CFD Algorithms
for
Hydraulic Engineering

By Nils Reidar B. Olsen

Department of Hydraulic and Environmental Engineering
The Norwegian University of Science and Technology

18. December 2000

ISBN 82 7598-044-5

In memory of my father, Ralph Eivind Olsen

Foreword

After finishing my *CFD Class Notes* (Olsen, 1999), I realized there were a number of topics not included in this basic text. Although there exist a number of books on CFD modelling, I thought it would be practical to have a book with a detailed and hopefully simple description of the more advanced topics for CFD in Hydraulic Engineering. It would provide a general insight into the workings of a CFD program as an advice to both users and people making new programs. The book also gives some insight into my research over the last years, and I hope it can give inspiration to further research in this field. Finally, it provides documentation of many of the algorithms in my *SSIIM* computer programs.

Unlike the *CFD Class Notes*, the present text is only focused on the finite volume method. I have included details of topics I think is relevant for hydraulic engineering. This applies especially to grid generation, roughness modelling and sediment transport.

I want to thank Prof. Morten Melaaen for his kind assistance and cooperation over the last ten years. I have relied heavily on his doctoral thesis in Chapter 4. I also want to thank Koen Blanchaert on assistance in collecting information for Chapter 6.11, bed load transport on a transverse sloping bed.

Examples where the algorithms have been used are numerous, but I have only presented a limited number in this text. The reader is referred to the list of literature at the end, the CFD Class Notes (Olsen, 1999) or one of the several web pages on CFD in hydraulic engineering, for example *www.bygg.ntnu.no/~nilsol/cfd*.

Table of content

1. Introduction

In recent years the science of Computational Fluid Dynamics has found its way to Hydraulic Engineering. A large number of hydraulic problems have been solved using CFD. Examples are given in the box below.

There exist a number of commercial, general-purpose CFD programs to solve fluid flow problems. However, Hydraulic Engineering poses many special problems requiring solutions not included in the general codes. A CFD program tailor made for Hydraulic Engineering, such as SSIIM, incorporates many special algorithms. It is the purpose of the present book to provide an insight into these. Some of the more basic text is also included, to provide background for the special algorithms

The next chapter gives a description of the various grids, with definitions and generation procedures. Algorithms for generation of structured and unstructured grids are described, together with adaptive grid movements in vertical and horizontal directions, used in wetting/drying problems. Adaptive grid cell shape changes are also described.

The third chapter describes the convection-diffusion equations. The chapter is similar to the information given in the CFD Class Notes.

The fourth chapter describes the Navier-Stokes equations. A more advanced approach is used, compared with the CFD Class Notes. Discretization of all the terms are given in detail, and tensor calculus is used together with the previously used discretization by physical information.

Examples of hydraulic engineering problems modelled using CFD:

- **Lake circulation** (Simons, 1974; Olsen and Lysne, 2000)
- **Flow pattern in a river** (Oestberg and Johanson, 1992; Olsen and Stokseth, 1995)
- **Flow pattern in a reservoir** (Olsen et. al. 1994)
- **Flow around groynes** (Seed, 1997)
- **Sediment deposition in a sand trap** (Olsen and Skoglund, 1994)
- **Local scour** (Olsen and Kjellesvig, 1998; Roulund, 2000)
- **Channel morphology** (Wu et. al. 1999)
- **Determination of coefficient of discharge for a spillway** (Olsen and Kjellesvig, 1998; Spaliviero and May, 1998)
- **Reservoir flushing** (Maurel et. al., 1998; Olsen, 1999)
- **Algae movements in a reservoir** (Olsen et. al., 2000)

The fifth chapter provides information about algorithms particularly used in Hydraulic Engineering.

The sixth and last chapter describes sediment transport modelling algorithms, including suspended load, bed load, bed movements, bed forms, critical shear stress, multiple grain sizes etc.

2. Grids

One of the main concepts behind CFD is to divide the water geometry into small cells. Equations for velocity, turbulence, water quality, sediment concentration etc. are then solved for each cell. The cells are obtained by dividing the water body into a grid. The composition and quality of the grid is important for the accuracy and stability of the solution of the equations.

2.1 Classifications

Grids can be classified according to several characteristics:

shape
orthogonality
structure
blocks
position of variable
grid movements

The **shape** of the cells is usually **triangular** or **quadrilateral** in 2D:

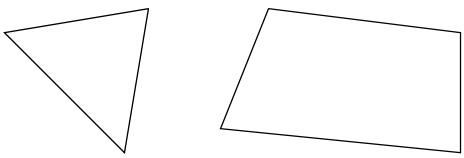


Fig. 2.1.1 Triangular and quadrilateral grid shapes

In 3D, the cells are **tetrahedral** (four sides) or **hexahedral** (six sides).

The **orthogonality** of the grid is determined by the angle between crossing grid lines. If the angle is 90 degrees, the grid is orthogonal. If it is different from 90 degrees, the grid is non-orthogonal.

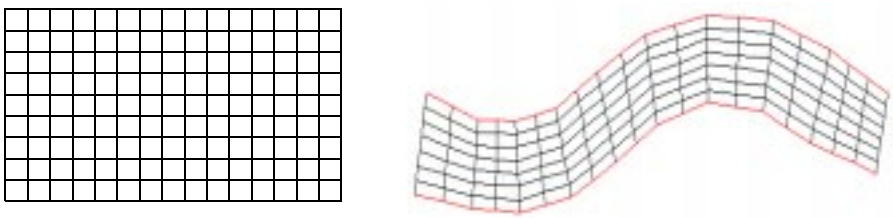
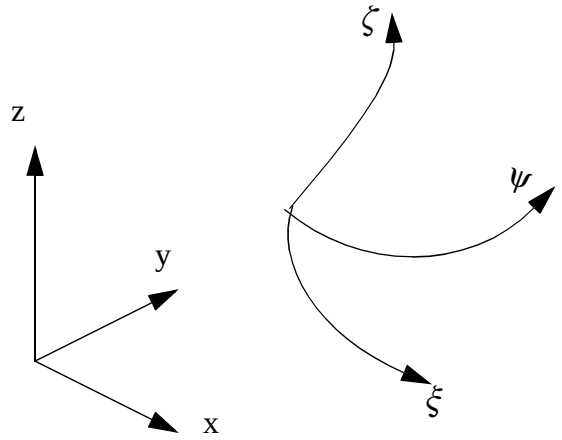


Fig. 2.1.2 Orthogonal grid (left) and non-orthogonal grid (right)

For non-orthogonal grids, a non-orthogonal coordinate system is often used to derive terms in the equations. The coordinates then follow the grid lines of a structured grid. The three non-orthogonal coordinate lines are often called x,y,z, corresponding to x,y and z in the orthogonal coordinate system. This is also shown on the figure below:

Fig. 2.1.3 Non-orthogonal coordinate system following the grid lines.



Grids can be **structured** or non-structured. Often a structured grid is used in finite volume methods and an unstructured grid is used in finite element methods. However, this is not always the case. The figure below shows a structured and an unstructured grid. In a structured grid it is possible to make a two-dimensional array indexing the grid cells. If this is not possible, the grid is unstructured. Almost all grids using triangular cells are unstructured.

A **multi-block** grid is a made from several structured grids as shown on the figure below.

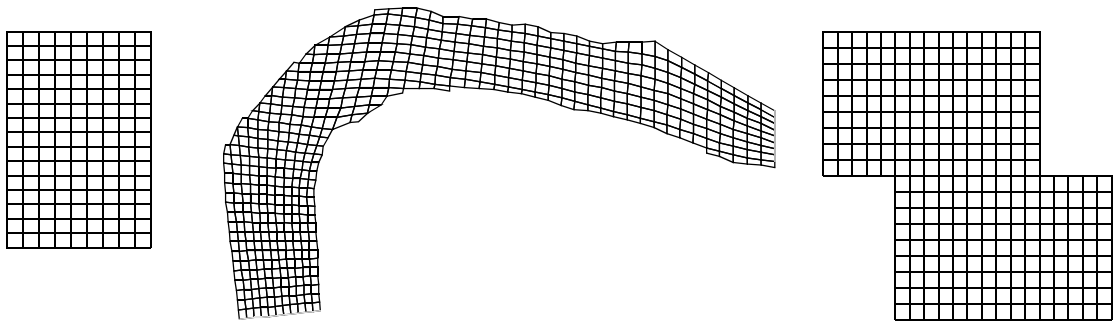


Fig. 2.1.4 Structured grid (left), unstructured grid (middle) and multi-block grid (right)

An **adaptive** grid moves according to the calculated flow field or the physics of the problem. When the water surface or the bed moves during a time step, it is possible to make

the grid move accordingly, to calculate the situation for the new geometry. Thereby time-dependent calculations of bed changes and water levels can be done.

An adaptive grid is used to model bed changes in for example sediment deposition, reservoir flushing or local scour. It is also used to model changes in the water surface when for example calculating a flood wave.

2.2 Structured grid generation

There are a number of different methods to create the internal points in a structured grid. The most used are transfinite interpolation and elliptic grid generation. Transfinite interpolation generates straight lines in one of the grid directions. Elliptic generation distributes the points more smoothly. This is done by solving a Laplace or Poisson equation for the location of the grid line intersections:

$$\nabla^2 \xi^i = P^i \qquad (2.2.1)$$

The location of the grid lines are denoted x. *P* is a source term used for attracting grid lines to a side or a point. An example of using elliptic grid generation is given below:

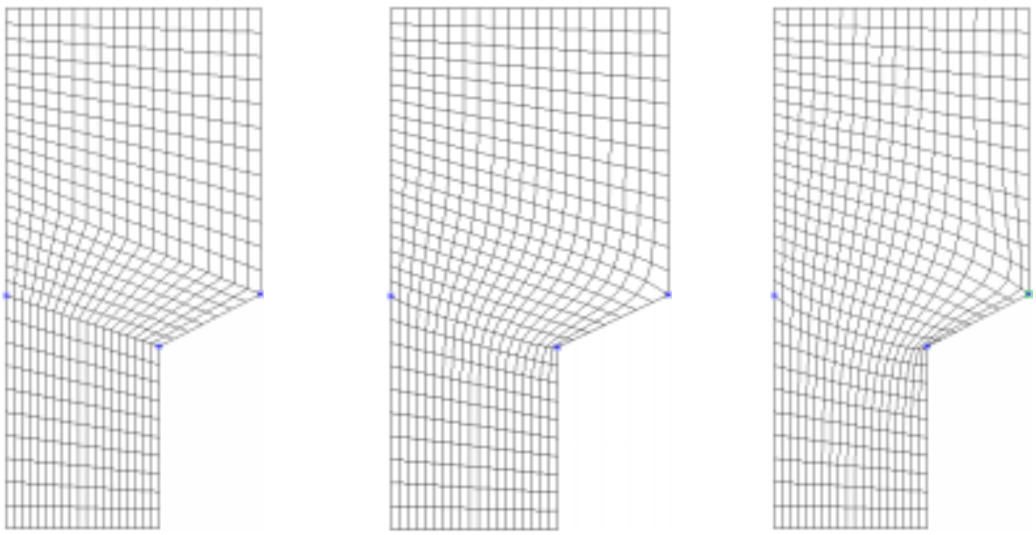


Fig. 2.2.1. Grid generation by transfinite interpolation (left), elliptic generation with no attraction (middle) and attraction to the left boundary (right)

The left grid is made by use of transfinite interpolation. This means that straight lines are made between the points on the wall.

Elliptic procedures are used to obtain the middle grid in Fig. 2.3.1, with no attractions. This means P=0 for Equation 2.3.1.

Looking at the lower point of the step, the cells are smaller closer to the wall. This is due to attraction functions. P > 0 for Equation 2.2.1.

The solution of Eq. 2.2.1 is explained in detail by Thompson et. al. (1985).

2.3 Unstructured grid generation

The advantages of the unstructured grid for modelling complex geometries is evident. Algorithms to generate 3D grids based on tetrahedral cells are numerous, and are included into most commercial CFD programs. However, hexahedral cells gives higher accuracy and speed of the calculation. The present-day automatic methods to generate unstructured hexahedral grids for general problems seems not to be very successful. However, in hydraulic engineering the nature of the problem often makes it possible to rapidly generate an unstructured hexahedral grid manually. The process is based on the knowledge of the geometry and insight into the flow problem.

In the following, such a method will be described. The method is used in the SSIIM 2 program, and has been successfully applied to several hydraulic engineering flow problems.

Domain decomposition in structured blocks

Modelling a natural river or a lake, there are some basic characteristics of the domain. The vertical dimensions are usually much smaller than the horizontal dimensions. It is therefore possible to use completely vertical grid cells in one direction, without causing too non-orthogonal angles between the grid lines. The advantage is that the 3D grid can be made relatively fast by first generating a 2D depth-integrated grid. The grid distribution in the vertical direction is described in the next chapter.

Generating the 2D depth-averaged grid, the geometry can be very complex. However, often rivers or long parts of lakes will have a relatively constant width. Usually, there is also a dominant flow direction along the longest side of a sub-domain. It is then convenient to use a structured grid

for each such part. The advantage is that the cells can be made longer in the dominant flow direction, where there are smaller gradients. Also, the grid alignment will decrease false diffusion.

In the approach used by SSIIM 2, the user generates blocks of structured grids graphically, and connects these afterwards. The process is relatively fast, and a complex grid can be generated in a couple of hours. An automatically generated unstructured grid will usually have lower qualities, so the extra work by the user is often justified.

An example is given below.

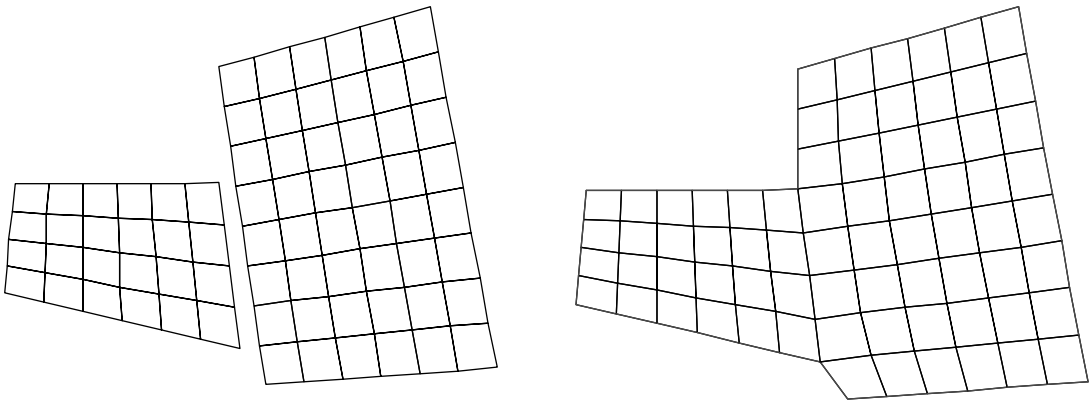
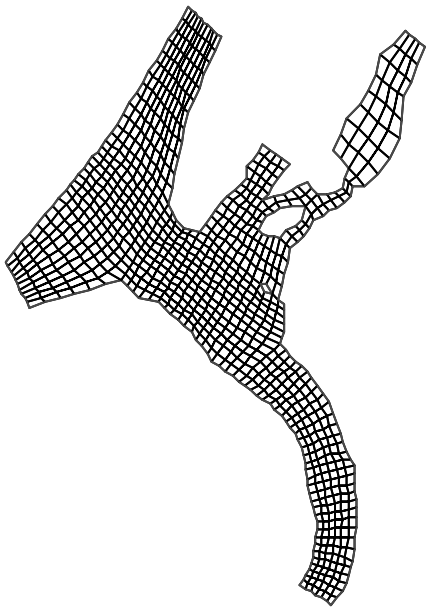


Figure 2.3.1 The left figure shows two structured blocks. These are connected on the right figure to form an unstructured grid.

Using several blocks, a relatively complex geometry can be made. An example is given below:

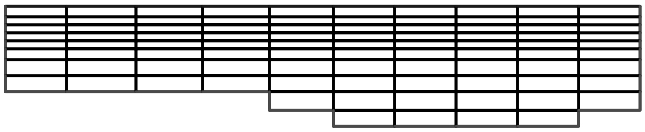
Figure 2.3.2. Grid of Lake Tyrifjorden, seen from above. The grid is made of eight blocks, and two islands are modelled in the grid. The grid was used to model dispersion of a radioactive pollutant by wind-induced currents. (Olsen and Tjomsland, 1998)



2.4 Vertical distribution of grid cells for an unstructured grid

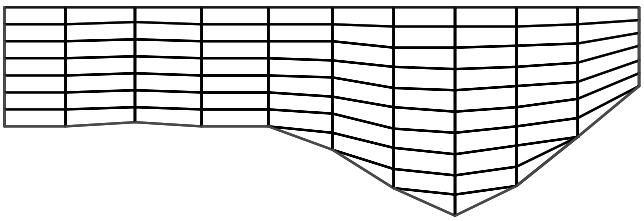
The 3D grid is based on extending the 2D depth-averaged grid with more cells in the vertical direction. Different algorithms can be used to distribute the cells, depending on the purpose of the CFD computation. If the flow field is influenced by thermal stratification, instabilities can occur if the horizontally neigbouring grid cells are not on the same level. Looking at a cross-section of the grid, it must therefore be generated orthogonally:

Figure 2.4.1 Orthogonal unstructured grid



If there is no thermal stratification, it is possible to use non-orthogonal grids in a cross-section. Triangular cells can also be used to model a complex bed geometry, for example like this:

Figure 2.4.2 Non-orthogonal unstructured grid, with triangular (tetrahedral in 3D) cells close to the bed



The grid is made up of cells with different number of sides. Figure below shows four types.

