

# The BEq code : A Draft Manual Version 0.1

Emanuele Cordano, Riccardo Rigon

## Abstract

The core of hydrological models is the budget of extensive quantities like mass (of water, soil, sediment particles or other chemical component), momentum and energy in a representative finite volume which usually derives from discretization of the appropriate partial differential equations. In this contribution some mathematical characteristics of these equations are emphasized and a method for the integration of this type of equation based on a new numerical scheme (*Brugnano and Casusulli, 2008; Casulli, 2008*) is shown. In particular, it is analyzed the Boussinesq equation (BEq). This equation is obtained by integrating Richards' Equation in the soil thickness if pressure head can be assumed approximately hydrostatic (*Cordano and Rigon, 2008*). The main physical observation on BEq is that the values of the conserved quantity, i.e. the soil water content, which is constrained to be positive or null, and in the amount of the driving force (pressure), which can be negative to properly describe the drying and the wetting of the catchment, are expressed as functions of the same variable. The correct formulation of the above equation should express explicitly this fact, and leads to nonlinearities (with localized discontinuities of the derivatives of the prognostic variable), that are usually ignored by most of the practitioners. The method has been further developed according to *Casulli (2008)*, and the BEq is expressed in terms of volume of water per cell, taking into account of the internal (theoretically continuous) spatial distribution of porosity and bottom topography. Thus, BEq can be solved in a coarse square grid but the solution accounts for topography and soil properties defined on a finer grid, and can be recast to maintain this information. The method is implemented in a C language code as Free software. It is here applied in a mountain catchment.

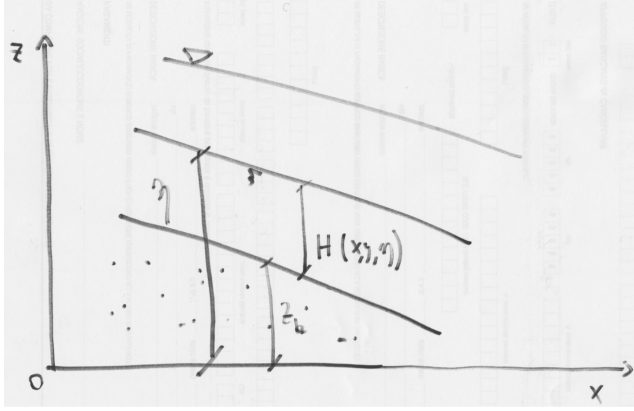


Figure 1: Graphical representation of a water-table profile:  $\eta$  elevation of water surface,  $z_b$  elevation of the bedrock or bottom,  $H$  is the thickness which can be found from  $\max\{0, \eta - z_b\}$ .

## 1 Introduction

The Boussinesq equation (BEq) had recently a revamp in literature because it was seen to represent a feasible alternative to more lumped modeling of hillslope processes at catchment scale [e.g. *Troch et al.* (2003); *Cordano and Rigon* (2013) and references therein]. This draft manual illustrates numerical program which solves the 2D Boussinesq Equation and provides useful tools for the integration of 3D Richards' Equation.

A study cases, related to a natural catchments (Matsch Valley, South Tyrol, Italy) are here presented.

## 2 2D Boussinesq equation

This section traces the various formal steps that led to a correct writing of the discretized equations.

### 2.1 2D Boussinesq equation in a continuous form

The Boussinesq equation for watertable dynamic (*Brutsaert*, 1994; *Troch et al.*, 2003; *Hilberts et al.*, 2007) is here written in 2D:

$$s \frac{\partial \eta}{\partial t} = \nabla \cdot [K_S H(\eta, x, y) \nabla \eta] + Q \quad (1)$$

where  $\eta$  is the piezometric elevation (unknown),  $t$  is time,  $\nabla$  is the space gradient operator,  $H(\eta, x, y)$  is the thickness of the aquifer which is a function of  $\eta$  and space, as shown in fig. 1,  $Q$  is a source term which also accounts for boundary conditions,  $K_S$  is the saturated hydraulic conductivity and  $s$  is porosity.

## 2.2 Simple finite-volume scheme

The equation (1) is discretized in space and time following a finite volume scheme. The domain is divided into a finite number of geometric elements (2D convex polygons and segments). The equation (1) can be written as a conservation law for a generic  $i$ -th cell:

$$s_i p_i \eta_i^{n+1} = s p_i \eta_i^n + \Delta t^n \sum_{j \in S_i} \sigma_{ij} \lambda_j u_j^{n+1} + \Delta t^n Q_i^{n+1} p_i \quad (2)$$

where the apices  $n$  and  $n + 1$ -th are referred to the  $n$ -th and  $n + 1$ -th time instants,  $\Delta t^n$  is the time step between the  $n + 1$ -th and  $n$ -th instants,  $p_i$  is the planimetric area of the  $i$ -th element and  $S_i$  is the set of the lines of the  $i$ -th polygon,  $\lambda_j$  is the length of the  $j$ -th line,  $u_j^{n+1}$  is the flux across the  $j$ -th line and  $\sigma_{ij}$  is an object defined as follows:

$$\sigma_{ij} = \begin{cases} -1 & \text{if } u_j \text{ is positive when outcoming the } i\text{-th polygon} \\ +1 & \text{if } u_j \text{ is positive when incoming the } i\text{-th polygon} \\ 0 & \text{if the } j\text{-th line is not a side of the } i\text{-th polygon} \end{cases} \quad (3)$$

The flux  $u_j^{n+1}$  are defined by a discretized form of Darcy's law:

$$u_j^{n+1} = -K_{Sj} H_j^{n+q} \frac{\eta_{r(j)}^{n+1} - \eta_{l(j)}^{n+1}}{\delta_{r(j),c(j)}} \quad (4)$$

where  $K_{Sj}$  is the saturated hydraulic conductivity related to the  $j$ -th line, the thickness  $H_j^{n+q}$  is an attribute of the  $j$ -th line at the  $n + q$ -th time instant between the two  $n$ -th and  $n + 1$ -th time instants ( $0 \leq q \leq 1$ ),  $\eta_{r(j)}^{n+1}$  is

piezometric surface at the centroid of the right-side polygon respect to the  $j$ -th line whereas  $\eta_{j(j)}^{n+1}$  is the one at the centroid of the left-side polygon,  $\delta_{r(j),c(j)}$  between the centroids of the left-side and right-side polygons. The flux  $u_j^{n+1}$  is positive when mass goes from the left-side to the right-side polygons, and negative otherwise. Thus, putting (4) into (2), it is:

$$s p_i \eta_i^{n+1} = s p_i \eta_i^n - \Delta t^n \sum_{j \in S_i} \sigma_{ij} \lambda_j K_{Sj} H_j^{n+q} \frac{\eta_{r(j)}^{n+1} - \eta_{l(j)}^{n+1}}{\delta_{r(j),c(j)}} + \Delta t^n Q_i^{n+1} p_i \quad (5)$$

Let us note that if it is  $r(j) = i$ , it becomes  $\sigma_{ij} = 1$  for (3), otherwise it is  $l(j) = i$  and  $\sigma_{ij} = -1$ , thus simplifying (5) as follows:

$$s p_i \eta_i^{n+1} = s p_i \eta_i^n + \Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^{n+q} \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} + \Delta t^n Q_i^{n+1} p_i \quad (6)$$

where  $m(i, j)$  is the index of the  $m$ -th polygon that shares the  $j$ -th line with the  $i$ -th polygon.  $m(i, j)$  is a function of the indices  $i$  and  $j$  and it is given by the topology.

In the case  $q = 0$ , the equation (6) gets linear and is so rearranged:

$$s_i p_i \eta_i^{n+1} - \Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^n \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} = s_i p_i \eta_i^n + \Delta t^n Q_i^{n+1} p_i \quad (7)$$

where  $Q_i^{n+1}$  is assumed to be independent from the unknown  $\eta$ . The equation (7) is solved for each polygon, therefore (7) is organized in a equation system as follows:

$$\mathbf{A} \cdot \eta^{n+1} = \mathbf{b} \quad (8)$$

where  $\eta^{n+1}$  is the vector of the unknowns for each polygon,  $\mathbf{A}$  is linear operator and  $\mathbf{b}$  is a vector given by the right hand side of (7). The vector  $\mathbf{b}$  is thus defined as follows:

$$b_i = s p_i \eta_i^n + \Delta t^n Q_i^{n+1} p_i \quad (9)$$

and the operator  $\mathbf{A}$  is defined as follows:

$$[\mathbf{A} \cdot \eta^{n+1}]_i = s p_i \eta_i^{n+1} - \Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^n \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} \quad (10)$$

The operator  $\mathbf{A}$  is linear and symmetric, thus the equation system (8) can be solved with the Conjugate Gradient method (*Schewchuk, 1994*).

