UNIVERSITEIT GENT

DEPARTMENT OF APPLIED MATHEMATICS,
BIOMETRICS AND PROCESS CONTROL

GREAT-ER

Technical Documentation
-
Chemical Fate Simulator

Geert Boeije

February 16, 1999

Coupure 653, B-9000 Gent, Belgium

tel.: +32(9)264.59.32 / fax: +32(9)223.49.41 / WWW: http://hobbes.rug.ac.be

Table of Contents

1. Introduction

To deal with statistically distributed inputs and outputs, a hybrid simulation approach is used, involving both stochastic and deterministic techniques. The model core is deterministic. By means of Monte Carlo simulation, a stochastic layer is added on top of this core. A large number of 'shots', which are discrete samples from the distributed data set, are generated. For each distributed input parameter, there exists a discrete counterpart in the 'shot', which was sampled at random from the input distribution. For each of these 'shots', the deterministic model is called, which contains a mechanistic description of the considered processes in the rivers and in the waste water drainage areas. Process rates are derived from knowledge about chemical properties and process specifics. Finally, the (discrete) results from each 'shot' are statistically analyzed, to obtain distributed results as simulation output.

For reasons of model and data set simplicity and computation performance, only steady-state model formulations are applied. Hence, a number of fundamental assumptions are made: (1) constant chemical emissions: diurnal patterns in product and water consumption are disregarded, as well as variations between different days of the week; (2) constant flows within each steady-state model calculation run; (3) constant environmental properties.

To allow a straightforward mass-balancing approach, all determinands (chemical levels, water flows) are expressed as fluxes. Chemical mass fluxes Φ *(mass/time)* are applied to describe chemical loads. Water flows are expressed as volumetric fluxes Q *(volume/time)*. In the models, chemical concentrations C *(mass/volume)* are not used, because they are not independent of water flows, and they do not describe chemical transport. Chemical mass fluxes, on the other hand, are independent of dilution or flow (unless this dependency is implicitly included in the models which are used to calculate chemical mass fluxes). When concentrations are explicitly required, they are derived from the chemical mass fluxes and the hydrological volumetric fluxes: $C = \Phi / Q$.

Simulations can be performed for different scenarios (i.e., evaluations of different chemicals and chemical consumption patterns). The simulation input consists of a scenario-independent and a scenario-dependent data set. These data are expressed as statistical frequency distributions, incorporating seasonality and parameter uncertainty. Environmental characteristics are constant, and hence non-scenario-dependent. These include the river network structure, flow and flow velocity distributions, discharge point locations, treatment plant information, emission data, properties of sewers and small surface waters, etc. Chemical-specific information is scenario-dependent: chemical properties (i.e., biological, chemical and physical properties, specific process rates,...), and chemical market data (i.e., per capita product consumption rates). Market data are geo-referenced in the same way as the waste water information (i.e., related to waste water discharge points).

The simulation input data are expressed as statistical frequency distributions. This allows to include both seasonality effects and parameter variability and/or uncertainty into the simulation input. For river flows and flow velocities, the lognormal distribution is used (after NRA, 1995). For hydrological information this distribution is described by the mean and the 5^{th} percentile. In the case of flows, there is a 95% probability that the 5^{th} percentile low flow (also referred to as *Q95*) is exceeded: $P(Q>Q95) = 0.95$.

The simulation results are frequency distributions of chemical concentrations, incorporating temporal variability. For risk assessment purposes, these can be expressed as lognormal distributions, defined by their mean and 95^{th} percentile values. Predicted concentrations are geo-referenced in the same way as the input data set: river concentrations are associated with a river network structure, and waste water drainage area concentrations are associated with discharge points. Within one location, a further differentiation is made between the maximal predicted concentrations (i.e., upon discharge), the minimal predicted

concentrations (i.e., after degradation processes), and an 'internal' average value. For the calculation of the latter, specific algorithms have to be provided in the deterministic models.

2. Overview of Simulation Principles

2.1. System Segmentation

2.1.1. General

Geographical systems are segmented at 3 hierarchical levels (Figure 1):

Geographical Segments
  └→ Processes
    └→ Sub-processes

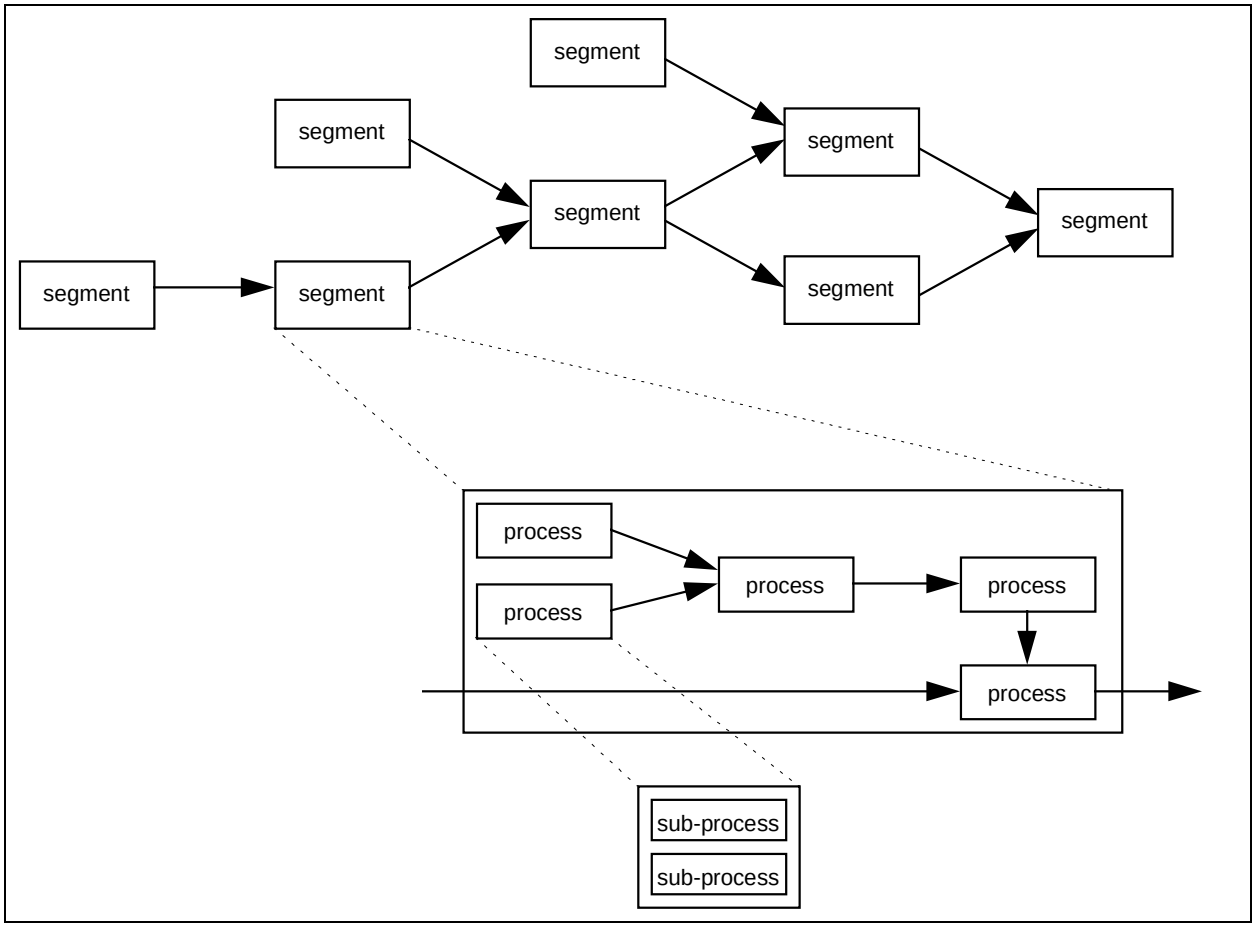

Figure 1. System segmentation

Geographical segments are connected in parallel or in series. One segment can have no, one or two inputs, and one or two outputs. The complete segmentation is considered as a binary tree, which implies that one final segment (the root) has to be present, representing the output of the entire system.
Within one geographical segment, several processes occur, also in parallel or in series. These can be transport / conversion or emission processes. One process may consist of different sub-processes, which occur in parallel. The segment input can be directed to any process or sub-process occurring in the segment

2.1.2. Applied

Geographical segments are actually river stretches, together with the associated 'waste water catchment' (i.e. the waste water drainage area, associated with a discharge into the considered river stretch).

The 'no-input' case represents the source of a river system. The 'two-inputs' case represents a confluence. The confluence of more than two rivers can be represented by means of several (hypothetical) 'two-inputs' segments. Bifurcations are be represented by the 'two-outputs' case.
The final segment (the root of the binary tree) represents the mouth of the river system. A hypothetical final segment can be used as root if several streams flowing into the sea or ocean are to be simulated as one larger system.

Two major segment types occur:

- 'discharge' river stretches with an associated waste water discharge point, and
- 'no discharge' river stretches which receive no such discharges.

In the 'discharge' case, a segment consists of several processes: the waste water pathway processes (i.e. emission, on-site treatment, sewers, treatment plant...) and the river process. In the 'no-discharge' case, only the river process occurs. In both cases, the input from upstream segments is directed to the river process. Some of the waste water pathway processes consist of several sub-processes. For example, in the waste water treatment plant process both 'no treatment' and 'treatment' sub-processes occur in parallel.

2.2. General Software Structure

The general software structure is presented in Figure 2. The simulator consists of the following sections:

- Control Section
- Data I/O Section
- Model Section

2.2.1. Control Section

The Control Section consists of two different hierarchical levels:

- Level 1 (Monte Carlo simulation level)

  - Obtain input parameters (call the Data I/O Section)
  - Generate a number of Monte Carlo simulation 'shots':
    - determine a random 'flow percentile' scenario
    - select discrete values from each parameter's distribution
    - call Level 2 for each 'shot'
  - Calculate distributions of the results (from the summary information provided by Level 2)
  - Store simulation output (call the Data I/O Section)

