WAQUA User Routines

SIMONA report number 2000-05

Rijkswaterstaat
Ministerie van Infrastructuur en Waterstaat

WAQUA User Routines

Compendium for the application of User Routines in WAQUA

Version	:	1.3, June 2009
Maintenance	:	see www.helpdeskwater.nl/waqua
Copyright	:	Rijkswaterstaat

Preface

Comprehensive introduction concerning the application of WAQUA User Routines.

Contents

1 About this manual

- Purpose of this report is to provide a summary of the application of the so-called User Routines in WAQUA.
  It contains a general description of the main aspects, which are for a substantial part adopted from the following available documents:
-
– User' s Guide WAQUA, General Information, section 3.9
– User' s Guide WAQUA, Appendix A (Examples)
– User' s Guide WAQPRO, section 2.3
– User' s Guide WAQPRE, section 2.9.5
– Quick Reference Guide WAQUA, chapter 8
-
- To illustrate the benefits of the offered features, three examples are included.

2 General

2.1 Features

An increasing percentage of hydraulic modelling activities is carried out to learn more about the behaviour of in water dissolved substances and on the reactions between these substances.
The range of possible applications in this field is enormous. Examples are water quality studies, modelling flocculation processes, turbulence, resuspension, sedimentation, temperature, spreading of algae, etc.

The user routine is a feature within WAQUA that provides the investigator with the possibility to use the by WAQUA computed water movements in combination with his self-defined water quality model.

Possibilities:

- Interactions between constituents
- Interactions between constituents and properties fixed to the bottom
- Reactions between bottom related parameters
- Adjustable diffusion coefficients

The reaction kinetics as well as the involved parameters and boundary conditions are under the user’s control.

2.2 Mathematical description

The following equations govern the transport computation.
The user routine gives access to the diffusion coefficient, the global source and the global sink.

$$
\begin{aligned}
&\frac{\partial}{\partial t}(hc) + \frac{\partial}{\partial x}(huc) + \frac{\partial}{\partial y}(hvc) + \frac{\partial}{\partial z}\big(h(w - w_{val})c\big) \\
&= \frac{\partial}{\partial x}\left(D_x \frac{\partial}{\partial x}(hc)\right) + \frac{\partial}{\partial y}\left(D_y \frac{\partial}{\partial y}(hc)\right) + \frac{\partial}{\partial z}\left(D_z \frac{\partial}{\partial z}(hc)\right) \\
&\qquad + S_{+} - Lc
\end{aligned}
$$

where

c = constituent concentration in [c]
D_x, D_y = horizontal diffusion coefficient in m^2 / s
D_z = vertical diffusion coefficient in m^2 / s
S = global source (explicit) per unit area in [c] · m^3 / s / m^2
L = global sink (implicit) per unit area in m^3 / s / m^2

2.3 Data input/output

To model water quality processes input data is necessary. Apart from the water movement which is computed by WAQUA the user should be enabled to feed the model with parameters he wants to take into account.

In WAQPRE the user can control the following flexible data blocks:

- real or integer data stating for instance a reaction constant
- time series for instance describing the inflow of energy due to a fluctuating radiation of the sun
- spatial distributed data like the initial thickness of a layer of sediment on the bottom of the model
- time and space dependent data which can be used to describe the seasonal variation in the growth rate of local bottom vegetation

3D-data like the magnitude of a 3D-space dependent coefficient can be offered to the user routine in layers using the spatial distributed data block.

For detailed information refer to the User’s Guide WAQPRE,
section USERDATA_TRANSPORT.

WAQPRE will store the user defined data and create workspace within the data structure for fields of output data, defined by the user, but computed during simulation.
In this way the SDS-file will, after computation, contain results of user defined parameters.

Within the user routine the unit numbers of the WAQPRO report- and message file are known. Using this information warnings and errors occurring during execution of WASUST can be taken care of in a structured way.

2.4 Computation

By a linking procedure the user routine programmed and compiled by the user becomes part of the WAQPRO executable. The name of the routine to be made by the user is WASUST. The heading of this routine has a fixed format, refer to the chapter on WASUST in the User’s Guide WAQPRO.

Within this user routine a part of the local WAQUA data structure is made available to the user. A list of the available data within WASUST is given in the User’s Guide WAQPRO. It contains time dependent data computed by WAQUA as well as a number of fixed data, refer to common CWAUST (file cwaust.i).

Within WASUST the SIMONA processing tool SIFODE can be used to enable the user to compute the interaction between dissolved substances and parameters fixed to a given location. In general, SIFODE controls the change in the amount of bottom fixed substances.

An example in which case SIFODE should be used, is the interaction between silt, which is modelled as a dissolved constituent, and the thickness of a layer of silt in a given grid cell.

WASUST is called by the WAQUA system every half time step, just before the advection diffusion solver.

Programming the content of WASUST is in fact only limited to the creativeness of the user.

2.5 Generating procedure

Within SIMONA the procedure GEN.WAQPRO is available that:

- compiles the WASUST routine(s)
- links the results to the object modules of WAQPRO
- makes a WAQPRO executable containing the user routine

For more information on how to use the WAQPRO generating procedure also refer to the chapter on GEN.WAQPRO in the Quick Reference Guide WAQUA.

3 Quick Reference Guide

3.1 Generation of an executable

gen.waqpro file1.f [file2.f […]] [&]

There must be exactly one subroutine be present in the FORTRAN-source file(s) -- *file1.f* , etc. -- named 'wasust' with the prescripted fixed set of parameters.

Modules (subroutines / functions) may be clustered in one FORTRAN-source file.

Optionally the generation will be running in the background by specifying '&' at the end of the command string.

Warning:
Be sure there are no unintended object files (i.e. with extension '.o') in the working directory, because they will be linked as well into the executable! When it happens to be that one of these objects is referenced in 'waqpro.exe', this may lead to unexpected (erroneous) results.

3.2 Execution of waqpro

waqpro.run etc.

Note: when running parallel WAQUA/TRIWAQ, no user routines can be implemented, so 'npart' should be 1.

4 Description of WAQUA User Routines

4.1 General information

The input is based on SIMONA key-word structure. Refer to the "User's Guide WAQUA", chapter 2 ("About SIMONA") in Section 1: "General Information".

Reminder: The input file is a structured ASCII-file. From the input file only the first 258 columns are read.