### 2.3 *Casulli* (2008)'s Modified discretized equation

The resolution of (7) could provide negative values of water-table thickness, specially during a drying process coupled with a relatively large time step  $\Delta t$ . The time derivative is correctly referred to the variation of water volume in a point or in a cell, if the water surface varies lower than the bottom elevation, the cell is empty and the water volume is null. Therefore, the equation (7) is rewritten in the new following form:

$$\begin{aligned} s_i p_i \eta_i^{n+1} - s p_i z_{b_i} - \Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^n \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} = \\ = s_i p_i \eta_i^n - s p_i z_{b_i} + \Delta t^n Q_i^{n+1} p_i \end{aligned} \quad (11)$$

where  $z_{b_i}$  is the elevation of the node of the  $i$ -th cells. The first two terms of the left hand side represents the amount of water volume stored at the time instant  $n+1$  whereas the first two terms of the right hand side corresponds to the stored water volume at the previous time instant  $n$ . The water volume are positive quantities and negative values of water volume which can be obtained by solving (11), thus the equation (11) is adjusted as follows:

$$\begin{aligned} \max\{0, s_i p_i \eta_i^{n+1} - s p_i z_{b_i}\} - \Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^n \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} = \\ = \max\{0, s p_i \eta_i^n - s p_i z_{b_i}\} + \Delta t^n Q_i^{n+1} p_i \end{aligned} \quad (12)$$

where in case of  $\eta_i < z_b$  the water volume is zero. The equation (12) is rewritten after introducing the following functions:

$$W_i(\eta_i) = \begin{cases} s p_i & \eta_i \geq z_{b_i} \\ 0 & \eta_i < z_{b_i} \end{cases} \quad (13)$$

and becomes:

$$\begin{aligned} \eta_i^{n+1} W_i(\eta_i^{n+1}) - \Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^n \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} = \\ = \eta_i^n W_i(\eta_i^n) - z_{b_i} [W_i(\eta_i^n) - W_i(\eta_i^{n+1})] + \Delta t^n Q_i^{n+1} p_i \end{aligned} \quad (14)$$

The equation (14) can be rewritten in a matrix format as follows:

$$[\mathbf{P}(\eta^{n+1}) + \mathbf{T}] \cdot \eta^{n+1} = \mathbf{b2}(\eta^{n+1}) \quad (15)$$

where  $\eta^{n+1}$  is the vector of the unknowns for each polygon,  $\mathbf{T}$  and  $\mathbf{P}(\eta^{n+1})$  are symmetric matrices and  $\mathbf{b}$  is a vector given by the right hand side of (14). The vector  $\mathbf{b2}$  is defined as follows:

$$b2_i(\eta_i^{n+1}) = \eta_i^n W_i(\eta_i^n) - z_{bi} [W_i(\eta_i^n) - W_i(\eta_i^{n+1})] + \Delta t^n Q_i^{n+1} p_i \quad (16)$$

and the matrix  $\mathbf{T}$  is defined as follows:

$$[\mathbf{T} \cdot \eta^{n+1}]_i = -\Delta t^n \sum_{j \in S_i} \lambda_j K_S H_j^n \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} \quad (17)$$

then the matrix  $\mathbf{P}$  is a diagonal matrix defined as follows:

$$P_{i,j}(\eta^{n+1}) = \begin{cases} W_i(\eta_i^{n+1}) & i = j \\ 0 & i \neq j \end{cases} \quad (18)$$

The equation system (19) is then solved with a Picard reiterative method:

$$[\mathbf{P}(\eta^{n+1}) + \mathbf{T}] \cdot \eta^{n+1} = \mathbf{b2} \quad (19)$$

where  $m$  is the reiteration level.

## 2.4 *Casulli (2008)*'s Modified discretized equation with variable bottom

The resolution of Boussinesq Equation is further improved taking into account low-scale variability of the bottom surface which is not further planar. To do so, some functions need to be defined.

- **3D porosity**  $P(x, y, z)$ : is a function defining the porosity also distributed in the soil thickness. Assuming an uniform vertical profile of porosity, it is defined as follows:

$$P(x, y, z) = \begin{cases} s(x, y) & z \geq z_{bi} \\ 0 & z < z_{bi} \end{cases} \quad (20)$$

- **Wet area in the  $i$ -th cell**  $W s_i(\eta_i)$  : This the area covered by water in the  $i$ -th cell within a certain water surface elevation  $\eta_i$  ([L<sup>2</sup>]:

$$W s_i(\eta_i) = \int_{p_i} P(x, y, \eta_i) dx dy \quad (21)$$

where  $p_i$  is the planimetric area of the  $i$ -th cell.

- **Water depth**  $Hw_i(x, y, \eta)$  [L]: This is the water thickness in a generic point  $(x, y)$  of the domain defined as follows:

$$Hw_i(x, y, \eta) = \int_{-\infty}^{\eta_i} P(x, y, z) dz = s(x, y) \cdot H(x, y, \eta) \quad (22)$$

where  $\eta$  is the water surface elevation,  $x$  and  $y$  are the planimetric coordinates and  $H(x, y, \eta)$  is the water-table thickness.

- **Stored Water Volume of  $i$ -th cell in function of  $\eta_i$**   $V_i(\eta_i)$ : the water volume is then introduced and calculated as follows:

$$V_i(\eta_i) = \int_{-\infty}^{\eta_i} W s_i(z) dz = \int_{p_i} Hw(x, y, \eta_i) dx dy \quad (23)$$

- **Vertical area over the  $j$ -th, line**  $A_j(\eta_j)$ : This is the vertical area shared by two adjacent cells,  $r(j)$ -th and the  $c(j)$ -th ones (respectively on the right and on the left of the  $j$ -th line)

$$A_j(\eta_{r(j),c(j)}) = \int_{l_1}^{l_2} H(x(l), y(l), \eta_{r(j),c(j)}) dl \quad (24)$$

where  $l$  is a parametric coordinate,  $l_1$  and  $l_2$  are values of  $l$  related to the edges of the  $j$ -th line,  $x(l)$  and  $y(l)$  are parametric laws which returns the coordinates of a generic point of the  $j$ -th line and  $\eta_{r(j),c(j)}$  is an average value between  $\eta_{r(j)}$  and  $\eta_{c(j)}$ .

Consequently, the discretized Boussinesq Equation (12) is modified as follows:

$$\begin{aligned} V_i(\eta_i^{n+1}) - \Delta t^n \sum_{j \in S_i} K_S A_j(\eta_{i,m(i,j)}^n) \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} = \\ = V_i(\eta_i^n) + \Delta t^n Q_i^{n+1} p_i \end{aligned} \quad (25)$$

where, in the particular case of cells with horizontal bottoms, the functions  $V_i(\eta_i)$  and  $A_j(\eta_{i,m(i,j)})$  are defined as follows:

$$V_i(\eta_i) = (\eta_i - z_{bi}) W_i(\eta_i) \quad (26)$$

and

$$A_j(\eta_{i,m(i,j)}) = \lambda_j H_j \quad (27)$$

The equation (25) can be organized in the system algebraic equation:

$$\mathbf{V}(\eta^{n+1}) + \mathbf{T} \cdot \eta^{n+1} = \mathbf{b3} \quad (28)$$

where  $\eta^{n+1}$  is the vector of the unknowns for each polygon,  $\mathbf{T}$  is a symmetric matrix,  $\mathbf{V}(\eta^{n+1})$  is the vector of the water volume  $V_i(\eta_i^{n+1})$  for each  $i$ -th polygon and  $\mathbf{b3}$  is a vector given by the right hand side of (25). The vector  $\mathbf{b2}$  is defined as follows:

$$b3_i = V_i(\eta_i^n) + \Delta t^n Q_i^{n+1} p_i \quad (29)$$



and the matrix  $\mathbf{T}$  is defined as follows:

$$[\mathbf{T} \cdot \eta^{n+1}]_i = -\Delta t^n \sum_{j \in S_i} K_S A_j(\eta_{i,m(i,j)}^n) \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} \quad (30)$$

The solution of (25) is obtained with the Newton-like iterative method (*Casulli, 2008*):

$${}^{m+1}\eta^{n+1} = {}^m\eta^{n+1} - [\mathbf{Ws}({}^m\eta^{n+1}) + \mathbf{T}]^{-1} [\mathbf{V}({}^m\eta^{n+1}) + \mathbf{T} \cdot {}^m\eta^{n+1} - \mathbf{b3}] \quad (31)$$

where  $m$  is the reiteration level. The introduced vector is  $\mathbf{Ws}({}^m\eta^{n+1})$  is the vector of the wet area for each cell, in fact from (32), it is:

$$Ws_i(\psi_i) = \frac{dV_i(\eta_i)}{d\eta_i} \quad (32)$$

The solution of (31) is unique and can be implemented (*Casulli, 2008*) .