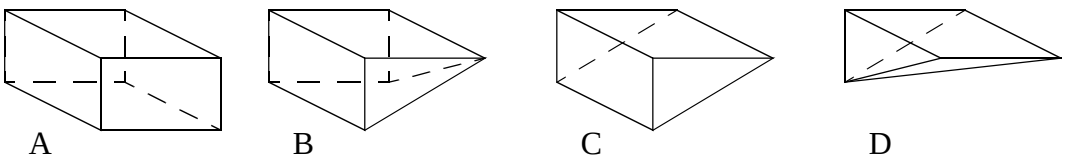


Figure 2.4.3 Different cell shapes, depending on the number of sides

A CFD model is often used for calculating sediment transport. The most important processes then takes place close to the bed. To get as good connection between the bed

cells as possible, it is preferred to use hexahedral cells close to the bed. Such a grid is given below:

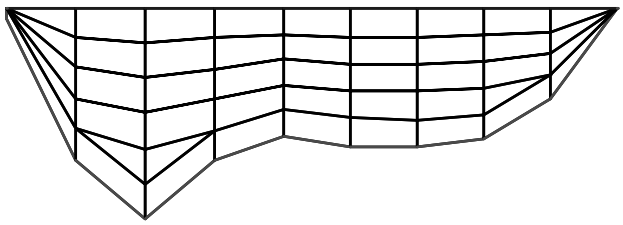


Figure 2.4.4 Unstructured non-orthogonal grid with quadrilateral (hexahedral in 3D) cells close to the bed

2.5 Transient grid changes

In many hydraulic situations, the water depth change due to varying water levels or sediment deposition or erosion. The number of grid cells in the vertical direction may then change, if an unstructured grid is used. Olsen (1999) used such a grid to model reservoir flushing. Then the grid was regenerated every 10th iterations. The values from the old grid was transferred the new grid before starting the calculation in the new time step. The estimate was based on linear extrapolation or interpolation in the vertical direction.

An example of such grid changes are given in Fig. 2.5.1:

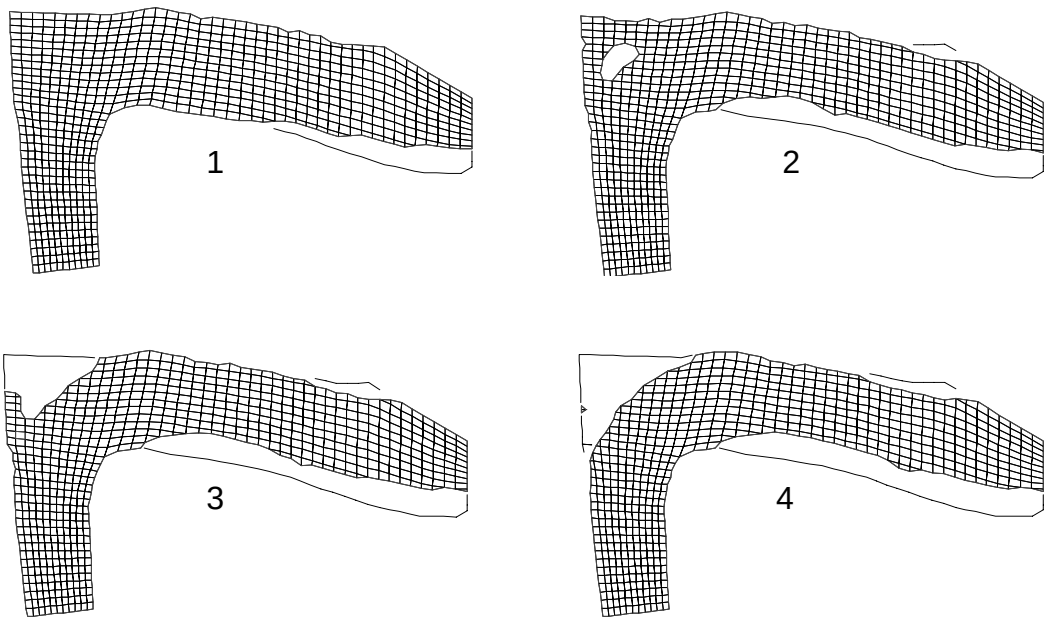


Figure 2.5.1. Grid seen from above for computation of reservoir flushing. The grid is shown for four different times.

Inflow/outflow locations

In an unstructured grid, it is difficult for the user to give the cell numbers when specifying the location of inflow and outflow of the geometry. Therefore, this is often done graphically, where the user interface of the CFD program calculates the cell numbers of the inflow and outflow cells. The cell numbers are stored in arrays, being used when giving the boundary conditions in the CFD model.

When the unstructured grid changes over time, the cell numbers are reassigned. Special algorithms are therefore needed to calculated the cell numbers of the inflow and outflow boundary. This can be based on fixed coordinates of the location of the initial boundary. Search algorithms are used for the boundary cells, to find the matching cell walls.

2.6 Changes in grid cell shapes

During changes in the bed and water levels, the horizontal location of the boundary of a geometry may change. If the grid cells have a fixed horizontal location, some grid cells may be removed or added, as described in the previous chapter. One problem may then be the edges of a grid boundary, where the side would otherwise be smooth.

One way to reduce this problem is to change the shape of the grid cells at the boundary. This is done according to the calculated water depths at the corners of the cell. The grid cell is deformed by moving a cell corner where the bed is above the water level. The movement of this “dry” cell corner can be divided in two cases:

- Movement of a free corner
- Movement of a boundary line.

The movement of a boundary line is described in Fig. 2.6.1.

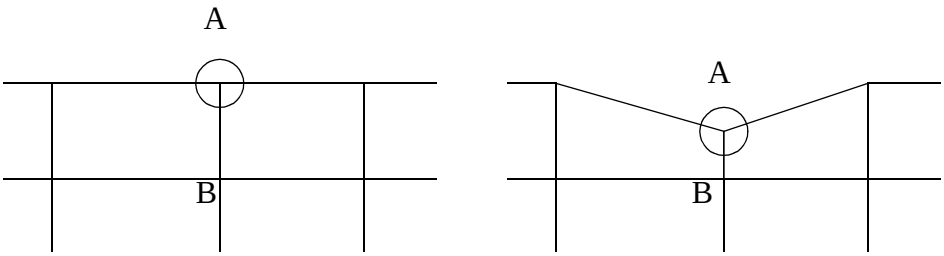


Figure 2.6.1 Movement of grid sides

The dry point A is moved in direction of the submerged point B. The magnitude, δ, of the movement is calculated from Eq. 3:

$$\delta = \delta_0 \frac{\Delta_A}{(\Delta_A + \Delta_B)} \qquad (2.6.1)$$

The original distance between A and B is denoted δ_0, and Δ_A and Δ_B is the vertical distance between the water surface and the bed at point A and B.

For a corner, the algorithms become slightly more complex, but the same principle is used. The algorithms are based on the location of the border and the depth of the corner of each cell. Fig. 2.6.2 shows a dry corner, A, and a two adjacent wet corners B and C.

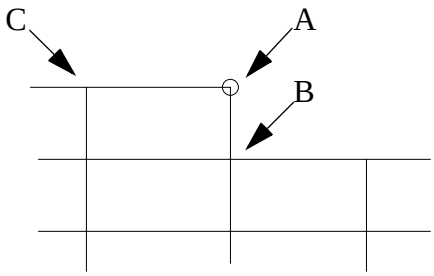


Figure 2.6.2 Movement of grid corner A

The corner point A is moved towards B or C, and Eq. 2.6.1 is applied similarly as for a side point. The choice of moving towards B or C is made from the vertical distance, Δ, between the bed and the water surface of the two points.

The horizontal grid cell deformation algorithm could potentially cause problems if the cells became very small, leading to high expansion ratios in the grid. Instabilities and unphysical results could follow. The problem is prevented by not allowing δ to become smaller than $0.5 * \delta_0$ in Eq. 2.6.1.

2.7 Nested grids

Many hydraulic phenomena involves processes at different scales. One example is pollution from point-source into a lake. The lake may have dimensions of kilometers, while the plume of the pollution may be in the orders of meters wide. The grid of the lake may have cell sizes of the order of hundreds of meters, and can not resolve the concentration profile of the pollution. A solution is then to use a nested grid. A coarse grid covers the lake, while a fine grid is located inside the coarse grid and only covers the pollution plume. Interpolation functions transfer informations like the velocity field and turbulence from the coarse grid to the fine grid.

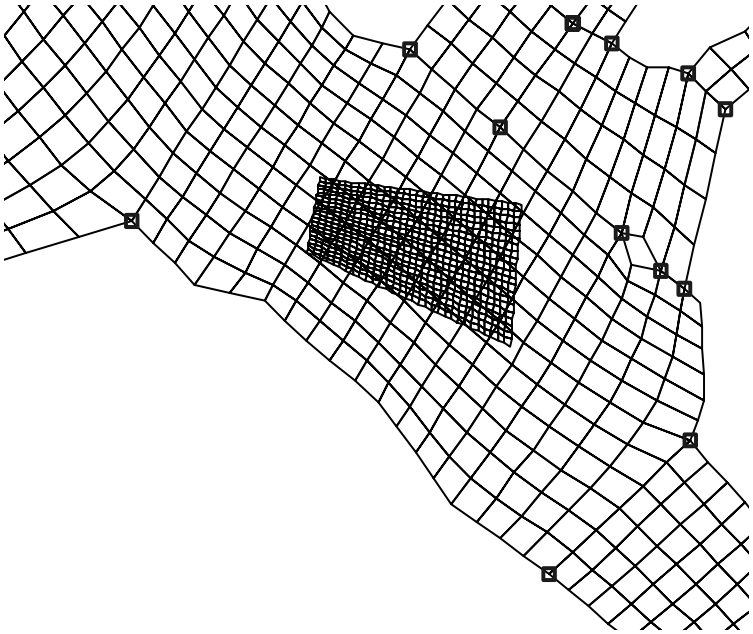


Figure 2.5.1. Nested grid inside the coarse grid of Lake Tyrifjorden (Fig. 2.3.1). The nested grid is located in the area where the pollutant will disperse.

Other examples where nested grids can be used are local scour in a wide river, where the fine grid covers the embankment or bridge pier area, and the coarse grid covers the whole river. The nested approach can also be used modelling spillways, where the spillway crest is modelled with a fine grid. The coarse grid covers the reservoir/lake, estimating the inflow boundary condition for the spillway.

The nested grid can be generated in two ways, depending on the alignment of the grid lines:

- The coarse grid cells can be divided in smaller sizes
- An arbitrarily aligned grid can be used

The advantage of the first approach is better stability and coupling between the grids. The advantage with the second approach is a more flexible layout of the nested grid. Also, it is possible to make the nested grid move horizontally in the coarse grid as a function of time or a computed variable.

The main concern with the second approach is to interpolate the values from the coarse grid to the fine grid for use as boundary condition. An upwind method can be used. The velocity on the boundary of the nested grid is first interpolated from the surrounding cells in the coarse grid. Then the velocity vector is used to find the cell in the upstream direction.

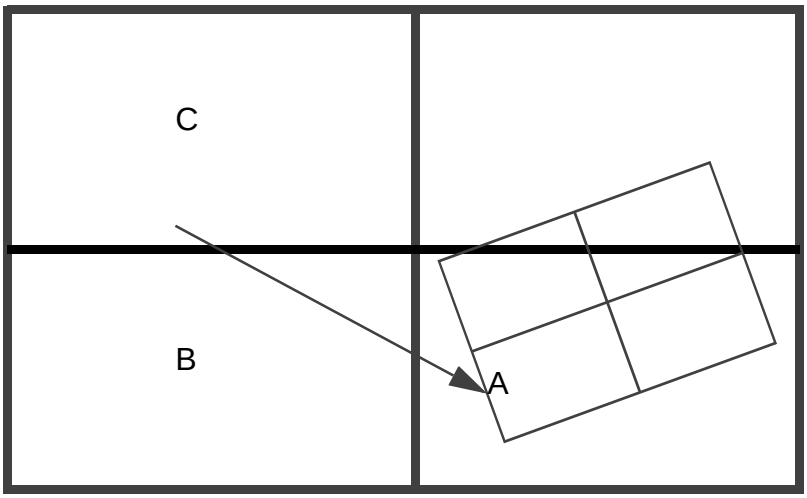


Figure 2.7.2 Upwind interpolation to the surface A in the nested grid (thin lines) from the coarse grid (thick lines). The vector is the velocity in point A. The values on the boundary in point A is taken from the coarse cells B and C, as a function of how close the velocity vector is to the points.

3. The convection-diffusion equation

3.1 Introduction

The movement and dispersion of suspended sediments, temperature, a pollutant etc. in a water body is described by its convection-diffusion equation. The equation is in general the same for almost all water quality parameters, and its solution procedure is therefore described separately in this chapter.

The convection-diffusion equation for steady sediment transport is:

$$U_j \frac{\partial c}{\partial x_j} + w \frac{\partial c}{\partial z} = \frac{\partial}{\partial x_j}\left(\Gamma_T \frac{\partial c}{\partial x_j}\right) \qquad (3.1.1)$$

The sediment concentration is denoted *c, w* is the fall velocity of the particles, *U* is the water velocity, *x* is a space dimension and G is the turbulent diffusivity. The three directions are x_1, x_2 and x_3, and the velocities in the three directions are U_1, U_2 and U_3.

The Einstein summation convention/tensor notation is used, meaning repeated indexes are summed over all directions. For three-dimensional flow this means that the equation can be written:

$$U\frac{\partial c}{\partial x} + V\frac{\partial c}{\partial y} + W\frac{\partial c}{\partial z} + w\frac{\partial c}{\partial z} = \frac{\partial}{\partial x}\left(\Gamma_T \frac{\partial c}{\partial x}\right) + \frac{\partial}{\partial y}\left(\Gamma_T \frac{\partial c}{\partial y}\right) + \frac{\partial}{\partial z}\left(\Gamma_T \frac{\partial c}{\partial z}\right) \qquad (3.1.2)$$

Here, *x*, *y* and *z* are used in the tree directions instead of x_1, x_2 and x_3, and *U, V* and *W* are used instead of U_1, U_2 and U_3.

Transport processes

There are two main transport processes: convection and diffusion. The convection is a movement by the average water velocity. The transport because of the fall velocity of the sediment particles is also a type of convective transport. When calculating the flux, F, through a given surface with area A, the following formula is used:

$$F = c * U * A \qquad (3.1.3)$$

U is the average velocity of the sediments normal to the surface, and c is the average sediment concentration over the area. The sediment velocity will be the sum of the water velocity and the sediment fall velocity.

As an example, we can look at a uniform flow, with zero vertical water velocity. If the surface is vertical, the sediment fall velocity component is zero normal to the surface. Then the velocity *U* will be equal to the horizontal water velocity. If the surface is parallel to the bed/water surface, then the water velocity component normal to the surface will be zero. The velocity *U* in Formula 3.1.1 will then be equal to the fall velocity of the sediment particles.

The other process is the turbulent diffusion of sediments. This is due to turbulent mixing and concentration gradients. The turbulent mixing process is usually modelled with a turbulence mixing coefficient, Γ, defined as the sediment flux divided by the concentration gradient:

$$\Gamma = \frac{\left(\frac{F}{A}\right)}{\left(\frac{dc}{dx}\right)} \qquad (3.1.4)$$

Normally, the convective transport will be dominating. But in some cases, the diffusive transport is important. An example is the reduced settling in a sand trap because of turbulence.

The processes and Formulas 3.1.1 and 3.1.2 will be used extensively in the following description of the finite volume method.

3.2 The First-Order Upstream Scheme

The discretization described here is by the control volume method.

The main point of the discretization is:

Discretization is:

To transform the partial differential equation into a new equation where the variable in one cell is a function of the variable in the neighbour cells

The new function can be thought of as a weighted average of the concentration in the neighbouring cells. For a two-dimensional situation, the following notation is used, according to directions north, south, east and west:

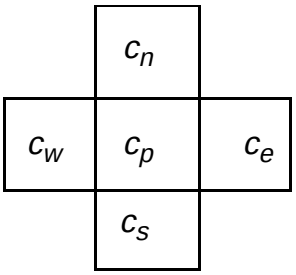


Figure 3.2.1 Discretization molecule. Computation of concentration, c, in the center cell, p, as a function of the concentration in the neighbouring cells n, s, e and w.

:
a_e : weighting factor for *cell e*
a_w : weighting factor for *cell w*
a_n : weighting factor for *cell n*
a_s : weighting factor for *cell s*
$a_p = a_e+a_w+a_n+a_s$

The formula becomes:

$$c_p = \frac{a_w c_w + a_e c_e + a_n c_n + a_s c_s}{a_p} \qquad (3.2.1)$$

The weighting factors for the neighbouring cells a_e, a_w, a_n and a_s are often denoted a_{nb}

What we want to obtain are formulas for a_{nb}.

There are a number of different discretization methods available for the control-volume approach. The difference is in *how the concentration on a cell surface is calculated*. Some methods are described in the following.

In a three-dimensional computation, the same principles are involved. But two more neighbouring cells are added: t (top) and b (bottom), resulting in six neighbour cells. The simple extension from 2D to 3D is one of the main advantages of the finite volume method.

Development of CFD algorithms was initially done in aeronautics. The fluid was air, and the methods were then called upwind instead of upstream. Both expressions are used, meaning the same.

For a non-staggered grid, the values of the variables are given in the center of the cells. Using the finite volume method, it is necessary to estimate variable values on the cell surfaces. The main idea of the upstream methods is to estimate the surface value from the *upstream* cell. The first order method uses information in only one cell upstream of the cell surface. In other words: the concentration at a cell surface for the first-order upstream method is the same as the concentration in the cell on the upstream side of the cell side.

The control volume method is based on continuity of sediments. The basis of the calculation is the fluxes on a cell surface. The surface area is denoted *A*; the velocity at the surface, normal to it, is denoted *U*; c is the concentration at the surface, and Γ is the turbulent diffusion at the surface.

The convective flux is calculated as: U * *A* * *c* (3.2.2)
The diffusive flux is calculated as: Γ * *A* * *dc* / *dx*(3.2.3)

The term *dc/dx* is calculated as the concentration difference between the cells on each side of the surface, divided by the distance between the centres of the cells. Looking at

the west side of cell *p*, Fig. 8.3.1 explains the variable locations and the fluxes in the center cell *P*.