4.1.1 Conventions used

For the input definition the following conventions are used:

[val] : real value[1]
[tval] : time specification in the form: *day hours:minutes* (e.g. 2 21:15). Times are given relative to midnight of a reference date, starting at 0 0:00.
[ival] : integer value
[iseq] : sequence number to indicate a point, curve, etc.
[text] : string (enclosed between quotes)
< ... > : repetition group

```
 | A
<       :   choice between A and B (A and B are mutually
 | B        exclusive)
```

& : continuation mark

In this document a part of the key-words is underlined (e.g, PRINT-OUTPUT). Only the underlined characters are significant. So the user must type at least PRINT in his input, but PRINTOUT is excepted as well.

The 'Explanation' part of the description of the various sections, subsections is divided in three columns:

KEYWORD **E** Explanation
E can be O, M, D, S, R, X.
O means key-word is optional.
M means key-word is mandatory.
D means key-word has a default value. When this key-word is omitted, the pre-processor will use the default value for the variable specified by means of this key-word.
S means this key-word is a sequential key-word: a key-word followed by an integer (e.g. P4). A sequential key-word can be used repeatedly.
R means key-word may occur more than once.
X Exactly one of a series of key-words should be given.

[1] Since all values are read in free format: integer notation (when reals are expected) will be converted to reals, so "val = 4" is identical to "val = 4.0".

4.1.2 Data fields

Data field input is to be specified in two blocks:

```
SPACE_VARYING_DATA
    GLOBAL
    LOCAL
```

SPACE_VARYING_DATA stands for any key-word representing spatial data. In GLOBAL the data for the complete field is to be given, specifying function values at all grid points. In LOCAL however the user can specify rectangular boxes in which he can change the value of the space varying data. For the case of 3D this definition is extended in such a way that the input for separate layers is possible.

4.1.2.1 GLOBAL

Global data can be specified in two ways: first by giving one value for the complete computational grid, second by giving values for each grid point. The order in which these values are to be given is specified by the layout flag.

```
GLOBAL
    LAYOUT = [ival]
    |   CONST_VALUE = [val]
   <
    |   VARIABLE_VALUES = < [val] >
```

Explanation:

LAYOUT = *[ival]* **D** Layout-indicator specifying the order in which the values from input file are assigned to the function value in a grid point. Possible values for LAYOUT and their meaning are:[2]

1. function values at grid points: [(m_1,n_1), (m_1,n_1+1) ... (m_1,n_2)], [(m_1+1,n_1) ... (m_1+1,n_2)] ... [(m_2,n_1) ... (m_2,n_2)]
- columns; first column is left; column values from bottom to top
2. function values at grid points: [(m_1,n_1), (m_1+1,n_1) ... (m_2,n_1)], [(m_1,n_1+1) ... (m_2,n_1+1)] ... [(m_1,n_2) ... (m_2,n_2)]
- rows; first row is bottom; row values from left to right
3. function values at grid points: [(m_2,n_1), (m_2,n_1+1) ... (m_2,n_2)], [(m_2-1,n_1) ... (m_2-1,n_2)] ... [(m_1,n_1) ... (m_1,n_2)]
- columns; first column is right; column values from bottom to top
4. function values at grid points: [(m_2,n_1), (m_2-1,n_1) ... (m_1,n_1)], [(m_2,n_1+1) ... (m_1,n_1+1)] ... [(m_2,n_2) ... (m_1,n_2)]
- rows; first row is bottom; row values from right to left
5. function values at grid points: [(m_1,n_2), (m_1,n_2-1) ... (m_1,n_1)], [(m_1+1,n_2) ... (m_1+1,n_1)] ... [(m_2,n_2) ... (m_2,n_1)]
- columns; first column is left; column values from top to bottom
6. function values at grid points: [(m_1,n_2), (m_1+1,n_2) ... (m_2,n_2)],
- [(m_1,n_2-1) ... (m_2,n_2-1)] ... [(m_1,n_1) ... (m_2,n_1)]
- rows; first row is top; row values from left to right

[2] Assume the limits of the box are given by (m_1, n_1) and (m_2, n_2) with $m_1 \leq m_2$ and $n_1 \leq n_2$. In the case of global input n_1 =1, n_2=NMAX, m_1 =1 and m_2= MMAX. The number of required function values is then n_{tot}*m_{tot}, where :
n_{tot}= (number of enclosed n grid points) = n_2 - n_1 +1
m_{tot}= (number of enclosed m grid points) = m_2 - m_1 +1

7. function values at grid points: [(m_2,n_2), (m_2,n_2-1) ... (m_2,n_1)],
- [(m_2-1,n_2) ... (m_2-1,n_1)] ... [(m_1,n_2) ... (m_1,n_1)]
- columns; first column is right; column values from top to bottom
8. function values at grid points: [(m_2,n_2), (m_2-1,n_2) ... (m_1,n_2)],
- [(m2,n2-1) ... (m1,n2-1)] ... [(m2,n1) ... (m1,n1)]
- rows; first row is top; row values from right to left

Default = 1

CONST_VALUE = *[val]* **D** Constant value for the complete field.
Default = 0

VARIABLE_VALUES = < *[val]* > **O** It is possible to specify a function value at each grid point .
The order in which the values are to be given is defined by means of the layout parameter.
In the case of 3D the information must be specified as a set of KMAX separate layers, each layer given according to the global layout-indicator (i.e. MMAX*NMAX*KMAX values must be specified, beginning with the top layer) .

4.1.2.2 LOCAL

In LOCAL the function values at grid points specified in GLOBAL can locally be overwritten by specifying boxes (i.e. rectangles). In the 3D-case a box is a rectangle drawn in the horizontal plane identified by the layer-index.

```
LOCAL
  < BOX:  MNMN = ( [ival], [ival] )  ( [ival], [ival] ) LAYER = [ival]
        | CONST_VALUES = [val]
       <
        | CORNER_VALUES = [val], [val], [val], [val]
       <
        | VARIABLE_VALUES = < [val] >
```

Explanation:

BOX **R** A BOX is defined by specifying its opposite corner points (m1,n1) and (m2,n2), where m1 ≤ 1m2 and n1 ≤ 2n2. In this rectangle the global function value of a "field" variable can be overwritten by new values. It is possible to define more than one box for one single "field" variable. When the rectangles defined in the boxes have common grid points, the latest values specified for those grid point will be used.
The data can be specified either by means of a single value defining all points within the box or by means of a array of data. In the latter case the data should be given according to the following scheme:

MNMN = (*[ival], [ival]*) (*[ival], [ival]*)
M Corner points of the rectangular box, specifying (m_1, n_1) (m_2, n_2), where m1 ≤ 3m2 and n1 ≤ 4n2.