## 2.5 Dirichlet Cells

Several analytic solutions of the Boussinesq Equation (*Hogarth and Parlange, 1999*) are coupled with Dirichlet boundary coditions which means that in a certain boundary nodes the water surface is an a priori known function of time. In several cases this condition is applied on the interface between water-table and rivers,lakes, reservoirs where water surface rapidly moves driven by external forcings.

To account this, the system (25) can be rewritten for each  $i$ -th cell:

$$\begin{aligned} (Ws_i({}^m\eta_i^{n+1}) + T_{i,i}) ({}^m\eta_i^{n+1} - {}^{m+1}\eta_i^{n+1}) + \sum_{j \in S_i} T_{i,m(i,j)} ({}^m\eta_{m(i,j)}^{n+1} - {}^{m+1}\eta_{m(i,j)}^{n+1}) = \\ = V_i({}^m\eta_i^{n+1}) + T_{i,i} {}^m\eta_i^{n+1} + \sum_{j \in S_i} T_{i,m(i,j)} {}^m\eta_{m(i,j)}^{n+1} - b3_i \end{aligned} \quad (33)$$

The equation (33) is not verified in the polygons where a Dirichlet condition is posed and is replaced by the following equation:

$${}^{m+1}\eta_i^{n+1} = h_i^{n+1} \quad (34)$$

where the  $i$ -th is here a Dirichlet cells and  $hf_i^{n+1}$  is the surface water elevation of the cell at the  $n + 1$ -th time instant. If the  $i$ -th cells is not a Dirichlet cell but is near one or more Dirichlet cells, the equation 33 is here rewritten:

$$\begin{aligned} (W_{S_i}(^m\eta_i^{n+1}) + \mathbf{T}_{i,i}^{ND})(^m\eta_i^{n+1} - ^{m+1}\eta_i^{n+1}) + \sum_{j \in S_i} \mathbf{T}_{i,m(i,j)}^{ND} (^m\eta_{m(i,j)}^{n+1} - ^{m+1}\eta_{m(i,j)}^{n+1}) = (35) \\ = V_i(^m\eta_i^{n+1}) + \mathbf{T}_{i,i}^{ND} ^m\eta_i^{n+1} + \sum_{j \in S_i} \mathbf{T}_{i,m(i,j)}^{ND} ^m\eta_{m(i,j)}^{n+1} + \sum_{j \in S_i} \mathbf{T}_{i,m(i,j)}^D h_{m(i,j)}^{n+1} - b3_i \end{aligned}$$

where the matrix  $\mathbf{T}$  is split into two components:  $\mathbf{T}^{ND}$  which is a sparse matrix whose non-zero entries are in the central diagonal and at the location  $i.m(i, j)$  where  $i$ -th and  $m(i, j)$  are non-Dirichlet cells and  $\mathbf{T}^D$  whose nonzero entries correspond to the connectivity between a Dirichlet and a non-Dirichlet cells. Then, it is:

$$\mathbf{T} = \mathbf{T}^{ND} + \mathbf{T}^D \quad (36)$$

In the presence of Dirichlet cells, the algebraic equation system (31) must be modified as regards the Dirichlet and neighboring cells.

## 2.6 Free drainage condition at the outlet cells as sink terms

In the previous subsections, Boussinesq Equation is solved with no-flux conditions on the whole boundary. In a case of a catchment this is reasonable except at the outlet where a soil water discharge occurs. Due to the symmetric mathematical structure of (1), the boundary can be assumed impermeable and the discharge at the outlet can be expressed as a sink term, as follows:

$$s \frac{\partial H(\eta, x, y)}{\partial t} = \nabla \cdot [K_S H(\eta, x, y) \nabla \eta] + Q - Q_R \quad (37)$$

where  $Q_R$  is the discharge which is non-null only at the outlet and has the same dimension of  $Q$  [ $L^2 T^{-1}$ ]. The quantity  $Q_R$  can be defined in the whole domain is expressed as follows:

$$Q_R = \int_{\Gamma} Q_{RN}(\xi) \delta(x - x(\xi)) \delta(y - y(\xi)) d\xi \quad (38)$$

where  $\Gamma$  is the closed boundary line delimiting the catchment (domain),  $\xi$  is a curvilinear metric coordinate along the boundary line. The expression

38 is an integral over the boundary line of the quantity  $Q_{RN}$  [ $L^2 T^{-1}$ ] which is the net discharge per unit of boundary length multiplied by two Delta Dirac functions ( $[L^{-1}]$  each). The relations  $x(\xi)$  and  $y(\xi)$  are the parametric equation of the boundary line and depend on its shape. The quantity  $Q_R$  is null in the most of domain and infinity in a certain portion of the boundary. But, if we consider a generic area  $\Omega(\Gamma p)$  of the domain containing a portion  $\Gamma p$  ( $\Gamma p \subseteq \Gamma$ ) of the boundary line, we can calculate the total discharge  $Q_{RTOT}$  [ $L^3 T^{-1}$ ] crossing the generic line  $\Gamma p$  as:

$$Q_{RTOT} = \int_{\Gamma p} Q_{RN}(\xi) d\xi \quad (39)$$

and we can also verify by the properties of the Delta Dirac function that:

$$Q_{RTOT} = \int_{\Omega(\Gamma p)} Q_R dx dy = \int_{\Omega(\Gamma p)} \int_{\Gamma} Q_{RN}(\xi) \delta(x - x(\xi)) \delta(y - y(\xi)) d\xi dx dy \quad (40)$$

The equation (39) and (40) are equivalent and are applied when (37) is discretized in the cells near the boundary.

The function  $Q_{RN}$  can have null or finite values and must be defined as a an external condition through a particular algebraic formula. Assuming that  $O \subset \Omega$  is the outlet width, the simplest formula which can be adopted is a power-law function:

$$Q_{RN} = \begin{cases} \Xi(x(\xi), y(\xi)) \cdot H^p & (x(\xi), y(\xi)) \in O \\ 0 & (x(\xi), y(\xi)) \notin O \end{cases} \quad (41)$$

where  $p$  is an empirical exponent (dimensionless) and  $\Xi(x(\xi), y(\xi))$  [ $L^{2-p} T^{-1}$ ] is an empirical positive coefficient resuming flow properties of the outlet.

Then, we can calculate by 41 and 39 the total discharge  $Q_{RTOTj}$  through the  $j$ -th line belonging to the outlet ( $\lambda_j \subset O$ ) as follows:

$$Q_{RTOTj} = \int_{\lambda_j} Q_{RN}(\xi) d\xi \quad (42)$$

Finally, the discretized equation (25) is corrected as follows:

$$\begin{aligned} V_i(\eta_i^{n+1}) - \Delta t^n \sum_{j \in S_i} K_S A_j(\eta_{i,m(i,j)}^n) \frac{\eta_{m(i,j)}^{n+1} - \eta_i^{n+1}}{\delta_{i,m(i,j)}} = \\ = V_i(\eta_i^n) + \Delta t^n Q_i^{n+1} p_i - \sum_{j \in O_i} \Delta t^n Q_{RTOTj}^n \end{aligned} \quad (43)$$

where  $O_i$  is the subset of lines which are edges of the  $i$ -th polygon and are portion of the outlet  $O$  ( $O_i \subseteq O$ ). The known-term vector  $b3$  defined by (44) is modified as follows:

$$b3_i = V_i(\eta_i^n) + \Delta t^n Q_i^{n+1} p_i - \sum_{j \in O_i} \Delta t^n Q_{RTOTj}^n \quad (44)$$

For the convergence of the Newton-like iterations, the sum of the elements of the vector  $b3$  are recommended to be positive, at least in the cells around the outlets. This affects the choice of the time step  $\Delta t^n$ .