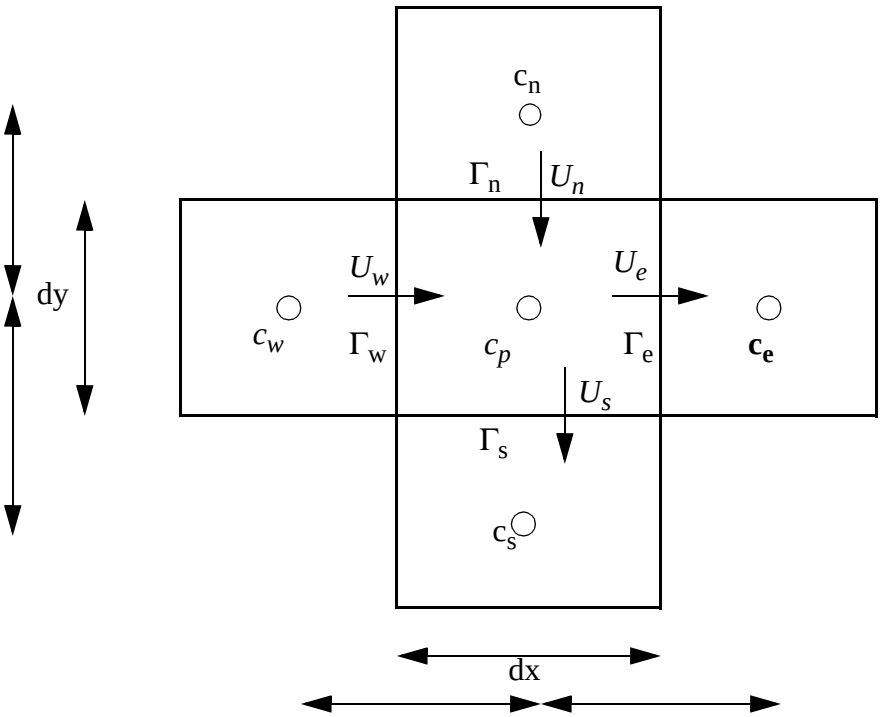


Figure 3.2.2 Fluxes through the walls of the center cell in a computational molecule. The cells have a width dx *and a height* dy.

The flux, F_w, through the west side of cell *P* then becomes:

$$F_w = U_w A_w c_w + \Gamma_w \frac{A_w (c_w - c_p)}{dx} \qquad (3.2.4)$$

where A_w is the area of the cell wall on the west side, equal to Δy times the height of the wall. For the other sides, the following fluxes are obtained:

$$F_e = U_e A_e c_p + \Gamma_e \frac{A_e (c_p - c_e)}{dx} \qquad (3.2.5)$$

$$F_s = U_s A_s c_p + \Gamma_s \frac{A_s (c_p - c_s)}{dy} \qquad (3.2.6)$$

$$F_n = U_n A_n c_n + \Gamma_n \frac{A_n (c_n - c_p)}{dy} \qquad (3.2.7)$$

Sediment continuity means the sum of the fluxes is zero, in other words:

$$F_w - F_e + F_n - F_s = 0 \qquad (3.2.8)$$

This gives the following equation:

$$\left(U_w A_w + U_e A_e + \Gamma_e \frac{A_e}{dx} + U_s A_s + \Gamma_s \frac{A_s}{dy} + \Gamma_n \frac{A_n}{dy}\right) c_p = \left(U_w A_w + \Gamma_w \frac{A_w}{dx}\right) c_w + \left(\Gamma_e \frac{A_e}{dx}\right) c_e + \left(U_n A_n + \Gamma_n \frac{A_n}{dy}\right) c_n + \left(\Gamma_s \frac{A_s}{dy}\right) c_s \qquad (3.2.8)$$

When we compare Equation 3.2.1 with Equation 3.2.8, we see they are the same. The concentration in Cell *P* is a function of the concentration in the neighbouring cells. The resulting weighting factors are:

$$a_p = \Gamma_w \frac{A_w}{dx} + U_e A_e + \Gamma_e \frac{A_e}{dx} + U_s A_s + \Gamma_s \frac{A_s}{dy} + \Gamma_n \frac{A_n}{dy} \quad (3.2.9)$$

$$a_w = U_w A_w + \Gamma_w \frac{A_w}{dx} \quad (3.2.10)$$

$$a_e = \Gamma_e \frac{A_e}{dx} \quad (3.2.11)$$

$$a_s = \Gamma_s \frac{A_s}{dy} \quad (3.2.12)$$

$$a_n = U_n A_n + \Gamma_n \frac{A_n}{dy} \quad (3.2.13)$$

The water continuity equation for the grid cell is:

$$U_e A_e - U_w A_w + U_n A_n - U_s A_s = 0 \quad (3.2.14)$$

or:

$$U_e A_e + U_n A_n = U_w A_w + U_s A_s \quad (3.2.15)$$

If the above equation is inserted into the expression for a_p, the equation

$$a_p = a_e + a_w + a_s + a_n \quad (3.2.16)$$

is verified to be correct.

Note that the equations above are only valid if the velocity flows in the same direction as given on the arrows in the figure above.

3.3 The Second Order Upstream Scheme

The Second-Order Upstream (SOU) method is based on a second-order accurate method to calculate the concentration on the cell surfaces. The method only involves the convective fluxes, and the diffusive terms are calculated as before. The following figure shows the calculation of the concentration on the west side of cell *p*: side *W*:

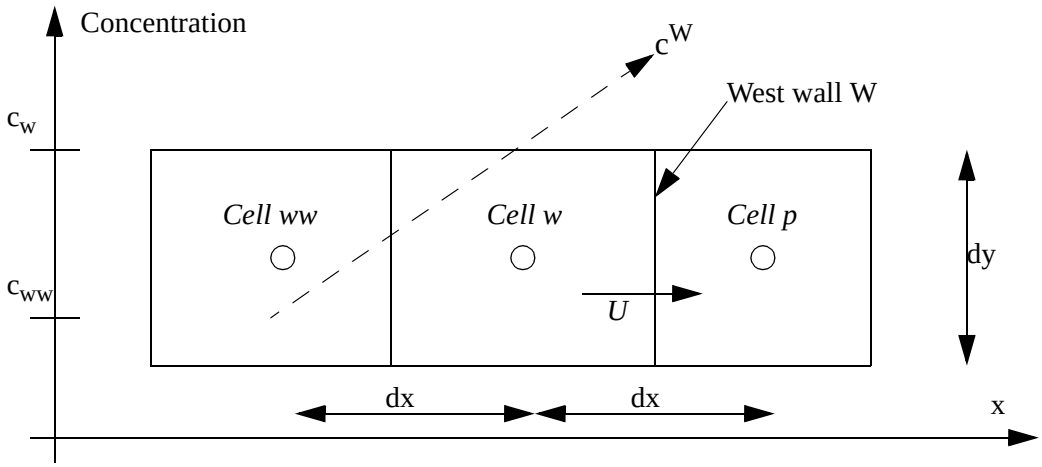


Figure 3.3.1 Definition sketch for concentration estimation at the wall for the SOU scheme.

The cell on the west side of cell *w* is called cell *ww*. The concentration in this cell is denoted c_{ww}. The concentration in cell *w* is denoted c_w and the concentration on side *W* of cell *p* is denoted c^W. The SOU scheme uses the concentration in cell *ww* and cell *w* to extrapolate linearly to side *W*. Given the width of the cell in the x-direction is *dx*, and the height in the y-direction is *dy*, it is possible to derive a formula for the concentration on side *W* by triangulation:

$$\frac{c^w - c_{ww}}{dx + 0.5dx} = \frac{c_w - c_{ww}}{dx} \qquad (3.3.1)$$

or

$$c^w = \frac{3}{2}c_w - \frac{1}{2}c_{ww} \qquad (3.3.2)$$

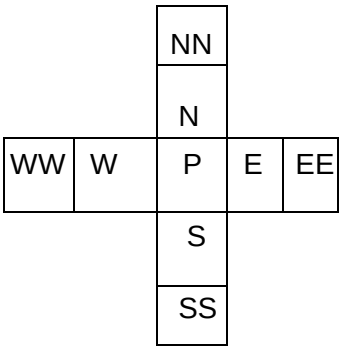


Figure 3.3.3 SOU nine-point calculation molecule

Equation 8.6.1 is only valid if the cells are of equal size. If the expansion ratio is different from unity, a separate formula needs to be applied, where the coefficients 3/2 and ½ are given as a function of the expansion ratio.

The calculation molecule now gets nine cells, as shown in the figure to the left

The flux through the west side of Cell *P* then becomes:

$$F_w = U_w A_w\left(\frac{3}{2}c_w - \frac{1}{2}c_{ww}\right) + \Gamma_w \frac{A_w(c_w - c_p)}{dx} \qquad (3.3.3)$$

For the other sides, the following fluxes are obtained:

$$F_e = U_e A_e\left(\frac{3}{2}c_p - \frac{1}{2}c_w\right) + \Gamma_e \frac{A_e(c_p - c_e)}{dx} \qquad (3.3.4)$$

$$F_s = U_s A_s\left(\frac{3}{2}c_p - \frac{1}{2}c_n\right) + \Gamma_s \frac{A_s(c_p - c_s)}{dy} \qquad (3.3.5)$$

$$F_n = U_n A_n\left(\frac{3}{2}c_n - \frac{1}{2}c_{nn}\right) + \Gamma_n \frac{A_n(c_n - c_p)}{dy} \qquad (3.3.6)$$

Again, the equations are only valid if the velocity vectors are in the same direction as in Fig. 8.3.1. The weighting factors becomes:

$$a_w = \frac{3}{2}U_w A_w + \Gamma_w \frac{A_w}{dx} + \frac{1}{2}U_e A_e \quad (3.3.7)$$

$$a_{ww} = -\frac{1}{2}U_w A_w \quad (3.3.8)$$

$$a_e = \Gamma_e \frac{A_e}{dx} \quad (3.3.9)$$

$$a_{ee} = 0 \quad (3.3.10)$$

$$a_n = \frac{3}{2}U_n A_n + \Gamma_n \frac{A_n}{dy} + \frac{1}{2}U_s A_s \quad (3.3.11)$$

$$a_{nn} = -\frac{1}{2}U_n A_n \quad (3.3.12)$$

$$a_s = \Gamma_s \frac{A_s}{dy} \quad (3.3.13)$$

$$a_{ss} = 0 \quad (3.3.14)$$

For the SOU scheme, Equation 3.3.1 now becomes:

$$c_p = \frac{a_w c_w + a_e c_e + a_n c_n + a_s c_s + a_{ww} c_{ww} + a_{nn} c_{nn}}{a_p} \quad (3.3.15)$$

The formula is used for a two-dimensional situation. In 3D, the terms for top and bottom is also added, giving four extra coefficients: a_t, a_{tt}, a_b, a_{bb}.

4. The Navier-Stokes equations

4.1 Introduction

The Navier-Stokes Equations describe the velocity and pressure fields in a water body. The equations were developed for laminar flow, but using the Reynolds-averaging and a turbulence model, the equations also describe turbulent flow.

The CFD Class Notes (Olsen, 1999) only gave an overview of some of the general-purpose algorithms used in CFD. The purpose of this chapter is to extend this with more in-depth details, especially with respect to numerical implementation of the Navier-Stokes equations:

$$\underset{1}{\frac{\partial U_i}{\partial t}} + \underset{2}{U_j \frac{\partial U_i}{\partial x_j}} = \frac{1}{\rho}\frac{\partial}{\partial x_j}\Big[\underbrace{-\left(P + \frac{2}{3}k\right)\delta_{ij}}_{3\ 4} + \underbrace{\nu_T\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)}_{5}\Big] \quad 6$$

(4.1.1)

The integers below the equation indicates the six terms. The first term is the transient term. Its discretization is given in Chapter 3.2. The convective term 2, and the diffusive term 5 is discretized similarly as for the convection-diffusion equation. The pressure term 3 and term 4 is usually solved together, as one unknown variable. Also, Term 4 is usually several orders of magnitude lower than the pressure, so it would not be a problem to neglect it altogether. The discretization of Term 6 is given in Chapter 4.8.

Basic discretization methods are given in Chapter 4.2 and 4.3.

4.2 Discretization by coordinate transformation

The control-volume method offers two methods to discretize equations. One method is to look at physical processes and parameters, and thereby derive the discretized equations. This approach is used by Olsen (1999). The method has the advantage of being easy to understand, facilitating debugging and implementation of new algorithms. However, three groups of terms are difficult to derive using a physically based method:

- The production of turbulence in the k-ε model.
- The stress terms in the Navier-Stokes equations
- The non-orthogonal diffusive terms in the Navier-Stokes equations

The terms are therefore usually discretized using a method called coordinate transformation. The basis of the method is given in the following.

The coordinate transformation is a transformation between the natural cartesian coordinate system and a system following the computational domain. Fig. 4.2.1 shows the two systems in 2D. Additionally, the third direction will be *z* in the cartesian system and ζ in the computational domain.

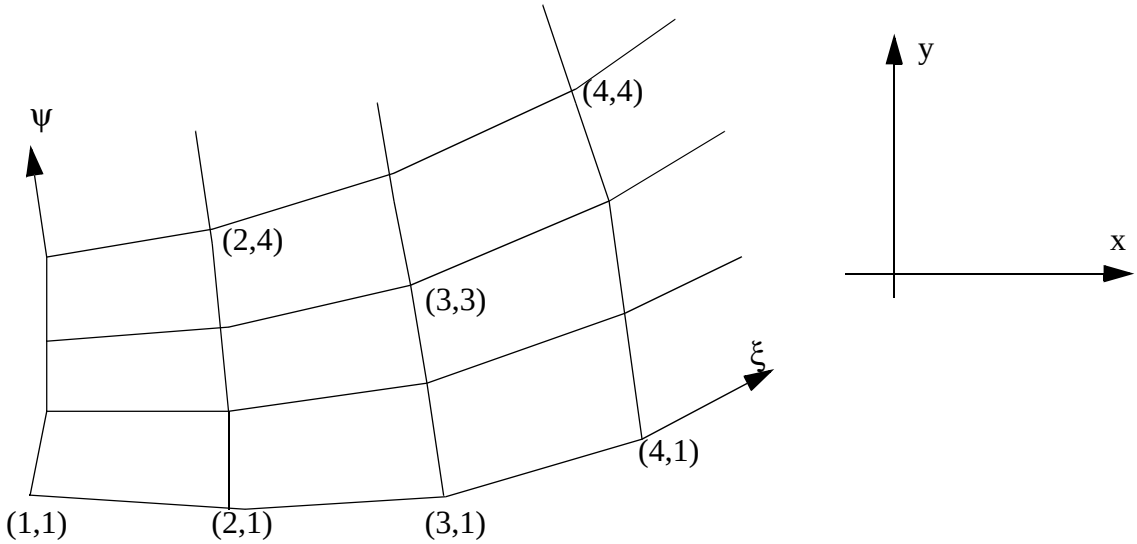


Figure 4.2.1 *The two coordinate systems are shown in two dimensions, where some values of ξ and ψ are given in brackets*

The three directions along the computational domain are often called (ξ,ψ) in 2D as shown in the figure above, or (ξ,ψ,ζ) in 3D, where the last index is the vertical direction. In the CD system, the distance between the grid lines are often set to unity, so it is easy to calculate gradients of variables. It means all δξ will be unity.

An important definition is the notation of the variables at a cell. Instead of using *x*,*y* and *z* directions, the non-orthogonal cell now uses the directions north, south, east, west, bottom and top. Another definition is to use indexes, as in

tensor notation. Then direction 1 is east-west, direction 2 is north-south and direction 3 is vertical. Using tensor notation, (ξ,ψ,ζ) can also be written (ξ^1,ξ^2,ξ^3).

Example:

A^1 is the area of the cell wall in the east-west direction

The surface of a cell side is denoted *A*. It may have two indexes. An upper index indicates which side it is, as given above.

It is also possible to define areas through the center of the cell. These can be though of as averages of the two side walls.

Examples:

$A^1{}_2$: The area component in the y-direction of the cell wall in the east-west direction.

$A^3{}_1$: The area component in the x-direction of the bottom/top cell wall.

A may also have a lower index, the area component in *x*,*y* or *z* direction. It is obtained by multiplying *A* with the component of the vector normal to the surface.

A useful formula for the discretization is the chain rule. Transforming the terms from one coordinate system to another, the following formula is used:

$$\frac{\partial U_i}{\partial x_k} = \frac{\partial U_i}{\partial \xi^j}\frac{\partial \xi^j}{\partial x_k} \qquad (4.2.1)$$

This can be transformed as follows

$$\frac{\partial U_i}{\partial x_k} = \frac{\partial U_i}{\partial \xi^j}\frac{\partial \xi^j}{\partial x_k}\frac{V}{V} = \frac{1}{V}\left(\frac{\partial U_i}{\partial \xi^j}\right)\left(\frac{\partial \xi^j}{\partial x_k}V\right) = \frac{\partial U_i}{\partial \xi^j}\frac{1}{V}A^j{}_k = \frac{1}{V}\omega^i{}_k \qquad (4.2.2)$$

V is the volume of the cell. The second step in Eq. 4.2.2 isolates the velocity gradients from the geometric terms. The third step can be derived using the Jacobi Matrix, and is further described by Melaaen (1990). From a physical point of view it seems logical, as the formula for the volume of a cell can be calculated as the area, *A*, times the distance Δx normal to the area. The fourth step defines ω, a shorter way of writing the term, which will be used later. Note that summing up over *j* gives three terms:

$$\omega^i{}_k = \frac{\partial U_i}{\partial \xi^j}A^j{}_k = \frac{\partial U_i}{\partial \xi}A^\xi{}_k + \frac{\partial U_i}{\partial \psi}A^\psi{}_k + \frac{\partial U_i}{\partial \zeta}A^\zeta{}_k \qquad (4.2.3)$$

Also note $\delta\xi = \delta\psi = \delta\zeta = 1$ can be chosen.

4.3 Discretization of source terms

The source terms of the convection-diffusion equation may be fluxes of sediments flowing into the geometry. For the Navier-Stokes equations, the sources are forces on a cell, for example gravity or body forces from obstacles in the flow. There are different ways to discretize the source terms, giving more or less stable solutions.

The convection-diffusion equation for sediment concentration is derived from continuity. The source term from adding sediments is therefore the sediment flux into the cell:

$$a_p c_p = \sum_{nb} a_{nb} c_{nb} + Flux_p . \qquad (4.3.1)$$

The Navier-Stokes equations are based on Newton's 2nd law, and the source terms are forces on a cell:

$$a_p U_p = \sum_{nb} a_{nb} U_{nb} + Force_p \qquad (4.3.2)$$

If the force is a pressure difference, the term could for example be discretized as:

$$Force_p = (P_e - P_w)\left(\frac{A_e + A_w}{2}\right) \qquad (4.3.3)$$

When the force is a function of the velocity, there are different ways of discretizing it. As an example, the force from a cylinder on the flow is considered:

$$Force = \frac{1}{2} C_D A \rho U^2 \qquad (4.3.4)$$

C_D is the drag coefficient, ρ is the water density and A is the cross-sectional area of the body, normal to the direction of the flow.