LAYER = *[ival]* **O** Layer index , where 0 ≤ layer ≤ kmax. If layer is not specified or layer=0, a uniform vertical distribution is assumed. However, when the function values belong to a data-array which is defined for layers 0 until kmax, layer=0 is only valid for the upper layer and layer=-1 will define the uniform vertical distribution. As default, 3D-arrays are assumed to be defined for layers 1 until kmax, unless stated otherwise in their input description.
LAYER is only relevant in the 3D-case.

CONST_VALUES = ***[val]***

- The function at all grid points in the box gets this value.

CORNER_VALUES = ***[val], [val], [val], [val]***

- The function values at the corner points of the box are given in the following order (m_1, n_1), (m_2, n_1), (m_2, n_2), (m_1, n_2). The function values at the other grid points enclosed by the box will be determined by means of bilinear interpolation.

VARIABLE_VALUES = < ***[val]*** **>**

- Inside the box for each grid point a function value is specified. The order in which the values are to be given is set by LAYOUT under keyword GLOBAL.

For example

```
GLOBAL
    CONST_VALUES = 40.5
    LAYOUT = 4
LOCAL
    BOX:  MNMN = (10, 5), (50, 100)
            CONST_VALUES = 38
```

or

```
GLOBAL
    CONST_VALUES = 0
    LAYOUT = 3
LOCAL
    BOX:  MNMN = (10, 5), (11, 7)
            VARIABLE_VALUES =   2    2.3    2.4
                                1.9  2.0    3.2
```

4.1.3 Time series

Time series are used for boundary conditions.
There are two possibilities in SIMONA to specify time series: ’regular’ and ’irregular’.

Regular time series are given by using a time frame (FRAME), defining a time first, time interval and time last (all times in minutes elapsed from midnight of the reference date as specified in FLOW, PROBLEM, TIMEFRAME, DATE). The values must be given at constant time intervals.

In case of irregular time series a time can be specified together with the values related to this time, repeatedly. In this case the times are given in day hour:minute. A minute can be specified with a decimal value (e.g. 5.75). In this notation midnight of the reference date as specified in FLOW, PROBLEM, TIMEFRAME, DATE is 0 0:00.

All time series are interpolated during computation.

’Timeseries’ is not a (sub-) keyword, but the name of an input structure that may be embedded in other key-word structures described in this guide.
Initial values for time series must be specified.

Examples:

1. Timeseries

```
        SERIES = 'regular'
        FRAME = 100.  5.  125.
        VALUES = 5  6  7  7  3  4

2. Timeseries
        SERIES = 'irregular'
        TIME_AND_VALUES = (0 1:00.5) 2
        TIME_AND_VALUES = (0 1:10.5) 8
        TIME_AND_VALUES = (0 2:00.5) 12
```

4.2 WAQPRE-input description

4.2.1 USERDATA_TRANSPORT (optional)

The keyword USERDATA_TRANSPORT triggers the use of a user routine in the transport module. The application of these user routines is described in the corresponding section of the general information and in the user's guide for the processor WAQPRO. Subsection USERDATA_TRANSPORT has 7 subsections.

```
USERDATA_TRANSPORT
    CONTROL
    REALS
    INTEGERS
    TIMEFUNCTIONS
    INPUT_SPATIAL_DATA
    TIME_DEPENDENT_DATA
    OUTPUT_SPATIAL_DATA
```

4.2.1.1 CONTROL (mandatory)

In this subsection the length of the work array in the user routine and the type number of the user routine must be given.

```
CONTROL
    TYPE= [ival]
    LENWRK= [ival]
```

Explanation:

TYPE = *[ival]* M Type number of user routine. A type number >= 100 indicates a standard (included in the mother version) user routine. Type number 0-99 are available for a user routine built by the user.

LENWRK = *[ival]* M Parameter for the length of the work array use in the user routine. The work array has the dimensions WORK (NMAX, -2:MMAX + 3, LENWRK), where (NMAX, -2:MMAX + 3) are the dimensions needed for one data field.

4.2.1.2 REALS (optional)

In REALS the data array USER of USERDATA_TRANS in the local data structure can be filled with single real values. Positions that are not used are set to zero.

```
REALS
    < USER[iseq]= [val] >
```

Explanation:

USER *[iseq] = [val]* s Sequence number of real value, followed by value.

4.2.1.3 INTEGERS (optional)

In INTEGERS the data array IUSER of USERDATA_TRANS in the local data structure can be filled with single integer values. Positions that are not used are set to zero.

```
INTEGERS
    < IUSER[iseq]= [ival] >
```

Explanation:

IUSER *[iseq] = [ival]* s Sequence number of integer value, followed by value.

4.2.1.4 TIMEFUNCTIONS (optional)

The function to be defined can be multi valued (f = (f_i), met i=1,..,n). The function f_i can be described in two ways, namely by the use of ’TIMESERIES’ or by the use of ’FOURIERSERIES’. In the user routine an array with function values is passed, corresponding to the simulation time step.

```
TIMEFUNCTIONS
    TIMESERIES
    FOURIER
```

TIMESERIES (optional)

```
TIMESERIES
    < TS:  ISEQ= [ival]
           SERIES= [text]
           | FRAME =[val1] [val2] [val3]
           | VALUES= < [val] >
          <
           | < TIME_AND_VALUES= [tval] [val] >
    >
```

Explanation:

ISEQ = *[ival]* M Sequence number of timeseries.

SERIES = *[text]* O SERIES can have two possible values: ’regular’ or ’irregular’.
When SERIES = ’regular’ keywords FRAME and VALUES are expected.
When SERIES = ’irregular’ keyword TIME_AND_VALUES is expected.

FRAME = *[val1] [val2] [val3]* O Time *[val1]* is the first time for which values are given.
Time *[val2]* is the interval time for which wind speed and angle are given.
Time *[val3]* is the last time for which values are given (all these times are given in elapsed minutes after midnight of the WAQUA start date).