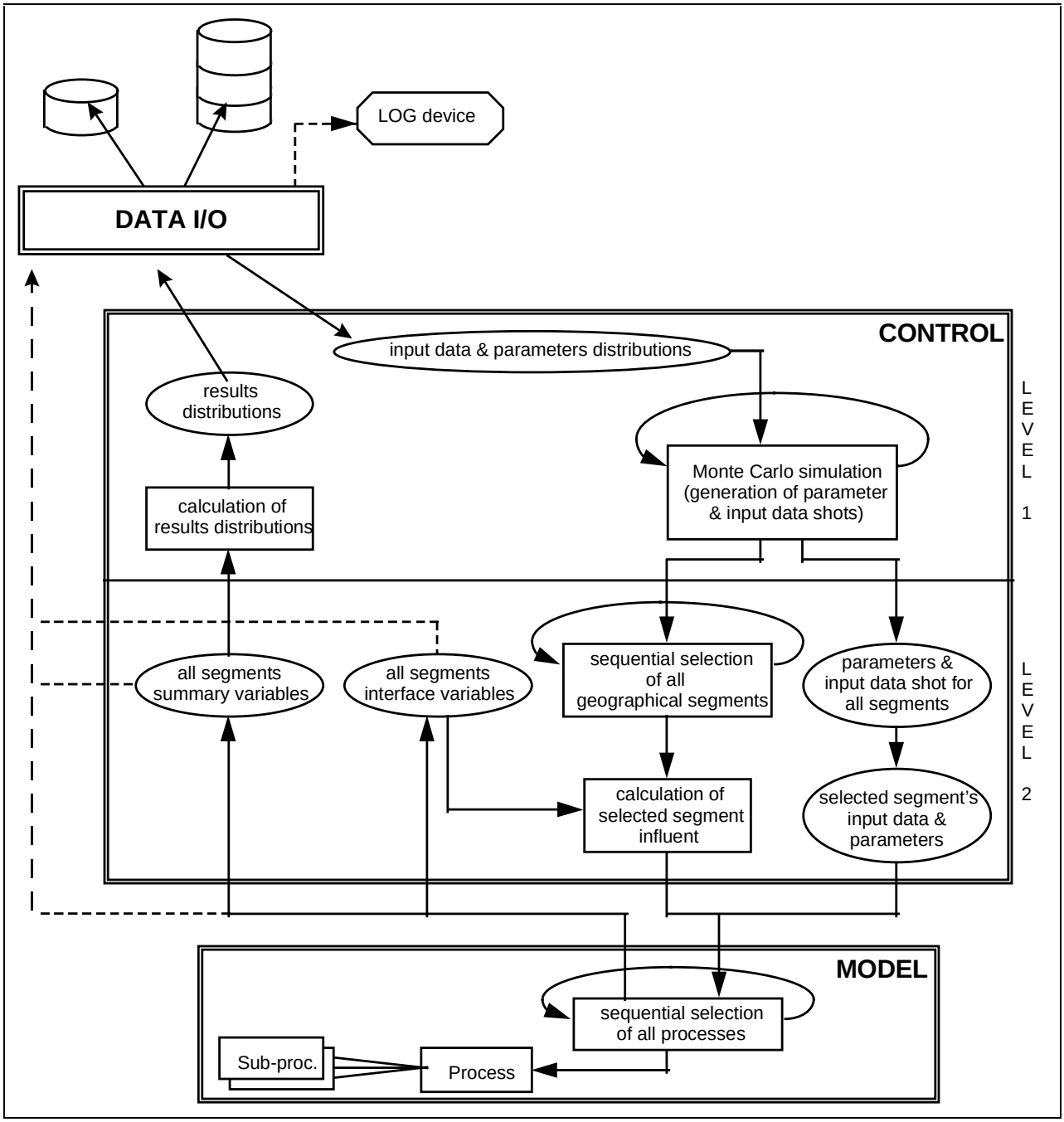

Figure 2. General software structure

- Level 2 (Segments Loop level)
  (applies to one discrete shot of the input data and parameters set, selected in Level 1)

  - Sequentially simulate all segments, from source to mouth
    - Prepare inflow data for each selected segment (from the interface variables)
    - Select the input data and parameters associated with the current segment
    - Simulate the current segment (Call Model Section)
  - Update the summary variables' distribution moments (needed to calculate the distributions in Level 1)

2.2.2. Data I/O Section

The Data I/O Section is called from the Control Section. It is mainly used to read and interpret the input provided by the geo-referenced and non-geo-referenced data banks, and to write the output back to a geo-referenced data bank. The following steps are performed:

- Input of control parameters
- Input of distribution types for each variable and parameter
- Input of non-geo-referenced parameters, river parameters, and discharge parameters
- Storage of model results
- Output of messages and commented results to log device, if required (log file and / or user interface)
- Processing of error messages

The input data originate from the GIS databank and the GIS user interface. They are sent to the models via ASCII files. After processing of the specific data subset, the model results are sent back to the GIS via an ASCII file.

2.2.3. Model Section

The Model Section is applied independently for each geographical segment. It is called from the Control Section, from which it also receives all required input data and model parameters. The input data represent a discrete event - no stochasticity is incorporated here. The Model Section consists of a Segment Control Section and one or more Processes. Each Process further consists of one or more Sub-processes.

- Segment Control Section (controls the simulation of an individual geographical segment)

  - Receive segment-specific input from the Control Section
  - Sequentially call the models applying to the different Processes
  - Calculate interface and summary variables for the current segment (from the model state variables)
  - Send the results (i.e. the interface and summary variables) back to the Control Section

- Processes (relate to individual processes occurring in a geographical segment)

  - Receive process-specific input from the Segment Control Section
  - Sequentially call the required Sub-Processes (one or several, depending on the process)
  - Perform model calculations (i.e. calculate emission or removal)
  - Send the results (i.e. the model state variables) back to the Segment Control Section

- Sub-Processes (relate to individual sub-processes within a specific Process)

  - Receive sub-process-specific input from the Process Model
  - Perform process rate calculations
  - Send the results (i.e. process rates) back to the Process Model

3. Simulation Approach

3.1. Control Section

The Control Section diagram is shown in Figure 3. It consists of two hierarchical levels. At the first level (left side of the figure), the simulation system is initialized, and the Monte Carlo simulation loop is started. After this, the results distributions are calculated and stored. For each 'shot' generated in the Monte Carlo simulation loop, the different segments are selected and simulated at the second level (right side of the figure).

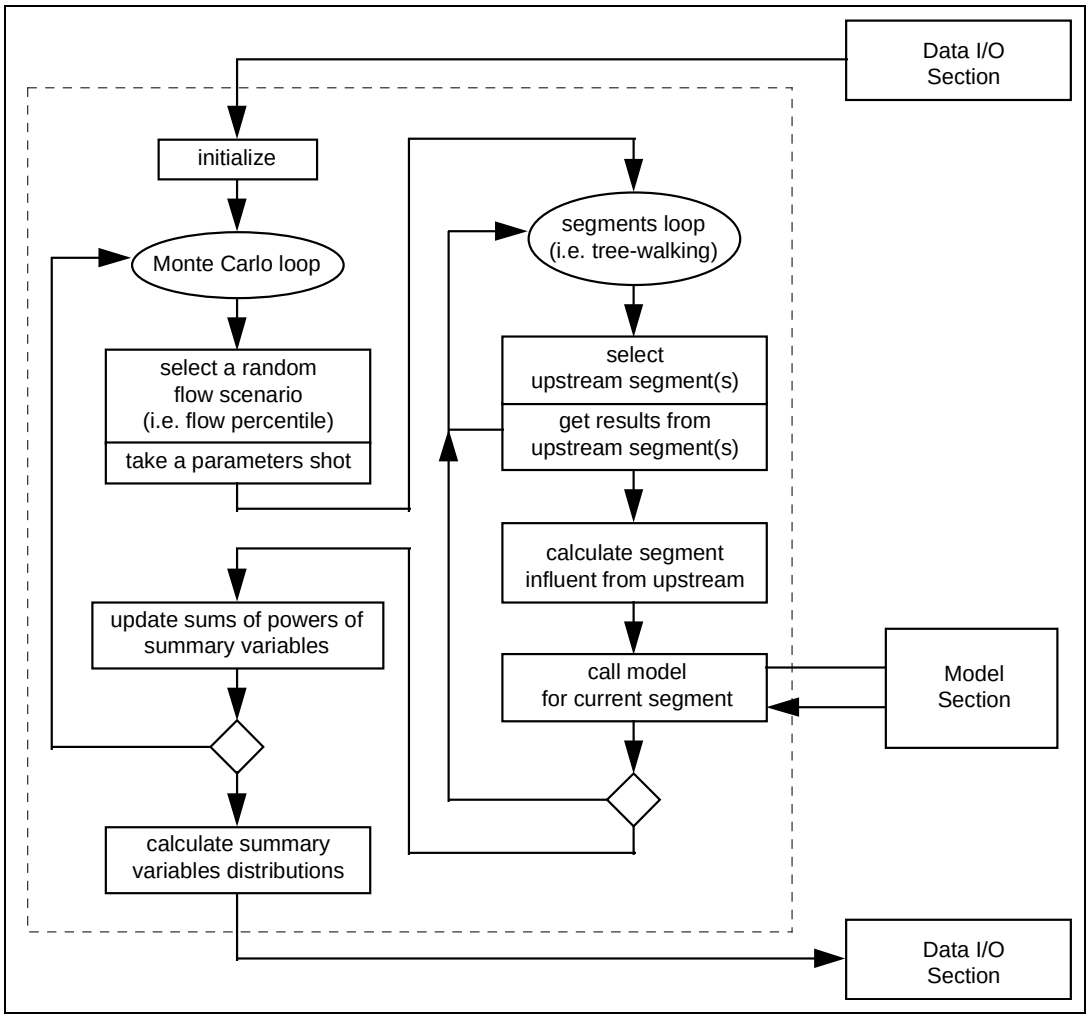

Figure 3. Control section

3.1.1. Pre-Simulation Actions

The Data I/O Section is called to obtain the required control parameters. The parameter arrays are allocated and initialized, as well as the different variables arrays and the calculation matrices. The different data files are read and the appropriate values are copied into the distribution parameters arrays.

Next, a number of 'navigation tools' (i.e., routines which help the simulator find its way in the different arrays) are initialized. Finally, some modifications are made to the input data. For the 'flow-type' distributions, the 'Q95 low flow' value is replaced by the (calculated) standard deviation (see below).

3.1.2. Level 1 - Monte Carlo Simulation

For a user-defined number of times, a random 'shot' is taken from the different parameter distributions.

First, a random flow percentile is taken from a uniform distribution between 0% and 100%. This single flow percentile is used throughout the 'shot'. Next, the discrete parameter arrays are generated from the distribution arrays (see also higher, 5.1.1 and Figure 9). Finally, the 'shot' values are verified.

After the generation of the discrete parameter arrays, the next level in the Control Section is called. Once this level has been completed, the sums of powers of the Summary Variables are updated. I.e., for each segment, the values of the current 'shot' Summary Variables are added to the previous 'sum of Summary Variables', the squares of the current values are added to the previous 'sum of squares of Summary Variables', etc.

Flow Correlated Parameters

The correct values for parameters which are inherently correlated to the flow scenario are derived from their distribution parameters and the flow percentile. Some parameters are 100% correlated to flow, such as River Flow Velocity, or River Depth. Others are not 100% correlated to flow, such as BOD or SS levels.

A correlation-corrected sampling method is applied to derive the 'shot' values of these flow-related parameters from their distribution and from the selected flow percentile.

Note that as the same flow percentile is applied throughout a 'shot', all segments use the same percentile - hence correlation between upstream and downstream flow percentiles is always assumed to be 100%.

Non-Correlated Parameters

All non-flow-related parameters are assumed to be uncorrelated. For these parameters, random 'shots' are taken from their distributions. The correlation / covariance matrix is assumed to be a unity matrix, and the Monte Carlo 'shot' generation requires no statistical corrections.

'Shot' Verification

As a next step, all the generated discrete values are checked and if necessary corrected. Irrelevant values may result from the random sampling from distributions. Indeed, some values must not be negative, or must be strictly positive; other values must be between 0 and 1. For example, generated chemical removal rates of higher than 100% or lower than 0% must be changed.

In the correction step, all irrelevant values are traced, and replaced by the nearest relevant values.

Generation of Distribution 'Shots'

Uniform Distribution

As a basis for the selection of random 'shots' from any statistical distribution, a random number generator was applied. This generator provides uniform random deviates between 0 and 1. A long period ($> 2 \times 10^{18}$) multiplicative congruential generator of L'Ecuyer, with Bays-Durham shuffle and added safeguards was implemented (Press et al., 1992; Knuth, 1982).