Incorporating the formula in the equation gives:

$$a_p U_p = \sum_{nb} a_{nb} U_{nb} - \frac{1}{2} C_D A \rho U_p^{\,2} \qquad (4.3.5)$$

The problem in evaluating the source term is that U_p is unknown. A possible solution would be to use U_p at the previous iteration. This could work, if the forces are small. If the forces are large, the approach could lead to instabilities. The alternative is to move the term to the other side of Equation 4.3.2:

$$\left(a_p + \frac{1}{2} C_D A \rho U_p\right) U_p = \sum_{nb} a_{nb} U_{nb} \qquad (4.3.6)$$

or

$$U_p = \frac{\sum_{nb} a_{nb} U_{nb}}{\left(a_p + \frac{1}{2} C_D A \rho |U_p|\right)} \qquad (4.3.7)$$

This formula is much more stable, as the source term will increase the value of the a_p coefficient. The two approaches are two solutions to the problem of linearize the source terms, meaning in practice to keep the source term as it is or put it into the a_p term. It is also possible to keep some in the source term and put the rest into a_p. For stability reasons, it is better to choose the option that keep the a_p term as large as possible.

A further discussion of the source term discretization is given by Patankar (1980).

4.4 The SIMPLE method

SIMPLE is an acronym for Semi-Implicit Method for Pressure-Linked Equations. The original description was given by Patankar and Spalding (1972). Patankar (1980) gives a very readable explanation of the method. The SIMPLE method and its extensions are used in most CFD calculations in the world today.

The main concept of the method to guess a pressure field and calculate the velocities. Then estimate the continuity defect, and use this to calculate a correction of the pressure field. To derive the equations for the pressure-correction, a special notation is used. The initially calculated variables do not satisfy continuity and are denoted with an index *. The correction of the variables is denoted with an index ‘. The variables after correction do not have a superscript. The process can then be written:

$$P = P^* + P' \qquad (4.4.1)$$

$$U = U^* + U' \qquad (4.4.2)$$

P is the pressure and U is the velocity. The index k on the velocity denotes direction, and runs from 1 to 3 for a 3D

calculation.

Given guessed values for the pressure, the discretized version of the Navier-Stokes equations is:

$$a_p U^*_p = \sum_{nb} a_{nb} U^*_{nb} + B_{u_k} - \left(A_k^j \frac{\partial P^*}{\partial \xi} \right) \qquad (4.4.3)$$

The convective and diffusive terms have been discretized as described in Chapter 3. The variable *B* contains the rest of the terms besides the convective term, the diffusive term and the pressure term. In the pressure term, *A* is the surface area on the cell wall, and ξ is an index for the non-orthogonal coordinate system, described in Chapter 2.1.

The discretized version of the Navier-Stokes equations based on the corrected variables can be written as:

$$a_p U_p = \sum_{nb} a_{nb} U_{nb} + B_{u_k} - \left(A_k^j \frac{\partial P}{\partial \xi} \right) \qquad (4.4.4)$$

If this equation is subtracted from Equation 4.2.3, the following equation emerges

$$: \quad a_p U_p - a_p U^*_p = \sum_{nb} a_{nb} U_{nb} - \sum_{nb} a_{nb} U^*_{nb} - \left(A_k^j \frac{\partial P}{\partial \xi} - A_k^j \frac{\partial P^*}{\partial \xi} \right) \qquad (4.4.5)$$

Using Eq. 4.4.1 and Eq. 4.4.2, this equation can be rewritten:

$$a_p U'_p = \sum_{nb} a_{nb} U'_{nb} - A_k^j \frac{\partial P'}{\partial \xi} \qquad (4.4.6)$$

In the SIMPLE method, the first term on the right side of the equation is omitted. The following formula for the velocity correction is then obtained:

$$U_{k_p}' = \left(\frac{A_k^j}{a_p} \frac{\partial P'}{\partial \xi^j} \right) \qquad (4.4.7)$$

The SIMPLEC method approximates the first term on the right hand side of Eq. 4.4.6 as:

$$\sum_{nb} a_{nb} U'_{nb} = \sum_{nb} a_{nb} U'_{k_p} \qquad (4.4.8)$$

This gives the following formula for the velocity correction:

$$U_{k_p}{}' = \left(\frac{A_k^j}{\left(a_p - \sum_{nb} a_{nb} \right)} \frac{\partial P'}{\partial \xi^j} \right) \qquad (4.4.9)$$

The SIMPLEC method uses the formula above. The above equations give the velocity-corrections once the pressure-corrections are known. To obtain the pressure-corrections, the continuity equation is used for the velocity correction for a cell:

$$\sum_{nb} A_k U_k = 0 \qquad \text{(k=1,2,3)} \qquad (4.4.10)$$

or

$$\sum_{nb} A_k U^*{}_k + \sum_{nb} A_k U'_k = 0 \qquad (4.4.11)$$

The first term is the water continuity defect, *V*, given the guessed velocities. The second term is solved using Eq. 4.4.8 for the velocity correction:

$$\sum_{nb} A_k \left(\frac{A_k^j}{a_p} \frac{\partial P'}{\partial \xi^j} \right)_k = -V \qquad (4.4.12)$$

Given for example a quadrilateral cell with sides east, west, south and north, the left term in Eq. 4.4.12 is discretized as:

$$\frac{A_w^2}{a_p \Delta \xi_w}(P'_P - P'_W) + \frac{A_S^2}{a_p \Delta \xi_S}(P'_P - P'_S) = \frac{A_E^2}{a_p \Delta \xi_E}(P'_E - P'_P) + \frac{A_N^2}{a_p \Delta \xi_N}(P'_N - P'_P) - V \qquad (4.4.13)$$

The result is an equation of the following form:

$$a^\circ{}_p P'_p = \sum_{nb} a^\circ{}_{nb} P'_{nb} + b \qquad (4.4.14)$$

The following formula is given for $a^o{}_e$:

$$a^\circ{}_e = \frac{\rho A_e^2}{a_{p,e}} \qquad (4.4.15)$$

A similar equation holds for the other $a^o{}_{nb}$ coefficients.

The index *e* is then replaced by *w, n, s, t* or b. The $a_{p,e}$ factor is the average a_p value in cell *p* and cell *e*.

The source term, *b*, in Equation 4.4.14 is the water continuity defect for the guessed velocity field. Equation 4.4.14 is solved in the same way as the other equations. The procedure is therefore:

1. Guess a pressure field, *P**
2. Calculate the velocity *U** by solving Equation 4.4.3
3. Solve equation 4.4.14 and obtain the pressure-correction, *P'*
4. Correct the pressure by adding *P'* to *P**
5. Correct the velocities *U** with *U'* using Equation 4.4.7, and also correct the fluxes on the cell walls
6. Iterate from point 2 to convergence

An equation for the pressure is not solved directly, only an equation for the pressure-**correction**. The pressure is obtained by accumulative addition of the pressure-correction values.

The SIMPLE method can give instabilities when calculating the pressure field. Therefore, the pressure-correction is often multiplied with a number below unity before being added to the pressure. The number is a relaxation coefficient. The value 0.2 is often used. The optimum factor depend on the flow situation and can be changed to give better convergence rates. Relaxation coefficients are further described in Chapter 4.5.

Regarding the difference between the SIMPLE and the SIMPLEC method, the SIMPLEC should be more consistent in theory, as a more correct formula is used. Looking at Equations 4.4.7 and 4.4.9, the SIMPLE method will give a lower correction than the SIMPLEC method, as the denominator will be larger. The SIMPLE method will therefore move slower towards convergence than the SIMPLEC method. If there are problems with instabilities, this can be an advantage.

Experience in debugging CFD programs shows that the cause of the instabilities can some times be traced to the flux-corrections in the SIMPLE algorithm. Removing the flux-corrections has on occasion been done successfully (Olsen, 1999), although this lead to some sediment and water continuity defect. Since the flux-corrections decrease the continuity defects, it is important to check water and sediment continuity if the flux-corrections are removed.

The SIMPLER method

The SIMPLER method is an extension of the SIMPLE method. The SIMPLE method usually gives good velocity corrections, but the correction of the pressure is less accurate. This is due to the omission of the term $\sum a_{nb}u_{nb}$. The SIMPLER method keeps the algorithms for computing the velocity-corrections, but uses another algorithm for computing the pressure. Pseudo velocities $\hat{U}$ are introduced:

$$\hat{U} = \frac{\sum a_{nb}u_{nb} + b}{a_p} \qquad (4.4.16)$$

The pseudo velocity is the same as the original velocity, except that the pressure term has been omitted in the equation. The relation between $\hat{U}$ and U is therefore:

$$U = \hat{U} + \frac{A}{a_p}\frac{\Delta P}{\Delta \xi} \qquad (4.4.17)$$

This formula is similar to Eq. 4.4.7, except that U^* has been replaced by $\hat{U}$. The solution is therefore also similar. There are only two differences:

- The source term in the new discretized equation is not the continuity defect based on the regular velocities U. Equation 4.4.16 has to be solved, and water continuity based on $\hat{U}$ has to be computed.

- Solving eq. 4.4.17, the result is the pressure, and not the pressure-correction.

The main algorithm for the SIMPLER method then becomes:

1. Guess the velocity field
2. Compute the coefficients a_{nb} in the momentum equation, and compute the starred velocities $\hat{u}$.
3. Compute the coefficients a^0_{nb} in the pressure equation (4.4.17), and solve it to obtain the pressure. The coefficients will be the same as given in Eq. 4.4.15. The source term will be the water continuity defect based on fluxes computed from $\hat{U}$.
4. Use the computed pressure field as P^* in the SIM PLE method. Calculate the velocity U^* by solving Equation 4.4.3
5. Solve equation 4.4.14 and obtain the pressure-correction, P'
6. Correct the velocities U^* with U' using Equation 4.4.7, and also correct the fluxes on the cell walls
7. Iterate from point 2 to convergence

The SIMPLER method was developed for a staggered grid. For a non-staggered grid, there are additional concerns when computing the water continuity defect based on Û, for example the use of the Rhie and Chow interpolation.

4.5 The Rhie and Chow interpolation

*A **cell-centred scheme** is also called a non-staggered variable location, or a non-staggered grid. All variables are then computed at the center of each cell. Previously, a staggered variable allocation was used, where the pressure was computed between the cells.*

The Rhie and Chow interpolation is used to prevent oscillations when using a cell-centred discretization scheme. Its derivation is given by Rhie and Chow (1983). The method introduces an additional term when calculating the fluxes on a cell surface. The term can be interpreted as fourth-order artificial diffusion. However, the method is consistent, as there are no calibration coefficients involved.

The motivation for introducing the method originates from the discussion of using staggered or non-staggered variable location in a grid. The staggered grid removes some instabilities experienced in non-staggered grids. The Rhie and Chow interpolation is used in connection with estimating the velocities on a grid cell boundary when fluxes are calculated. The method uses the pressure gradients from several grid cells to add an extra term to the fluxes.

The method is derived by looking at the grid given below

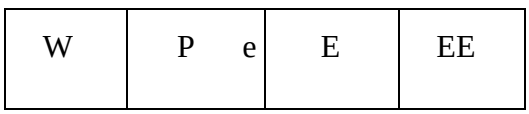


***Fig. 4.5.1** Grid for estimation on the fluxes on surface* e*, between cell* P *and* E

The discretized Navier-Stokes Equations can be written as:

$$U = K + S - \left(\frac{A_j^i \partial P}{a_p \partial \xi^j} \right) \qquad (4.5.1)$$

where *K* is the convective terms, *S* is the source terms and the last term is the pressure term. Calculating the equations in position e, between Cell *P* and Cell *E* in Fig. 4.5.1 will be similar to a staggered grid. Then the pressure is calculated in the center of the grid cell, and the velocities on the walls. Equation 4.5.1 can then be written:

$$U_e = K_e + S_e - \left(\frac{A_j^i \partial P}{a_p \partial \xi^j} \right)_e \tag{4.5.2}$$

If a non-staggered grid is used, Eq. 4.5.1 becomes:

$$U_P = K_P + S_P - \left(\frac{A_j^i \partial P}{a_p \partial \xi^j} \right)_P \tag{4.5.3}$$

$$U_E = K_E + S_E - \left(\frac{A_j^i \partial P}{a_p \partial \xi^j} \right)_E \tag{4.5.4}$$

for Cell *P* and Cell *E* respectively. The notation $\overline{K}_e$ is introduced, to denote a linear interpolation of K_P and K_E to the surface *e*. If the grid cell sizes are equal, the following formula defines the notation:

$$\overline{K_e} = \frac{K_P + K_E}{2} \tag{4.5.5}$$

The derivation of the Rhie and Chow method now assumes $\overline{K}_e = K_e$ and $\overline{S}_e = S_e$. Inserting this into Eq. 4.5.2 gives:

$$U_e = \overline{K}_e + \overline{S}_e - \left(\frac{A_j^i \partial P}{a_p \partial \xi^j} \right)_e \tag{4.5.6}$$

If a linear interpolation is used for all the variables in Eq. 3.4.1 from the center of Cell *P* and Cell *E* to the surface *e*, the following equation emerges:

$$\overline{U_e} = \overline{K_e} + \overline{S_e} - \overline{\left(\frac{A_j^i \partial P}{a_p \partial \xi^j} \right)_e} \tag{4.5.7}$$

Subtracting Eq. 4.5.7 from Eq. 4.5.6 gives the following equation:

$$U_e = \overline{U_e} + \frac{A_e^2}{\overline{a_p}} \left[\overline{\left(\frac{\partial p}{\partial \xi^1} \right)_e} - \left(\frac{\partial p}{\partial \xi^1} \right)_e \right] \tag{4.5.8}$$

This formula could be used for the interpolation. The only problem is that a_p depends on the relaxation factor α for the velocity equations, so the flow field would be slightly different for different values of α. Multiplying the last term with α removes the problem. The final equation becomes:

$$U_e = \overline{U_e} + \alpha \frac{A_e^2}{\overline{a_p}} \left[\overline{\left(\frac{\partial p}{\partial \xi^1} \right)}_e - \left(\frac{\partial p}{\partial \xi^1} \right)_e \right] \qquad (4.5.9)$$

To evaluate the pressure gradients in Eq. 4.5.9, the following equations are used:

$$\left(\frac{\partial p}{\partial \xi^1} \right)_e = p_E - p_P \qquad (4.5.10)$$

$$\overline{\left(\frac{\partial p}{\partial \xi^1} \right)}_e = \frac{(p_W - p_W) + (p_P - p_{EE})}{2} \qquad (4.5.11)$$

The indexes *E*, *W*, *P* and *EE* is defined in Fig. 4.5.1. The same derivation and formula can be derived for the other cell walls, only with changing indexes.

The Rhie and Chow interpolation formula has occasionally given rise to instabilities, when there were large source terms in the equations (Olsen and Kjellesvig, 1998b). But for almost all cases the method will give improved stability.

4.6 Wall laws

Early work on wall laws was carried out by Hermann Schlichting (1979), initially to study air flow around planes. The shape of the velocity profile close to the wall in a boundary layer was found to be universal, and Schlicting's experiments lead to formulas describing the profile. In a CFD program, the formulas are used to describe the velocity profile between the wall and the center of the cell closest to the wall. If wall laws were not used, it would be necessary to use a large number of additional grid cells to resolve the velocity gradients at the wall.

Early CFD models were only applied to situations with smooth walls, and only incorporated such wall laws. However, Schlichting also made wall laws for rough walls. The same formulas are used in hydraulic engineering to describe the vertical velocity profile in a wide channel/river with uniform flow:

$$\frac{U}{U_*} = \frac{1}{\kappa} \ln \left(\frac{30y}{k_s} \right) \qquad (4.6.1)$$

U is the velocity, U_* is the shear velocity, y is the distance from the wall and k_s is the wall roughness, where Schlicht-

ing used spheres glued to a flat plate. Later studies have related the roughness to the bed sediment grain size distribution:

$$k_s = 3d_{90} \tag{4.6.2}$$

where d_{90} is the grain size fraction of the bed where 90 % of the material is smaller.

Schlichting's experiments suggested the wall laws are valid in the range where $y+$ is between 30 and 3000, given as:

$$y^+ = \frac{U_* y}{\nu} \tag{4.6.3}$$

where ν is the kinematic viscosity of water. Schlichting's experiments suggest that the formula can also be used for larger y^+ values, as long as we can assume Eq. 4.6.1 holds between the wall and the center of the cell closest to the wall. For uniform flow in a wide channel, the equation is used all the way to the water surface, so this may be a valid assumption for many cases. On the other hand, if y^+ is very small there may be other problems. The physical interpretation is that the height of the roughness elements exceeds the vertical size of the bed cell. Algorithms with some sort of porosity can then be used (Olsen and Stokseth, 1995; Fischer-Antze et. al., 2001). A further description is given in Chapter 5.2.

Implementing the wall laws in the CFD code, the main idea is to include a sink term for the velocity equations. The shear stress at the wall will be a negative source in the velocity equations. The shear stress is estimated using the wall laws.

The forces on the bed cell is:

$$F = \tau_w A_w \tag{4.6.4}$$

where A_w is the wall area of the bed cell. The shear stress, τ_w, is found by using the wall laws:

$$U_w^+ = \frac{\kappa U_w}{\ln\left(\frac{30\delta n}{k_s}\right)} \tag{4.6.5}$$

$$\tau_w = \rho (U^+)^2 \tag{4.6.6}$$

It would have been possible to insert Eq. 4.6.3 into Eq. 4.6.2, and used this in Eq. 4.6.1. The source term would then be a function of U raised to the power 2. This would

give a less stable discretization than using a formula where U is raised to a power 1. To achieve the latter, the assumption of turbulent production = dissipation is used, giving the important equation:

$$\tau_w = \sqrt{c_\mu}\rho k \tag{4.6.7}$$

which can be rewritten:

$$U^+ = c_\mu^{\frac{1}{4}}\sqrt{k} \tag{4.6.8}$$

The forces on the bed cell can then be written using Eq. 4.6.1, 4.6.2, 4.6.3 and 4.6.5:

$$F = A_w\rho\frac{\kappa U_w}{\ln\left(\frac{30\delta n}{k_s}\right)}\left(c_\mu^{\frac{1}{4}}\sqrt{k}\right) \tag{4.6.9}$$

and the source term for a_p becomes

$$a_p = a_p + A_w\rho\frac{\kappa}{\ln\left(\frac{30\delta n}{k_s}\right)}\left(c_\mu^{\frac{1}{4}}\sqrt{k}\right) \tag{4.6.10}$$

An additional general term is added to take into accounts effects of the velocity being not parallel to the wall. Its derivation can be taken from the formula for the shear stress as a function of the velocity:

$$\tau_w = \mu\frac{\partial U_{paralell}}{\partial n} \tag{4.6.11}$$

The details are given by Melaaen (1990). The result is

$$source = source - A_w\rho\frac{\kappa}{\ln\left(\frac{30\delta n}{k_s}\right)}\left(c_\mu^{\frac{1}{4}}\sqrt{k}\right)\left(\vec{U}\bullet\vec{\bar{n}}\right)n^i \tag{4.6.12}$$

where i is an index for the equation being solved, and $\bar{n}$ is the unit vector normal to the wall. Usually, the term is very small compared with the other terms. The term is zero if the flow is parallel to the wall.