VALUES = < *[val]* > O The values are given for the times as defined at the keyword frame.

TIME_AND_VALUES = *[tval] [val]* O In this case it is possible to give values for open boundaries at non-equidistant times.

FOURIER (optional)

```
FOURIER
    GENERAL
        OMEGA= < [val] >
    SERIES
        <     F: ISEQ= [ival]
                AZERO= [val]
                AMPL= < [val] >
                PHASE= < [val] >
        >
```

Explanation:

OMEGA = < *[val]* > M The angular frequencies are given here for N components (10^{-4} rad / s).

ISEQ = *[ival]* M Sequence number of Fourier series.

AZERO = *[val]* M Amplitude at point *[iseq]* for zero frequency (m).

AMPL = < *[val]* > M Sequence of amplitudes for N frequencies at point *[iseq]* (m).

PHASE = < *[val]* > M Sequence of phases at point *[iseq]* for N frequencies (rad).

The function used is:

$$f_i(t) = A_{i0} + \sum_j A_{ij} \cos((\omega_j \,.\, 10^4)t + \varphi_{ij})$$

where:

$f_i(t)$	=	function value, where i is the value of ISEQ
A_{i0}	=	amplitude at zero frequency (AZERO)
j	=	Fourier component
A_{ij}	=	amplitude of the jth component (AMPL)
ω_j	=	angular frequency of the jth component (OMEGA)
φ_{ij}	=	phase of the jth component (PHASE)

4.2.1.5 INPUT_SPATIAL_DATA (optional)

In this subsection the input spatial data can be given.

```
INPUT_SPATIAL_DATA
    < IS[iseq]
          GLOBAL
          LOCAL
    >
```

GLOBAL (mandatory)

Global data can be specified in two ways: first by giving one value for the complete computational grid, second by giving values for each grid point.
The way of ordering these values is determined by the layout flag.

```
GLOBAL
    LAYOUT = [ival]
    | CONST_VALUES = [val]
   <
    | VARIABLE_VALUES = < [val] >
```

Explanation:

CONST_VALUES = *[val]* O See paragraph 0

Default = 0

VARIABLE_VALUES = < *[val]* > **O** See paragraph 0

LAYOUT = *[ival]* **D** See paragraph 0
Default = 1

LOCAL (optional)

```
LOCAL
  < BOX:  MNMN = ( [ival], [ival] )  ( [ival], [ival] )
          | CONST_VALUES = [val]
         <
          | CORNER_VALUES = [val], [val], [val], [val]
         <
          | VARIABLE_VALUES = < [val] >
  >
```

Explanation*:*

BOX **R** See paragraph 0

MNMN = (*[ival], [ival]*) (*[ival], [ival]*)
M See paragraph 0

CONST_VALUES = *[val]* **O** See paragraph 0

CORNER_VALUES = *[val], [val], [val], [val]*
O See paragraph 0

VARIABLE_VALUES = < *[val]* > **O** See paragraph 0

4.2.1.6 TIME_DEPENDENT_DATA (optional)

In this subsection the time dependent spatial data can be given.

```
TIME_DEPENDENT_DATA
    < TDS
           GLOBAL
           LOCAL
    >
```

Explanation*:*

TDS Each definition of a time dependent data field must start with this keyword.

GLOBAL (mandatory)

Global data can be specified in two ways: firstly by giving one value for the complete computational grid, secondly by giving values for each grid point.
The way of ordering these values is determined by the layout flag.

```
GLOBAL
    ISEQ = [ival]
    TIME = [tval]
    LAYOUT = [ival]
    | CONST_VALUES = [val]
  <
    | VARIABLE_VALUES = < [val] >
```

Explanation:

ISEQ = *[ival]* **M** Sequence number.

TIME = *[tval]* **M** Time valid for this field.

CONST_VALUES = *[val]* **O** See paragraph 0
Default = 0

VARIABLE_VALUES = < ***[val]*** **>** **O** See paragraph 0

LAYOUT = ***[ival]*** **D** See paragraph 0
Default = 1

LOCAL (optional)

```
LOCAL
   <BOX:  MNMN = ( [ival], [ival] )  ( [ival], [ival] )
          | CONST_VALUES = [val]
        <
          | CORNER_VALUES = [val], [val], [val], [val]
        <
          | VARIABLE_VALUES = < [val] >
   >
```

Explanation:

BOX **R** See paragraph 0

MNMN = (***[ival], [ival]*** **) (** ***[ival], [ival]*** **)**
M See paragraph 0

CONST_VALUES = ***[val]*** **O** See paragraph 0

CORNER_VALUES = ***[val], [val], [val], [val]***
O See paragraph 0

VARIABLE_VALUES = < ***[val]*** **>** **O** See paragraph 0

4.2.1.7 OUTPUT_SPATIAL_DATA (optional)

Output spatial data have the same structure as input spatial data. The difference is that output spatial data are written to the SDS-file on so-called map times. That is why the possibility exists to give these data a name and a unit that can be used by a postprocessing program.

```
OUTPUT_SPATIAL_DATA
   < OS [iseq]
          NAMES
             NAME= [text]
             UNIT= [text]
          GLOBAL
              (See paragraph 0)
          LOCAL
              (See paragraph 0)
   >
```

Explanation:

OS ***[iseq]*** **S** Output spatial data sequence number.

NAMES **O** Each name and unit definition must start with this keyword.

NAME = ***[text]*** **M** Name of this field. The maximum size of the text for this field is 20 characters.

UNIT = ***[text]*** **M** Unit of this field. The maximum size of the text for this field is also 20 characters.

5 Examples

5.1 Mussel filter

5.1.1 General

The aim of the following example is not to solve the problem in a correct manner, but to demonstrate the utilities of the user routine WASUST and of the SIMONA processing tool SIFODE. Also is shown how to write results to a self-defined file within the SIMONA system.

5.1.2 Model description

In this example we try to model a so-called mussel filter. This filter consists of an amount of parallel nets, which are placed perpendicular to the bottom and perpendicular to the stream flow. Each net contains several mussels.

