensional grid, the grid has to be made in the third direction. This is done in the menu option *Generate -> 3D grid*. It is possible to have varying numbers of grid cells in the vertical direction. There are many algorithms how this can be done. The figure below shows two possibilities. A stepped grid with horizontal grid lines can be used, or using non-horizontal grid lines including the possibility of having triangular cells. The different options are selected by varying numbers on the *F 64* data set in the *control* file. Or the *F 488* data set.




Note that grid A above (*F 64 0*) is no longer the default option. When using this option, then the levels of the grid lines (in meters) must be given on the *G 3* data set. Option B can be specified using the *F 64* data set (currently default). When using the *F 64 11* and *F 64 13* options, the values on the *G 3* data set are not used. The new default value of the *F 64* data set is 11, and this will gie a grid similar to figure B above, except that a layer of hex-cells will be generated at the bed. A layer of hex-cells at the bed is

important if bedload is to be computed.

The horizontal grid lines are often preferred when calculating lakes where density stratification is very important. Then the non-orthogonal grid may cause instabilities. Also, for a lake with stratification, most of the velocities occur close to the surface, and if the grid is not exact at the bed, this may not affect the results much. The non-horizontal grid lines are used where a very accurate description of the bed is necessary, and where density gradients are relatively small.

If the user wants to create a nested grid, the menu option *Add nested block* should be chosen. The user chooses the corners of the nested grid inside the already generated coarse grid. But instead of choosing *Block size*, the option *Nested block* should be chosen. Then a dialog box is given, where the user choose parameters for the nested block. The nested block is automatically connected to the rest of the grid in the three-dimensional generation of the grid. The nested grids are further described in Chapter 4.7

**Important note:**

After having edited the grid, it is advisable to write the content to the *unstruc* file. This is done in the *File* option of the main menu. Also note that the 3D grid first have to be made, using *Generate* and *3D grid* in the menu.

Using the latest wetting/drying algorithms of SSIIM 2, it is often best to use only one block. Then the water level should be chosen so high that it is above the bed at all places. Additional *geodata* points may be generated in areas where there are too few points. An *unstruc* file is generated for the block, where no drying has taken place. This unstruc file is then used in all following computations. Inflow/outflow is specified in this file. Often, the water level at the start of the computation is below the highest water level. Then a *koordina* file can be made with the initial water level, often using a spreadsheet to modify the *koordina.t* file. The *F 112 1* data set is used when reading the *unstruc* file. This causes the original *unstruc* file to be read first, then the *koordina* file with the lower water level is read, and then the grid is regenerated to the lowered water level. IMPORTANT: The *unstruc* file must not be written after this procedure, when the water level has been lowered. Only the original *unstruc* file must be used in the following work. Otherwise, no geometrical information will be stored for the dry areas, causing problems later.

### 3.3.3 Bed interpolation algorithm

The algorithm provide a method to interpolate the bed levels of the grid, given a large number of measured points, randomly distributed on the bed. The measured points are given in the *geodata* file.

In the following, the index $p$ is given for the point where the algorithm computes the bed level. This is at a grid line intesection in the 2D depth-averaged grid. The algorithm then computes the vertical elevation of this point, $z_p$, given the coordinates for this point ($x_p$, $y_p$) and a number of *geodata* points: ($x_i$,$y_i$,$z_i$).

The algorithm divides all the *geodata* points in four groups, according to their position in the $x$-$y$ plane. The first group has both larger $x$ and $y$ values than point $p$. The second group has larger $x$ values and

smaller y values. The third group has smaller x and y values than p. And the fourth group has larger *x* values and smaller y values. From each group, the point closest to *p* is chosen. This gives normally four points, all surrounding point p. A linear interpolation of the $z_p$ value from these four points is then done, based on the inverse of the distance from the points to *p*. The following line is written to the *boogie.bed* file if *F 1 D* is given in the *control* file:

*First: 10 20 345 456 345 321*

The two first integers are the i and j indexes for point *p*. The four following indexes denote the four surrounding points. The numbering follow the order of the points in the *geodata* file. The first point in the *geodata* file is 1, the second 2 etc.

Some special algorithms are used if there are no *geodata* points in one or more of the groups. If there are no points in two or more groups, then the algorithm may fail. Then a zero $z_p$ value will be given to point *p*. Also, the following warning message will be written to the *boogie.bed* file:

*Warning, null value first loop for i,j=10 20*

For this example, *p* is located at (10,20).

It is recommended to check this after a bed interpolation, or look at the contour map of the bed levels. This should be checked for areas where the bed level is zero. If this occur, the $z_p$ level could be given in the *koordina* file, using an editor. Or the correct value can be specified directly in the *GridEditor*.

Also note that if a *geodata* point is located within a distance of 0.05 meters from *p*, then the *z* value of this point is used instead of the interpolation. Then the following line is written to the *boogie.bed* file:

*Using exact value for 10 20 4.567*

In this case, the value for point (10,20) is 4.567 meters.

Note, the value 0.05 meters can be changed by the user, by giving it on the *F 47* data set.

### 3.3.4 Displaying measured bed changes in SSIIM 2 for Windows

Measured bed changes can be displayed in SSIIM 2 by using the following method:

1. First a grid has to be made of the geometry. The grid would typically be made from a *geodata* file, taken from the assumed lowest bed levels. For example after the flushing, or before sediment deposition. The grid is then written to the *unstruc* file. When this file is written, a file called *koomin.bed* is also written. Rename this file, so it is not overwritten later.

2. Add an *F 249 1* data set to the *control* file. This will allow both positive and negative values of the bed elevation changes. Make sure you are using a SSIIM 2 executable made after 14. February 2010.

3. Use the same grid as in point 1, but now change the *geodata* file to the one from the other bed level. Use this file to make a new bed level with the same grid as in point 1. Write the *unstruc* file

4. Start the program with the *unstruc* file from point 3 and the *koomin* file from point 1. The *koomin* file must now be called *koomin* and it must not have any extension.

5. The variable "sediment thickness" will now be the difference between the two bed levels in the two geodata files. This can be displayed in the SSIIM graphics, or written to the ParaView file. If the variable does not show up when the Tecplot/ParaView file is written directly from the menu, it is necessary to add the parameter '*t*' on the *G 24* data set and use an *F 48* option in the *control* file to produce the ParaView file.

6. The initial water volume of the reservoir is written to the *boogie* file at the beginning of each transient sediment computation. Using the two *unstruc* files from point 1 and 3 will give two volumes for the reservoir. As long as the water level is the same, the difference in volumes will give the volume of the deposited/eroded sediments in the reservoir.

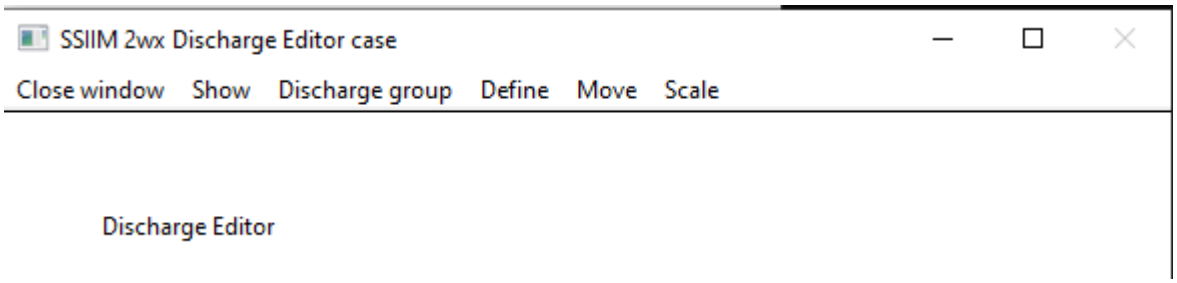
## 3.4 The *DischargeEditor*

Because the SSIIM 2 grid is unstructured, it is not generally possible to give the location of inflow/outflow water discharges in the *control* file. The numbering of the grid lines are not easily obtained. Instead, the discharges are given in the *DischargeEditor*. The values and locations are stored in the *unstruc* file.

The *DischargeEditor* is invoked from the *View* option in the main menu of the Windows versions of SSIIM 2. It is somewhat similar to the *GridEditor*, in that the grid outline is shown in plan view. The inflow and outflow on sides of the domain are specified.

**Discharge groups**

The discharges are organized in maximum ten groups. Each group has a water discharge, and some other characteristics. Among the characteristics is if it is an inflow or an outflow. The user first chooses which group to give data for. This is done by choosing *Discharge Group* in the meny and then a number. The most tested option is to have inflow as group 1 and outflow as group 2. But in principle this could be the other way around and more inflow/outflow group could be added. Then choose *Define->Set discharge values*. A dialog box emerges, and the water discharge in m3/s is given in the editfield. Also, cross off if the discharge is inflow or outflow. The zero gradien option may work.



Then choose Define->*Add surfaces* in the menu. Click on the side of the surface of the domain where you want the inflow/outflow to be. If the wrong surface is added, choose *Define->Remove all in group* from the menu to do it over again. It is also possible to add 10 sides at a time, or 100. This is convenient for fine grids. .

Note that, there must be at least two groups: one inflow group and one outflow group. Also, the sum of the discharges in the inflow group must be equal to the sum of the discharges for the outflow group.

One line is written to the *boogie* file for each discharge group above zero. This can be used to check if the sum of the discharges are zero, and if all or too many discharge groups are used.

The tutorials show examples of how to use the *DischargeEditor.*

***DischargeEditor* needed?**

Often, a channel or a river is to be modelled, where there is only one inflow over the whole upstream cross-section and one outflow over the whole downstream cross-section. Then the *DischargeEditor* is not needed. Instead, add *F 314 1 1* in the *control* file and give the water discharges on two *F 237* data sets.

## 3.5 Presentation graphics

There are two to three graphics modules for presentation of results. These can be invoked any time during the calculation or afterwards. More than one module can run simultaneously for the OS/2 version. The modules are choices under the Graphics option of the main menu:

- Map (plan view) of grid/vectors
- Contour map (plan view)
- Contourvector map (plan view)
- Longitudinal profile (ProfileI)
- Cross-sectional profile (ProfileJ)

Map presents the geometry seen from above. It is possible to get velocity vector plots and plots and bar plots of concentration, diffusivity, k, etc. It is also possible to plot the grid and change between different vertical levels. In the Windows versions, the contour map is also given from the same menu option.

Contour map presents the variables as contour plots, seen from above. The user can give the values of the contour lines on the *L* data set in the *control* file. If the *L* data set is not given, seven contour lines will be used. These are calculated to be within the range of the calculated variable field. The values of the different lines are given on the colour scale at the lower left corner of the window. The text gives the values of the maximum and minimum values, and there are six equal interval between maximum and minimum values (default).

Profile presents a longitudinal/cross-section profile of the geometry. Graphs with different parameters as a function of depth along the longitudinal profile is obtained. It is also possible to view the grid or the velocity vectors. It is possible to change between different longitudinal profiles.

**Graphics menu**

The option Move is used to move the plot upwards, downwards or sideways. The arrow keys can be used instead of the menu.

The option Scale is used to enlarge, shrink or distort the plot. The keys <Page Up> and <Page Down> can be used for scaling. Some of the graphs also use <CTRL> + arrows for distorting the plot.

An option in the menu bar is Graph. The pull-down menu displays different parameters that can be shown.

Printing from the Windows versions is done directly with the Print option in the main menu.

# Chapter 4. More detailed advice for using SSIIM

## 4.1 The grid

Making the grid is often the most time-consuming process in the preparation of input data for SSIIM or other CFD computations. The general idea is to divide the water body into cells. The size and alignment of the cells will strongly influence the accuracy of the calculation, the convergence and the computational time.

The grid used in SSIIM 2 is unstructured. This makes it easier to adapt the grid to complex geometries without loss of accuracy or slow convergence. In the following there are given some guidelines of how to make a good grid. Also, it is recommended to read Chapter 4.4 first.

Here is some advice on how to shape the cells:

1. Make the grid line intersections as perpendicular as possible. It is not advisable to have intersections with an angle of less than 45 degrees. Non-orthogonality in the grid will make the convergence slower. It is recommended to use the elliptic grid generator to make the grid smoother.

2. Try to align the grid lines in the streamwise direction parallel to the velocity vectors. This will decrease false diffusion.

3. The distortion ratio should not be too great. The distortion ratio is the dimension of the grid in one direction divided by the dimension in another direction. Some people say this should be less than 2 (two), but other people have obtained good results for ratios up to 10. On occasions, ratios of up to 100 have been used. This gave reasonable results, but it required very low relaxation coefficients and an extremely large number of iterations to converge.

4. The size of a grid cell should not be too much larger than its neighbours. Some people say the increase in size should not be greater than 20 %. On some occasions this value have been over 1000 % (a factor 10). Some of these cases gave reasonable results, but other cases gave unphysical results. A recommendation is to try to stay within 50 %, but if much larger values are used, be aware that unphysical results may occur. Unphysical results can be velocity vectors that point in another direction than what seems natural, for example not parallel to walls.

Chapter 4.2 give more advice on convergence and interpretation of the results.

## 4.2 Experience with convergence and stability

There are three ways SSIIM 2 can crash:

1. The program stops calculation and there is an error message in the program window.
2. The program stops and exits, the menu and the dialog box disappear.
3. The program stops with a system error, usually floating point invalid operation, overflow or underflow error.

Often an error message is written to the *boogie* file before the crash occur. This happens particularly for crash type 1 and 2 of this list above. If an error in the input files are detected, an error message is written to the *boogie* file and type 2 crash occurs. If the program crashes, it is therefore recommended to take a look at the *boogie* file. If crash type 1 occurs, it is also possible to look at the graphics to try to see where in the geometry the problem is.

**Convergence**

If the program does not crash, the next problem is to get a converged solution. This is often a problem for many CFD cases. Some of the important factors are:

- A good grid
- Proper relaxation coefficients
- Correct boundary conditions
- A fast solver
- Stable numerical algorithms

It is always a good thing to check the boundary conditions in case of slow convergence or strange results. Also note that warning messages may be written to the boogie file if particular undesirable configurations are present. Therefore, check the *boogie* file if problems occur.

Experience shows that the degree of non-orthogonality of the grid will affect the convergence. A higher degree of non-orthogonality will give slower convergence. A slower convergence will also be experienced where strong gradients are present. This applies for example at the inflow of a jet from a wall.

The convergence speed is strongly influenced by the choice of the solver. For most cases, using a multi-grid method for shallow flows will give improved convergence. This is invoked by using *F 168* and *K 5 0 0 0 10 0 0* in the *control* file.

For most cases lower relaxation coefficients will give less instabilities during the convergence, but a slower convergence. Higher relaxation coefficients will give more rapid convergence if there are no instabilities. This is however not always the case. Instabilities can be observed during the iterations when the residual or the velocities increase and decrease periodically. But for most cases of instability problem it is recommended to lower the values on the *K 3* data set in the *control* file. The default values are 0.8 0.8 0.8 0.2 0.5 0.5. The first lowering may be to *K 3 0.4 0.4 0.4 0.1 0.2 0.2*. If the instabilities are still there, it is possible to try *K 3 0.2 0.2 0.2 0.05 0.1 0.1*, for example.

For the convergence of the *k* and epsilon equations for river problems, the size of the cell closest to the bed is important. In SSIIM 1, this can be changed by changing the second number in the G 3 data set in the *control* file. This parameter also depends on the roughness of the bed. The height of the bed cell should not be too much smaller than the roughness of the bed. The following formula is used to determine the roughness of the bed, given the Stricklers friction coefficient *n* (van Rijn, 1982):

$$n=\frac{d_{90}^{1/6}}{26} \quad \text{and} \quad k_s=3\,d_{90} \tag{3.2.1}$$

If the solution blows up after the first few iterations, it is possible to set the relaxation coefficients very low, and then increase the coefficients for the following iterations.

For some cases there have been problems getting the solution to converge if there are parts of the geometry that has relatively low total velocity. Experience has shown that the initial conditions then may be important. If such a situation is present it is important to start the iterations with very low initial velocities. This is done with the *G 8* data sets.

An advantage with SSIIM compared with other CFD programs is that the graphics is connected directly with the computational module. This enables the user to see the results while the program is doing the computations. If problems occur, it is recommended that the user look at the graphics to try to identify these. This would be similar to a physical model study, where the engineer will watch what happens during the run. Looking at velocity vectors, pressure fields, bed level changes and the location of the water surface gives an idea about reasons for crashes.

For convergence problems, SSIIM 2 has an option of showing plan view of the residuals in 2D. This makes it easier to identify areas of high residuals.

**Transient calculations**

If during steady calculations there are oscillations in the velocities, this may be a sign that the flow field has a transient character. One may then invoke the transient calculation and a periodical transient solution may be observed. However, another possibility is that the oscillations disappear after adding transient terms. The transient terms can have a stabilizing effect on the solution for some cases.

Note that a steady state solution may very well emerge even though the corresponding prototype flow field has transient oscillations (Olsen and Melaaen, 1996). The reason for this is usually that the turbulence model overpredicts the eddy-viscosity, or the grid is so coarse that false diffusion dampen out the eddies.

For very large grids (over 50 million cells) an experience with SSIIM 2 is that a time step is needed initially to stabilize the solution. However, after some time the decrease in the residuals will be very low. A faster convergence has then been achieved if the result file is first written and then the computations are restarted without the time step, but by first reading the result file.

**Convergence problems with shallow/triangular cells during wetting/drying computations**

Transient computations with wetting/drying sometimes give convergence problems and crashes at the boundary of the geometry. This situation can be identified when looking at the *Map Graphics*, where large velocity vectors are observed after a crash. During computations, the *Map Graphics* may show very high residuals at the side boundary. Also, the velocity vectors may seem to oscillate there before the crash. This is typically taking place in triangular cells or very shallow ares, often at a flow separation point on the side boundary. It is recommended to look at the velocity vectors in the graphics when oscillations occur. Then problems may be more easily identified.

Sometimes the problem is due to low depth/roughness ratios. In the *Map Graphics*, it is possible to see this ratio directly. If the ratio is low, then the wall laws (Eq. 6.1.16) can give negative U+ values. This will cause instability and crashes.

Advice for solving instability issuses is to introduce stabilization algorithms by adding data sets to the *control* file:

**1.** Lower the relaxation coefficients on the *K 3* data set

**2.** Introduce an inner time step with the *F 292* data set in the *control* file. For example *F 292 1 1.0*, where 1.0 is the inner time step in seconds. The inner time step should be chosen much smaller than the time step on the *F 33* data set. The inner time step will only contain information about the flow field from the last iteration, and not from the previous time step. Therefore, the inner time step will act more like a relaxation factor, and dampen changes between each inner iteration.

**3.** If the instabilities are due to low depth/roughness ratios, the minimum depth on the *F 94* data set can be increased. This an effective measure to dampen instabilities due to oscillations. The reason for these type of instabilities is most often an extreme length/height ratio of the cells. If an *F 94* data set is already used, then the numbers should be increased. If the *F 94* set is not used, then look in the *boogie* file to see what the default values are. Then introduce an *F 94* data set with higher values. The disadvantage of using higher values on the *F 94* data set is that the most shallow areas of the geometry will not be modelled, or modelled with only one cell in the vertical direction. An alternative/additional option is to increase the minimum *U+* value. This is set on the *F 139* data set. The default value is 1.0. This can be increased to for example 3 or 4.

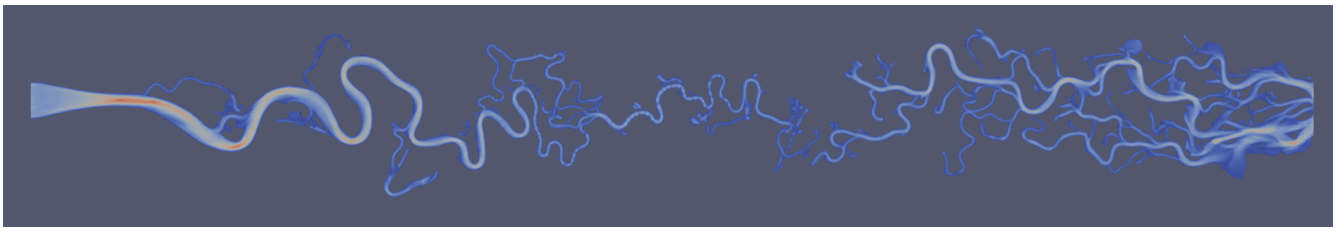
**4.** Use the *F 159 d*ata set to change some of the grid generation methods. For example, the second parameter can be changed to 9, removing ridges in the grid.

**5.** Sometimes a combination of unfortunate pressure/velocities may cause a crash. If the *F 219* option is invoked, the program will set the pressure to zero and start over again in the current time step when the residual reaches $10^8$. A warning message is then written to the *boogie* file. This is called a flushing of the pressure field.

**Grid splitting**

This instability may occur when using a free surface algorithm in shallow areas. The water surface changes may be so large that a complete part of the computational domain between the inflow and outflow disappears. The grid is then split in two, with no connection between the parts. This is both

numerically incorrect and an unphysical situation.



The problem is most frequent occurring when using the free surface algorithms invoked by the *F 36 2* or *7* data sets in the *control* file. The first thing to do then is to lower the relaxation coefficient for the water level movements on the *G 6* data set. The two last numbers on the *G 6* data set are floats, the relaxation coefficient is the first of these. The coefficient can be lowered to 0.01 from the default value of 0.1 However, the water level will then also sink more slowly, so it should be checked if this cause a problem. If so, then the time step has to be lowered.

An alternative solution for the problem is to use an implicit Shallow-Water Equations solver to compute the changes in the water levels. This is usually more stable than methods based on the pressure from the Navier-Stokes equations. Then, *F 36 9* is used in combination with *F 323 1*.

Algorithms to improve the convergence of the *F 36 9* options can be given on the *F 343* data set. *F 343 3* invokes a classical multigrid method, while *F 343 2* invokes a block-correction method in 1D in the streamwise direction of the geometry.

## 4.3 Interpretation of results

As mentioned earlier, it is advisable to have experience in computational fluid dynamics when proper interpretation of the results is required. Some guidance is given below.

An important numerical effect that can deteriorate the results is false diffusion. This effect is most noticed for first-order schemes, including the POW scheme. However, also high-order schemes introduce false diffusion, although less than the first order schemes. False diffusion depends on how well the flow velocity vectors are aligned with the grid lines. For small alignment angles, the effect is small. Maximum false diffusion will happen when the grid lines are aligned 45 degrees with the flow. The amount of false diffusion also depends on the size of the grid cells.

There are three methods to decrease the amount of false diffusion:

1. Decrease the size of the grid cells = increase the number of cells
2. Align the grid with the flow field
3. Use a high order scheme

Point 2 may be difficult in a practical situation. However, the calculations should be carried out using approach 1 and/or 3 to assess the effect of false diffusion.

Another important aspect is the boundary condition. This especially applies to the inflowing boundary. If the velocity field at the inflowing boundary is not known exactly, one should try different velocity distributions to try to assess the effect of this parameter. For a river running into a reservoir, it is possible to model a part of the river upstream of the reservoir, and thereby obtaining a better estimate for the velocity distribution where the river enters the reservoir. The upstream boundary condition is also important for sediment calculations, where both the total amount of sediment inflow and the sediment grain size distribution can be varied. For the bed boundary, it is possible to vary the roughness to investigate the effect of this parameter. It can also sometimes be advantageous to make variations for the formula for sediment concentration close to the bed. This especially applies for sediment particles outside the range for which the formula is applied for.

When interpreting the results from the model it is also important to keep the accuracy of model in mind. The k-ε turbulence model has limitations in how accurate the turbulence field is predicted. This will also affect the velocity field. For example, when calculating the recirculation zone for a step case, the length of the recirculation zone can often not be predicted with any better accuracy than 10-30 %.

In some situations the water flow will be time-dependent. An example can be oscillations behind a cylinder or in an expansion. It is possible to obtain a converged steady solution from the model although the physical problem is unsteady (Olsen and Melaaen, 1993 and 1996). This must be considered when interpreting the results. The effects of an unsteady solution compared to the given solution should be assessed if this is probable. When an unsteady case is solved with a steady method there can be convergence problems. If the relaxation factors have to be fairly low to get the solution to converge, this can be an indication that the flow field may be unsteady.

Another topic of interpretation is the resolution of the calculated flow field compared with the size of the grid cells. Several cells are required to dissolve a recirculating zone. Flow field characteristics smaller than about 4-7 cells may not show up in the solution. And a recirculation zone with very few cells may be inaccurately calculated because a certain resolution is required.

For some flow geometries the Navier-Stokes equations may have more than one solution. This can for example be seen in a flume with a symmetric expanding channel, where the jet may follow one side of the channel. If an obstacle is inserted into the side with the jet, the jet moves to the other side of the channel. This phenomena have also been experienced in non- symmetrical expansions, when varying the number of grid cells caused the Navier-Stokes equations to converge with a completely different flow field. It is thought that this problem has a higher risk of occurrence in geometries with expansions and recirculation zones, and also if the Second-Order Upwind scheme is used instead of the Power-Law Scheme.

**Transient sediment computations**

Over the last years SSIIM has been used for several cases with transient sediment computations. The bed then changes over time in a geomorphological simulation. Some experiences have been obtained, given in the following.

In current SSIIM versions, the sediment transport boundary condition for the bed is based on van Rijn's formula. This formula is developed for fine uniform sediments in a flow with relatively low water

velocity and great depth. It is not certain that the formula will produce good results for other cases. However, it may do so. It is important to assess the results by comparisons with laboratory experiments or field data. In van Rijn's formula, one parameter is the distance from the bed. In SSIIM, this is chosen to be half the height of the cell closest to the bed. This means it may be possible to obtain different results for the sediment transport depending on the magnitude of the cell closest to the bed. It is therefore advisable to include this variable in a parameter sensitivity test. An algorithm is included in SSIIM to take into account unusual values of the ratio of

*a/b*

where *a* is the distance from the bed and *b* is the bed form height. The algorithm is based on the Hunter-Rouse distribution of suspended sediments. It is invoked by using the *F 60 / F 110* data sets.

Doing a sediment computation, it is important to use the correct shear stress at the bed. The shear stress is a function of the bed roughness. There are several options on how this is computed, and it is recommended to use a reasonable value. It is possible to let SSIIM compute the effective roughness as a function of the sediment grain size distribution and the bed form height. This is of course the most elegant solution. However, the bed form roughness is computed by van Rijn's method, which may not give accurate results for all cases. The method is derived from uniform grain size data, giving higher bed forms than less uniform bed grain size distributions. The default roughness in SSIIM is 2 cm, which should not be used unless this is a reasonable value. Determination of the roughness algorithm is done on the *F 90* data set.

The numerical algorithms are approximations to exact formulas. The accuracy may therefore be variable. It is recommended to check the sediment continuity when doing sedimentation computation. This can be done by using the *F 1 D* option in the *control* file. Sediment continuity fluxes are then written to the boogie file for each iteration. One common reason for sediment disappearing is that the solution of the equations for the sediment grain size distribution has not converged. This is particularly the case for multiple sediment sizes. The convergence criteria and number of iterations for the solver is given on the *F 4* data set. It is recommended to try to lower the convergence criteria and increase the numbers of iterations if there are sediment continuity problems. This will, however, lead to increased computational time, especially if there are many sediment sizes.

Transient sediment computations are often done in connection with a moving free surface. This may lead to stability problems. One reason may be that the water level is only updated each 10th iteration (default). The water depth may then change a lot, and supercritical flow may occur in some locations. The problem may be solved by reducing the time step or decrease the number of iterations between each water surface update. This is done on the *F 105* data set.

When wetting/drying occur, some part of the water body may form ponds in the grid, separate from the main flow. If the ponds do not have an inflow or outflow, the coefficients in the discretized equations will be zero. This may lead to a situation with division on a very low number, close to zero, and instabilities may result. An algorithm to avoid the problem can invoked by *F 104* data set. The number given on the data set is added to the $a_p$ term for the cells that have low $a_p$ values. Another option is to use the *F 394 10* data set, which removes ponds from the grid.

The accuracy of the computation is often a function of the number of grid cells. For the vertical

direction, this can be controlled by the *F 87* data set. It is recommended to change the grid resolution to see how this affects the results.

Another empirical parameter is the minimum grid cell size. This is given on the *F 94* data set. Experience from the Kali Gandaki reservoir flushing case showed that values of 1 cm worked well when modelling a physical model with water depths of around 5 cm. The horizontal sizes of the grid cells were 10-30 cm for this case.

## 4.4 Common problems

By own experience and watching students using SSIIM, there seems to be some frequent problems that can occur.

**Sequence of actions**

In some examples, the *F 2* option is given is given in the *control* file, so that the computations starts automatically. It is most often not a good idea to read input files or start new computations while the program is computing something. This can easily cause the program to crash. If using the menu to start computations, make sure the grid is read completely before starting for example water flow computations. And make sure the water flow computations are finished before the sediment computations are done when using the menu for starting the computations.

**SSIIM 2 inflow/outflow specification**

In SSIIM 2, the location of the inflow and outflow regions are specified graphically in the *DishargeEditor*.

It can also be useful to test that the inflow/outflow areas are correctly defined by checking the Discharge Editor while the program is running.

**Problems with *unstruc* files made by program versions before summer 2012.**

When the grid is made and the *unstruc* file is written, it may not look correct when it is read back in again. Typically, some of the grid lines along the boundary will move to incorrect coordinates. The reason for the problem lies in the way the geometry data is organized for the wetting and drying computation. To make it possible for lateral erosion to take place, the wetting procedure must have information about where the grid cells should be located outside the original river. To do this, the initial grid has to cover also the dry areas. This is done by using a high water level when making the initial grid and *unstruc* file. The water level should then be higher than all points in the bed, so that the grid will look structured when generated. The unstruc file generated with this water level has always to be used later when starting the program. To start the program with the correct, lower water level, this lower water level has to be given in the *koordina* file, together with the *F 112 1* data set. However, if a new *unstruc* file is written after the program is started with a lower water level, this will not contain the information about the grid in the dry areas. The strange lines will therefore appear when such an unstruc file is read.

This problem was corrected summer 2012, when also the *x* and *y* coordinates for the dry areas were written to the *unstruc* file. However, using *unstruc* files made before this time, the problem may still occur.

## 4.5 Bugs and bug finding

Sometimes SSIIM 2 will crash or give strange results. This can be due to bugs. It can be difficult to find bugs in computer programs with 100 000 lines of code. However, the approximate location of a bug can often be found without any access to the source code. Some guidelines are given in the following that will be helpful in finding the bug.

1. Is the bug reproducible? How is the bug reproduced?

2. Is the bug only in the latest version of the program or also in older versions?

3. Are there any warning or error messages in the *boogie* file? Use *F 1 D* to get more information in the file.

4. Is the bug connected with any input files? If you try different input files, or omit some input files, will the bug still be there? This applies especially to *bedres* files.

5. If the program ran well on an earlier case or a similar other case, what is the difference between the other case and the current case?

6. Often, a bug can be related to one of the algorithms invoked by the data sets in the *control* file. By removing some of the data sets in the file, it may be possible to connect the bug to a specific data set. Or a combination of data sets and input files.

**Complex bugs**

A bug can show up right away after the program starts or it can emerge after long computational times. A bug that shows up right away is usually easy to find by debugging the source code. A bug causing incorrect results after a longer computational time is more difficult to find. This can be called a complex bug. Here are some advice in how to find such a bug:

**1.** Similar to a physical laboratory model, you can look at the physics of the problem, observed in the SSIIM graphics during the computation: Is the water surface location correct? Is the water depth correct? Is the velocity field correct? Are the secondary currents correct? Is the roughness correct? Is the shear stress correct and follow patterns of depth/velocity/roughness? Does the sediment concentration and bed changes follow the pattern of the shear stress? Is there enough sediments on the bed for erosion to take place? The thickness of the sediments is seen in the *Map* graphics.

For example, if the erosion is too deep, this can be due to a too small sediment size, to little inflowing sediments or too high shear stress. The too high shear stress can be due to too high velocities or too

high roughness. If the velocity is too high, this can be due to too low water depth. Too low water depth can be due to too low pressure. By following the parameters in the graphics, it is often possible to trace the bug to an algorithm for specific physical process. SSIIM often has several algorithms for the same process, for example computation of the water level. Then other alternative algorithms for the same process can be tested.

During this testing, it can be useful to make ParaView or Tecplot animations of a number of variables, for example velocities, pressure, water levels, bed levels, bed changes, roughness etc.

For sediment computations, it is useful to look at the sediment continuity. The *boogie* file will provide more detailed results if *F 1 D* is given in the *control* file.

**2.** If the bug is in a complex case/geometry, it can be easier to find the bug in a simpler similar case. A typical simpler case is a straight channel with constant depth and velocity. Then, parameters as shear stress, sediment transport capacity and bed changes can be computed by hand and compared with the results from SSIIM 2. Another simplification is to use one sediment size instead of many.

**3.** It is often necessary to rerun SSIIM several times with different input parameters to find the bug. Considerable time can then be saved by using a coarse grid instead of a fine grid for the same case. It doesn't matter that the coarse grid will not give accurate results, as long as the bug is reproduced. It has happened that a bug was discovered after 6 weeks of computational time on a grid with 4 million cells. If a grid with 4000 cells had been used instead and the other parameters were the same, the bug had been found after a computational time of 1 hour.

## 4.6 Frequently asked questions

*1. The program crashes. What do I do?*

See Chapter 4.2

*2. I think there is a bug in the program. What do I do?*

See Chapter 4.5

*3. How is the roughness specified in SSIIM 2?*

Roughness parameters are used in the wall laws for rough boundaries. The roughness will affect how large the bed/wall shear stress will be. The wall law for rough boundaries includes a roughness value in meters. One roughness value is used for all the side walls. This is called *XksWall* internally in the program. For the bed, a separate value is given for each cell in the two-dimensional array called *Xks[i] [j]* in the computer program.

The default values of the roughness variables are given by converting the Manning-Strickler value of the *W 1* data set to a roughness height.

The user can override this value by using the *F 16* data set. Then both *XksWall* and *Xks[i][j]* will take this value. If the user wants to specify a spatially varying roughness at the bed, this can be done by making a *bedrough* file. Then only the *Xks[i][j]* values will be affected, and not the *XksWall* value. The roughness specified in the *bedrough* file will override the value specified on the *F 16* data set. This means that using both a *bedrough* file and the *F 16* data set, the value on the *F 16* data set will only be used for the side walls.

The roughness given in the *XksWall* and *Xks[i][j]* variables will not affect the location of the initial water surface. However, the variables will affect the shear stress and thereby the pressure in the flow field. So if the algorithm is used where the water surface is updated as a function of the pressure field, the water surface will be affected by the roughness values.

In the time-dependent sediment computations, invoked by using the *F 37* data set, it is possible to further change the roughness values in the *Xks[i][j]* array. This is done by using the *F 90* data set. Then the bed form height can be computed, and thereby also the bed roughness. This will again affect the location of the water surface if it is recomputed during the computation.

If you are unsure about what the roughness is for a given configuration of input, it is recommended to start the program and look at the roughness parameter in the Map graphics.

*4. How is the free surface computed?*

See Chapter 2.5.

*5. How can I convert between sediment fluxes, bed elevation changes and concentrations?*

The concentrations in SSIIM is given as volume fractions, as in van Rijns formulas. That is the volume of the sediment particles as a fraction of the total volume of the water/sediment mixture. If the density of the sediments is 2.65 kg/litre, the mass fraction will be 2.65 times higher than the volume fraction. Or in other words, if you have a mass concentration, you need to divide by 2.65 to get it in volume concentrations. Note that concentrations in ppm given in litterature are often mass concentrations.

The specific density of the sediments can be modified on the *F 11* data set in the *control* file, if you want a different specific density than 2.65.

A flux, $F$, of sediments in kg/s is obtained by the following formula: $F = c \rho_s U A$

$U$ is the water velocity in m/s normal to the area $A$ in $m^2$. The sediment concentration in volume fraction is $c$ and $\rho_s$ is the density of sediments in kg/$m^3$, for example 2650 kg/$m^3$.

In SSIIM 1, the inflow of sediments can be specified on the *I* data sets in the *control* file, in kg/s. They can also be specified for example in the *timei* file as a concentration. If a concentration is to be given, then this is the volume fraction. How this is computed is most easily shown in an example:

Let us say we have a water inflow of 3 $m^3$/s and a sediment inflow of 0.1 kg/s. The sediments will take up a volume of 0.1 kg / 2.65 kg/l = 0.03777 litres. The 3 $m^3$/s of water will take up a volume of 3000

litres. The volume fraction will then be 0.03777 litres / 3000 litres = $1.25 \times 10^{-5}$. The concentration value in the *timei* file should be 1.25e-5 or 0.0000125.

The next question is how much volume on the bed does a given amount of sediment take up? The default sediment packing on the bed for SSIIM is 0.5, meaning that 50 % is sediment particles and 50 % is water. This can be changed on the F 26 data set. One cubic meter of sediments on the bed therefore contains 0.5 $m^3$ sediments. This will have a dry mass of

0.5 m3 x 2650 kg/m3 = 1320 kg.

Of course, if the mass is wet, then the 500 kg of water must also be added, giving 1820 kg.

*6. How do I specify an initial grain size distribution on the bed in SSIIM 2?*

If the initial bed grain size distribution is uniform over the whole geometry, the S, N and B data sets can be used in the control file. If the initial distribution varies spatailly,the *fracres* file can be used, as described in Chapter 5.20.

The grain size distribution on the bed can also be given in the *bedres* file, which can be used to initiate values from an earlier run. The *bedres* file is written from SSIIM 2.

Note that the *F 306 1* data set must be given for the sediment information from the *bedres* file to be used. Also *F 306 1* have to be given for the *fracres* file to be read.

It is useful to look in the SSIIM 2 graphics after startup of the program to see if you have the initial values of the sediment parameters you want.

*7. I get problems with generating the grid in SSIIM 2 when wetting/drying occurs.*

See Chapter 2.8 and 2.9.

## 4.7 Nested grids

Nested grids are used for resolving finer scale problems in some part of the geometry. The nested grids can only be used in SSIIM 2. They are generated using the *GridEditor*.

Several nested grids can be used. The numbering of the grids are important and must be kept in mind when giving information later in the *control* file. The first grid generated is numbered 1, second 2 etc., whether they are nested or not.

When making a system with nested grids, some decisions have to be made regarding the order of solving equations for the different grid parts, and which equations to solve. The specification of which grid is to have its water flow computed is determined on the *G 25* data set. This data set reads a number of integers. The number of integers must be the same as the number of grids. The integer gives the

order of the computation. For example:

*G 25 3 1 1 2*

The first integer on the *G 25* data set says there are three working blocks. The two first blocks are computed first, as they have number 1. The third block is computed afterwards. The third block may be a nested block. The computation will then first be done on the main grid, and then on the nested grid afterwards. To compute all blocks at the same time, including the nested block, the following data set can be given:

*G 25 3 1 1 1*

Note that the default method is to compute all blocks at the same time, so then it is not necessary to specify the *G 25* data set.

If one block is not to be computed at all, a zero can be given. For example:

*G 25 3 0 0 1*

Then only the last grid is computed. This can be used when the bed changes only happen in the last block, and the water flow does not change in block 1 and 2. Then the steady water flow field is computed first, and the result file written. The result file is read before the unsteady computation is started.

The water flow is computed using the procedure above for each time step. Afterwards, the sediment transport may be computed. The specification of which block to compute the sediment is given on the *F 188* data set. The *F 188* data set only has one integer. If it is 0 (default), sediment transport will be computed in all blocks. If the integer is a positive number, only the block with this number is computed. If the integer is negative, all nested blocks are computed.

**Importing nested grids**

The typical application of nested grids can be local scour or investigating something at a smaller scale than what is given in the coarse grid. The coarse grid is then used to find the boundary conditions as for example the upstream velocity profile for the smaller domain. Then it is possible to make the grids in two parts. First, a fine grid is made of the smaller domain, for example around a bridge pier this grid is stored in a *koosurf* file, giving the vertical level of both the bed and the water. This file can be made in SSIIM 1 or SSIIM 2. This file is then imported into the SSIIM 2 *GridEditor.* The *koosurf* file must be renamed *koosurf.nes*, before it is read into SSIIM 2. Also, an additional line must be given at the beginning of the file. This line must start with the integer 999999 or 999998. Then six integers and five floating point numbers must follow.

The first four integers describes an outblocked region in the nested grid. This is used to block out a region of the nested grid, for example if one is to model flow around a cylinder. The four integers will have the same values as the 2nd to 5th integer on the *G 13* data set. If no outblocking is to be used, the integers should be set to zero. It is still recommended to use the *G 13* data set in the *control* file to block out the cell in the coarse grid. Then the *j* values need to be increased. The new numbers are most

easily found in the *GridEditor*, showing the grid indexes.

The following two integers gives the indexes for the center of the nested grid. This could typically be the *i* and *j* values of the center of a cylinder, if the nested grid contains a cylinder.

The following two floating point numbers gives the coordinates in *x* and *y* direction of the center of the nested grid. The values are given in the coordinate system of the coarse grid. The nested grid will thereby be moved to this location in the nested grid.

The three last floating point numbers on the line are: *scaling parameter*, *rotation* and *water level*. The *scaling parameter* will scale the nested grid. If a scaling is not wanted, a value of 1.0 should be given. The *rotation parameter* is a value in degrees, where the nested grid will be rotated around its center in the clockwise direction. The *water level* is a parameter which is used to specify a water level if it is different than the values in the *koosurf* file. The value is only used if the first number on the line is 999999. If the number is 999998, the original water level of the *koosurf* file will be used. However, a value must still be given, so that the total line will contain seven integers and five floating point numbers.

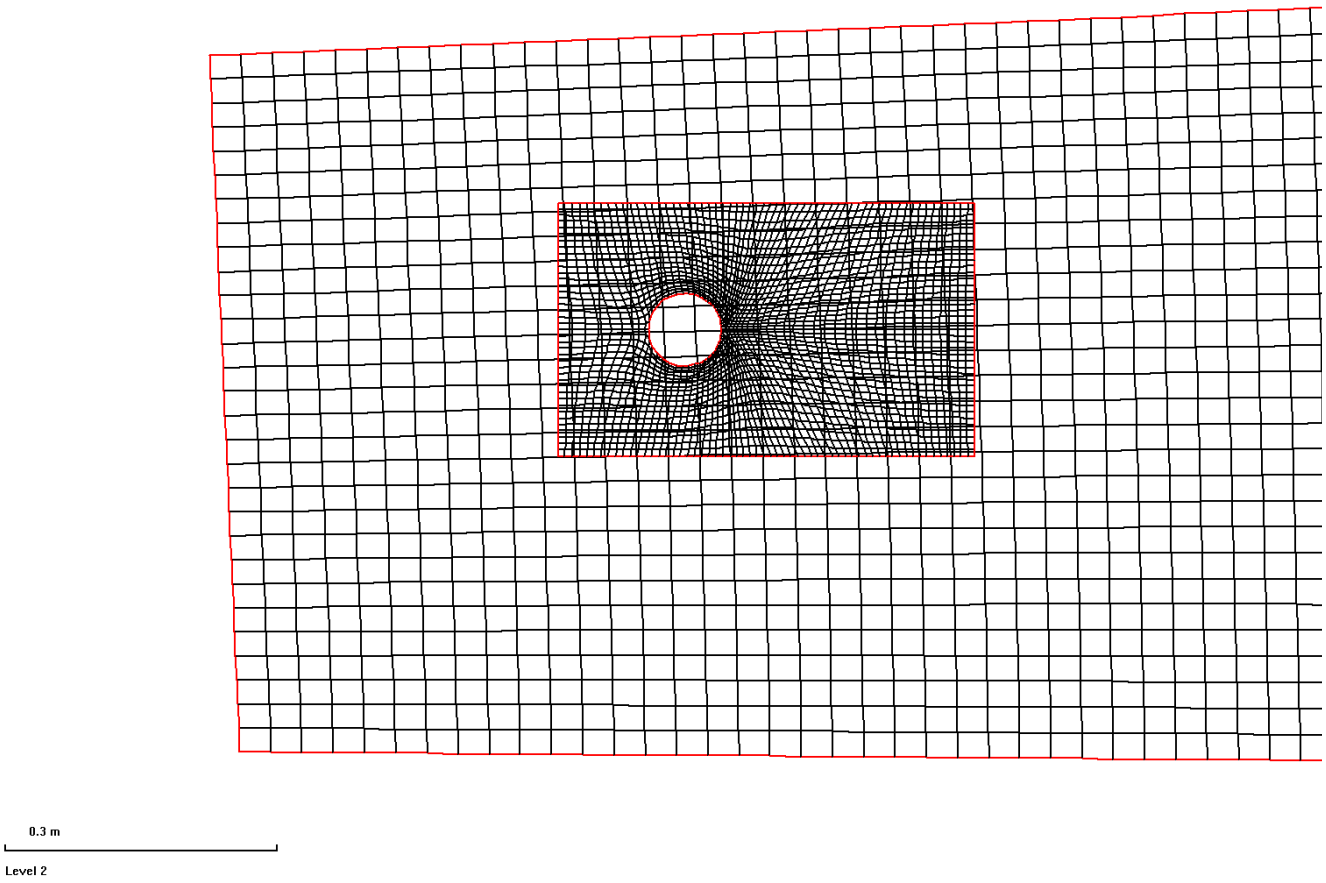
Then the coarse grid is made of the larger domain, for example a river. Afterwards, the fine grid is imported into the coarse grid domain using the *GridEditor*. The menu option *Block → Add nested block from koosurf* is used.