4.7 The k-ε turbulence model

The k-ε model is one of the most widely used turbulence models in CFD. It is described in detail by Rodi (1980). One of the main strengths of the model is its universality, as it has been applied successfully to a large number of different flow situations. Its empirical constants are relatively universal. Another characteristic is that it may overpredict the turbulent eddy-viscosity for some situations. This increases the stability of the CFD model, but it may also lead to inaccurate results for some cases. An example is modelling flow around a cylinder, where the increased eddy-viscosity can prevent emergence of vortex shedding.

The k-ε model models the eddy-viscosity as:

$$\nu_T = c_\mu \frac{k}{\varepsilon^2} \tag{4.7.1}$$

k is turbulent kinetic energy, defined by:

$$k \equiv \frac{1}{2}\overline{u_i u_i} \tag{4.7.2}$$

k is modelled as:

$$\frac{\partial k}{\partial t} + U_j \frac{\partial k}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\nu_T}{\sigma_k}\frac{\partial k}{\partial x_j}\right) + P_k - \varepsilon \tag{4.7.3}$$

where P_k is given by:

$$P_k = \nu_T \frac{\partial U_j}{\partial x_i}\left(\frac{\partial U_j}{\partial x_i} + \frac{\partial U_i}{\partial x_j}\right) \tag{4.7.4}$$

The dissipation of *k* is denoted ε, and modelled as:

$$\frac{\partial \varepsilon}{\partial t} + U_j \frac{\partial \varepsilon}{\partial x_j} = \frac{\partial}{\partial x_j}\left(\frac{\nu_T}{\sigma_k}\frac{\partial \varepsilon}{\partial x_j}\right) + C_{\varepsilon 1}\frac{\varepsilon}{k}P_k + C_{\varepsilon 2}\frac{\varepsilon^2}{k} \tag{4.7.5}$$

The constants in the *k*-ε model have the following standard values:

$$\begin{aligned} c_\mu &= 0.09 \\ C_{\varepsilon 1} &= 1.44 \\ C_{\varepsilon 2} &= 1.92 \\ \sigma_k &= 1.0 \\ \sigma_\varepsilon &= 1.3 \end{aligned} \tag{4.7.6}$$

The numerical implementation of the *k*-ε model involves solving the two partial differential equations for *k* and ε (4.7.3) and (4.7.5). The terms on the left side of the equations and the first term on the right side form a convection-diffusion equation, which can be solved using standard al-

gorithms (Olsen, 1999). Afterwards, Eq. 4.7.1 is solved to obtain the eddy-viscosity.

The main work from a numerical point of view is the calculation of P_k and taking care of the boundary conditions.

Estimating P_k

P_k is calculated by integrating Eq. 3.4.5 over a control volume. The discretized version for a non-orthogonal 3D grid is then (Melaaen, 1990):

$$P_k = \nu_T \frac{\partial U_j}{\partial x_i}\left(\frac{\partial U_j}{\partial x_i} + \frac{\partial U_i}{\partial x_j}\right) = \nu_T\left[\left(\frac{\partial U_j}{\partial x_i}\right)^2 + \frac{\partial U_j}{\partial x_i}\frac{\partial U_i}{\partial x_j}\right] \tag{4.7.7}$$

The equation can be rewritten using Eq. 4.2.2:

$$P_k = \frac{\nu_T}{V^2}[(\omega^i{}_j)^2 + \omega^i{}_j\omega^j{}_i] \tag{4.7.8}$$

or

$$P_k = \frac{\nu_T}{V^2}[2(\omega_1^1)^2 + 2(\omega_2^2)^2 + 2(\omega_3^3)^2 + (\omega_2^1 + \omega_1^2)^2 + (\omega_3^1 + \omega_1^3)^2 + (\omega_3^2 + \omega_2^3)^2] \tag{4.7.9}$$

Each ω has three terms, so 27 terms are obtained altogether. If the lines in the η direction are identical to the z direction, $A^1{}_3$ and $A^2{}_3$ will be zero, so some terms will disappear.

The velocity gradients can be obtained by central differencing. This means for example:'

$$\frac{\partial U_3}{\partial \xi_2} = \frac{1}{2}(U_3^n - U_3^s) \tag{4.7.10}$$

where *n* denotes the cell north of the current cell, and *s* denotes the south cell.

Integration of the term over the cell means that the source term in the *k* equation is given by P_k multiplied with the volume of the cell.

Boundary conditions

Zero gradients are used as boundary conditions close to the water surface and at outlets. Alternatively, Rodi (1980) gives a method to account for damping of *k* from the water surface.

Boundary conditions are also needed for *k* and ε at the wall. Formulas are given by Rodi (1980), based on the assumption that turbulent production is equal to the dissipation of k near the wall. A formula for ε is then

$$\varepsilon = \frac{C_\mu^{3/4} k^{3/2}}{\kappa \delta n}: \qquad (4.7.11)$$

The formula gives the dissipation in the bed cell directly. It is therefore specified in the cell, and it would not be necessary to solve the convection-diffusion equation for e for these cells. However, the solvers usually need a standard method will all grid cells. To make the solver give the result of Eq. 3.4.12, the result is multiplied with 10^{30} and added to the source, simultaneously as 10^{30} is added to a_p. The solver will thereby come up with the result of Equation 3.4.12 in the bed cell.

The source terms for *k* is obtained by integrating the source of the *k*-equation (Eq. 4.7.3) over the bed cell:

$$\int_{\delta V} (P_k - \rho\varepsilon) = \left[\frac{\tau_w U_w}{\delta n} - \frac{\rho C_\mu^{3/4} k_w^{3/2} U_w^+}{\delta n} \right] \qquad (4.7.12)$$

The wall law is then used to find τ_w and U^+_w:

$$U_w^+ = \frac{\kappa U_w}{\ln\left(\frac{30\delta n}{k_s}\right)} \qquad (4.7.13)$$

$$\tau_w = \rho (u^+)^2 \qquad (4.7.14)$$

The velocity in the bed cell, U_w, is taken from the result of the last iteration.

The terms in Eq. 4.7.13 are added to the source. The addition to a_p coefficient becomes:

$$a_p = a_p + \frac{\rho C_\mu^{3/4} k_w^{1/2} U_w^+ V}{\delta n} \qquad (4.7.15)$$

After having used Eqs. 4.7.14 and 4.7.15.
The addition to the source becomes:

$$source = source - \frac{\tau_w U_w V}{\delta n} \qquad (4.7.16)$$

Eqs. 4.7.14 and 4.7.15 has also been used.

Inflow boundary

On inflow boundaries, the values of *k* and ε have to be specified. A procedure outlined by Olsen (1991) can be used for free surface flows. It is based on the investigations of the magnitude (Keefer, 1971) and vertical profile (Naas, 1977) of the eddy-viscosity in rivers. The average eddy-viscosity is given by:

$$\nu_T = 0.11 u_* h \qquad (4.7.17)$$

where u_* is the shear velocity and *h* is the water depth. Note this equation is based on natural rivers. For straight wide channels with uniform flow, a different coefficient from 0.11 is obtained.

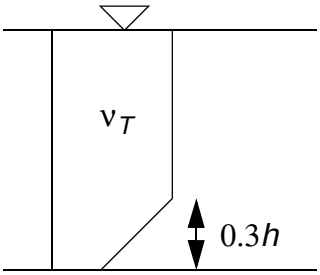


Figure 4.7.1 Vertical profile of the eddy-viscosity according to Naas (1977)

The profile of the eddy-viscosity is given in Fig. 4.7.1. The value of k at the bed can be estimated from the shear stress:

$$k = \frac{\tau}{\rho \sqrt{c_\mu}} \qquad (4.7.18)$$

The value of ε at the boundary is given by Eq. 4.7.11. Thereby the bed value of the eddy-viscosity in Fig. 4.7.1 can be computed, using Eq. 4.7.1. Assuming for example a linear vertical profile of *k*, with a surface value equal to half the bed value, Eq. 4.7.1 can be used to compute the vertical profile of ε.

4.8 The stress terms

The Boussinesq approximation for the turbulent stress term is given as:

$$-\overline{u_i u_j} = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\nu_T\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)\right] \qquad (4.8.1)$$

The first term on the right side is the ordinary diffusive term, used for solving the convection-diffusion equation. The second term is here denoted the stress term. It is often neglected in CFD programs tailor-made for hydraulic engi-

neering, such as SSIIM and Telemac. The reasons are given in the following.

In hydraulic engineering flow fields, there is usually one dominant flow direction. Let us assume such a situation, with the x-direction being in the flow direction in an orthogonal grid. The z direction is vertical and y is in the cross-streamwise direction. The stress term for the velocity U in the x direction can then be written:

$$\frac{1}{\rho}\frac{\partial}{\partial_j}\left[\nu_T\frac{\partial U_j}{\partial x}\right] = \frac{1}{\rho}\frac{\partial}{\partial x}\left[\nu_T\frac{\partial U}{\partial x}\right] + \frac{1}{\rho}\frac{\partial}{\partial x}\left[\nu_T\frac{\partial V}{\partial x}\right] + \frac{1}{\rho}\frac{\partial}{\partial x}\left[\nu_T\frac{\partial W}{\partial x}\right] \tag{4.8.2}$$

The two last terms will have negliable influence on *U*, because *U* is much larger than *V* and *W*. The first term is the diffusion in the streamwise direction. This term is similar to the diffusive term calculated when solving the convection-diffusion equation. Looking at an analytical solution, the diffusion in this direction will go to zero for large Pechlet numbers (velocity/diffusion).

The velocities in the *y* and *z* directions will be much smaller than in the *x* direction, and will therefore have less influence on the total flow field.

The main diffusive term for this flow field will be:

$$\frac{1}{\rho}\frac{\partial}{\partial z}\left[\nu_T\frac{\partial U}{\partial z}\right] \tag{4.8.3}$$

for the velocity in the *x*-direction. The vertical gradient of the main horizontal velocity will dominate the boundary layer. This term is an ordinary diffusive term included in the convection-diffusion equation.

The effect of the stress terms has been estimated by several researchers. Suriyaarchchi (2000) found negliable differences computing flow around a groyne. Olsen and Aryal (2001) also found negliable influences on the velocity field in a sand trap. Booker (2000) found an average change of 3 % in the velocity close to the bed for flow in a natural river.

Discretization

The discretization of the stress terms can be derived by tensor notation. Using the tensor formulas derived in Chapter 3,1, the term is written:

$$Stress, i = \frac{1}{\rho}\frac{\partial}{\partial x_j}\left[\nu_T\left(\frac{\partial U_j}{\partial x_i}\right)\right] = \frac{\nu_T}{\rho V}\left[\frac{\partial}{\partial x_j}(\omega^j{}_i)\right] \quad (4.8.4)$$

where

$$\omega^j{}_i = \frac{\partial U_j}{\partial \xi}A^{\xi}{}_i + \frac{\partial U_j}{\partial \psi}A^{\zeta}{}_i + \frac{\partial U_j}{\partial \zeta}A^{\eta}{}_i \quad (4.8.5)$$

The values of the areas A in the formula above are components in direction *i*.

The stress term is integrated over the volume of the cell, and the Gauss theorem applied. The term is evaluated at the surfaces between the cells, corresponding well with the physical concept of stresses on a finite element. As an example, we look at the surface between cell *i* and *i+1* in a three-dimensional structured grid. This is at the east side of cell *i*. After integration, the source term for Equation *i*, for the east wall can be written:

$$Source = \frac{\nu_T}{\rho V}\sum_{j=1}^{3} A_j(\omega^j{}_i) \quad (4.8.6)$$

The values of V and ν_T are interpolated linearly from the values in the center of the two cells. The areas *A* are also taken directly from the physical areas of the wall, or interpolated linearly from the two cells.

The three velocity gradients are given in the directions ξ,ψ and ζ. The gradient in the ξ direction (east-west) can be evaluated as:

$$\frac{\partial U}{\partial \xi} = U_{i+1} - U_i \quad (4.8.7)$$

This gradient is in the direction normal to the surface. The two other directions are along the surface, in the ψ (north-south) and ζ (top-bottom) direction. The gradients in these directions are obtained by first interpolating the velocities to the sides of the surface, according to Fig. 4.8.1 below:

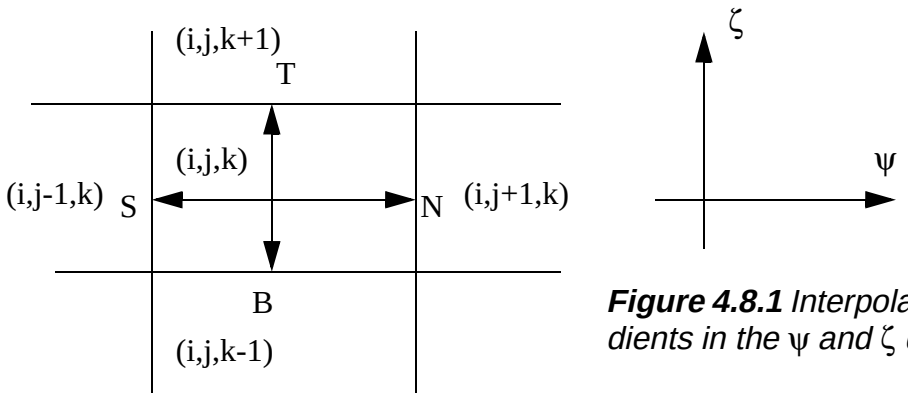


Figure 4.8.1 *Interpolation of gradients in the ψ and ζ directions*

The gradients in the ψ and ζ directions becomes:

$$\frac{\partial U}{\partial \psi} = U_N - U_S \tag{4.8.8}$$

$$\frac{\partial U}{\partial \zeta} = U_T - U_B \tag{4.8.9}$$

The values in position S, N, B and T can be evaluated by interpolation from the surrounding cells. A linear interpolation can be used, where the distances from the position to the center of the cells are used. A simplification is to assume equal interpolation coefficient for all four cells surrounding each position. For example, for position N, this gives the following formula:

$$U_N = 0.25(U_{i,j,k} + U_{i,j+1,k} + U_{i+1,j,k} + U_{i+1,j+1,k}) \tag{4.8.10}$$

The formula gives the stresses for one equation, on one cell side. Each cell has six surfaces, so there will be six source terms line Eq. 4.8.6 for each cell. The source term for one cell will be the same as the sink term for the cell on the other side of the surface. Therefore, three evaluations need to be made for each cell. Note the ω term depend on the velocity direction, and will change for each equation.

4.9 Non-orthogonal terms

The non-orthogonal terms are caused by turbulent diffusion in a non-orthogonal grid. The terms originate from the diffusive terms of all convection-diffusion equations, both for calculating water flow and sediment concentration. A physical interpretation of the terms is given in the example below. Looking at a non-orthogonal grid with a concentra-

tion gradient in Fig. 4.9.1, the physical explanation of the non-orthogonal terms are the fluxes through cell surface *e*.

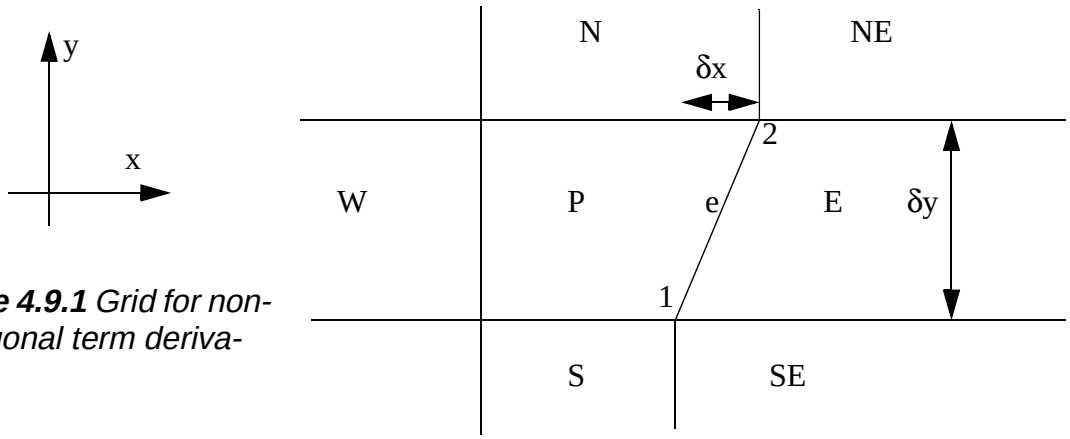


Figure 4.9.1 *Grid for non-orthogonal term derivation*

To simplify the problem, it is assumed that the velocities are zero, so the convective term is not included. There is a fixed amount of turbulent diffusion, Γ. The concentration gradient in the *x* direction is zero, and the concentration gradient in the *y* direction is constant equal to δc. The orthogonal flux through Surface *e* is then zero, as the concentration gradient between Cell *P* and Cell *E* is zero. The flux through Surface e is caused by the non-orthogonal grid compared with the main concentration gradient direction. The flux can be calculated by a non-orthogonal term.

In the Figure above, the non-orthogonal contribution is:

$$F = \Gamma_e A \frac{\delta c}{\delta y} = \underbrace{\left(\frac{\Gamma_P + \Gamma_E}{2}\right)}_{1}\underbrace{\left(\frac{\delta x}{\delta y}\right)}_{2}\underbrace{(c_2 - c_1)}_{3} \qquad (4.9.1)$$

Equation 4.9.1 is relatively simple, since the problem in Fig. 4.9.1 is relatively uncomplicated. For a general 3D case, the non-orthogonal terms are more involved.