The mussels capture suspended matter which is contaminated with cadmium. A part of the cadmium is accumulated by the mussels in their tissues; the rest ends up in pseudofaeces. The pseudofaeces consist of smaller particles and sedimentate. Consequently the concentration of suspended matter in water and hence the concentration of cadmium decreases. Expressions for filtering and pseudofaeces production are taken from:
'*Filtration rate and pseudofaeces production in Zebra Mussels and their application in water quality management*'
(R. Noordhuis et. al., Limnologie Aktuell (4) 1992).

The schematization of the water system in this model is chosen as small as possible: M = 3 and N = 3, the filter is positioned in the middle (M = 2, N = 2). At one boundary a discharge of 18 m^3 / s is imposed with a concentration of suspended matter of 12 g / m^3, which is also the initial concentration. At the other boundary a water level of 0 m is imposed; the overall depth is 5 m. This resembles the situation of "Lake Volkerak".

Besides suspended matter and cadmium in a mussel also 'continuity' is modelled to check the mass conservation of the numerical scheme.

The filter starts when the water movement is steady; this occurs after about one day (1440 minutes).
The complete simulation takes two days.

The mathematical description of the filter is defined by two partial differential equations.

The total change of suspended matter is described by:

$$\frac{\partial}{\partial t}(\textit{Suspended matter}) = \begin{array}{l} -\textit{" advection"} \\ +\textit{" dispersion"} \\ -\textit{" filtration by mussels"} \end{array}$$

In general only the first two terms are solved by the standard transport solver. The solver also accounts for the last term, when the user routine WASUST is called, where sources or sinks get a value defined by the user.

The last term acts as a ’sink’. This term is proportional to the concentration of suspended matter, the filtration rate of a single mussel and the total amount of mussels. The filtration rate itself depends on the concentration of suspended matter, which actually makes this term unlinear; nevertheless in this example implicit modelling (SIMP) is applied.

The change of cadmium in a single mussel is described by:

$$\frac{d}{dt}(\textit{Cadmium in mussel}) = \begin{array}{l} +\textit{" intake"} \\ -\textit{" faeces production"} \end{array}$$

The constituent ’cadmium in a mussel’ is not a constituent in terms of WAQUA: it is not effected by advection or dispersion. For that reason the array SOLUSR and the routine SIFODE are available. SIFODE is solving an ordinary differential equation: for linear (sink) terms coefficient α5 has to be defined, all other terms use coefficient β6. In fact the routine SIFODE does the same as the transport solver of WAQUA/TRIWAQ without advection and dispersion. Coefficient α can be compared with SIMP, coefficient β7 with SEXP. In this example both ‘intake’ and ‘faeces production’ are modelled explicitly by using coefficient beta.

Routine SIFODE has to be called by the user, unlike WASUST, which is called automatically by the system.

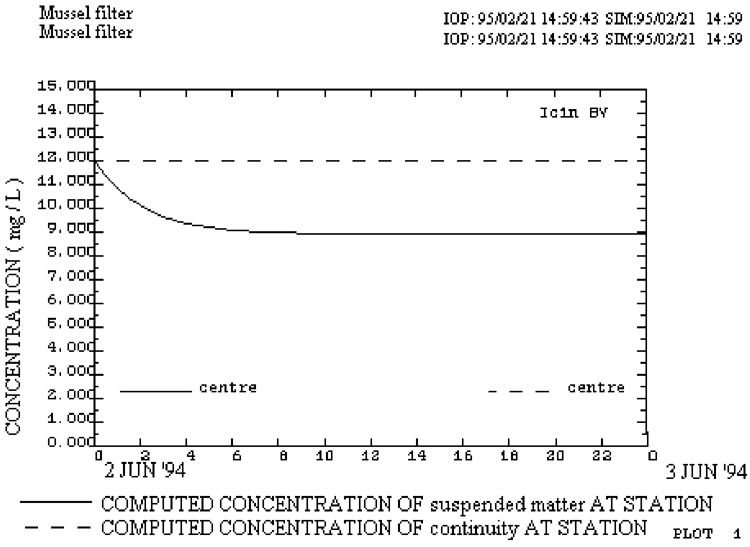

Fig. 0.1

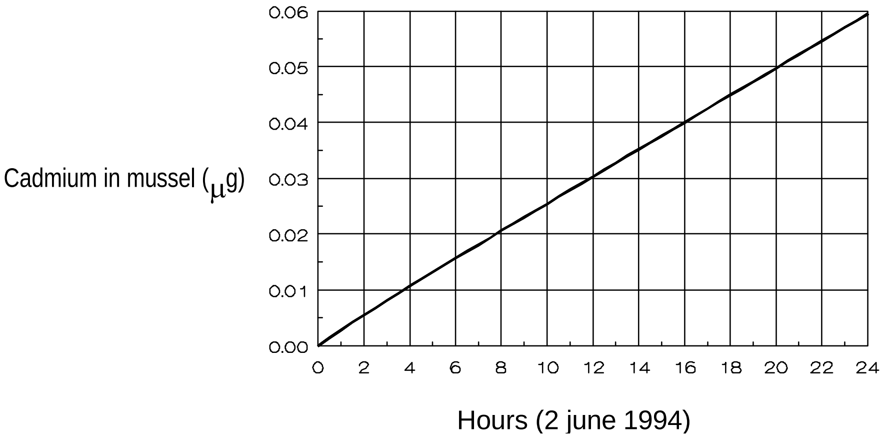

Fig. 0.2

A more detailed description can be found in the source code (pseudo code and comments). Results are graphically shown in Fig. 0.1 and Fig. 0.2.