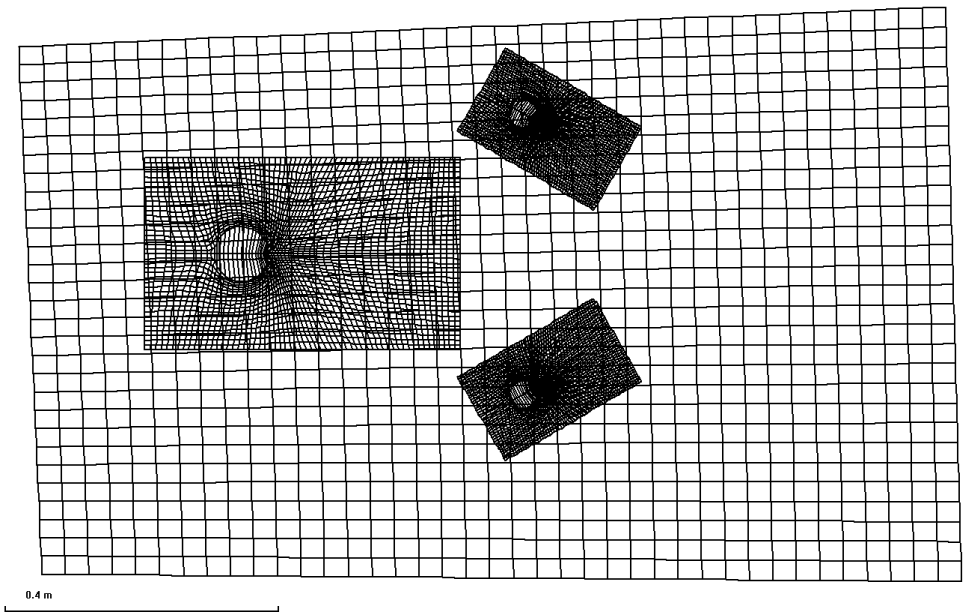
Note that multiple *koosurf.nes* files can be imported. Let us say that two *koosurf* files are made of two bridge piers. The two files are given the names *koosurf.1* and *koosurf.2*. Then the coarse grid is made in the *GridEditor*. Then the *koosurf.1* file is copied to the file *koosurf.nes*. Then the *koosurf.nes* file is imported into the *GridEditor*. Then the *koosurf.2* file is copied to the *koosurf.nes* file. Then the new *koosurf.nes* file is imported into the *GridEditor*. This can be repeated several times with multiple nested grids.

It is also possible to just change the first line in the *koosurf.nes* file between importing new nested grids. For example the location of the center of the nested grid.

The figures below shows how the grids may look with a combination of coarse and nested grids.

The actual *koosurf.nes* file used in the figures below to model a circular cylinder with a nested grid can be found on the web page: http://folk.ntnu.no/nilsol/ssiim/koosurf.nes.


0.3 m
Level 2


0.4 m

## 4.8 Displaying measured bed changes in SSIIM 2

It is possible to use the bed interpolation algorithms in SSIIM 2 for visualizing measured bed elevation changes. The measured bed elevations at two different times are used. The values can be exported to Tecplot or Paraview for nice 3D colour graphics. The volume of the changes can also be computed.

The following steps must be taken:

1. First a grid has to be made of the geometry. The grid would typically be made from a *geodata* file, taken from the assumed lowest bed levels. For example after the flushing, or before sediment deposition. The grid is then written to the *unstruc* file. When this file is written, a file called *koomin.bed* is also written. Rename this file to *koomin.bed1*, so it is not overwritten later. Note that a *koomin* file must not exist in the directory when the *unstruc* file is made.

2. Add an *F 249 1* data set to the *control* file. This will allow both positive and negative values of the bed elevation changes. Make sure you are using a SSIIM 2 executable made after 14. February 2010.

3. Use the same grid as in point 1, but now change the *geodata* file to the one from the other bed level. Use this file to make a new bed level with the same grid as in point 1. Make sure there is still no *koomin* file in the directory. Write the *unstruc* file.

4. Rename the *koomin.bed1* from point 1 to *koomin* without an extension. Then start the program with the *unstruc* file from point 3 and the *koomin* file from point 1.

5. The variable *sediment thickness* will now be the difference between the two bed levels from the two *geodata* files. This can be displayed in the SSIIM 2 graphics, or written to the ParaView or Tecplot files. If the variable does not show up when the Tecplot/ParaView file is written directly from the menu, it is necessary to add the parameter '*t'* on the *G 24* data set and use an *F 48* option in the *control* file to produce the ParaView file when writing the *result* file.

6. The initial water volume of the reservoir is written to the *boogie* file at the beginning of each transient sediment computation. Using the two *unstruc* files from point 1 and 3 will give two volumes for the reservoir. As long as the water level is the same, the difference in volumes will give the volume of the deposited/eroded sediments in the reservoir.

## 4.9 Modelling vegetation

Rivers may have vegetation on the banks/overbanks that can affect the velocity field, turbulence and particle concentrations. When modelling a natural river, the grid cells are often much larger than an individual vegetation branch. The approach to modelling vegetation is therefore to use a drag formula and itroduce a sink term in the Navier-Stokes equations. This requires a drag coefficient in each cell and also a parameter saying something about how many stems there are in a cells and how thick they are. This information is given on the *vegdata* file, described in more detail in Chapter 5.8. The file

gives different values of this vegetation paramter at varying depths for each 2D cell. Therby, the values are interpolated vertically to give the correct values when the grid moves according to erosion/deposition and changes in the water level.

## 4.10 Immersed boundary method

The grid used in SSIIM 2 has its limitations in that the vertical lines always have to be completely vertical. It is therefore diffult to model structures with very complex geometris. Such structures may be bridge piers, groynes or larger trees in a river. A solution is to use an immersed boundary method. Numercally, this is a bit similar to modelling vegetation, but the structure is now larger than the grid cells. So that several cells in the grid describeing the structure are "blocked out" and have zero velocity inside.

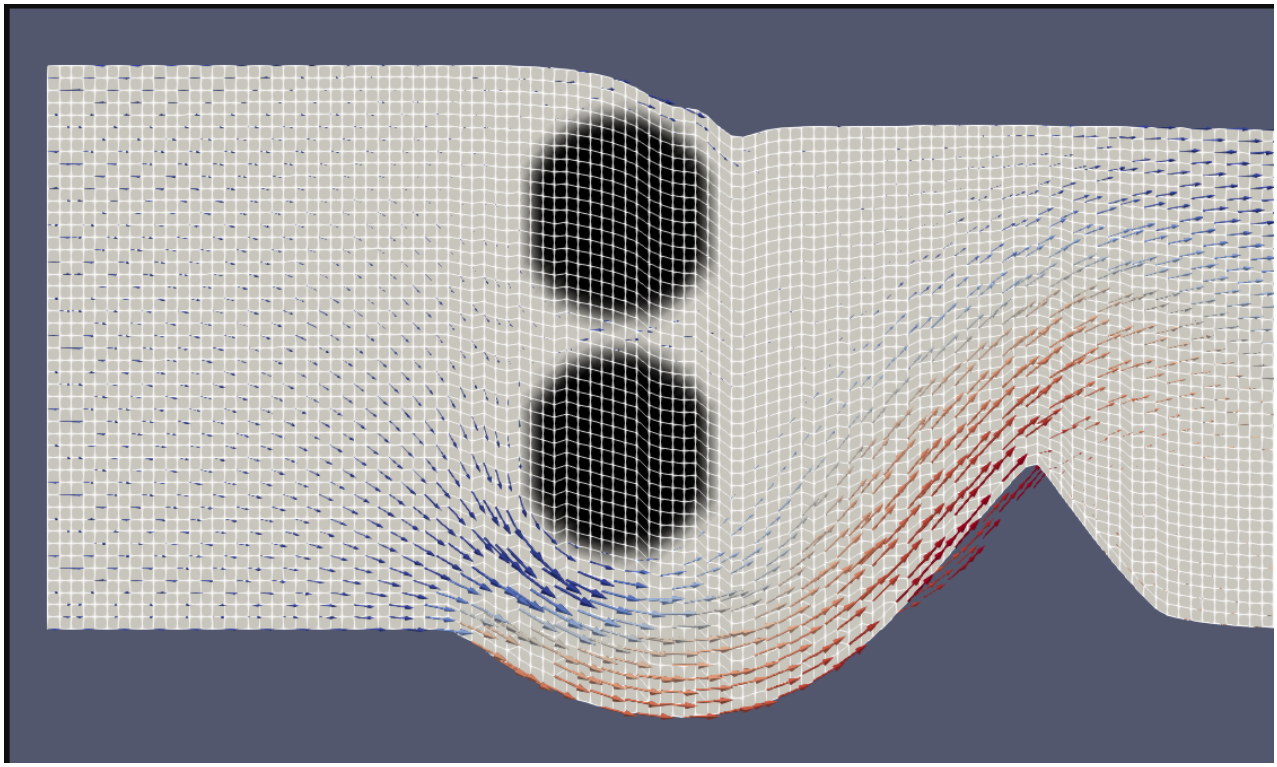
A complex structure is often described by an STL file. The STL file can be made with a CAD program. The file is often large, and it takes considerable computational time to determine if a cell in the grid is inside the STL geometry or not. If the grid was stationary, this would not be a problem, as this computation would only be done once, when the computation started. However, if the grid is moving vertically due to erosion/deposition and/or variations in the free surface, the use of the STL file would be required for each time step. This is avoided in SSIIM 2 by transferring the STL data to a vegdata file. The vegdata file is much smaller than the STL file, and it is much easier and faster for SSIIM 2 to search for the correct cells that blocked out.

The new SSIIM 2 version therefore has an algorithm that makes a vegdata file from an STL file. This only works if the STL file describes a closed geometry. Also, it has only been tested on simple horizontal cylinders, so it may not work for more complex geometries.

The algorith is invoked from the main menu in SSIIM 2, *File->STL file to vegdata*. Note that a grid has to be read first. The vegdata will be specific for this grid. So if a new, finer or coarser grid is made, then a new vegdata needs to be generated.

The figure below shows a longitudinal profile of a leaky barrier setup. Water is flowing from left to right. The black areas are regions of the grid that is blocked out by the vegdata file. These are two circular logs placed perpendicular to the flow direction. Velocity vectors are also seen, where the colours of the vectors indicate the sediment concentration. Red is high concentration and blue is low concentrations. The grid is also seen. The free water surface and the bed move as erosion/deposition occurs.

The *vegdata* file is describe in Chapter 5.8.



## 4.11 OpenFOAM grid generation

One of the most popular CFD programs today is OpenFOAM. The program is free and has open source code. However, it can be difficult to make the grid in OpenFOAM. The new SSIIM 2 version includes a function that converts a SSIIM 2 grid to an OpenFOAM grid.

OpenFOAM uses an unstructured grid, similar to SSIIM 2. The main input file for the grid in OpenFOAM is a file called blockMeshDict. This file is converted to a proper OpenFOAM grid using an OpenFOAM program called blockMesh. The blockMeshDict file describes a number of blocks and how they are connected. The conversion function in SSIIM 2 assumes that each cell in the SSIIM 2 grid is one block, and then writes the blockMeshDict file on this basis. This means that the blockMesh function in OpenFOAM may take some computational time to do the conversion for fine grids. But this is only done once, before the OpenFOAM computation starts.

The data set F 488 2 needs to be in the control file when generating the SSIIM grid that will later be used to make OpenFOAM input files. The F 488 2 algorithm produces non-hexahadral cells that conform to the OpenFOAM numbering convention.

The blockMeshDict file also contains information about the boundaries of the computational domain. An OpenFOAM input file can have many different combinations of inflow/outflow/wall/free surface boundaries. The blockMeshDict file made by SSIIM 2 simplifies this to a channel with six boundaries. One inflow region is at the upstream boundary and one outflow region is at the downstream boundary.

The top of the grid has zero gradients, similar to a free surface, and the bed has wall laws. There is also a right and a left side. These are set to walls by default, but can be changed to zero gradients for a 2D case.

SSIIM 2 also produces some of the boundary condition files for OpenFOAM, that are to be used in the */0/* directory in the OpenFOAM case folder. Many of these files need to be modified before the OpenFOAM computation starts. For example, the upstream velocity boundary contidion is given as a velocity in the *U* file, and not as a discharge.

OpenFOAM will usually gives some error messages when starting the first computations. These error messages gives indications of what needs to be corrected in the input files.

# Chapter 5. Input/result files

## 5.1 The file structure

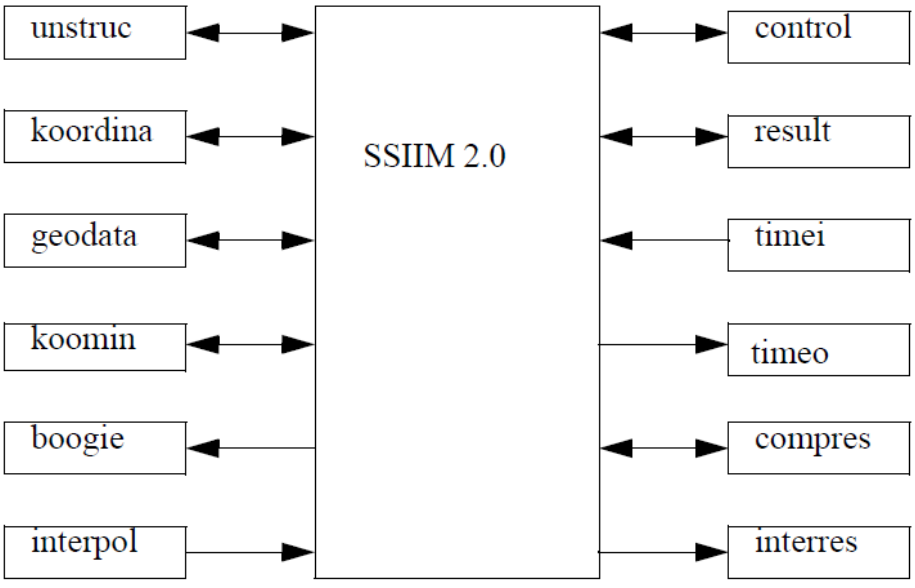
The Windows and Linux versions use the same input files and produce the same result files. The files can be interchanged between the versions. All files are ASCII files, and can be edited with a standard editor. Some of the files can also be made with a spreadsheet.

A flowchart describing the various files are given below. Note that most of the files are only used for special purposes and they are normally not required. Some of the files are output files. The program can produce many of the input files. For simpler cases all the necessary input files can be generated by the program.

Note that the names of the files can not be changed. The best way to run different simulations is therefore to create a sub-directory for each case.

The two main input files are the *control* file and the *koordina* file for SSIIM 1. The *koordina* file contain the grid geometry. The *control* file contain many of the other parameters. The files have to be present when the program starts. If not, a dialog box will emerge and the user is prompted for the main parameters. The program then generates default files. The *control* file can then be edited afterwards, using a standard editor. The *koordina* file may also be generated by a spreadsheet.

For SSIIM 2, the *unstruc* file contains the grid and water discharges. As it can only be produced by SSIIM 2, it is not necessary to be present when the program starts. SSIIM 2 also need the *control* file, but default values will be used if it is not present at the start of the calculation. The *control* file for SSIIM 1 and SSIIM 2 are similar, and much of the content is the same for the two versions. However, there are some data sets that are unique to SSIIM 1 or SSIIM 2.



In the following, each file is described.

## 5.2 The *boogie* file

This is a file that shows a print-out of intermediate results from the calculations. It also shows parameters as average water velocity, shear stress and water depth in the initialisation. Trap efficiency and sediment grain size distribution is also written here. If errors occur, an explanation is also often written to this file before the program stops.

The option *D* on the *F 1* data set will give additional print-out to the file.

Initially in the file it is written how much memory that is occupied by the arrays that are dynamically allocated. To estimate the total recommended memory requirement for SSIIM, add the size of the SSIIM executable to this value.

For SSIIM 1, a table then follows, which shows the cross-sectional area, hydraulic radius, average velocity and water level at the cross-sections that have been used for initializing the water surface. If the option *D* on the *F 1* data set is used, this information is written for all the cross-sections additionally. Then a table of waterlevels for all cross-sections follows. An example is given below:

Loop1,iter,area,radius,velocity,waterlevel: 12 1.002389e+00 1.002389e+00 9.976163e-01 1.002390e+00
Loop1,iter,area,radius,velocity,waterlevel: 11 1.001588e+00 1.001588e+00 9.984146e-01 1.001589e+00
Waterlevel = 1.000398 meters for cross-section i = 10
Waterlevel = 1.000797 meters for cross-section i = 9
Waterlevel = 1.001195 meters for cross-section i = 8

If the *MB-flow* module is used, the residual norms are written. Then follows a sequence of two lines for each iteration of *MB-flow*. An example with four iterations is shown below:

Iter: 5, Resid: 1.69e-05 4.10e-06 2.73e-05 1.17e-04 1.38e-02 1.13e-02
Cont: 9.23e-08, DefMax: 1.65e-03, U,V,W(96,7,20): 6.40e-01 -5.14e-03 5.76e-02
Iter: 6, Resid: 1.62e-05 3.85e-06 2.62e-05 1.10e-04 1.31e-02 1.08e-02
Cont: 9.23e-08, DefMax: 1.56e-03, U,V,W(96,7,20): 6.40e-01 -5.14e-03 5.76e-02
Iter: 7, Resid: 1.57e-05 3.65e-06 2.50e-05 1.04e-04 1.25e-02 1.03e-02
Cont: 9.23e-08, DefMax: 1.48e-03, U,V,W(96,7,20): 6.40e-01 -5.14e-03 5.76e-02
Iter: 8, Resid: 1.51e-05 3.46e-06 2.38e-05 9.86e-05 1.18e-02 9.77e-03
Cont: 9.23e-08, DefMax: 1.41e-03, U,V,W(96,7,20): 6.40e-01 -5.14e-03 5.76e-02

The first line has the word *Iter* at first. Then an integer follows, which shows the number of the iteration. In the example above this runs from iteration number 5 to 9. Then the residuals for the six

equations are shown. The x,y and z velocity equations are first, then the pressure equation and the k and e equation follow. All these must be under 10-3 before the solution has converged.

The second line starts with the word *Cont*. Then a floating point value is shown. This is the sum of all the water inflow and outflow in the geometry. This should be a very low value, typically under 10-7. If a larger value is given, check the boundary conditions. Then the word *DefMax* is written. The residual for the cell with largest water continuity defect is then written. The indexes for this cell are then written, with the velocities in the three directions for this cell. In iteration 9 for the example above, the maximum water continuity defect was 1.41e-3 kg/s for cell $i$=96, $j$=7, $k$=20. The velocity in the $x$-direction for this cell was 0.64 m/s, the velocity in the $y$ direction was -5.14 mm/s and the velocity in the vertical direction was 5.76 cm/s.

## 5.3 The *control* file

The *control* file gives most of the parameters the model needs. The main parameters are the size of the arrays used for the program. To generate the water surface it is necessary to know a downstream water level, together with the water discharge and the Manning-Strickler's friction factor. These parameters are given on the *G 1* and the *W 1* data set in the *control* file. If the *control* file does not exist, the user is prompted for these parameters in a dialog box. The user can then later choose Write *control* in the *File* option of the main menu, and get a *control* file written to the disk (as *control.new*). This can then be edited according to the user's needs. Note that only the most used parameters are written to the *control.new* file.

During the water flow calculations there are several parameters that can be varied. These parameters affect the accuracy and the convergence of the solution. Some of the parameters can be modified while the water flow field is being calculated. A dialog box with the parameters is invoked by choosing Waterflow parameters from the Input Editor choice in the main menu.

The *control* file contains most of the other data necessary for the program. SSIIM reads each character of the file one by one, and stops if a capital letter is encountered. Then a data set is read, depending on the letter. A data set is here defined as one or more numbers or letters that the program uses. This can for example be the water discharge, or the Manning-Strickler's friction coefficient. It is possible to use lower-case letters between the data sets, and it is possible to have more than one data set on each line. Not all data sets are required, but some are. Default values are given when a non-required data set is missing. SSIIM checks the data sets in the *control* file to a certain degree, and if an error is found, a message is written to the boogie file and the program is terminated. Note that if more than one numbers are needed on a data set, these are separated by a space, and not by any other character.

Important note: The *F* and *G* data sets should be given before any other data sets. These data sets should also be ordered according to their number. This is because the data may be checked against the size of the grid.

In the following the data sets for the *control* file is described:

### 5.3.1 The *F* data sets

**F 1** Debugging option. If the character that follows is a *D*, one will get a more extensive printout to the boogie file. If the character is a *C*, the coefficients in the discretized equations will be printed to the *boogie* file.

For SSIIM 1, if the character is *Q*, then the program will allow a grid with negative areas. Otherwise, the program will stop when negative areas occur. The *Q* option can be useful when making a complex grid.

If the character is a *P*, then time-stamps are written to the boogie file from different parts of the program. The purpose is to find parts of the program where improvements can be made in the parallelization.

If the character is an *A*, then the grid is checked for possible errors, and error messages are written to the *boogie* file.

**F 2** Automatic execution possibility. Some parts of the program will be executed directly after the initialisation if a character is placed in this field. The modules will be executed in the order they are given. The possibilities are:

| | |
|---|---|
| *R* | Read the result file |
| *I* | Initialize sediment concentration computation |
| *S* | Calculate sediment concentration |
| *W* | Start the water flow computation |
| *U* | Read the unstruc file |
| *C* | Compute water quality |
| *E* | Write the results from the water quality computation |
| *O* | Read the results from the water quality computation |
| *X* | End the program and exit |
| *M* | Write the result file |
| *H* | Write the unstruc file |
| *Y* | Regenerate the grid |
| *V* | Interpolate the bed levels from the *geodata* file |
| *K* | Compute bed roughness from the *geodata* file and write the *bedrough.new* file |

Example: *F 2 URS*

The program will first read the *unstruc* file, and then read an initial water flow field from the *result* file. Then the sediment transport is computed. This combination is often used for time-dependent morphological computations.

There might be up to 10 letters on the *F 2* data set. The program will then read 10 characters after encountering *F 2* in the *control* file. It is therefore advisable to have a number of characters which are not capital letters on the line after the data set. Lower-case charachters in the *control* file will be ignored.

Note that there must only be spaces between the number 2 and the letters on the data sets. If tabs are used, the letters may not be read.

**F 4** Three numbers are read: First a float, then an integer and then a float. Only the integer is used

this is the maximum iterations for computations of sediment transport in the morphology calculation. The default for the integer is 50. If the user wants a lower value, for example 20, the data set will be

F 4 0.0 20 0.0

**F 6** Coefficients for formula for bed concentration. Default is van Rijn's coefficients: 0.015, 1.5 and 0.3. If one uses this option, the sediment transport formula given in dataset *F 10* must be *R*, which means that van Rijn's formula is used. This is the default formula.

**F 11** Density of sediments and Shield's coefficient of critical bed shear for movement of a sediment particle.

Default: *F 11 2.65 -0.047*

If a negative Shield's coefficient is given, the program will calculate it according to a parameterization of the original curve.

**F 12** Schmidt's coefficient, which is a correction factor for deviation between the turbulent diffusivity and the eddy-viscosity. The factor will affect both the water velocity and the sediment concentration computations. Default: 1.0

If multiple sediment sizes are to be modelled, it is possible to give multiple Schmidt numbers. Example for three sediment sizes:

Example: *F 12 1.0 1.2 1.3*

**F 15** An integer giving a choice between several algorithms for wall laws. Zero means normal rough wall laws. A value of 8 will use a combination of rough and smooth wall laws if the roughness is small. A value of 19 will use smooth wall laws on the sides and rough on the bed.

Default: *F 15 0*

**F 16** Roughness coefficient, $k_s$, which is used on the side walls and the bed, according to the following formula for the velocity profile, *U(y)*:

$$\frac{U(y)}{U_*} = \frac{1}{\kappa} \ln\left(\frac{30\,y}{k_s}\right) \qquad (6.1.3)$$

The distance to the bed/wall is given as $y$, and $U_*$ is the shear velocity. The paramater $\kappa$ is equal to 0.4.

If the *F 16* data set is not present in the control file, the $k_s$ value is calculated from the Manning-Strickler's friction coefficient (van Rijn, 1982) given on the *W 1* data set. The default value of this parameter is 50. Values in the *bedrough* file gives $k_s$ values that override the *F 16/W 1* values for the bed cells. The use of the *F 90* data set also overrides the *F 16* value for the bed by using a formula for the roughness.

**F 18** Density current source. A float is read, and if it is above $10^{-6}$, the sediment density term in the

Navier-Stokes equation is added. The float is multiplied with the density term, so a value of 1.0 is recommended when this term is needed. Default 0.0 (the term is not used).

**F 21** Relaxation coefficient for the Rhie and Chow interpolation. Normally a value between 0.0 and 1.0 is used. When 0.0 is used the Rhie and Chow interpolation will have no effect. When 1.0 is used the Rhie and Chow interpolation will be used normally. Default 1.0.

**F 24** Turbulence model. An integer is read, which corresponds to the following models:

0: standard k-ε model (default)
1: k- ε model with some RNG extensions
3: local k- ε model based on water velocity
4: constant isotropic eddy-viscosity model, value given on *F 72* data set.
5: local k-ε model based on wind shear
6: constant non-istotropic eddy-viscosity model, vertical and horizontal values given on the F 77 data set
7: eddy viscosity = 0.11 * depth * shear velocity (Keefer, 1971)
10: zero-equation used in shallow areas only, k- ε elsewhere.

**F 26** Fraction of compacted sediments in bed deposits. Or 1.0 minus the water content of sediments at the bed.

Default: F 26 0.5 (50 % water)

**F 33** Transient water flow parameters. A float and an integer is read. The float is the time step in seconds. The integer is the number of inner iterations for each iteration. Transient terms will be included in the equations if this data set is present.

**F 36** Options for computation of the vertical elevation of the water surface. Several algorithms are available. Some are described by Olsen (2015).

An integer is read. The most used algorithms is based on the computed pressure field. This algorithm is used if the integer is 2. The cell given on the *G 6* data set will be kept fixed as a reference level.

More algorithms can be used: *F 36 3*: the initial water surface is moved up/down equally in all cells according to downstream changes in water surface specified in the timei file.

*F 36 7*: the water surface elevation is computed solving a partial differential equation for the local slope as a function of the pressure gradients to the eight neighboring cells. This is described as the IPDA (Implicit Pressure Difference with Adaptive grid) method by Olsen (2015). Note that some variations of the IPDA algorithm are invoked by the F 278 data set.

The IDWA (Impicit Diffusive Wave, Adaptive grid) method (Olsen 2015) is invoked by *F 36 9.*

The CGA method in SSIIM 2 is invoked by *F 36 15*. One of the example cases on the SSIIM web page uses this option.

Default: *F 36 0* (no change in the water surface elevation)

**F 37** Transient sediment computation. An integer is read. If this is 1, the transient sediment computation (TSC) algorithms will be used. This option is always used when doing time-dependent computations of sediment transport, and always when computing changes in bed levels. If the integer is 2, then a different algorithm is used for the bed cells, where the sediment concentration formula is converted into an entrainment rate. This can give slightly smaller bed movements where the bed sediment concentration is not in equilibrium.

Default: *F 37 0*

If the integer is 3, then an algorithm is invoked where more than two bed layers can be used. The default is to have two layers, an active and an inactive layer. This is used for *F 37 1* and *2*.

**F 41** Sediment thickness in meters. Default: 10 meters.

**F 42** Level of non-erodible material. This data set can be used instead of the *koomin* file.

**F 47** Bed interpolation parameter in meters. One float is read, which is used in the bed interpolation routine that are called from the *GridEditor*. If the points given in the *geodata* file are located a horizontal distance smaller than the interpolation limit, than the interpolation routine will use the exact value of the *geodata* point instead of interpolating from surrounding points. Default: 0.05 m.

**F 48** Parameter for print-out of special files and interpolation of results. **Note that a more detailed print-out is possible by using the F 329 option instead.** An integer is read. If 0, then a normal *result* file will be written when this routine is invoked. If higher values are given, the program will not write the *result* file. It will search for the *interpol* file and use this file to write other files. If the integer is between 1 and 4 an *interres* file is written. If the value on the *F 48* data set is 4, the bed levels will be written to the *interres* file. If 2, the velocities, k and ε will be written to the file. If 3, then water quality parameters will be written. If the integer is 5, the water velocities in the surface cell is written to the *interres* file. If the integer is 14, the value of the average velocity times the depth is written to the *interres* file. If the integer is 10, a three-dimensional ParaView file is written. If the integer is 12, a two-dimensional ParaView file is written from the cross-section defined in the *interpol* file, with the water velocities normal to the surface. If the integer is 19, 2D Paraview files are written for both the coarse and the nested grid. If the integer is 25, a *habitat* file is written.

The options 88, 89, 91 and 92 will do the same as options 8, 9, 11 and 12, respectively, but the *result* and *bedres* files will not be written simultaneously.

Default *F 48 0.*

**F 50** Number of water quality parameters. Default F 50 0.

**F 51** Coriolis parameter.

Example: *F 51 0.0001263* (lakes in South Norway)

**F 52** Wind forces on the surface of the lake. Three floats are read. The first float is the magnitude of the wind speed in meters/sec.10 meters above the water level. The second and third floats are components of the unity vector in the *x* and *y* direction. Note that the vector sum must be unity.

Example: *F 52 4.0 -1.0 0.0*

Note for time-varying wind, the values in the *timei* file overrides the values on this data set.

**F 53** Print iterations. Four integers are read, which gives the interval for when printout to files are done. The first integer applies to the residual printout to the *boogie* file. The second integer applies to writing the *result* file. The third integer applies to writing data to the *forcelog* file. The fourth integer applies to when data is written to the *timeo* file.

Default: *F 53 100 100 1 1*

This means that for example the *result* file is written each 100th iteration.

**F 54** A float is read, which is a limit for the residual during the transient calculation. When the maximum residual goes below this value, the inner iterations end, and a new time step starts. This is recommended when doing transient computations. Default $10^{-7}$

**F 56** Sand slide algorithm. An integer and a float is read. The float is tan(angle of repose). The integer is the number of iterations the algorithm will go through the whole grid. If 200 is chosen, a particular algorithm is used which has given better results than the others.

Default: *F 56 0 1.0* (not used)

Typical values if used: *F 56 200 0.62*

Note that the angle of repose for the sediments is lower for sediments submerged in water than in air. The algorithm will give a sediment continuity defect if used with more than one sediment size. If multiple sediment layers are used, the option *F 56 202* should be used. This is made to give correct continuity also with multiple sediment sizes.

**F 59** Number of iterations in the Gauss-Seidel procedure. This is an integer which will affect the convergence and speed of the program. A lower value will increase the number of iterations pr. time, while slowing the relative convergence pr. iteration. For some cases, a lower value has given decreased computational time. Note that if the TDMA solver (*K 10* data set) is used, then the *F 59* data set will have no effect. Default: 10.

**F 60** Correction of van Rijn's formula for distance from the center of the bed cell to the bed. The data set is used to invoke inclusion of the extrapolation of the Hunter-Rouse sediment distribution when the vertical height of the center of the bed cell is different from what van Rijn subscribed.

Default: *F 60 0 0* (no use of the extrapolation)
Example: *F 60 1 0* (the extrapolation is used)

**F 62** Integer to invoke time-dependent calculation of water quality parameters, if 1.

Default: *F 62 0*

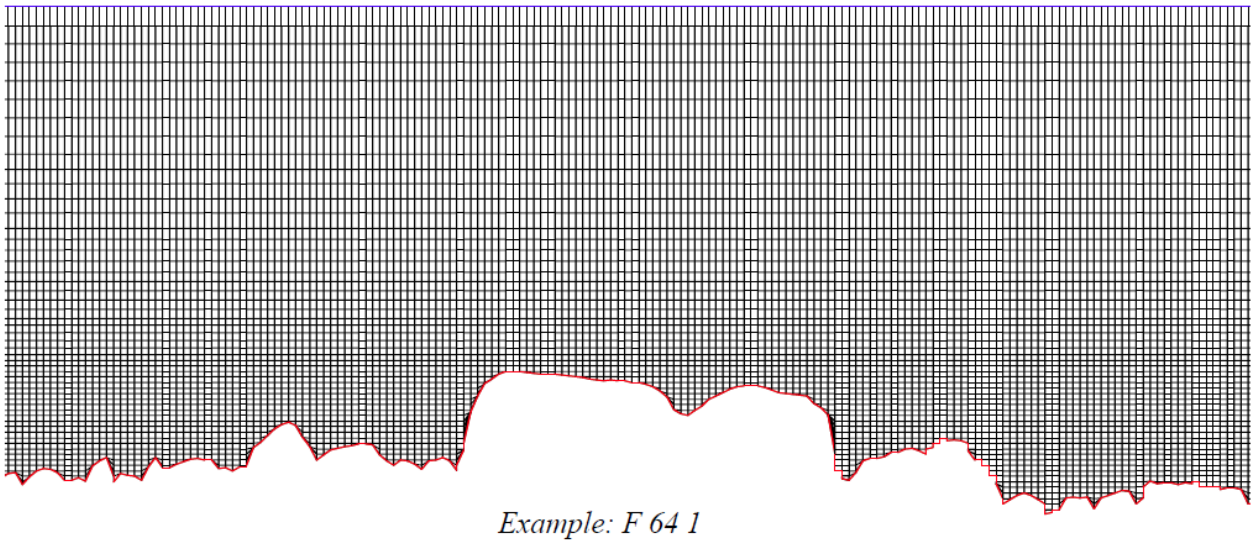
**F 63** Second-order upwind scheme for water quality constituents if the integer 1 follows.

Default: *F 63 0*

**F 64** Choise of algorithm to generate the grid lines in the longitudinal and lateral direction.

Default: *F 64 11*

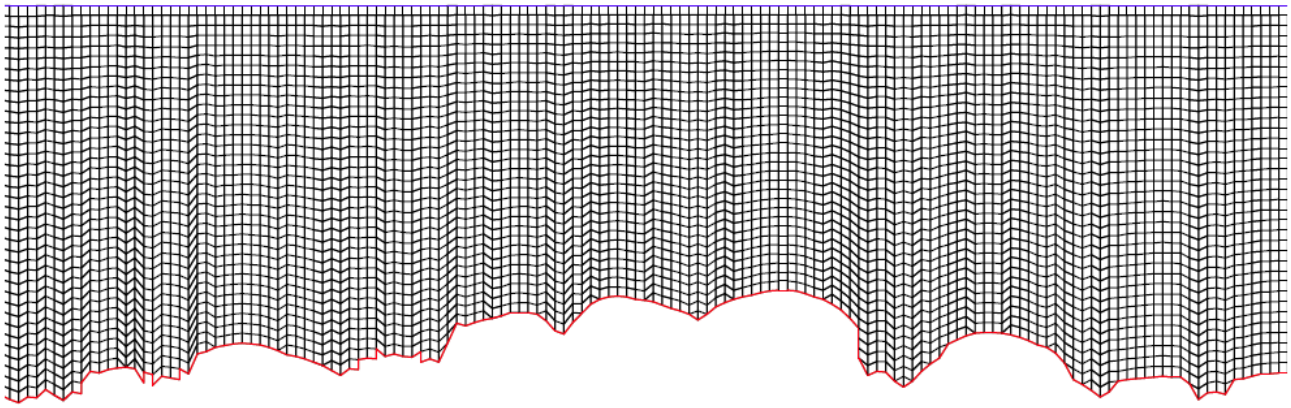
An integer is read. If it is 0, completely horizontal grid lines will be generated. This is typically used when modelling lakes, especially with density gradients.



*Example: F 64 1*

*The F 64 1* option will also create horizontal grid lines, but the bed cells will not be horizontal. Tetrahedral cells and hexahedral cells with non-horizontal lines will be used along the bed.

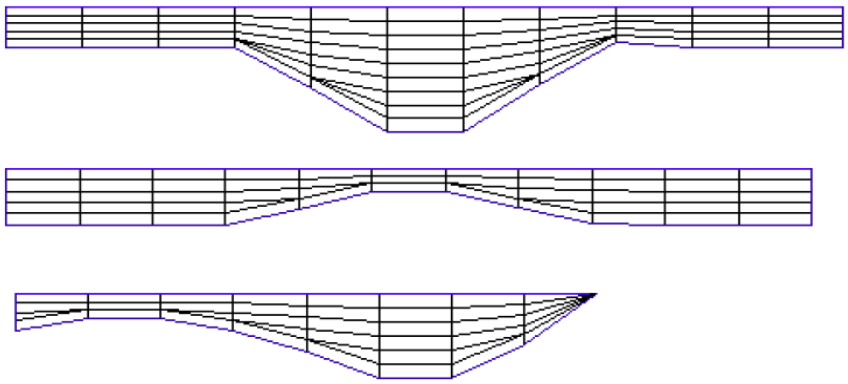
*The F 64 2* option will be similar to *F 64 1*, but all the non-vertical grid lines may be non-horizontal. An example is shown in the figure below.



*Example: F 64 2*

For sediment transport computations in rivers, the most tested option is *F 64 11*. This will give a body-fitted grid with priority to hexahedral cells close to the bed. The hexahedral cells will give superior performance compared to tetrahedral cells. Since most of the sediment is transported close to the bed, it is important that the bed cells are hexahedral in sediment computations.

*Example:*
*F 64 11*



The option 13 is similar to 11, but has different generation criteria for wetting and drying. The two values on the *F 94* data set is then used. If the *F 64 13* data set is used, then the cells will be generated initially just like using the *F 64 11* option. However, cells will not be generated in areas that were dry in the previous iteration, unless the cell depths are above 0.0 meters. The value 0.0 meters can be changed by giving a different number on the *F 162* data set.

See also Chapters 2.2 and 3.2 for generation of grids.

**F 65** Maximum grid size for the unstructured grid. SSIIM 2 has to allocate the arrays before the grid is read. Because it is possible to expand the grid after it is read, it is necessary to give the grid array sizes in the input file.

Five integers are read. The first integer is the maximum number of grid cells in the grid. The second integer is the maximum number of surfaces in the grid. The third integer is the maximum number of grid corner points. The fourth integer is the maximum number of surfaces in connection between blocks. The sixth integer is the maximum number of connection points, used in the *GridEditor.*

Default: *F 65 50000 80000 50000 10000 1000*

If the grid has been made, and the grid size will not increase later, it is possible to read the size in the *unstruc* file, and modify the *F 65* data set so that the program does not allocate more memory than necessary.

**F 67** Temperature calculation with feedback from water density on the water flow calculation if it the integer 1 follows.

Default: *F 67 0*

**F 68** Parameter for choice of water flow computation. An integer is read.

Default: *F 68 0*

If the parameter is 2, the transient sediment computation will not re-compute the water flow field after an update of the bed. This means a quasi-steady situation is modelled.

If the parameter is 1, the three-dimensional calculation of the water flow will be done using the hydrostatic pressure assumption. This means that the Navier-Stokes solutions will not be solved in the vertical direction, and vertical velocities will be found by the continuity equation. This option causes an extra grid block to be made. The extra block is the last block and is a depth-averaged version of the whole grid. Note that this parameter have to be the same when calculations are done as when the grid was created.

**F 70** Wall laws removal option. An integer is read. If 1, the wall laws will be removed from the side surfaces. This is typically used if a 2D width-averaged grid is used. If the integer is 2, the wall laws will be removed from the side and the bed surfaces.

Default: *F 70 0*

**F 71** Turbulence decrease due to density stratification. An integer is read. If 1, the turbulent eddy-viscosity is decreased according to the Richardson number and the procedure given by Rodi (1980).

Default: *F 71 0*

**F 72** Minimum turbulent eddy-viscosity. A float is read. The eddy-viscosity for water flow, turbulent kinetic energy and water quality constituent calculations will not be decreased below this value.

Default: *F 72 0.001*

**F 73** Choise of wind friction formula. This formula gives the shear stress of the water surface as a function of the wind. The options are:

0: van Dorn (1953), $c_D$ = 1.0x10-3 under 5.6 m/s, increases over 5.6 m/s
1: Bengtsson (1973), $c_D$ = 1.1x10-3
2: Wu (1969), three non-continous bands of $c_D$ as a function of wind speed

Default: *F 73 0*

**F 76** Turbulence reduction factors. Three floats are read. The two first floats are multiplied with the eddy-viscosity in the horizontal and the vertical direction respectively, thereby reducing and possibly creating a non-isotropic eddy-viscosity. The third float is related to the decreased eddy-viscosity because of density stratification. If it is zero, the eddy-viscosity decrease factor will be applied isotropically. If it is unity, the factor will only be applied to the vertical eddy-viscosity. For values between zero and unity, the factor will partly be applied only vertically and partly isotropically, creating a more or less non-isotropic eddy-viscosity.

Default: *F 76 1.0 1.0 0.0*

**F 77** Constant non-isotropic turbulent eddy-viscosity. Two floats are read, the horizontal and the vertical eddy-viscosity. Note that the *F 24* data set must have the parameter 6 for this data set to be used.

Default: *F 77 1.0 1.0*

**F 78** Bed load vector parameters. An integer and a float is read. The bed load on a transverse sloping bed may not move in the direction of the water velocity close to the bed. If the integer is 1, an algorithm taking this into account is used. Note that this is fairly untested yet for SSIIM 2.

In SSIIM 2, an integer 10 can be given as the first integer. The algorithm will then use the average sediment diameter instead of the diameter of each size. In SSIIM 2, this is the only option for using the algorithm with multiple sediment sizes.

**F 81** Maximum number of time steps in the *timei* file. An integer is read.

Default: *F 81 200*

**F 82** Parameters to decrease the eddy-viscosity as a function of the water density gradients. The eddy viscosity from the k-ε model is multiplied with a factor given in the following formula (Rodi, 1980):

$$\nu_T = \nu_{T,0}\left[1+\beta\left(\frac{-g}{\rho}\frac{\frac{\partial \rho}{\partial z}}{\left(\frac{\partial U}{\partial z}\right)^2}\right)\right]^{\alpha}$$

The eddy viscosity is denoted $\nu_T$, a and *b* are a constants, ρ is the density of the water/sediment mixture, *U* is the velocity, *z* is the geometrical distance in the vertical direction, *g* is the acceleration of gravity. The data set gives the values of the *a* and the *b* constants in the equation. Four floats are read.

Default: *F 82 -0.5 10.0 -1.5 3.33*

The two first numbers are a and b for the velocity equations. The two last float are the a and b constants for the turbulence.

**F 83** Coefficients in van Rijn's formula for bed load sediment transport. Four floats are read.

Default: *F 83 0.053 2.1 0.3 1.5*

**F 84** Flag to indicate the sediment transport formula. An integer is read:

0: Suspended load formula by van Rijn
1: Bed load formula by van Rijn
2: Both suspended and bed load formulas by van Rjin
3: Wu's formula
5: Meyer-Peter and Mullers formula
6: Engelund-Hansens formula
7: Hillebrands formula
8: Modified (critical shear) Engelund-Hansens formula
9: Modified (critical shear + d50) Engelund-Hansens formula
11: Modified (d50) Engelund-Hanses formula

Default: *F 84 0*

The Engelund-Hansen formula is developed for uniform sediment sizes. Used on each fraction for non-uniform sediments, it will give a different total value than if d50 had been used. The modifications take this into account.

The Engelun-Hansen formula will also give a sediment transport capacity above zero if the shear stress is below the critical Shields value. The modification will correct this.

**F 85** Flag indicating that the computation should not stop if water continuity is not satisfied. This is invoked if 1 is given.

Default: *F 85 0* (program will stop if there is a significant water continuity defect)

**F 86** Minimum grid corner height. This is used to prevent very small grid cell heights. If the corner grid cell is lower than this value, it is set to zero. Note that this only works for the *F 64 5/11/13* options.

Default: *F 86 0.001* (1 mm)

This information can alternatively be given on the *F 94* data set. Then the parameter on the *F 86* data set will be equivalent to the second data set on the *F 94* data set. The first parameter on the *F 94*

data set will be set to 1/4 of the *F 86* value.

**F 87** Parameter for number of grid cells, *n*, in the vertical direction as a function of water depth, *y*. The value is the *p* parameter in the following formula:

$$n = n_{max} \left( \frac{depth}{depth_{max}} \right)^{p}$$

The $n_{max}$ number is the third integer on the *G 1* data set. The maximum depth is computed when the program starts. Since it may vary over time, it is also possible to give it in on the *F 209* data set.

Default: *F 87 0.6*

**F 90** Roughness option. An integer is read, giving several options how the roughness in the wall laws should be calculated:

0: The value on the *W 1* or *F 16* data set is used.
1: The bedrough file is used
2: The roughness is calculated from the bed grain size distribution (3.0 * $d_{90}$)
3: The roughness is calculated from $d_{90}$ and the bed form height, using the formula below
4: Same as 3, but the critical shear stress for sediment movement is reduced so that only the grain roughness effect is taken into account.

Default: *F 90 0*

The bed form height, Δ, is computed by the formula of van Rijn (1987). The effective roughness is computed as (van Rijn, 1987):

$$k_s = 3 d_{90} + 1.1 \Delta \left( 1 - e^{\frac{-25\Delta}{\tau_c}} \right)$$

where *l* is the bedform length, calculated as 7.3 times the water depth.

Note that van Rjin's equations for bed form roughness was developed on mostly uniform sediments. For non-uniform sediments, the bed forms will be smaller.

**F 92** Algorithms to reduce velocities in cells with small depths. An integer is read:

1: Wall laws are used in more than one cell
2: A drag formula is used for velocities where the roughness is higher than the cells

Default: *F 92 0* (Algorithm is not invoked)

**F 94** Minimum grid corner height and maximum grid corner height for generation of one cell.

Two floats are read. The first float is used to prevent very small grid cell heights. If all the four corners

of a 2D cell is lower than this value, the cell will not be created. It will be a dry cell.

The second float gives the depth distinguishing if one more more than one cell is generated. If one of the four corners of a cell has a greater depth than the float, more than one cell over the depth will be generated. If not, this will be a 2D cell. Note that the algorithm only works for the F 64 11 and 13 options. F 64 11 is the default option.

Example: *F 94 0.001 0.01* (default)

Grid cells with all four corner depths under 1 mm will not be generated, and all cells with all four corner depths under 1 cm will be 2D (only one cell over the depth).