The reader note that the discharge at the outlet is here implemented in an explicit way as a sink term. The possibility to treat the darainage in an implicit way must be investigated.

## 2.7 Surface water flow - added on Nov 5, 2009

The Boussinesq model is extended to the description of surface flow according to (Casulli, 2008). In the following some passages are shown, both surface equation solved by 2D De Saint Venant Equation and subsurface flow solved with Boussinesq Equation are lumped in an unique equation which is than discretized. The mass balance of a cell in a discretized form is described by The equation (1) can be written as a conservation law for a generic  $i$ -th cell:

$$\begin{aligned} V_i(\eta_i^{n+1}) = & V_i(\eta_i^n) + \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SURFj}(\eta_{i,m(i,j)}^n) v_{SURFj}^{n+1} + \\ & + \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SUBSj}(\eta_{i,m(i,j)}^n)^{n+1} v_{SUBSj}^{n+1} + \Delta t^n Q_i^{n+q} p_i \end{aligned} \quad (45)$$

where the volume function  $V_i(\eta_i^{n+1})$  accounts for both surface and subsurface water. The functions  $A_{SURFj}(\eta_{i,m(i,j)}^n)$  and  $A_{SUBSj}(\eta_{i,m(i,j)}^n)$  are vertical surface and subsurface areas respectively and they are calculated according to 24:

- **Vertical surface area over the  $j$ -th, line  $A_{SURFj}(\eta_j)$ :** This is the vertical area shared by two adjacent cells,  $r(j)$ -th and the  $c(j)$ -th ones (respectively on the right and on the left of the  $j$ -th line)

$$A_{SURFj}(\eta_{r(j),c(j)}) = \int_{l_1}^{l_2} \max\{\eta_{r(j),c(j)}, z_s(x, y)\} dl \quad (46)$$

where  $z_s(x, y)$  is the elevation of the terrain surface.

- **Vertical subsurface area over the  $j$ -th, line  $A_{SUBSj}(\eta_j)$ :** This is the vertical area shared by two adjacent cells,  $r(j)$ -th and the  $c(j)$ -th ones (respectively on the right and on the left of the  $j$ -th line)

$$A_{SUBSj}(\eta_{r(j),c(j)}) = \int_{l_1}^{l_2} \max\{\min\{\eta_{r(j),c(j)}, z_s(x, y)\}, z_b(x, y)\} dl \quad (47)$$

where  $z_b(x, y)$  is the elevation of the bedrock surface.

The quantities  $v_{SURFj}^{n+1}$  and  $v_{SUBSj}^{n+1}$  are the surface and subsurface velocities normal to the  $j$ -th line and can be calculated as follows:

- **Subsurface velocity the  $j$ -th, line  $v_{SUBSj}^{n+1}$**  obtained by Darcy' law:

$$v_{SUBSj}^{n+1} = -K_{Sj} \frac{\eta_{r(j)}^{n+1} - \eta_{l(j)}^{n+1}}{\delta_{r(j),c(j)}} \quad (48)$$

- **Surface velocity the  $j$ -th, line  $v_{SURFj}^{n+1}$**  : obtained by momentum budget equation for surface water and discretized according to *Casulli, 2008* :

$$v_{SURFj}^{n+1} = \frac{A_{SURFj}(\eta_{r(j),c(j)})}{\lambda_j \Delta t^n} \left( v_{SURFj}^n - g \Delta t^n \frac{\eta_{r(j)}^{n+1} - \eta_{l(j)}^{n+1}}{\delta_{r(j),c(j)}} \right) \cdot \left[ \frac{A_{SURFj}(\eta_{r(j),c(j)})}{\lambda_j \Delta t^n} + \gamma (v_{SURFj}^n)^{p-1} \right]^{-1} \quad (49)$$

where  $g$  is the gravity acceleration [ $L T^{-2}$ ] ( $9.81 \text{ m s}^{-2}$ ), the coefficient  $\gamma$  [ $L^{2-p} T^{-2+p}$ ] and the exponent  $p$  are quantities related to bottom surface water dissipation and they depend on land use and terrain roughness.

The writing of equation 49 can be simplified as follows:

$$v_{SURFj}^{n+1} = F_j^n v_{SURFj}^n - g \Delta t^n F_j^n \frac{\eta_{r(j)}^{n+1} - \eta_{l(j)}^{n+1}}{\delta_{r(j),c(j)}} \quad (50)$$

where the quantity  $F_j^n$  (dimensionless) is obtained as follows:

$$F_j^n = \frac{A_{SURFj}(\eta_{r(j),c(j)})}{\lambda_j \Delta t^n} \left[ \frac{A_{SURFj}(\eta_{r(j),c(j)})}{\lambda_j \Delta t^n} + \gamma (v_{SURFj}^n)^{p-1} \right]^{-1} \quad (51)$$

Finally, the surface velocity can be split into two components: the first is ‘called “symmetric” and depends on  $\eta^n$ , the second called “asymmetric” only depends the state of the previous time step. Then, it is:

$$v_{SSj}^{n+1} = -g \Delta t^n F_j^n \frac{\eta_{r(j)}^{n+1} - \eta_{l(j)}^{n+1}}{\delta_{r(j),c(j)}} \quad (52)$$

$$v_{SAj}^{n+1} = F_j^n v_{SURFj}^n \quad (53)$$

where  $v_{SSj}^{n+1}$  and  $v_{SAj}^{n+1}$  are the “symetric” and “asymmetric” surface velocities ( $v_{SURFj}^{n+1} = v_{SSj}^{n+1} + v_{SAj}^{n+1}$ ). The equation (45) is then raarranged:

$$\begin{aligned} & V_i(\eta_i^{n+1}) - \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SURFj}(\eta_{i,m(i,j)}^n) v_{SSj}^{n+1} + \\ & - \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SUBSj}(\eta_{i,m(i,j)}^n)^{n+1} v_{SUBSj}^{n+1} = \\ & = V_i(\eta_i^n) + \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SURFj}(\eta_{i,m(i,j)}^n) v_{SAj}^{n+1} + \Delta t^n Q_i^{n+q} p_i \end{aligned} \quad (54)$$

Then, it is assumed:

$$\begin{aligned} [\mathbf{T} \cdot \eta^{n+1}]_i &= -\Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SURFj}(\eta_{i,m(i,j)}^n) v_{SSj}^{n+1} + \\ & - \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SUBSj}(\eta_{i,m(i,j)}^n)^{n+1} v_{SUBSj}^{n+1} \end{aligned} \quad (55)$$

and

$$b3_i = V_i(\eta_i^n) + \Delta t^n \sum_{j \in S_i} \sigma_{ij} A_{SURFj}(\eta_{i,m(i,j)}^n) v_{SAj}^{n+1} + \Delta t^n Q_i^{n+q} \quad (56)$$

Consequently, equation (45) is written as a weakly-nonlinear equation system (*Casulli*, 2008) like (28) :

$$\mathbf{V}(\eta^{n+1}) + \mathbf{T} \cdot \eta^{n+1} = \mathbf{b3}$$

The system is solved with the Newton-like iteration method according to *Casulli* (2008).

## 2.8 Notes

The function water volume  $V_i(\eta_i)$  and wet area  $W s_i(\eta_i)$  defined by (32) and (21) can be modified and consider the water content in the above unsaturated zone according to *Hilberts et al.* (2005, 2007); *Cordano and Rigon* (2008), i.e. the soil water pressure is hydrostatically distributed in the whole soil. Nevertheless, the equation (31) always converges to the unique, only if the function  $V_i(\eta_i)$  increases with  $\eta_i$  from 0 to infinity and is upward concave (*Casulli*, 2008). The Newton iterative method is inadequate for confined aquifer where water surface  $\eta_i$  can exceed the elevation of the water-table thickness and other alterative methods are required.