The numbers below Eq. 4.9.1 denotes the three terms in brackets:

1. The turbulent diffusion coefficient
2. The geometry term,
3. The concentration difference.

The turbulent diffusion coefficient is interpolated linearly from the surrounding cells, and so is the concentration in the cell corners. This is a relatively straightforward procedure, also in a general 3D geometry. The main difficulty is the estimation of the geometry term. The term is also called a **geometric diffusion coefficient**, and has the general

expression of a **surface area divided by a distance** in 3D. Its units are in meters (m^2/m). The general 3D expression for the geometric diffusion coefficient can be derived using the algebraic expressions in Chapter 4.2 and tensor calculus (Melaaen, 1990). Alternatively, it can be derived by using physical reasoning and vectors. The latter option is used in the following, as shown in the figure below.

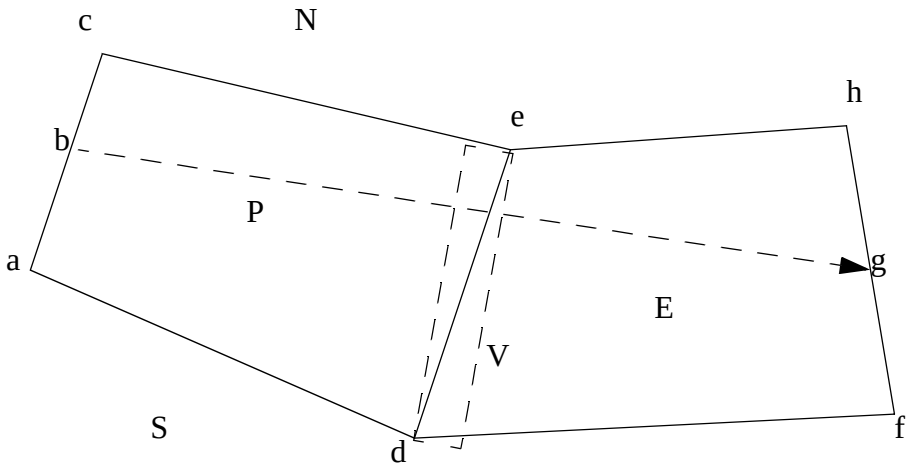


Figure 4.9.1 *Non-orthogonal flux through the surface between Cell* P *and Cell* E

Fig. 4.9.1 shows Cell *P* and Cell *E*, and the surface *d-e* between the cells. The corners of *Cell P* are *a*, *d*, *e*, *c* and of *Cell E* are *d, f, h, e*. The point *b* is midway between *a* and *c*, and the point *g* is midway between point *h* and *f*. The non-orthogonal flux over surface *d-e* is to be computed. The flux will take place in the volume marked *V*, surrounding line *d-e*. The main difficulty from a physical perspective is to find **the thickness of this volume**, **normal to the *d-e* direction**. This is done by using the dot product of the vector *d-e* and *b-g*.

The vector *b-g* is denoted $\mathbf{v}^{\xi}$, the vector *d-e* is denoted $\mathbf{v}^{\psi}$ and the vector in the third direction is denoted $\mathbf{v}^{\zeta}$. The starting and ending points of $\mathbf{v}^{\zeta}$ are midway between point *d* and *e*. The cross-sectional area of the non-orthogonal surface will then be:

$$A = \left(v^{\xi} \bullet \frac{v^{\psi}}{\left| v^{\psi} \right|} \right) \times v^{\zeta} \qquad (4.9.2)$$

The dot product of vector $\mathbf{v}^{\psi}$ and the unit vector in direction $\mathbf{v}^{\xi}$ gives a vector the direction normal to *d-e*, with the length equal to the thickness of the non-orthogonal volume. This is cross-multiplied with the vector in the third direction to get the non-orthogonal surface area.

The geometric diffusion coefficient can be estimated by dividing A with the distance between *d* and *e*:

$$G = \frac{A}{\left|v^{\xi}\right|} = \frac{v^{\xi} \bullet v^{\psi}}{\left|v^{\xi}\right|\left|v^{\psi}\right|} \times v^{\zeta} \qquad (4.9.3)$$

The last step is an approximation, as the distance will be slightly smaller, depending on the non-orthogonality of the grid. Also, the calculation of the length of the vectors takes relatively long computational time. Therefore, the denominator can be replaced by the cross-product of the vectors. The following equation then emerges, corresponding to the exact equation derived by tensor calculus (Melaaen, 1990):

Note: the dot product in Eq. 4.9.3 will be zero if the grid is orthogonal, making the term vanish.

$$G = \frac{(v^{\xi} \bullet v^{\psi}) \times v^{\zeta}}{v^{\xi} \times v^{\psi}} \qquad (4.9.3)$$

Equation 4.9.1 gives the flux in a two-dimensional case. For a 3D case, there are two non-orthogonal fluxes for each surface, in the two directions following the surface. The terms will then give the flux through one surface. Each cell has six surfaces, so there are twelve fluxes to calculate for each cell. This can be simplified by realizing the non-orthogonal flux into one cell is the same as the flux flowing out of the neighbouring cell, reducing the number of flux computations pr. cell to six.

The non-orthogonal term will usually be smaller than the orthogonal term, also for strongly non-orthogonal grids. The non-orthogonal term further includes the turbulent diffusion, Γ. As said previously, Γ is difficult to estimate exactly. The k-ε turbulence model estimates a uniform Γ in all three spatial directions, even though experiments show it can vary with an order of magnitude between the smallest and largest directional value. The results of the CFD model still becomes reasonably correct, and this is because the computation is not extremely sensitive to the diffusive terms. Some researchers have even introduced artificial diffusion, where the effective diffusion is increased by orders of magnitude to get a stable solution. And still the results are reasonable. Therefore the non-orthogonal terms will usually have very little effect on the solution, and they are neglected in some CFD models.

4.10 Solvers

The discretized Navier-Stokes equations, or the convection-diffusion equation for sediment concentration, gives a formula for the unknown, ϕ, in a cell, *p*, as a function of the surrounding cells:

$$\phi_p = \frac{\sum_{nb} a_{nb}\phi_{nb}}{a_p} + \frac{S_p}{a_p} \qquad (4.10.1)$$

Solving the convection-diffusion equation for sediment concentration, the velocities are known, so the a_{nb} coefficients can be calculated before the solution of the equations. In the Navier-Stokes equations, the a_{nb} coefficients are functions of a variable velocity. The coefficients have to be updated during the solution procedure.

There exist a large number of solvers for the equations. A classification is whether the solver is **direct or iterative**. A direct solver will establish a matrix for the system of equations, and invert it to obtain the solution. This is extremely time-consuming from a computational point of view, and requires very large computer memory. Also, if the Navier-Stokes equations are solved, the SIMPLE method is used and the turbulence model will have to give the values of the turbulent eddy-viscosity. It is not possible to solve all these six unknown simultaneously, so some type of iteration has to be used.

Solving Eq. 4.10.1 to a large degree of accuracy would take very long time. Since the coefficients a_{nb} are obtained from the velocity field from a previous iterations, they are not exactly correct anyway. An iterative solver is therefore used in most CFD finite volume programs, where only some iterations are done to improve an already guessed solution. Then the a_{nb} coefficients are updated, the SIMPLE method applied for the pressure and a turbulence model used to calculate the eddy-viscosity, before more iterations are done.

If there are few cells, this approach will solve the equations relatively rapidly. However, for many cells, the changes in variables will progress slowly across the domain. Some particular solution methods are then useful:

The TDMA method

TDMA is an abbreviation for the Tri-Diagonal Matrix Algorithm, also called the Thomas Algorithm. It is a one-dimensional solution procedure, solving the system of equations directly. When used in three dimensions, it can not solve the equations directly, but it can be used in an iterative procedure. For a structured grid, each row of cells in one direction can be thought of as a one-dimensional system. The solution is done for all the rows, and repeated for all three directions.

Derivation of the TDMA method is based on a one-dimensional array of unknowns. Let us for example assume we solve for the east-west direction. Eq. 4.10.1 can then be written:

$$A_i\phi_i = B_i\phi_{i+1} + C_i\phi_{i-1} + D_i \qquad (4.10.2)$$

where

$$\begin{aligned} A &= a_p \\ B &= a_e \\ C &= a_w \\ D &= Source + a_n\phi_n + a_s\phi_s + a_t\phi_t + a_b\phi_b \end{aligned} \qquad (4.10.3)$$

The TDMA method computes the correct values in the whole array by using two one-dimensional auxiliary arrays, P and Q defined from:

$$\phi_i = P_i\phi_{i+1} + Q_i \qquad (4.10.4)$$

or

$$\phi_{i-1} = P_{i-1}\phi_i + Q_{i-1} \qquad (4.10.5)$$

Inserting Eq. 4.10.5 into Eq. 4.10.2 gives:

$$A_i\phi_i = B_i\phi_{i+1} + C_i(P_{i-1}\phi_i + Q_{i-1}) + D_i \qquad (4.10.6)$$

This removes the dependence of ϕ_i on ϕ_{i-1}. Solving this equation with respect to ϕ_i gives:

$$\phi = \frac{B_i}{A_i - C_iP_{i-1}}\phi_{i+1} + \frac{C_iQ_{i-1} + D_i}{A_i - C_iP_{i-1}} \qquad (4.10.7)$$

Comparing Eq. 4.10.7 with Eq. 4.10.4, they are the same if P and Q are computed as:

$$P_i = \frac{B_i}{A_i - C_iP_{i-1}} \qquad (4.10.8)$$

$$Q_i = \frac{C_i Q_{i-1} + D_i}{A_i - C_i P_{i-1}} \qquad (4.10.9)$$

The TDMA method uses two sweeps. The computation starts the first sweep at *i*=1, and computes the values of *P* and *Q* for the whole array until *i=n*, where *n* is the length of the array. Then it starts the second sweep at *i=n*, and computes ϕ for all the arrays by using Eq. 4.10.4.

Multi-grid solvers

The solution time, *T*, for a given solver can be given by the following equation:

$$T = an^b + c \qquad (4.10.2)$$

where *n* is the number of grid cells and *a*, *b* and *c* are coefficients specific for each solver. For large grids, the most important coefficient is usually *b*. It can be shown that the theoretical minimum *b* value is 1.0. Of all the different solvers, some multi-grid methods have the lowest values of *b*, close to unity.

One of the main problems of fine grids is to get information about the unknown variables to move through the grid. Assuming a Gauss-Seidel solver, the information may only move one grid cell for each sweep. Convergence will then be very much dependent on the number of grid cells. The multi-grid solver addresses this problem by using several grids for the same domain, with different number of grid cells. The coarsest grid will move information through the grid very fast, while the finer grid will resolve steep gradients. Information about the variables are transferred between the different grids. There may be several grids with varying fineness, between the coarsest and finest grid.

One type of multi-grid algorithms is called **block-correction**. It is mostly used for structured grids. For a two-dimensional situation, the grids then look like this:

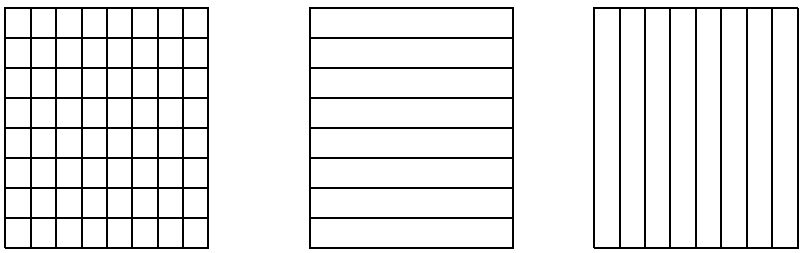


Figure 4.10.1. Grids for the block-correction. Original grid (left), coarse grid, direction 1 (middle) and coarse grid, direction 2 (right).

The iterations are started on the original grid. Then all variables are summed in a slice of the grid, so that a one-dimensional grid emerges. This is solved, and the result is used to correct the original values. This is repeated in all directions, shown here with two coarse grids, for a two-dimensional situation. For a 3D situation, there would be three coarse grids.

The TDMA method is very applicable for solving the 1D equations on the coarse grids.

5. Special algorithms for use in hydraulics

Most general-purpose CFD models are made for use in Mechanical Engineering, for example modelling air flow around a vehicle or and aeroplane, or a process inside an engine. Algorithms for special Hydraulic Engineering problems are often not included. The bed roughness of a river is a typical example, together with sediment transport.

5.1 A limiter scheme for the wall laws

The previously described wall law for rough boundaries (Eq. 4.6.1) was developed for relatively small roughness compared with the water depth. If the roughness height is larger then 15 times the height of the bed cell, a negative velocity will be predicted. When the CFD model calculates the water depth and the roughness, such a situation can easily occur, leading to stability problems. To maintain stability, a limiter of the roughness/cell depth ratio is often imposed. This can be thought to be similar to a flux limiter in a convective discretization scheme. The following formula for the law of the wall can be used:

$$\frac{U}{U_*} = max\left(0.01, \ln\frac{30y}{k_s}\right) \qquad (5.1.1)$$

Although the limiter algorithm will prevent program crashes, the result may not be correct. The sink term may be too small, overpredicting the velocities for high roughness. A more correct approach is to add extra negative source terms for high roughness/cell height ratios. This can be based on porosity or flow in vegetated areas, as further described in Chapter 5.2.

5.2 Modelling large roughness elements

The main two types of large roughness elements found in river engineering are rocks and vegetation. Rocks have been modelled by Olsen and Stokseth (1995) by using a porosity approach. A formula for groundwater flow by Engelund (1953) was used:

$$I = \beta_0 \frac{1-n}{n^3}\frac{U^3}{gd} \qquad (5.2.1)$$

Here, I is the hydraulic gradient, β_0 is a constant, n is the porosity, g is the acceleration of gravity and d is the char-

acteristic particle diameter. A β_0 value of 3.0 was suggested by Engelund, and used by Olsen and Stokseth. Additionally, an algorithm to estimate the porosity as a function of the bed topography was suggested by Olsen and Stokseth (1995):

$$n_k = 1 - c_k\left(1 - \frac{m_t - m_k}{m_t + 0.5}\right) \qquad (5.2.2)$$

The formula is based on a number of randomly measured points (x,y,z) in a river. In the 2D depth-averaged cell there are m_t points. The porosity, n_k, at level k is given by Eq. 5.2.2, where m_k is the number of points in a cell above level k. The empirical parameter c_k was varied and a value of 0.3 was found to produce reasonable results for one particular river. Olsen and Stokseth used wall laws in the cells above the porous cells where Eq. 5.2.1 was used, also for the turbulence variables.

Later, Fischer-Antze et. al. (2000) used a formula for drag on a cylinder to simulate vegetation in a river. A laboratory experiment was modelled, where vertical circular rods were used to simulate the vegetation. The drag formula was used:

$$F = \frac{1}{2} C_D \rho U^2 A \qquad (5.2.3)$$

F is the drag force on an object, C_D is the drag coefficient, ρ is the water density, U is the velocity and A is the surface area of the object, projected normal to the flow direction. The discretized version is obtained by integration over a control volume, and linearizing the source into the a_p term:

$$a_p = a_p + \frac{1}{2} C_D \rho U n d \delta z \qquad (5.2.4)$$

where n is the number of stems in a cell, d is the stem diameter and δz is the height of the cell. Fisher-Antze et. al. (2001) obtained very good results although no special modifications of the turbulence model was used.

5.3 Including gravity for spillway and steep floodwave modelling

In general, the location of the water levels can be estimated using a known water surface elevation and the pressure close to the water surface computed by the CFD model. The known water level is usually at the downstream outlet for subcritical flow. For supercritical flow, it is possible to specify the upstream water level.

However, in some cases both the upstream and downstream water levels are unknown. An example is flow over a spillway, where the water level is determined by the critical flow over the spillway crest. In a general spillway geometry, it is not known where the critical section is. The CFD model therefore have to calculate the water surface location without giving any user input for known water levels. A physical model study of the same case presents a similar problem.

The method to solve the problem in the CFD model is to include the gravity in the calculation:

$$Source_{vertical} = Vg\rho_w \qquad (5.3.1)$$

V is the volume of the cell, g is the acceleration of gravity and ρ_w is the water density.

The gravity term is usually very large compared with the other terms, causing instabilities. A transient term has to be added to get a stable solution, with a very small time step. The computation is started with a horizontal water surface. It is recommended to start well above the assumed final location of the water levels. The gravity term in the CFD model then pulls the water surface down. This is visualized in the figure below:

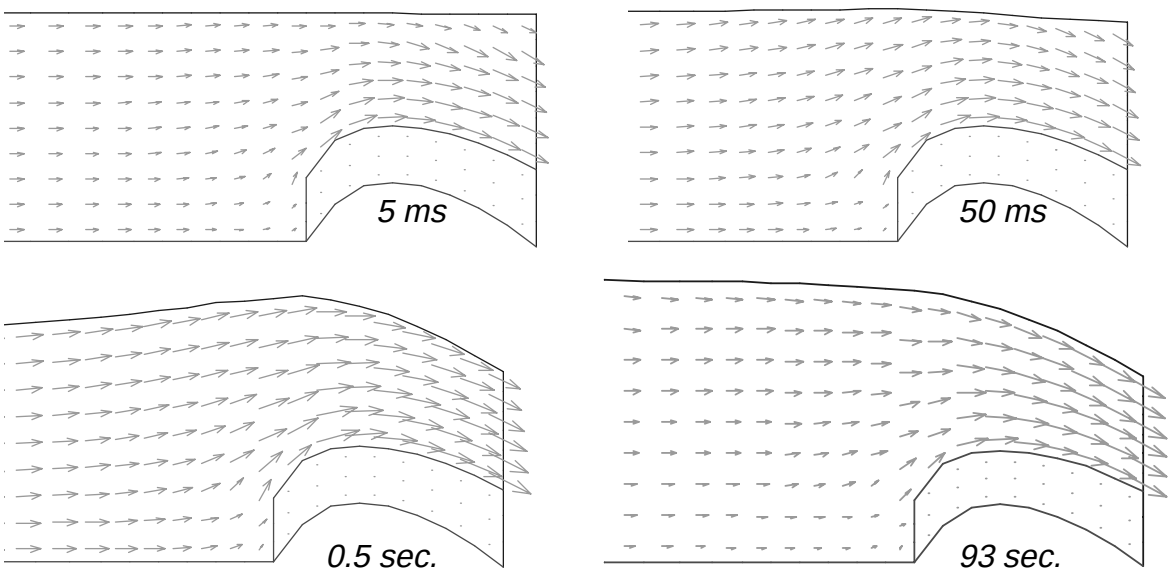


Figure 5.3.1 *Longitudinal profile of the water level and velocities for computation of coefficient of discharge for a spillway. The numbers show the computed time.*

The gravity term could in principle be introduced in all hydraulic models, but since it causes instabilities, it is not included when it is not necessary. Other than spillway cases, the term has also been used to simulate flood waves hitting an object (Lovoll et. al, 1995). The water surface is then very complex, and both upstream and downstream water levels are unknown.