**F 99** Integer to determine how often the grid is regenerated for time-dependent sediment flow calculations with moveable bed.

Default: *F 99 1*

Example: *F 99 10*

The grid will only be regenerated for every 10th time step.

Sometimes it is necessary to use *F 99 1* to be able to read the *bedres* file together with the other files, when starting from a previous computation.

**F 102** Integer to invoke an algorithm to change the shape of the grid cells close to the boundary. If the integer is 1, the algorithm is included. This algorithm is recommended for wetting/drying computations.

Default: *F 102 0*

**F 104** Integer invoking an algorithm to prevent crashes when a single cell is formed. The algorithm is only invoked if the integer is above zero. The algorithm will add a value to the $a_p$ coefficient in the pressure-correction equation of SIMPLE. This will only be used in the cells that have no neighbour cells. The value added is the integer multiplied with the volume of the cell.

Default: *F 104 0*

**F 105** An integer is read, giving the number of iterations between each update of the water surface elevation for time-dependent calculations.

Default: *F 105 10*

**F 106** A float is read, giving the thickness of the upper active sediment layer. The default value is equal to the maximum sediment grain size diameter on the *S* data sets.

**F 109** Parameters in Brook's formula for reduction of the critical sediment particle shear stress when the bed slopes.

Default: *F 109 1.23 0.78 0.2*

The two first floats are the inverse of tan(θ) for uphill and downhill slopes, where θ is a kind of angle of repose for the sediments. θ is actually an empirical parameter based on flume studies. The third float is a minimum value for the reduction factor.

**F 110** Parameters for limiting the effect of the Hunter-Rouse extrapolation for sediment concentration computed in a reference level different from what van Rijn subscribed. The parameter will only have effect when used with the *F 60* data set.

Default: *F 110 2.0 0.01*

**F 112** An integer is read. If it is 1, the program will regenerate the grid automatically right after it has read the *unstruc* file. The regeneration will be made from the water levels given in the *koordina* file. The option is used when computing wetting/drying in a situation where the initial water level is lower than what it will be later. In other words, the program is started with dry cells. The program needs to know the grid layout of the areas that will be wetted, so the *unstruc* file needs to cover the whole geometry. If the initial computational domain will expand horizontally at a later time, than the initial water level can be given in the *koordina* file, and the program will start with a grid based on this.

Note that if the *G 6* option is used in this case, the indexes refer to the grid in the original *unstruc* file, and not on the regeneration after the *koordina* file is used. Also, the inflow/outflow specification is done on the grid in the original *unstruc* file.

One of the most common problems for new SSIIM 2 users has been that they write the *unstruc* file with dry cells. The latest versions of SSIIM 2 will therefore not allow the *GridEditor* or the *DischargeEditor* to open if the *F 112* data set is used in the *control* file.

**F 113** Algorithms to stabilize the solution in very shallow regions close to the side walls. The algorithms use different interpolation algorithms from the center of the cells to the cell surfaces. Different algorithms have been tested, using integers from 1 to 7.

3,4: Using second-order interpolation instead of third-order interpolations for pressure gradients

5: Setting the Rhie and Chow term to zero for 2D cells in shallow areas. Shallow defined as depths below the values given on the first value of the *F 94* data set.

7: Flux-limiter: the extra term from the Rhie and Chow interpolation should not be more than 20 % of the linear interpolation term.

Note that these algorithms are not tested extensively

Default: *F 113 0* (algorithms not used)

**F 114** An integer is read. If it is 9, 10 or 20, special algorithms to compute the pressure gradients are used. This can give a more stable solution where there are very shallow areas inside the main grid. A similar routine is also used on the pressure-correction gradients in the SIMPLE method if the integer is 10.

Default: *F 114 0* (algorithms not used)

**F 115** Algorthms for the *vegdata* file. An integer is read. If it is 1, the program will read the *vegdata* file. Please see Chapter 5.8 for more information about this file.

**F 131** Cohesive parameters. And integer and a float is read. The integer determines the sediment size. The float is a critical shear stress for erosion of each particle size. Multiple *F 131* data sets are given for multiple sediment sizes. Default: 0.0 for all sizes. Note that this number is only used to determine the critical shear stress for erosion of a particle. It is not used in the sediment transport formulas.

**F 132** Maximum Froude number for update of water surface. A float is read. This is a stabilizing depth-limiter algorithm for the water surface computation. The water level is limited in size so that high Froude numbers do not emerge. The critical Froude number is given on the float. Default: 0.0, meaning the algorithm is not used. Note that this algorithm may introduce unphysical water levels, with corresponding instabilities.

**F 134** Number of bed layers. This is used in an algorithm using several sediment layers under the bed. The purpose is to compute for example sediment composition in a delta. The *F 37 3* option should be used when the *F 134* integer is larger than 2.

**F 138** Minimum value of the turbulent kinetic energy. A float is read. Default: $10^{-20}$.

**F 139** Two floats are read. Minimum values of u+ in the wall laws for the velocity and turbulent kinetic energy.

Default: *K 139 1.0 1.0*

Better stability has been obtained by using higher values than 1.0, for example 3.0. The questions is if the bed shear stress then will be correct. This needs to be tested for each case.

**F 141** A limiter for epsilon. A float is read. This is the maximum value epsilon will get. Default: 100.0. (Minimum value for epsilon is hard-coded to $10^{-16}$)

**F 144** Bed form smoothing algorithm. An algorithm is invoked to smooth the bed as a function of the bed form characteristics. An integer and a float is read. The smoothing is used if the integer is above 0. The default value is 0.

**F 147** Parameters for use in extrapolation of initial values to newly wetted cells.

Default: *F 147 20 0 1 0.2 1.0 1.0* .

The first integer gives how many iterations the extrapolation algorithm should be used. This should be larger than the number of cells in one direction in the wetted area. The second integer determines which of two interpolation algorithms are used when transferring variables between two grids. The default algorithm (0) uses a linear interpolation based on the vertical elevation of the cells. An alternative algorithm (1) uses the cell indexes. The third integer invokes the part of the algorithm that extrapolates over multiple cells for each wetting situation. The fourth number, a float, gives a relaxation factor for the velocity values. The fifth and sixth number, both floats, give relaxation factors for k and epsilon.

During initial testing, we have not found any improvements in the results by using values different than the default.

**F 148** Maximum slope of the water surface. A float is read. If the number is above zero, an algorithm is invoked that prevents the water surface to be steeper than the value given. Default -0.1 (algorithm not used).

**F 149** Non-isotropic turbulence parameters. Two floats are read. The first float is multiplied with the isotropic diffusion to produce the vertical diffusion used by the program. The second parameter does the same in the horizontal direction.

Default: *F 149 1.0 1.0*.

**F 154** Smoothing algorithm for the water surface. An integer and a float is read. Smoothing is done if the integer is 1. The smoothing is done by taking the average of the four neighbour values and multiplying this with the float parameter on the data set. Then, (1.0- the parameter) is multiplied with the old value and added. This gives the new value.

If the integer is 3, dry areas will not be smoothed. If the integer is 1, also dry areas will be smoothed. Integers 2 and 4 will give the same as 1 and 3, respectively, but the boundary will also be smoothed.

**F 156** Option to pause the program during a computation. This is used for debugging purposes. An integer is read. The following options are possible for the pause location:

1. Before the start of regenerating the grid
2. After the regeneration of the grid
3. Before the velocity and flux corrections in the SIMPLE algorithm
4. After the velocity and flux corrections in the SIMPLE algorithm
5. Before interpolating the variables to the new grid
6. After interpolating the variables to the new grid
7. At the end of a time step computing the bed level changes (*F 37 1/2*)

The option can be used in combination of the *Pause* option in the *Compute* menu of the program.

**F 159** Algorithms to improve stability by avoiding grid problems. Five integers are read, for five different groups of algorithms. The algorithms are invoked if the integer is non-zero. The algorithms are only used when the parameter on the *F 64* data set is equal to 8, 11, 13, 38 or greater than 100.

The first algorithm tries to remove dead-end channels that are only one cell wide. A maximum of 30 cells can be removed.

The second integer invokes different algorithms dealing with the problem of ridges between wet cells. Some algorithms try to give better connection between two neighbour column of cells that are separated by a high ridge. This is done by increasing the water depth. Several algorithms are used with different depths. An integer of 1 sets the depth to 80 % of the first parameter of the F 94 data set. An integer of 2 or 3 sets the water depth to minimum the value of the first parameter on the F 94 data set. An integer of 4 does the same as 1, but uses a value of 200 % of the first F 94 parameter, instead of 80 %. An integer of 3 sets the number of vertical grid lines in the corners between the cells to 1 if one of

the corners have a negative depth and the sum of the two corners is smaller than 10 % of the first parameter on the F 94 data set. An integer of 7 does the same thing as 1, but uses 100 % of the second parameter on the F 94 data set instead of 80 % of the first. Other algorithms sets internal walls in the ridges. The integer is then set to 9 or 10.

The third algorithm will try to remove holes in the grid, where there is only one cell with no connections to side neighbours, only with its neighbours below and above. This is in a wetted area, where in 2D, the neighbours exist.

The fourth algorithm will remove single wet cells with only dry neighbours in 2D.

The fifth integer invokes different algorithms to increase the water depth in partially dry cells, by lowering the bed levels. So far, none of these algorithms have been successful. An integer of 1 lowers the bed level to the first value given on the F 94 data set. This also happens if the integer is 2, but then multiple cells in the vertical direction is allowed, also on side walls. If the integer is 3, then the depth is increased so that the bed slope is not above 20 degrees. If the integer is 5, then neighbours are disconnected if the area between them is smaller than a small number.

*Default: F 159 1 0 0 1 0*

**F 160** Flag to decide which algorithm is used to compute the available sediment at the bed when deciding the limits of sediment concentration. An integer is read. If it is 0, the available sediments will be equal to the bed cell area times the depth of the sediments. If the integer is 1, then also neighbour cells will be included.

Default: F 160 0.

**F 162** Value to determine whether a cell is generated or not in regions where the bed was dry in the previous time step. Default 0.0 meters. The parameter only works in connection with the *F 64 13* option.

**F 163** Flag to decide which algorithms are to be used to limit the movement of the bed. An integer is read. Three options are possible:

*0*. The bed level will not be allowed to move below the movable bed, but it will be allowed to move above the water surface.

*1*. The bed level will not be allowed to move above the water surface or below the limit of the movable bed.

*2.* The bed level may move above the water level or below the limit of the movable bed.

**F 164** Different versions of OpenMP multi-core solvers. An integer is read, specifying the number of the solution algorithm.

The *F 164 31* version will be slower than the default, but is designed to give consistent results when running the program two times with the same input files.

Default: F 164 0.

**F 165** Grid numbering direction. When the unstructured grid is made, the indexing of the cells start at the lower left corner, or the first point marked in the *GridEditor*. One integer is read on this data set. If the integer is 1, then the cell indexing will start on the corner diagonal opposite in the block. This is mostly useful for debugging purposes.

Default: *F 165 0*

**F 166** Regeneration of grid after water surface update. An integer is read. If it is 1, then the grid will be regenerated after each time the water surface is updated, also if time-dependent sediment transport is computed. This is usually not done, as the bed changes are normally recomputed more frequently than the water surface changes, and the grid is always regenerated after each bed change.

Default: *F 166 0*

**F 168** Multi-grid solver for the pressure-correction equation. An integer is read, and this is the number of levels in the grid nesting. If the integers is 0, the algorithm is not used. Note that the *K 5* data set also needs to be used to get the algorithms to work.

Default: *F 168 0*

**F 169** Hiding/exposure parameters. An integer and a float is read. The integer decides which algorithm is to be used.

First integer is 2: Buffington and Montgomery's method
First integer is 4: Wu's (2000) method

**F 174** Cyclic boundary conditions. Two integers are read. The first integer applies to the water flow computation, and the second to the sediment concentration computation. If the integer is positive, say for example 500, then the inflow boundary values will be set equal to the outflow values for each 500th iteration. If a negative number is given, say -8, then the program will run to convergence, then update the inflow boundary condition and restart the computation. This will be repeated 8 times.

Default *F 174 0 0* (no cyclic boundary condition used)

**F 178** Smoothing algorithms for the free surface, used for the F 36 parameter being 1 or between 11 and 39. An integer is read, which can be between 1 and 4. Each integer corresponds to a different smoothing algorithm.

Default: *F 178 0* (algorithm not used)

**F 179** Upwind function used for the *F 36* parameter being 1 or between 11 and 39. Two integers are read. The algorithm is invoked if the first integer is 1. The second integer will cause a reduction the upwind effect if it is 1.

Default: *F 179 1 0*

**F 182** Reduction of critical bed shear stress due to a sloping bed. An integer is read. If it is 1, the same

algorithm of Brooks (1963) is used. If the integer is 2, then the algorithm is only used for cells that have a dry neighbour. If the integer is 3, then the lateral slope in the formula will be the maximum slope. The same formula is used only for the side cells if the integer is 4. If the integer is 5, the empirical formula by Dey (2003) is used. It is only used for the side cells if the integer is 6. The formula by Lane (1955) is used if the integer is 7 or 8. If the integer is 8, the formula will only be applied to the border cells.

Default: *F 182 0*

**F 185** Damping of turbulence close to the water surface. An integer and a float is read. If the integer is 1, a formula similar to the wall laws will be used for the epsilon equation. This will give increased values of epsilon at the water surface and turbulence damping. The float is an empirical coefficient in the damping function. Typical values: 0.0046-0.43.

**F 187** An algorithm to reduce the water level gradients at the borders of the geometry. The algorithm is used in connection with *F 36* 2, 4, 7, 8, 9. An integer is read. If it is 1, the algorithm is used.

Default: *F 187 0*.

**F 188** Integer determining which block is to be computed for sediment transport. Options:

0: All blocks (default)
-1: only nested blocks
n: only block no. n.

Example: *F 188 3* : Only compute sediment transport for block no. 3.

Default: *F 188 0* (all blocks)

**F 189** Large roughness algorithms. Special algorithms designed for situations where the bed roughness is larger than the vertical grid cell size close to the bed. An integer and a float is read. The integer determines which algorithm is used, and the float is a parameter in the algorithm.

Several algorithms and approaches have been tested, but not extensively. Currently, the most promising algorithm seems to be an immersed boundary method with a linear velocity profile: *F 189 5 0.5*, but much more work needs to be done.

**F 191** When connecting blocks in the *GridEditor*, the graphics pointer have to be placed within a certain accuracy. The default is 0.01 mm. For large geometries, this value may be raised to get a successful connection.

**F 192** The parameter decides how many cells are used in the vertical directdion in the nested grid in relation to the coarse grid. The parameter is the ratio of:

*number of cells in the vertical direction for the nested grid*

to

*number of cells in the coarse grid*

Default: 1.0

**F 200** Residual norms for $k$ and $\varepsilon$. An integer and two floats are read. The first float is the residual norm for $k$ and the second is the residual norm for $\varepsilon$. The values are used instead of the average inflow values if the integer is above zero.

**F 201** Parameters for the *vegdata* file. Three integers are read. The first is the number of variables in the *vegdata* file. The second is the number of vertical elevations. The third is the number of time steps in the file.

This data set has to be present in the *control* file if values for multiple times are to be used in the *vegdata* file.

The values on this data set will be compared with the same values in the *vegdata* file, and an error message is produced if they are not the same.

Default: *F 201 1 4 1*

**F 202** Logarithmic inflow velocity profile. An integer and a float are read. If the integer is above zero, then the inflow sections with have a logarithmic profile in the vertical direction. The float is the roughness value that will be used in generating the logarithmic profiles. Note that all inflow profiles will be logarithmic if this data set is used. If the parameter is not used, a uniform profile is used.

**F 205** Upwind option for movement of sediments. An integer and a float is read. Depending on the integer value, different upwind options are used when moving the bed elevation changes from the center of the bed cell to the corner. The option *F 205 15 0.0* will move negative bed movements more in the direction of the higer corner and positive bed movments more in the direction of the lower corners, thereby smoothing instabilities in the bed.

Default: *F 205 0 0.0* (algorithm not used)

**F 206** Maximum number of processors used for the parallel versions of SSIIM. An integer is read, which gives the maximum number of processors. If it is above the number of processors available on the computer, then the maximum available processors are used. The actual number of processors used is written to the *boogie* file.

Default: *F 206 0* (parallellized algorithms not used)

**F 208** Order of the discretization of the time-term. An integer is read. If this is 1, then a second-order backwards scheme is used for the time term. Default value: 0, meaning a first-order backward Euler method is used. This data set is usually only used if modelling large eddies (URANS), as the solution will become much more unstable.

**F 209** Maximum depth in the geometry for scaling grid generation (meters). The default value is the largest depth in the geometry. The value is used for deciding how many grid cells there will be over the depth for a given location in the geometry. See formula in the explenation of the *F 87* data set.

**F 212** Special post-processing print-out. An integer is read. Depending on the value and which SSIIM version, varying extra print-out is given. The print-out is done when the *result* file is written.

12. Writes a file *cellvol*. This file uses the *geodata* file and the grid to specify how much of the cell volume is located at given levels. 100 values of levels in meters and volume are written to the file. The information can be used to make a capacity curve for the lake/reservoir.

15: Writes an extra line to the boogie file with the average vertical cell size in the grid (meters), together with the maximum and minimum vertical grid elevation.

16: Writes and extra line to the boogie file with the average height/length ratio of all the cells in the grid.

**F 218** Debug information. Two integers is read. When the program has done as many iterations as the first integer, a large number of debug information will be written to the boogie file from the cell which has the same number as the second integer.

**F 219** Parameter to invoke an automatic restart after the program has crashed. The relaxation parameters are lowered before the restart and the main water flow parameters are reinitialized. An integer is read. If it is above zero, the algorithm is used.

**F 221** Convergence stopping of the program. In a steady situation, the program stops when the residuals are below 0.001. In a time-dependent case, the program will not stop if the residuals are below 0.001. However, if time-terms are used to improve convergence of a steady situation, the program should still stop.

An integer is read. If it is 0, the program will not stop in time-dependent computations, when the residual is below 0.001. If the integer is 1, the program will stop.

Default: *F 221 0*

**F 222** Downstream depth algorithm. An integer is read, which can be 0 or 1. If 1, an algorithm is invoked to try to prevent the downstream bed level to rise to heights where it will block the outflow.

Default: *F 222 1* (the algorithm is used by default)

**F 224** One float is read. In a time-dependent water flow computation with a free surface, the free surface will only be updated if the residuals for the Navier-Stokes equations are below 100.0. Using this data set, the value can be changed as the float is used instead of 100.0.

Default *F 224 100.0*

**F 225** Two floats are read, which are used to scale the values in the *geodata* file. The first value is a factor that is multiplied with all the x, y and z values. The second value is added to all the x, y and zvalues. This can for example be used to model a physical model test if measurements of the prototype exist, or the other way around.

Note that the scaling is done only when reading the values. When editing the *geodata* points in the GridEditor, and then writing a new *geodata* file, the scaled values are written. Reading the newly generated *geodata* file back in again and keeping the *F 225* data set gives a double scaling of the points.

**F 233** Algorithm to use a depth-averaged pressure field to compute the water surface elevation changes instead of the pressure in the surface cells. An integer 6 can be used. An integer 7 will use an algorithm similar to 6, but with smaller elevation changes due to the inclination of the surface of the cell.

**F 234** Algorithm to modify the downstream boundary condition when using a free surface algorithm with gravity, for example *F 36 15*. An integer is read. If 1, a hydrostatic pressure is used. If 2, the water continuity defect in the downstream boundary cells are set to zero.

**F 235** Algorithms to reduce instabilities in triangular cells. An integer is read, and depending on its value, a number of different algorithms can be chosen. One of the most successful algorithms is 10, giving extra relaxation in the triangular cells. The relaxation factors can be modified on the *F 244* data set.

Default: *F 235 0* (not used)

**F 237** The data set can be used to specify the water discharges instead of using the values in the *unstruc* file. An integer and a floating point number is read. The integer is an index for the discharge group. The floating point is the discharge. Multiple *F 237* data sets can be used for multiple discharge groups.

**F 238** Parameters in the nested grid algorithms. An integer and a float is read. If the integer is 1, the vertical elevation of the top of the nested grid will get the value of the float. If the integer is 2, the bottom of the nested grid will get the value of the float. If the integer is 3, the water level in the nested grid is interpolated from the coarse grid. If the integer is 4, the pressure in the nested grid so that the water level is on average approximately the same in the nested and the coarse grid. The algorithm is only implemented for *F 64 8, 11* and *13*.

Default: *F 238 0 0.0*

**F 239** Two integers are read, giving the maximum number of iterations in the algorithm for the free surface. The first integer is used for the *F 36 7,8,9* and *10* options. The second integer is used for the *F 36 4* option.

Default: *F 239 50 350*.

**F 242** Number of maximum loops in bed change algorithm to distribute bed changes to corners.

Default: *F 242 100*

If erosion is predicted in a cell and one of the corners do not have sediments, then an iteration will be done to distribute the remaining negative volume on the other corners. If there is not sufficient sediments in any of the corners, the loop will stop after 100 iterations, resulting in a sediment continuity defect.

**F 244** Two relaxation factors used in the algorithms to reduce instabilities in triangular cells. The first

floating point is used for the velocities in the cells, in the *F 235 10* algorithm. The second integer is used for the fluxes on the cell surfaces, if *F 235* is between 8 and 23.

Default: *F 244 0.5 0.8*

**F 246** Algorithms to stabilize the free surface algorithm *F 36 7*. Three integers and a float are read. The algorithm works on the connection between the cell that is to be computed and each of its four or eight neighbors.

If the first integer is 1 and the water is coming into the cell, or the integer is 2, a limiter will be invoked on the water depth in each cell. Different types of limiters can be used, depending on what the third integer is.

If the second integer is 1, the $a_p$ term in the equation is increased if the Froude number is between 0.9 and 1.1. Around supercritical flow the above $a_p$ term sometimes can be close to zero, causing a crash. If the second integer is 2, the $a_p$ term will never be below 1.0. If the second integer is 3, the ap term will be increased if the Froude number is between 0.5 and 1.5.

The third integer decides the magnitude and conditions of the depth limiter invoked by the first integer. The following options for the limiter are:

-1: the depth if the Froude number is above 1
-2: the depth * 1.1 if the Froude number is above 1
-3: the depth * the Froude number if the Froude number is above 1
-4: the depth if the Froude number is above 0.9
-5: same as option -3, except the Froude number is taken as the maximum of the cell and its neigbors
-6: the depth if the Froude number is above 0.5

The float is a limiter on the allowable surface slope used in the determination of the $a_{nb}$ coefficients for the *F 36 7* algorithm.

Default: *F 246 1 1 0 0.1*

**F 247** Emodulus (Pa) and Poisson ratio for the geotechnical slide algorithm.

Default *F 247 65000000.0 0.4*

**F 249** An integer is read. If 1, then negative sediment thicknesses are allowed. The feature is used to present measured bed elevation changes in the graphics.

**F 251** Different Shields curves. An integer is read. If non-zero, a lower or higher Shields curve will be used.

**F 255** Bed change algorithms. An integer is read, indicating which algorithm is used to compute the grid changes as a function of the bed elevation changes in the center of the cells.

0: Default algorithm, pushes bed changes from the center of the cell to the four corners
5: An algorithm where the vertical movement of the grid line intersection is computed from the four cells surrounding it.

**F 256** Maximum sediment concentration at the bed. Default 0.1 by volume.

**F 257** Option with more print-out if the program crashes. An integer is read. 1 will give more information.

**F 261** Nested grid algorithm. An integer and a float are read. If the integer is 1, the nested domain will be coupled with the coarse grid when computing water flow. If the integer is 2, the coupling will also be applied for the sediment concentrations. The float is a relaxation coefficient for the interpolation. Its value is between 0 and 1.

**F 262** Groundwater parameters. And integer and a float is read. The float is the conductivity of the soil. The integer is used to decide if groundwater is computations are done. The integer then has to be above zero.

Default: *F 262 0 5.3x10$^{-6}$*

**F 264** Algorithm to take a vertical variation of the bed sediment density into account. An integer is read. This algorithm is not much tested.

**F 265** Sieve size for graphics output. The graphics option has the possibility to display a charachteristic diameter for the sediments at the bed. The default is $d_{50}$. If the user want a different sieve size, this can be given on the data set.

Default: *F 265 0.5* ($d_{50}$)

**F 268** Decision about the nested blocks being printed to the *result*/Paraview files. An integer is read. If it is 0 (default), all the blocks are printed. If 1, then only the coarse grid is printed. If the integer is 2, then only the nested blocks are printed.

**F 272** Alternative algorithm to compute the production of turbulent kinetic energy, $P_k$, as proposed by Kato and Launder (1993). This is invoked if the integer read on the data set is 1.

Default: *F 272 0*.

**F 274** Bed angle parameters. An integer is read. If it is larger than zero then the program will try to read the *bedangle* file. This file will enable a spatial variation in the angle of repose for the bed material.

Default: *F 274 0*

**F 275** Modifications of the constants in the epsilon equation. Two floating point numbers are read, which gives two of the epsilon constants in the k-epsilon turbulence model.

Default: *F 275 1.44 1.92*

**F 277** Option to display a graphic view of the sediment continuity defects. This is done if an integer

above zero is read. Then the continuity defect is seen when choosing *Bed change defect* in the graphics menu of the contour map. The letter used to display this for the ParaView files is "Q" on the G 24 data set.

Default: *F 277 0*

**F 278** Variations of the *F 36 7* algorithm. The integer 16 uses the same formula as described by Olsen (2015).

Default: *F 278 0*

**F 279** Variations in the formula computing the bed shear stress. It is recommended to use the *F 499* data set instead.

**F 280** Interpolation algorithms for the nested grid. An integer is read. Four options are present, 0,1,2 and 3.

**F 281** Increase in the vertical eddy-viscosity due to dunes. An integer and a float are read. If the integer is 1, the vertical eddy-viscosity is increased by adding a value proportional to the float and the bed form height.

Default: *F 281 0 1.0* (algorithm not used):

**F 283** Extra damping in the changes of the water surface elevations for the *F 36 7* algorithm, if the Froude number is above 1. An integer is read. If it is 1, then the algorithm is used.

Default: *F 283 0* (algorithm not used)

**F 287** Algorithm that tries to estimate the outflow water flux from a reservoir, given the inflow and the changes in the water surface elevation of the reservoir. An integer is read. The algorithm is invoked if the integer is 2. The computed outflow flux will override the specified outflow in the *timei* file. If the integer is 3, the computed outflow flux will be checked against the value specified in the *timei* file, and not be allowed to go below this value.

If the integer is 5, the outflow will then be limited to be between 0.5 and 2.0 times the discharge in the timei file. If the integer is 6, the same limitation procedure will be used, but then two floats are read after the integer. These two floats will then be used instead of 0.5 and 2.0.

Default: *F 287 0* (algorithm not used)

**F 289** Geotechnial slide parameters. Four integers are read. The first integer decides which of multiple functions are used to compute the location of the slides. Options 4 and 5 include the groundwater computation.

The second integer decides for different gridding options for the slides

The third integer is not used

The fourth integer decides different options for damping the velocities in the slide.

Default: *F 289 0 0 0 0*

**F 292** Inner time step. An integer and a float is read. If the integer is 1, the float is used in an inner time step. This means the time step is used for each inner iteration. The purpose is to prevent instabilities, and the method seem to be more effective then most of the earlier methods. This especially applies to velocity instabilities in triangular cells.

Default: *F 292 0 0.0* (algorithm not used)

**F 293** A file where residuals are written for each iteration. The file is called *sigurd*. It makes it easier to make graphics of the residuals in a spreadsheet.

**F 294** An integer is read, giving the number of time steps in the time-dependent *inspace* file.

**F 295** A float is read. The float is multiplied with the average vertical distance between the geodata points an the bed level, to produce the roughness values.

Default: *F 295 1.0*

The data set is used in combination with F 2 K, which computes the roughness from a geodata point cloud. This is then printed to a bedrough.new file.

**F 297** An integer is read, indicating which version of the unstruc file is to be written. Default after 11. June 2012: 3. Default before 11. June 2012: 2. The difference between version 2 and 3 is that version 3 contains geometry information also for dry areas, making it possible to start a computation with a dried up area without using a *koordina/koosurf* file. Version 3 also contains more accurate information for the different blocks, making it possible to increase the *G 1* data set parameters and still use the same *unstruc* file. These two features were not possible in version 2.

There is also a version 4 of the unstruc file, which only contains the 2D grid and not the 3D grid. This will be much smaller than version 3, but may produce different number of 3D cells on different computers.

Default value as of October 2023 *F 297 3*.

**F 301** A float is read. It is the maximum residual defining a crash of the program.

Default *F 301 1.0e10*

**F 304** Bed flux option for fractional sediment transport (multiple sediment sizes). An integer is read. If it is above zero, then the erosional flux is larger than the concentration of the given fraction in the bed cell times the fall velocity. If the integer is 2, the flux is smaller than the concentration computed from the sediment transport formula times the fall velocity (no fractions taken into account).

Default *F 304 0*

Using *F 304 2* will make the fractional sediment transport less dependent on the bed grain size distribution. This will give a higher total sediment transport capacity. It will also reduce the sensitivity to for example the active bed sediment layer thickness.

**F 305** An alternative function for the wall laws. An integer is read. If it is 1, the alternative function is used. This is only implemented for *F 15 0, 8* and *15*.

**F 306** Use the grain size distribution in the *bedres* file. An integer is read. If it is 1, the sediment grain size distribution in the *bedres* file will be used instead of the *fracres* or the *N* data sets when starting a new computation.

Default *F 306 0*

**F 308** MPI option. An integer is read. If it is above 0, MPI algorithms will be used. This option has only been tested on two simpler cases. It is not recommended to use it.

**F 310** Algorithms to adjust the sediment thickness. The sediment thickness can be given in two ways: The *koomin* file in combination with the *unstruc/koordina* file. And the *fracres/bedres* files. The F 310 data set provides algorithms that deal with the situation where the sediment thickness is not the same in the two approaches. An integer is read. If 1, the *koomin* values are used. If the integer is 2, the *fracres/bedres* values are used.

Default: *F 310 0* (no corrections of the sediment thickness)

**F 314** Automatic setting of inflow/outflow areas. Two integers are read. If the first integer is 1, then the inflow area in *Discharge Group 1* will cover the whole upstream cross-section of the first block and be an inflow area. If the second integer is 1, then the whole downstream cross-section of the first block will be defined in *Discharge Group 2* to be outflow. The data set can be used instead of using the *Discharge Editor*.

**F 315-F 319** MPI data sets

**F 322** Shallow Water Equation parameters. Two integers are read. If the first one is 1, the time step source terms will be included in the solution. If the second is 1, the convective term in the Shallow Water Equation are included in the solution. Note that this is only for the computation of the free surface, with *F 36 9* and *F 321 1*.

Default: *F 322 0 0*

**F 323** New free surface function. An integer is read, and the new function is used instead of the old one of the integer is 1. The new surface function includes the Shallow Water equation option, *F 36 9*.

Default: *F 323 0*

**F 325** Geotechnical slide algorithm. An integer is read. Depending on its value, different algorithms are used to find the height of the slide block.

**F 326** Geotechnical slide time step and inner iterations. Similar to the *F 33* data set, but for the geotechnical slide movements.

**F 328** New function for the water flow computation. An integer is read. If it is 1, the new function is used.

**F 329** Print-out options. A series of up to 19 integers are read. Each integer specifies a file that can be printed out each time the *P 10* iteration is reached. The following list gives the options:

| | |
|---|---|
| 1st | *result* file |
| 2nd | *bedres* file |
| 3rd | Tecplot 2D |
| 4th | Tecplot 2D - nested version |
| 5th | Tecplot 3D |
| 6th | ParaView 3D |
| 7th | ParaView 2D |
| 8th | ParaView 2D written to multiple blocks |
| 9th | *NagelPtk* file |
| 10th | *beddata.sim*: x,y and z values of bed cells |
| 11th | a paramerter is read which tells what type of *interres* file is written: the number is the same as on the F 48 data set. |
| 12th | *Halvor* file: 2D depth-averaged boundary of the geometry |
| 13th | *habitat* file |
| 14th | *boogie.spe* file with bed area fraction that has slope over 0.2967 radians. |
| 15th | *boogie.spe* file with special print-out, depending on which integer is written: 2: x, water level, slide level, groundwater level |
| 16th | *fracres* file |

Default: *F 329 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0* (only *result* and *bedres* files are written)

It is not necessary to give in all the zero numbers. For example, if only the Tecplot 2D file is to be written, the following data set can be used:

*F 329 0 0 1*

**F 330** Cohesion data for the bed sediments. Cohesion at different distances below the bed is given. A linear interpolation is done between the specified points. An integer is first read, giving how many points. Then the depth and cohesion are given for each point.

Example: *F 330 3 0.001 0.01 0.2 0.3 1.0 0.6*

The cohesion 1 mm below the bed is 0.01 Pa. The cohesion 0.2 m below the bed is 0.3 Pa and the cohesion 1.0 m below the bed is 0.6 Pa. These are three points, thefore is the first integer 3.

The default is to have no cohesion.

**F 334** Geotechnical parameters for the viscosity of the moving slide. Two floating point numbers are read. The first is used in the formula for the eddy viscosity of the soil. The second is a reduction factor for the soil viscosity in the shear layer of the movement, at the bottom of the slide.

Default: *F 334 1.0 0.2*

**F 335** An ending time for the computations. A float is read, which is the time in seconds when the transient sediment computation stops.

**F 337** MPI data set, describing connection between the blocks.

**F 338** Minimum viscosity of the slide.

Default *F 338 1000.0* ($m^2/s$)

**F 339** Unstructured grid block for the geotechnical slides. An integer is read. If it is 1, the grid will be made.

**F 343** Convergence improvement algorithm for the free surface computation. An integer is read. If it is 2, then a block-correction algorithm is used along the main axis of the grid. If the integer is 3 then a classical multi-grid method is used.

Note that the convergence acceleration algorithms may give poorer stability for large time steps.

**F 345** Convergence criteria for the Casulli free surface algorithm. Default $10^{-5}$.

**F 349** OpenFOAM alpha reading. An integer is read. If it is 1, SSIIM 2 will try ty read an alpha file made by interFoam. Then the water level in SSIIM 2 will be adjusted to the water level in the alpha file and a new grid genereated in SSIIM 2. (option not well tested)

**F 350** Variable time step parameters as function of discharge. Four floats are read. The first float is the maximum time step. The second float is the minimum time step. The third float is a reference discharge, $Q_{ref}$, and the fourth float is the parameter, $n$. The two last parameters are used in the formula for the variable time step, which is given by Olsen and Hillebrand (2018), Eq. 5.

**F 353** Downstream free surface damping algorithm. Used in connection with the F 36 16 algorithm. An integer is read. If it is 3, the 10 most downstream cells will be dampened to an almosst horizontal water surface. This will prevent free surface instabilities at the downstream boundary.

**F 355** An integer is read which chooses which algorithm is used to compute pressure gradients for computations with free surface and gravity.

**F 364** Two floats are read. These are the minimum and maximum values used to scale the colors for the *geodata* points in the *GridEditor*.

**F 363** Scaling factors for the x and y values. Two floats are read. If they are different from 1.0, they will be multiplied with the geometrical grid values when reading the *unstruc* file if the F 112 data set is 1.

**F 365** An integer is read. If it is 1, then the points in the *interpol* file will be shown in the *Map* graphics.

**F 366** An integer is read. If it is 1, the momentum of the upstream inflowing water is set to zero.

**F 368** Braided channel information written to the *braided* file as a time series. An integer, four floats and an integer are read. The first integer is the iteration when the program starts writing the *braided* file. The four floats are four critical shear stresses for determining the Braiding Intensity (BI). A very low value can be used for the total BI. A higher value is chosen for the active BI, $BI_A$, close to the critical shear for the particles. Since this value may not be easy to decide exactly, three values are given. Then three $BI_A$ values are computed and written to the *braided* file. The last integer is the number of cells needed over the width to include the channel in the BI index.

**F 369** An integer is read. If it is 1, then some algorithms will be invoked to check for some type of bugs in the program. If bugs are found, information is written to the *boogie* file.

**F 370** Dune parameters written from a dune tracking algorithm to a file called *dune*. Five integers are read. The first integer is the iteration where the program starts writing the dune file. The following two integers indicate the starting and ending cross-section of the grid where the algorithm is used. The fourth integer indicate which point in the cross-section is used. The last integer gives a search interval in number of cells for the dune tracking algorithm.

For the antidune computations by Olsen (2022), the following data set was used:

*F 370 20000 10 200 2 8*

**F 371** Algorithms to avoid grid splitting. Grid splitting may occur if the wetting/drying algorithm divides the grid in more than one part, and the inflow/outflow regions are no longer connected. Three integers are is read. The first integer indicates which version of the algorithm is used. Default 0: No grid splitting supression algorithm is used.

*F 371 9 0 0 1.0* has been used to model a braided river. This option also reads a float at the end, which is a critical Froude number. The F 371 9 algorithm does a computation for each cross-section in the grid to try to find the water level which has the critical Froude number. This water level is then transferred to an array, and the computed water level from the F 36 4 algorithm is not allowed to go below this level.

The algorithm is not perfect, as a channel may end on the right side of the geometry and emerge on the left side. This will give a grids split even if there is a finite water depth and grid cells in all cross-sections.

**F 378** An integer is read. If it is 1, a file called *thalweg* will be written. This contains the x and y coordinates of the deepest point in each cross-section of the grid.

**F 379** Lowering of water level in cells with very low bed shear stress, so that these are removed from the grid. An integer and two floats are read. The algorithm is used if the integer is 1. The second integer is multiplied with the first integer on the *F 94* data set. The algorithm will only be used if the water depth is below this number. The second float is the critical bed shear stress for invoking the algorithm.

**F 380** The data set has different options of removing cells that has a low depth. F 380 4 or 8 will remove cells where the water depth is lower than the first float on the F 94 data set. F 380 7 will remove all cells that has a depth lower than zero.

**F 385** Cohesion parameters. An integer and a float is read. Sediment cohesion is used if the integer is 1. The float is the cohesion in Pascal.

**F 386** Dreding algorithm. Two integers are read. If the first is 1, the dredging algorithm is invoked. Then, the *bagger* file has to be used. The second integer is the number of time steps in the *bagger* file.

**F 387** Minimum depth in the grid (meters). This is used to reduce stability problems caused by too small water depths.

**F 388** Relaxation coefficient for the velocities in the default SIMPLE correction algorithm.

**F 389** Integer saying how many iterations are computed between writing *alldata* files.

**F 390** Grid fixation at inflow boundary. An integer is read. If it is 1, and algorithms will be used that fixes the grid at the inflow boundary, not allowing the cells to change in shape as a function of the wetting/drying procedure.

**F 392** An integer is read. If it is 1, then the bed changes in the *alldata* file will be set to zero when reading the *alldata* file. This means the bed changes shown by SSIIM 2 will only be values computed **after** the *alldata* file is read.

**F 393** An integer is read. If it is 1, then the limit of the movable bed will be set equal to the original bed level read from the *alldata* file.

**F 394** Removes ponds in the grid, made up of more than one cell. A pond is a wetted area that is not connected with the inflow/outflow region. An integer is read. The integer indicates many variations in the algorithms. *F 394 1070* has been used. The last two digits indicates an iteration number for the algorithm (70).

Default *F 394 0* (algoritm not used)

**F 396** Flag that affects the bed level changes at the grid boundary. If it is 1, less changes will occur at the boundary. Default 0.

**F 398** Coefficient in van Rijns formula for computing the roughness as a function of $d_{90}$. Default 3.0.

**F 399** An integer is read. If it is 1, the program will **not** stop if the x or y coordinates in the koomin file is not the same as in the grid.

**F 405** An integer is read. If it is 1, a time step is added in the equation for the free water surface location for the algorithm invoked by *F 36 7.*

**F 406** A float is read. This is added to the $a_{nb}$ coefficient in the free surface algorithm invoked by the *F 36 7* data set.

**F 407** Computation of critical water level (Frode number == 1) for the anti-grid-split algorithm. An

integer is read. If it is 1, a function is called that computes the critical water level. The function is made in February 2018. It was later replaced by another function made in March 2019.

**F 410** Emodulus (Pa) and Poisson ratio for the soil in the geotechnical slide algorithm. Two floats and an integer is read. If the integer is zero, the Emodulus is used in both horizontal directions. If the integer is 1, the Gmodulus is used in the direction parallel to the main water flow.

**F 411** Bank cohesion used in the geotechnical slide algorithm. A float is read (Pascal).

**F 417** Plastic algorithms. An integer is read. If it is 1, a negative fall velocoity of the particles on the S data sets will be allowed. Also, the boundary conditions will be adjusted for plastic particles with a negative fall velocity (positive rise velocity). The Z 106 parameter on the G 24 data set will show plastic deposition.

**F 421** Computations stopping criteria for computing the filling of a reservoir with sediments. A float is read, between zero and 1, telling when the computations should stop.

> Example *F 421 0.7*. SSIIM 2 stops the computation when 70 % of the grid volume is filled with sediments.

**F 431** Increasee in the Anb coefficients for the free surface computation F 36 4. An integer and a float is read. If the integer is 1, the coefficients are added the following value:

> min(10, 15 - floating point number * depth / *MinVertDist1*)

*MinVertDist1* is the 2nd float on the F 94 data set.

**F 433** Print-out of added sediment fractions. An integer and a float are read. If the integer is 1, a print-out of some of the tracer fractions are done. This is done to a depth equal to the float (in meters). The option is used in connection with artificially feeded sediments to a river.

**F 448** Sediment continuity improvement algorithm. An integer is read. If it is 1, then the suspended sediment concentration will be scaled according to inflow/outflow/bed changes, in order to improve the sediment continuity.

**F 453** Longitudinal profile printing option. An integer is read. For example 200. Then at each 200th iteration, a profile in the middle of the geometry/river is printed to a file called *profile*. The file contains the distance from the upstream end, bed levels, water levels and some sediment fluxes. Note that the file is overwritten each 200th time, so only the last version will be on the disk.

**F 462** Print-out of sediment fractions to the ParaView files. An integer is read. If it is 1, then the additional print-out is done. Default *F 462 0.*

**F 464** Outflow concentration print-out. An integer is read. For example 300. For each 300th iteration, the outflow sediment concentration will be written to a file called "*outconc"*. Default *F 464 0.*

**F 465** Iterations for geotechnical sand slides solver computing the location of the slides. Two integers

are read. The first for the horizontal location of the slides, and the second for the vertical location. Default *F 465 200 200*.

**F 469** Damping factor for the water level movements at the side walls computing the free surface with the F 36 1 or F 36 16 algorithms. Default *F 469 2.0*. A low value means more damping.

**F 471** Algorithm for connecting geotechical sand slide grid with the standard grid. An integer is read. If it is 1, the algorithm is used.

**F 476** Formula for bedform height. An integer is read. If it is 0 (default) or 1, van Rijns formula is used. If the integer is 0, the bed form height is limited between so that it is not larger than the sediment thickness or the water depth. If the integer is 1, there are no such limitations.

If the integer is 2 or 3, one of Karims (1999) formulas is used. If the integer is 2, the formula for antidunes and standing waves is used. If the integer is 3, the formula for dunes, ripples and transitions is used.

**F 478** New water surface elevation function. If a grid has been generated and the user wants to start with a different water surface elevation (it has to be a flat, constant water level), F 478 1 can be used when reading the grid. Then the program will take the water surface elevation on the W 1 data set and make a new grid with this water level.

This may be useful if a wetting/drying situation is to be modelled. Then the initial grid has to cover the whole domain, meaning it has to have a high water level during the generation. After the unstruc file is made, then a lower water level can be given on the W 1 data set in the control file together with F 478 1. Then the computation will start with a grid that has the lower water level.

**F 481** Scaling of the pressure field. An integer is read. If it is above 0, then the pressure field from the SIMPLE method it scaled so that the reference cell (most often downstream) will get a zero pressure. If gravity is not used, the absolute value of the pressure from the SIMPLE method will not be decided. Only the relative pressure between cells will be used.

**F 482** Removal of time step in the Navier-Stokes equations. An integer is read. If it is 1, then the time step will not be included in the solution of the Navier-Stokes equations. Default, *F 482 0*.

**F 486** Multi-grid algorithm for the Casulli free water surface computation (*F 36 9*). An integet is read. By default (*F 486 0*), a tailor-made multi-grid algorithm is used for the Casulli solver. If the standard multi-grid solver is to be used, then *F 486 1* can be given.

**F 488** New grid function. An integer is read. If it is 2, the new function is used. If it is 1, a version of the new function with partially horizontal grid lines is used. This function can be used for lakes and reservoirs with turbidity currents.

**F 489** Number of inner iterations for the first time step. An integer is read. If a velocty field is not read before the computation is started, then there may be some initial instabilities in a time-dependent computation. With the current data set, the first time step can have more inner iterations than what is given on the F 33 data set, giving better initial stability.

**F 491** Fraction of maximum depth, used in the ponds algorithm. Default F 491 0.01 (1 % of the

depth). The ponds algorithm removes the cells in parts of the grid that has no connection with the inflow/outflow region.

**F 492** Ridges algorithm for F 488 > 0. Ridges are grid lines that are above the water surface, although the cells on both sides exist. This can cause stability problems. An integer is read. If it is 1 or 2, an algorithm is used to try to reduce the problem.

**F 493** Algorithms that smoothens the edges of the grid boundary, seen in plan view. An integer is read. If it is between 0 and 6, different algorithms will be used. Default *F 493 0*.

**F 494** Effect of sediment concentrations on the velocity field. An integer is read. If it is 1, a formula is used in the wall laws that takes the sediment concentration into account when computing the bed shear stress. Default *F 494 0*.