Normal Distribution

The Box-Muller method was applied to obtain normal (Gaussian) random deviates. It can be shown that y_1 and y_2 in equation (1) are standard normal $N(0,1)$ deviates (Press et al., 1992).

$$\begin{cases} y_1 = v_1 \cdot \sqrt{-2\dfrac{ln\left(v_1^{\,2} + v_2^{\,2}\right)}{v_1^{\,2} + v_2^{\,2}}} \\ y_2 = v_2 \cdot \sqrt{-2\dfrac{ln\left(v_1^{\,2} + v_2^{\,2}\right)}{v_1^{\,2} + v_2^{\,2}}} \end{cases} \tag{1}$$

with (v_1, v_2) a random point inside the unit circle around the origin $(0,0)$

Lognormal Distribution

Random 'shots' of a lognormal distribution are obtained by converting the lognormal distribution to the associated normal distribution, taking a normal 'shot' from the latter, and then converting the obtained value back to the lognormal distribution:

$$LN(\mu,\sigma) = exp\left[N(m,s)\right] \tag{2}$$

The conversion between the parameters of the normal to the lognormal distribution is given in (3).

$$\begin{cases} \mu = exp\left[m + \dfrac{s^2}{2}\right] \\ \sigma = \mu \cdot \sqrt{exp\left(s^2\right) - 1} \end{cases} \tag{3}$$

The standard deviation σ of a lognormal distribution $LN(\mu,\sigma)$ can be derived from its mean μ and a percentile value P_n. This is shown in (4) for the 5th percentile $P_5^{LN(\mu,\sigma)}$. (with $P_5^{N(0,1)}$ the 5th percentile value of the standard normal distribution, which is equal to -1.6449).

$$\sigma = \mu \cdot \sqrt{\exp\left[\left(\sqrt{\left(P_5^{N(0,1)}\right)^2 - 2 \cdot \ln\left(\frac{P_5^{LN(\mu,\sigma)}}{\mu}\right)} + P_5^{N(0,1)}\right)^2\right] - 1} \tag{4}$$

3.1.3. Level 2 - Segments Loop: Tree-Walking Algorithm

A recursive tree-walking algorithm is used to find the correct sequence in the considered river system. The simulation is started at the 'initial segment' (root of the tree) which is defined in the simulation control file. Several options are possible, depending on the segment type:

- Nothing Special (1 input, 1 output)
  - recursively call the tree-walking algorithm for the upstream segment
  - prepare the current segment's influent (i.e., the upstream effluent)
  - call the model for the current segment
  - return to the segment from which this segment was called

- Confluence (2 inputs, 1 output)
  - recursively call the tree-walking algorithm for both upstream segments
  - prepare the current segment's influent (i.e., the sum of both upstream effluents)
  - call the model for the current segment
  - return to the segment from which this segment was called

- End of Tree (0 inputs, 1 output)
  - call the model for the current segment
  - return to the segment from which this segment was called

- Bifurcation (1 input, 2 outputs)
  - recursively call the tree-walking algorithm for the upstream segment
  - prepare the current segment's influent (i.e., a fraction from the upstream effluent)
  - call the model for the current segment
  - return to the segment from which this segment was called

- Hypothetical Root Segment ('the sea') (2 inputs, 1 output)
  - recursively call the tree-walking algorithm for both upstream segments
  - no calculations are performed
  - return to the segment from which this segment was called

A special case is the bifurcation. In the described simulation system, the stretch upstream from the bifurcation has the 'nothing special' attribute, and both downstream stretches have the 'bifurcation' attribute. The bifurcation stretches have one upstream code referring to the upstream stretch, while the other upstream code is used to refer to the 'neighboring' stretch. The fraction of the chemical flux from upstream, sent to either bifurcation stretch, is assumed to be proportional to the flow fractionation (which is given in the data set).

To avoid duplicate calculations for identical upstream systems (in the case of bifurcations), it is checked whether a stretch has not already been simulated before the simulation is performed.

In order to simulate a system completely, a root segment has to be provided - even if there is no physical root segment in the real geography. In the latter case, a hypothetical root segment can be used (one can say that this represents 'the sea').

An example of different segment types is given in Figure 4 below.

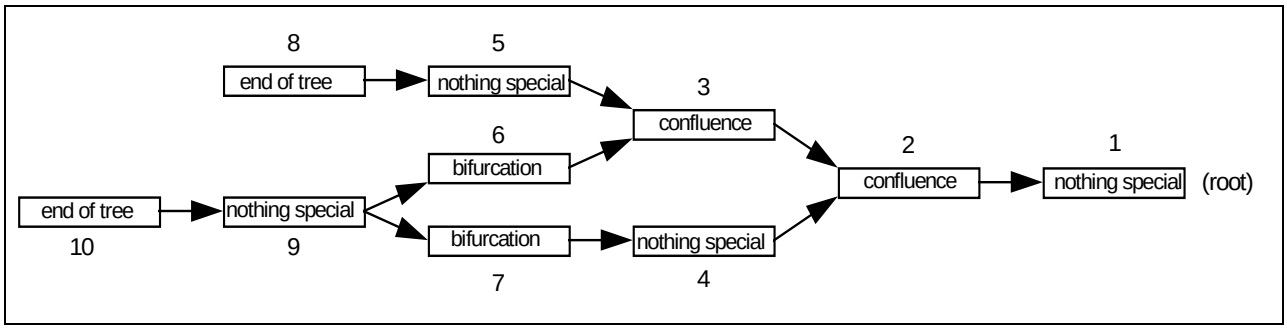

Figure 4. Segment types

In this example, the simulation sequence is as follows:

start: 1 is called
- 1 calls 2
  - 2 calls 3
    - 3 calls 5
      - 5 calls 8
        - *8 is simulated*
      - *5 is simulated*
    - 3 calls 6
      - 6 calls 9
        - 9 calls 10
          - *10 is simulated*
        - *9 is simulated*
      - *6 is simulated*
    - *3 is simulated*
  - 2 calls 4
    - 4 calls 7
      - 7 calls 9
        - 9 was already simulated - return to 7
      - *7 is simulated*
    - *4 is simulated*
  - *2 is simulated*
- *1 is simulated*

stop: all segments have been simulated

In special situations which can not be represented using the described segment types, zero-length segments have to be used to create the desired structure. For example, a confluence of three rivers immediately followed by a bifurcation can be represented by a zero-length confluence segment, followed by another zero-length confluence, followed by a bifurcation.

3.1.4. Post-Simulation Actions

Calculation of Results Distributions

After completing the Monte Carlo simulation loop, sums of powers of Summary Variables are available for each segment. From this, the distribution parameters of the Summary Variables are calculated (limited to mean and standard deviation):

Mean

$$\mu_k = \frac{\sum_{i=1}^{N} (SV_k)_i}{N} \qquad \text{with} \begin{vmatrix} \mu_k = \text{mean of Summary Variable k} \\ (SV_k)_i = \text{Summary Variable k in shot i} \\ N = \text{Nr. of Monte - Carlo 'shots'} \end{vmatrix} \qquad (5)$$

Standard Deviation

$$\sigma_k = \sqrt{\frac{\sum_{i=1}^{N}\left[\left(SV_k\right)_i^{\,2}\right] - N \cdot \mu_k^{\,2}}{N-1}} \qquad \text{with} \begin{vmatrix} \mu_k = \text{mean of Summary Variable k} \\ \sigma_k = \text{standard deviation of Summary Variable k} \\ \left(SV_k\right)_i = \text{Summary Variable k in shot i} \\ N = \text{Nr. of Monte - Carlo 'shots'} \end{vmatrix} \qquad (6)$$

For 'PEC-type' distributions (i.e., concentrations), the 95th high percentile is given instead of the standard deviation. This is obtained by applying equation (4) with $P_{95}^{N(0,1)} = 1.6449$. In this step, it is currently assumed that the 'PEC-type' distributions are lognormal.

Last steps

The Data I/O Section is called to store (and if required display) the results. Finally, the different arrays are de-allocated and control is returned to the system which called the simulator (i.e., the operating system or the GIS).

3.2. Model Section

3.2.1. Model building block interconnection

Within one segment, the model is built from a number of processes. Each of these processes consists of one or more sub-processes. An example of the interconnection of model building blocks is shown in the lower part of Figure 1.

Process models are used to describe distinct processes which occur in a segment. These processes can be independent from each other (e.g. different emission types), or they can depend on upstream processes (e.g. treatment, which depends on emission). Independent processes are connected in parallel, while dependent processes are connected in series. The sequence of the different processes is upstream to downstream (backflows can not be modeled using the described system).

Sub-processes are used to describe several options which can be followed within one process. For each flow fraction, only one option can be selected (e.g. in the case of waste water treatment, one specific waste water flow fraction is either treated or untreated). It follows that within a process, different sub-processes are always independent from each other. Hence these sub-processes are connected in parallel.

Connection of different model building blocks is achieved by means of outflow destination links. The State Variable values at each sub-process's outflow terminal, are split into different fractions. These are sent to the inflow terminals of the specified sub-processes, in one or more of the downstream processes. This interconnection technique is shown in Figure 5.

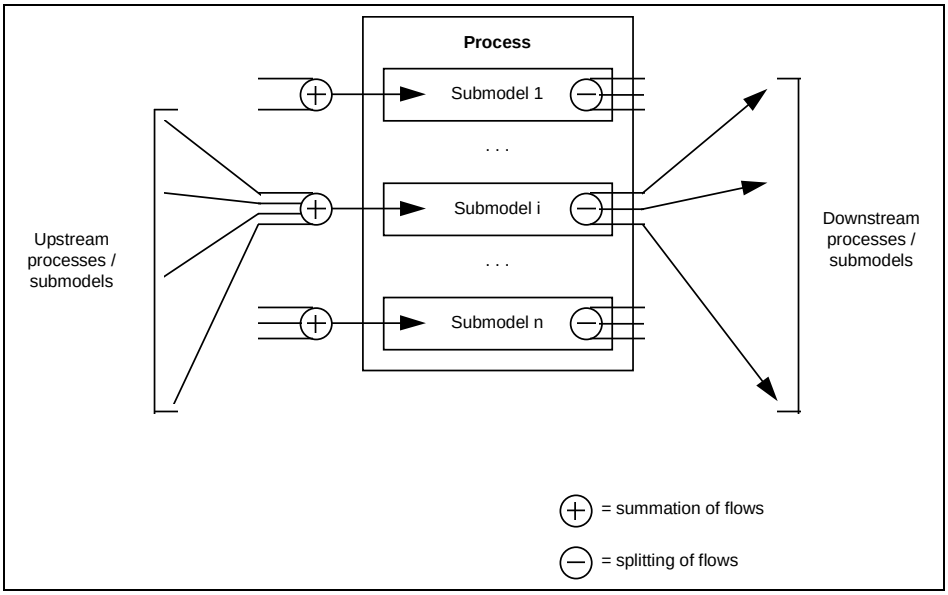

Figure 5. Processes and sub-processes interconnection

The list of possible destinations (processes / sub-processes) of each process is hard-coded. This list of destinations is identical for each sub-process within one process. The splitter fractions to each possible destination are sub-process-specific, and are part of the geo-referenced parameter set.

This way, each sub-process has all the required information to send its effluent to the next step(s). On the other hand, it requires no information on the origin of its influent. The latter is completely determined by the upstream sub-processes' outflow destination links.