The hydraulic transmissivity  $K_S A_j(\eta_{i,m(i,j)}^n)$  between the  $i$ -th and the  $m(i, j)$ -th cells is express function of  $\eta_i$  and  $\eta_m(i, j)$  as follows:

$$K_S A_j(\eta_{i,m(i,j)}^n) = \max \{ K_S A_j(\eta_i^n), K_S A_j(\eta_{m(i,j)}^n) \} \quad (57)$$

where the “wetter” cells dominates, otherwise other possible formulas for transmissivity ara applicable:

$$K_S A_j(\eta_{i,m(i,j)}^n) = \frac{K_S A_j(\eta_i^n) + K_S A_j(\eta_{m(i,j)}^n)}{2} \quad (58)$$

or

$$K_S A_j(\eta_{i,m(i,j)}^n) = \min \left\{ \frac{K_S A_j(\eta_i^n) + K_S A_j(\eta_{m(i,j)}^n)}{2}, \max \{ K_S A_j(\eta_i^n), K_S A_j(\eta_{m(i,j)}^n) \} \right\} \quad (59)$$

All of these formulas affect the transient regimes and must be checked with real data or exact analytical solutions of Boussinesq Equation since they exist).

## 3 The space discretization in the Boussinesq Model

The Boussinesq Model discretizes (1) and solves its discretized form for every cell of the mash. If we consider the equation (25) as the discretized form of Boussinsq Equation, the domain is divided into two meshes. In a coarse mesh, Boussinesq Equation is discretized, in the finer one, all topographical

data and soil properties are contained and utilized to calculate the stored water volume and the wet area for each coarse cell. At the actual state, the code works on raster maps in a grass or esriascii format but its architecture is thought to manage also unstructured meshes.

Before solving numerically the Boussinesq Equation, some classes and some objects must be defined and introduced. These classes are referred to the geometric information of the resolution mesh, “C” data structure types and they are inspired to the FluidTurtle functions (*Rigon and Zanotti, 2002*). Some objects useful for the numerical solution of (25), like the two meshes, are then obtained by a realization of these classes. The realization of these classes creates in the models to create specific objects for the numerical finite-volume solutions.

### 3.1 Basic Classes for one square grid

Before introducing the whole grid (or mesh), the model defines the following basic classes:

- **POINT** : contains an integer value (which is an index) and three floating point values which are the cartesian coordinates;
- **LINE** : contains an integer value (which is an index), two points which are the start and ends points and a floating point value which is the length of the line segment
- **LINEVECTOR**: is an array of LINE classes where the index of each line corresponds to the position of the line in the LINEVECTOR.

Since each line of the mesh is considered as a LINE class object with a pre-defined LINEVECTOR class object, each segment of the mesh is numbered. So, other classes are created using only the line indices:

- **POLYGON** contains an integer value (which is an index), a vector of integer numbers (LONGVECTOR in the FluidTurtle formalism (*Rigon and Zanotti, 2002*)) which are the indices of the edges, a POINT class which is the centroid of the polygon (*Casulli, 2008*) and the floating point value which is the topographic area of the polygon;
- **POLYGONVECTOR** : is an array of POLYGON classes where the index of each polygon corresponds to the position of the polygon in the POLYGONVECTOR.



Then, we can also consider that all polygons in the mesh and are all numbered in a POLYGONVECTOR object. Consequently, a generic property, like water surface or bottom elevation which is defined in a line segment or in a polygon cell, can be saved as an array where the value at the  $i$ -th position is related to the  $i$ -th line segment or polygon respectively.

Further information about the connections among neighboring polygons are needed to solve (25) and can be obtained by the POLYGONVECTOR object by checking the indices of the common edges. A new class is here introduces:

- **polygon\_connection\_attributes** contains a vector of the indices of the neighbouring polygons and a vector of floating point values which are the distance from the centroid to the centroid of the neighbouring polygon (the size of the two vectors is given by the number of the edges and the elements are ordered following the order of the edge indices in the related POLYGON object)(If the edge belongs to the boundary of the mesh, the index and the centroid distance is replaced by conventional boundary-indicator negative values) .

As processed for LINEVECTOR and POLYGONVECTOR classes, the array of *polygon\_connection\_attributes* is here introduced:

- **polygon\_connection\_attribute\_array**: is an array of several *polygon\_connection\_attributes* objects where the generic  $i$ -th element is referred to the  $i$ -th polygon of the related POLYGONVECTOR class object.

Finally, a new type is introduces to define the whole resolution grid wich is:

- **GRID**: which contains a LINEVECTOR object, a POLGNVECTOR object and a *polygon\_connection\_attribute\_array* object. (The POLYGONVECTOR and the *polygon\_connection\_attribute\_array* must be of the same size)

The class GRID can be realized in a mesh with all necessary information to solve the Boussinesq Equation. In the Boussinesq Model the mesh is extracted by a raster map excluding the null pixels. A surface filling curve crosses each non-null pixels and assign an integer index (between 1 and the number of non-null pixels ) to the crossed pixel. The polygons are square and correspond to the non-null pixels, the border between a non-null pixels and the neighboring ones is an edge, and then a segment line, thus a new class called **SQUARE\_GRID** is created and has the following fields:

- a GRID object.
- a raster map where each pixels is filled with the indices of the po index in the GRID.POLYGONVECTOR object;
- a raster map where each pixels is filled with the index of its (top) horizontal edge in the GRID.LINEVECTOR object;
- a raster map where each pixel contains the index of its (top) vertical edge in the GRID.LINEVECTOR object;
- a raster map where each pixels contains the index of its (top-left) vertex.

Then, a square mesh built with the pixels of raster maps is then solved. The raster map fiels of the class easily translate possible raster map of distributed properties (bottom elevation, water surface properties, porosity, etc..), given as raster map into vectors which are processed by the Boussinesq Model. A careful reader might note that the raster contained in the SQUARE\_GRID class have different null pixels: a null pixels could have a left or top edge which belongs to the left or top neighboring pixel and to he boundary. For this reason, the raster map of input/output data must contains null values at the borders, this avoids error due to memory allocation and access during the execution.

### 3.2 Double Square Grid

:

The class introduced in the previous subsection allows the integration of the Boussinesq Equation in a square mesh but does not take into account to use two different spatial resolution: the first for the numerical solving and the second for some spatial distributed topographic and soil properties. For simplicity, the two grids must be multiple, i.e. all the segment lines of coarse grid belongs to the lines of the fine grid and no fine cell is divided by a line segment of the coarse grid. To do this, a new class called **DOUBLESQUARE\_GRID** whose fields are the following:

- a SQUARE\_GRID which is the grid with coarse mesh;
- a SQUARE\_GRID which is the grid with fine mesh;

- a vector of matrices which contains the indices of the fine pixels belonging to each coarse cell;
- a vector of vectors which contains the indices of the fine line segments belonging to each big line

This data structure contains the two grids and the references between the coarse grid and the fine one. The coarse and fine grids are read by the input data, e.g. bottom elevation map, which must be provided at two resolutions as input data, the references between the two grids are consequently calculated. A realization of this class in an object defines the wet aerea and stored water volume function reported in (25) for each coarse cell at every time steps, and the Boussinesq Model could run.

### 3.3 Remarks

Further documentation about the implemented functions and data structures in the code is automatically generated by Doxygen and available on line in HTML format.

## 4 How to set a simulation

This section reports the necessary information to set a simulation with a Boussinesq map. Every input/output files is associated to the keys which constitutes a hashmap as already developed in the GEOTop Distributed Hydrological Model ([www.geotop.org](http://www.geotop.org)) (*Rigon et al.*, 2006).