**F 499** How the bed shear stress is computed for the sediment transport computations. An integer is read. Alternatives are:

0: 300 * turbulent kinetic energy (default)
2: From the velocity in the bed cell and Mannings equation
3: From the water depth and the water surface slope
4: From maximum of option 0 and 2
5: From harmonic interpolation between option 0 and 2
6: Average of option 0, 2 and 3

**F 500** Coefficient in Engelund-Hanses sediment transport formula. A float is read. Default *F 500 0.05*.

**F 507** Increase in bed shear stress as number of cells over the depth. An accurate computation of the bed shear stress often requires a certain number of cells over the depth. Fewer cells may reduce the bed shear stress. The 10 floats are factors increasing the bed shear stress. The first number is for bed cells that have only 1 cell over the depth. The second number is for bed cells that has 2 cells over the depth. Etc. It is assumed that with 10 cells over the depth, there is no need for an increase in bed shear stress.

Default: *F 507 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0*

**F 512** Algorithm to prevent instabilities and crashes related to splitting of the outflow area. Two floats are read. The first one is a minimum shear stress in Pascal. The second is a depth in meters. If the water depth is more shallow, the bed shear stress along the boundary where the water is flowing out will always be above the minimum value. Default: *F 512 0.0 0.0* (algorithm not used)

**F 516** OpenFOAM grid expansion factors. Three integers are read. They will be transferred to the blockMeshDict file in OpenFOAM, increasing the number of grid cells in each direction. If you know the blockMeshDict file well, you may instead give the numbers directly in the file. Default: *F 516 1 1* 1

**F 519** Grid generation option for the free surface. An integer is read. If it is 1, the grid will not be allowed to go above the free water surface for the *F 488 1* option. Default *F 519 1*.

**F 520** Number of smoothing options in the multi-grid solver. An integer is read. Default *F 520 3*.

**F 532** Parameters in computing a variable time step. An integer and two floats are read. The algorithm is used if the integer is larger than zero. Then two floats are read, the starting and the end time step. The starting time step will be used at the first time step. Then the time step will increase until it reaches the last time step. The increase will take place over the same number of time steps as the first integer on the data set. The percentage increase will be the same over the whole time period. After the time step has reached the end time step, it will stay constant.

The option is typically used when computing local scour, where the process is going very fast initially, and slower over time.

**F 533** Horizontal movement of the grid boundaries - not the corners. An integer is read. If it is 1, the cell shapes along the boundary will change depending on the water depth. The algorithm will not change corners, that is done with the F 493 data set. Default *F 533 1* (always used)

### 5.3.2 The G data sets

**G 1** Four integers are read, called *xnumber*, *ynumber*, *znumber* and *lnumber*.

For SSIIM 2, the unstructured grid will also use structured arrays for connecting the different blocks. The blocks will be put beside each other in the lateral direction of the grid. The default maximum grid size is 300x300 cells. If the grid size is smaller, it may be an advantage of using smaller numbers on this data set, to reduce the memory requirement. If the length of one block (number of cross-sections) is larger than 300, this data set must be changed accordingly. If the number of blocks times its widths is larger than 300, this data set must also be changed accordingly. The third integer is the maximum number of grid lines in the vertical direction. The last integer is the number of sediment sizes.

**G 3** Vertical levels of the grid lines. For *F 64 0* (default) and *F 64 1*, the values on the *G 3* data set are level in meters, and the longitudinal and lateral grid lines are completely horizontal.

**G 6** Data set for calculating water surface elevation with an adaptive grid. Three integers and two floats are read:

*iSurf*
*jSurf*
*kSurf*
*RelaxSurface*
*ConvSurface*

The *RelaxSurface* variable is a float relaxing the estimation of the increment to the new recalculated water surface. Use low values if you experience instabilities.

The *ConvSurface* variable is a float setting the limit for when the water surface should be

recalculated. The water surface will be updated when the maximum residual of the equations are below this parameter. Recommended value: 0.01 - 1.0

In SSIIM 2 the grid is unstructured, so it is only necessary with one index to reference a cell. The first index, iSurf, is then used to point to this cell. The water level above this cell will not be moved during the computations. Normally, the other two variables, *jSurf* and *kSurf* should be set to zero. However, it is also possible to use two reference cells that do not move. This is not correct from a hydraulic point of view, but it can be done to improve stability. The number of the second cell is then given as the second integer, *jSurf*. The third integer, *kSurf*, is normally zero. However, if the water level is changed at only one location, for example the downstream boundary, the *kSurf* integer can be set equal to the number of the discharge group at the outlet, given in the *DischargeEditor*. This will then cause the water level over the whole outlet cross-section to be lowered simultaneously and in a horizontal line, instead of being lowered at one point.

Selecting the surface index is usually done in the map graphics, where the cell indexes are seen. Note that a dynamically changing grid must not be computed during this procedure, and the *F 112* data set must not be used in the *control* file when finding the number. The *F 112* data set can be re-inserted into the *control* file after the *G 6* numbers are found.

When the grid cell indexes change for a dynamic grid, the location of the cells in the initial grid is used.

Example: *G 6 35310 43673 0 0.1 0.1*

The water surface above cells 35310 and 43673 is fixed during the movement of the water surface.

For SSIIM 2, also see the *G 62* data set.

**G 13** Outblocking option that is used when a region of the geometry is blocked out by a solid object. An integer is read first, which determines which sides the wall laws will be applied on. The following options are possible:

0: No wall laws are specified
1: Wall laws are used on the sides of the block
2: Wall laws are used on the sides and the top of the block
3: Wall laws are used on the sides, the top and the bottom of the block

Six integers are then read: *i1 i2 j1 j2 k1 k2*. These integers define the cells of the block. The two first integers are the first and the last cell in the *i*-direction. The next two integers are the first and the last cell in the *j*-direction. The last two integers are the first and the last cell in the vertical direction.

When making blocks, note that there must be at least two free cells (not blocked out) between each block or to the wall. It is recommended to use more free cells than two, to resolve the flow pattern between the blocks.

Up to 49 *G 13* data sets can be used.

For SSIIM 2, the first and the two last integers are not used, but numbers must be given in anyway for compatibility with SSIIM 1. In SSIIM 2, the data set will only have an effect when given in the *control* file before the grid is generated.

**G 22** Data set to model horizontal constructions, for example bridges. Four integers and three floats are read. The first four integers, *i1,i2,j1,j2* are indexes describing an area of cells in the 2D depth-averaged grid. This is where the construction is to be placed. The first float is the lowest elevation of the structure. The second float is the highest elevation of the structure. The third float is a velocity reduction coefficient. For solid objects, it can have a value of 1.0 or more. Higher values will give more reduction and lower velocity. For porous objects, the value can be lower than 1.0. If the value is lower than 0.7, then the velocity reduction formula is changed into a porous drag formula, where the velocity reduction coefficient is the solid volume fraction of the object.

The program automatically finds which of the vertical cells are within the area of the construction. This is useful when the grid moves vertically.

Up to 19 *G 22* data sets can be used.

**G 24** Data set to determine which variables are written to the ParaView/Tecplot files. First, an integer is read giving the number of variables on the data set. Then, for each variable, a character and two integers are read. The character indicates the name of the variable, as detailed in Chapter 5.19. The first integer gives the level above the bed. The second integer gives the sediment size. For some variables, this information is not relevant. Integers still have to be given, but they are then not used.

Example: *G 24 3 u 1 0 p 2 0 c 1 2*

**G 40** Initial vertical distribution of temperature. An integer, n, is read, which gives the number of points in the profile. Then n pairs of floats are read, the first float being a vertical level in meters, and the second float is the corresponding temperature in degrees Centigrade.

Example: *G 40 2 101.0 20.0 88.0 17.0*

At the start of the calculation, the temperature is 20 degrees Centigrade at level 101.0 meters and 17 degrees at level 88.0. There is linear variation between the given levels.

The highest levels must be first in the data set.

**G 41** Initial vertical distribution of a water quality parameter. Two integers are first read, where the first integer is the number of the water quality parameter. This corresponds to the Q data sets. The second integer is the number of data points, n, in the vertical direction. Then n pairs of floats are read, the first float being a vertical level in meters, and the second float is the corresponding temperature in degrees Centigrade.

Example: *G 41 4 2 101.0 20.0 88.0 17.0*

At the start of the calculation, the water quality parameter no. 4 is 20 at level 101 meters and 17 level 88.0. There is linear variation between the given levels.

The highest levels must be first in the data set.

**G 42** Multiple vertical surfaces for the OpenGL graphics. A surface is give on each G 42 data set, and up t 20 *G 42* data sets can be given. For each data set, two integers are read. The first integer is a grid node number. This indicates the starting point of the surface. The second integer is a flag indicating the

direction of the surface. A zero is in one direction, and 1 is the other. It is necessary with a trial and error run to get the second parameter right.

**G 50** OpenMP parallelization parameters for debugging. Ten integers are read. Each integer can be 0 or 1. The algorithms in SSIIM has two versions: One parallel and one non-parallel. If F 206 is above zero, the parallel versions are used. With the G 50 data set, the user can force a combination of serial and parallel algorithms.

Default: *G 50 0 0 0 0 0 0 0 0 0 0*

If one of the integers on the *G 50* data set is 1, then the serial algorithm will be used for this part, even though *F 206* > 0 is used and parallel versions of all the other algorithms are used.

The purpose of the data set is debugging. If the parallel version and the serial version do not give the same result, the *G 50* data set can be used to see which algorithms cause the problem.

The different algorithms used are for SSIIM 2:

1st: coefficient generation for the Navier-Stokes equation, the Power-law scheme
2nd: computation of eddy-viscosity, setting anb coefficients to zero, computing the water flux and making the coefficients for the second-order upwind scheme
3rd: wall laws and source term for the pressure equation
4th: water fluxes computation for the pressure-correction equation
5th: SIMPLE corrections
6th: turbulence computations, wall laws and coefficient generation for k
7th: turbulence computations, wall laws and coefficient generation for epsilon
8th: computation of residuals
9th: solver
10th: time step and residual for k

**G 62** Data set for calculating water surface elevation with an adaptive grid. Four integers and two floats are read:

*iSurf*
*jSurf*
*kSurf*
*kSurf2*
*RelaxSurface*
*ConvSurface*

The data set is identical with the *G 6* data set, except that four integers are read instead of three. The fourth integer, *kSurf2*, is the number of the discharge group that has the cell number in the second integer, *jSurf*.

If *jSurf2* is non-zero, the water level over the whole cross-section of the discharge group is changed simultaneously and in a horizontal line, instead of being lowered at one point.

### 5.3.4 The *K* data sets

**K 1** Number of iterations for flow procedure and number that determines the minimum iterations between updates of water surface. Two integers.

Default: *K 1 40000 50000*

**K 2** Two integers that indicate if laws of the wall are being used for the water flow computation. If 0, wall laws are used, and if 1, zero-gradients are used. The first integer applies to the side walls. The second integer applies for the surface. Wall laws are always used for the bed, if not changed by the *W 4* data set. Default: *K 2 0 1*.

**K 3** Relaxation factors. Six floats. For the three velocity equations, the pressure correction equation and the k and ε equation. For further description of the relaxation factors, see Chapter 3.

Default: *K 3 0.8 0.8 0.8 0.2 0.5 0.5*

**K 4** Number of iteration for each equation. Six integers. Default: *K 4 1 1 1 5 1 1*

**K 5** Block-correction. Six integers are read, one for each of the six water flow equations. If the integer is 1, block-correction is used. For the TSC algorithm in SSIIM 2 for Windows, the integer 2 indicates that the block-correction is only used in the first of the inner iterations in each time step. For SSIIM 2, the integer 10 used in connection with *F 168* will invoke a multi-grid algorithm.

Default: *K 5 0 0 0 0 0 0*

**K 6** Six integers are read, one for each of the six water flow equations. The integers determines which discretization scheme is to be used for the convective terms in the Navier-Stokes equations.

0: first-order power-law (POW) scheme
1: second-order upwind (SOU) scheme
10: QUICK scheme
11: Cubic upwind scheme
12: SMART scheme
13: H-QUICK scheme
14: van Leer scheme
15: Superbee scheme
16: Minmod scheme

Note that the different options only applies to the velocity and turbulence equations. Options 10 and above only applies to the velocity equations. The pressure-correction equation will use a different approach for the discretization. This means that the value of the fourth integer will not affect the computations.

Default: *K 6 0 0 0 0 0 0*

Also note that using a different scheme than the first-order upwind scheme for the turbulence equations usually leads to stability problems.

### 5.3.5 The *L* data set

*L* Specification of isoline values for *ContourMap* plot. First, an integer is read, which gives the number of isolines. Then, this number of floats are read. The floats specify the isolines. Example:

*L 6 55.0 56.0 57.0 58.0 59.0 60.0*

If the geometry has bed levels between 55 and 60 meters, and the user chooses bed levels from the graphics options, a contour map of the bed levels will be displayed. There will be contour lines for each meter, from 55 to 60 meters.

Default: The program finds the maximum and minimum value of the variable in the field, and uses 7 lines located between the values. The maximum and minimum value can be changed by the user from the menu.

### 5.3.6 The *M* data set

The M data set specifies inflow of a water quality constituent or sediments. A maximum of 9 groups of M data sets can be used. Each group corresponds to an inflow section, specified in the *DischargeEditor*. Two integers are read first. The first integer is an index for the group. The first group has index zero. The second integer is an index for the water quality constituent/sediment size. Zero is the smallest number possible. After the two integers, a float is read. This is the value of the inflow concentration of this parameter.

Example: *M 0 2 0.001*

If sediments are computed: In inflow section 0, the concentration of size 2 is 0.001.

For example, if there are three inflow sections, and five water quality constituents, it will make sense to have 15 *M* data sets. If no *M* data set is given for a particular inflow section and water quality parameter, the inflow is zero. M data sets given for sections with water outflow has no effect on the computation, and should not be used.

The values of the *M* data set does not change over time, unless altered by using *M* data sets in the *timei* file. This is the recommended procedure to model time-varying inflow of water quality components.

It is recommended to model time-varying inflow of sediment concentration by specifying the concentrations in the timei file, without using *M* data sets in the *control* file.

### 5.3.7 The *N* data set

The *N* data set composes the size fractions of different groups of sediments. Two integers and one float is read. The first integer is an index for the group. The second integer is an index for the sediment size, similar to the first index on the *S* and *I* data sets. The float is the fraction of the size in the group.

The first group has index 0 (zero).

Normally, multiple *N* data sets are used for making a group. For example if there are three groups and five sediment sizes, there should be 15 *N* data sets.

The different sediment groups are distributed in the geometry by using the *B* data sets.

Note that more than one sediment group can only be used in SSIIM 1. In SSIIM 2, only group zero is used over the whole geometry. To use a spatial variation in the grain size distribution in SSIIM 2, the *fracres* file has to be used.

### 5.3.8 The *B* data set

The *B* data set distributes different sediment groups to different locations of the geometry. The groups are made using the *N* data set.

Five integers are read. The first integer is an index for the group number, similar to the first integer on the *N* data set. The second and third integer are indexes for the cell numbers in the streamwise direction. The fourth and fifth integer are indexes for the cell numbers in the lateral direction.

Example: *B 0 2 6 2 9*

Sediment group no. 0 is placed in the cells from $i$=2 to $i$=6 and $j$=2 to $j$=9, where $i$ is an integer for the streamwise cell number and $j$ is an integer for the cell numbering in the lateral direction.

If only one group is used, it is possible to give *B 0 0 0 0 0*, which indicates that this group is distributed in all the cells.

### 5.3.9 The *P* data sets

**P 2** Five floating points that give scaling for the graphical presentation. The first three gives scales in streamwise, lateral and vertical direction. The fourth and fifth give movements in left-right and vertical direction. Defaults: 1.0 for the scales, and 0.0 for the movements.

**P 3** Four integers that give initial location of the graphical plots in streamwise, lateral and vertical direction, and sediment fraction number.

**P 4** A character that indicates initial type of plot is used initially in the graphics windows. The character "*g*" gives the grid, "*v*" gives velocity lines, "*V*" gives velocity vectors, "*c*" gives concentration. Further options are given in Chapter 5.19.

**P 6** Minimum and maximum values for blue and red colors in the OpenGL graphics. Two floats are read.

**P 10** Integer for the number of global iterations between printing *result* files for time-dependent computations. In SSIIM 2, a number of different files can be written, as given on the *F 329* data set.

**P 11** Extra print-out to the *boogie* file for each time step. An integer is read. Depending on the integer, different information is printed.

1: Total amount of water quality constituent 1 for the whole grid
2: Average value of water quality constituent 1 in the surface cells
4: Total number of grid cells, surfaces and points.

Default *P 11 0* (no extra print-out)

**P 14** A float is read, which gives a time interval for when a graphic file is automatically saved in the profile and contour plot. The graphic window have to be open and the timer has to be running. This only works for time-dependent calculations. The time interval is given in seconds.

### 5.3.10 The *Q* data sets

There are two types of *Q* data sets. One is the *Q 0* data set, which gives the name of the variable. The other type specifies the variables and constants in the source terms in the convection-diffusion equations.

**Q 0** An integer is first read, corresponding to the equation number. Then a text string of up to 15 character is given. The text is the name of the variable in the equation.

**Q (1-..)** All other Q data sets have the following in common:

The first integer specifies which source term is used. If the integer is below 1000 or above 3000, the source term is applied for all the cells. If the integer is above 1000, but below 2000, the source term is only applied to the cells closest to the water surface. These terms can typically be used for specification of fluxes across the water surface. If the integer is above 2000 but under 3000, the source term only applies to the cells closest to the bed.

The second integer after the *Q* is an index equal to the number of the equation the source term is applied to.

In the following, note that this is not repeated, and the two fist integers are not described further.

**Q 1** The second number is a float, giving the value of a constant source.

**Q 2** The second number is an integer. This is an index for the variable in the source term. The third number is a float, giving a coefficient which is multiplied with the variable.

Example: *Q 2 1 2 0.05*

This means that a source term for equation 1 is used, and the source term is 0.05 multiplied with the value of variable 2.

**Q 3** The second number is an integer. This is an index for the first variable in the source term. The third number is an integer, giving the second variable in the source term. The fourth number is a float, which is multiplied with both variables.

Example: *Q 3 1 2 1 0.04*

This means that a source term for equation 1 is used, and the source term is 0.04 multiplied with the value of variable 2, multiplied with the value of variable 1 (the same variable as calculated for this source term).

**Q 11** Fall velocity data set. This is a special data set to include the effect of the fall velocity of a water quality constituent, for example sediments or algae. A constant fall velocity is given. Note that calculating algae with variable fall velocity as a function of light irradiance, for example the *Q 151 / Q 221* data sets can be used instead.

The second parameter is a float, which is the fall velocity in meters/second. A positive number denotes a rise velocity, while a negative number denotes a fall velocity in downwards direction.

Example: *Q 11 2 -0.0001*

Variable no. 2 has a fall velocity of 0.1 mm/second.

**Q 42** Special data set to calculate algae growth where there is only one species of algae, and this is only growth-limited by light. Four floats are read. The first two floats are constants in the regression formula for the vertical attenuation coefficient as a function of the algae concentration. The last two floats are the coefficients *c0* and *c1* in the formula for algae growth.

**Q 51** Special data set for fall velocity for dinoflagellates as a function of irradiance. Five floats are read after the control integer. The two first are coefficients in the formula for the light attenuation coefficients. The third is the optimum irradiance. The fourth is the maximum fall/rise velocity in m/s. The fifth is the difference between optimum and actual irradiance when the fall/rise velocity is maximum.

Example: *Q 51 1 0.45 4.8 100.0 0.01 100.0*

The algae concentration is in variable number 1.

**Q 113** Special data set to calculate light history, if more than one algae species or sediment variable contribute to the light shading:

integer 1: pointer to irradiance variable
float 1: averaging time period in seconds

**Q 121** Special data set to calculate predatation. This data set must be used for both the phytoplankton and the zooplankton.If phytoplankton is calculated, the grazing rate is negative. If zooplankton is calculated, the grazing rate is positive.Note that if zooplankton is calculated, the coefficient also includes the grazing efficiency and possible conversion between phytoplankton and zooplankton units.

Example: *Q 121 2 4 -0.01 1.067*

Algae is calculated as variable no. 2, zooplankton as variable no. 4. The grazing rate is 0.01 and the temperature coefficient is 1.067.

**Q 122** Special data set for reduction of patogens from light. One integer and two floats are read. The second integer is a pointer to the irradiance variable. The first float is the proportionality constant for decrease of patogens. The second float is a temperature constant.

Example: *Q 122 1 3 -0.001 1.067*

Pathogens are calculated as variable no. 1 and irradiance as variable no. 3. The proportionality constant is -0.001 and the temperature coefficient is 1.067.

**Q 123** Special data set for sorption, containing one integer and two floats. The integer is an index to the sediment variable. The first float is the absorption coefficient. The second float is the fall velocity of the sediment.

Example: *Q 123 2 4 0.1 0.005*

Variable no. 2 is adsorped to variable no. 4, with the adsorption coefficient 0.1.
The fall velocity of variable 4 is 5 mm/second.

**Q 131** Special data set for calculation of algae fall velocity from density variable

integer1: pointer to algae density
float 1: algae diameter in meters, from *Q 221*
float 2: form resistance (around 1.0-1.15), from *Q 221*
float 3: temperature in degree Centigrade

**Q 132** Special data set for calculation of light history from one species of algae. The light history is calculated by a formula and fixed.

integer 1: pointer to algae concentration
float 1: a - regression constant in formula for specific vertical attenuation coefficient, as *Q 42*
float 2: b - regression constant in formula for specific vertical attenuation coefficient, as *Q 42*
float 3: averaging period in seconds, for example: 86400.0 for 24 hours

**Q 133** Special data set for calculation of light history from one species of algae. The light history is calculated as a passive contaminant.

integer 1: pointer to algae concentration
float 1: a - regression constant in formula for specific vertical attenuation coefficient, as *Q 42*
float 2: b - regression constant in formula for specific vertical attenuation coefficient, as *Q 42*
float 3: averaging period in seconds, for example: 86400.0 for 24 hours

**Q 151** This is a special data set to calculate the density of algae in a lake. The second number is an integer, which is an index for the variable number which is the algae concentration. The algae concentration is used when calculating the light transmission. After the second integer, five floats are read. The first float is the light transmissivity. The second, third and fourth float are coefficients *c1,c2* and *c3* in the formula for algae density. The fifth float is the half-saturation irradiance for maximum rate of density increase. This is also used in the density formula.

Note that the convection-diffusion equation is not solved for the algae density, but from a user point of view, this variable is similar to other variables.

Example: *Q 151 1 2 1.0 2.0 3.0 4.0 5.0*

Algal density is in variable no. 1, and algae concentration is in variable no. 2. The light transmissivity is 1.0, and the coefficients in the density equations are: *c1* = 2.0, *c2* = 3.0, *c3* = 4.0. The half-saturation irradiance coefficient is 5.0. Note that these numbers are arbitrarily chosen, and may be unphysical.

Also note that if a *Q 151* data set is present in the *control* file, and the *timei* file is read for time-dependent calculations, an extra float is read as the last number of each line. The float is the irradiance.

**Q 161** Special data set to calculate algal density for cyanobacteria. The second number is an integer, which is an index for the variable number which is the algae concentration. The algae concentration is used when calculating the light transmission. After the second integer, five floats are read. The first float is the light transmissivity. The second, third and fourth float are coefficients *c1,c2* and c3 in the formula for algae density. The fifth float is the half-saturation irradiance for maximum rate of density increase. This is also used in the density formula.

Note that the convection-diffusion equation is not solved for the algae density, but from a user point of view, this variable is similar to other variables.

Example: *Q 161 1 2 0.67 0.087 2.0 3.0 4.0 5.0*

Algal density is in variable no. 1, and algae concentration is in variable no. 2. The coefficients in the equation for light transmissivity are 0.67 and 0.087, and the coefficients in the density equations are: *c1* = 2.0, c2 = 3.0, c3 = 4.0. The half-saturation irradiance coefficient is 5.0. Note that these numbers are arbitrarily chosen, and may be unphysical.

Also note that if a *Q 161* data set is present in the *control* file, and the *timei* file is read for time-dependent calculations, an extra float is read as the last number of each line. The float is the irradiance.

**Q 221** This a special data set that calculates the fall velocity of algae in a lake. The second and third number are integers. The second number points to the variable number of the algae density. The third integer points to the variable number of the temperature. Then two floats follow. The first float is the algae diameter in metres. The second float is a form resistance coefficient, usually around 1.0.

Example: *Q 221 2 1 0 0.00001 1.0*

The algae concentration is variable no. 2, the algae density is variable no. 1, and the temperature is variable no. 0. The algae diameter is 0.00001 meters and the form resistance coefficient is 1.0.

**Q 252** Special data set to calculate algae growth in case of only one nutrient + irradiance limiting the

growth:

integer 1: integer pointing to algae concentration
integer 2: pointer to light irradiance variable
integer 3: pointer to nutrient variable
float 1: $c_0$ - formula for growth rate, *k*, irradiance.
float 2: $c_1$ - formula for growth rate, *k*, irradiance.
float 3: *kmax* - maximum growth rate
float 4: *Ks* constant for nutrient
float 5: temperature coefficient

**Q 261** Special data set for nutrient depletion from algae growth, for multiple algae species limiting the light, and one nutrient. Three integers and seven floats are read:

integer 1: integer pointing to nutrient variable
integer 2: integer pointing to light variable
integer 3: integer pointing to algae variable
float 1: $c_0$ - formula for growth rate, *k*, irradiance.
float 2: $c_1$ - formula for growth rate, *k*, irradiance.
float 3: kmax - maximum growth rate
float 4: *Ks* constant for the nutrient being calculated
float 5: temperature coefficient
float 6: fraction of nutrient in algae growth

**Q 281** Special data set for algae growth, with phosphorus, nitrogen and light as the limiting variables:

integer 1: integer pointing to algae variable
integer 2: integer pointing to phosphorous variable
integer 3: integer pointing to nitrogen variable
float 1: a - regression constant in formula for specific vertical attenuation coefficient
float 2: b - regression constant in formula for specific vertical attenuation coefficient
float 3: c0 - formula for growth rate, k, irradiance.
float 4: c1 - formula for growth rate, k, irradiance.
float 5: kmax - maximum growth rate
float 6: Ks constant for phosphorous
float 7: Ks constant for nitrogen
float 8: temperature coefficient, Kt

The first four floats are similar to the *Q 42* data set.

**Q 282** Special data set for calculation of algae density from one species of algae:

integer 1: integer pointing to algae density
integer 2: pointer to algae concentration
integer 3: pointer to light history
float 1: a - regression constant in formula for specific vertical attenuation coefficient, as *Q 42*
float 2: b - regression constant in formula for specific vertical attenuation coefficient, as *Q 42*

float 3: c1 - formula 5, SCUM, as *Q 151*
float 4: c2 - formula 5, SCUM, as *Q 151*
float 5: c3 - formula 5, SCUM, as *Q 151*
float 6: Ki - half-saturation irradiance, formula 5, SCUM, as *Q 151*
float 7: minimum algae density
float 8: maximum algae density

**Q 291** Special data set for nutrient depletion from algae growth, for two nutrients and one algae shading the light. Two integers and nine floats are read:

integer 1: integer pointing to nutrient parameter
integer 2: integer pointing to algae variable
integer 3: integer pointing to the other nutrient variable
float 1: a - regression constant in formula for specific vertical attenuation coefficient
float 2: b - regression constant in formula for specific vertical attenuation coefficient
float 3: c0 - formula for growth rate, k, irradiance.
float 4: c1 - formula for growth rate, k, irradiance.
float 5: kmax - maximum growth rate
float 6: Ks constant for the nutrient being calculated
float 7: Ks constant for the other nutrient
float 8: temperature coefficient
float 9: fraction of nutrient in algae growth

The data set is used for both nitrogen and phosphorous. It is similar to Q 281, but the last float is the fraction of the algae growth.

**Q 361** Special data set to calculate algae growth in case of two nutrients+light limiting the growth:

integer 1: integer pointing to algae concentration
integer 2: pointer to light irradiance variable
integer 3: pointer to phosphorous variable
integer 4: pointer to nitrogen variable
float 1: c0 - formula for growth rate, k, irradiance.
float 2: c1 - formula for growth rate, k, irradiance.
float 3: kmax - maximum growth rate
float 4: Ks constant for phosphorous
float 5: Ks constant for nitrogen
float 6: temperature coefficient

**Q 371 S**pecial data set for nutrient depletion from algae growth, for multiple algae species limiting the light, and two nutrients. Three integers and seven floats are read:

integer 1: integer pointing to nutrient parameter
integer 2: integer pointing to light variable
integer 3: integer pointing to algae variable
integer 4: integer pointing to the other nutrient variable
float 1: c0 - formula for growth rate, k, irradiance.

float 2: c1 - formula for growth rate, k, irradiance.
float 3: kmax - maximum growth rate
float 4: Ks constant for the nutrient being calculated
float 5: Ks constant for the other nutrient
float 6: temperature coefficient
float 7: fraction of nutrient in algae growth

The data set is similar to *Q 291*, except for the extra integer pointing to the light irradiance, and the omission of the attenuation coefficients.

**Q 1001** The second number is a float, giving the value of a constant source.

**Q 1002** The second and third numbers are floats. The source term is:

Source = float1 * (float2 - variable )

Example: *Q 2 1 0.0005 20.0*

Source1 = 0.0005 * (20.0 - variable1 )

This term can be used for calculation of temperature fluxes across the water surface, where the first float is the heat exchange coefficient for the water surface and the second float is the air temperature.

**Q 1003** The second number is an integer. This is an index for the first variable in the source term. The third number is an integer, giving the second variable in the source term. The fourth number is a float, which is multiplied with both variables.

Example: *Q 3100 1 2 1 0.04*

This means that a source term for equation 1 is used, and the source term is 0.04 multiplied with the value of variable 2, multiplied with the value of variable 1 (the same variable as calculated for this source term)

**Q 1012** The second number is an integer. This is an index for the variable in the source term. The third number is a float, giving a coefficient which is multiplied with the variable.

Example: *Q 1012 1 2 0.05*

This means that a source term for equation 1 is used, and the source term is 0.05 multiplied with the value of variable 2.

**Q 1060** Special data set to calculate temperature flux at the water surface. Six floats are read:

float 1: multiplication factor, *B*, for irradiance from the timei file (1.0)
float 2: a coefficient, *A*, for light attenuation (0.5-0.7)
float 3: air vapour pressure, eair (mmHg)

float 4: reflection coefficient, RL (0.03)
float 5: water emissivity, e (0.97)
float 6: Bowen's coefficient, c1 (0.47 mmHg/C)

**Q 1102** Special data set to calculate gas transfer at the water surface. The data set has two floats as parameters: cair and a coefficient, r, which is multiplied with the eddy-viscosity to give the effective diffusion. The data set is only applied to the cells bordering the water surface.

**Q 2001** The second number is a float, giving the value of a constant source.

**Q 2002** The second number is an integer. This is an index for the variable in the source term. The third number is a float, giving a coefficient which is multiplied with the variable.

**Q 2003** The second number is an integer. This is an index for the first variable in the source term. The third number is an integer, giving the second variable in the source term. The fourth number is a float, which is multiplied with both variables.

**Q 2030** Special data set to calculate resuspension, where three floats are read. The first float is the critical shear stress for initiation of resuspension. The second integer is the coefficient *a* in the resuspension equation, being proportional to the shear stress surplus raised to the third coefficient. The critical shear stress is given in Pascal.

Example: *Q 2030 3 0.3 0.1 1.5*

The resuspension is calculated for variable no. 3, and the critical shear stress is 0.3 Pa.

**Q 6100** Special data set for calculating irradiance when there are more than one algae species or sediment concentration. The data set reads six integers and ten floats. The first integer is the number of algae species. The following integers are indexes for the variable numbers of the algae. Then ten floats are read, which are pairs of coefficients for the light attenuation curve.

Example: *Q 6100 9 2 5 6 0 0 0 0.07 0.2 0.06 0.23 0.0 0.0 0.0 0.0 0.0 0.0*

Here, variable no. 9 is the light irradiance. Two species of algae is used: variable no. 5 and no. 6. The light transmissivity coefficients are 0.07 and 0.2 for the first algae species and 0.06 and 0.23 for the second algae species.

Note that the component reducing the light does not have to be an algae. It is also possible to specify for example an inorganic sediment.

### 5.3.11 The *S* data set

The S data sets gives the size and fall velocity of the sediments. An integer is first read, indicating the size group. Then the diameter in meters are given and then the fall velocity in m/s.

Example: *S 1 0.001 0.06*

Sediment size 1 has a diameter of 1 mm, and a fall velocity of 6 cm/s.

Note that the coarsest sediment size should be number 1, and then the finer sizes should follow with the finest size having the highest index.

### 5.3.12 The *T* data set

*T* Title field. The following 30 characters are used in the graphics.

### 5.3.13 The *W* data sets

**W 1** Stricklers number, discharge and downstream water level. This data set must be present in the file. The parameters given here are used to generate the water level for the calculations using a standard backwater calculation.

Default values for SSIIM 2: *W 1 50.0 1.0 1.0*

Note that the Strickler value *M* is *1.0/n*, where n is the Manning's number.

**W 6** *Fixedpoints* - a point which is used in the *GridEditor*. Two integers are read, which is the numbers of the i and j grid lines. The intersection of these lines are not moved by the elliptic grid generator. One *W 6* data set is required for each *fixedpoint*. Maximum 19 999 points can be used. The *W 6* data set is usually generated by the *GridEditor*. When the *control.new* file is written, it contains the W 6 data sets. These can be copied to the *control* file.

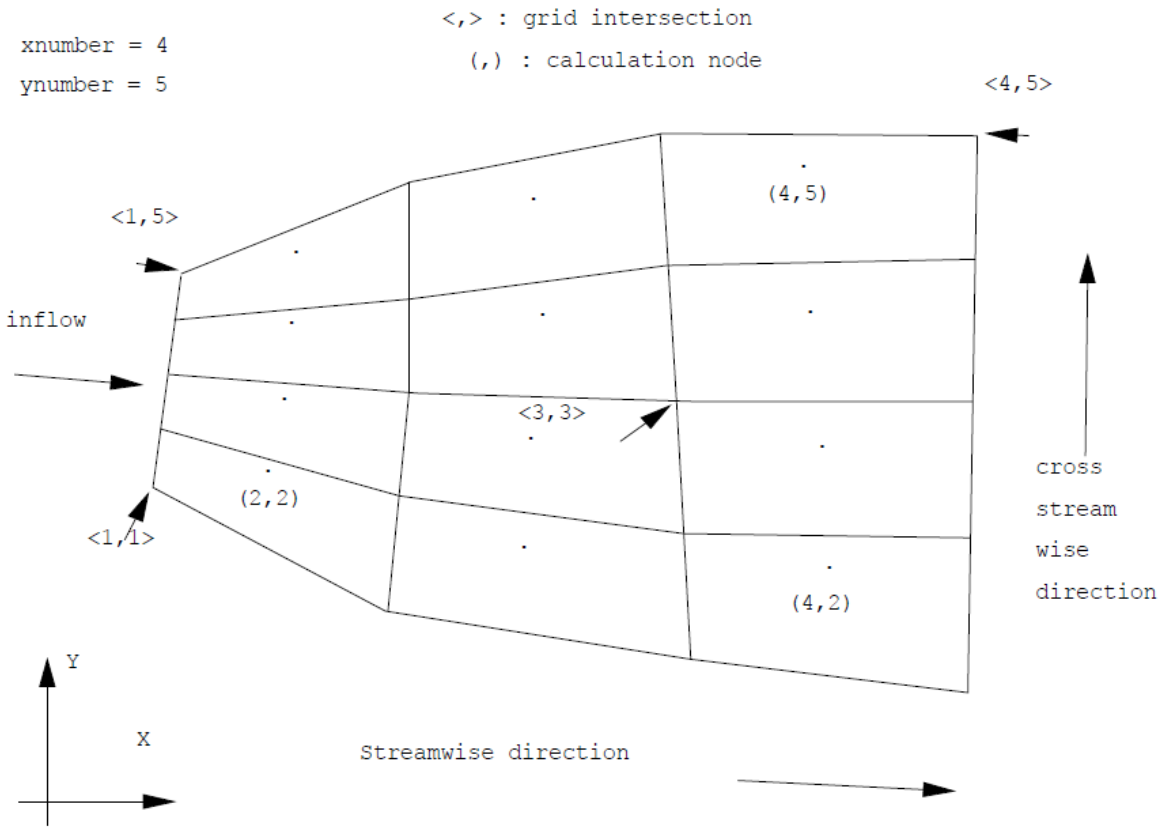
**W 7** Attraction point used in the *GridEditor*. Each *W 7* data set represents attraction to one grid line or point. Maximum 1999 attraction points can be used. An integer is read first which tells the type of attraction. The following options are given:

0: Point attraction in i-direction
1: Point attraction in j-direction
2: Line attraction in i-direction
3: Line attraction in j-direction

Then two integers are read, that tell which grid line intersection the attraction is towards. For the line attraction, only one of the integers are used. Then the two attraction parameters are read, which are floats.

## 5.4 The *koordina, koosurf* and *koomin* files

The *koordina* file describes the bed of the geometry with a structured grid (SSIIM 1). An example is show in the figure below. The grid can be made using a map, a spreadsheet or the *GridEditor*.




The necessary input data is the *x*, *y* and *z* coordinates of the points where the grid lines meet. The format of the data is given below.

*i j x y z*

An example:

*1 1 0.34 0.54 0.11*
*1 2 0.35 0.66 0.12*
...

The first two numbers are integers, while the following three are floats. The numbers are read in a free format, which means that the distance between them does not matter. The sequence of the points are not important, as long as all points are included. This is not controlled by the model, so the user must do the check by looking at the grid in the graphic modules of the program.

If a tunnel is simulated, or the user wants to specify the water surface, an additional floating point number is read for each line. This gives the top level for each grid intersection. An example:

*1 1 0.34 0.54 0.11 1.21*
*1 2 0.35 0.66 0.12 1.33*
...

The file *koordina* file that contains the water level is called *koosurf*. The *koosurf* file is mainly used in SSIIM 2, while the *koordina* file was used in SSIM 1.

Some words about indexing and numbering of grid lines and cells. The variable names for the number of grid lines in the three directions are:

*xnumber* : number of cross-sections
*ynumber* : number of longitudinal lines, or number of points in a cross-section
*znumber* : number of horizontal planes, or number of lines in the vertical direction

The numbering of the grid lines goes from 1 to *xnumber* in the streamwise direction, and similarly for the other two directions.

However, the grid lines define cells between the grid lines. The variables are calculated in the center of each cell. This means that a numbering system for the cells is also required. The word node is often used for the center of a cell.

From a geometrical view of the grid, it is observed that the number of lines always exceeds the number of cells by one in each direction. When the arrays are defined, it is therefore a choice for the programmer to start the numbering of the cells on one or two. The choice that is made in SSIIM is that the numbering starts on two. This means that the cell that is defined by grid lines i=1 and i=2 and j=1 and j=2 has the number (2,2). Cell number (1,1) does not exist. The numbering of the cells is also shown in Fig. 1. The numbering of the grid lines is shown with the <,> sign, while the numbering of the calculation nodes is shown with the (,) sign. The grid is non-staggered.

The data on the *koordina* file defines a surface. It is possible to make a file with exactly the same format and call it *koomin*. This surface is then used as a minimum elevation surface for bed changes. The bed will not be lowered under this surface.

If SSIIM terminates right after startup and the *boogie* file contains the following message: Error, negative areas for cell i=13, j=2, or some other combinations of i and j, then there is an error in the *koordina* file. The indexes denote the cell number, so for i=13, j=2, the x and y coordinates for the following grid line intersections should be checked: (13,2); (13,1); (12,2) and (12,1).

The *G 1* data set allocates arrays for the grid. The values on this data set must always be larger than the xnumber and ynumber values in the grid. Otherwise, and error message is written to the *boogie* file and the program stops.

## 5.5 The *unstruc* file

The geometry data is stored in the unstruc file. This file contains the coordinates of all grid line intersections, which cells are connected with other cells, which surfaces are connected to which cells etc. It also contains information about inflow/outflow of water and water quality constituents.

Because of the complexity of the file, it is not possible to generate this file using an editor or a spreadsheet. This file must be generated by SSIIM 2.

Note that the first line in the file gives the grid size for allocation of arrays. The first integer is a number below 10, for the files generated with the newest versions of SSIIM 2. The second number is the number of grid cells. The third number is the number of surfaces. The fourth number is the number of blocks in the grid. The fifth number is the number of connections between the blocks. The sixth integer is the number of connection surfaces between the blocks. The seventh integer is the number of outer surfaces on the nested blocks. The last integer on the line is the number of points in the grid.

This information can be used to set the size of the arrays allocated by SSIIM 2 on the *F 65* data set.

Example: *1 22500 75600 1 0 0 0 308041*

The line specifies that the *unstruc* file is generated by a newer version of SSIIM 2, since the first integer is below 10. The grid has 22500 cells, 75600 surfaces and 308041 points. It consists of only one block, and there are no connections between blocks (since it is only one block) and no nested surfaces (since there are no nested blocks).

In *unstruc* files generated by older versions of SSIIM 2, the first integer is the number of points in the grid. This will always be above 10. Then there will be one less integer on the first line, compared with the *unstruc* file produced by the newest SSIIM 2 versions. The newest SSIIM 2 version can read both formats and is therefore backwards compatible with old *unstruc* files.

**Automatic grid changes**

The default water surface elevation is completely horizontal when starting to make a grid using SSIIM 2. When modelling a river, one often would start with a sloping water surface. Also, one might want to start with a dry bed that can be wetted later. The unstruc file must also contain some information about the grid in the dry areas. The initial grid of the unstruc file therefore must cover the whole geometry that can be wetted, which means a fairly large water surface elevation must be used. Starting the computation with a lower water surface is then done by using the *F 112* data set. If the integer on this data set is set to 1, then the grid is regenerated right after the unstruc file is made. Before the regeneration, the *koordina* file is read. In the *koordina* file, the water surface elevation can be specified, and this can be a sloping surface. The water surface can also be below the bed, giving a startup with a partially dry geometry.

## 5.6 The *geodata* file

This file contains a number of *x*,*y* and *z* coordinates. An example is shown below:

*E 2.2 3.3 3.4*
*E 4.5 3.3 2.2*
*E 3.3 4.2 1.2*

*Z 3.0 3.0 3.0*

The letter *E* is used for counting the number of points. The letter Z is used for the last line in the file. The numbers on this line are not used.

The purpose of this file is to use geometrical data that has been obtained from the field, a digitized map or a GIS system.

The file is used in the *GridEditor*, to display the points in the file when generating the grid. The View option of the menu and the *geodata* points option in the pull-down menu activates this. The grid points are displayed with different colors according to which level they are.

The second use for this file is also in the *GridEditor*. It is then used to compute the vertical elevation of the bed in the grid. A linear interpolation procedure is used. The procedure is further described in Chapter 4.3.

The *GridEditor* in SSIIM 2 makes it possible to add new *geodata* points graphically. This is done by selecting *Geodata -> Add/delete geodata points* in the menu. New points are then added graphically, with the left mouse pointer. Existing points can be deleted by cliking on the points with the right mouse button. Afterwards, the geodata points can be written to a new *geodata* file. Note that only 5000 new *geodata* points can be added at once. If you want to add more points, write the *geodata* file to the disk, and then read it back again. Then you can add another 5000 points. This procedure can be repeated as many times as you like.

## 5.7 The *bedrough* file

This file is used to give a roughness height to individual bed cells. Values in this file overrides the value calculated from Manning-Strickler's coefficient, and the value given on the F 16 data set. On each line a character, two integer and a float are given. The first character is a B, and the two following integers are indexes for the bed cell. The float is the roughness in meters. An example is given below:

*B 19 2 0.001*
*B 19 3 0.001*
*B 19 4 0.001*

## 5.8 The *porosity* and *vegdata* files

**Note:** When calculating porosity, use a *P* on the *F 7* data set in the *control* file. When using the *vegdata* file, the *F 115* data set has to be included in the *control* file

The *porosity* file is only used in SSIIM 1. The *vegdata* file is used in both SSIIM 1 and SSIIM 2.

The two files are used when the bed of the river is covered by stones or vegetation, introducing a sink term for the velocities.

**The *porosity* file (SSIIM 1 only)**

The file describes the location and magnitude of the porosity or vegetation in the geometry. An example of a porosity file is given below:

*P 17 6 3.349774 3.399189 3.450101 3.499517 0.000000 0.700000 0.833333 1.000000*
*P 17 7 3.358273 3.413603 3.470610 3.525940 0.000000 0.653846 0.807692 1.000000*
*P 17 8 3.403323 3.426084 3.449536 3.472297 0.000000 0.642857 0.785714 1.000000*

First, the character *P* is read. Then two indexes for the *i* and *j* number of the bed cell is read. Then four vertical levels are read, which have the same zero reference as the *koordina* file. The porosities in each of these levels are then read.

The porosity file can be made from a *koordina* file and a *geodata* file. The module that does this is activated from the main menu of the user interface, from File and Make porosity file in the pull-down menu. The procedure goes through each element and used the points in the *geodata* file to obtain the data. The procedure is described in more detail in Chapter 2.3.

**The *vegdata* file**

The vegetation is modelled as a number of vertical cylinders in each cell. A drag formula is used to determine the force between the water and the stems. The drag formula computes the sink in the velocity equations as proportional to the velocity squared.

The vegdata file is similar to a porosity file, in that each line describes one bed cell and each line starts with a letter and then two indexes for the cell is given. Then four levels are given.

There are different types of vegdata set. The types are identified by the first letter in the line.

A V data set is similar to the P data set, except instead of the porosity, four vegetation values have to be given. The vegetation values are defined by multiplying the following three parameters:

1. The drag coefficient for the vegetation stems. This is usually around unity
2. The diameter of the stems (in meters)
3. The number of stems in each cell.