5.4 Density currents and gravity

A possible direct approach to modelling density currents is to include gravity into the calculations, and let the density vary according to the resulting concentration field. This would work in principal, but as mentioned in the previous chapter, large source terms will cause instabilities.

A more stable approach is to include only the gravity on the effect of the density **increase**, compared with the water density. A gravity current will usually move close to the bed, giving large source terms only on cells in this region. Most of the cells may therefore not be affected by large source terms. Also, the density increase is usually one or two orders of magnitude smaller than the water density. The magnitude of the source terms therefore decreases correspondingly.

This approach was used by Olsen and Tesaker (1995) to model a turbidity current in a flume.

The additional source term in the vertical direction in the Navier-Stokes equations becomes:

$$Source_{vertical} = Vg(\rho_s - \rho_w)c \qquad (5.4.1)$$

V is the volume of the cell, g is the acceleration of gravity, ρ_s and ρ_w is the sediment and water density and c is the sediment concentration.

The effect of the density variations on the turbulence taken into account by introducing a modified eddy viscosity. The eddy viscosity from the k-ε model is multiplied with a factor taking into account the velocity and concentration gradients (Rodi, 1980):

$$\nu_T = \nu_{T,0}\left[1 + \beta\left(-\frac{g}{\rho}\frac{\frac{\partial \rho}{\partial z}}{\langle\frac{\partial U}{\partial z}\rangle^2}\right)\right]^{\alpha} \qquad (5.4.2)$$

The eddy viscosity is denoted ν_T, β is a constant equal to 10, ρ is the density of the water/sediment mixture, *U* is the velocity, *z* is the geometrical distance in the vertical direction and α is a constant equal to -0.5. The values of α and β are given by Rodi (1980). Olsen and Lysne (2000) found values of α equal to -1.3 together with β equal to 10 give better results when computing currents in an ice-covered lake.

5.5 Modelling horizontal density gradients for stratified lakes

A number of studies have been carried out modelling stratified lakes in three dimensions. Early work was carried out by Simons (1976). A quasi-3D approach was used, where the pressure field was computed from a depth-averaged 2D grid, and hydrostatic pressure was assumed in the vertical direction.

The approach described here is fully 3D with non-hydrostatic pressure computation. Modelling stratified flows using gravity in the vertical direction would lead to large source terms and instabilities. Adding additional forces from density deviation from standard water density would affect most of the cells in the geometry, as the density variation is often over the whole depth of a lake. Instabilities would thereby occur.

A solution is to use the same approach as the quasi-3D models, where horizontal density gradients are computed and included into the Navier-Stokes equations only in the horizontal directions. A sketch of the physics of the gradients is given in Fig. 5.5.1.

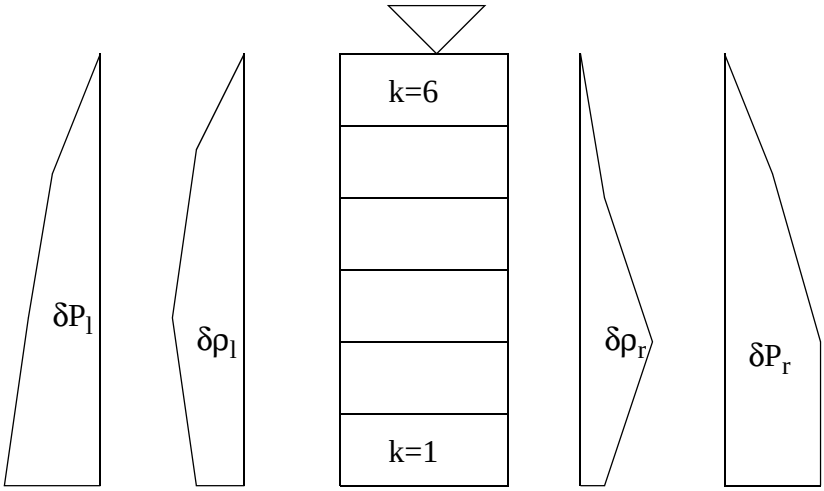


Figure 5.5.1 *Vertical density and pressure gradients*

The δρ notation indicates the density of the water, when the standard water density, ρ_{std}, is subtracted:

$$\delta\rho = \rho - \rho_{std} \tag{5.5.1}$$

The pressure distribution, δP, in the vertical direction due to the density variation is then computed for cell *m*:

$$\delta P^m = \sum_{k=m}^{n} g(\delta z)_k(\delta\rho)_k \tag{5.5.2}$$

where δz is the vertical height of a grid cell and *n* is the number of grid cells in the vertical direction.

The horizontal pressure gradient source term, S_m, for cell *m* is then:

$$S_m = \delta A_m \frac{\delta P_l^{\,m} - \delta P_r^{\,m}}{2} \tag{5.5.3}$$

where the index *l* and *r* denotes left and right cell, respectively, according to Fig. 5.5.1.

The source term above applies locally to one grid cell. Additionally, there will be a horizontal gradient due to the sum of all the density differences over the whole vertical colounmn. This force will affect all the cells in the vertical column. The source term can be computed as:

$$S_m = \frac{\sum_{k=1}^{n} \delta A_k \frac{\delta P_l^k - \delta P_r^k}{2}}{\sum_n \delta z_k} \delta z_m \qquad (5.5.4)$$

The equations above work well as long as the cell sizes have similar vertical distribution. In other words, if the horizontal grid lines are completely horizontal. If a bed-boundary fitted grid is used, instabilities and inaccurate results will occur.

6. Sediment transport modelling

The science of sediment transport is complex, and there are many uncertainties and conflicting views from various schools of researchers. There exist a very large amount of literature on the subject. It would be very difficult to incorporate all sediment research into a CFD computer program. In the following, some more well-accepted methods have been selected.

6.1 Convection-diffusion equations for suspended sediment

The convection-diffusion equation for suspended sediment concentration is solved using the methods described in Chapter 3. The only change in the equation is the added term for the fall velocity of the sediment particles. It is the third term on the left side of the equation below:

$$\frac{\partial c}{\partial t} + U_j\frac{\partial c}{\partial x_j} + w\frac{\partial c}{\partial z} = \frac{\partial}{\partial x_j}\left(\Gamma_T\frac{\partial c}{\partial x_j}\right) \qquad (6.1.1)$$

The sediment concentration is denoted c, w is the fall velocity of the particles, U is the water velocity, x is a space dimension and Γ is the turbulent diffusivity. The three directions are x_1, x_2 and x_3, and the velocities in the three directions are U_1, U_2 and U_3.

The most convenient way of discretizing the fall velocity term is to add the fall velocity to the water velocity in the vertical direction. The convective terms can then be written:

$$U_1\frac{\partial c}{\partial x_1} + U_2\frac{\partial c}{\partial x_2} + (U_3 + w)\frac{\partial c}{\partial x_3} \qquad (6.1.2)$$

Alternatively, the term can be added to the source in the vertical direction:

$$Source = c_u A_3 w \qquad (6.1.3)$$

The top area of the cell is denoted A_3, and the concentration in the cell above the current cell is denoted c_u. A first-order upstream approximation has been used.

6.2 Bed boundary condition for suspended load

The theory behind the bed boundary condition for the sediment concentration was initially developed by Einstein (1950). Einstein assumed the sediment concentration close to the bed was a function of the forces on particles at the bed and the relative weight of the particle. As it was difficult to use such a formula directly at that time, an integration over the depth was done, resulting in a sediment transport formula where a logarithmic velocity profile was assumed. Most sediment transport formulas assume a nearly uniform flow, and includes the water depth as a parameter. In a 3D case with recirculation zones, the assumption of a logarithmic velocity profile is not valid. Most of the existing formulas for sediment transport in rivers can therefore not be used directly.

A formula for sediment concentration close to the bed have to include only local bed parameters: sediment characteristics, bed shear stress and turbulence. Van Rijn (1987) carried out early research on such formulas, where he used dimensionless numbers of shear stress and particle diameter to correlate empirical coefficients against observations from the field and flume experiments. According to van Rijn (1987), the boundary condition at the bed for the convection-diffusion equation can be specified in one of two ways:

- Adding a source term for the bed cells, where the sediment pick-up rate is estimated
- Specifying an equilibrium sediment concentration close to the bed

The second approach is most used today, and van Rijn's formula is given as:

$$c_{bed} = 0.015 \frac{d^{0.3}}{a} \frac{\left[\frac{\tau - \tau_c}{\tau_c}\right]^{1.5}}{\left[\frac{(\rho_s - \rho_w)}{\rho_w \nu^2}\right]^{0.1}} \qquad (6.2.1)$$

The sediment particle diameter is denoted *d*, *a* is a reference level set equal to the roughness height, τ is the bed shear stress, τ_c is the critical bed shear stress for movement of sediment particles, ρ_w and ρ_s are the density of wa-

ter and sediment, ν is the viscosity of the water and *g* is the acceleration of gravity.

The sediment concentration in the bed cell is computed using Eq. 6.2.1. A special solved that do not compute the concentration for the bed cells can be used. Alternatively, the concentration can be multiplied with a large number, and added to the source and a_p:

$$a_p = 10^{20}$$

$$source = c_b 10^{20}$$

Because of the large number, the other terms become negliable, and the effective concentration becomes:

$source/a_p = c_b$.

6.3 Modelling bed load with suspended load

Traditionally, sediment transport has been divided into bed load and suspended load. This concept was derived by Bagnold (1973) from observations of wind-blown sand in the desert. A definition of difference between the bed load and suspended load is generally not agreed upon, and the whole concept has caused much debate in the scientific community. Einstein (1950) defined bed load as particles moving in a region two particle diameters from the bed. This meant the particles were mostly rolling on the bed, instead of moving up into the main current to become suspended. Van Rijn (1987) admitted there were no clear distinction between bed load and suspended load, but he used Bagnolds approach.

Van Rijn's formula for bed load is:

$$\frac{q_b}{D_{50}^{1.5}\sqrt{\frac{(\rho-\rho)g}{\rho}}} = 0.053\frac{\left[\frac{\tau-\tau_c}{\tau_c}\right]^{1.5}}{D_{50}^{0.3}\left[\left(\frac{(\rho-\rho)g}{\rho\nu^2}\right)\right]^{0.1}} \qquad (6.3.1)$$

6.4 Bedform modelling

The classical literature on sediment transport also contain a number of formulas for bed forms. However, the same problem as with the formulas for sediment transport exist, as the formula often includes the water depth. Van Rijn (1987) also looked into this problem, and suggested formulas for the bed form height, Δ, independent of the water depth:

$$\frac{\Delta}{d} = 0.11\left(\frac{D_{50}}{d}\right)^{0.3}\left(1 - e^{-\left[\frac{\tau - \tau_c}{2\tau_c}\right]}\right)\left(25 - \left[\frac{\tau - \tau_c}{\tau_c}\right]\right) \quad (6.4.1)$$

where d is the water depth. The effective roughness was then computed as (van Rijn, 1987):

$$k_{s+\Delta} = 3D_{90} + 1.1\Delta\left(1 - e^{-\frac{25\Delta}{\lambda}}\right) \quad (6.4.2)$$

where λ is the bedform length, calculated as $7.3d$.

Note that van Rjin's equations for bed form roughness was developed on mostly uniform sediments. For non-uniform sediments, smaller bed forms was observed in the FCF Series C experiment at HR Wallingford (Bryant, 1999).

6.5 Shear stress and particle movement

The shear stress on the bed is one of the most important parameters when modelling sediment transport. Two questions need to be answered:

- How is the shear stress on the bed calculated?
- How to calculate the critical shear stress for movement of sediment particles?

Calculation of the bed shear stress

The shear stress on the bed can be calculated by the wall laws. Given the velocity in the bed cell, the height of the bed cell and the roughness, the only unknown is the shear velocity. The shear stress is calculated as

$$\tau_{bed} = \rho U_*^2 = \frac{U_{bed}\kappa}{\ln\left(\frac{30\delta n}{k_s}\right)} \quad (6.5.1)$$

If the k-ε turbulence model is used, it is often assumed that production and dissipation of turbulence is in equilibrium close to the wall. Then the shear stress can be computed directly from the value of *k* near the bed:

$$\tau_{bed} = \sqrt{c_\rho}\rho k = 300k \qquad (6.5.2)$$

Calculation of the critical shear stress

The critical shear stress, τ_c, for a bed sediment particle to move is calculated from Shield's diagram. The diagram relates the dimensionless shear stress and the dimensionless particle diameter. The curve can be parameterized using the following formulas:

For R* > 500:

$$\tau_* = 0.06 \qquad (6.5.3)$$

For R* < 500:

$$\log(\tau_*) = a\ \log R_* + b(\log R_*)^2 + c(\log R_*)^3 + d(\log R_*)^4$$
$$a = -0.99863612$$
$$b = -0.92539586$$
$$c = 0.54283631$$
$$d = -0.08454406$$

(6.5.4)

where

$$R_* = \frac{U_* d_s}{\nu} \qquad (6.5.5)$$

and

$$\tau_* = \frac{\tau_c}{g(\rho_s - \rho_w)d_s} \qquad (6.5.6)$$

If the bed slopes upwards or sideways compared to the velocity vector, the critical shear stress for the particle will change. The decrease factor, *K*, as a function of the sloping bed was given by Brooks (1963):

$$K = -\frac{\sin\phi\sin\alpha}{\tan\theta} + \sqrt{\left(\frac{\sin\phi\sin\alpha}{\tan\theta}\right) - cos^2\phi\left[1 - \left(\frac{\tan\phi}{\tan\theta}\right)^2\right]}$$

(6.5.7)

The angle between the flow direction and a line normal to bed plane is denoted α. The slope angle is denoted ϕ and θ is a slope parameter. The factor K is multiplied with the critical shear stress for a horizontal surface to give the effective critical shear stress for a sediment particle.

The bed form roughness is only to be applied to the water flow field. When calculating the shear stress at the bed for the sediment transport, only the grain roughness is to be used. The shear stress therefore has to be reduced by the factor F:

$$F = \left(\frac{k_s}{k_s + k_\Delta}\right)^{\frac{1}{3}} \tag{6.5.8}$$

where k_s is the roughness due to the sediments, and k_Δ is the roughness due to the bed forms.

6.6 Bed movements

The vertical changes, δz_0, in the bed geometry can be calculated from the continuity equation for sediment deposition/erosion in a cell close to the bed:

$$A\delta z_0 = r(Inflow - Outflow) \tag{6.6.1}$$

A is the bed cell area component in the horizontal plane. The conversion factor between the sediment flux and the sediment in the bed is denoted *r*. If the sediment concentration is calculated as a volume fraction, *r* is typically around 2.

The *Inflow* of sediments is calculated according to the following formula, derived from the continuity equation:

$$Inflow = \sum_{nb} a_{nb} c_{nb} \tag{6.6.2}$$

The outflow is computed according to the following formula:

$$Outflow = c_{bed} \sum_{\Pi} a_{nb} \tag{6.6.3}$$

The parameter Π here denotes the neighbouring cells, where the a_{nb} are taken only from those surfaces bordering

the cell we are looking at. For example, looking at cell i, and the water is flowing from cell *i* to cell *i*+1, a_W for cell *i* will be included.

The average bed movement in each cell must be transferred to the corners of the cell, where the vertical levels of the bed are defined. Fig. 6.6.1 shows the grid.
:

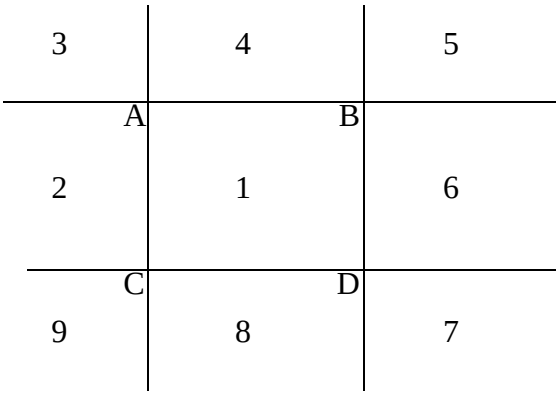


Figure 6.6.1 *Definition of cell (numbers) and corner (letters) indexes for calculation of bed movements*

Raising point *A* a vertical distance δz_A, makes the following volume, based on the formula for a pyramid:

$$V_A = \frac{1}{3}\delta z_A(A_1 + A_2 + A_3 + A_4) \qquad (6.6.4)$$

The volume made up of the movements in the four corners is equal to the volume made up of the estimated movement of the bed cell:

$$V_A + V_B + V_C + V_D = A_1\delta z_0 \qquad (6.6.5)$$

Further, it is assumed that the movement of all the corners are the same. This gives:

$$A_1\delta z_0 = \frac{1}{3}\delta z(4A_1 + 2A_2 + 2A_4 + 2A_6 + 2A_8 + A_3 + A_5 + A_7 + A_9) \qquad (6.6.6)$$

$$\delta z = \frac{3A_1}{(4A_1 + 2A_2 + 2A_4 + 2A_6 + 2A_8 + A_3 + A_5 + A_7 + A_9)}\delta z_0 \qquad (6.6.7)$$

Two particular problems have to be dealt with: limits of movable beds, and sloping bed in relation to the main water

flow direction. The latter question is discussed in Chapter 6.11.

Limits of movable bed

The problem will typically occur when modelling scour in alluvial material, when there is bedrock below the sediments. The numerical algorithms have been developed when modelling flushing of sediments from reservoirs. The results were compared with a laboratory model where the bed was made of concrete.

If erosion takes place, and there is a limit to the moveable bed, it may not be possible to move one or more corners as much as what is given by Eq. 6.6.4. Then the system of equations must be solved with this limitation. It is possible to use Eq. 6.6.1 and 6.6.2, compute the volumes for each corner movement, and thereby solve the problem.

Another implication of limits to movable bed is a limit to the sediment concentration close to the bed. If this exceeds a given value, the algorithm may want to erode more sediment than what is in the bed. This available sediments in the bed must be computed, and a corresponding maximum sediment concentration close to the bed must be found. The concentration close to the bed must not be allowed to exceed this value, and this must be taken care of by the solution procedure.