### 4.1 List of the keys

The keys are keywords are all listed in the ascii text file **bossinesq.init** and always updated within the code . The actual list of all possible keywords is here reported in the following list:

- **I\_MORPHO\_ELEVATION\_COARSE** : Digital terrain model at the coarse grid resolution

- **I\_MORPHO\_ELEVATION\_FINE** : Digital terrain model at the fine grid resolution
- **I\_INITCOND\_WATERSURFACE\_ELEVATION** : map of initial water surface of porosity (coarse grid)
- **I\_POROSITY\_FINE** : map of porosity (fine grid)
- **I\_SOURCEMAPSERIES\_COARSE** : series of maps at different time instant (suffix)
- **I\_PARAM\_FT\_FILE** : Fluidturtle ascii files with parameters (name with extension)
- **I\_TIMES\_FT\_FILE** : Fluidturtle ascii files with time information for source/rainfalls (name with extension)
- **O\_RESUME\_FILES** : Output files containing grid resolution
- **O\_COARSEMAP\_INDEX\_CELL** : Output map containing indices of coarse cells
- **O\_COARSEMAP\_INDEX\_LINES** : Output map containing indices of coarse lines
- **O\_FINEMAP\_INDEX\_CELL** : Output map containing indices of fine cells
- **O\_FINEMAP\_INDEX\_LINES** : Output map containing indices of fine lines

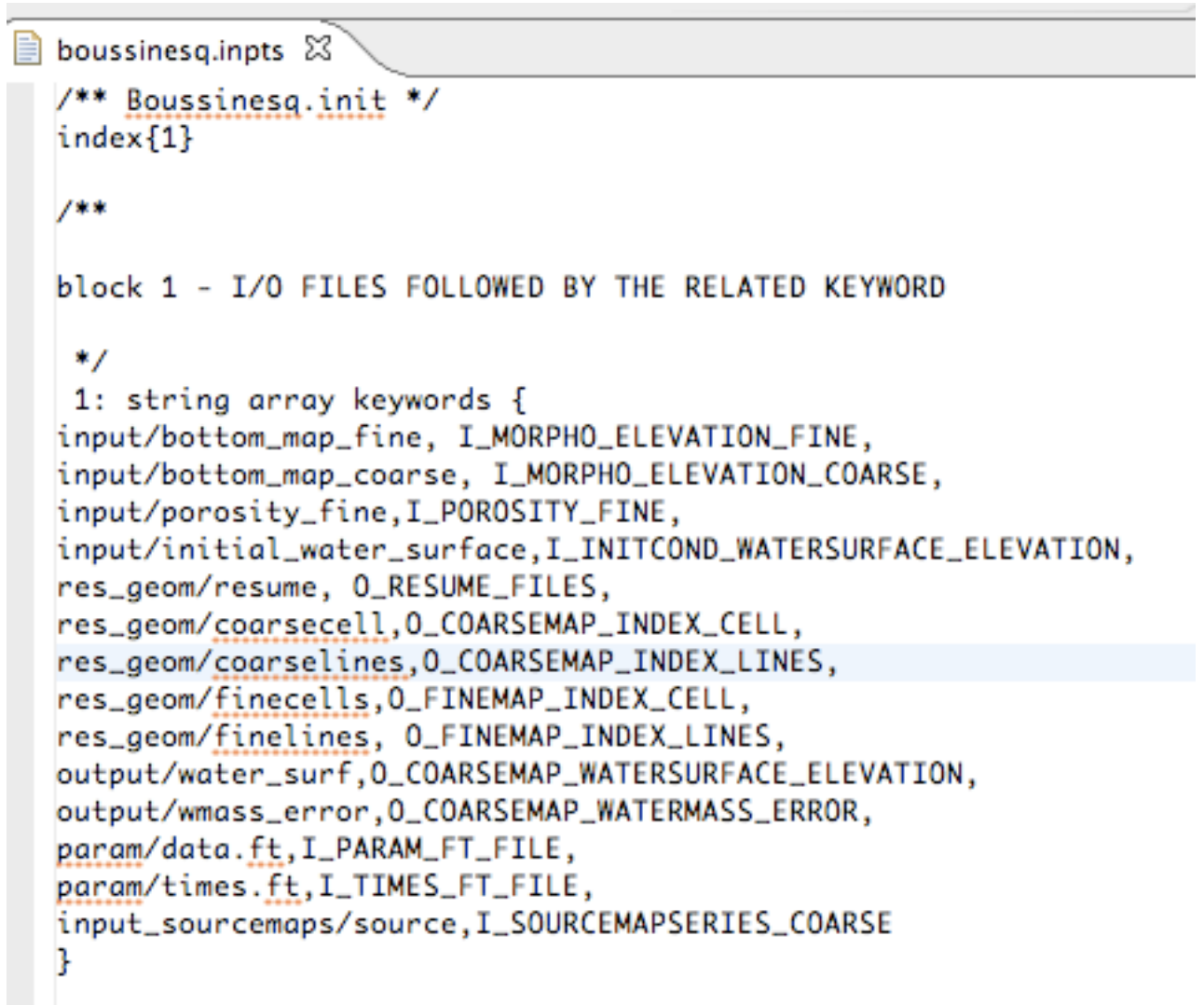
- **O\_COARSEMAP\_WATERSURFACE\_ELEVATION** : Output map containing water surface elevation
- **O\_COARSEMAP\_WATERMASS\_ERROR** : Output map containing water surface mass error

## 4.2 Example of a simulation folder

The values of the keys are strings and are assigned in a file called **boussinesq.inpts** . An example of the file *boussinesq.inpts* is illustrated in the screenshot of figure 4.2.

In this case, the data are organized in sub-directories, which the user can modify according to his/her preferences. Coherently to the configuration as shown in figure 4.2, the tree of the simulation setting is displayed 4.2.

- **input:** contains the raster maps of bottom elevation at the two resolution, the map of porosity (at the fine resolution) and the map of the initial water surface elevation (at the coarse resolution) (The maps have the extension .asc and are in grassasci format ).
- **input\_sourcemaps** : contains the raster maps of sources at certain time steps. The maps are the name give by the value of the key **L\_SOURCEMAPSERIES\_COARSE** plus the string written in the file whose key is **L\_TIMES\_FT\_FILE** .
- **param** : contains two input files whose keys are **L\_PARAM\_FT\_FILE** and **L\_TIMES\_FT\_FILE** (see figure 4.2 and 4.2). The first one contains physical and numerical parameters, in the second one there are the time instant at which source/rainfall maps (expressed in m/s) are provided and the respective suffices following the value of **L\_SOURCEMAPSERIES\_COARSE** which forms the name of the specific maps.
- **res\_geom** : contains output map and text with the properties related to the **DOUBLESQUARE\_GRID** object.
- **log** : directory where one can save the output on the terminal of the Boussinesq model.



```

boussinesq.inpts
/** Boussinesq.init */
index{1}

/**

block 1 - I/O FILES FOLLOWED BY THE RELATED KEYWORD

*/
1: string array keywords {
input/bottom_map_fine, I_MORPHO_ELEVATION_FINE,
input/bottom_map_coarse, I_MORPHO_ELEVATION_COARSE,
input/porosity_fine, I_POROSITY_FINE,
input/initial_water_surface, I_INITCOND_WATERSURFACE_ELEVATION,
res_geom/resume, O_RESUME_FILES,
res_geom/coarsecell, O_COARSEMAP_INDEX_CELL,
res_geom/coarselines, O_COARSEMAP_INDEX_LINES,
res_geom/finecells, O_FINEMAP_INDEX_CELL,
res_geom/finelines, O_FINEMAP_INDEX_LINES,
output/water_surf, O_COARSEMAP_WATERSURFACE_ELEVATION,
output/wmass_error, O_COARSEMAP_WATERMASS_ERROR,
param/data.ft, I_PARAM_FT_FILE,
param/times.ft, I_TIMES_FT_FILE,
input_sourcemaps/source, I_SOURCEMAPSERIES_COARSE
}