3.2.2. Mathematical approach

The system of model equations for each determinand (i.e. flow, chemical flux, etc.) can be expressed as:

$$X_i = A_i \cdot X_i + B_i \qquad (7)$$

with
- X_i = state variables vector for determinand i, with elements x_k^i
- A_i = (square) transport / conversion matrix for determinand i, with elements $a_{k,l}^i$
- B_i = emission vector for determinand i, with elements b_k^i

However, this system can not be solved using the matrix calculation:

$$X_i = (I - A_i)^{-1} \cdot B_i \qquad \text{with } I \text{ the unity matrix} \qquad (8)$$

because the elements of the A_i matrices are obtained during the calculations. Values in A_i at row k may be derived from (previously calculated) values of any determinand in any upstream process:

$$a_{k,l}^i = f\left(x_{1\cdots k-1}^1, \cdots, x_{1\cdots k-1}^m\right) \qquad \text{with } m = \text{the number of determinands} \qquad (9)$$

Vector and Matrix Structure

The vectors X_i are partitioned per process, within a process per terminal, and finally within a terminal per sub-process. The process sequence in the arrays is upstream to downstream and the terminal sequence is input to output. The emission vectors B_i use an identical structure, hence the emission value b_k^i from the vector B_i

corresponds to the same process, terminal and sub-process as the state variable value x^i_k from the vector X_i, as shown below:

$$X_i = \begin{bmatrix} \vdots \\ \hline \textit{process j} \ \textit{(emission)} & \textit{emiss.} & x^i_{\textit{proc.j - emission - submodel 1}} \\ & & \vdots \\ & & x^i_{\textit{proc.j - emission - submodel } n_j} \\ \hline \vdots \\ \hline & \textit{input} & x^i_{\textit{proc.k - input - submodel 1}} \\ & & \vdots \\ \textit{process k} & & x^i_{\textit{proc.j - input - submodel } n_k} \\ \textit{(trans/conv)} & & x^i_{\textit{proc.k - output - submodel 1}} \\ & \textit{output} & \vdots \\ & & x^i_{\textit{proc.j - output - submodel } n_k} \\ \hline \vdots \end{bmatrix} \qquad B_i = \begin{bmatrix} \vdots \\ \hline b^i_{\textit{proc.j - emission - submodel 1}} \\ \vdots \\ b^i_{\textit{proc.j - emission - submodel } n_j} \\ \hline \vdots \\ \hline b^i_{\textit{proc.k - input - submodel 1}} \\ \vdots \\ b^i_{\textit{proc.j - input - submodel } n_k} \\ b^i_{\textit{proc.k - output - submodel 1}} \\ \vdots \\ b^i_{\textit{proc.j - output - submodel } n_k} \\ \hline \vdots \end{bmatrix} \tag{10}$$

The transport / conversion matrices A_i are square matrices, as shown below in (11). The element $a^i_{k,l}$ from matrix A_i is the conversion coefficient from state variable x_i in situation l to situation k:

- In conversions from the input terminal to the output terminal of a sub-process, this coefficient represents the non-removed fraction of x_i.
- In transportation from a process/sub-process with index l to a next process/sub-process with index k, this coefficient is the fraction of the output terminal value from l which is sent to the input terminal of k.

$$A_i = \begin{bmatrix} a^i_{1,1} & \cdots & a^i_{1,n} \\ \vdots & \ddots & \vdots \\ a^i_{1,n} & \cdots & a^i_{n,n} \end{bmatrix} \quad \text{with } a^i_{k,l} = \text{conversion coeffcient from } x^i_l \text{ to } x^i_k \tag{11}$$

Architecture of the Transport / Conversion Matrix A

Lower triangular matrix

In the different vectors and matrices, values from upstream processes occur before values from dependent downstream processes. Likewise, input values occur before the dependent output values. Because of this structure, state variable values at a specific situation are independent of state variable values at any subsequent situation. Also, in the applied models a state variable value at a specific situation is not dependent on itself.

Hence, in the A-matrix (12) is valid. Because of this A_i is a lower triangular matrix, with an all-zero diagonal.

$$\forall\ k \le l:\ x^i_k \text{ is independent of } x^i_l \qquad \Rightarrow \forall\ k \le l:\ a^i_{k,l} = 0 \tag{12}$$

Emission Processes Section

Emission values are derived completely from the B_i matrices. In the matrix calculations, all state variable values in these processes are independent of upstream values. Hence, condition (13) is valid, which entails that the submatrices $A^i_{emission}$ are all-zero.

$$\forall\, l : x^i_k \in \left[emission\ process\right] \quad \Rightarrow a^i_{k,l} = 0 \tag{13}$$

The architecture of the A_i matrices, based on (12) and (13) is shown below in (14):

$$A_i = \left[\begin{array}{cccccccc} 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 & 0 \\ \hline a^i_{n_1+1,1} & a^i_{n_1+1,2} & \cdots & a^i_{n_1+1,n_1} & 0 & \cdots & 0 & 0 \\ a^i_{n_1+2,1} & a^i_{n_1+2,2} & \cdots & a^i_{n_1+2,n_1} & a^i_{n_1+2,n_1+1} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ a^i_{n_1+n_2,1} & a^i_{n_1+n_2,2} & \cdots & a^i_{n_1+n_2,n_1} & a^i_{n_1+n_2,n_1+1} & \cdots & a^i_{n_1+n_2,n_1+n_2-1} & 0 \end{array}\right] \begin{array}{l} \left.\begin{array}{l}\\\\\\\end{array}\right\} n_1\ rows \\ \left.\begin{array}{l}\\\\\\\end{array}\right\} n_2\ rows \end{array} \tag{14}$$

with n_1 rows used for emission processes, and n_2 rows used for transport / conversion processes.

Transport / Conversion Processes Section - Input

Input terminal values of a state variable can depend on the values of any upstream process, associated with the emission, input or output terminals of these upstream processes. Input terminal values can not depend on input terminal values from other sub-processes within the same process - as these sub-processes are strictly connected in parallel. Because of this, the $A^i_{trans\,/\,conv\,-\,input}$ submatrices (15) are partitioned into:

- a first set of columns:
  - relating to upstream influences
  - contains the upstream outflow fractions, directed towards the considered process and sub-process
- a second set of columns:
  - relating to the current process and to further downstream processes
  - is all-zero

$$A^i_{trans/conv\,-\,input} = \left[\begin{array}{l|l} \left.\begin{array}{ccc} a^i_{k,1} & \cdots & a^i_{k,k-1} \\ a^i_{k+1,1} & \cdots & a^i_{k+1,k-1} \\ \vdots & \ddots & \vdots \\ \underbrace{a^i_{k+n-1,1}}_{k\text{-}1\ columns} & \cdots & \underbrace{a^i_{k+n-1,k-1}}_{k\text{-}1\ columns} \end{array}\right\} n\ rows & \left.\begin{array}{ccc} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ \underbrace{0}_{m-(k-1)\ colums} & \cdots & \underbrace{0}_{m-(k-1)\ colums} \end{array}\right. \end{array}\right] \tag{15}$$

with
n = number of submodels in the considered process
k = X-vector index of the first submodel's input value in the considered process
m = total number of rows / columns in the A matrix

Transport / Conversion Processes Section - Output

It is assumed that in transport / conversion processes, sub-process output terminal values are only dependent on the corresponding sub-process input terminal values. This implies that the $A^i_{trans\,/\,conv\,-\,output}$ submatrices (16) are partitioned into:

- a first set of columns:
  - relating to upstream influences
  - is all-zero
- a second set of columns:
  - relating to the input terminals of the considered process
  - is a diagonal matrix, with the conversion coeffcients from the input to the output terminals of each sub-process on the diagonal
- a third set of columns:
  - relating to the output terminals of the current process and to further downstream situations
  - is all-zero

$$A^i_{trans/conv\,-\,output} = \left[\begin{array}{c|c|c} \underbrace{\begin{matrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{matrix}}_{k\text{-}1\ columns} & \underbrace{\begin{matrix} a^i_{k+n,k} & 0 & \cdots & 0 \\ 0 & a^i_{k+n+1,k+1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a^i_{k+2n-1,k+n-1} \end{matrix}}_{n\ colums} & \underbrace{\begin{matrix} 0 & \cdots & 0 \\ 0 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{matrix}}_{m-(k-1+n)\ columns} \end{array}\right]_{n\ rows} \qquad (16)$$

with $\left|\begin{array}{l} n = \text{number of submodels in the considered process} \\ k = \text{X-vector index of the first submodel's input value in the considered process} \\ m = \text{total number of rows / columns in the A matrix} \end{array}\right.$

Internal Representation of the A matrix

As shown above in (14), (15) and (16), a large number of positions in the A matrix are inherently zero. These values are not stored in the memory representation of the matrix. Instead, a vector A_{memory} is used, containing only the values associated with positions in A which are not inherently zero (17). Hence, only values associated with transport / conversion models are stored into A_{memory}. A function is provided which maps 2-dimensional A matrix co-ordinates to the vector co-ordinates used to address A_{memory}.

$$\left[\cdots \underbrace{\overbrace{a^i_{k,1} \cdots a^i_{k,k-1}}^{input\ -\ (k\text{-}1)\cdot n\ values} \ddots \overbrace{a^i_{k+n-1,1} \cdots a^i_{k+n-1,k-1}}^{}}_{transport\,/\,conversion\ process\ \ (n\ submodels)} \overbrace{a^i_{k+n,k} \; a^i_{k+n+1,k+1} \ddots a^i_{k+2\cdot n-1,k+n-1}}^{output\ -\ n\ values} \cdots\right]^T \qquad (17)$$

Solution of Model Equations

Matrix equation (7) is a set of *n* equations with n unknown variables x^i. As A_i is a lower triangular matrix with all-zero diagonal, the solution for x^i_k only depends on the solutions for x^i_0 through $x^i_{k\text{-}1}$ and on the emission value b^i_k. Hence sequential row-by-row calculation of (18) gives the complete solution for X_i.

$$\begin{cases} x_1^i = b_1^i \\ x_2^i = a_{2,1}^i \cdot x_1^i + b_2^i \\ \quad \vdots \\ x_n^i = \sum_{k=1}^{n-1} a_{n,k}^i \cdot x_k^i + b_n^i \end{cases} \qquad (18)$$

Internally, (18) is solved in a slightly different way. No calculations are performed with $a_{i,j}$ values which are inherently zero. Hence three different calculation approaches are followed:

- Emission Process:

$$\forall\, x_k^i \in \left[emission\ process \right] : x_k^i = b_k^i \qquad (19)$$

- Transport / Conversion Process Input calculations:

$$x_k^i = \sum_{j=1}^{l} a_{k,j}^i \cdot x_j^i + b_k^i \qquad \text{with } l+1 = \text{first row of the current process} \qquad (20)$$

- Transport / Conversion Process Output calculations:

$$x_k^i = a_{k,k-l}^i \cdot x_{k-l}^i + b_k^i \qquad \text{with } l = \text{nr. of submodels in the current process} \qquad (21)$$

3.2.3. Model Section Actions

Inflow Preparation

Inflow into a segment is modeled as an emission into the segment from outside. These so-called emission values are stored in the B vector. Hence, in the Segment Control Section, initially the B vector values associated with the segment's inflow are prepared. The b^i values associated with the input terminals of the process receiving the inflow are set to the values of the input interface variables. In the applied case, this receiving process is the River Process. This concept is illustrated in Figure 6 below.