5.1.3 Input file mussel filter

```
# 2 constituents: continuity and suspended matter
# run 001 number of nets = 100 ; density mussels per m2 = 5000
# discharge = 18 m3/s; filter calculations after 1440 minutes
# cadmium calculation
IDENTification
  WAQUA
  EXPERIMENT='001' OVERWRITE
  MODID='Cdmussel'
  TITLE='Mussel filter'

MESH
  GRID
    AREA
      ANGLEgrid = 0.0                          # angle Y-axis and North
      MMAX = 3     NMAX = 3                    # number of grids in X and Y
      LATItude = 52.5

    RECTilinear
      STEPsize = 500.0                          # distance between grid points

POINTS                                          # checkpoints
    P 1=(M=1,N=2,NAME='left')
    P 2=(M=2,N=2,NAME='centre')
    P 3=(M=3,N=2,NAME='right')

  BOUNdaries
    OPENings
      OPEN1: LINE (P1 , P1  NAME = 'Q opening')
      OPEN2: LINE (P3 , P3  NAME = 'W-level opening')

  BATHYMetry
    GLOBAL
      CONST_VALUES =5.0

GENERAL
  DIFFusion
    GLOBAL
      CONST_values = 25                         # local diffusion coef.

  PHYSicalparameters
    WATDENSity= 1000.0

FLOW
  PROBlem
    TIMEFrame
      DATE= '01-jun-1994'  TSTART= 0.   TSTOP= 2880.     # 2 days

    METHODvariables
      TSTEP= 1                                  # time step in minutes
      ITERACCURVel= 0.005                       # convergence criterium flow velocities

    FRICTion
      GLOBAL
      UDIRec
        GLOBAL CONST_values = 0.026    # friction value
      VDIRec
        GLOBAL CONST_values = 0.026    # friction value

  FORCings
    INITial

    BOUNDAries
      B: OPEN1 BTYPe='disch' BDEF='series' SAME
      B: OPEN2 BTYPe='wl'    BDEF='series' SAME

    TIMESERies
      S: P1 TID= 18.0       # discharge 18 m3/s
      S: P3 TID=  0.0       # water level

  CHECKPoints
    LEVELStations   P1, P2, P3
    CURRENtstations P1, P2

TRANSPORT
  PROBlem
    CONSTITuents
      CO1: POLUTant = 'continuity',       PUNIT = 'mg/l'
      CO2: POLUTant = 'suspended matter', PUNIT = 'mg/l'

  FORCings
    INITial
      CONSTITuents
```

```
        CO1  GLOBAL CONST_values = 12.0    # g/m3 = mg/l
        CO2  GLOBAL CONST_values = 12.0    # g/m3 = mg/l

    BOUNDaries
      RETURNtime
        CRET: P1, TCRETA=1     # Return time after current ...
        CRET: P3, TCRETA=1     # ... reverse to inward flow

      TIMESeries
         TS: CO1, P1, CINIT = 12.0
         TS: CO2, P1, CINIT = 12.0

  CHECKPOINTS
    CONSTITUENT_stations  P1, P2, P3

  USERData_transport
    CONTRol
      TYPE = 1 LENWrk = 0

    REALS
      USER1 = 1440.      # starting time effect mussel filter
      USER2 = 2880.      # = TSTOP (needed for closing file in WASUST)
      USER3 =   30.      # time interval writing results in WASUST

    INTEGers
      IUSER1: 5000       # density of mussels per square m
      IUSER2:  100       # amount of nets in filter
      IUSER3:    1       # identify number for output file

    OUTPUT_SPatial_data
      OS1
        NAMES        NAME = 'Cd_mussel'
                 UNIT = 'μg/mussel'
        GLOBAL       CONST_value = 0.0    # starting values

SDSOUTput
  MAPS
    TFMAPs=  1440.  TIMAPs= 1440.  TLMAPs=  2880.

  HISTories
    TFHISto= 1440.  TIHISto=  30.  TLHISto= 2880.

PRINToutput
  TRANSport
    CO1 CO2

  CONTRol
    TFRAMEHist=1440,30,2880
```