As an example, say there are five stems in each cell, with diameter 7 cm, and the stems are circular with a drag coefficient of 1.0. The vegetation parameter becomes: 1.0x.07x5 = 0.35.

An A data set is similar to a V data set, except the number of stems pr. square meter is given instead of number of stems in each cell.

Example:

*A 76 16 37.05 37.20 37.35 37.55 3.0 1.0 1.2 0.5*
*A 76 17 37.05 37.20 37.35 37.65 3.0 1.0 0.8 0.5*
*A 76 18 37.05 37.20 37.25 37.75 4.0 1.0 1.1 0.5*
...

A *B* data set is similar to an *A* data set, but 4x4 more floats are read on each line. First, four numbers are read, giving the number of stems pr. square meter for each of the four levels. Then four numbers are read giving the diameter of the stems. The next four numbers are the drag coefficients. The final four numbers are empirical parameters in the epsilon equation. The additional terms are used for source terms in the *k* and ε equations.

The vegetation parameter can be seen in the map graphics. This is a good way of testing that the correct values are given in the input file.

Using SSIIM 2, there is only one index for each cell. In the *vegdata* file, two indexes are given. The two indexes corresponds to the indexes used in a structured grid. This system is necessary as in SSIIM 2, as the numbering of the cells changes over time during wetting and drying.

If the vegetation changes over time, it is possible to describe the vegetation at several times in the *vegdata* file. To do this, a *T* data set has to be given between each group of vegetation data sets. The T data set also has to have a floating point number, which will be the time when the data set is valid. The program will interpolate between the different times given in the *vegdata* file.

When time-dependent changes in the vegetation is computed, then the *F 201* data set has to be used in the *control* file. Also an *U* data set has to be used in the vegdata file. The *U* data set has to have the same integers as on the *F 201* data set.

**Example**:

The *control* file: contains *F 201 1 4 3* (three times)

The *vegdata* file:

*U 1 4 3*
*T 0.0*
*A 76 16 37.05 37.20 37.35 37.55 3.0 1.0 1.2 0.5*
*A 76 17 37.05 37.20 37.35 37.65 3.0 1.0 0.8 0.5*
*A 76 18 37.05 37.20 37.25 37.75 4.0 1.0 1.1 0.5*
... (all A data sets for this geometry and time= 0.0 seconds)
*T 10000.0*
*A 76 16 37.05 37.20 37.35 37.55 5.0 2.0 1.2 0.5*
*A 76 17 37.05 37.20 37.35 37.65 5.0 2.0 0.8 0.5*
*A 76 18 37.05 37.20 37.25 37.75 5.0 2.0 1.1 0.5*
.. (all A data sets for time=10000 seconds)
*T 30000.0*
*A 76 16 37.05 37.20 37.35 37.55 3.0 1.0 1.2 0.5*
*A 76 17 37.05 37.20 37.35 37.65 3.0 1.0 0.8 0.5*

*A 76 18 37.05 37.20 37.25 37.75 4.0 1.0 1.1 0.5*
.. (all A data sets for time=30000 seconds)

## 5.9 The *alldata* file

The *alldata* file is an attempt to combine the information from the *unstruc*, *result* and *bedres* files into one file. The idea was to be able to restart a morphological computation from one file. This sometimes works and sometimes there are problems.

The file can be written from the menu or by giving an *F 389* data set in the *control* file. The integer specifies how often the *alldata* file is written in a transient computation. If the integer is 1000, the *alldata* file is written each 1000$^{th}$ time step. The file can also be read before the computation starts. Then an *unstruc* file is not needed.

## 5.10 The *result* file

This file contains the results from the water flow calculations, with velocities, pressure and turbulence. The *result* file is written when the prescribed number of iterations have been calculated or when the solution has converged. The results are velocities in three dimensions, $k$, ε, pressure, and the fluxes on all the walls of the cells. The data from this file is used as input for the sediment flow calculations. This file can also be read when the user wants to start the water flow calculations from where the *result* file was last written (hot start).

An example of a result file for SSIIM 2:

*Results, iter = 3601*
*Residuals: 0.054663 0.079991 0.018223 0.029072 5.240989 218.573547*
*Roughness : 0.005000*
*U 16316 11940 39679 1 0 0*
*i u v w k e p*
*C 1 5.20752712e-002 3.87287893e-002 -2.91152289e-002 1.83900666e-004 2.20008542e-004 6.98685775e+001*
*C 2 -2.97928650e-003 1.18274871e-002 -2.83168656e-002 4.95138596e-005 1.79039199e-005 6.98566087e+001*
*C 3 -2.34130748e-003 1.18835706e-002 -2.32318333e-002 2.81046242e-005 1.03435021e-005 7.16544156e+001*
*C 4 -4.91827925e-003 3.44957829e-002 -1.59890336e-002 5.50987357e-005 3.38688381e-005 7.04038680e+001*
*C 5 -9.62809076e-003 5.08444254e-002 -1.13473535e-002 7.34700051e-005 1.08660059e-004 6.95710922e+001*
*C 6 -4.36046706e-003 -4.54215010e-002 -2.87338762e-004 1.13683579e-004 4.87075237e-004 6.32869612e+001*
*C 7 -4.54495962e-003 -4.54039464e-002 -2.24038016e-004 2.32742813e-004 2.86840184e-004 6.32996242e+001*

.....
*F 25735 4.35128401e-002*
*F 25736 -4.35876127e-002*
*F 25737 0.00000000e+000*
*F 25738 0.00000000e+000*
*F 25739 0.00000000e+000*
*F 25740 1.61427895e-001*
*F 25741 9.94855359e-002*
*F 25742 0.00000000e+000*
*F 25743 0.00000000e+000*
*F 25744 1.27727494e-001*
*F 25745 1.46505934e-001*
*F 25746 0.00000000e+000*
..

The first lines give the residuals, the roughness and the grid size. The *U* data set gives the number of points, number of cells and number of surfaces in the grid. Then the number of blocks, connections and connection surfaces are given. After the *U* data set, the water velocity parameters are given for each cell. Each line gives the values in one cell. The line starts with the letter *C* starts the line, then the number of the cell is given in the unstructured grid. Then the three velocities, the *k* and ε values and the pressure for this cell is given. After all the data for the cells, the fluxes on all the surfaces are given. Each line gives a flux for one surface. The line starts with the letter *F*. Then the number of the surfaces is given. And then the flux is given, in kg/s of water.

All the results are given in SI units, so that for example the velocities are in meters/second, turbulent kinetic energy is given in $m^2/s^2$, epsilon is given in $m^2/s^3$ and the pressure is in Pascal, or Newton/$m^2$.

## 5.11 The *con2res* file

The *con2res* files contains the sediment concentrations in the grid. It is written after the sediment concentration calculation has finished. Each line in the file contains first one index indicating the cell number. Then the total concentration is written and then *lnumber* floats that give the concentration for the sizes. An example is given below:

*1 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00*
*2 5.075079e-04 0.000000e+00 1.026459e-04 1.030081e-04 1.032351e-04*
*3 4.064470e-04 0.000000e+00 1.011788e-04 1.015358e-04 1.017596e-04*
*4 4.004061e-04 0.000000e+00 9.967497e-05 1.000267e-04 1.002471e-04*

If a file *conres.pre* exist, this will be read by the program before the sediment calculation starts. The *conres.pre* file has the same format as the *conres* file. Note only the sediment concentrations are given in the file, and not the grain size distribution of the bed. The concentrations are given in volume fractions.

## 5.12 The *interpol* and *interres* files

Vertical profiles of velocity or concentrations are sometimes needed. Coordinates for the locations where the profiles are wanted are given in this file. When the write results routine is activated and the integer on the *F 48* data set in the *control* file is above 1, it will search for the *interpol* file. If this file is not found, it will proceed normally and write the result or the *conres* file. If the *interpol* file is found and the integer on the *F 48* data set is above 1, the program will not write to the *result* file, but write the interpolated vertical velocities to a file named *interres*. An example of an *interpol* file is given below:

*M 2.03 0.5*
*M 4.06 0.39*
*M 4.06 0.5*

The character *M* is read first, and then the *x* and *y* coordinates for the point one wishes to interpolate the vertical profile to. If the value on the *F 48* data set is 3 the concentrations are interpolated. If the value on the *F 48* data set is 2, the velocities, *k* and ε are written to the file. If the value on the *F 48* data set is 1, the bed levels are written (no profile, only one point). The results are written to a file called *interres*.

An example of an *interres* file is given below:

*C 9.250000 0.100000, size = 0*
*1.530130 0.000000e+00*
*1.491877 5.266262e-05*
*1.415370 9.303652e-05*
*1.338864 1.349549e-04*
*1.262357 1.740915e-04*
*1.185851 2.075269e-04*

The first line starts with a *C*, and then the *x* and *y* coordinates of the vertical profile is given. Two columns follow. The first column is the *z* value of the vertical profile. The second column is the concentration. If velocities are interpolated instead of concentrations, an *M* is written on the first line instead of the *C*. Instead of the concentration column, five columns are written. The variables in the columns are: velocity in *x*, *y* and *z* directions, *k* and ε.

**Time-dependent sediment computations**

If a time-dependent sediment computation is done, then the question is at what time should the concentrations be written? If the integer 19 is used on the *F 48* data set, then a time in seconds will be read first on each line of the *interpol* file. Also, a *T* is used as an index instead of an *M*. Example:

*T 400.0 2.03 0.5*
*T 800.0 4.06 0.39*
*T 900.0 4.06 0.5*

The *interres* file will then contain the sediment concentrations at this time.

The time in the file is given in seconds.

**Printing cross-sections for ParaView from SSIIM 2.**

Sometimes the user wants to see variables in a cross-section from a SSIIM2 grid. This can be done by specifying the integer 12 on the *F 48* data set, and print the "result" file. Then a ParaView file will be written from the cross-section defined in the *interpol* file. Only one cross-section can then be given in the *interpol* file.

## 5.13 The *verify* file

This file was used as input for the *VerifyProfile* graphics option, which was used in SSIIM 1. It is not used in SSIIM 2, but it is described in this Users Manual as some *verify* files exist that may be of use in presenting measured results in a spreadsheet.

The horizontal location of the profiles were determined by the user, so this method of presentation could be used for both cross-sections, longitudinal section or other user-defined sections. An example of a *verify* file is shown below:

*P 40.0 10.0 3*
*5.0 0.5 0.1*
*6.5 0.6 0.12*
*7.0 0.8 0.05*
*P 70.0 10.0 2*
*6.0 1.0 0.1*
*8.0 1.5 0.8*

Each profile is identified with a capital *P*. Then the *x* and *y* coordinates of the point follow. The coordinates are given in the same system as the grid. After the coordinates, an integer is read. The integer tells how many points are measured in this profile. Up to 11 points can be given. On the following lines the data is given. There must be the same number of lines as the integer on the *P* line. Three floats are given on each line. The first is the vertical coordinate where the data is taken. This is given in the same coordinate system as the grid. The following two floats are velocities or concentrations, depending on what is calculated. If velocities are calculated, the second float is the velocity component in the x-direction and the third float is the velocity in the y- direction. If concentration is calculated, the second float is the measured concentration and the third float is a dummy number which is not used. If concentrations are given, the user can choose in the graphics presentation whether a total concentration or a fractional concentration is presented.

The example above gives two profiles. The first is located at x = 40.0 meters and y = 10.0 meters. Three data points have been measured. The first point is located at global coordinate z = 5.0 meters, and have a measured x-component velocity of 0.5 m/s and a y-component velocity of 0.1 m/s. The second point is located at global coordinate z = 6.5 meters, and have a measured x-component velocity of 0.6 m/s and a y-component velocity value of 0.12 m/s.

## 5.14 The *timei* and *timeo* files

The two files are relevant to transient calculations. The first file, *timei*, is an input file for time series of discharge, waterlevel, sediment concentration and control for output. The second file, *timeo*, is an output file with time series from the model.

There are several controls for the correctness of the data given in the *timei* file. If an error is detected by the program, a message will be written to the *boogie* file and the program will terminate.

The timei file can read different types of data sets, which all begins with a capital letter. The data sets beginning with *I* and *O* can be read in both SSIIM 1 and SSIIM 2.

**The *O* data set**

The *O* data set controls the output to the *timeo* file. A maximum of 20 *O* data sets can be used. The data set begins with a capital *O*, and then reads a float, *Time*, and an integer, *Vars*. *Time* is the time for when the print out starts, and *Vars* tells how many variables are written to the *timeo* file for each time step. Maximum 3000 variables can be written. After reading *Time* and *Vars*, the next lines read which variables are printed and in which cell. *Vars* lines are read. Each line starts with a lower case character, and then three integers (four in case of sediment concentration) are read. This is for SSIIM 1. For SSIIM 2, only one integer is read (two for sediment concentrations). The integers indicate which cell the variable is located in. If sediment concentration is read (*c*), the last integer tells which fraction is written. 0 indicates the sum of the fractions. If water quality is read (*q*), another integer is also read, corresponding to which constituent is written. The characters corresponding to different variables are listed below:

| character | variable |
|---|---|
| *u* | velocity in x-direction |
| *v* | velocity in y-direction |
| *w* | vertical velocity |
| *p* | pressure |
| *k* | turb. kin. energy |
| *e* | epsilon |
| *d* | turb. diffusion |
| *z* | vertical grid elevation |
| *c* | sediment concentration * |
| *q* | water quality component * |

* These variables requires an extra integer read after the index(es) for the cell(s)

The following characters will produce one or more variables that are characteristic for the whole grid. A letter and an integer are read. For most of the data sets, the integer will not be used. If it is used, it is

given in the table below.

| | |
|---|---|
| *b* | Water surface elevation difference between upstream and downstream boundary (meters). Two correction factors for free water surface algorithm (dimensionless). Three values are computed. |
| *g* | Water discharge ($m^3/s$) |
| *n* | Number of cells in the grid |
| *o 0* | Bed shear force computed from the turbulent kinetic energy and the bed cell area, summed up for all bed cells. Force given in Newton. |
| *o 1* | Bed shear force computed from the water surface slope and the water depth, and the bed cell area, summed up for all bed cells. Force given in Newton. |
| *o 2* | Bed shear force computed from the bed velocity and the bed cell area, summed up for all bed cells. Force given in Newton. |
| *o 3* | Bed shear force computed from the pressure gradients at the bed the bed cell area, summed up for all bed cells. Force given in Newton. |
| *r* | Average water depth for whole grid (meters) |
| *s* | Bed level change in $m^3$. |
| *t* | Pressure at upstream and downstream boundary (Pascal). 2 values are computed. |
| *x* | Max. residual for sediment concentrations. |
| *y* | Water levels at upstream and downstream boundary (meters). 2 values are computed. |
| *i* | Average width of eroded channel (meters), based on bed elevation changes, and average width of channel (meters), based on wetted areas (with cells). |

**The *I* data set**

The second type of data set is input data. The line starts with a capital I and then five floats are read. The first float indicates the time for when the following variables are used. The second float is the upstream water discharge. The third float is the downstream water discharge. The fourth float is the upstream water level, and the fifth float is the downstream water level. Sometimes one of the last variables are unknown, and then a negative value can be inserted and the program will try to calculate the value.

If the TFS method (*F 37 1*) is used in SSIIM 1, lnumber floats are read additionally, indicating the upstream inflowing sediment concentration (volume fraction).

If the TFS method is used in SSIIM 2, *N* x *lnumber* floats are read additionally. *N* is the number of inflow groups. The numbers indicate sediment concentration (volume fraction) of the inflowing sediments for each grain size. The sediment concentrations are grouped according to the inflow group. So that first, all the concentrations for the first group is given. Then the next group etc.

An example of a *timei* file for SSIIM 1 is given below:

*O 0.0 6*
*u 2 2 2*
*c 2 2 2 0*
*p 2 2 2*
*z 41 3 6*

*z 1 2 1*
*z 2 2 1*
*I 0.0 10.0 10.0 -20.0 19.5.0 0.000*
*I 100.0 10.0 10.0 -20.0 19.0 0.000*
*I 200.0 10.0 10.0 -20.0 18.5 0.000*
*I 300.0 10.0 10.0 -20.0 18.0 0.000*
*I 400.0 10.0 10.0 -20.0 17.5 0.000*

For SSIIM 2, the downstream water level on the I data set in the *timei* file corresponds to the first point in the *G 6* data set in the *control* file. If two fixed points are given in the *G 6* data set, the upstream water level on the *I* data set in the *timei* file will correspond. Note that this means the corresponding indexes on the G 6 data set and the I data set in the *timei* file are given in reverse order.

**The *D* data set**

For SSIIM 2, it is possible to have more than one inflow and one outflow group. Then it is also possible to vary the water discharge in each group over time. This is done using the additional *D* data set in the timei file. The data set gives the discharges for each time step, similar to the *I* data set. The first float read is the time. This must be identical to the time on the corresponding *I* data set. Then ten additional floats are read. The first nine are discharges in the nine first discharge groups. The last float is the water level. Note that zero's are given for the discharge groups that are not used.

If the water inflow is not equal to the outflow, there will be a water continuity error. To suppress the error message, and keep the program running, the *F 85* data set in the *control* file can be used.

An example of the *timei f*ile for SSIIM 2 is given below. There are three sediment sizes given in the *control* file. In the *unstruc* file, two inflow groups are given and one outflow group.

*I 0 0.0 0.0 0.0 0.0 0.0 0.0012 0.0027 0.0 0.00029 0.00032*
*I 86400 0.0 0.0 0.0 0.0 0.0 0.0018 0.0021 0.0 0.00026 0.00038*
*I 172800 0.0 0.0 0.0 0.0 0.0 0.0011 0.0017 0.0 0.00027 0.00029*
*D 0 5.6 0.74 5.9 0.0 0.0 0.0 0.0 0.0 0.0 33.0*
*D 86400 5.4 0.54 15.1 0.0 0.0 0.0 0.0 0.0 0.0 33.0*
*D 172800 5.7 0.84 25.7 0.0 0.0 0.0 0.0 0.0 0.0 32.9*

The time-varying discharge can also be used in the same way for water quality computations.

**The *Q* data set**

This data set only works for SSIIM 2. Ten floating point numbers are read. The first is the time step. The following nine are discharges in the nine discharge groups.

**The *C* data set**

This data set only works for SSIIM 2. A time step is first read. Then n times m floats are read, where m is the number of sediment sizes as given on the *G 1* data set, and n is the number of discharge groups where there is an inflow.

**The *M* data set**

The *M* data set us used to specify time-dependent inflow of water quality constituents for SSIIM 2. An *M* data set is also needed in the *control* file for each of the varying inflow concentrations. But instead of using the *I* or *J* data set in the *timei* file, an *M* data set is used. On this data set, the time is given first, similar to the *I* data set. Then a float is given, which is a variable time step. The variable time step is not coded yet, but it has but be given in the data set. Then the three floats for the wind is given, if wind is specified in the *control* file. Then up to 20 inflow concentrations are given. Note that the depth and water discharges from the *I* data set is not given. The number of inflow concentrations is the same as the number of *M* data sets in the *control* file. Irradiance read as the last number of the line, if specified by the *Q* data sets in the *control* file.

The order of the *M* data sets in the *control* file is not important, but the order of the data on the M data in the timei file has to be the same as the order of the *M* data sets in the *control* file.

**The *timeo* file**

The *timeo* file gives the output time series from the calculation. For each time step, a line of variables are written to the file. Each line has Vars floats, according to which variables were given in the timei file. In the above file six variables are written. The first variable is the velocity in x-direction for cell (2,2,2). The second variable is the sum of all sizes of concentration for cell (2,2,2). The pressure and vertical grid elevations for various grid intersections are written.

The second group of parameters in the *timei* file is an input time series. Each time step in the series are given on one line. The line starts with a capital *I*. Then the time step is given. Then four floats are read. This is the upstream and downstream water discharge and upstream and downstream water level, respectively. If a negative value is given, the program will try to calculate the value. The last float(s) are sediment concentration(s) for the various sediment sizes. These are only read if the transient sediment calculation is used. Note that the time steps does not have to correspond to the time steps of the program. The sum of the calculated time steps is calculated by the program and compared with the time step in the file. If the calculated time exceeds the time from the file, the values from this data line will be used. The times must be in increasing order.

An example of a *timeo* file corresponding to the above *timei* file (SSIIM 1) is given below:

*0.0000e+00*
*2.0000e+00 8.804354e-01 6.837692e-04 -1.853305e+01 1.950000e+01 1.792479e+01*
*4.0000e+00 9.185002e-01 7.081983e-04 -3.949717e+01 1.950000e+01 1.792458e+01*
*6.0000e+00 1.100297e+00 1.121773e-03 5.289925e+01 1.950000e+01 1.792421e+01*
*8.0000e+00 1.156041e+00 1.389383e-03 8.122703e+01 1.950000e+01 1.792374e+01*
*1.0000e+01 1.207883e+00 1.614782e-03 9.418469e+01 1.950000e+01 1.792317e+01*
...

Each line corresponds to a time step. A time step of 2 seconds have been used. The first float is the calculated time. The second float is the velocity in cell (2,2,2). Then the sum of the concentrations in cell (2,2,2) is written. Then the pressure and the vertical grid elevations.

This format is chosen so that it is easy to import the file into a spreadsheet for presentations.

## 5.15 The *inspace* file

The inspace file enables the user to give a spatially varying inflow sediment concentration at the upstream boundary. The concentration is given as a function of the (*x,y,z*) coordinates of the center of the cell surfaces at the inflow boundary. The following function is used:

$$c = k_1 + k_2 x + k_3 y + k_4 h + k_5 x^2 + k_6 y^2 + k_7 h^2 + k_8 xy + k_9 xh + k_{10} yh \qquad (5.22.1)$$

The *x,y* and *z* coordinates use the same coordinate system as the grid. Instead of using the *z* coordinate, the *h* parameter is used, where *h* is the water depth, or the distance between the water surface to the center of the inflow surface, *z*. In other words: *h* = water level - *z*.

The values of the coefficients $k_1$-$k_{10}$ are read on the *B* data sets or the *C* data sets in the inspace file. The *B* data sets are used if the concentrations do not change over time. The *C* data sets are used to specify time-variations of the parameters. For multiple grain sizes, one *B/C* data set is used for each size. An integer is read first, before the $k_1$-$k_{10}$ values. The integer denotes the sediment size.

For time-variation of the parameters in Eq. 5.22.1, the number of time steps in the file must be given on the *F 294* data set in the *control* file. Also, *T* data sets must be given between each series of *C* data sets. The *T* data sets also contain a float, which is the time when the data set is used.

The *inspace* file starts with A 1.

Example: Two sediment sizes and two time steps:

```
A 1
T 0.0
C 1 0.0 0.0 0.0 0.0 0.0 0.0 0.001 0.001 0.0002 0.0002
C 1 0.0 0.0 0.0 0.0 0.0 0.0 0.002 0.001 0.0002 0.0002
T 3600.0
C 1 0.0 0.0 0.0 0.0 0.0 0.0 0.003 0.001 0.0002 0.0002
C 1 0.0 0.0 0.0 0.0 0.0 0.0 0.004 0.001 0.0002 0.0002
```

## 5.16 The *bagger* file

The *bagger* file is used to specify a dredging operation. "Bagger" is the German word for dredging. The dredging function was first used in the Iffezheim reservoir, on the German-French border (Olsen and Hillebrand, 2018).

The dredging computations are invoked with the *F 386* data set in the *control* file. An integer is read, indicating how many dredging operations are done in the simulated time period. If the integer is larger than zero, then SSIIM will search for a file named *bagger* . The file describes the location, time and amount of the dredged material.

The *bagger* file has three types of data sets: *T, N* and *K*.

***T***: The *T* data set specifies the time period for the dredging. Two floats are read: the start and the end time of the dredgning. The times are in seconds, in the same reference as the times in the *timei* file. A maximum of 100 *T* data sets (dredging operations) can be used.

***N***: The *N* data set specifies which area of the grid is to be dredged. Four integers are read: $i_1$, $i_2$, $j_1$ and $j_2$. The integers are indexes for the (i,j) grid lines, as specified in the *koordina/koosurf/koomin* files. The lowest *i* index is denoted $i_1$, and the largest *i* index is $i_2$. The same for the *j* direction.

The *N* data set must follow directly after the *T* data set. There can be up to 20 *N* data sets for each *T* data set.

The *K* data sets contain the bed coordinates for grid line intersections at the end of the dredging period. The coordinate values are similar to what is given in the *koordina* file. For example:

*K 3 5 34.4 124.3 8.4*

The two integers give the grid line intersection indexes, while the following three floats give the *x,y* and *z* coordinates of the bed **right after** the dredging operation. Note that the *x*, *y* and *z* coordinates must be in the same reference system as the *unstruc/koordina* file. It is only the *z* coordinate that is used by the dredging algorithm.

The *K* data sets follows directly after the *N* data set. If there are more than one *N* data set for each *T* data set, then the following sequence is to be used:

*T* (first dredgning)
*N* (first area to be dredged)
*K* (first grid line intersection)
*K*
..
*K* (last grid line intersection)
*N* (second area to be dredged)
*K* (first grid line intersection in the second area)
*K*
..
*K*
*T* (second dredgning)
etc.

The volume of dredged material for each time step the dredging occurs is written to a file called *bag_res*. This is produced by SSIIM 2.

**Generating the *K* data sets**

A convenient method to generate the *K* data sets is from a point clout of measured bed elevations after the dredging operation. Often, such a point cloud have been measured to document how much material is removed from the bed. The measured coordinates can be transformed to a *geodata* file. Then this file is read into the *GridEditor* using the computational grid of the geometry. Then the bed is interpolated from the *geodata* points in the *GridEditor*, and the grid is regenerated. Then the *koordina* file is written from the SSIIM 2 menu. This file will have the correct data for the *K* data sets in the *bagger* file. The *koordina* file is read into a spreadsheet, and the data for the dredged area is copied to another spreadsheet. Then the *K* is added in the first column, before the data from the *koordina* file. This is then saved to an ASCII file and copied into the *bagger* file.

Remember to store the *unstruc* file first in another location, and copy it back to the working directory before the simulation is started.

## 5.17 The *bedres* file

The *bedres* file contains information about the bed sediments, including grain size distribution, thickness and bed form height. It also contains information about the bed roughness. This information is important when storing the results from a run that takes long computational times. The *bedres* file can then be read by SSIIM 2, to produce graphical results from the run, with regards to the sediments. When the *result* file is written from SSIIM 2, the *bedres.t* file is also written at the same time. If the user renames the file to *bedres* without an extension, then SSIIM 2 will search for this file before it reads the *result* file. It will then use the *bedres* file to update the grid so that it will be similar to the grid written to the *result* file. The result file will then be read into the grid it was produced by.

When using the P 10 option in the *control* file, multiple *bedres* files are written during a time-dependent run. These files can also be renamed to have no extension, and read by SSIIM together with the *result* file from the same time step.

The second line in the *bedres* file gives the structured grid size and the number of sediment sizes as the first four integers. This is the same integers as on the *G 1* data set in the *control* file. Then the number of 2D depth-averaged cells are written, and then the maximum depth. The maximum depth is used to determine the number of cells in the vertical direction, and is used in the same formula as the parameter on the *F 87* data set. The last number on the second line is an integer, giving how many cells there are in the grid.

Then there is a block of data where each line gives information about one depth-averaged cell. The first two integers on the line gives the i and j indexes of the cell in the 2D structured grid. The third number is a float. The integer value of this float gives the cell number of the bed cell in the 2D grid. Then four additional floating point numbers are given: the bed level, the water level, the limit of movable bed

(from the *koomin* file) and the bed movement.

The last block of data gives information for each 2D depth-averaged cell in the grid. The 2D cell number is given first (which is not the same as the 3D cell number). Then the 3D cell number is given, then the roughness is given twice, the first time from a 3D array and the second time from a 2D array. Of course, these should be the same. Then the bedform height is given. After this, there will be 2(1+L) floating point numbers given, where *L* is the number of sediment sizes, as given on the *G 1* data set. First, the thickness of the active (top) bed layer is given in meters. Then the sediment fraction for each sediment size is given. Then the same is repeated for the inactive layer.

If multiple layers are used (*F 37 3* and *F 134*), then *N*(1+2L) floating point numbers are given, where *N* is the number of bed layers as given on the *F 134* data set.

The combination of reading the *bedres* and the *result* files have not always been successful. To avoid this problem, the *alldata* file can be used instead. It contains the grid, bed sediment information and the information in the *result* file.

## 5.18 The *habitat* file

The *habitat* file gives the availability function commonly used in analysis of fish habitat. The file is written at the same time as the *result* file. The availability function is divided in 20 groups, and tabulated so that it can easily be imported into a spreadsheet. This can be used to determine the preferences for the fish.

The file is written when the result file is written, if the *F 48* data set has the parameter 5.

## 5.19 Files for ParaView

SSIIM can generate graphic files that are read directly into the ParaView program. There are basically two version of the files: A 2D version and a 3D version. The files can be written from the user interface through the menu or from the main program. The main program can write multiple files for one time-dependent run, making it possible to create animations in ParaView. The selection whether a 2D or a 3D file is generated can be specified on the *F 48* data set.

The files written by the main program (not by the SSIIM user interface menu) can write user-specified variables. The specification of the variables is given on the *G 24* data set. This data set reads a number of data groups, one group for each variable. Up to 30 groups/variables can be used. Each group consists of one character and two integers. The following table shows the details:

| **Variable** | **Character** | **First integer** | **Second integer** |
|---|---|---|---|
| Horizontal velocity | *u or V* | Level | Not used |

| | | | |
|---|---|---|---|
| Vertical velocity | *w* | Level | Not used |
| Pressure | *p* | Level | Not used |
| Turbulent kinetic energy | *k* | Level | Not used |
| Epsilon | *e* | Level | Not used |
| Eddy-viscosity | *d* | Level | Not used |
| Sediment concentration | *c* | Level | Size number |
| Water quality | *q* | Level | Parameter number |
| Residual for horizontal velocity | *H* | Level | Not used |
| Residual for vertical velocity | *W* | Level | Not used |
| Residual for pressure | *R* | Level | Not used |
| Residual for turbulent kinetic energy | *K* | Level | Not used |
| Residual for epsilon | *E* | Level | Not used |
| Porosity or vegetation parameter | *P* | Level | Not used |
| Water level | *v* | Not used | Not used |
| Water depth | *y* | Not used | Not used |
| Froude number | *F* | Not used | Not used |
| Depth-averaged horizontal velocity | *D* | Not used | Not used |
| Number of cells in the vertical direction | *i* | Not used | Not used |
| Bed level | *z* | Not used | Not used |
| Bed shear stress | *m* | Not used | Not used |
| Roughness | *r* | Not used | Not used |
| Roughness to depth ratio | *h* | Not used | Not used |
| Bed form height | *b* | Not used | Not used |
| Sediment fraction | *f* | sediment layer no. | Not used |
| Sum of all sediment layer thicknesses | *l* | Not used | Not used |
| Sediment thickness of layer | *L* | sediment layer no. | Not used |
| Sediment thickness based on cell corners | *t* | Not used | Not used |
| Bed sediment $d_{50}$ | *a* | Not used | Not used |
| Bed movement, accumulated over time | *s* | Not used | Not used |
| Special temperature gradient | *x* | Not used | Not used |
| Bed angle | *B* | Not used | Not used |
| Secondary current angle | *C* | Not used | Not used |
| Potential bed sediment concentration | *O* | Not used | Not used |

| | | | |
|---|---|---|---|
| Groundwater level | *G* | Not used | Not used |
| Bed changes the last time step | *U* | Not used | Not used |
| Sediment slide movements | *M* | Not used | Not used |
| Bed solid fraction | *N* | sediment layer no. | Not used |
| Sediment cohesion | *j* | sediment layer no. | Not used |

The *Paraview.vtk* file can be viewed in the ParaView program.

In SSIIM 2, a three-dimensional ParaView file is written instead of a *result* file if the *F 48 10* is given in the *control* file. A more advanced option is to use the F 329 data set for making the ParaView files.

**Making files with cross-sections for ParaView from SSIIM 2.**

Sometimes the user wants to see variables in a cross-section from a SSIIM 2 grid. This can bed done by specifying the integer 12 on the *F 48* data set, and print the *result* file. Then a ParaView file will be written from the cross-section defined in the *interpol* file. Note only one cross-section can then be given in the file.

Both the Tecplot and Paraview programs may show strange results if using UTM coordinates with a large number of digits are used for the grid.

**Making multiple files simultaneously**

The *F 48* option in the *control* file has its limitations with respect to the choice of files that is written. A more flexible option is to use the *F 329* option. Then the user can chose to make (or not to make) fourteen different files for each print-out iteration given on the *P 10* data set.

## 5.20 The *fracres* file

The *fracres* file contains information about the thickness of the active and inactive layer and the grain size distribution in the two layers. Each line in the file describes one bed cell. The first two integers are the i and j values, identifying the cell in a structured grid.

After the two integers, a floating point number is given, which is the thickness of the active sediment layer in meters. Then the fractions of each sediment size are given as floating point numbers. After that, the thickness of the inactive layer is given (meters). Then the fractions for the inactive layer are given. A total of 2 x ( 1 + n ) floats are then read, where n is the number of sediment fractions.

Example: Three sediment sizes:

2 3 0.01 0.4 0.3 0.3 2.0 0.3 0.4 0.3

The line is for cell (2,3). The thickness of the active layer is 0.01 m. In the active layer there is 40 % sediments of size 1, 30 % of size 2 and 30 % of size 3. The inactive layer has a thickness of 2.0 meters, and contains 30 % of sediment size 1, 40 % of size 2 and 30 % of size 3.
The *fracres* file can be made by using a spreadsheet.

The *fracres* file is read by using an *O* (letter, not zero) on the *F 2* data set. Or reading the *compres/con2res* file from the menu. Also note that the active layer thickness in the *fracres* file must be the same as what is given on the *F 106* data set. If the *F 106* data set is not used, the active layer must be the same as the maximum grain size on the *S* data sets.

Note that the order of the bed cells in the *fracres* file is not important. Also note that the values for the same cell can be given multiple times. Then it is only the last value that will be used. This "feature" can be useful when regions of different sediments are to be described. Then the whole geometry can be covered in one particle distribution, and then the region of different distribution can be given afterwards. It is not necessary to remove the originally given values before adding new ones. This can be nested as many times as necessary. The only limitation is the length of the *fracres* file, which must have less lines than 20 x the number of bed cells.

The file can be made by using a spreadsheet.

After the *fracres* file is made, the user should read it and look at the grain size distribution and the sediment thickness in the SSIIM graphics. Then possible mistakes can be seen.

The *fracres* file specifies the thickness of each sediment layer, and thereby also the thickness of the total amount of sediments on the bed. This information is also given by the difference between the actual bed level and the limit of movable bed given in the *koomin* file. If the two ways of specifying the sediment thickness differ, then the input is undefined. In SSIIM 2, the *F 310* data set modifies the values from the *koomin* file or the *fracres* file to give correspondence.

After the file is made, the user should read it and look at the grain size distribution and the sediment thickness in the SSIIM graphics. Then possible mistakes can be seen.

## 5.21 The *bedangle* file

The purpose of the bedangle file is to enable the user to give a spatially varying angle of repose for the sediments on the bed. The file is only used if an *F 274 1* data set in given in the *control* file.

The angles of repose are given on one line for each bed cell in the grid. Each line has two integers and four floating point numbers. The two integers identify the bed cell. Then four floating point numbers are read. The first two floating points are the angle of repose used to compute the reduction in the critical shear stress for erosion of a particle, for upstream slopes and downstream slopes. These values are typically used in the Brooks or Dey's formula. The values are given in degrees. Typical values are 25-50 degrees.

The third floating point number on the line is a lower limiter for reduction of critical shear stress. The reduction of the critical shear stress will not be below this value. The parameter should be between 0 and 1.

The first three parameters can also be given values on the *F 109* data set, for all the cells. Note that the angles given on the *F 109* data set is the inverse tangens to the angle, not the angle in degrees as given in the *bedangle* file.

The fourth parameter is the angle of repose used in the sand slide algorithm. This is also given in degrees. The variation in angle of repose for the sand slide algorithm is only coded for the *F 56 200* option.

Example:

*3 4 40.0 35.0 0.2 33.0*

# Chapter 6. Tutorials

The tutorials show the use of the input files, GridEditor and DischargeEditor. A fairly coarse grid is used as the figures are then clearer. Also, the computational time is much shorter. For actual cases, it is also recommended to start with a coarse grid. When the computations are running successfully on the coarse grid, make a finer grid.

## 6.1 Tutorial 1. Plastic particles in a straight flume

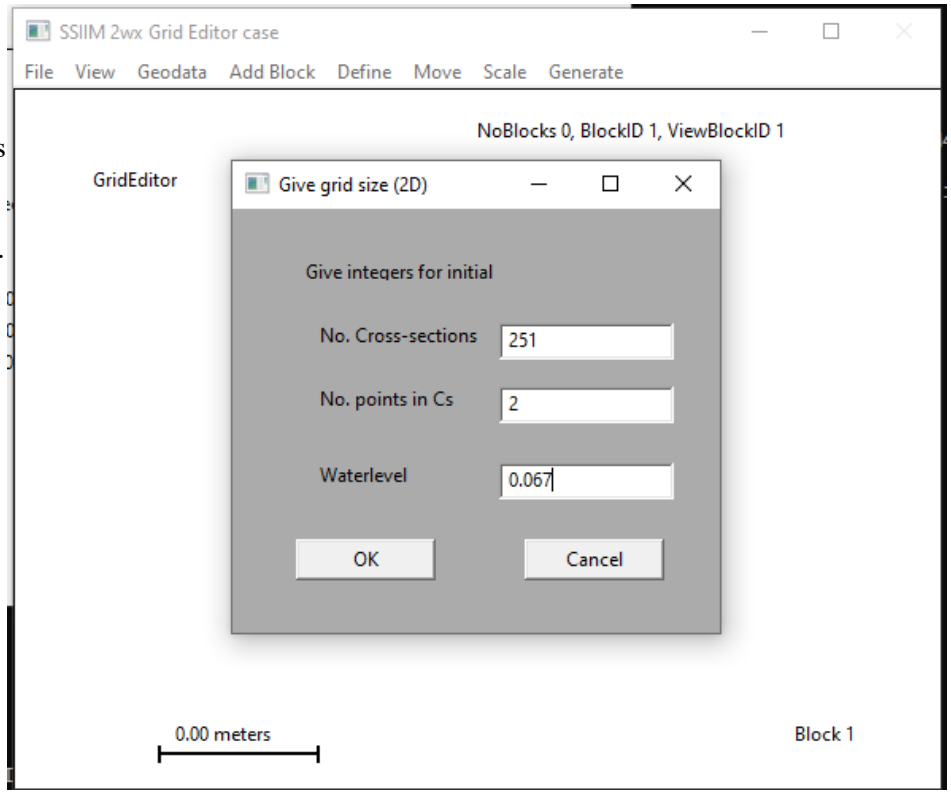
Born et al (2023) carried out laboratory flume tests of suspended plastic particles in a straight flume, where vertical concentration profiles were measured. Olsen et al (2023) computed a profile using an OpenFOAM solver, *sediDriftFoam*. It is also possible to use SSIIM 2 to compute the plastic concentrations.

### 6.1.1- Generating the grid.

A 2D width-averaged grid is used here. The flume was 12.4 meters long and the measurements were made 7 meters downstream of the entrance. Here, we will only model the most upstream 8 meters, similar to what Olsen et al (2023) did. The water depth was 6.7 cm. The width is set it to 0.3 meters. This is not important in a width-average grid as long as we get the discharge/average water velocity correct.

We use 251 cross-sections and 31 grid lines in the vetical direction. This gives 250 cell in the streamwise direction and 30 cell in the vertical direction. The cell length will be 8 meters / 250 cells = 32 mm. The cell height will be 67 mm / 30 = 2.2 mm. The aspect ratio of the cell is fairly large, so more cells in the streamwise direction could be used. However, the water flow is one-dimensional and then this may be ok.

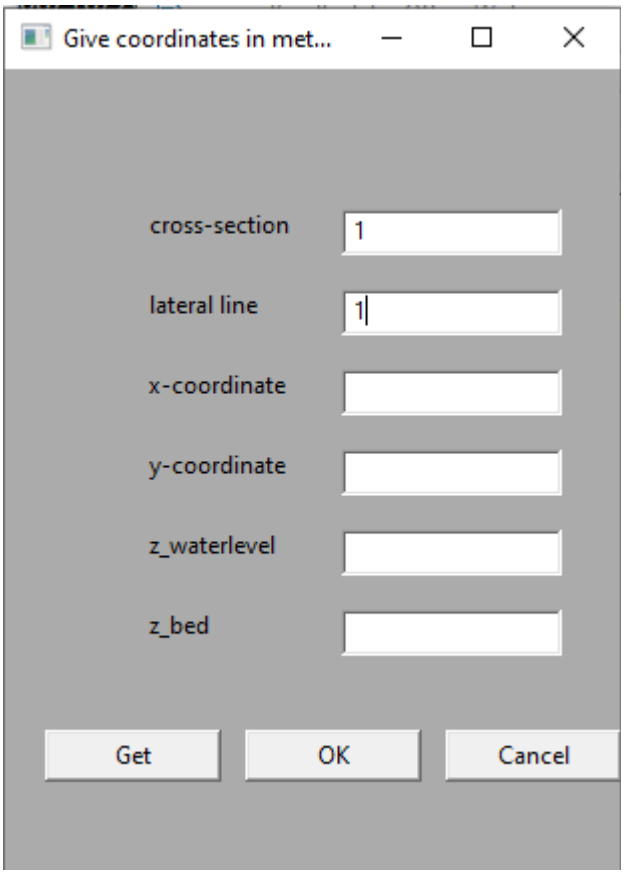
The SSIIM 2 program is started and the *GridEditor* is chosen. The number of cross-sections are given. A 2D width-averaged grid will have 2 points in a cross-section, one on the left side and one on the right side. The water level is given equal to the water depth. The default bed is located at level z=0.0.

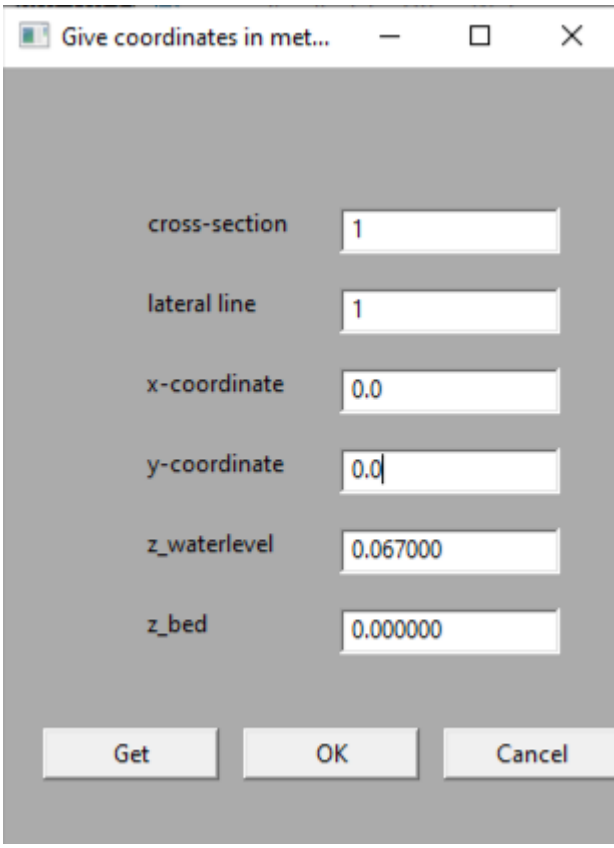



On the menu, choose A*dd block->New - size*. A dialog box like shown in the figure on the right emerges. Give the data as shown in the figure and click *OK*.

The default grid that is made has cell sizes 1x1 meters. This is not what we want, so we have to give the correct (x,y) values of the four corners of the grid.

We go to the menu option *Define->Coordinates*. A dialog box emerges. We want the main flow direction of the channel to be in the x direction and y is the lateral direction. The upstream right corner is then located at x=0 meters,y=0 meters. In the dialog box we give in 1 as the cross-section and 1 as the lateral line, as shown in the figure below to the left. Then click on *Get*. This gives the current (incorrect) (x,y) values. Change this to 0 and 0 as shown in the figure to the right, below.




Then click *OK*. Do the same with the other four corners. Point (1,2) have coordinates x=0 meters and y=0.3 meters. Point (251,1) have coordinates x=8.0 meters and y=0.0 meters and it is the downstream right corner. The downstream left corner is point (251,2) and it has coordnates x=8 meters, y=0.3 meters. After all four corner points are given, choose the menu option *Generate->sides* and *Generate->TransfiniteI*. This gives the correct outline of the grid. Then choose *Generate->3D grid*. Finally, go back to the menu of the main program (before you started the *GridEditor*) and on the menu choose *File->Save unstruc*. Make sure the file is completely written before you end the program. You can see it by observing that the file size doesn't increase. Also, a text is written in the main window, if you choose *update* on the menu.

### 6.1.2 Making the *control* file

It would have been possible to use the *DischargeEditor* to specify the discharges. However, in the current case it is easier to specify the discharges in the *control* file. This is an ordinary text file that can be made by Notepad in Word or another text editor. The file has to be called control, and it can not have any extension. Often, Notepad will add a .txt extension, but this has to be removed. This can be difficult in the graphics user interface. If a command prompt window is opened i Windows, it is possible to use the command *ren control.txt control*.