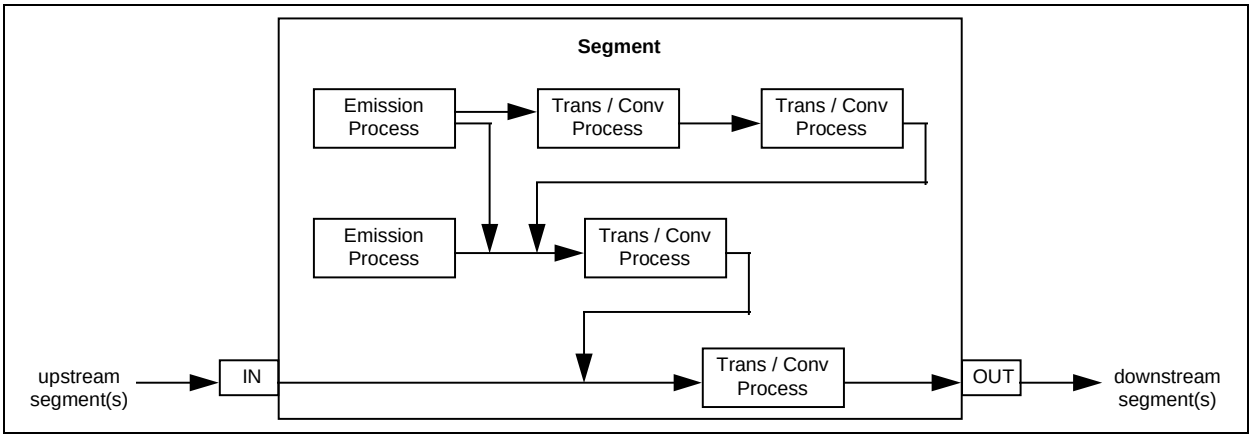

Figure 6. Input into and Output from a segment

Model Section Architecture

The general architecture of the Model Section is shown in *Figure 7*. In the processes loop, the different processes are selected, from upstream to downstream. For each process, the appropriate process model is called. There, the associated sub-processes loop is started, where the process rates are calculated, and the emission values and / or removal rates are determined. After the sub-processes loop, the A matrix and B vector values are calculated, and the updated A matrix and B vector are sent back to the processes loop for further use in the model calculations.

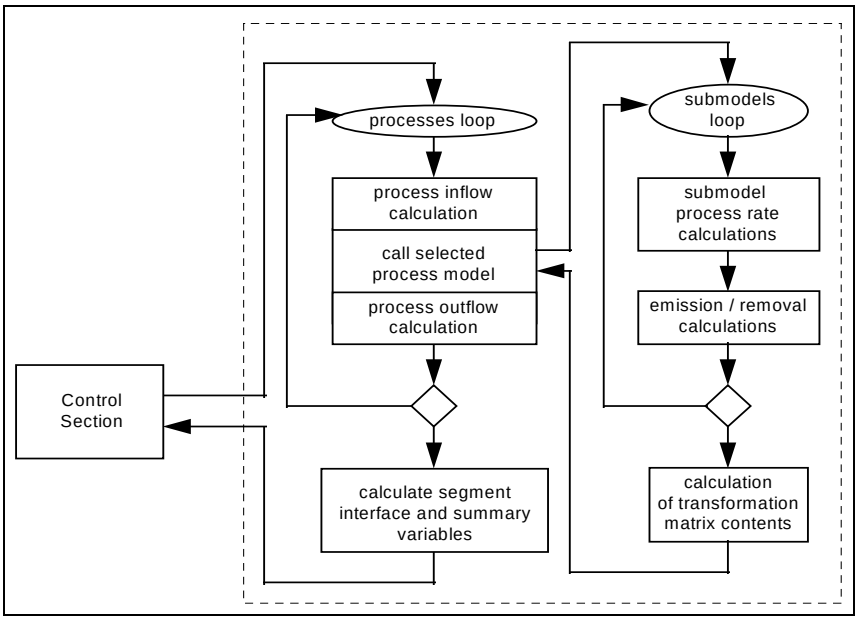

Figure 7. Model section

Processes Loop

In the processes loop, the equations (18) are sequentially solved, simultaneously with the calculation of the A and B matrix values. For each process (with k as row-index for the first terminal - input or emission - and with n sub-processes) the following steps are performed:

- first sub-matrix operation
  - = process inflow calculation
  - not performed for emission processes
  - according to equation (20) for transport / conversion processes - input section

- calling of the appropriate Process Model for the calculation of A and B values

- second sub-matrix operation
  - = process outflow calculation
  - according to equation (19) for emission processes
  - according to equation (21) for transport / conversion processes - output section

Process Models

In the Process Models, the A and B values associated with each process are calculated. This is performed during the sub-processes loop, where all the sub-processes in the considered process are sequentially selected.

Conversion

Conversion only applies to transport / conversion processes. Hence, this step is skipped when calculating emission processes. Conversion coefficients are entered into the A matrix rows which are associated with the output terminals of the process, as shown in equation (16). These general calculation method is given in (22).

$$\alpha_i = \prod_{j=1}^{n} \left(1 - R_j^i\right) \tag{22}$$

with
$\alpha_i =$ conversion coefficient for state variable i in a specific submodel
$R_j^i =$ removal efficiency of state variable i in conversion step j of the submodel
$n =$ number of conversion steps in the specific submodel

In the applied case, there is either only 1 conversion step for each sub-process (i.e., removal); or 2 conversion steps (i.e., removal and ground water leakage). The removal efficiencies R^i_j are calculated in the individual sub-processes. The conversion coefficient α_i is calculated in the process model.

Emission

Emission can occur in all processes (i.e., in emission as well as transport / conversion processes). If required, the appropriate sub-processes are called for the calculation of emission values. These values are stored in the B_i vectors, at the positions associated with the considered processes, terminals and sub-processes.

Process Outflow

The state variable values at the output terminal of a specific process, are split into fractions and sent to their different destinations. This is done by means of outflow destination links. The fraction going from the situation with X-vector index p to the situation with X-vector index q, is entered in the A_i matrices at the location (q,p). Equation (23) shows that in the p^{th} column of the A^i matrices, the values below element $a^i_{p,p}$ represent the different destination fractions of x^i_p.

$$\forall\, q > p : a^i_{q,p} = f_{x^i_p \to x^i_q} \qquad \text{hence} \quad A^i_{col.\,p} = \begin{bmatrix} \vdots & \} \; p \text{ rows} \\ \hline f_{x^i_p \to x^i_{p+1}} & \\ \vdots & \} \; n - p \text{ rows} \\ f_{x^i_p \to x^i_n} & \end{bmatrix} \tag{23}$$

In the parameters set, the list of outflow destination fractions is not complete. If a specific process has n destinations, then only n-1 fractions are in the parameters set. These are the fractions f_2 through f_n. The remaining fraction f_1 is calculated as:

$$f_1 = 1 - \sum_{i=2}^{n} f_i \tag{24}$$

Hence it is clear that in the 'single-output' case, no destination fractions are present in the parameters set, and the fraction f_1 = 1.

Groundwater Leakage

Leakage to groundwater can be an exception to the regular outflow destination system. It can either occur before or after the conversions in the considered process. This is determined in the non-geo-referenced parameters set, with the parameter *groundwater_conservative*.

If this parameter is set to 0, the regular outflow destination system is used, i.e. the leaks occur after the conversions in the process.

If *groundwater_conservative* is set to 1, leaks occur before the process conversion steps:

- The process outflow destination link to groundwater now points to the input terminal of the process (instead of the output terminal). This way, the unmodified values are sent to the Groundwater Process.
- The other outflow destination fractions are modified. The 'to groundwater' fraction is moved to the new destination link (i.e., the one associated with the process input terminal). Hence, the other destination fractions are to be updated. They are re-scaled, as their sum has to be 1.

River Model

In the river fate model, a specific calculation is performed to obtain the 'internal' concentration value $C_{internal}$. This is defined as the calculated arithmetic average over the length of a stretch assuming 1st order exponential decay: (with HRT = hydraulic residence time in the stretch [time], k = the in-stream removal rate [time^{-1}], C_{start} = concentration at the beginning of the stretch):

$$C_{internal} = \frac{1}{HRT} \cdot \int_0^{HRT} C_{start} \cdot e^{-kt} \cdot dt = \frac{C_{start}}{HRT} \cdot \left[\frac{1}{-k} \cdot e^{-kt} \right]_0^{HRT} = \frac{C_{start}}{k \cdot HRT} \cdot \left(1 - e^{-k \cdot HRT}\right)$$

If k = 0, $C_{internal}$ is set equal to C_{start}.

Calculation of Interface Variables and Summary Variables

The final step in each segment is the calculation of the Interface Variables (i.e., the variables which are used for communication between segments) and the Summary Variables (i.e., the variables which are used for the calculation of the final simulation results).

Interface Variables

The Interface Variables are taken directly from the state variables associated with the input and output 'interface processes'. The latter are:

- the process which receives influent from upstream segments, and
- the process which sends its effluent to downstream segments.

In the applied case for GREAT-ER, the River Process represents both the input and the output interface. Hence, the Interface Variables are identical to the River Process state variables.

Summary Variables

The Summary Variables are not necessarily taken directly from the state variables. They can also be derived from state variables, or they can be calculated individually in the different process models. The choice and the calculation of the Summary Variables is model dependent.

The Summary Variables have no influence on the simulation process itself, only on the simulation's output.

3.3. Data I/O Section

The Data I/O Section is called from the Control Section. It is used to read the ASCII input file(s) provided by the GIS, and to write the ASCII output file which can later be read by the GIS. It is also used to display output information on a user-specified log device. The Data I/O Section can also be called from the model section, to display detailed simulation results of individual segments and 'shots'.

3.3.1. Input Section

The simulation control parameters are read from the appropriate files. Non-geo-referenced data and geo-referenced datasets (river data, discharge data and emission data) for each segment are read from the input data files. These files use the ASCII format. For a detailed description of the file structures, see higher (4.1).

3.3.2. Output Section

Several output processes can be performed:

Output Process to GIS

- The model results (i.e. the Summary Variable distributions for each segment) which are to be sent to the GIS are stored in an ASCII file. The values are tab-separated within a segment, and new line separated between segments.

Reporting

Depending on user selections, defined in the Output initialization file, the following can be reported:

- A report of the Summary Variables distributions for each segment can be generated. This report can be logged to a user-specified output device (screen or file).
- A report of the State Variables, Interface Variables and Summary Variables for each 'shot' and each segment can be sent to a user-specified output device.
- A report of the discrete parameter values, generated in each 'shot' can also be generated and sent to the output device.

Error Messages

Error messages are also processed by the Data I/O Section. These messages are sent to the Error Log File and also to the selected output device(s).

Output Devices

The user-specified output device can be an ASCII file, and / or the computer screen. If on-screen logging is selected, simulator messages are also displayed. If the Graphical User Interface (GUI) is operational, an indication of the simulation progress is given.

3.3.3. User Interface

Under the Microsoft Windows NT operating system, a Graphical User Interface (GUI) is available. This service is not provided in the UNIX version of the simulator.

Progress Window

If the GUI is operational, and the simulator was set to operate visibly (using the command line switches), the simulation progress is displayed in a window, as well as an estimate of the still required simulation time (Figure 8).

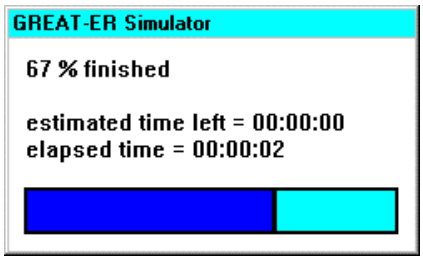

Figure 8. User Interface

User Break

By clicking a mouse button while the mouse arrow is within the GREAT-ER Simulator Window, or by pressing a key when the GREAT-ER Simulator Window is activated, the simulation is paused and the simulation window can be moved by dragging it with the mouse. Alternatively, the simulation can be interrupted by the user.