5.1.4 Subroutine WASUST

```
      subroutine wasust( irogeo, kf    , guuvv , xydep , rp    , vel   ,
     +                   wind  , wval  , zk    , difuv , difw  , vicow ,
     +                   rho   , user  , iuser , fvalue, spainp, spatim,
     +                   solusr, work  , simp  , sexp  , iparll, mypart,
     +                   numprc, mydom , ndom  , npart , myprc , mcordt,
     +                   ncordt, icocad, dum2  , dum3  )
c
c====================================================================
c     Programmers    Carlijn Bak, Dirk Vlag
c
c     Version 1.1    Date    04-12-2006     m291865: added ICOCAD to
c                                  parameter list (BvtH,
c                                  VORtech)
c     Version 1.0    Date    21-02-1995
c     Local filename  mosfil.f
```

```
c
c
c    Copyright (c) "Rijkswaterstaat".
c    Permission to copy or distribute this software or documentation
c    in hard copy or soft copy granted only by written license
c    obtained from "Rijkswaterstaat".
c    All rights reserved. No part of this publication may be
c    reproduced, stored in a retrieval system (e.g., in memory, disk,
c    or core) or be transmitted by any means, electronic, mechanical,
c    photocopy, recording, or otherwise, without written permission
c    from the publisher.
c
c ******************************************************************
c
c    DESCRIPTION
c
c Example user routine wasust. Empty body is filled with code to
c describe the simulation of a mussel filter: decreasing suspended
c matter by use of a mussel filter.
c cadmium balance, suspended matter, mussel and pseudo faeces.
c Musselfilter in the centre of the WAQUA grid(M = 2, N = 2).
c Effect filter starts one day after the beginning of the experiment.
c
c ******************************************************************
c
c    INPUT / OUTPUT    PARAMETERS
c
c Because of the length this part is shortened. A complete description
c can be found in User's guide WAQUA, section 5 (processor WAQPRO).
c
c    iuser(1)      i    mussel density on a net [number of mussels/m2]
c    iuser(2)      i    number of nets
c    iuser(3)      i    identify number output file
c    user(1)       i    starting time of effect filter (1440 min.)
c    user(2)       i    = TSTOP
c    user(3)       i    time interval printing results to own file
c    solusr(2,2,1)  o    cadmium in a mussel [μg/mussel]
c
c ******************************************************************
c
c   LOCAL PARAMETERS
c
    integer iinput(3),info(4),irefun,mvak,nvak
    real alfa(1,1),beta(1,1),c(1,1),cd_in,cd_psp,cf,cp
    real dt,fr,rinput(1),r1,r2,r3,r4,r5,totmus
    character fname*100,name*6
    parameter (mvak=2,nvak=2)
    parameter (r1=187.1,r2=0.037,r3=4.e-6,r4=5.54,r5=0.97)
    parameter (cf=1.e-6/3600.,cp=1.e-3/(24.*3600.))
    save        alfa,c,dt,iinput,irefun,totmus
    data name /'WASUST'/
c
c    alfa     'full matrix' form alfa coefficient [1/s] (SIFODE)
c    beta     'full matrix' form beta coefficient [g/s/mussel]
c             (SIFODE)
c    c        'full matrix' form with results from SIFODE [g/mussel]
c    cd_in    mussel intake of cadmium [g/s/mussel]
c    cd_psp   cadmium in pseudo faeces [g/s/mussel]
c    cf       conversion [ml/h] --> [m3/s]
c    cp       conversion [mg/day] --> [g/s]
c    dt       dtsec/2
c    fname    filename for own results
c    fr       filtration rate [m3/s/mussel] (in Noordhuis
c             [ml/h/mussel])
c    iinput   (1) length of IINPUT - 1 (SIFODE)
c             (2) flag indicating:
c                  0: time integration over all grid points
c                  1: IROGEO table is used
c    info     (1)  status file: 0 = old, 1 = new, 2 = old/new
c             (2)  access file: 0 = seq., 1 = direct
c             (3)  format file: 0 = formatted, 1 = unformatted
c             (4)  record length for direct
```

```
c    irefun    ref. to unitnr. of fname
c    mvak      location filter in WAQUA-grid
c    name      name of subroutine (WASUST)
c    nvak      location filter in WAQUA-grid
c    r1        constant {Noordhuis et. al.}
c    r2        constant {Noordhuis et. al.}
c    r3        constant (fraction of Cd in susp. matter)
c    r4        constant {Noordhuis et. al.}
c    r5        constant {Noordhuis et. al.}
c    rinput    not yet used (SIFODE)
c    totmus    total amount of mussels
c
c *******************************************************************
c
c    I / O
c
c    /export/home/bak/waqua/MOSSEL/mosout.???
C    with ??? = iuser(3)
c    The unit number of this file is supplied by the Simona system
c
c *******************************************************************
c
c    SUBROUTINES CALLED
c
c    SIFLCL      : closing file with Cd results
c    SIFLOP      : opening file with Cd results
c    SIFODE      : computing amount of Cd in one mussel after a time
c                  step
c    SITXRB      : get position last non-blank character in string
c
c *******************************************************************
c
c    ERROR MESSAGES
c
c    None.
c
c *******************************************************************
c
c    PSEUDO CODE
c
c    d(SM*Vol)/dt = - fr * totmus * SM,   Vol = h*dx*dy
c      SM := rp(.,.,.,2) suspended matter [mg/l]
c      simp(.,.,.,2) = fr * totmus / Area,   Area = dx*dy
c
c    d(cd_mus)/dt = fr * SM * r3 - PSP * r3
c      cd_mus := solusr(.,.,1) cadmium in a mussel [μg/mussel]
c      PSP := pseudo faeces production [μg/s/mussel]
c
c    use routine SIFODE with beta = fr * rp(.,.,.,2) * r3 - PSP * r3
c
c ===================================================================
c
     if (((timnow-tstart)*60.) - dtsec .lt. -0.1) then
c
c First call, initializing
c
c Calculating total amount of mussels
c
       totmus = iuser(1)*iuser(2) * dy * (-1. * zk(nvak,mvak,1))
c
c Initial values needed for SIFODE
c
       dt=dtsec/2.
       alfa(1,1)=0.
       c(1,1)=solusr(nvak,mvak,1)*1.0e-6
       iinput(1)=2
       iinput(2)=0
c
c Setting up and opening the output file for recording Cd in one mussel
c
       fname='mosout.'
c
```

```
c Determine length of fname with simona tool sitxrb
c
        call sitxrb(fname,ifnb,ilnb)
        write(fname(ilnb+1:ilnb+3),'(i3.3)') iuser(3)
        info(1)=2
        info(2)=0
        info(3)=0
        call siflop(fname,info,irefun,name)
      endif
c
c End of initializing
c
      if(timnow .gt. user(1)) then
c
c Computing effect mussel nets after user(1) minutes on suspended
c matter
c
        fr = r1 * exp(- r2 * rp(nvak,mvak,1,2)) * cf
        simp(nvak,mvak,1,2) = fr * totmus / (dx*dy)
c
c Computing accumulation of Cd in one mussel
c
        cd_in = fr * rp(nvak,mvak,1,2) * r3
        cd_psp = (r4 + r5 * rp(nvak,mvak,1,2)) * cp * r3
        beta(1,1) = cd_in - cd_psp
        call sifode(c,alfa,beta,1,1,dt,iinput,rinput,irogeo,norows)
c
c Writing to solusr (in ug/mussel)
c
        solusr(nvak,mvak,1)=c(1,1)*1.0e+6
c
c Every user(3) time interval writing Cd to file
c
        if(mod(timnow,user(3)) .lt. 0.01) then
          write(ireffl(1,irefun),'(2g12.5)')
     +            timnow,solusr(nvak,mvak,1)
        endif
      else
c
c timnow < user(1): no effect mussel nets
c
        simp(nvak,mvak,1,2) = 0.
        solusr(nvak,mvak,1) = 0.
      endif
c
c At TSTOP(=user(2)) closing the file with Cd results
c
      if(abs(user(2)-timnow) .lt. .001) then
        call siflcl(irefun,name)
      endif
      end
```

5.2 Conversion of salinity to chlorosity

Purpose:
User wants mapping to the SDS-file of the computed salinity in terms of chlorosity, in order to be able comparing these values to previously observed chlorosity data (P. Lievense, WTZ database).

Solution:
Make sure that 'SALINITY' is constituent nr 1.
Add an extra constituent 'CHLOROSITY' (seq. nr. *n*).
Define conversion factor from salinity to chlorosity: 1. / 1.773

This results in the following (part of the) WAQPRE-input:

```
TRANSPORT
   PROBLEM
      CONSTITUENTS
         CO 1:   POLUTANT = 'SALINITY',
                  PUNIT = 'KG / M**3'
           .
           .
         CO n:   POLUTANT = 'CHLOROSITY',
                   PUNIT = ' KG / M**3 / 1.773'
      SALINITY:  CO 1
```

behind the TRANSPORT-section:

```
USERDATA_TRANSPORT

   CONTROL
      TYPE = 01            #   (irrelevant, however mandatory)
      LENWRK = 1

   REALS
      USER 1 = 1.773

   OUTPUT_SPATIAL_DATA
      OS 1
         NAMES
            NAME =  'CHLOROSITY layer 1'
            UNIT   =  'KG /M**3 / 1.773'
            GLOBAL
               CONST_VALUE = 16.36
               LAYOUT = 1
            LOCAL
               BOX MNMN = (1,  1,     72, 16),
                    VARIABLE_VAL =
                      INCLUDE FILE =  'chlorositydata.box1'
               BOX MNMN = (1, 17,  72, 31),
                  VARIABLE_VAL =
                      INCLUDE FILE =  'chlorositydata.box2'
      OS 2
```

.
.