```

Figure 2: Screenshot of the file boussinesq.inpts
































Name	Date Modified	Size ▾	Kind
 boussinesq.init	Jun 1, 2009, 7:08 PM	4 KB	Document
 boussinesq.inpts	Today, 12:22 PM	4 KB	Document
▼  input	Jun 29, 2009, 10:47 AM	--	Folder
 bottom_map_fine.asc	Jun 29, 2009, 6:57 PM	1.8 MB	Arc A...ument
 porosity_fine.asc	Jun 29, 2009, 11:30 PM	1.5 MB	Arc A...ument
 bottom_map_coarse.asc	Jun 29, 2009, 6:57 PM	200 KB	Arc A...ument
 initial_water_surface.asc	Jun 29, 2009, 11:30 PM	200 KB	Arc A...ument
▼  input_sourcemaps	Jun 29, 2009, 10:47 AM	--	Folder
 sourceN001.asc	Jun 29, 2009, 11:30 PM	120 KB	Arc A...ument
 sourceN002.asc	Jun 29, 2009, 11:30 PM	120 KB	Arc A...ument
▼  log	Jun 29, 2009, 6:53 PM	--	Folder
 Terminal Saved Output_log.txt	Jun 29, 2009, 6:53 PM	77.3 MB	Plain text
▶  output	Jun 29, 2009, 6:49 PM	--	Folder
▼  param	Jun 10, 2009, 12:27 PM	--	Folder
 data.ft	Today, 12:34 PM	4 KB	Document
 times.ft	Jun 16, 2009, 5:23 PM	4 KB	Document
▼  res_geom	Jun 29, 2009, 9:53 AM	--	Folder
 resume_fine_lines.txt	Jun 29, 2009, 10:55 AM	60.5 MB	Plain text
 resume_fine_polygons.txt	Jun 29, 2009, 10:55 AM	50 MB	Plain text
 resume_fine_connections.txt	Jun 29, 2009, 10:55 AM	34.4 MB	Plain text
 resume_coarse_lines.txt	Jun 29, 2009, 10:55 AM	6.7 MB	Plain text
 resume_coarse_polygons.txt	Jun 29, 2009, 10:55 AM	5.4 MB	Plain text
 finelines_vertical_lines.asc	Jun 29, 2009, 10:55 AM	3.9 MB	Arc A...ument
 finelines_horizontal_lines.asc	Jun 29, 2009, 10:55 AM	3.8 MB	Arc A...ument
 finecells.asc	Jun 29, 2009, 10:55 AM	3.8 MB	Arc A...ument
 resume_coarse_connections.txt	Jun 29, 2009, 10:55 AM	3.7 MB	Plain text
 resume_c_line.txt	Jun 29, 2009, 10:55 AM	2.9 MB	Plain text
 resume_c_polygon.txt	Jun 29, 2009, 10:55 AM	2.6 MB	Plain text
 coarselines_vertical_lines.asc	Jun 29, 2009, 10:55 AM	416 KB	Arc A...ument
 coarsecell.asc	Jun 29, 2009, 10:55 AM	404 KB	Arc A...ument
 coarselines_horizontal_lines.asc	Jun 29, 2009, 10:55 AM	404 KB	Arc A...ument

Figure 3: Screenshot of the simulation directory tree

```

index{1}

/**
Files contained physical data for the Boussinesq Equation Solver
*/
/** block 1 - SCALAR DATA
#1 K_s hydraulic saturated conductivity -
#2 t_start - initial time
#3 t_end - end time
#3 dt integration time step
#4 dt results printing time steps time
#5 maximum mass error admitted [m]
#6 maximum absolute tolerance on water surface elevation [m]
#7 variation of water source elevation utilized for the first derivative of cell volume vs eta
*/
1: double array scalars {0.1,0,1e+8,1e+4,1e+4,1.0e-5,1.0e-5,1e-5}

```

Figure 4: Screenshot of the I\_PARAM\_FT\_FILE file

- **output** : which contains the results of the Models, i.e. the maps of water surface elevation and the water mass balance error for each pixels.

### 4.3 Running a simulation and results

When all input file are prepared in the simulation directory, the Boussinesq Model can be launched with following arguments:

- ‘-wpath \${working\_path} -creates\_the\_grid (-verbose)’

Then the program starts running and creates the maps of water surface elevation as thickness.

An example simulation is here reported and is applied to a mountain catchment (Saldur Creek, Matsch Valley, South Tyrol, Italy) (figure 4.3). The basin area is about 100 km<sup>2</sup>, the bottom elevation is taken equal to the digital terrain model which is available at a resolution of 20 m. The Boussinesq Equation is here solved at a resolution of 60 m, as shown in table 4.3.

The time step for the numerical integration is now fixed to 10<sup>4</sup>, the hydraulic conductivity and the porosity are assumed to be constant as reported



```
index{2}  
  
/**  
Files contained indices for rainfall/sources map  
*/  
/** instant times time at which instantaneous rainfall/sources intensity are given [s] */  
1: double array times {0.0,1e+9}  
/** string array with the suffixes */  
2: string array suffix{N001,N002}
```

Figure 5: Screenshot of the LTIMES\_FT\_FILE file

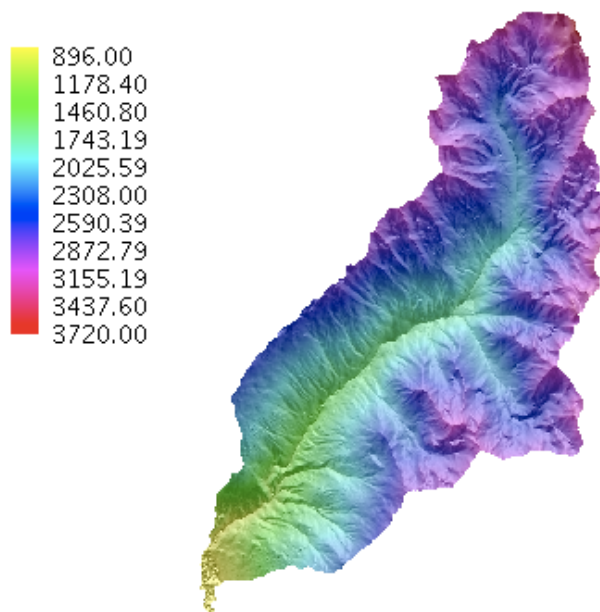


Figure 6: DTM of Saldur Creek (South Tyrol, Italy) at a 20 m resolution.

Table 1: Utilized parameters

$k_S$ [m/s]	Porosity	Area [m <sup>2</sup> ]	Fine Grid Res.[m]	Coarse Cell. Res. [m]	$\Delta t$ [s]
$10^{-1}$	0.4	$9.897 \cdot 10^7$	20.0	60.0	$10^4$

in table 4.3. Initially, it is assumed that water is uniformly distributed in the whole catchment and the initial water surface elevation is above 1 m. No rainfall and no sources are not considered and they are assumed to be zero. As expected water flows downslope very fast along the hillslopes and accumulates in the hollows. Figure 4.3 shows the water thickness (which is  $H(\eta, x, y)$  and illustrated in figure 1 ) after  $10^4$  s,  $10^5$  s,  $10^6$  s and  $10^7$  of simulation. After long times water tends to accumulate at the bottom of the catchment with ‘very big “unrealistic” values of thickness, it is reminded that all the boundary (outlet included) is assumed to be impermeable. This condition helps the user to verify the global conservation of the mass during the simulation. Once verified, a suitable boundary condition can be posed at the outlet to simulate the complete “real” behaviour of the catchment.

## 5 GNU General Public License

The model (executable file) and the source code are open-source Free Software under the GNU General Public License (GPL) 3.0 or later. See on [www.boussinesq.org](http://www.boussinesq.org) for further details.