3.3.4. Simulator Command Line

The simulator is started by executing the command ‘SIMUL’ in the directory where the simulator files are located. Next to the simulation control and initialization file, which is described higher, several control actions can be called through the command line options, which are:

-x <window left x coord>	progress window location X
-y <window top y coord>	progress window location Y
-t <window title>	progress window title
-d <work directory>	simulation directory name
-i <INI file name>	initialization file name
-e <error file name>	error messages file name
-s <nr Monte Carlo shots>	number of Monte Carlo shots to be used
-h	switch to hide progress window
-m	switch to show control messages
-w	switch to hide error warning messages
-l	switch to create a log file
-?	display command line options

4. File Structure

4.1. Input Files

Several data files (ASCII format) are used for the transfer of all the required input data and model parameters of all considered geographical segments:

- Control Files:
  - Simulation control and initialization file
  - Distribution type file

- Data Files:
  - Simulation Scenario Independent data files:
    - Non-geo-referenced data file
    - River class data file
    - River data file
    - Waste water pathway data file
    - Waste water treatment plant data file
  - Simulation Scenario Dependent data files:
    - Chemical data file
    - Emission data file

4.1.1. Control Files

Initialization file

The simulation control and initialization file contains information on the geographical system which has to be simulated and on the way the simulation is performed. It also describes which output has to be sent to the log device, and which is the log device. It must have a strictly defined format, as shown below:

```
LOG CONTROL PARAMS = 0 or 1
LOG SHOT PARAMETERS = 0 or 1
LOG SHOT RESULTS = 0 or 1
LOG SUMMARY RESULTS = 0 or 1
NON-GEOREF FILE = non-geo-referenced filename
CHEMICAL FILE = chemical filename
RIVER FILE = river filename
RIVER CLASS FILE = river class filename
DISCHARGE FILE = waste water pathway filename
WWTP FILE = WWTP filename
EMISSION FILE = emission data filename
WWTP EXCEPTIONS FILE = “none” or WWTP exceptions filename
RIVER EXCEPTIONS FILE = “none” or river exceptions filename
DISTRIBUTION TYPE FILE = distribution type filename
OUTPUT FILE = output filename
RESULTS FILE = results filename
USE SITE-SPECIFIC EXCEPTIONS = 0 or 1
DEFAULT COMPLEXITY MODE RIVER = 1, 2 or 3
DEFAULT COMPLEXITY MODE WWTP = 1, 2 or 3
DEFAULT COMPLEXITY MODE SEWER = 1, 2 or 3
```

```
IMPOSE DEFAULT COMPLEXITY MODE RIVER = 0 or 1
IMPOSE DEFAULT COMPLEXITY MODE WWTP = 0 or 1
IMPOSE DEFAULT COMPLEXITY MODE SEWER = 0 or 1
FIRST SEGMENT = root segment
```

The meaning of this information is:

LOG CONTROL PARAMS
: switch to log simulation control parameters or not

LOG SHOT PARAMETERS
: switch to log parameters from each Monte Carlo shot or not

LOG SHOT RESULTS
: switch to log results for each Monte Carlo shot or not

LOG SUMMARY RESULTS
: switch to log the statistically analyzed results or not

NON-GEOREF FILE
: name of file which contains the non-geo-referenced data

CHEMICAL FILE
: name of file which contains the chemical-specific data

RIVER FILE
: name of file which contains the river network information

RIVER CLASS FILE
: name of file which contains the river class data

DISCHARGE FILE
: name of file which contains the waste water discharges data

WWTP FILE
: name of file which contains the WWTP information

EMISSION FILE
: name of file which contains the emission data

WWTP EXCEPTIONS FILE
: name of file which contains WWTP-specific removal exceptions

RIVER EXCEPTIONS FILE
: name of file which contains river-specific in-stream removal exceptions

DISTRIBUTION TYPE FILE
: name of file in which the distribution type of each parameter is given

OUTPUT FILE
: name of file to which the model output (only values) is to be written

RESULTS FILE
: name of file to which the model results (values + text) is to be written

USE SITE-SPECIFIC EXCEPTIONS
: switch to use site-specific exceptions for removal or not

DEFAULT COMPLEXITY MODE RIVER
: default model complexity to be used for the river model

DEFAULT COMPLEXITY MODE WWTP
: default model complexity to be used for the WWTP model

DEFAULT COMPLEXITY MODE SEWER
: default model complexity to be used for the sewer model

IMPOSE DEFAULT COMPLEXITY MODE RIVER
: switch to use the default or the site-specific preferred river model complexity mode

IMPOSE DEFAULT COMPLEXITY MODE WWTP

switch to use the default or the site-specific preferred WWTP model complexity mode

IMPOSE DEFAULT COMPLEXITY MODE SEWER

switch to use the default or the site-specific preferred sewer model complexity mode

FIRST SEGMENT

GIS reference code of root segment

Distribution type file

The distribution type file contains information on the statistical distribution type which is used for each model input parameter and by each summary variable. The different possible distributions are represented by means of codes, as shown in Table 1.

Table 1. Distribution types

Code	Distribution name	Nr. of parameters	Distribution described by:
0	<none>	1	mean
1	normal	2	mean + standard deviation
2	lognormal	2	mean + standard deviation
3	uniform	2	minimum + maximum
5	'flow-type'	2	mean + 5^{th} percentile (= Q95 low flow)

The file consists of the following sections:

- Non-geo-referenced parameters section
- Chemical section
- River section
- Discharge section
- Emission Data section
- Summary Variables section

Each section contains the distribution type code for all the parameters (or variables) relating to the section. The sequence of the codes is determined by the (hard-coded) parameters (or summary variables) structure. Hence, it depends on the model implementation. The file format is not strictly defined, however a number of rules (described below) must be followed.

4.1.2. Simulation Scenario Independent Data Files

Five files are used to describe the considered river system: the non-geo-referenced data file, the river and river class data files, the waste water pathway data file, and the WWTP data file. These files contain environmental information that is assumed constant. The contents only have to be changed when significant changes to the environment occur (e.g. new treatment infrastructure, change of the population density,...). Hence, these files can be considered 'read-only'. The actual number of values per parameter is determined by the distribution type file. The sequence of the parameters is determined by the (hard-coded) parameters structure.

Non-geo-referenced data file

This file contains the values for each non-geo-referenced parameter. Non-geo-referenced parameters apply to each segment, and are used for all chemicals, e.g. the organic carbon fraction in sewage solids. The sequence of the non-geo-referenced parameters is determined by the (hard-coded) parameters structure.

```
<Non-geo-referenced data file>
<Set of non-geographically-referenced values and model parameters>
```

River class data file

The river class data file contains 'shared' information, which applies to each river stretch belonging to a specific river class. The first value is the number of segments present in the file. The structure is as follows:

```
<River class data file>
N_river classes
<River Class 1: River Class ID>
<River Class 1: river model parameters...>
...
<River Class N: River Class ID>
<River Class: river model parameters...>
```

River data file

The river data file holds the general information about each segment as well as the river fate model parameters. The first value is the number of segments present in the file. The second is the ID of the most downstream segment (from which the tree-walking algorithm should start). The general segment information contains the river network interconnection data (see section 5.2). The file structure is as follows:

```
<River data file>
N_river
<Segment ID of most downstream segment>
<Segment 1: Segment ID>
<Segment 1: general information...>
<Segment 1: river model parameters...>
...
<Segment N: Segment ID>
<Segment N: general information...>
<Segment N: river model parameters...>
```

Waste water pathway data file

The waste water pathway (or discharge) data file contains the waste water pathway model parameters. The first value is the number of discharge points present in the file. The file structure is as follows:

```
<Waste water pathway data file>
N_discharge
<Discharge 1: Segment ID>
<Discharge 1: Emission ID>
<Discharge 1: waste water pathway model parameters...>
...
```

```
<Discharge N: Segment ID>
<Discharge N: Emission ID>
<Discharge N: waste water pathway model parameters...>
```

The discharge data file is related to the river data file by means of the Segment ID (i.e. the geographical reference code of the segment): the discharge with Segment ID <nnn> is assumed to enter the river into the stretch with the same Segment ID <nnn>. For each waste water discharge, and Emission ID is also required. This points to information in the emission data file. The number of discharge points in a river system need not be the same as the number of river stretches, as some river stretches do not receive waste water inputs.

WWTP data file

The WWTP data file contains information on waste water treatment plants. The first value is the number of WWTPs present in the file. The file structure is as follows:

```
<WWTP data file>
N_WWTP
<WWTP 1: WWTP ID>
<WWTP 1: waste water treatment plant model...>
...
<WWTP N: WWTP ID>
<WWTP N: waste water treatment plant model...>
```

4.1.3. Simulation-Dependent Data Files

Two files are used to contain the simulation-dependent information. One is non-geo-referenced, and contains data which has no spatial variation. A geo-referenced file contains the chemical emission data. The actual number of values per parameter is determined by the distribution type file.

Next to this, 2 optional files can be used: the WWTP and river removal exception files. In these files, site-specific removal percentages (for WWTP) or in-stream removal rates (for rivers) can be specified.

Chemical data file

This file contains the values for all chemical-specific parameter. These parameters apply to each segment, e.g. the chemical molar mass. The sequence of the chemical parameters is determined by the (hard-coded) parameters structure.

```
<Chemical data file>
<Set of chemical-specific values and model parameters>
```

Emission data file

The emission data file contains the specific product consumption per capita, the non-domestic chemical emission fluxes, and the chemical emissions due to runoff. The emission data file is related to the waste water pathway data file by means of the Emission ID. For each discharge point present in the waste water pathway file, an Emission ID is given, which points to and emission data information point in the emission data file. The first value in the file is the number of emission data information points present. The file structure is as follows:

```
<Emission data file>
N_emissions
<Emission Data Information Point 1: Emission ID>
<Emission Data Information Point 1: per capita product consumption>
<Emission Data Information Point 1: nondomestic emission flux>
<Emission Data Information Point 1: runoff emission flux>
...
<Emission Data Information Point N: Segment ID>
<Emission Data Information Point N: per capita product consumption>
<Emission Data Information Point N: nondomestic emission flux>
<Emission Data Information Point N: runoff emission flux>
```

Exception files

In the exception files, site-specific removal can be specified.

```
<WWTP exceptions file>
N_WWTP exceptions
<WWTP Removal Exception Point 1: Segment ID>
<WWTP Removal Exception Point 1: Site-specific removal efficiency>
...
<WWTP Removal Exception Point N: Segment ID>
<WWTP Removal Exception Point N: Site-specific removal efficiency>
```

```
<River exceptions file>
N_river exceptions
<River In-stream Removal Exception Point 1: Segment ID>
<River In-stream Removal Exception Point 1: In-stream removal rate>
...
<River In-stream Removal Exception Point N: Segment ID>
<River In-stream Removal Exception Point N: In-stream removal rate>
```

4.1.4. Input File Format

The initialization file must strictly follow the format shown above.

For the distribution type file and the data files, a less rigid format is required. All numerical values in these files are read, in order of appearance. Values can be separated by any non-numerical character or character string, as well as tabs and new lines. Comment regions can be created by using a sequence of two slash characters. When '//' is encountered on a line, the rest of the line is skipped.

For geo-referenced data files, it is advised to use tab-separation within a segment, and new line separation between segments. This way, the data files can easily be imported and manipulated in a spreadsheet program (such as Excel).

4.2. Output Files

4.2.1. Output File

The 'technical' output file is a plain ASCII file which is used to transfer the simulation results between the simulator and the GIS user interface. Each row in this file corresponds with one geographical segment. The values in one row are tab-separated. The first value in each row is the geographical segment ID. The following values correspond with respectively:

river summary variable: input	concentration at start of river stretch
river summary variable: output	concentration at end of river stretch
river summary variable: internal	average concentration in river stretch
waste water pathway summary variable: input	WWTP influent concentration
waste water pathway summary variable: output	WWTP effluent concentration
waste water pathway summary variable: internal	not used

Each of these variables may be described by 1 or 2 parameters, depending on whether a distribution was selected or not. Typically log-normal or normal distributions are used, hence for each variable the first value is the mean and the second is the standard deviation.

4.2.2. Results File

The results file is only created when this is requested by the -l command line switch. Depending on the different LOG switches in the initialization file, the results file may contain a description of the control parameters, the segment's parameters, the individual shot results, and the summary results. No strict format was followed, as the output is not intended to be processed by software but is to be informative to the user.