After execution of the WAQPRO-simulation the chlorosity-values will be mapped as 'CHLOROSITY layer 1' constituent (unfiltered/filtered squares, unfiltered/filtered contours) to the SDS-file and can be visualized by the post-processing program SIMVIEW.

SIMVIEW-visualization:

1. Create the desired visualization on the SIMVIEW-desktop
   (caution: to get a readable legenda, the relevant legenda-height
   should be slightly increased)
2. Select via menu 'File/Print Setup' as driver:
   'Driver for the UNIRAS software rasterizer'
3. Select as device: 'TIFF RGB Intensity Device 300 d.p.i'
4. After pressing 'OK': select 'File/Print';
   now the picture is saved as a 'TIFF" in the working directory
5. FTP (from the UNIX-workstation) this file to your PC.
6. Start MS-WORD; open a new file
7. Select 'Insert/Picture'; select the picture
8. TIFF-file will be converted and (after 'OK') inserted into selected position in the MS-WORD-document
9. Eventually edit the picture, using 'Format/Picture'

5.3 Tracking of tidal extremes, computing mean values

Note:
This example is adopted from the project 'MHWMLW01', where it definitely is a non-standard application with respect to the use of a WAQUA User Routine.
It is meant to illustrate some features of a User Routine, which are not so obvious, but nevertheless available provided some precautions have been foreseen at programming time (see the notes in the section "DESCRIPTION" of the source code of this WASUST).

Purpose:
Determination during a user-defined simulation period of all occurring values of low- and high tides per waterlevel point (m,n) in the computational grid of a WAQUA/TRIWAQ simulation, taking into account dry/wet status. At the end, print a summary of the mean values per grid point.
Also: compute the mean concentration for all constituents (ignoring dry instances of a grid point) and print a summary per computational (m,n) grid point during a user-defined simulation period.

Outlines:
Fixed name of the printfile for waterlevels: "mhwmlw.out"
Fixed name of the printfile for concentrations: "conc.out"

WASUST performs at the first call a number of checks on data from the WAQPRE input file. If serious errors are detected, the program stops ("sistop"), otherwise a brief overview of User Routine related data will be printed to the message file.

WASUST needs the following data in the WAQPRE input file under the main key TRANSPORT.

example:

```
USERDATA_TRANSPORT
#
# CONTROL-section is "mandatory"
# for this specimen of WASUST: TYPE and LENWRK are set to zero
#
   CONTROL
      TYPE      = 0
      LENWRK = 0
#
# REALS-section is mandatory for this WASUST;
# if omitted or incomplete (less than two values):
# print error message and stop program
#
   REALS
#
#  tstart and tstop of WASUST period of interest for waterlevels:
#  elapsed minutes after midnight of WAQUA start date
#
      USER 1 = 0.0
      USER 2 = 14400.0
#
# tstart and tstop of WASUST period of interest for constituent
# concentrations; when omitted, no means for concentrations will be
# computed, if only mean concentrations must be computed:
# set USER 1 = USER 2
#
      USER 3 = 7200.0
      USER 4 = 14400.0
#
# N.B.: all tstart and tstop values of WASUST should be realistic with
#          respect to the tstart and tstop of the WAQUA simulation
#
```

Computation of average lowest- and highest waterlevels

For every relevant grid point the mean lowest- and highest waterlevels (i.e. tidal extremes) are computed (if a WASUST-interval is specified). WASUST employs a window-of-interest of 240 minutes (eventually to be changed in the source code of *wasust.f*). This period is also the minimum interval length specified in the WAQPRE input file (USER 2 – USER 1, see above). However, such a short period is not useful, because to detect at least two minima and maxima, a period of 24 hours (preferably more) should be specified.

As soon as the WASUST-window is filled with WAQUA-simulated

waterlevels (from array "zk"), the trends (increasing/decreasing waterlevels) during the first and last part (60 minutes, also modifiable in the source code of *wasust.f*) of the window are inspected.
If the trend in the first part is "increasing" and in the last part "decreasing", then a maximum (high tide) is expected in the WASUST-window, in the reverse case a minimum (low tide) is expected. The highest resp. lowest value in the WASUST-window will be searched and taken into account for the current average. Otherwise (no opposite trends), no action.

General rule: a minimum should be preceded by a maximum, vice versa.

After inspection of a filled WASUST-window all values are shifted to the "left" over 60 minutes (first part period) and the values in the last part (60 minutes) will be overwritten by newly computed waterlevels. After completion, the detection procedure is repeated, and so on until the end of the WASUST period of interest is reached.
At that moment a summary of the mean extreme waterlevels will be printed for every relevant grid point (file "mhwmlw.out" in the working directory). If no minimum or maximum could be detected for a point during the WASUST period of interest, the "nodata"-value (–9999.0) will be assigned.

For drying points the following rules are applied:
- if a point was permanently dry during the WASUST period of interest then it is not mentioned in the summary
- if a point has been dry during the WASUST period of interest the the average minimum gets the "nodata"-value
- the transition of "dry" to "wet" is interpreted as passing a minimum, i.e. the next extreme must be a maximum

The mean value of high waterlevels should never have the "nodata"-value, unless a point starts with wet values while no maximum could be detected, and stays dry afterwards.

Computation of average constituent concentrations

For every relevant grid point the mean concentrations of all constituents will be computed (if "lmax" > 0 and a WASUST–interval is specified in the WAQPRE input file).
For the time being there is a maximum in the printfile of four constituents.

For drying points the following rules are applied:
- if a point was permanently dry during the WASUST period of interest then it is not mentioned in the summary
- concentrations at dry time instances are not taken into account

6 Appendices

6.1 Log-sheet

document version	**date**	**Changes with respect to previous version**
1.0	September 2000	TUTOR_USER: first version of "WAQUA User-Routines"
1.1	December 2006	m291865: update header w.r.t. icocad in wasust example
1.2	June 2007	c71236: removed obsolete keywords
1.3	June 2009	m375281: removed bounding box from formulas and figures