The water velocity in the current case was 1.4 m/s. The discharge is therefore 1.4 m/s * 0.3 m * 0.067 m = 0.028 $m^3/s$. The following data sets must in the control file. The data sets are given by a letter and a number. The text

following the data on the data set are explenations of the data set.:

```
T  tutorial                                  title text
F 2 US                          read the unstructured grid and compute the plastic particle concentrations
F 16 0.0001                  roughness of the bed/walls
F 33 5.0 10 time step and number of inner iterations
F 37 2 sediment computation
F 41 0.000001  thickness of bed sediment layer
F 168 8 multigrid solver
F 185 1 0.4 turbulence damping at water surface
F 200 1 0.0003 0.004 inflow k epsilon
F 202 1 0.0001 logarithmic inflow profile
F 237 1 0.028 water discharge inflow
F 237 2 0.028 water discharge outflow
F 263 2 1 zero gradient outflow
F 314 1 1 inflow/outflow specification when grideditor is not used
F 417 1 plastic sediments
G 1 300 20 31 2 grid and sediment sizes
K 3 0.4 0.4 0.4 0.05 0.2 0.2  relaxation coefficients for the Navier-Stokes solver
K 5 0 0 0 10 0 0 multigrid method for the pressure
S 1 0.2 0.1   sediment  diameter and fall velocity for size 1  (bed)
S 2 0.0015 -0.0396   sediment diameter and fall velocity for size 2 (plastic)
N 0 1 1.0  100 % gravel on bed
N 0 2 0.0   0 % plastic on the bed
```

### 6.1.3 Making the *timei* file.

The inflow concentrations in SSIIM 2 have to be given in a *timei* file. This file can specify a time-varying concentration, but in this tutorial it will be constant.

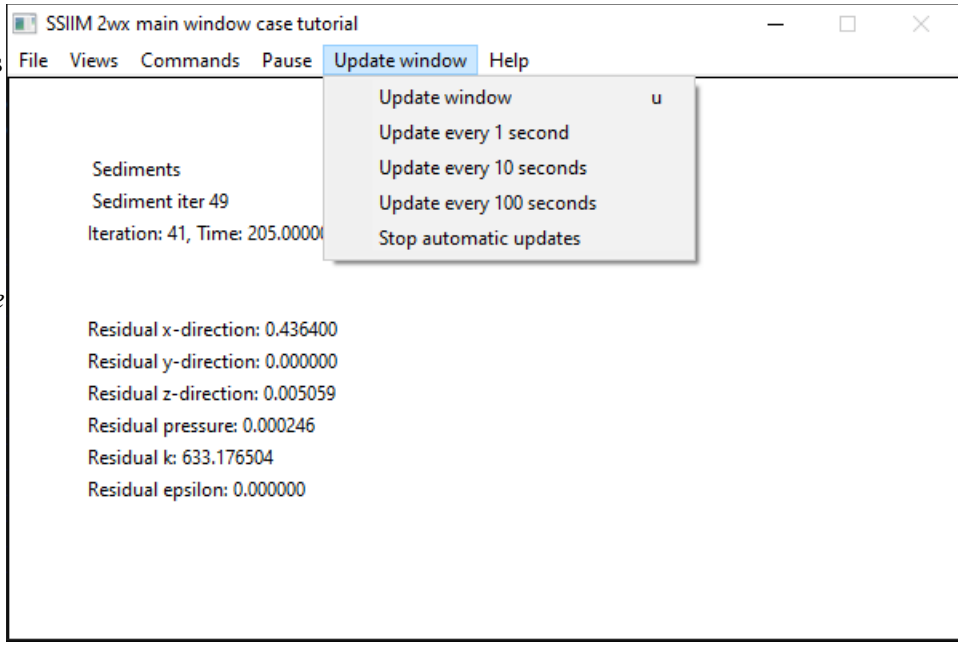
The sediment concentrations are given on the I data sets in the timei file. The following file gives zero concentration of size 1 and 0.0001 of size 2 (100 ppm by volume). The D data set gives the discharges in an out of the geometry.

The absolute value of the negative numbers are not important in the file. The important numbers are highligted with yellow colour.

```
I 0.0 -0.1 0.028 -1.5 -0.2457 0.0 0.0001 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
I 90010.0 -1.0 0.028 -1.5 -0.2457 0.0 0.0001 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
D 0.0     0.028  0.028 0.0          0          0          0          0          0          -852
D 90010.0 0.028  0.028 0.0          0          0          0          0          0          -852
```

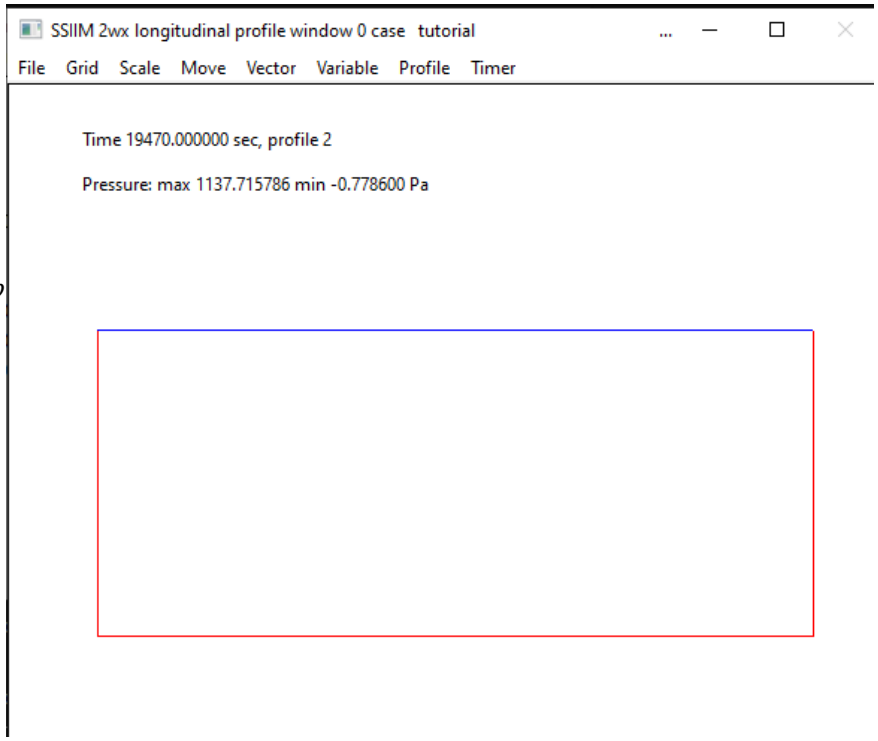
### 6.1.4 Compute the water flow and particle concentrations

Start the program from the directory where the *control* file and the *unstruc* files are located. You can do this graphically, by making a directory where these two files are located, and also copy the ssiim2wx538.exe file there. Then double-click on the executable file. A window shows up on the screen, like in the figure to the right. Go to the menu option *Update Window*, or type *u* on the keyboard. You will see information about the iteration and the residuals for the Navier-Stokes equations.




### 6.1.5 View the results

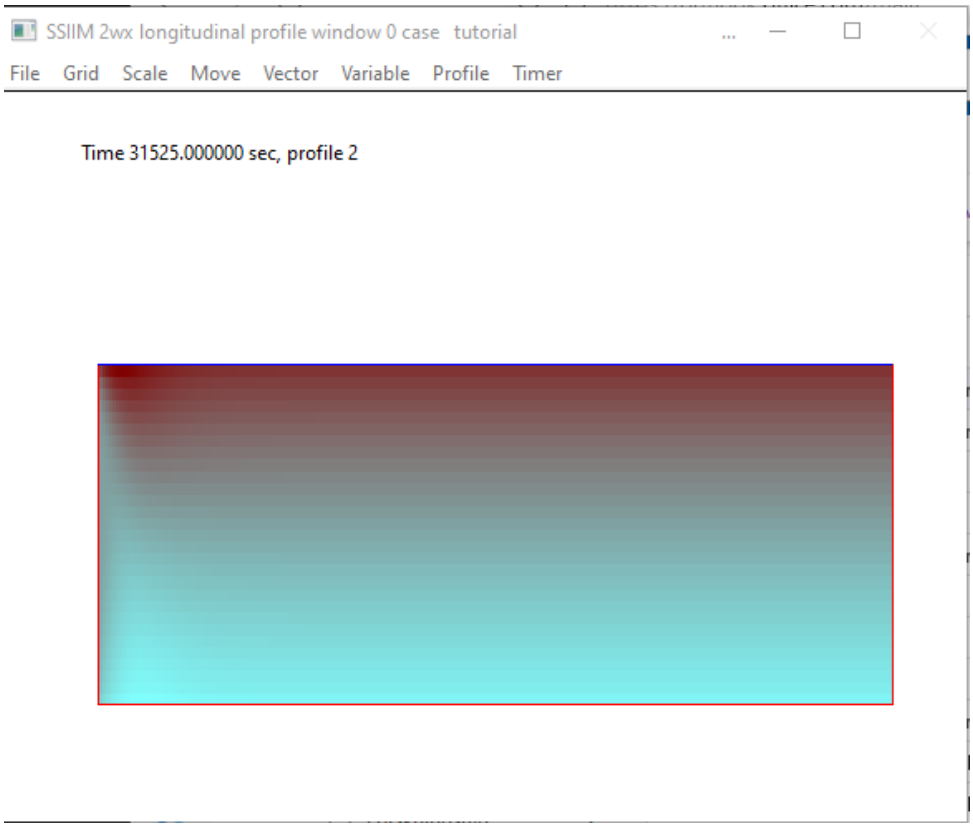
After a time of around 2000 seconds, you can pause the computation. Click on *Pause->Before SIMPLE corrections*. Then choose on the menu *Views->Longitudinal profile*. A new window emerges. The profile is 8 meters long and 6.7 cm high, so it is difficult to see the vertical resolution. Push the *PageDown* key on the keyboard one time and then *Ctrl+Arrow Up* multiple times until you see the profile with a reasonable distortion in the vertical direction, like the figure to the right..



Then on the menu choose Variable->Sediment concentration. And *Grid->Fill*. This is a simple colour map of the concentrations, with one colour for each cell, as seen below.

You may also choose other variables.

This simple graphics is not meant as a replacement for more advanced programs, for example ParaView.




**6.1.6 Get a profile of the concentrations to plot in a spreadsheet.**

This is most easily done using a file called *interpol*. It gives one or more 2D coordinates (x,y), and SSIIM 2 then writes a file called *interres* with the vertical profile in this/these points.

The measured concentration profile for our case is 7 meters downstream of the upstream boundary. And in the middle of the flume. The (x,y) coordinates are therefore (7,0.15), since the channel is 0.3 meters wide. The interpol file then only has one line:

*M 7.0 0.15*

To get the *interres* file written, we have to restart the computation with the *interpol* file in the same directory as the other files. But first we need to add the following data sets to the *control* file:

*F 329 0 0 0 0 0 1 0 0 0 0 6 0 0 0 0 0*
*G 24 6 k 0 0 e 0 0 u 0 0 d 1 1 p 0 0 c 0 0*

After the program has restarted and run to a steady state, go to the menu of the main (first) window, and choose *File->write result file*. Then the *interres* file is produced, and also a file called *paru3_00000.vtk*. The vtk file can be used to view the graphics in ParaView. Which has much better graphics options than the window from the previous chapter. The *interres* file can be importet into a spreadsheet.

## 6.2 Tutorial 2. Two-block grid

This tutorial is meant for first time users of the SSIIM 2 program. The tutorial shows the main features of the user interface, with the grid editing and the presentation graphics. The user is not required to edit files, but some knowledge of grids is recommended. The tutorial does not show the more advanced features of the program, which necessitates editing of the input files.

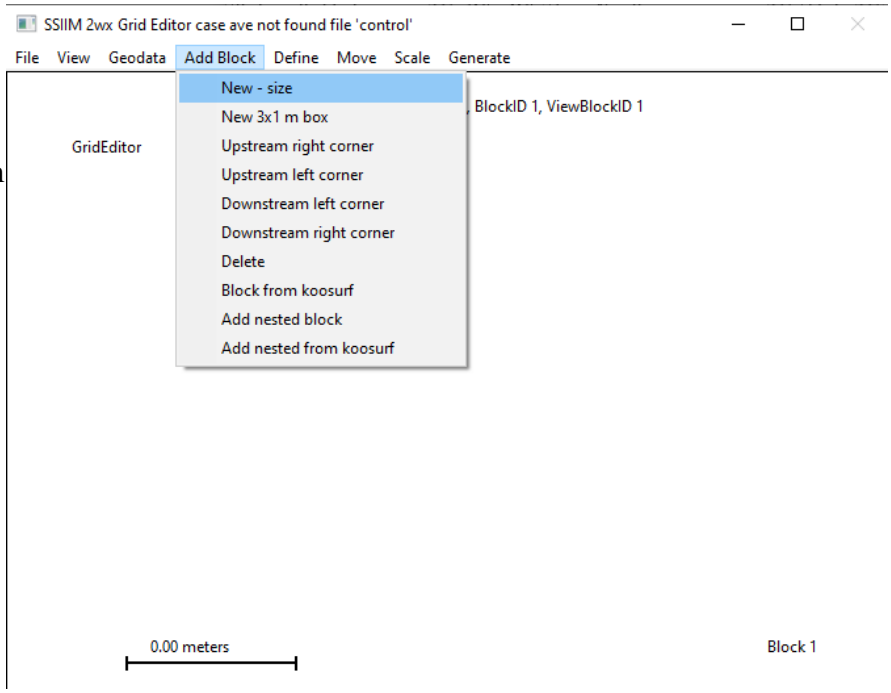
The tutorial is divided in four stages. During the first stage the grid is made. The second stage shows how to specify inflow/outflow water discharge. Then the calculations of the velocity flow field is made in the third stage. The presentation graphics is shown in the fourth stage.

The main user interface is a window with a menu. Choosing *Text* in the *View* option of the main menu, the intermediate results from the calculations and other messages are shown. The menu is also used for viewing graphics and starting different modules of the SSIIM program.

### 6.2.1 Generating the grid

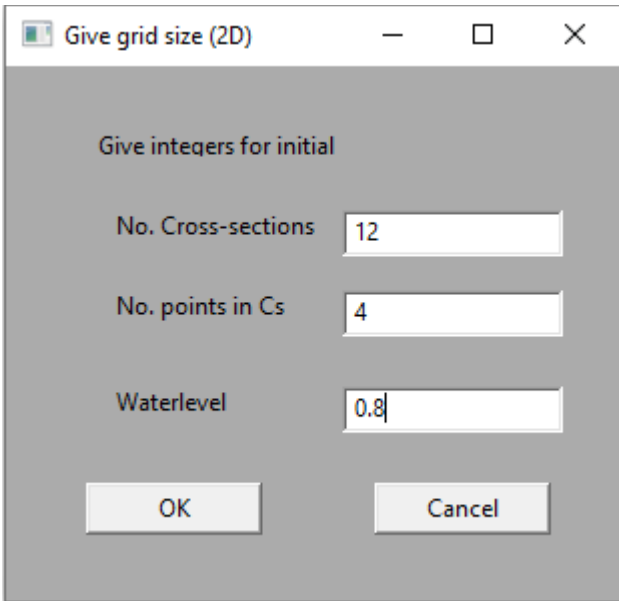
Start up SSIIM from a directory where the *control* and *koordina* files do not exist. This is done by writing c:\path\ssiim2wx538 <CR> when you are on this directory. The "path" is here the path were the executable program ssiim2wn.exe is placed. Or you can copy the ssiim2wx538.exe file to the working directory and double click on the program.

After the main window is shown on the screen, click on the *GridEditor* in the *View* option of the menu. This starts the *GridEditor*, and a white area appear in the window where the grid is made. Click on *Add block->New-size* in the menu.




A dialog box emerges, as shown on the next page. In the dialog box, give in the number of cross-sections and the number of points in each cross-section. This intial grid is orthogonal, structured and has cells that are 1x1 meters in the horizontal direction.

The grid is seen by choosing on the menu *View- >View grid cells*.

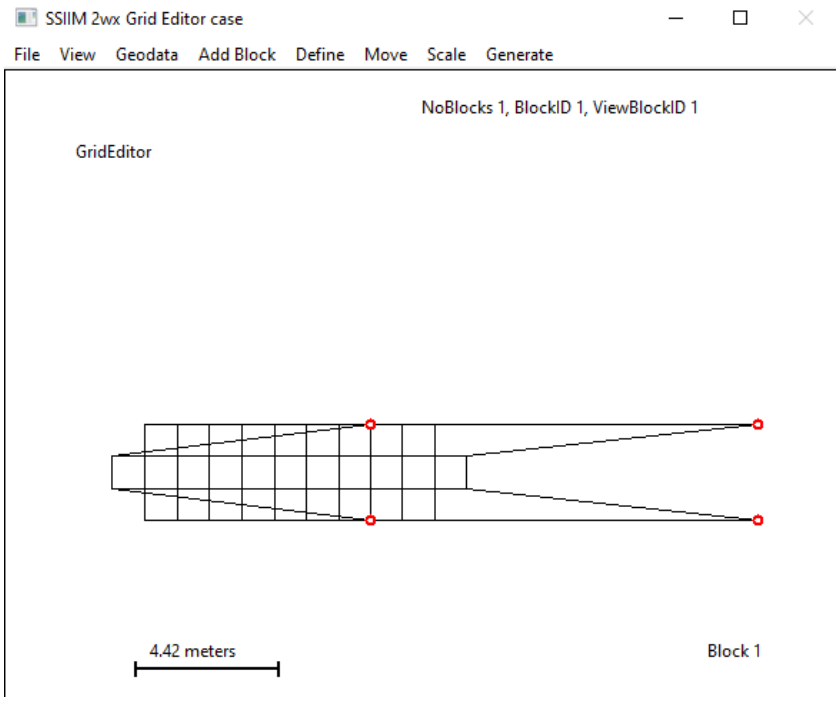



The location of the four corners of the initial grid can be given with mouse clicks. Before each corner is made, choose from the menu which corner to make. The four corners indicate a channel or a river, with an upstream left corner, a downstream left corner, an upstream right corner and a downstream right corner. The left/right side is when looking in the downstream direction. Alternatively, it is possible to give the exact coordinates of the corners in a dialog box. This is what we will do here.

The procedure is similar to what was done in Tutorial 1. Klick on the menu option *Define->coordiantes*, and specity the following four points:

(1,1) at x = 9 m, y=1 m
(1,4) at x = 9 m, y = 5 m
(12,1) at x = 21 m, y = 1 m
(12,4) at x = 21 m, y = 6 m

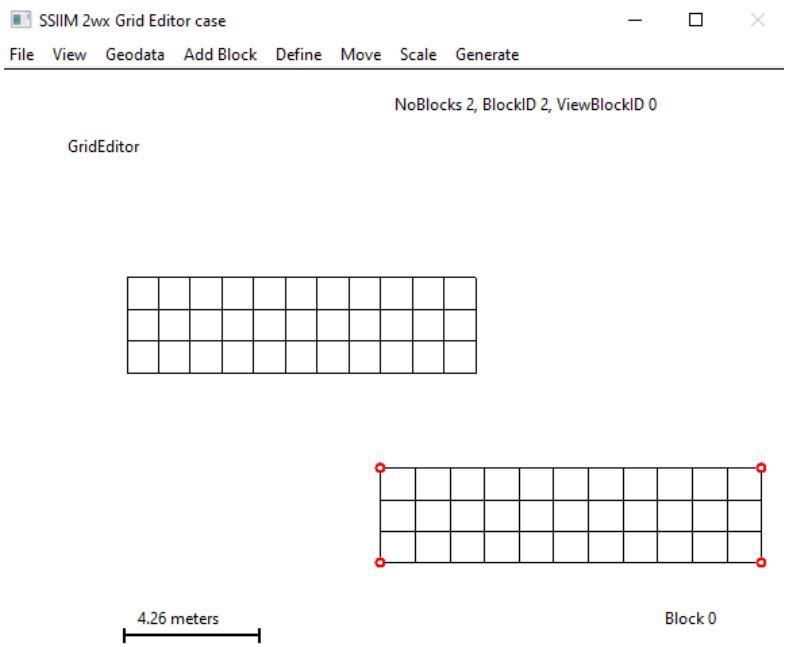
The GridEditor window will look similar to the figure on the right.

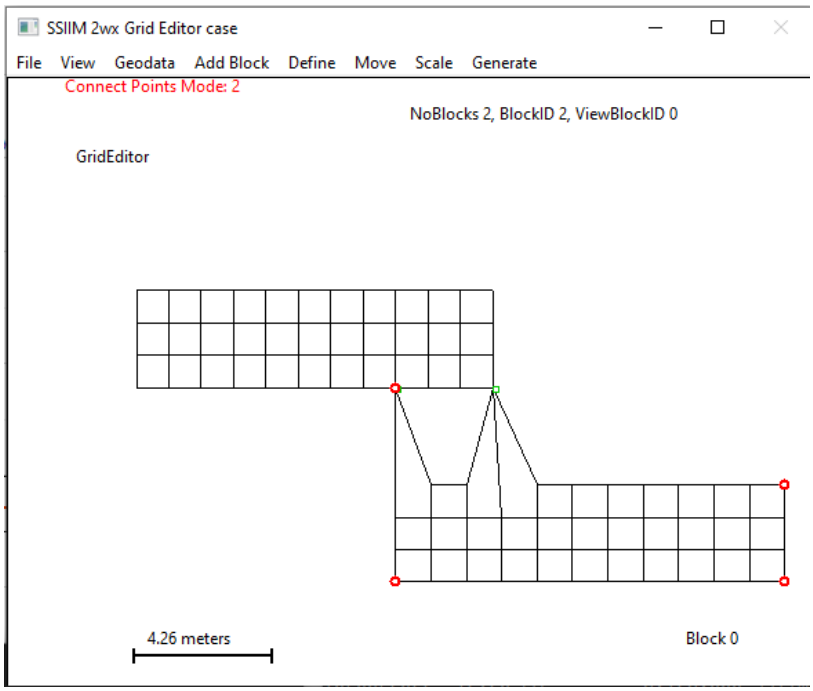



Then choose from the menu *Generate->boundary, Generate->TransfiniteI, Generate->3D grid.* The

grid is now moved a bit up to the right. This was the first block.

The second block is made with the same procedure. We use the same grid size in the dialog box. However, we don't change the coordiantes. After the block is made, choose from the menu: *View->all blocks*. Then both blocks are shown.



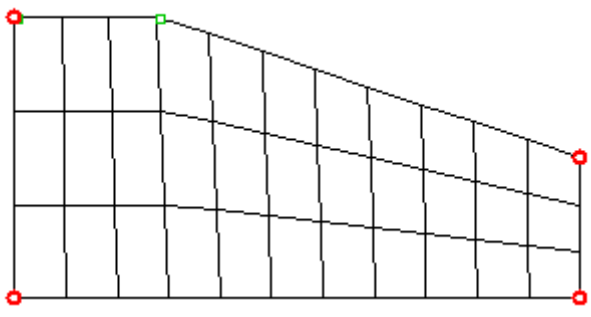




The next thing to do is to connect the blocks. This is done by connecting two grid intersections in one block with two intersections in the other block. In the menu choose *Define->Connect points mode*. Then click with the left mouse button on one of the two grid intersections. The one that will stay fixed. Afterwards, click with the right mouse button on the corresponding grid intersection in the other grid. Do the same thing with the second connections. A figure is shown to the left.

If a mistake is made, choose from the menu *Define->Delete last connection.*

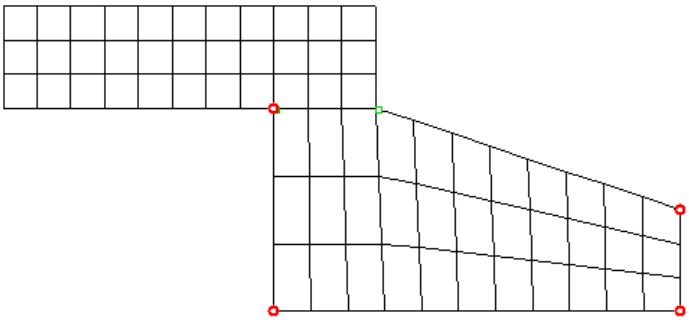
Then choose *Define->Connect points mode* in the menu again, to get back to normal editing mode.

The lower block now needs to be edited. Choose from the menu: *View->Block1*. Then *Generate->Sides, Generate->TransfiniteI, Generate->3D*



*Grid.* The block will look like the figure to the right.

Finally, choose f rom the menu: *View->All blocks*. Then the grid will look similar to the figure below.

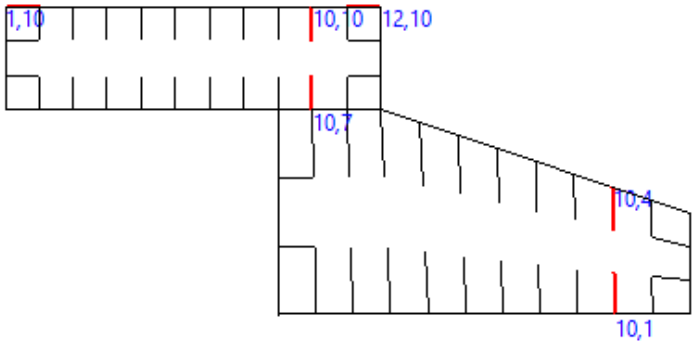


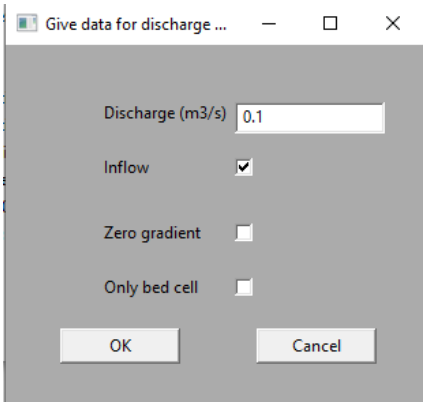
Note that there must be perfect correspondance between the grid intersections between the blocks. If not, the blocks are not connected. The connection algorithm is invoked by the *generate-3D Grid* option.

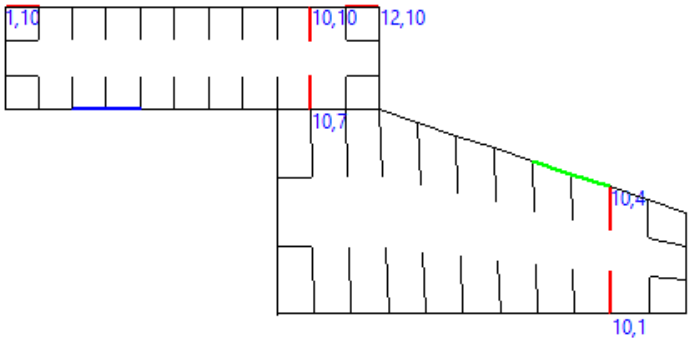
In the main (first) window, choose from the menu: *File-Save unstruc*. Then the grid is saved. End the program afterwards.

### 6.2.2 Specifying inflow and outflow

Start the program and read the previously generated *unstruc* file. Then choose *View -> DischargeEditor*. The grid seen from above emerges. The outlide of the grid is seen and some lines normal to the outline, indicating the location of the grid line intersections. Choose *Define->Set Discharge Values*. A dialog box emerges, where the water discharge is specified in m3/s. Give a value of 0.1 for the inflow discharge.




Click on *OK*. Then choose from the menu *Define->Give Discharge Boundary 1 cel*l. Then click on two of the sides of the cells bordering the grid.

Once clicked, the lines change colour to blue. Then choose *Define->SetDischargeValuesno. -> 2.* In the dialog box, click on the *Inflow* button. This specifies an outflow of the geometry. Give a value of 0.1 for the discharge.Then choose *OK*. Click on two grid lines bordering the geometry. The lines change colour to green, indicating outflow. Go back to the main (first) window, and choose from the menu *File -> Save unstruc.* The discharge information is included in the *unstruc* file.

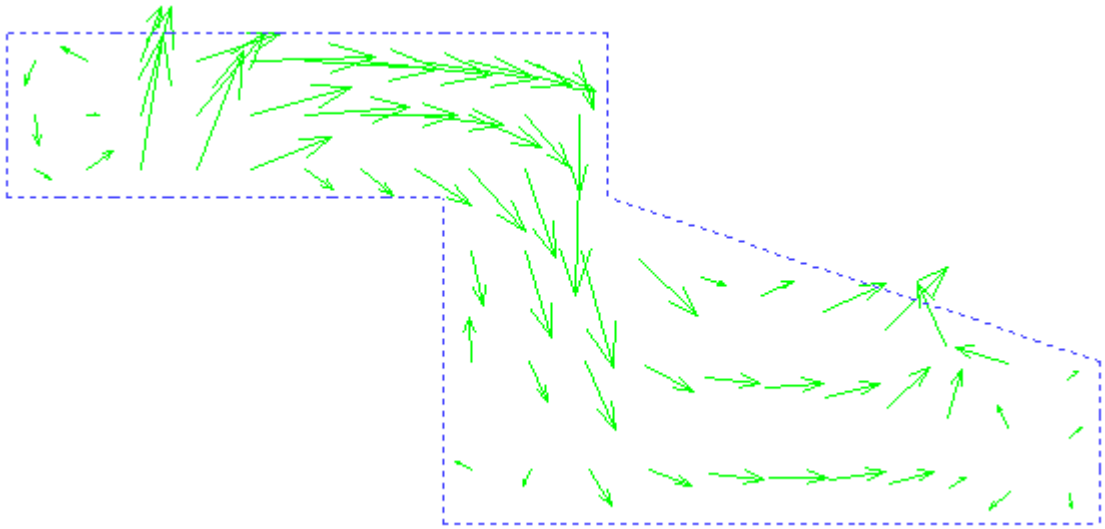
It is possible to give up to 10 groups of water inflows and outflows. But the sum of the inflow discharges has to be equal to the sum of the discharges going out.

### 6.2.3 Calculating the water flow

Start the program again and read the unstruc file. From the menu, choose *Commands->Water flow computations*. Since the grid is extremely coarse, the solution of the Navier-Stokes equaitons converges almost immediately.

### 6.2.4 Viewing the results

From the main window, choose the menu option *Views -> Plan view vector/grid*. In the window that opens, choose from the menu *Grid -> all blocks*. Then choose *Vector -> Show*. Then you see the velocity vectors, like in the figure below.



You can scale and move the plot by using the <*Page up*>, <*Page down*> and arrow keys on the keyboard. You can also scale the velocity vectors by choosing *Vector->Enlarge* from the menu.

Unfortunately, much of the other graphics in SSIIM 2 is made for one block only, meaning it is only possible to see one of the blocks at the time. The ParaView vtk files will also show the first block.

## 6.3 Tutorial 3. Channel contraction

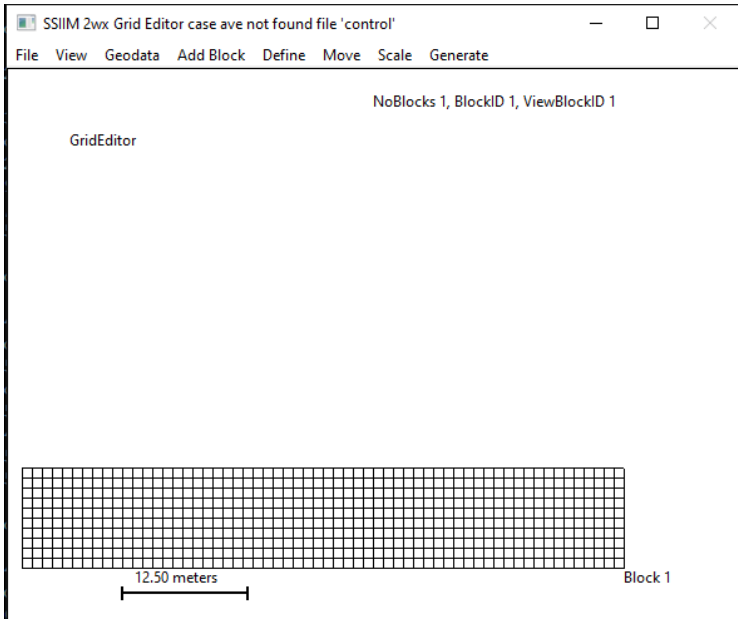
This tutorial is meant for first time users of the SSIIM program. The tutorial shows the main features of the user interface, with the presentation graphics, animation and grid editing. The user is not required to edit files, but some knowledge of grids is recommended. It is important to know the purpose of the *control* and *koordina* input files. During the tutorial the files will be made interactively. The tutorial does not show the more advanced features of the program, which necessitates editing of the input files.

### 6.3.1 Making input files

In this section the two input files *control* and *unstruc* are made.

Start SSIIM 2 from a directory where the *control* file do not exist. This is done in a command prompt by writing c:\path\ssiim2wx538 <CR> when you are on this directory. The "path" is the path to where the executable program ssiim2wx538.exe is placed. Or you can copy the ssiim2wx538.exe file to this directory and double-click on it in the graphical file manager.

The main window shows up. On the menu, choose *View->GridEditor*. The *GridEditor* window shows up. In this window, choose *Add Block->New - size.* A dialog box shows up. Fill inn 61 crosss-sections and 11 points in the cross-section. Clik *OK*. The outline of a channel shows up on the screen. The grid can be seen by choosing *View->Grid cells* from the menu. The *GridEditor* window will then look like the figure below. On the menu, choose *Generate->3D grid*. Go back to the first window, and choose *File-Save unstruc*. This saves the grid to the *unstruc* file.




The next step is to specify the discharge in the channel. This can be done using the *control* file. A control file has been made in the working directory. Edit this with the notepad. In the *Windows File Manager*, click on the control file with the right mouse button, and open the file with the *Notepad* program. Remove the text "no values given", and add the following lines:

*F 314 1 1*
*F 237 1 0.1*
*F 237 2 0.1*

Then save the file. The *F 314* data set now tells the program that the inflow is on the left side of the grid and the outflow is on the right side. The left side discharge group is numbered 1. The first *F 237* data set specifies the discharge for group 1 to be 0.1 $m^3/s$. The second *F 237* data st specifies the discharge for group 2 to be

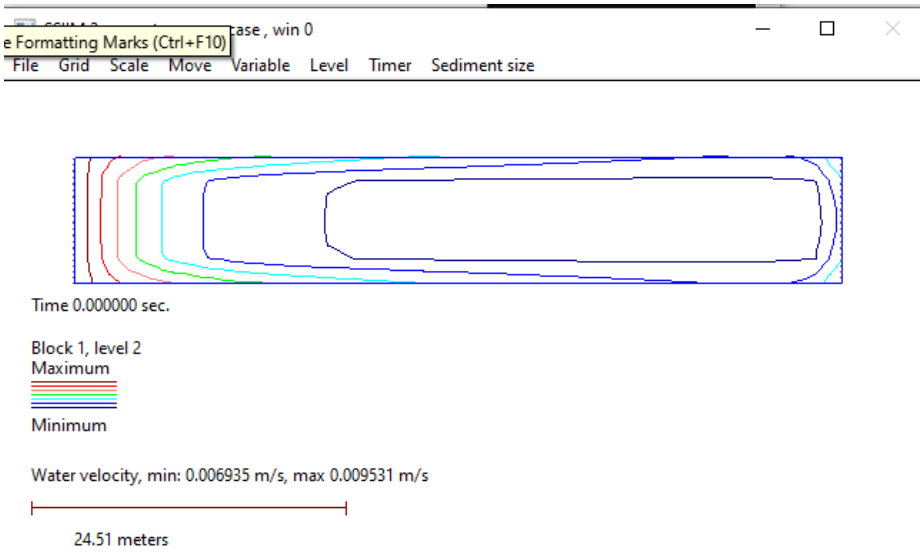
0.1 $m^3/s$. Discharge group 2 is on the right side of the grid.

An alternative to using these data sets in the *control* file is to specify the discharges in the *DischargeEditor*.

### 6.3.2 Computing the flow field and looking at the results

Stop the program and restart it again. The program will only read the *control* file at startup, so if something is modified in the control file, the program has to be restarted. On the menu, first choose *File-Read unstruc*. Then choose *Commands-Water flow computations*. This starts the solution of the Navier-Stokes equations. Choose *Update window* on the menu, or push the "*u*" key of the keyboard multiple times. The residuals are written to the main window. The solution is converged when all the residuals are below 0.001.

In the main window, choose *View-Plan view contours*. This shows a contour plot of the variables in the grid. However, it is first necessary to select which variable to view. On the menu, choose *Variable->Horizontal velocity*. The window will look similar to the figure on the right, when scaled and moved with the arrow keys on the keyboard, and the <PageUp> and <PageDown> keys.

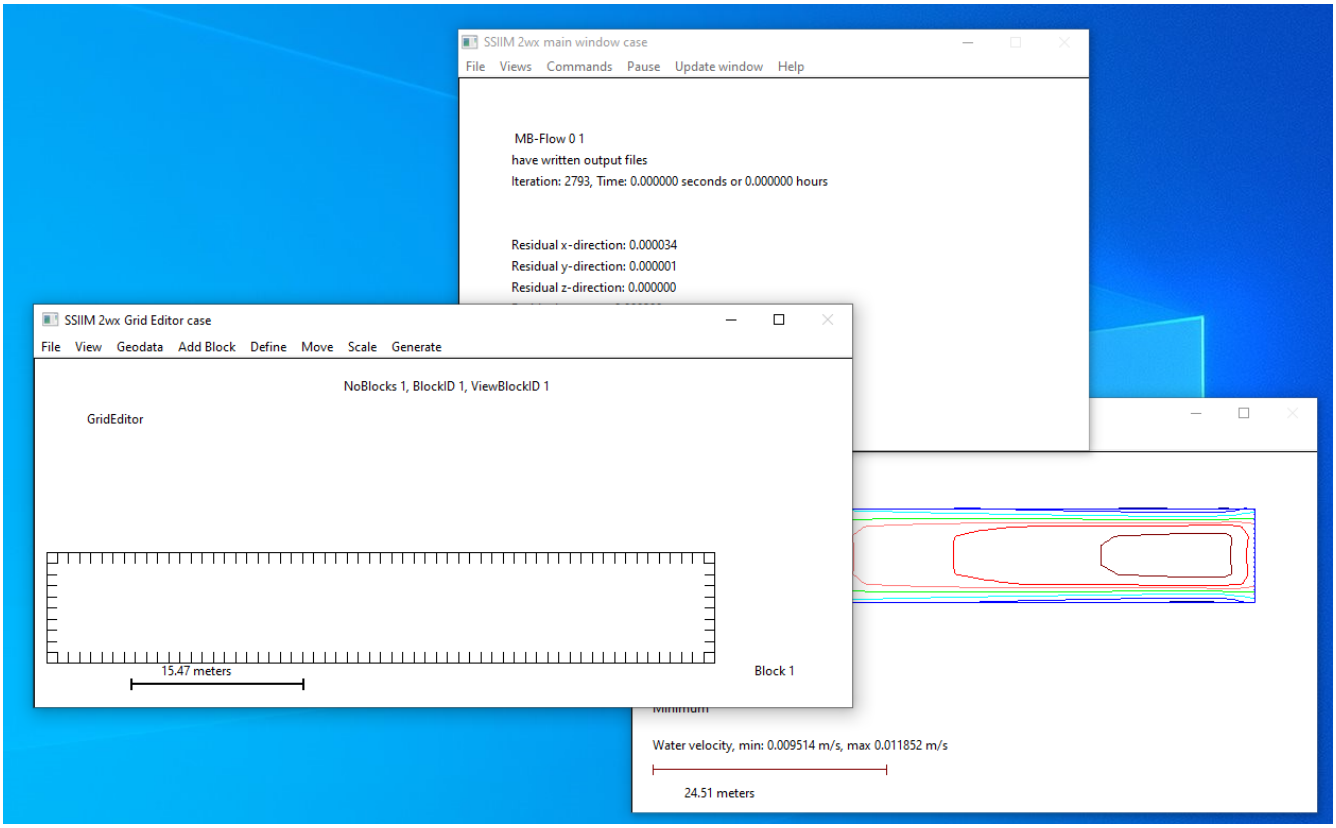



The figure shows the velocity at the bed. The inflow section has a uniform velocity profile, where the velocity at the bed is the same as at the water surface. When the velocity profile develops into a more logarithmic shape, the velocity will be lower at the bed than at the surface. Therefore, the bed velocity will be reduced in the streamwise direction. As shown on the figure. Choose *Level->Surface* on the menu to see the similar figure for the velocities close to the water surface.

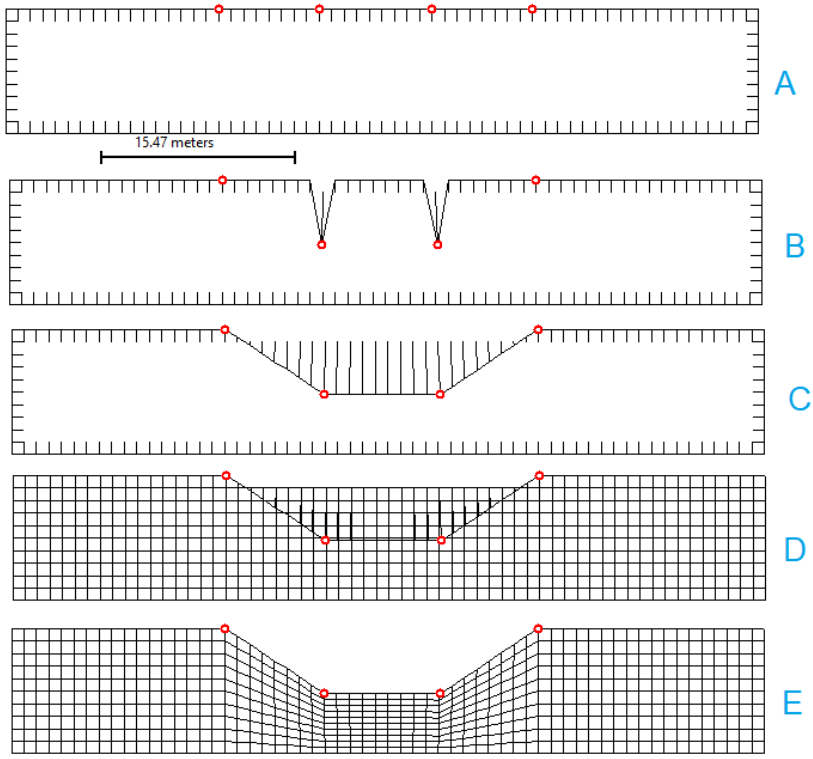
The velocity field is not too exciting for this channel. To make a more complex flow pattern, you need to change the geometry. This is done next.

### 6.3.3 Changing the grid

In this section we will focus on the *GridEditor*. Start the *GridEditor* by choosing *Views* from the main menu bar and *GridEditor* from the pull-down menu. Scale and move the window and the grid using the keyboard arrows and <PageUp> and <PageDown> keys. You may have multiple SSIIM 2 windows open at the same time, as shown in the figure below.




To edit the grid, you first choose some points which will not be affected by the interpolation routines. This is done in a special mode of the editor. This mode is invoked by choosing *Define->Set Fixedpoints all.* To verify that this mode is chosen, the letters "*Add Fixedpoints all 0*" is shown on the lower part of the window. The integer shows how many points you have chosen. We want to choose four points, all on the upper boundary of the grid. A point is chosen by clicking with the mouse on a grid intersection. If successful, a red circle emerges at the grid intersection. This is shown in the figure to the right, Fig. A. After choosing the points we return to normal mode. This is done by choosing *Define->Set Fixedpoints all* again. We observe that this mode

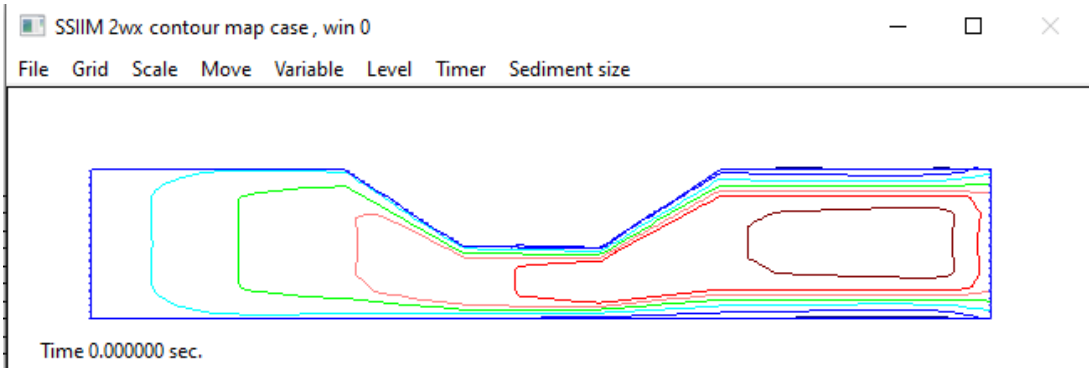


disappears because the text "*Point mode, 4*" disappears.

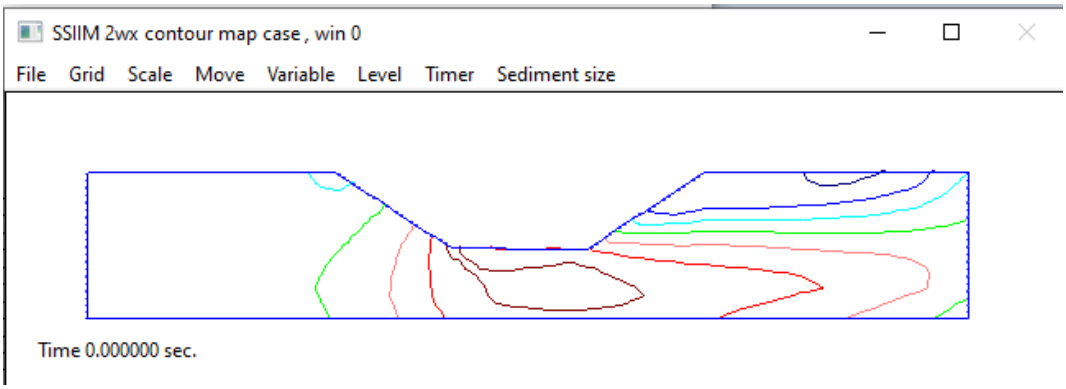
We want to model a channel contraction, and this is done by moving the two middle fixedpoints towards the channel center. Click on the points with the left mouse button and then click on the new location with the right mouse button. The result will look something like Fig. B above. On the menu, click Generate->Sides. This will create straight lines between the fixedpoints, and the geometry will look like Fig. C above.