4.2.3. Error File

if fatal errors occur during the simulation, the file error.txt is created in the work directory. In this file, an error message is contained. If the simulation was successful, the file error.txt does not exist.

5. Internal Data Structure

5.1. Parameters and Variables

Two types of data are used:

- parameters and input data
  - non-geo-referenced parameters
    (constant for one specific simulation, within the considered geographical system).
    - general non-geo-referenced parameters
    - chemical parameters
    - river class data
  - geo-referenced parameters (segment-specific)
    - river data
    - waste water pathway and WWTP data
    - emission data

- variables
  - state variables
    - used for the model calculations
    - exist only within one segment (discarded after simulation of the current segment)
  - interface variables:
    - represent the input / output of a complete segment (considered as a 'black box')
    - used as interface between different segments
    - exist for all segments, within one 'shot' (discarded after simulation of the current 'shot')
  - summary variables:
    - used to memorize any state variables which are required in the simulation output
    - exist for all segments, within one 'shot' (discarded after simulation of the current 'shot')
  - between-'shot' sums of powers of summary variables
    - used to calculate result distribution parameters

5.1.1. Distributed and Discrete Data

The parameters and input data consist of distribution arrays and - in parallel - discrete arrays. The distribution arrays are obtained directly from the data files, and their internal structures are identical. The discrete arrays are generated from the distribution arrays by means of Monte Carlo simulation: a discrete array contains one 'shot' of the distributed parameters set. They are re-initialized each time a new 'shot' is performed.

The discrete versions of the river and the discharge data are joined into one discrete array of geo-referenced data. The non-geo-referenced data and the emission data have separate discrete data arrays.

The state and interface variables only exist in the discrete environment of one 'shot'. The summary variables also consist of a discrete array. However, the between-'shots' sums of powers of summary variables (i.e. sums, sums of squares, and if required sums of higher order) are memorized and updated during the length of the simulation. These sums are finally used to calculate the parameters of the results' distributions. The latter are stored in the distributed summary variables array.

The different arrays and the links between them are shown in Figure 9.

5.1.2. Distribution Data Array Structures

The structure of the distribution data arrays is identical to the structure in the data files (see 4.1).

5.1.3. Discrete Data Array Structures

Non-geo-referenced Data

The sequence of the non-geo-referenced parameters, of the chemical parameters and of the river class parameters in their array is identical to the sequence in the respective data files, and is determined by hard-coded parameter definitions. These non-geo-referenced arrays are global variables, hence they are always available to each segment's calculation.

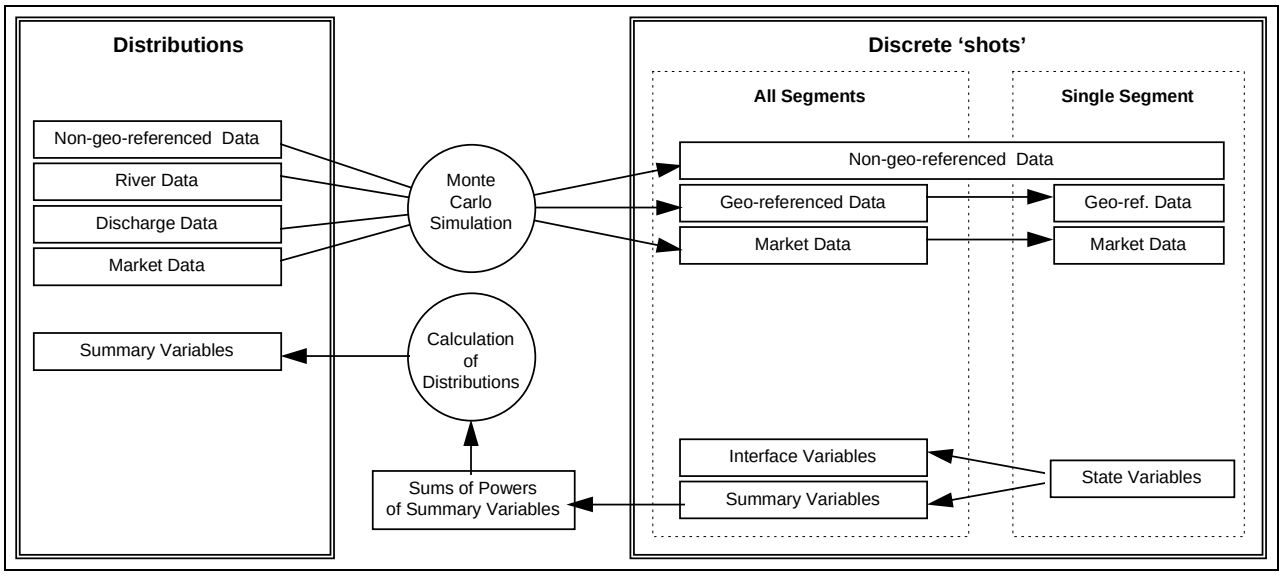

Figure 9. Distributed and discrete data

Geo-referenced Parameters

The Geo-referenced parameters array consists of sequential blocks, which represent the different segments. Blocks are further split up per process. Two types of blocks are used: one type contains only the general segment information and the river model parameters, while the other type also contains the discharge model parameters. The latter type is only used in the case a waste water discharge occurs into the considered segment.

For the simulation of each individual segment, only the block associated with this segment is passed on. For the simulation of one specific process, the subset of parameter values associated with this process is selected and passed on. The subset concerning the general segment information is also passed on to each process model.

The data structure within one block (i.e. segment) is shown below:

- general segment information (geographical structure)
- river parameters list
- discharge parameters list (if applicable):
  - 1st process
  - ...
  - n^{th} process

Emission Data

The emission data array is split up per segment. For each segment, the Segment ID and the product consumption is given. For the simulation of individual segments, only the product consumption value is passed on.

5.1.4. Distribution Variables Array Structures

The structure of the distribution Summary Variables uses the same segment sequence as the river data file. For one segment, the Summary Variable distribution parameters are listed according to a hard-coded sequence. The number of distribution parameters for each Summary Variable is determined by their distribution type (which is given in the distribution type file).

5.1.5. Discrete Variables Array Structures

State Variables

State Variables are always expressed as fluxes. They are only used internally within the simulation of individual segments. The information contained in the state variables array is lost each time a next segment is selected. The array is split up per state, the different states are:

- Water Flow (m^3/s)
- Chemical Flux (g/s)

Within a state's sub-array, different subsets contain the values for each process and each process terminal. The sequence of processes is always: (1) emission processes, (2) transport / conversion processes. The different terminals are emission (for the emission model class), or input and output (for the transport / conversion model class). Within transport / conversion processes, the terminal sequence is always (1) input, (2) output. Within one terminal, there is a further division per sub-process.

Data structure within one state:

- Process 1 (assume: emission process)
  - emission:
    - from sub-process 1-1
    - ...
    - from sub-process 1-n_1
- ...
- process n (assume: transport / conversion process)
  - input:
    - into sub-process n-1
    - ...
    - into sub-process n-n_n
  - output:
    - from sub-process n-1
    - ...
    - from sub-process n-n_n

Interface Variables

A segment's input Interface Variables are calculated from the upstream segment(s) Interface Variables. The output Interface Variables are taken from the river output values in the State Variables set. The sequence of the segments in the array is identical to the sequence in the geo-referenced parameters array. Within one segment, Interface Variables are grouped by terminal (i.e. input and output).

Data structure within one block (i.e. segment):

- Segment Input
  - Water Flow
  - Chemical Flux
- Segment Output
  - Water Flow
  - Chemical Flux

Summary Variables

The sequence of the segments in the array is identical to the sequence in the geo-referenced parameters array. The information which is memorized, and the sequence within one segment, is determined by (hard-coded) definitions. Summary Variables are not necessarily copied from State Variables, but can be calculated from them. Concentrations are used instead of separate chemical fluxes and water flows.

The following are used as Summary Variables:

- Maximal chemical concentration in the river stretch (i.e. after mixing with a discharge)
- Minimal chemical concentration in the river stretch (i.e. upon outflow)
- Internal chemical concentration in the river stretch (i.e. an average value for the stretch)

For segments containing a discharge, the following Summary Variables are also included:

- Maximal chemical concentration in the small surface waters of the discharge catchment
- Minimal chemical concentration in the small surface waters of the discharge catchment
- Internal chemical concentration in the small surface waters of the discharge catchment

5.2. Representation of River Network Structure

the river network structure is contained in the general segment information of the geo-referenced parameters. This is found in the river data file. For each segment, the following information is needed to perform the recursive tree-walking simulation algorithm (see higher in section 3.1.3):

Segment ID	segment geographical reference code (integer value, negative values are also allowed)
Segment Type	type of 'geographical event' occurring in the segment: 0 = most downstream stretch (= 'root' of the network) 1 = nothing special (1 upstream stretch) 2 = confluence (2 upstream stretches) -2 = bifurcation stretch * 9 = hypothetical stretch, receiving two upstream stretches (cf. the sea)
Discharge	does the segment receive a waste water emission ? (1 or 0)
Upstream Code 1	Segment ID of first upstream stretch
Upstream Code 2	Segment ID of second upstream stretch * (only used for confluences and bifurcations)

* Bifurcation stretches: these are 2 'sister' stretches with one upstream 'parent'. The parent ID is given in Upstream Code 1; the sister ID is given in Upstream Code 2. Hence, the stretch directly upstream of a bifurcation is a normal stretch, while the 2 stretches immediately downstream of it are bifurcation stretches.

6. C Code Documentation

6.1. Description of Modules

6.1.1. C files

Initialization and Utilities:

simul.c	start of the simulation
init.c	declaration (+ if required initialization) of global variables / arrays (also contains the process interconnection information)
util.c	several system utilities
navigate.c	tools to navigate in the different arrays tools to navigate between segments

Data I/O Section:

data_in.c	reading of the input data from files
data_out.c	writing of the output to file (if required) writing to the log device
error.c	processing and displaying of error messages
usriface.c	generic user interface routines
gui.c	graphical user interface routines

Control Section:

control.c	calling of initialization calling of data input Monte Carlo simulation loop (calls the segments loop) calling of results distribution calculation calling of data output
mtcarlo.c	random shot generation calculation of distributions
mc-check.c	verification of random shots
treewalk.c	segments loop (calls the processes loop)

Model Section:

model.c	processes loop (calls the sub-processes loop) process models (include. sub-processes loop) calls submatrix operations
modcalc.c	specific calculation algorithms
modutil.c	submatrix operations outflow destination link calculations
r_river.c	river model calculations
r_wwtp.c	WWTP model calculations
r_misc.c	other model model calculations

6.1.2. Header files

For each C module, a header file is provided, in which the module's functions and global variables are declared. These are included in all modules which depend on that specific module. The header file init.h (which contains definitions of all global variables) is included in all modules.

Associated with gui.c, the header file extgui.h is included in any module which requires interaction with gui.c.