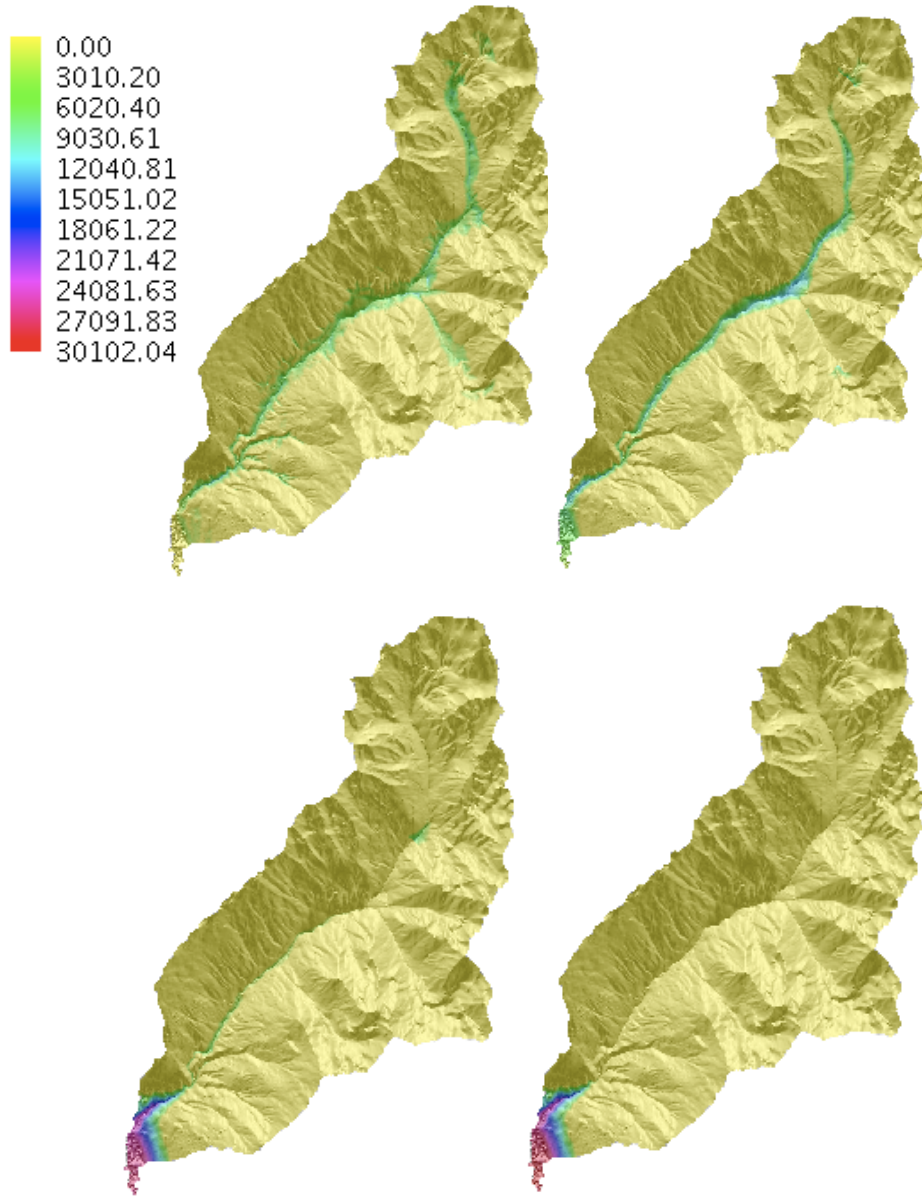


Figure 7: Maps of simulated water surface thickness expressed in  $\cdot 10^{-2}$  (centimeters) m at time  $10^4$  s (left top),  $10^5$  s (right top),  $10^6$  (left bottom) and  $10^7$  (right bottom).

## 6 List of Symbols

Symbol	Dimension	Definition
<b>b</b>	$[L^3]$	vector of known terms introduced in the equation systems (8) and defined by (9)
<b>b2</b>	$[L^3]$	vector of known terms introduced in the equation systems (15),(19) and defined by (16)
<b>b3</b>	$[L^3]$	vector of known terms introduced in the equation systems (28) and defined by (16)
$l(j)$		index of the cell on the left-hand side of the $j$ -th line segment
$m(i,j)$		index of the cell that shares the $j$ -th line segment with the $i$ -th cell
$p_i$	$[L^2]$	topographic area of the $i$ -th cell
$r(j)$		index of the cell on the right-hand side of the $j$ -th line segment
$s, s(x, y)$		porosity
$s_i$		porosity at the $i$ -th cell
$t$	$[T]$	time
$u_j^n$	$[L \ T^{-1}]$	water flux crossing the $j$ -th line segment at the $n$ -th time instant
$x, y$	$[L]$	space coordinate
$z_b$	$[L]$	soil bottom elevation
$z_{bi}$	$[L]$	soil bottom elevation at the $i$ -th cell
<b>A</b>	$[L^2]$	symmetric matrix introduced in the equation system (8) and defined by (10)
$A_i(\eta r(j), c(j))$	$[L^2]$	Vertical area over the $j$ -th line
$H, H(x, y, \eta)$	$[L]$	water-table thickness
$H_j^n$	$[L]$	water-table thickness on the $j$ -th line segment at the $n$ -th time instant
$Hw(x, y, \eta)$	$[L]$	water depth
$K_S$	$[L \ T^{-1}]$	saturated hydraulic conductivity
$P(x, y, z)$		3D porosity
$\mathbf{P}(\eta^{\mathbf{n}+1, \mathbf{m}})$	$[L^2]$	diagonal matrix defined by (18) ( <i>Brugnano and Casusulli, 2008</i> )
$Q$	$[L \ T^{-1}]$	source term (water-table discharge)

$Q_i^n$	[L T <sup>-1</sup> ]	source term (water-table discharge) at the $i$ -th cell in the $n$ -th time instant
$S_i$	*	set of the edges of the $i$ -th cell
$\mathbf{T}$	[L <sup>2</sup> ]	symmetric matrix introduced in the equation systems (15),(19) and defined by (17), this symbol is newly re-used and redefined by (??) and defined by (30)
$V_i(\eta_i)$	[L <sup>3</sup> ]	water volume stored in the $i$ -th cell defined as a function of water surface elevation $\eta_i$
$\mathbf{V}(\eta)$	[L <sup>3</sup> ]	vectors of water volume stored each cell defined as a function of vector of water surface elevation $\eta$
$W_i(\eta_i)$	[L <sup>2</sup> ]	water volume per thickness unit stored in the $i$ -th cell defined as function of water surface elevation by (??) (in case of horizontal bottom)
$W s_i(\eta_i)$	[L <sup>2</sup> ]	Wet area in the $i$ -th cell
$\delta_{iq}$	[L]	euclidean distance between the centroids of the $i$ -th and $q$ -th cells
$\eta_i$	[L]	water surface elevation at the $i$ -th cell
$\eta_{i,d}$	[L]	averaged water surface elevation between the $i$ -th and $d$ -th cells
$\eta_i^n$	[L]	water surface elevation at the $i$ -th cell in the $n$ -th time instant
$\eta^{n+1}$	[L]	vector of water surface elevation at the $(n+1)$ -th time instant in the equation system (19) and (15)
$\eta^{n+1,m}$	[L]	vector of water surface elevation at the $(n+1)$ -th time at the $m$ -th iteration level instant in the equation system (19)
$\lambda_j$	[L]	length of the $j$ -th line segment
$\sigma_{ij}$		object defined by equation (3)

$\Delta t^n$

[T]

time step between the  $n$ -th and the  $(n+1)$ -th  
time instants

## References

- Brugnano, L., and V. Casusulli (2008), Iterative solution of piecewise linear systems, *J. Sci. Comput.*, *30*(1), 463–472.
- Brutsaert, W. (1994), The unit response of groundwater outflow from a hill-slope, *Water Resour. Res.*, *30*(10), 2759–2764.
- Casulli, V. (2008), A high-resolution wetting and drying algorithm for free-surface hydrodynamics, *Int. J. Numer. Meth. Fluids*.
- Cordano, E., and R. Rigon (2008), A perturbative view on the subsurface water pressure response at hillslope scale, *Water Resour. Res.*, *44*.
- Cordano, E., and R. Rigon (2013), A mass-conservative method for the integration of the two-dimensional groundwater (boussinesq) equation, *Water Resources Research*, *49*(2), 1058–1078, doi:10.1002/wrcr.20072.
- Hilberts, A. G., P. A. Troch, and C. Paniconi (2005), Storage-dependent drainable porosity for complex hillslopes, *Water Resour. Res.*, *41*, W06,001, doi:10.1029/2004WR003725.
- Hilberts, A. G. J., P. A. Troch, C. Paniconi, and J. Boll (2007), Low-dimensional modeling of hillslope subsurface flow: Relationship between rainfall, recharge, and unsaturated storage dynamics, *Wat. Resour. Res.*, *43*, W03,445 doi:10.1029/2006WR004,964.
- Hogarth, W. L., and J. Y. Parlange (1999), Solving the boussinesq equation using solutions of the blasius equation, *Water Resour. Res.*, *35*(3), 885–887.
- Rigon, R., and F. Zanotti (2002), *The Fluid Turtle Library, Version 0.750*, CUDAM - Dipartimento di Ingegneria Civile e Ambientale Università degli Studi di Trento.
- Rigon, R., G. Bertodi, and T. Over (2006), A Distributed Hydrological Model with Coupled Water and Energy Budgets., *Journal of Hydrometeorology*, *7*(3), 371–388.
- Schewchuk, J. R. (1994), An introduction to the conjugate gradient method without the agonizing pain, *Tech. rep.*, School of Computer Science,



Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, Carnegie Mellon University Pittsburgh, PA 15213, USA.

Troch, P. A., C. Paniconi, and E. E. van Loon (2003), Hillslope-storage Boussinesq model for subsurface flow and variable source areas along complex hillslopes: 1. formulation and characteristic response, *Water Resour. Res.*, *39*(11), 1316, doi:10.1029/2002WR001728.