What has been done up to now is to move the boundary of the grid. To see the whole grid, choose View->View grid cells on the menu. You will see something like Fig. D above. The internal grid points have not been moved. To fix this, choose Generate->TransfininteI on the menu. The grid will now look ok, like Fig. E above. Then choose Generate->3D grid on the menu. This will transfer the grid information to the main program.

You can now go back to the Contour Map window. Click Timer->Update window. You will see something like the figure below.




This is not the correct flow field for this geometry. To get the correct flow field, we need to recompute it by solving the Navier-Stokes equations. This is done by going back to the first window, and on the menu select Commands-Water flow computations. Push the "u" key on the keyboard until the solution is converged. Then go to the contour map window and look a the velocity field.




It is also possible to see the velocity vectors, by going back to the main window and choosing *Views-Plan view vector/grid*. When the window opens, choose on the menu *Vector->Show*.

## 6.4 Tutorial 4. Flow in a bend

The purpose of this tutorial is to show how a spreadsheet can be used to generate a simple the geometry. In this case, a bend. The coordinates of the sides of the bend are made using the spreadsheet, and are stored in a file called *koosurf*. Also, the points on the sides are fixed defining them as *Fixedpoints*.

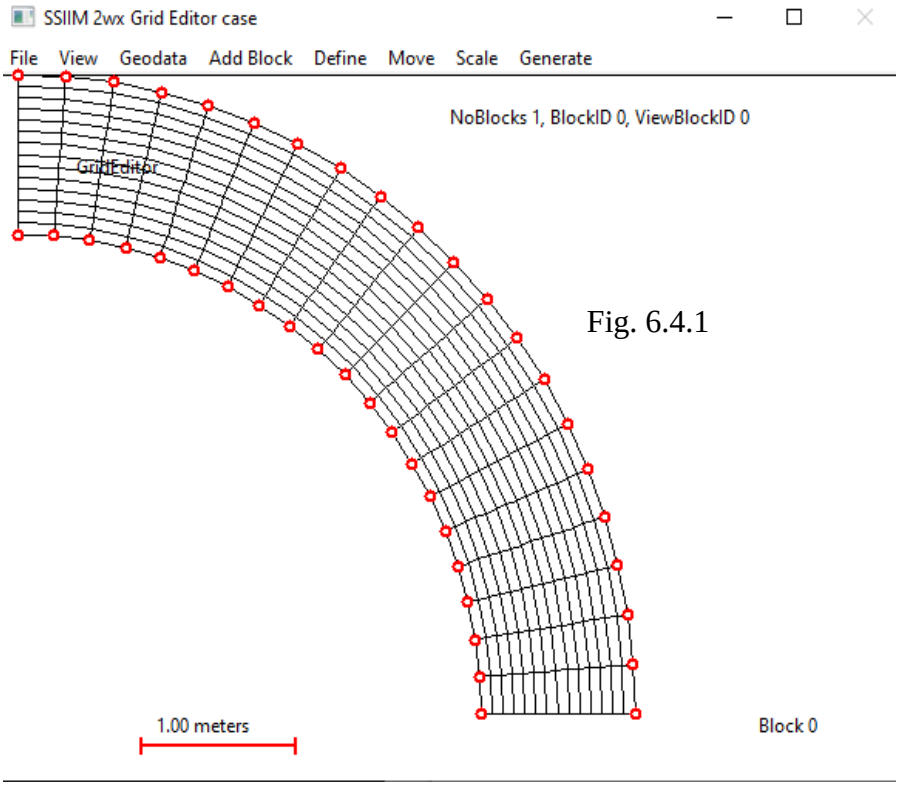



Fig. 6.4.1

In the tutorial, it is necessary to edit files. It is also required that the user knows how to use a spreadsheet.

First, we decide for the dimensions of the channel and the grid size. We want to compute a 90 degree bend, with 21 grid lines in the streamwise direction and 15 grid lines in the lateral direction. The inner radius of the bend is 3 meter and the outer radius is 4 meters, giving a channel width of 1 meter.

### 6.4.1. Making the *koosurf* file.

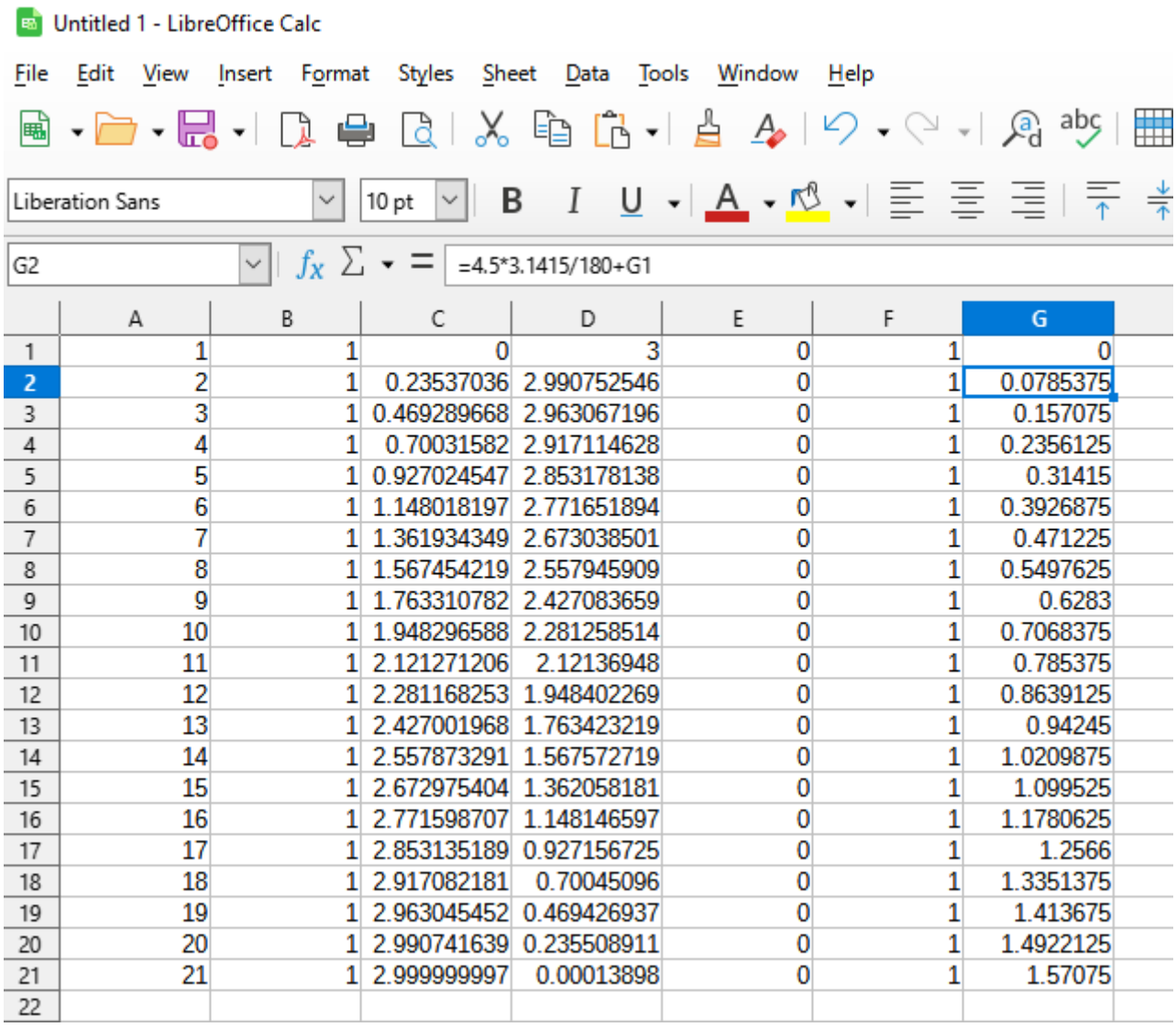
With 21 grid lines over 90 degrees, each cell will have a sector of 90 / 20 = 4.5 degrees. The x coordinates of the grid line on the inner bank can then be computed by the following formula:

$$x = 3\sin(\alpha) \qquad (6.4.1)$$

$$x = 3\cos(\alpha) \qquad (6.4.2)$$

These formulas are then coded in the spreadsheet, where angle for each grid line is also given. The formula for the angle will start with zero for the first point, and then it will be the 4.5 degrees plus the value in the cell above.

The figure below shows how the spreadsheet looks like.




| | A | B | C | D | E | F | G |
|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 0 | 3 | 0 | 1 | 0 |
| 2 | 2 | 1 | 0.23537036 | 2.990752546 | 0 | 1 | 0.0785375 |
| 3 | 3 | 1 | 0.469289668 | 2.963067196 | 0 | 1 | 0.157075 |
| 4 | 4 | 1 | 0.70031582 | 2.917114628 | 0 | 1 | 0.2356125 |
| 5 | 5 | 1 | 0.927024547 | 2.853178138 | 0 | 1 | 0.31415 |
| 6 | 6 | 1 | 1.148018197 | 2.771651894 | 0 | 1 | 0.3926875 |
| 7 | 7 | 1 | 1.361934349 | 2.673038501 | 0 | 1 | 0.471225 |
| 8 | 8 | 1 | 1.567454219 | 2.557945909 | 0 | 1 | 0.5497625 |
| 9 | 9 | 1 | 1.763310782 | 2.427083659 | 0 | 1 | 0.6283 |
| 10 | 10 | 1 | 1.948296588 | 2.281258514 | 0 | 1 | 0.7068375 |
| 11 | 11 | 1 | 2.121271206 | 2.12136948 | 0 | 1 | 0.785375 |
| 12 | 12 | 1 | 2.281168253 | 1.948402269 | 0 | 1 | 0.8639125 |
| 13 | 13 | 1 | 2.427001968 | 1.763423219 | 0 | 1 | 0.94245 |
| 14 | 14 | 1 | 2.557873291 | 1.567572719 | 0 | 1 | 1.0209875 |
| 15 | 15 | 1 | 2.672975404 | 1.362058181 | 0 | 1 | 1.099525 |
| 16 | 16 | 1 | 2.771598707 | 1.148146597 | 0 | 1 | 1.1780625 |
| 17 | 17 | 1 | 2.853135189 | 0.927156725 | 0 | 1 | 1.2566 |
| 18 | 18 | 1 | 2.917082181 | 0.70045096 | 0 | 1 | 1.3351375 |
| 19 | 19 | 1 | 2.963045452 | 0.469426937 | 0 | 1 | 1.413675 |
| 20 | 20 | 1 | 2.990741639 | 0.235508911 | 0 | 1 | 1.4922125 |
| 21 | 21 | 1 | 2.999999997 | 0.00013898 | 0 | 1 | 1.57075 |
| 22 | | | | | | | |

The spreadsheet also needs to compute the similar coordinates for the outside bank of the channel. This will be the same as for the inside bank, except the radius will be 4 m instead of 3 m in Eqs. 6.4.1 and 6.4.2. And the *j* values will be 15 instead of 1.

After the spreadsheet is made, the content must be written to a file called *koosurf*. This can be done in many ways, but one way is to open the Notepad program in Windows. Mark the areas from the spreadsheet and copy it into the Notepad. Only the *i,j,x,y,zbed* and *zsurface* must be copied, not the angle. Also, there must be one set of coordinates on each line, with no blank lines in between. Make sure there are six numbers on each line.

In Notepad, the file can be saved as *koosurf*. However, Notepad will add an extension .txt to the file name. This must be removed.

Note that the indexing system for *i* and *j* is not arbitrary. The *i* indexes will typically be the number of a cross-section, and *j* will indicate each point in a cross-section. Typically, the *i* index will increase in the downstream direction, and the *j* index will increase from the right to the left bank, looking in the

streamwise direction. This is the default configuration of the grid. Given this indexing numbering, it is possible to have water flowing in/out of the sides. Also, it is possible to have water flowing into the last cross-section and out of the first. However, in most cases a channel or a river is modelled, where the first cross-section covers an inflow area and the last section has an outflow region. In such cases, there are some automatic definitions of inflow and outflow, which saves some time making the input files. So this is the recommended setup.

### 6.4.2. Making the *control* file

The next thing to do is to make SSIIM 2 see these points as fixed points. This is done by making several *W 6* data sets in the *control* file. This is also most easily done in a spreadsheet. The following text must be made and inserted into the *control* file. Note that the *control* file also must be without extension. And at the present stage, it does not need to contain any other information than the *W 6* data sets. The file will then look like the following:

```
W    6    1    1
W    6    2    1
W    6    3    1
W    6    4    1
W    6    5    1
W    6    6    1
...
W    6    21   1
W    6    1    15
W    6    2    15
W    6    3    15
W    6    4    15
W    6    5    15
W    6    6    15
..
W    6    21   15
```

The points between *i* index 7 and 20 are not given above, but they have to be in the *control* file. The *control* file will then have 42 *W 6* data sets.

### 6.4.3. Making the grid

Make a directory with the SSIIM 2 program and the *control* and *koosurf* file made earlier. Then start the SSIIM 2 program. From the main menu, choose *Views->GridEditor*. Then choose the menu option *Add Block>Block from koosurf*. The outside bend of the grid is then seen in plan view, together with some other lines, as shown in Fig. 6.4.3. The grid lines will look confusing, since we only have made the sides of the grid and not the grid interior. The rest of the grid is made by choosing from the menu: *Generate->sides, Generate->TransfiniteI, Generate->3D grid* and *View->View grid cells*. The grid then looks like what is shown in Fig. 6.4.1.

To save the grid, use the menu option in the main window: *File* and *Save unstruc*. After this, it is possible to end the program and restart it reading back the *unstruc* file from the *File* menu.

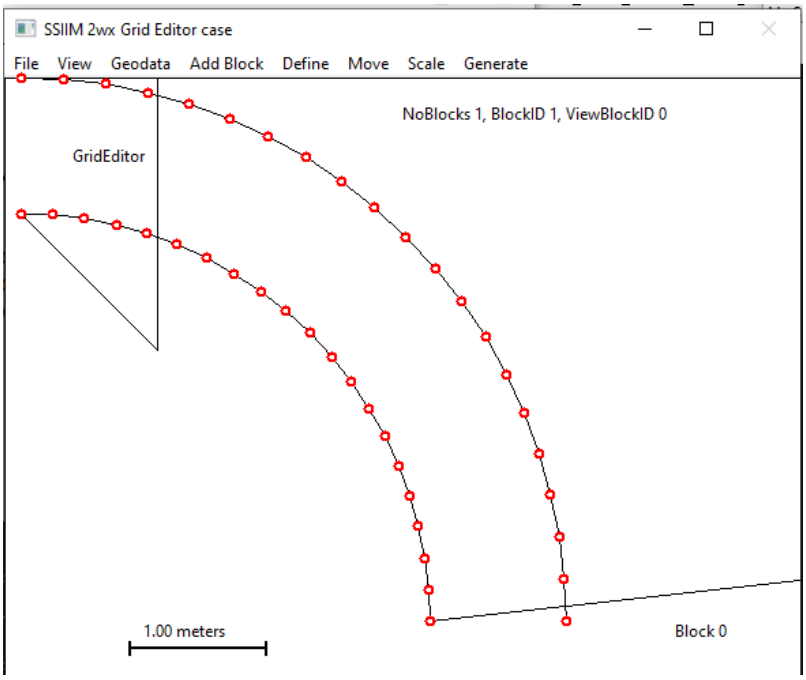



*Figure 6.4.3*

### 6.4.4. Specifying inflow and outflow discharges

There are two ways of specifying inflow and outflow discharges and their location: A simple method and the *DishcargeEditor*. The simple method is only possible if the water is flowing only into cross-section number 1 and out of the last cross-section. Then the discharge can be given in the *control* file, on the *F 237* data sets. One data set for the inflow and another for the outflow. Also, the *F 314 1 1* data set must be given. The latter option is suggested here. Add the following data sets to the control file:

*F 314 1 1*
*F 237 1 0.6*
*F 237 2 0.6*

Then save the file.

### 6.4.5 Computing the water velocities

Restart the program and read the *unstruc* file from the *File* menu. Then on the Commands menu, choose Water flow computations. Press the "*u*" key on the keyboard until you see that the residuals have gone down sufficiently and the solution is converged.

### 6.4.6 Looking at the results

From the main menu, choose *Views* and *Plan view vector/grid.* Then choose *Vector->show.* Scale the vectors with the "*e*" and "*s*" keys on the keyboard until they are of appropriate length. Also, you can scale or move the view with the arrow keys and the keyboard keys <*PgUp*> and <*PgDn*>. Choosing on the menu *Level->Bed* and *Level->Surface*, you can see how the direction of the vectors at the outlet changes due to the secondary currents.

## 6.5 Tutorial 5. Natural river

The purpose of this tutorial is to show how a grid can be made for a natural river, where the input is a scanned geometry with a cloud of (x,y,z) points. It is required that the user knows how to edit files.

SSIIM 2 includes wetting and drying, and it is convenient to use this algorithm to get a grid that follows the bank of the river as the water level rises and sinks.

### 6.5.1 Making the grid

The geometry of a natural river is most often fairly complex. The most used way to describe the geometry is by measuring a large number of coordinates (x,y,z) values of points at the river bed. SSIIM use these points in a file called *geodata*. Each point is given on one line in the file, with the three floating point numbers, x,y and z. There has to be a capital E at the start of each line, before the numbers, and the numbers must be separated by one or more spaces.

In the present tutorial, we will use a geodata file made up from a virtual case. The file can be downloaded from the web page: *http://folk.ntnu.no/nilsol/cases/tutorial_river/geodata*.

Make a new directory on your PC, download this *geodata* file to the directory. Then start the program. In the main menu, go to *View* and *GridEditor*. Then go to *View* and *geodata points*. The measured points now emerge in the window as seen in Fig. 6.5.1.

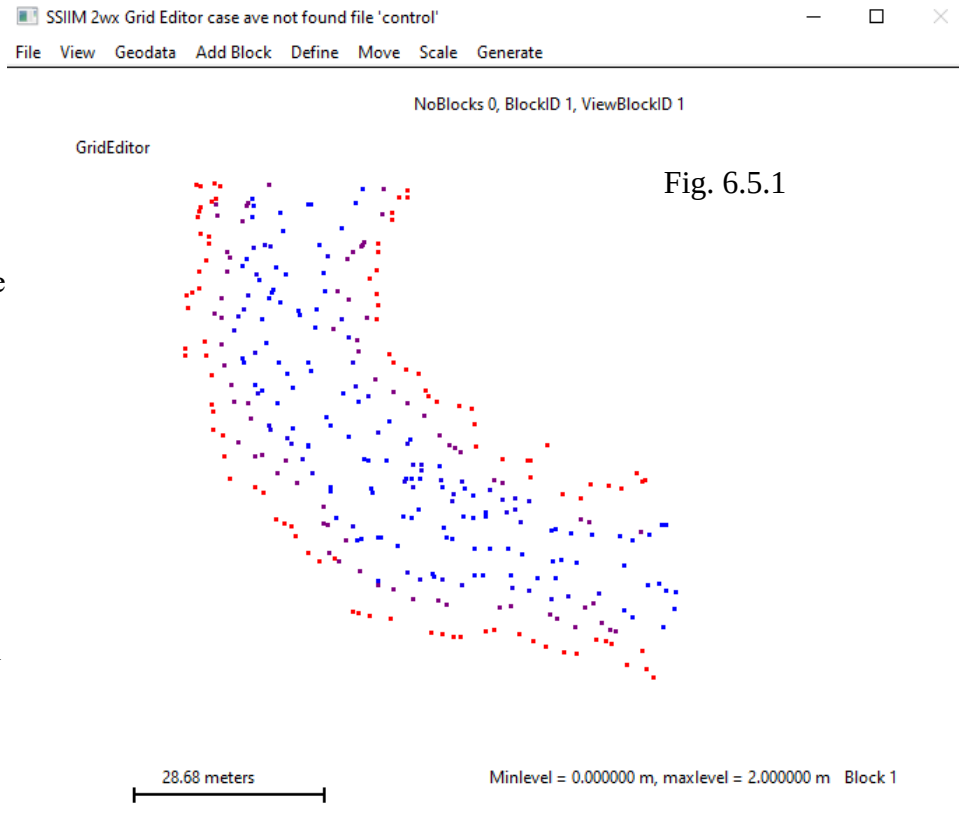



Fig. 6.5.1

The colours of the geodata points shows the water depth. Red is high water depths and blue is low water depths. Fig. 6.5.1 shows that the river is flowing in a bend. We now assume the inflow is at the the top of the figure and the outflow is at the right side.

The first thing we need to do is to make a grid block covering the river. This is done from the menu, by choosing *Add Block* and *New - size*. Choose 71 cross-sections and 21 points in each cross-section. An outline of the initial block is seen below the points. This needs to be fitted to the geometry.

We start with moving the four corners of the block. Click with the left mouse button on one corner and then click with the right mouse button on the location where you want the point to be moved. In the initial grid, the water is flowing in from the left and out on the right side. This should be kept in mind when moving the corner points. After the corner movemetns, the GridEditor window may look something like Fig. 6.5.2.

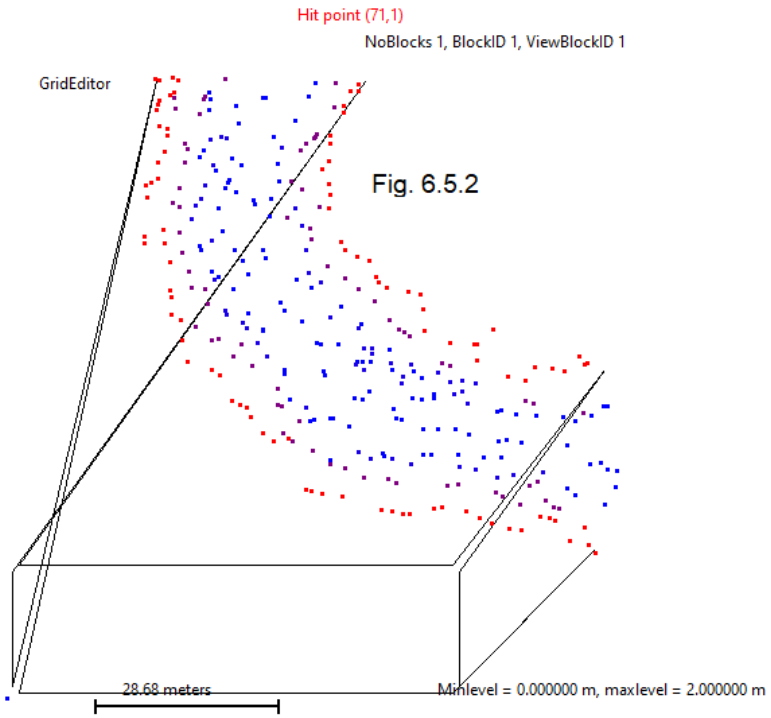


Fig. 6.5.2

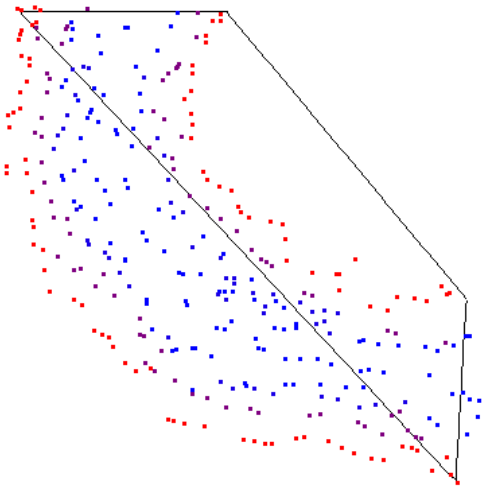


Fig. 6.5.3

Then on the menu, choose *Generate->Sides* and *Generate->TransfiniteI*. The outline will look like Fig. 6.5.3.

We see that the inflow and outflow boundary looks ok. But the sides needs to be moved according to the bend geometry.

The adaptions of the sides is done by adding fixedpoints. On the menu, choose *Define->Set fixedpoints all*. Then click on some points along the side bounday line. When you hit a grid line intersection, a red circle will emerge. Make a number of these fixedpoints on each side.

Then again, choose from the menu *Define-Set fixedpoints all*. It is now possible to move the fixedpoints with the mouse. Click on one fixedpoint with the left mouse button, and click on the location where you want it with the right mouse button. Like shown in Fig. 6.5.4. Do this with all the fixedpoints. The result will look something like what is shown in Fig. 6.5.5.

Then on the menu, choose *Generate->Sides*. The grid changes to what is seen in FIg. 6.5.6. Choose on the menu *Generate -> TransfiniteI* and *Generate -> 3D grid*. Then choose on the menu *View->View grid cells*. The grid is shown, as given in Fig. 6.5.7.

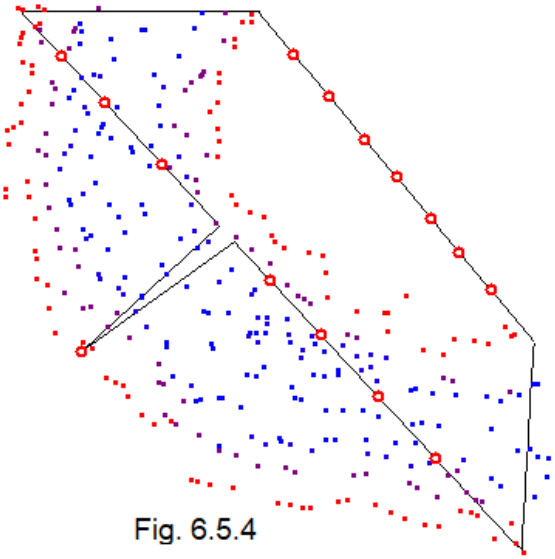


Fig. 6.5.4

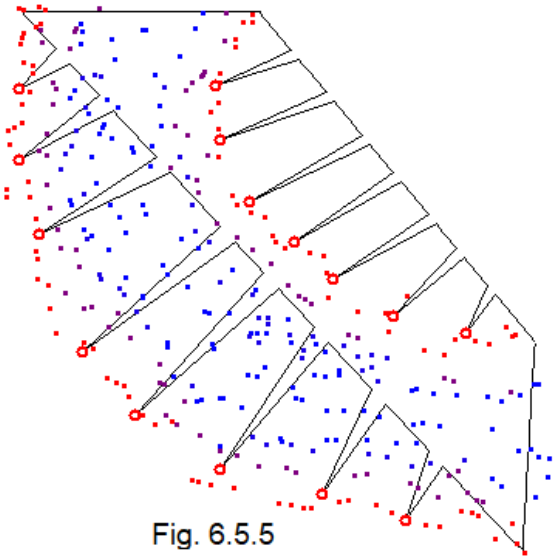


Fig. 6.5.5

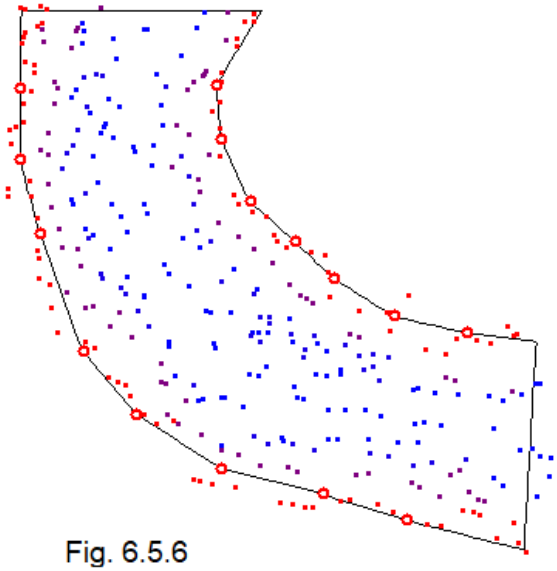


Fig. 6.5.6

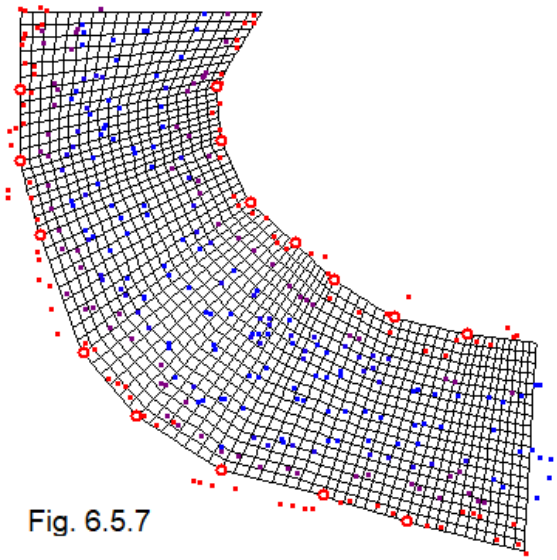


Fig. 6.5.7

The procedure so far has given the horizontal layout of the grid. The vertical extension of the grid also needs to be decided. This is done by using the *geodata* points to determine the bed level of the grid by interpolation. The menu option is *Generate* and *bed interpolation*. After this, use the menu again with *Generate* and *3D Grid*. The grid should then be saved to the *unstruc* file again.

From the main menu, choose *Views->Plan view contour*s. In the new window that opens, choose from the menu *Variable->Depth*. A contour map of the interpolated depth is seen, as given in Fig. 6.5.8.

The grid we have made may not be optimal. There may be holes or hills where the bed interpolation algorithm has not done a good job. Then it is possible to go back to the *GridEditor* and give in the vertical level on individual grid intersections. Click on the grid intersection and from the menu choose *Define* and *Give coordinates*. Then give a better *z* value in the dialog box. Repeat this for all problem areas and then from the menu choose *Generate* and *3D Grid*, and then save the *unstruc* file. Do not generate new bed levels, as the algorithm with overwrite any manual changes to the coordinates.

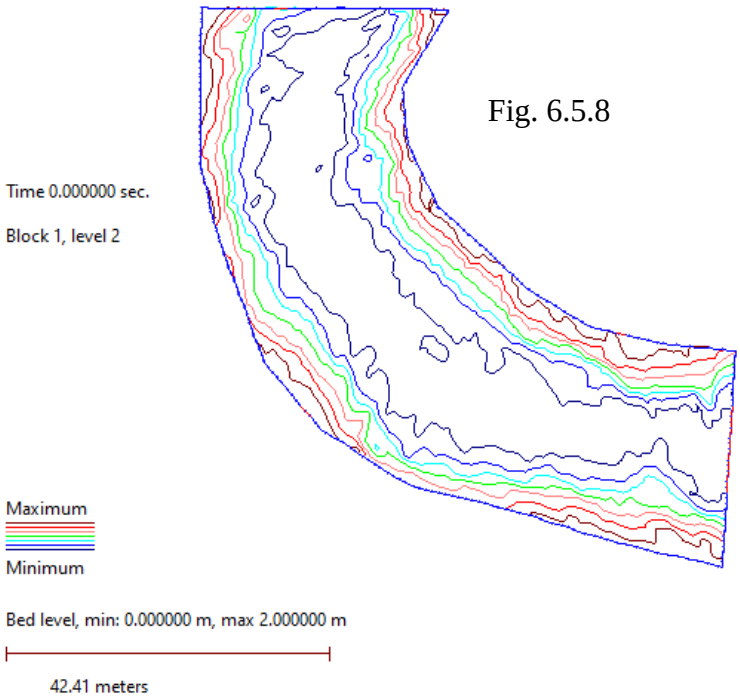



Fig. 6.5.8

Note that it is also possible to improve the grid by adding or removing *geodata* points in the *GridEditor*. It is possible to add *geodata* points on a line by holding down the left mouse button and move the mouse pointer at the same time. It is only possible to add 5000 new *geodata* points at one time. If more points are added, the program may stop. If you need more points, you can save the *geodata* file, stop the program and restart. Then you may add another 5000 points. This can be repeated as many times as needed.

### 6.5.2 Specifying a water discharge

The water discharge can be specified using the *Discharge Editor*, as described in the earlier tutorials. However, for a river where there is only one block and the water is flowing into one of the four sides and out of the opposite side, like we have in the current case, there is another option. This is to specify the following data sets in the *control* file:

*F 314 1 1*
*F 237 1 1.5*
*F 237 2 1.5*

The *F 337* data sets specify the water discharge in $m^3/s$, while the *F 314* data set sets the grid ends to inflow and outflow.

If we open SSIIM 2 again, read the *unstruc* file and look in the *DischargeEditor*, we can see the coloured lines from the inflow and outflow. If we look in the dialog box, we see that the water discharge is the same as specified in the *control* file.

### 6.5.3 Computing the water flow and looking at the results

Start SSIIM 2, read the *unstruc* file and from the menu choose *Compute* and *Waterflow*
Push the *u* button on the keyboard and watch the residuals decrease. When they are under 0.001, the solution is converged.

Modelling a natural river, the grid may not be smooth and there may be many cells that are not hexahedral. This will make it more difficult to reach a converged solution for the Navier-Stokes equaitons. Chapter 4.2 gives more information about possible measures to improve the convergence. Some recommendations:

1. Lower the relaxation factors on the *K 3* data set in the control file.
2. User a multi-grid solver (*K 5 + F 168*)
3. Introduce a time step (*F 33*)

An example *control* file where the solution converged for this case (after Chapter 6.5.4):

*F 1 D debug information to boogie file*
*F 2 UW read unstruc file and compute water velocities*
*F 33 1.0 2 time step and inner iterations*
*F 102 1 smoother channel sides*
*F 112 1 reconstruct grid at startup*
*F 168 8 mg solver*
*F 314 1 1 inflow and outflow of water*
*F 237 1 5.0 discharge in*
*F 237 2 5.0 discharge out*
*K 3 0.4 0.4 0.4 0.05 0.1 0.1 relaxation coefficients*
*K 5 0 0 0 10 0 0 mg solverfor pressure*

To see the results, choose *Views* and *Plan view vector/grid* from the menu. In the window that opens, choose from the menu *Vector->Show*. Scale the vectors with the *e* or *s* key on the keyboard until they have the right size.

### 6.5.4. Making a lower initial water surface

The top of the grid made in Chapter 6.5.1 is horizontal and at the level of the highest point in the *geodata* file. This is at 3 meters with the current data. Often the geometrical points measured in the river include the overbanks. This makes sense when modeling a flood. However, if we want to model a water discharge smaller than a flood, we also want a lower water level. This can be done in two ways. If the intial water surface is horizontal, then the water level can be given in the dialog box by choosing on the menu *Define->New water level*. Afterwards, choose *Generate->3D grid*. Go to back to the main window and choose *Views->Plan view vectors/grid* to see the new grid, or look at the new water depths in the contour map window.

If the new water level is sloping, the *koosurf* file can be edited in a spreadsheet, and a more complex initial water surface can be given. Then this *koosurf* file can be renamed *koordina* and placed in the working directory. Add *F 2 U* and *F 112 1* to the *control* file. When the program starts, it will first read

the *unstruc* file. Then it will read the *koordina* file, and then regenerate the grid with the new water level from the *koordina* file.

Looking at the grid, the edges following the banks may not look too smooth. To improve the river bank cells, the *control* file needs to be modified. Open the file with an editor and add the data set *F 102 1*. Save the file, restart SSIIM 2, go to the *GridEditor* and choose from the menu *Generate* and *3D Grid*. Go back to the *Map* view and see how the edges are smoother, similar to Fig. 6.5.9. Then save the *unstruc* file.

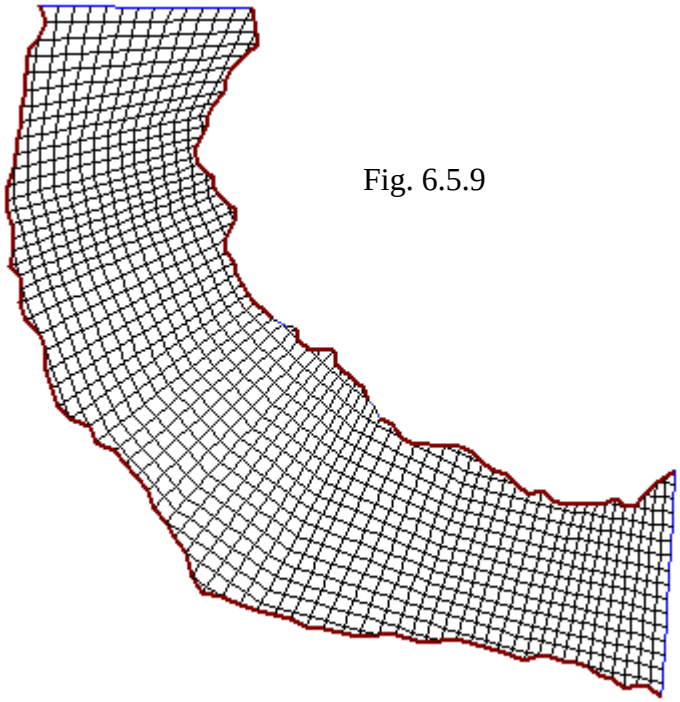


Fig. 6.5.9

### 6.5.5 Computing a more complex water level

Although the chosen flat level of the water may correspond reasonably well to the real world, it may sometimes be necessary to make a more accurate model. There are several algorithms in SSIIM 2 for computing the water level. The most relevant ones for river modeling assume subcritical flow, where the downstream water level is specified. The user then has to specify a cell with a given water level, which is unaffected by the water surface computation in the other cells. The reference cell must be specified on the *G 6* data set in the *control* file. The next step therefore involves which cell to choose and how to choose it.

Remove the *F 112* data set from the *control* file, restart the program and read the *unstruc* file. On the menu, choose *Views* and *Plan view->vector/grid*. Scale and move the grid so that a close-up of the middle of the outflow section is seen. Then choose *Grid->show* and *Numbers->cell.* Then choose *Level->surface.* You will see something like Fig. 6.5.10 to the right.

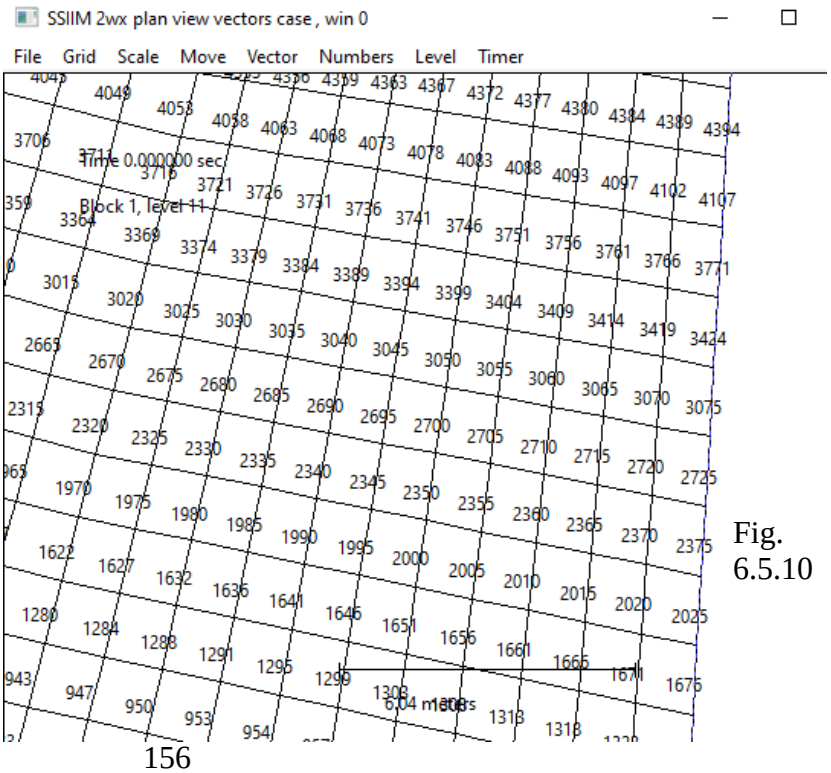



Fig. 6.5.10

If you have a very fine grid, it is improtant to do the scaling/moving first, and then choose to show the grid cells and the cell number afterwards. Otherwise the graphics will be very slow.

Then choose the numbers in one of the cells bordering the outflow boundary. Typically in the middle of the outflow region. For example 3075 in Fig. 6.5.10. Then give the following data set in the *control* file: *G 6 3075 0 0 0.01 0.1*

It is also necessary to specify a time step and which algorithm is to be used for the free water surface computation. To the *control* file, add

*F 33 2.0 20*
*F 36 2*

Then restart the program, read the *unstruc* file and start the water flow computations. In the *Views->Plan view contours* graphics, look at how the water level changes (Fig. 6.5.11). Note that the water level will only be updated if the residuals are fairly low. Lowering the time step or increasing the number of inner iterations on the *F 33* data set may be necessary to reduce the residuals. Also, the grid will change every time the water level changes, which means that the computational time will be much longer and convergence will be slower.

If you look at the pressure field, chosen from the *Variable* menu, then you will see the correspondence between the water surface elevation and the pressure.

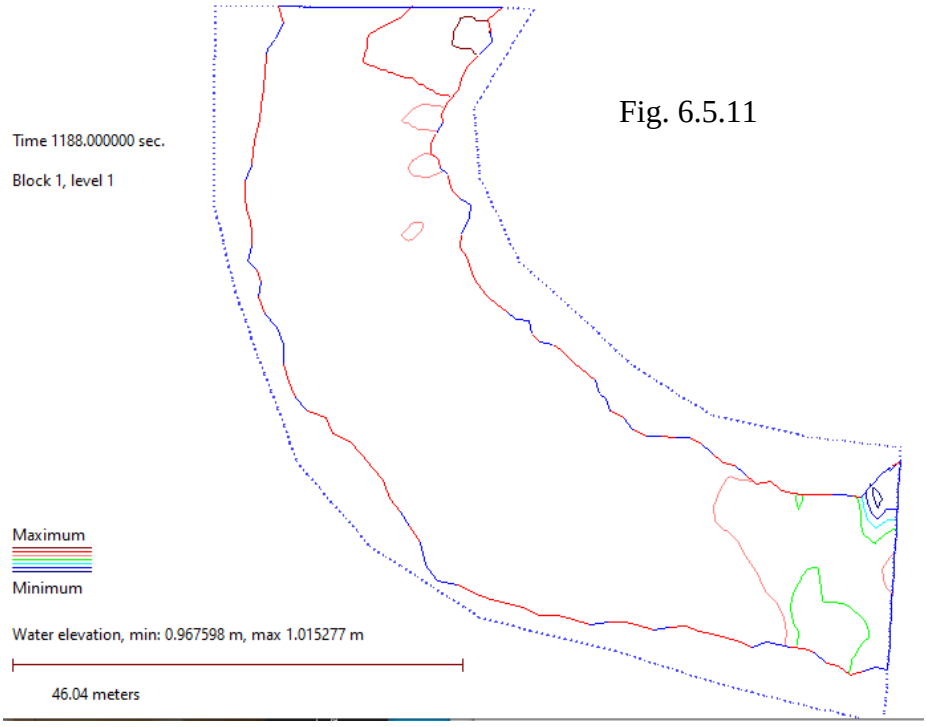



Fig. 6.5.11

## 6.6 Tutorial 6. Sand trap with experimental data

This tutorial is based on the data from the laboratory experiment by Olsen and Skoglund (1994). The case is also used by Olsen et al (2023). A laboratory sand trap were modelled and compared with measured water velocities and sediment concentrations. The geometry and measured data are taken from the web page: *http://www.pvv.ntnu.no/~nilsol/cases/doris*. (Doris is the name of the lab flume)

### 6.6.1 Grid

On the bottom of the web page *http://www.pvv.ntnu.no/~nilsol/cases/doris*, there is a video showing how to make the geometry of the sand trap in SSIIM 2. The video is made for the old SSIIM 2 version from 2018, but the *GridEditor* is fairly similar, so the video will still be useful. The main differences between the *GridEditors* in the new and old version of SSIIM 2 are:

- The *fixedpoints* in the new SSIIM 2 wx version were called *NoMovePoints* in the old version.
- In the old version, the grid line intersections can be moved by clicking on them with the left mouse button, and then dragging the intersections to another location while holding down the left mouse button. In the new SSIIM 2 wx version, you click on the grid intersection with the left mouse button, release the left button and then click on the new location with the right mouse button.
- The dialog box for specifying coordinates is different. In the new version, it is possible to get existing coordinates by specifying the *i* and *j* indexes of the grid. This is not possible in the old version.

Since a detailed description of the grid generation is given in the video, this will not be repeated here. A *koosurf* file with the geometry can be downloaded from the web page *http://www.pvv.ntnu.no/~nilsol/cases/doris*. The *unstruc* file is generated in the first step using the *koosurf* file. This can be done from the *GridEditor*, or it can be done automatically by modifying the *control* file. The *F 2* data set in the *control* file must then be: *F 2 YH test*. The *Y* letter invoke functions that read the koosurf file first, and then make the 3D grid. The *H* letter will invoke a function writing the *unstruc* file. Or the *control.g2* file from the web site can be used and renamed *control* without extension. After the *unstruc* file is made, SSIIM 2 must be terminated before next step.

### 6.6.2 Water velocities

The next step is to compute the water velocities. Before this is done, the discharges need to be specified. This can be done in the *DischargeEditor*, or alternatively, by using the *F 314* and *F 237* data sets, as given in the *control* file from the web site. Edit the *control* file and replace the *YH* on the *F 2* data set with *UW*. Or use the *control.u2* file from the web site, renamed without extension. Then restart the program and wait until it converges. The measured water velocities are given in the *verify.u* file. This file was used in SSIIM 1 to compare measured and computed values directly in the SSIIM 1 graphics. It is actually better to use a spreadsheet for this comparision - the graphics will be better, with scales, legends etc. The *verify* file gives the velocities at profiles specified with (x,y) coordinates. This is the same coordinate system as used for making the grid. Comparing the computed and measured results, it is necessary to produce profiles of computed velocities at the same location as the measurements. This is done with the *interpol* file. A file called *interpol.u* is given on the web page. This can be downloaded and put in the working directory. It also has to be renamed *interpol*, without an extension. Then, the *control* file has to contain the data set *F 48 2*. When this data set and the *interpol*

file is used and SSIIM 2 writes the results, SSIIM 2 will also produce a file called *interres*. This contains the computed velocity profiles at the locations from the *interpol* file. Which is the same as the measured profiles. The data from the *interres* file and the *verify.u* file can then be taken into a spreadsheet and compared.