6.7 Sand slide algorithms

During bed changes, the bed slopes may increase to a level above the angle of repose, ϕ. The sediment will then slide downwards, so that the bed slope is reduced. This process is often seen in physical models, for example of local scour. The sand slide process have to be taken into account by the numerical model. One algorithm is described by the following.

All grid lines are checked for steep slopes. If one line has a slope above the angle of repose, it is moved vertically a distance δz. Fig. 6.8.1 below shows the grid from above.

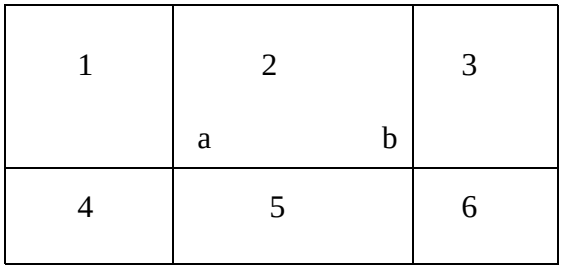


Figure 6.8.1 *Grid indexes*

The grid line *a-b* has too steep slope, with *b* being the higher point. It is surrounded by the cells 1,2,3,4,5 and 6. Point *a* have to be raised, and point *b* have to be lowered. It is important that the sediment continuity is satisfied, so the sum of the bed volume changes have to be zero. The areas of the cell projections in the horizontal planes for the grid cells are called A_1, A_2 ,A_3 ,A_4, A_5 and A_6, for grid cells 1,2,3,4,5 and 6, respectively. Then the following formula for sediment continuity is used:

$$\delta z_a(A_1 + A_2 + A_4 + A_5) = \delta z_b(A_2 + A_3 + A_5 + A_6) \tag{6.8.1}$$

The other equation used is the angle of the slope for line *a-b*:

$$\tan\phi = \frac{(z_b + \delta z_b) - (z_a + \delta z_a)}{\sqrt{(x_a - x_b)^2 + (y_a - y_b)^2}} \tag{6.8.2}$$

The two equations Eq. 6.8.1 and Eq. 6.8.2 are solved to give the values of the two unknown δz_a and δz_b.

The algorithm only applies for non-cohesive sediments. If the sediments contain cohesive material, the bed slope may exceed the angle of repose for non-cohesive material. A more complex method is then required. Also note the angle of repose is lower under water then in dry condition. Typical angles of repose are 38-45 degrees for dry sand, and 30-38 degrees for submerged sand.

6.8 Multiple sediment sizes

In a natural river there will always be a large number of sediment sizes. The CFD model can handle this situation by modelling several sediment sizes. The sediments are divided into groups, and a convection-diffusion equation for

each group is solved. The theory is the same as for a one-dimensional numerical model.

The bed can be divided into three layers:

1. The water layer closest to the bed, where the mixture of sediments and water move.
2. The upper sediment layer, where the sediments do not move with the water, but there is an exchange of sediments from this layer and layer 1. This is also called the active layer.
3. The lower sediment layer, below layer 1. This is also called the inactive layer

According to Einsteins theory and van Rijn's formula, the sediment concentration in the water layer is a function of the shear stress and the sediments in the active layer. If multiple sediments are present, the transport capacity of each size is reduced by a factor *f*. It is often assumed that *f* is equal to the size fraction *F* of the particular size in the active layer. This seems logical, when considering the following situation: Ten sediment sizes are modelled, were each size has a different colour but has otherwise the same size, shape and density. Equilibrium of sediment deposition and erosion is assumed, and the size fractions in the bed of each size is equal. F_i is then 1/10=0.1. Modelling the ten sizes should give equal total transport capacity as modelling one size. Then $C_b = 10C_{b,i}$. The only solution is:

$$f_i = F_i$$

The question of bed armoring is addressed in Chapter 6.9.

Changes in bed grain size distribution

The vertical size of the active layer is usually kept constant. When erosion takes place, the active layer may be fed from the inactive layer below. Also, when deposition takes place, the inactive layer is increased. Sediment continuity for each fraction has to be maintained, giving the following algorithms. If deposition occurs, each fraction, f_a, in the active layer will get the following composition:

$$f_a = \frac{f_{a,0} z_a + f_d z_d}{z_a + z_d} \qquad (6.8.1)$$

Here, z is the vertical magnitude of the layer, and f_d is the fractional composition of the deposited sediment. Remember, *f* should always be between zero and unity, and this is a good test when debugging the program.

The vertical magnitude of the active layer usually stays the same during the computation, so all of the deposited sediments can not stay in the active layer. The sediments then have to be transferred to the inactive layer. The inactive layer then gets the following composition:

$$f_i = \frac{f_{i,0} z_i + f_a z_a}{z_i + z_a} \tag{6.8.2}$$

The index i denotes the inactive layer. The formula applies to each size fraction.

If erosion occur, the same principle of continuity applies. Assuming the erosion only takes place in the active layer, the following equations emerge:

$$f_a = \frac{f_{a,0} z_a - f_d z_d}{z_a - z_d} \tag{6.8.3}$$

The inactive layer changes correspondingly:

$$f_i = \frac{f_{i,0} z_i - f_a z_a}{z_i - z_a} \tag{6.8.4}$$

The equations for deposition and erosion are similar and can be used in situations where both processes occur. For example, coarse material may be deposited while finer material is eroded.

Erosion of the inactive layer

If the active layer is thin, and the time step is long, there may be a possibility that erosion would also take place in the inactive layer. There are several ways to avoid the problem:

1. Choose a larger size of the active layer
2. Choose a smaller time step
3. Include erosion of the inactive layer

Option 3 gives the following formulas for the fractions:

$$f_a = \frac{f_{a,0} z_a - f_d z_d}{z_a - z_d} \tag{6.8.5}$$

The inactive layer changes correspondingly:

$$f_i = \frac{f_{i,0} z_i - f_a z_a}{z_i - z_a} \tag{6.8.6}$$

In a natural river, the inactive layer may be composed of very coarse non-erodible material. The algorithm calculating the sediment concentration close to the bed has to take

this into account by limiting the maximum concentration, preventing erosion of the inactive layer.

6.9 Bed armoring

When multiple sediment sizes are present in the bed, some of the finer sediments will hide behind the larger particles. The probability of eroding the smaller particles will therefore be less than if the bed was composed of only fine particles. The effect is called bed armoring.

One of the most accepted methods of computing bed armoring was developed by Gessler (1971). The theory is based on the assumption that the bed shear stress, τ, has a Gaussian distribution. The probability for occurrence of a shear stress greater than the critical shear stress, τ_c, for a particle becomes:

$$P = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\frac{\tau_c}{\tau}-1} e^{-\frac{x^2}{2\sigma}} dx \qquad (6.9.1)$$

Here, σ is the standard deviation of the shear stress. Gessler found it to be 0.57.

Gessler found that this probability could be related to the mass fraction of the particle in the bed.

Eq. 6.9.1 can be evaluated numerically using formulas based on a curve-fit. The formulas are:

$$x = \frac{\frac{\tau}{\tau_c} - 1}{\sigma} \qquad (6.9.2)$$

$$T = \frac{1}{1 + 0.2316419|x|} \qquad (6.9.3)$$

$$P = T(0.31938153 + T(-0.35653782 + T(1.7814779 + T(-1.821256 + 1.3130274T)))) \qquad (6.9.4)$$

A given sediment size, *i*, will have an original fraction *F* in the bed. The fraction, *f*, after erosion can be obtained using Eq. 6.9.2-4:

$$f_i = P_i F_i \qquad (6.9.5)$$

The fractions f_i must then be increased so that the sum adds up to unity.

The theory predicts the armoring layer after a long time. If transient CFD computations are used with relatively short time step, the theory is not applicable directly. This is a topic for further research.

6.10 The effect of high sediment concentration on the water flow

Einstein and Ning Chien (1955) carried out classical experiments on the velocity profile in a flume with very high concentration of sediments. The velocity profile changes in a similar way as if the roughness of the bed had been increased. The physical explanation is that sediment particles jump up into the flow which loses inertia by pushing the sediments downstream. Einstein and Ning Chien suggested the following formula for the change in the κ parameter in the wall law:

$$\frac{\kappa}{\kappa_0} = \frac{1}{(2.5 + c)} \qquad (6.10.1)$$

where c is the volume concentration of sediments and κ_0 is 0.4. The formula is valid for concentrations up to 4 % by volume close to the bed. It is very seldom the concentrations in natural rivers exceeds this value.

6.11 Sediment transport on a sloping bed

In Chapter 6.5, the question of critical shear stress for a sediment particle on a sloping bed was addressed. When using the convection-diffusion equation to compute the sediment movement, the sediment will move parallel to the velocity vectors. However, if the bed is sloping in the direction normal to the velocity vector, the individual particles may jump slightly more downhill. The effect only applies to sediment moving close to the bed. A number of researchers (Olesen, 1987; Talmon et. al., 1995; Kikkawa et. al., 1976; Sekine and Parker, 1992) have studied the phenomena, and derived formulas for the angle, Φ, between the sediment transport vector and the velocity vector. For example, Kikkawa et. al. (1976) gives the following formula:

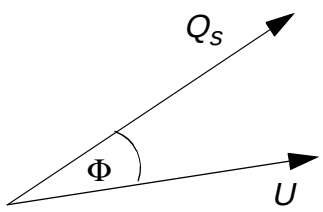


Figure 6.11.1 The angle, Φ, between the velocity vector, U, and the sediment transport vector Q_s

$$\Phi = \frac{0.6}{\sqrt{\tau^*}} \tan\alpha \qquad (6.11.1)$$

where α is the angle of the slope perpendicular to the water flow direction, and τ^* is a Shields parameter, given by:

$$\tau^* = \frac{\tau}{d(\gamma_s - \gamma_w)} \qquad (6.11.2)$$

where d is the particle size, τ is the shear tress on the bed, γ_s is the specific weight of the sediments and γ is the specific weight of water. The subscript c on the shear stress in Eq. 6.11.1 denotes the critical value for movement of the particle.

The formula is fairly straightforward to implement in a CFD program. The actual movement of the sediments then have to be computed. One solution is to change the directions of the velocity vectors according to Eq. 6.11.1 and then compute new fluxes on the cell sides. This can be used to compute new coefficients for the sediment transport equations close to the bed. The sediment transport can then be computed with the same algorithms as before.

List of symbols

Latin

C_m, C_1, C_2	constants in the k-ε model
c	concentration of sediments
D_*	parameter in van Rijn's formula for sediment concentration
d, d_s, d_{50}	mean diameter of sediment particle
d_{90}	diameter of sediment particle for which 90% is smaller
g	acceleration of gravity
h	depth of water flow
k	turbulent kinetic energy
k_s	roughness at wall
M	Strickler's friction coefficient in Manning's formula
P	pressure
P_k	term for production of turbulence
q_w	water discharge pr. unit width of canal
Sc	Schmidt number, ratio of turbulent eddy viscosity to sediment concentration diffusivity
T	parameter in van Rijn's formula for sediment concentration
U	average velocity
u	fluctuating velocity
u_*	shear velocity
x	coordinate
y	coordinate
z	coordinate

Greek

δ_{ij}	Kronecker delta: 1 if i=j, else zero
ε	dissipation rate of turbulent kinetic energy
Γ	turbulent diffusivity
κ	constant in wall function
ν	kinematic viscosity of water
ν_T	turbulent eddy viscosity
ρ_s	density of sediment
ρ_w	density of water
σ_k, σ_e	constants in the k-ε turbulence model
τ	bed shear stress

Literature

Bagnold, R. A. (1973) “The nature of saltation and of ‘bed-load’ transport in water”, Proceedings of the Royal Society of London, A332. pp. 473-504.

Booker, D. J. (2000) “Modelling and monitoring sediment transport in pool-riffle sequences”, PhD thesis, Department of Geography, University of Southampton, UK.

Brooks, H. N. (1963), discussion of "Boundary Shear Stresses in Curved Trapezoidal Channels", by A. T. Ippen and P. A. Drinker, ASCE Journal of Hydraulic Engineering, Vol. 89, No. HY3.

Bryant, T. (2000) “Sediment transport in meandering river channels with overbank flow“, PhD thesis, University of Bristol, UK.

Einstein, H. A. (1950) “The Bed-Load Function for Sediment Transportation in Open-Channel Flows”, Technical Bulletin no. 1026, US Department of Agriculture, Soil Conservation Service, Washington DC, USA

Einstein, H. A. and Ning Chien (1955) “Effects of heavy sediment concentration near the bed on velocity and sediment distribution”, University of California, Institute of Engineering Research, USA.

Engelund, F. and Hansen, E. (1967) “A monograph on sediment transport in alluvial streams”, Teknisk Forlag, Copenhagen, Denmark.

Fisher-Antze, T., Stoesser, T., Bates, P. and Olsen, N. R. B. (2001) “3D Numerical Modelling of Open-Channel Flow with Submerged Vegetation”, Accepted to Journal of Hydraulic Research.

Gessler, J. (1971) “Critical shear stress for sediment mixtures”, IAHR 14th Congress, Paris, France.

Keefer, T. N. (1971) “The relation of turbulence to diffusion in open-channel flow”, PhD dissertation, Colorado State University, USA.

Kikkawa, H., Ikeda, J. and Kitagawa, A. (1976):"Flow and bed topography in curved open channels," ASCE Journal of Hydraulic Engineering Vol.102, No. HY9, p.1327

Lysne, D. K. (1969) "Movement of sand in tunnels", ASCE Journal of Hydraulic Engineering, Vol. 95, No. 6, November.

Løvoll, 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.

Maurel, F., Bertier, C. and Hervouet, J. M. (1998) “2D numerical modelling of sediment resuspension in reservoirs”, Hydroinformatics -98, Copenhagen, Denmark.

McAnally, W. H. Jr., Letter, J. V. and Thomas, W. A. (1986) “Two and three-dimensional modeling systems for sedimentation”, Third International Symposium on River Sedimentation, Jackson, USA.

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.

Miller, A. C. (1971) “Turbulent diffusion and longitudinal dispersion measurements in a hydrodynamically rough open channel flow”, PhD dissertation, Colorado State University, USA.

Naas, S. L. (1977) “Flow behavior in alluvial channel bends”, PhD dissertation, Colorado State University, USA.

Oestberg, J. and Johansson, N. (1992) “Mathematical modelling of flow pattern”, Hydroinformatics-92, Valencia, Spain.

Olesen, K. W. (1987) “Bed topography in shallow river bends”, PhD thesis, Delft University of Technology.

Olsen, N. R. B. and Melaaen, M. C. (1993) "Three-dimensional numerical modeling of scour around cylinders", ASCE Journal of Hydraulic Engineering, Vol. 119, No. 9, September.

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. 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 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 Melaaen, M. C. (1996) "Three-dimensional numeral 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. and Kjellesvig, H. M. (1998a) "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. (1998b) "Three-dimensional numerical flow modelling for estimation of spillway capacity", IAHR Journal of Hydraulic Research, No. 5.

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. (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. 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., Hedger, R. D. and George, D. G. (2000) "3D Numerical Modelling of Microcystis Distribution in a Water Reservoir", ASCE Journal of Environmental Engineering, Vol. 126, No. 10, October.

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. (2000a) "Unstructured hexahedral 3D grids for CFD modelling in fluvial geomorphology", Hydroinformatics 2000, Iowa, USA.

Olsen, N. R. B. (2000b) "CFD modelling of bed changes during flushing of a reservoir", Hydroinformatics 2000, Iowa, USA.

Olsen, N. R. B. and Aryal, P. R. (2001) “3D CFD modelling of water flow in Khimti sand traps, Nepal”, HYDRO-POWER 2001, Bergen, Norway.

Patankar, S. V. (1980) "Numerical Heat Transfer and Fluid Flow", McGraw-Hill Book Company, New York.

van Rijn, L. C. (1987) "Mathematical modeling of morphological processes in the case of suspended sediment transport", Ph.D Thesis, Delft University of Technology.

van Rijn, L. C. (1982) “Equivalent roughness of alluvial material”, ASCE Journal of Hydraulic Engineering, Vol. 108, No. 10.

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.

Rodi, W. (1980) "Turbulence models and their application in hydraulics", IAHR State-of-the-art paper.

Roulund, A. (2000) “Three-dimensional numerical modelling of flow around a bottom mounted pile and its application to scour”, PhD Thesis, Department of Hydrodynamics and Water Resources, Technical University of Denmark.

Rouse, H (1937) "Modern Conceptions of the Mechanics of Fluid Turbulence", Transactions, ASCE, Vol. 102, Paper No. 1965.

Schlichting, H. (1979) "Boundary layer theory", McGraw-Hill Book Company.

Seed, D. (1997) "River training and channel protection - Validation of a 3D numerical model", Report SR 480, HR Wallingford, UK.

Sekine, M. and Parker, G. (1992) “Bed-load transport on transverse slope, I”, ASCE Journal of Hydraulic Engineering, No. 4.

Simons, T. J. (1974) “Verification of numerical models of Lake Ontario: Part I, Circulation in spring and early summer”, Journal of Physical Oceanography, No. 4, pp 507-523.

Spaliviero, F. and May, R. (1998) “Numerical modelling of 3D flow in hydraulic structures”, IAHR/SHSG seminar on Hydraulic Engineering, Glasgow, UK.

Suriyaarchchi, H. (2000) “CFD modelling of flow around river protection structures”, MSc thesis, Department of Hydraulic and Environmental Engineering, The Norwegian University of Science and Technology.

Tamamidis, P. and Assanis, D. N. (1993) “Evaluation of various high-order accuracy schemes with and without flux limiters”, International Journal for Numerical Methods in Fluids, Vol. 16, pp. 931-948.

Thompson, J. F., Warsi, Z. U. A. and Mastin, C. W. (1985) “Numerical Grid Generation, Foundations and Applications”, Elsevier Science Publishing Co., New York.

Vanoni, V., et al (1975) "Sedimentation Engineering", ASCE Manuals and reports on engineering practice - No. 54.

White, F. M. (1974) “Viscous Fluid Flow”, McGraw-Hill Book Company.

Wu, W., Rodi, W. and Wenka, T. (2000) "3D Numerical Modeling of Flow and Sediment Transport in Open Channels", ASCE Journal of Hydraulic Engineering, Vol. 126, No. 1, January.