### 6.6.3 Sediment concentrations

The third step is to compute the sediment concentrations. To do this, the *control* file has to be modified, or the *control.s2* file used. Add *F 37 2*, *F 33 10.0 10* and *F 68 2*. The *F 68 2* data set will invoke a computation where the water velocities are not recomputed for each time step, only the sediment concentrations. This will save computational time and not deteriorate the results as the bed elevation changes were very small. Also, replace *UW* on the *F 2* data set with *F 2 URSM*. The *M* invokes a function writing a new *interres* file. Also, change the *F 48* data set to *F 48 6* instead of *2,* which will produce concentrations instead of velocities. This time, we will use the *intepol.c* file, which can be downloaded and renamed interpol. Also, add an *F 1 D* data set in the *control* file. This will write the sediment fluxes to the boogie file, enabling computation of the trap efficiency for the sand trap. The first parameter on the *K 1* data set can be reduced to 100, to save computational time. This means only 100 time steps are used. A transient computation of the sediment concentration is done here, but over a long time so that a steady state is produced at the end of the computation. Also, the *timei* file from the web site has to be used.

### 6.6.4 sediDriftFoam

An OpenFOAM solver called *sediDriftFoam* has been made that also can compute this case (Olsen et al, 2023). More instructions how to make the program, source code and input files are found on the web page *http://pvv.org/~nilsol/sediDriftFoam*.

# Literature


Ackers, P. and White, R. W. (1973) "Sediment Transport: New Approach and Analysis", Journal of Hydraulic Engineering, ASCE, Vol. 99, No. HY11.

Alfredsen, K. (1998) "Quantification of impacts of river regulations on fish: An energetic modelling approach", HYDROINFORMACTICS '98, Copenhagen, Denmark.

Almeland, S. K., Olsen, N. R. B., Bråtveit, K. and Aryal, P. R. (2019) Multiple solutions of the Navier-Stokes equations computing water flow in sand traps (OpenAccess), Engineering Applications of Computational Fluid Mechanics, Vol. 13, No. 1, pp 199-219. doi:10.1080/19942060.2019.1566094.

Baranya, S. and Jozsa, J. (2007) "Numerical and laboratory investigation of the hydrodynamic complexity of a river confluence", Periodica Polytechnica, Civil Engineering, Vol. 51, No. 1, pp. 3-8.

Baranya, S. and Jozsa, J. (2007) "Comparative CFD and scale model analysis of the flow complexity at a river confluence", 32nd Congress of IAHR, July 1-6 2007, Venice, Italy.

Baranya, S. and Jozsa, J. (2009) "Morphological modeling of a sand-bed reach in the Hungarian Danube", Proceedings of the 33rd Congress of the International Association of Hydraulic Engineering and Research, Vancouver, Canada.

Baranya, S., Olsen, N. R. B., Stoesser, T. and Sturm, T. (2012) "Three-dimensional RANS modeling of flow around circular piers using nested grids", Engineering Applications of Computational Fluid Mechanics, Vol. 6, No. 4, pp 648-662.

Baranya, S., Olsen, N. R. B., Stoesser, T. and Sturm, T. (2013) "A nested grid based computational fluid dynamics model to predict bridge pier scour", Water Management, DOI: 10.1680/wama.12.00104.

Baranya, S., Olsen, N. R. B. and Jozsa, J. (2015) "Flow analysis of a river confluence with field measurements and RANS model with nested grid approach", River Research and Applications, Vol. 31, Issue 1, pp 28-41.

Bengtsson, L. (1973) "Wind Stress on Small Lakes", Tekniska Hogskolan, Lund, Sweden.

Bihs H. and Olsen N.R.B. (2007) "Three-Dimensional Numerical Modeling of Contraction Scour", 32nd IAHR Congress, Venice, Italy.

Bihs, H. and Olsen, N. R. B. (2008) "Three dimensional numerical modeling of secondary flows in channels with longitudinal bedforms", RiverFlow 2008, International Conference on Fluvial Hydraulics, Cesme, Izmir, Turkey, Vol. 1, pp. 179-183.

Bihs, H. and Olsen, N. R. B. (2008) "Three dimensional numerical modeling of pier scour", Fourth International Conferece on Scour and Erosion, Tokyo, Japan.

Bihs, H. and Olsen, N. R. B. (2010) "Numerical investigations of local scour around a trapezoidal abutment using the finite volume method", 1st Conference of the European section of the IAHR, Edinburgh, Scotland.

Bihs, H. and Olsen, N. R. B. (2011), Numerical Modeling of Abutment Scour with the Focus on the Incipient Motion on Sloping Beds, Journal of Hydraulic Engineering, No. 10, pp. 1287-1292. doi:10.1061/(ASCE)HY.1943-7900.0000401

Bindloss, M. (1976) "The light-climate of Loch Leven, a shallow Scottish lake, in relation to primary production of phytoplankton", Freshwater Biology, No. 6.

Booker, D. J., Dunbar, M. J. and Ibbotson, A. (2004) "Predicting juvenile salmonid drift-feeding habitat quality using a three-dimensional hydraulic-bioenergetic model", Ecological Modelling, 177 (1-2): pp. 157-177.

Booker, D. J. (2003) "Hydraulic modelling of fish habitat in urban rivers during high flows" Hydrological Processes, 17 (3): pp 577-599.

Booker, D.J., Sear, D.A. and Payne, A.J. (2001) "Modelling three-dimensional flow structures and patterns of boundary shear stress in a natural pool-riffle sequence", Earth Surface Processes and Landforms, Vol. 26, No. 5, May, page 553-576

Booker, D. J. (2000) "Modelling and monitoring sediment transport in pool-riffle sequences", PhD thesis, Department of Geography, University of Southampton, UK.

Born, M., Brüll, C., Schaefer, D., Hillebrand, G. and Schüttrumpf, H. (2023) Determination of Microplastics' vertical concentration transport (Rouse) profiles in flumes, Environmental Science and Technology, doi:10.1021/acs.est.2c06885

Bowles, C., Daffern, C. D. and Ashforth-Frost, S. (1998) "The Independent Validation of SSIIM - a 3D Numerical Model", HYDROINFORMATICS '98, Copenhagen, Denmark.

Brooks, H. N. (1963), discussion of "Boundary Shear Stresses in Curved Trapezoidal Channels", by A. T. Ippen and P. A. Drinker, Journal of Hydraulic Engineering, ASCE, Vol. 89, No. HY3.

Chandrashekhar, J. (1994) "Numerical Simulation of Sediment Movement in Desilting Basins using SSIIM", M.S. Thesis, Division of Hydraulic and Environmental Engineering, The Norwegian Institute of Technology, Trondheim.

Chapra, S. C. (1997) "Surface water-quality modeling", McGraw-Hill, ISBN 0-07-115242-3.

Conway, P., O'Sullivan, J. J. and Lambert, M. F. (2013) "Stage-discharge prediction in straight compound channels using 3D numerical models", Water Management, Vol. 166, Issue 1, pp 3-15.

Dey, S. (2003) "Threshold of sediment motion on combined transverse and longitudinal sloping beds", Journal of Hydraulic Research, Vol. 41, No. 4, pp. 405-415.

Dordevic, D. (2013) "Numerical study of 3D flow at right-angled confluences with and without upstream planform curvature", Journal of Hydroinformatics, Vol. 15, Issue 4, pp. 1073-1088.

Dordevic, D. and Biron, P. M. (2008) "Role of upstream planform curvature at asymmetrical river confluences - laboratory experiments revisited", River Flow 2008, Vol. 3, pp. 2277-2286.

Dorfmann, C. and Knoblauch, H. (2009) "Calibration of 2D and 3D numerical models of a large reservoir using ADCP measurements", Proceedings of the 33rd Congress of the International Association of Hydraulic Engineering and Research, Vancouver, Canada.

van Dorn, W. (1953) "Wind stress on an artificial pond", Journal of Marine Research, No. 12, pp. 249-276.

Eilertsen, R., Olsen, N. R. B., Ruther, N. and Zinke, P. (2013) "Channel-bed changes in distributaries of the lake Oyeren delta, southern Norway, revealed by interferometric sidescan sonar", Norwegian Journal of Geology, Vol. 93, No. 1.

Einstein, H. A. and Ning Chien (1955) "Effects of heavy sediment concentration near the bed on velocity and sediment distribution", UCLA - Berkeley, Institute of Engineering Research.

Engelund, F. (1953) "On the Laminar and Turbulent Flows of Ground Water through Homogeneous Sand", Transactions of the Danish Academy of Technical Sciences, No. 3.

Engelund, F. and Hansen, E. (1967) "A monograph on sediment transport in alluvial streams", Teknisk Forlag, Copenhagen, Denmark.

Feurich, R., Boubee, J. and Olsen, N. R. B. (2012), Improvement of fish passage in culverts using CFD, Ecological Engineering, Vol 47, Oct. 2012, pp 1-8; doi:10.1016/j.ecoleng.2012.06.013.

Feurich, R. and Olsen, N. R. B. (2012), Finding the Free Surface of Supercritical Flows - a Numerical Investigation, Engineering Applications of Computational Fluid Mechanics, Vol. 6, No. 3, pp 307-315.

Feurich, R. and Olsen, N. R. B. (2012) "3D numerical simulation of supercritical junction flow", IAHR European Conference, Munich, Germany.

Feurich, R. and Olsen, N. R. B. (2011) "Three-Dimensional Modeling of Nonuniform Sediment Transport in an S-shaped Channel", Journal of Hydraulic Engineering, 137, 493 (2011); doi:10.1061/(ASCE)HY.1943-7900.0000321.

Feurich, R., Boubee, J. and Olsen, N. R. B. (2011), Spoiler Baffles in Circular Culverts, Journal of Environmental Engineering. Vol. 137, No. 9. pp. 854-857; doi:10.1061/(ASCE)EE.1943-7870.0000384.

Feurich, R., Kettner, F. and Olsen, N. R. B. (2011) "Three-Dimensional Numerical Investigation of Spillway and Reservoir Hydraulics", 34th IAHR Congress, Brisbane, Australia.

Feurich, R. and Olsen, N. R. B. (2010) "Computation of bed deformation in an S-shaped channel using 3D numerical simulation", 1st Conference of the European section of the IAHR, Edinburgh, Scotland.

Fischer-Antze, T., Stosser, T., Bates, P. and Olsen, N. R. B. (2001) "3D numerical modelling of open-channel flow with submerged vegetation", IAHR Journal of Hydraulic Research, No. 3.

Fischer-Antze, T., Gutknecht, D. and Olsen, N. R. B. (2004) "3D numerical modelling of morphological bed changes in the Danube river", Second International Conference on Fluvial Hydraulics, Naples, Italy.

Fischer-Antze, T., Olsen, N. R. B. and Gutknecht, D. (2008) "Three-dimensional CFD modeling of morphological bed changes in the Danube River", Water Resources Research, 44, W09422, doi:10.1029/2007WR006402.

Fischer-Antze, T., Ruether, N., Olsen, N. R. B and Gutknecht, D. (2009) "3D modeling of non-uniform sediment transport in a channel bend with unsteady flow", Journal of Hydraulic Engineering and Research, Vol. 47, No. 5, pp. 670-675.

Harb, G., Haun, S., Schneider, J. and Olsen, N. R. B. (2014) "Numerical analysis of synthetic granulate deposition in a physical model study", International Journal of Sediment Research, Vol. 29, Issue 1, pp 110-117.

Haun, S., Kjærås, H., Lovfall, S. and Olsen, N. R. B. (2013) "Three-dimensional measurements and numerical modelling of suspended sediments in a hydropower reservoir", Journal of Hydrology, 479, pp 180-188.

Haun, S. and Olsen, N. R. B. (2012) "Three-dimensional numerical modelling of reservoir flushing in a prototype scale", International Journal of River Basin Management, 10:4; pp. 341-349, DOI:10.1080/15715124.2012.736388.

Haun, S. and Olsen, N. R. B. (2012), Three-dimensional numerical modelling of the flushing process of the Kali Gandaki Hydropower Reservoir, Lakes and Reservoirs, Research and Management, Vol. 17, issue 1, pp 25-33.

Haun, S. and Olsen, N. R. B. (2012) "3D numerical simulation of the flushing process in the Angostura reservoir", RiverFlow 2012, San Jose, Costa Rica.

Haun S. and Olsen N.R.B. (2012), "Numerical simulation of sediment deposition in a hydropower reservoir", Proceedings of the 18th IAHR Asia-Pacific Congress, Jeju-Island, South Korea, 2012.

Haun, S., Dorfmann, C., Harb, G. and Olsen, N. R. B. (2012), 3D numerical modelling of the reservoir flushing of the Bodendorf reservoir, Austria, IAHR European Conference, Munich, Germany.

Haun, S., Olsen, N. R. B. and Feurich, R. (2011), Numerical modelling of flow over a trapezoidal broad-crested weir, Engineering Applications of Computational Fluid Mechanics, Vol. 5, No. 3, pp 397-405.

Haun, S., Hoven, L., Olsen, N. R. B., Rodriguez Meza, C. R. and Lizano, L. (2011) "3D numerical modelling of sediment deposition and flushing in the Angostura reservoir, Costa Rica", 34th IAHR Congress, Brisbane, Australia.

Hedger, R. D., Olsen, N. R. B., Malthus, T. J., Atkinson, P. M., George, D. G. (1998) "Dynamically modelling the spatial variation in chlorophyll-a concentration in a remotely sensed image of Loch Leven", 24th Annual Conference of the Remote Sensing Society, Greenwich, UK.

Hedger, R.D., Olsen, N.R.B., George, D.G., Atkinson, P.M., Malthus, T.J. (1999) "Dynamic modelling of the spatio-temporal distribution of phytoplankton in a small productive lake", 4th International Conference on GeoComputation, Fredericksberg, USA.

Hedger, R.D., Olsen, N.R.B., Malthus, T.J., Atkinson, P.M. (1999) "The analysis of water quality spatial distributions in lakes by the integration of computational fluid dynamics with remote sensing", 25th Annual Conference of the Remote Sensing Society, Cardiff, UK.

Hedger, R.D., Olsen, N.R.B., Malthus, T.J., Atkinson, P.M., (2002) "Coupling remote sensing with computational fluid dynamics modelling to estimate lake chlorophyll-a concentration", Remote Sensing of Environment, Vol. 79, No. 1, pp. 116-122.

Hedger, R. D., Olsen NRB, George DG, Malthus TJ, Atkinson PM (2004) "Modelling spatial distributions of Ceratium hirundnella and Microcystis spp. in a small productive British lake", Hydrobiologia, 528 (1-3): pp. 217-227 Oct.

Hillebrand, G., Klassen, I. and Olsen, N. R. B. (2017), 3D CFD modelling of velocities and sediment transport in the Iffezheim hydropower reservoir, Hydrology Research, Vol. 48, Issue 1, pp. 147-159. doi:10.2166/nh.2016.197.

Hillebrand, G., Klassen, I., Olsen, N. R. B. and Vollmer, S. (2012) "Modelling fractionated sediment transport and deposition in the Iffezheim reservoir", 10th International Conference on Hydroinformatics, Hamburg, Germany.

Hillebrand, G. and Olsen, N. R. B. (2011) "Towards modeling consolidation of fine sediments upstream of the Iffezheim barrage, Upper Rhine River, Germany", RCEM 2011, The 7th IAHR Symposium on River, Coastal and Estuarine Morphodynamics, Beijing, China.

Hillebrand, G. and Olsen, N. R. B. (2010) "Hydraulic characteristics of the open annular flume - experiment and numerical simulation", 1st Conference of the European section of the IAHR, Edinburgh, Scotland.

Hoven, L. E. (2010) "Three-dimensional numerical modelling of sedimens in water reservoirs", MSc Thesis, Department of Hydraulic and Environmental Engineering, The Norwegian University of Science and Technology. http://folk.ntnu.no/nilsol/cases/angostura/LisaHoven.pdf.

Jacobsen, J. and Olsen, N. R. B. (2010) "3D numerical modelling of the capacity for a complex spillway", Water Management, Vol. 163, Issue WM6, pp. 283-288.

Jacobsen, T. (1998) "Sediment Problems in Reservoirs. Control of Sediment Deposits", Dr. Ing. Dissertation, Division of Hydraulic and Environmental Engineering, The Norwegian University of Science and Technology.

Kang, H. (2013) "Flow Characteristics and Morphological Changes in Open-Channel Flows with Alternate Vegetation Zones", KSCE Journal of Civil Engineering, 17(5), pp 1157-1165.

Karim, F. (1999) "Bed-form geometry in sand-bed flows", Journal of Hydraulic Engineering, ASCE, Vol. 125, No. 12, pp1253-1261.

Kato, M. and Launder, B. E. (1993), "The Modeling of Turbulent Flow Around Stationary and Vibrating Square Cylinders", Proc. 9th Symposium on Turbulent Shear Flows, Kyoto, August 1993, pp. 10.4.1-10.4.6.

Kjellesvig, H. M. (1996) "Numerical modelling of flow over a spillway", HYDROINFORMATICS-96, Zurich.

Kjellesvig, H. M. and Stole, H. (1996) "Physical and numerical modeling of the Himalaya Intake", 2nd Int. Conf. on Modelling, Testing and Monitoring for Hydro Powerplants, Lausanne, Switzerland.

Kjærås, H. (2012) "Sediments in Angostura Hydropower Reservoir", MSc Thesis, Department of Hydraulic and Environmental Engineering, The Norwegian University of Science and Technology.

Klassen, I., Hillebrand, G., Olsen, N. R. B., Vollmer, S., Lehmann, B. and Nestmann, F. (2011) "Modeling fine sediment aggregation processes considering varying fractal dimensions", RCEM 2011. The 7th IAHR Symposium on River, Coastal and Estuarine Morphodynamics, Beijing, China.

Klassen, I., Gonzalez, W., Seidel, F., Nestmann, F., Hillebrand, G., Hoffmann, T. and Olsen, N. R. B. (2021) 3D numerical analyses of suspended sediment concentrations in a meandering river upstream a hydropower reservoir, 6th IAHR Europe conference, Warsaw, Poland.

Kohler, B. (2001) "Hydraulic parameters controlling fish behaviour and stranding in a laboratory river", Diploma Thesis, Institute of Hydraulic Engineering, University of Stuttgart, Germany.

Kruger, S. and Olsen, N. R. B. (2001) "Shock-wave computations in channel contractions", XXIX IAHR Congress, Beijing, China.

Lane, E. W. (2003) "Design of stable channels", Transactions, ASCE, 120 (1), pp. 1234-1260.

Launder, B. E. and Spalding, D. B. 1974. The numerical computation of turbulent flows, Comput. Meths. Appl. Mech. Eng., 3 (2) : 269-289. DOI:10.1016/0045-7825(74)90029-2.

Lovoll, A., Lysne, D. K. and Olsen, N. R .B. (1995) "Numerical and physical modelling of dynamic impact on structures from a flood wave", IAHR 26th. Biennial Congress, London.

Lovoll, A. (1996) "Hydrodynamic forces from steep waves in rivers", Dr. Ing. Dissertation,

Department of Hydraulic and Environmental Engineering, The Norwegian University of Science and Technology.

Lysne, D. K., Olsen, N. R. B., Stole, H and Jacobsen, T. (1995) "Withdrawal of water from sediment carrying rivers. Recent developments in planning and operation of headworks", International Journal on Hydropower and Dams, March.

Lovfall, S. (2012) "Three-dimensional numerical modelling of sedment deposition in Angostura hydropower reservoir", MSc Thesis, Department of Hydraulic and Environmental Engineering, The Norwegian University of Science and Technology.

Maot, D., Bouchard, J. P. and Alexis, A (2005) "Reservoir Bank Deformation Modelling: Application to Grangent Reservoir", Journal of Hydraulic Engineering, July, pp. 586-595.

Mayer-Peter, E. and Mueller, R. (1948) "Formulas for bed load transport", Report on Second Meeting of International Association for Hydraulic Research, Stockholm, Sweden.

Melaaen, M. C. (1992) "Calculation of fluid flows with staggered and nonstaggered curvilinear nonorthogonal grids - the theory", Numerical Heat Transfer, Part B, vol. 21, pp 1- 19.

Melaaen, M. C. (1990) "Analysis of curvelinear non-orthogonal coordinates for numerical calculation of fluid flow in complex geometries", Dr. Ing. Thesis, Division of Thermodynamics, The Norwegian Institute of Technology.

Minor, B., Rennie, C. D. and Townsend, R. D. (2007) ""Barbs" for river bend bank protection: application of a three-dimensional numerical model", Canadian Journal of Civil Engineering, Vol. 34, pp. 1087-1095.

Naden, P., Rameshwaran, P., Wilson, C. A. M. E., Malki, R., Shukla, D. R. and Shiono, K. (2006) "Inter-comparison of CFD codes using data from a large-scale physical model", IAHR/IWA International Conference on Hydroinformatics, Nice, France, Theme 1D3-4.

Novik, H., Dudhraj, A., Olsen, N. R. B., Bishwakarma, M. B. and Lia, L. (2014) "Numerical modelling of non-uniform flow in settling basins", Hydro Nepal, Issue no. 14, pp 27-35.

Olsen, N. R. B. (1991) "A numerical model for simulation of sediment movements in water intakes", Dr. Ing. Dissertation, The Norwegian Institute of Technology, Trondheim.

Olsen, N. R. B. and Melaaen, M. C. (1993) "Numerical Modeling of Erosion around a Cylinder and Sediment Deposition in a Hydropower Reservoir", Eight International Conference on Numerical Methods in Laminar and Turbulent Flow, Swansea, UK.

Olsen, N. R. B. and Melaaen, M. C. (1993) "Three-dimensional numerical modeling of scour around cylinders", Journal of Hydraulic Engineering, ASCE, Vol. 119, No. 9, September.

Olsen, N. R. B., Jimenez, O., Lovoll, A. and Abrahamsen, L. (1994) "Calculation of water and sediment flow in hydropower reservoirs", 1st. International Conference on Modeling, Testing and

Monitoring of Hydropower Plants, Hungary.

Olsen, N. R. B. and Alfredsen, K. (1994) "A three-dimensional model for calculation of hydraulic parameters for fish habitat", IAHR Conference on Habitat Hydraulics, Trondheim, Norway.

Olsen, N. R. B. (1994) "SSIIM - A three-dimensional numerical model for simulation of water and sediment flow", HYDROSOFT-94, Porto Carras, Greece.

Olsen, N. R. B. and Skoglund, M. (1994) "Three-dimensional numerical modeling of water and sediment flow in a sand trap", IAHR Journal of Hydraulic Research, No. 6.

Olsen, N. R. B. and Tesaker, E. (1995) "Numerical and physical modeling of a turbidity current", IAHR 26th. Biennial Congress, London.

Olsen, N. R. B. and Oldervik, O. (1995) "Three-dimensional numerical modeling of water flow through a gate plug", IAHR 26th. Biennial Congress, London.

Olsen, N. R. B. and Stokseth, S. (1995) "Three-dimensional numerical modeling of water flow in a river with large bed roughness", IAHR Journal of Hydraulic Research, Vol. 33, No. 4.

Olsen, N. R. B. and Chandrashekhar, J. (1995) "Calculation of water and sediment flow in desilting basins", 6th. International Symposium on River Sedimentation, New Delhi, India.

Olsen, N. R. B. (1995) "Numerical modelling of hydropower reservoir flushing", International Conference on Hydropower into the next Century", Barcelona, Spain.

Olsen, N. R. B. and Melaaen, M. C. (1996) "Three-dimensional numerical modeling of transient turbulent flow around a circular cylinder", 2nd. Int. Conf. on Modelling, Testing and Monitoring for Hydro Powerplants, Lausanne, Switzerland.

Olsen, N. R. B. (1996) "Three-dimensional numerical modelling of local scour", Hydroinformatics-96, Zurich.

Olsen, N. R. B. (1997) "Computational fluid dynamics as a tool for prediction of sedimentation and erosion in reservoirs", Q. 74 a), ICOLD Conference, Florence, Italy.

Olsen, N. R. B. (1997) "A framework for a 3D numerical model for hydropower reservoir water quality", HYDROPOWER '97, Trondheim, Norway.

Olsen, N. R. B. and Kjellesvig, H. M. (1997) "A 3D numerical model for determination of spillway capacity", HYDROPOWER '97, Trondheim, Norway.

Olsen, N. R. B. and Kjellesvig, H. M. (1997) "3D Numerical Modelling of Sediment Deposition and Bed Changes in a Tunnel-Type Sand Trap", IAHR/ASCE Congress, San Francisco, USA.

Olsen, N. R. B. (1997) "Computational Fluid Dynamics in Hydraulic and Sedimentation Engineering", Class notes, Division of Hydraulic and Environmental Engineering,

The Norwegian University of Science and Technology.

Olsen, N. R. B. and Kjellesvig, H. M. (1998) "Three-dimensional numerical flow modelling for estimation of maximum local scour depth", IAHR Journal of Hydraulic Research, No. 4.

Olsen, N. R. B. and Kjellesvig, H. M. (1998) "Three-dimensional numerical flow modelling for estimation of spillway capacity", IAHR Journal of Hydraulic Research, No. 5.

Olsen, N. R. B. and Heslop, S. (1998) "Flow visualisation by particle animation for 3D CFD modelling in hydraulic engineering", HYDROINFORMATICS '98, Copenhagen, Denmark.

Olsen, N. R. B. (1998) "Unstructured and nested grids for 3D CFD modelling in hydraulic engineering", HYDROINFORMATICS '98, Copenhagen, Denmark.

Olsen, N. R. B. and Tjomsland, T. (1998) "3D CFD modelling of wind-induced currents and radioactive tracer movements in a lake", 3rd. International Conference on Hydroscience and Engineering, Cottbus, Germany.

Olsen, N. R. B., Hedger, R. D., George, D. G. and Heslop, S. (1998) "3D CFD modelling of spatial distribution of algae in Loch Leven, Scotland", 3rd. International Conference on Hydroscience and Engineering, Cottbus, Germany.

Olsen, N. R. B. (1999) "Two-dimensional numerical modelling of flushing processes in water reservoirs", IAHR Journal of Hydraulic Research, Vol. 1.

Olsen, N. R. B. and Wilson, C. A. M. E. (1999) "CFD modelling of rivers and reservoirs", Int. Seminar on Optimum Operation of Run of River Reservoirs, Trondheim, Norway.

Olsen, N. R. B., Jimenez, O., Lovoll, A. and Abrahamsen, L. (1999) "3D CFD modelling of water and sediment flow in a hydropower reservoir", International Journal of Sediment Research, Vol. 14, No. 1.

Olsen, N. R. B. and Kjellesvig, H. M. (1999) "Three-dimensional numerical modelling of bed changes in a sand trap", IAHR Journal of Hydraulic Research, Vol. 37, No. 2. abstract

Olsen, N. R. B. and Lysne, D. K. (1999) "3D Numerical Modelling of Water Currents in an Ice-Covered Lake", IAHR 28th. Biennial Congress, Graz, Austria.

Olsen, N. R. B., Hedger, R. D. and George, D. G. (1999) "Modelling the Horizontal Distribution of Algae in a Water Supply Reservoir", IAHR 28th. Biennial Congress, Graz, Austria.

Olsen, N.R.B., Hedger, R. D., Heslop, S., George, D. G. (1999) "Computing spatial variation of algae in water supply reservoirs", International Conference on Computing and Control for the Water Industry, CCWI'99, Exeter, UK.

Olsen, N. R. B. and Lysne, D. K. (2000) "Numerical modelling of circulation in Lake Sperillen, Norway", Nordic Hydrology, Vol. 31, No. 1.

Olsen, N. R. B. (2000) "Unstructured hexahedral 3D grids for CFD modelling in fluvial geomorphology", Hydroinformatics 2000, Iowa, USA.

Olsen, N. R. B. (2000) "CFD modelling of bed changes during flushing of a reservoir", Hydroinformatics 2000, Iowa, USA.

Olsen, N. R. B., Hedger, R. D. and George, D. G. (2000) "3D Numerical Modelling of Microcystis Distribution in a Water Reservoir", Journal of Environmental Engineering, ASCE, Vol. 126, No. 10, October.

Olsen, N. R. B. and Aryal, P. R. (2001) "3D CFD modelling of water flow in the sand trap of Khimti Hydropower Plant, Nepal", Hydropower -2001, Bergen, Norway.

Olsen, N. R. B. (2002) "Estimating meandering channel evolution using a 3D CFD model", Hydroinformatics 2002, Cardiff, UK.

Olsen, N. R. B. (2003) "3D CFD Modeling of a Self-Forming Meandering Channel", Journal of Hydraulic Engineering, ASCE, No. 5, May.

Olsen, N. R. B., Pegg, I., Alfredsen, K. T., Fergus, T. and Fjeldstad, H-P. (2004) "3D CFD modelling of sediment deposition in habitat improvement structures", 5th International Symposium on Ecohydraulics, Madrid, Spain.

Olsen, N. R. B. (2004) Closure to "Three-dimensional CFD modeling of self-forming meandering channel", Journal of Hydraulic Engineering, ASCE, Vol. 130, No 8, pp. 838-839.

Olsen, N. R. B., Pegg, I, Molliex, J., Berger, H. and Fjeldstad, H-P, (2005) "3D CFD mdelling of erosion in habitat improvement gravel", 31st. IAHR Congress, Seoul, Korea.

Olsen, N. R. B., Aberle, J. and Koll, K. (2010) "Resolving large bed roughness elements with an unstructured hexahedral grid", RiverFlow 2010, Braunschweig, Germany.

Olsen, N. R. B. and Haun, S. (2010) "Free surface algorithms for 3D numerical modelling of reservoir flushing", RiverFlow 2010, Braunschweig, Germany.

Olsen, N. R. B. (2010) "Result assessment methods for 3D CFD models in sediment transport computations", 1st Conference of the European section of the IAHR, Edinburgh, Scotland.

Olsen, N. R. B. (2013) "Numerical Algorithms for Predicting Sediment Slides in Water Reservoirs", Electronic Journal of Geotechnical Engineering, Vol. 18, Bund.Y, Paper 2013.496.

Olsen, N. R. B. (2015) "Olsen, N. R. B. (2015), Four free surface algorithms for the 3D Navier-Stokes equations", Journal of Hydroinformatics, Vol. 17, Issue 6, pp 845-856.

Olsen, N. R. B. (2017) “Numerical modelling of downstream-migrating antidunes”, Earth Surface Processes and Landforms.Vol 42, pp 2393–2401. doi:10.1002/esp.4193.

Olsen, N. R. B. and Hillebrand, G. (2018) “Long-time 3D CFD modelling of sedimentation with dredging in a hydropower reservoir”, Journal of Soils and Sediments. doi:10.1007/s11368-018-1989-0.

Olsen, N. R. B. (2020) Numerical modelling of sediment deposition in Lake Mills, USA. River Flow 2020, Delft, The Netherlands.

Olsen, N. R. B. and Haun, S. (2020) A numerical geotechnical model for computing soil slides at banks of water reservoirs, International Journal of Geo-Engineering. Vol. 11, Article no. 22, doi:10.1186/s40703-020-00129-w.

Olsen, N. R. B. (2021) 3D numerical modelling of braided channel formation, Geomorphology, 375, 107528. doi:10.1016/j.geomorph.2020.107528.

Olsen, N. R. B., Vollmer, S., Pribil, D., Hollmann, N. and Gintz, D. (2021) Numerical modelling of granjanite tracer movements in the Rhine river, 6th IAHR Europe conference, Warsaw, Poland.

Olsen, N. R. B., Aberle, J. and Núñez-González, F. (2022) Parametrization of turbulence over antidunes, Proceedings of the 39th IAHR World Congress, Granada, Spain.

Olsen, N. R. B. (2022) Explaining the formation of sedimentary structures under antidunes using a 2D width-averaged numerical model, Norwegian Journal of Geology, Vol. 102, doi: 10.17850/njg102-3-2.

Olsen, N. R. B., Kadia, S., Pummer, E. and Hillebrand, G. (2023) An OpenFOAM solver for computing suspended particles in water currents, Journal of Hydroinformatics. doi:10.2166/hydro.2023.309.

Olsen, N. R. B. (2023) Numerial modelling of a laboratory scale turbidity current, Proc. 40th IAHR Congress, Vienna, Austria.

Patankar, S. V. (1980) "Numerical Heat Transfer and Fluid Flow", McGraw-Hill Book Company, New York.

Perez, S. and Caliendo, W. (2008) "Calculating marine propeller scour using SSIIM CFD software", Journal of Marine Environmental Engineering.

Rameshwaran, P., Naden, P., Wilson, C. A. M. E., Malki, R., Shukla, D. R. and Shiono, K. (2013) "Inter-comparison and validation of computational fluid dynamics codes in two-stage meandering channel flows", Applied Mathematical Modelling, (37) pp 8652-8672.

Reynolds, C. S. (1984) "The ecology of freshwater phytoplankton", Cambridge University Press, Cambridge, UK.

Rhie, C.-M, and Chow, W. L. (1983) "Numerical study of the turbulent flow past an airfoil with trailing edge separation", AIAA Journal, Vol. 21, No. 11.

van Rijn, L. C. (1982) "Equivalent Roughness of Alluvial Bed", Journal of Hydraulic Engineering, ASCE, Vol. 108, No. 10.

van Rijn, L. C. (1987) "Mathematical modeling of morphological processes in the case of suspended sediment transport", Ph.D Thesis, Delft University of Technology.

Roberts, M., Olsen, N. R. B. and Vollmer, S. (2012) "3D modelling of changes in sediment transport, bed composition and porosity during a flood event at the river Elbe", 10th International Coneference on Hydroinformatics, Hamburg, Germany.

Rodi, W. (1980) "Turbulence models and their application in hydraulics", IAHR State-of- the-art paper.

Rouse, H (1937) "Modern Conceptions of the Mechanics of Fluid Turbulence", Transactions, ASCE, Vol. 102, Paper No. 1965.

Ruether, N. and Olsen. N.R.B. (2003) "CFD modeling of meandering river evolution", XXX IAHR Congress, Thessaloniki, Greece.

Ruether, N. and Olsen, N. R. B. (2003) "CFD modeling of alluvial channel instabilities", 3rd IAHR Symposium on River, Coastal and Estuarine Morphodynamics, Barcelona, Spain.

Ruether, N. and Olsen. N.R.B. (2004) "Three dimensional modeling of sediment transport in a channel bend", Second International Conference on Fluvial Hydraulics, Naples, Italy.

Ruether, N., Singh, J. M., Olsen, N. R. B. and Atkinson, E. (2005) "Three-dimensional modelling of sediment transport at water intakes", Proceedings of the Institution of Civil Engineers, UK, Water Management, Vol. 158, March, pp 1-7.

Ruether, N. and Olsen. N.R.B. (2005) "Three dimensional modeling of sediment transport in a narrow 90 degree channel bend", Journal of Hydraulic Engineering, ASCE, Vol. 131, pp. 917-920, doi:10.1061/(ASCE)0733-9429(2005)131:10(917).

Ruether, N. and Olsen, N. R. B. (2005) "Advances in 3D modeling of free-forming meander formation from initially straight alluvial channels", 31st. IAHR Congress, Seoul, Korea.

Ruether, N., Olsen. N.R.B. (2006) "Towards the prediction of free-forming meander formation using 3D computational fluid dynamics", Flow Simulation in Hydraulic Engineering, Dresden, Germany, 9-11 March 2006.

Ruether, N., Olsen. N.R.B. (2006) "3D modeling of transient bed deformation in a sine-generated laboratory channel with two different width to depth ratios", Third International Conference on Fluvial Hydraulics, River Flow 2006, Lisbon, Portugal.

Ruether, N. and Olsen, N. R. B. (2007) "Modelling free-forming meander evolution in a laboratory channel using three-dimensional computational fluid dynamics", Geomorphology, No. 89, pp. 308-319.

Rueter, N., Olsen, N. R. B. and Eilertsen, R. (2008) "3D modeling of flow and sediment transport over natural dunes", RiverFlow 2008, International Conference on Fluvial Hydraulics, Cesme, Izmir, Turkey, Vol. 2, pp. 1479-1485.

Ruther, N., Jacobsen, J., Olsen, N. R. B. and Vatne, G. (2010) "Prediction of the three dimensional flow field and bed shear stresses in a regulated river in Mid-Norway", Hydrology Research, Vol, 41, No. 2, pp. 145-152.

Schaefer, D., Hillebrand, G. and Olsen, N. R. B. (2021) Numerical 3D-Modelling of Hydrodynamics and Microplastic Transport of a section of the Rhine, 6th IAHR Europe conference, Warsaw, Poland.

Schlichting, H. (1979) "Boundary layer theory", McGraw-Hill.

Seed, D. (1997) "River training and channel protection - Validation of a 3D numerical model", Report SR 480, HR Wallingford, UK.

Sintic, A. (1996) "Numerical models for dam-break flood routing", MS Thesis, Institute of Hydraulic Engineering and Water Resources Management, University of Technology, Aachen, Germany.

Sokoray-Varga B., Baranya S. and Jozsa J. (2006): "Turbulence analysis of vertical slot fish pass by in situ measurements and CFD modelling', Third International Conference on Fluvial Hydraulics, River Flow 2006, Lisbon, Portugal.

Sokoray-Varga, B., Baranya, S. and Jozsa, J. (2007) "Turbulent strucutures in vertical slot fish pass revealed by in situ measurements and numerical modelling", Fifth International Symposium on Environmental Hydraulics (ISEH V). Tempe, Arizona.

Spalart, P. R. and Allmaras, S. R. (1994) "A one-equation turbulence model for aerodynamic flows", La Recherche Aerospatiale, no. 1, pp. 5-21

Stoesser, T. (1998) "Numerical Modelling as Tool in Hydraulics - In Case of Reservoir Sedimentation Processes in Mountainous Regions", Dipl. Ing. Thesis, Institute of Hydraulic Engineering and Water Resources Management, University of Karlsruhe, Germany.

Stoesser, T., Rodi, W. and Olsen, N. R. (2006) "Large Eddy and RANS flow simulation above dunes", Flow Simulation in Hydraulic Engineering, Dresden, Germany, 9-11 March 2006.

Stoesser, T., von Terzi, D., Rodi, W. and Olsen, N. R. B. (2006) "RANS simulations and LES over dunes at low relative submergence ratios", 7th Int. Conf. on Hydroscience and Engineering, Philadelphia, USA, 10-13 September 2006.

Stoesser, T., Ruether, N. and Olsen, N. R. B. (2008) "Near-Bed Flow Behavior in a Meandering Channel", RiverFlow 2008, International Conference on Fluvial Hydraulics, Cesme, Izmir, Turkey, Vol. 1, pp. 793-799.

Stoesser, T., Ruether, N. and Olsen, N. R. B. (2009) "Calculation of Primary and Secondary Flow and Boundary Shear Stresses in a Meandering Channel", Advances in Water Resources, doi:10.1016/j.advwatres.2009.11.001.

Streeter, H. W. and Phelps, E- B. (1925) "A study of the pollution and natural purification of the Ohio

River", US Public Health Service, Washington DC, Bulletin 146.

Tritthart, M. and Gutknecht, D. (2007) "3-D computation of flood processes in sharp river bends", Proceedings of the Institution of Civil Engineers - Water Management, Vol. 160, Issue 4, pp. 233-247.

Vanoni, V., et al (1975) "Sedimentation Engineering", ASCE Manuals and reports on engineering practice - No54.

Vingerhagen, S. and Olsen, N. R. B. (2012), 3D numerical modelling of the capacity for a partially pressurized spillway, IAHR European Conference, Munich, Germany.

Viscardi, J. M., Pujol, A., Weitbrecht, V., Jirka, G. H. and Olsen, N. R. B. (2006) "Numerical Simulations on the Parana de las Palmas River", Third International Conference on Fluvial Hydraulics, River Flow 2006, Lisbon, Portugal.

Wilcox, D. C. (2000) "Turbulence modelling for CFD", DCW industries, ISBN. 0-9636051-5-1.

Wildhagen, J., Ruether, N., Olsen, N. R. B. and Guymer, I. (2005) "Three-dimensional modelling of sediment transport in a sharply curved menadering channel", 31st. IAHR Congress, Seoul, Korea.

Wilson, C. A. M. E., Olsen, N. R. B., Boxall, J. B. and Guymer, I. (2003) "Three-dimensional numerical simulation of solute transport in a meandering channel" XXX IAHR Congress, Thessaloniki, Greece.

Wilson C. A. M. E., Stoesser T., Olsen N. R. B. and Bates P. D. (2003) "Application and Validation of Numerical Codes in the Prediction of Compound Channel Flows", Proceedings of ICE, Water, Maritime and Energy 153 pp. 117-128.

Wilson C. A. M. E., Boxall J. B., Guymer I. and Olsen N. R. B. (2003) "Validation of a 3D numerical code in the simulation of pseudo-natural meandering flows", Journal of Hydraulic Engineering, ASCE, Vol. 129, No. 10, October.

Wilson, C. A. M. E., Yagci, O., Olsen, N. R. B. and Rauch, H. P. (2004) "3D numerical modelling of vegetated compound channel flows", 6th International Conference on Hydroinformatics, Singapore.

Wilson, C. A. M. E., Stoesser, T. and Olsen, N. R. B. (2004) "Validation of a 3D Computational Dynamics Code in the Simulation of Meandering Compound Channel Flows", The Sixth International Conference on Hydroscience and Engineering, Brisbane, Australia.

Wilson, C. A. M. E., Yagci, O., Rauch, H.-P. and Olsen, N. R. B. (2006) "3D numerical modelling of a willow vegetated river/floodplain system", Journal of Hydrology, Vol. 327, July 2006, pp. 13-21.

Wilson, C. A. M. E., Guymer, I., Boxall, J. B. and Olsen, N. R. B. (2007) "Three-dimensional numerical simulation of solute transport in a meandering self-formed river channel", Journal of Hydraulic Research, October.

Wormleaton, P. R. and Ewunetu, M. (2006) "Three-dimensional k-epsilon numerical modeling of

overbank flow in a mobile bed meandering channel with floodplains of different depth, roughness and planform", Journal of Hydraulic Research, vol. 44, Issue 1, pp. 18-32.

Wu, J. (1969) "Wind Shear Stress and Surface Roughness at Air-Sea Interface", Journal of Geophysical Research, No. 74, pp. 444-455.

Yang, T. C. (1973) "Incipient Motion and Sediment Transport", Journal of Hydraulic Engineering, ASCE, Vol. 99, No HY10.

Zinke, P. and Olsen N.R.B. (2007) "Modeling of sediment deposition in a partly vegetated open channel", 32nd IAHR Congress, Venice, Italy.

Zinke, P., Olsen, N. R. B. and Sukhodolova, T. (2008) "Modeling of hydraulics and morphodynamics in a vegetated river reach", RiverFlow 2008, International Conference on Fluvial Hydraulics, Cesme, Izmir, Turkey, Vol. 1, pp. 367-376.

Zinke, P., Olsen, N. R. B. and Ruether, N. (2008) "3D Modelling of the flow distribution in the delta of Lake Oyern (Norway)", International Conference on Hydroscience and Engineering, Nagoya, Japan, pp. 270-280.

Zinke, P., Ruether, N., Olsen. N.R.B., Eilertsen, R.S. (2009) "3D Modelling of morphodynamics in deltas due to Hydropower regulation", Proc. of the Annual Conference on Hydraulic Engineering: "Water power and climate change", Dresdner Wasserbauliche Mitteilung, Technical University Dresden, Germany.

Zinke, P. and Olsen, N. R. B. (2009) "Numerical modeling of water and sediment flow in a delta with natural vegetation", 33rd IAHR Congress, Vancouver, Canada.

Zinke, P., Olsen, N. R. B., Bogen, J. and Ruther, N. (2010) "3D modelling of the flow distribution in the delta of Lake Oyern, Norway", Hydrology Research, Vol, 41, No. 2, pp. 92-103.

Zinke, P., Olsen, N. R. B. and Bogen, J. (2011), "3D numerical modeling of levee depositions in a Scandinavian freshwater delta", Geomorphology 129 (2011), pp. 320-333; doi:10.1016/j.geomorph.2011.02.027.

# Appendix I. Transfer of grid between SSIIM 1 and SSIIM 2.

**Transfer of grid from SSIIM 1 to SSIIM 2.**

1. Import the *koordina* file from SSIIM 1 into a spreadsheet.
2. Add the water surface as the last column, similar to the *koosurf* file in SSIIM 1
3. Export the file to an ASCII file, and rename it *koosurf*, without any extension
4. Start up SSIIM 2 and its *GridEditor*.
5. In the *GridEditor* menu, choose *Blocks->Add from koosurf*
6. In the *GridEditor* menu, choose *Generate->3D grid*
7. In the main program interface, choose *Files -> Save unstruc*

If the program stops and the user interface disappears during the procedure, you need to add or modify some parameters in the *control* file and repeat the procedure from point 4. More detailed instructions for the *control* file data sets modifications are written to the *boogie* file.

**Transfer of grid from SSIIM 2 to SSIIM 1.**

Transfer of grid from SSIIM 2 to SSIIM 1 is only possible if the grid has only one block.

1. Start up SSIIM 2 and read the grid
2. In the main menu, choose *File->write koordina*
3. A file called *koordina.si1* is written. Remove the extension and use it as the normal *koordina* file in SSIIM 1.
4. Edit the *G 1* data set in the *control* file to fit the size of the *koordina* file.