Next to these, a number of specific header files are used:

def.h ..... general: file names, distribution types, segment types,...
model specific: state variables, processes, sub-processes, destinations
(included in all modules)

names.h ..... names of state variables, units, processes, sub-processes,...
(included in data_in.c, data_out.c, model.c)

params.h ..... parameters structures
(included in init.h - and hence in all modules via init.h)

random.h ..... definitions required by the random number generator in mtcarlo.c
(included in mtcarlo.c)

6.2. Customized Data Types (typedefs)

struct	*NonGeoRef_ParPtr	pointer to structured non-geo-referenced data
struct	*Chemical_ParPtr	pointer to structured chemical data
struct	*RivClass_ParPtr	pointer to structured river class data
struct	*General_ParPtr	pointer to structured general segment data
struct	*River_ParPtr	pointer to structured river data
struct	*DomesticEm_ParPtr	pointer to structured domestic emission data
struct	*NonDomesticEm_ParPtr	pointer to structured non-domestic emission data
struct	*RunoffEm_ParPtr	pointer to structured runoff emission data
struct	*OnSiteTreatment_ParPtr	pointer to structured on-site treatment data
struct	*Sewer_ParPtr	pointer to structured sewer data
struct	*Treatment_ParPtr	pointer to structured treatment data
struct	*SmallSurf_ParPtr	pointer to structured small surface water data
struct	*GroundWat_ParPtr	pointer to structured ground water data
struct	*WWTP_ParPtr	pointer to structured WWTP data
struct	*PrimParPtr	pointer to structured primary settler data
struct	*ASParPtr	pointer to structured activated sludge data
struct	*TFParPtr	pointer to structured trickling filter data
struct	*EmissionParPtr	pointer to structured emission data
struct	ChemProp	structured chemical properties, derived once per shot
struct	*Map_Ptr	pointer to structured map data - to easily locate a segment's data (links to array location of upstream segments, river class, discharge data, emission data, WWTP data)

double	FloatType	floating point value
double*	FloatPtr	pointer to floating point value

6.3. Global variables available in all modules

6.3.1. General

char	SimulationTitle[]	title of current simulation run
char	WorkDirectory[]	name of work directory of the current simulation (i.e. location of input an doutput files)
int	NrMonteCarloShots	nr. of Monte Carlo iterations
int	LogCtrlParams	switch to write control parameters to the log file
int	LogShotParameters	switch to write individual shot parameters to the log file
int	LogShotResults	switch to write individual shot results to the log file
int	LogSummaryResults	switch to write summary results to the log file
int	LogToFile	switch to write anything to the log file or not
int	ShowMessages	switch to send messages to the user interface
int	ShowErrorMessages	switch to write control parameters to the log file
int	Visible	switch to write control parameters to the log file
int	ComplexityModeRiver	default complexity mode for the river model
int	ComplexityModeWWTP	default complexity mode for the WWTP model
int	ComplexityModeSewer	default complexity mode for the sewer model
int	ImposeComplexityModeRiver	switch to use the default or the site-specific preferred river model complexity mode
int	ImposeComplexityModeWWTP	switch to use the default or the site-specific preferred WWTP model complexity mode
int	ImposeComplexityModeSewer	switch to use the default or the site-specific preferred sewer model complexity mode
int	UseExceptions	switch to use site-specific exception data or not

6.3.2. Miscellaneous Simulation and Calculation Aids

int	FloatSize	size of a floating-point variable (in bytes)
int	IntSize	size of an integer variable (in bytes)
int*	SegmentDone	switch to check whether a segment has already been simulated
int*	SegmentParOffset	offset to locate a segment's parameters within the total parameter set
int	ParametersArrayOffset []	offset to locate a process's parameters within a segment's parameter set
int	StateVarsArrayOffset[]	offset to locate a process's state variables within the (segment's) set of state variables
int*	AsubMatrixOffset	offset to locate a process's information in the A matrix
FloatPtr	A	the A matrix (transport / conversion)
FloatPtr	B	the B vector (emission)

FloatPtr	X	the state variables vector
Map_Ptr	Map	'map' to easily locate a segment's data (links to array location of upstream segments, river class, discharge data, emission data, WWTP data)
int	ErrorFree	check to see if any errors were encountered
int	AllFinished	check to see if all required actions were undertaken successfully
char	C[]	character string used for several output purposes

6.3.3. Structural Aids

int	NrEmissionSubmodels	number of process in 'emission'
int	NrTransTreatSubmodels	number of process in 'transport / treatment'
int	NrSubmodels	total number of processes
int	ModelClass[]	array containing the model class of each process
int	NrSub[]	array containing the nr. of sub-processes of each process
int	NrDestinations[]	nr. of destinations for each process
int	Destination[][][2]	each destination code for each process (both process and sub-process)
int	DestNrBypass[]	nr. of destination to which each process's bypass is directed
int	NrParamsInProcess[]	nr. of parameters used by each process
int	NrParamsOnlyRiver	nr. of parameters if no waste water discharge occurs
int	NrParamsDischarge	nr. of parameters needed for a waste water discharge
int	OneStateVarSize	memory size needed for 1 state variable
int	AllStateVarSize	memory size needed for all state variables
int	AmatrixSize	memory size needed for the A matrix
int	AsubMatrixSize	memory size needed for A submatrices
long	AllIfaceSize	memory size needed for all interface variables
long	AllParamSize	memory size needed for all parameters
long	AllSummarySize	memory size needed for all summary variables
int	NrDistriParams[]	nr. of parameters needed for each distribution type

6.3.4. Size parameters of the simulated catchment

int	NrSegments	number of segments
int	NrSegmentsOnlyRiver	number of segments without waste water emissions
int	NrSegmentsDischarge	number of segments with waste water emissions
int	NrRivClasses	number of river classes
int	NrEmissions	number of emission data points
int	NrWWTPs	number of WWTP data points
int	NrWWTPExceptions	number of specific WWTP removal exceptions
int	NrRiverExceptions	number of specific river removal exceptions

6.3.5. Structure of Distributed Parameters and Variables

int*	NonGeoRefDistriParIndex	index to locate each non-georeferenced parameter in the array containing the distributed data
int*	ChemicalDistriParIndex	index to locate each chemical parameter in the array containing the distributed data
int*	RiverDistriParIndex	index to locate each river parameter in the array containing the distributed data
int*	DischargeDistriParIndex	index to locate each waste water pathway parameter in the array containing the distributed data
int*	EmissionDistriParIndex	index to locate each emission parameter in the array containing the distributed data
int*	RivClassDistriParIndex	index to locate each river class parameter in the array containing the distributed data
int*	WWTPDistriParIndex	index to locate each WWTP parameter in the array containing the distributed data
int*	SummaryDistriParIndex	index to locate each summary variablein the array containing the distributed summary variables
int*	NonGeoRefDistriType	array containing the distribution type of each non-geo-referenced parameter
int*	ChemicalDistriType	array containing the distribution type of each chemical parameter
int*	RiverDistriType	array containing the distribution type of each river parameter
int*	DischargeDistriType	array containing the distribution type of each waste water pathway parameter
int*	EmissionDistriType	array containing the distribution type of each emission parameter
int*	RivClassDistriType	array containing the distribution type of each river class parameter
int*	WWTPDistriType	array containing the distribution type of each WWTP parameter
int*	SummaryDistriType	array containing the distribution type of each summary variable
int	TotNonGeoRefDistriPar	total nr. of non-geo-referenced parameters (i.e. including the distribution's parameters)
int	TotChemicalDistriPar	total nr. of chemical parameters (i.e. including the distribution's parameters)
int	TotDischargeDistriPar	total nr. of waste water pathway parameters (i.e. including the distribution's parameters)
int	TotRiverDistriPar	total nr. of river parameters (i.e. including the distribution's parameters)
int	TotEmissionDistriPar	total nr. of emission parameters (i.e. including the distribution's parameters)
int	TotWWTPDistriPar	total nr. of WWTP parameters (i.e. including the distribution's parameters)
int	TotRivClassDistriPar	total nr. of river class parameters (i.e. including the distribution's parameters)

int	TotSummaryDistriPar	total nr. of summary variable values (i.e. including the distribution's parameters)
int	WWTP_ID_Location	location of WWTP ID in the distributed waste water pathway parameters file
int	Emission_ID_Location	location of Emision ID in the distributed waste water pathway parameters file

6.3.6. Parameters and Variables

FloatPtr	Distri_NonGeoRefParams	distributed non-geo-referenced parameters
FloatPtr	Distri_ChemicalParams	distributed chemical parameters
FloatPtr	Distri_RiverParams	distributed river parameters
FloatPtr	Distri_DischargeParams	distributed waste water pathway parameters
FloatPtr	Distri_EmissionParams	distributed emission parameters
FloatPtr	Distri_RivClassParams	distributed river class parameters
FloatPtr	Distri_WWTPParams	distributed WWTP parameters
FloatPtr	Distri_SummaryVars	distributed summary variables
NonGeoRef_ParPtr	NonGeoRefParams	discrete shot of non-geo-referenced parameters
Chemical_ParPtr	ChemicalParams	discrete shot of chemical parameters
FloatPtr	EmissionData	discrete shot of non-geo-referenced parameters
FloatPtr	WWTPData	discrete shot of non-geo-referenced parameters
FloatPtr	CSewer_Q_DWF	discrete shot of the ratio between actual flow and dry weather flow in the combined sewer system, for each river class
FloatPtr	WWTPExceptions	WWTP removal exceptions (not distributed)
FloatPtr	RiverExceptions	river removal exceptions (not distributed)

6.3.7. File System

FILE*	OutputFile	output file (containing the results sent to GIS)
FILE*	ErrorFile	error log file
FILE*	LogFile	log file (containing the results as text)
char	NonGeoRefFileName[]	file name of non-geo-referenced data file
char	ChemicalFileName[]	file name of chemical data file
char	RiverFileName[]	file name of river data file
char	DischargeFileName[]	file name of waste water pathway data file
char	EmissionFileName[]	file name of emission data file
char	WWTPFileName[]	file name of WWTP data file
char	RivClassFileName[]	file name of river class data file
char	DistriFileName[]	file name of distribution type file
char	OutputFileName[]	file name of output file
char	ResultsFileName[]	file name of log file
char	ErrorFileName[]	file name of error file
char	INIFileName[]	file name of initialization file
char	WWTPExceptionsFileName[]	file name of WWTP removal exceptions file
char	RiverExceptionsFileName[]	file name of river removal exceptions file

6.4. Global variables available in some modules

6.4.1. Defined in control.c

int	Shot	nr. of current Monte Carlo shot
time_t	StartTime	time at which the simulation was started

6.4.2. Defined in gui.c

int	Change	switch to see if user interface needs to be changed
int	ExitCode	switch to see if user interrupted the simulation
char	DoneBarTxt[]	text in progress window

6.4.3. Defined in model.c

FloatType	RiverInternal[]	internal river state variables for this segment
FloatPtr	SegmentRivClass	river class associated with current segment
Map_Ptr	ThisSegmentMap	Map associated with current segment
FloatType	ThisSegmentID	current segment ID
EmissionParPtr	Em	emission data of this segment
ChemProp	CP	chemical properties
FloatType	FugacityAir	chemical fugacity in air (mode 3 WWTP model)
FloatType	DryWeatherFlow	domestic dry waste water flow in this segment
FloatType	WWTP_DryWeatherFlow	dry WWTP influent flow in this segment
FloatType	WaterUse	domestic per capita water use in this segment
FloatType	FugacityWater	chemical fugacity in water (mode 3 WWTP model)
FloatType	FugacityML	chemical fugacity in mixed liquor (mode 3 WWTP model)
FloatType	FugacityPrimSludge	chemical fugacity in primary sludge (mode 3 WWTP model)
FloatType	DOSaturationML	saturation dissolved oxygen in mixed liquor (mode 3 WWTP model)

6.4.4. Defined in mtcarlo.c

FloatType	FlowPercentile	flow percentile used in the current shot

6.4.5. Defined in navigate.c

struct	RowType[]	structure used to assist in navigating the A matrix

6.4.6. Defined in r_wwtp.c

FloatType	PrimaryEffluentSS	suspended solid in primary effluent
FloatType	fTreatedPrimary	fraction of waste water receiving primary treatment
FloatType	PurePrimaryRemoval	predicted chemical removal in primary treatment only
int	CurrentWWTPComplMode	complexity mode used in the current WWTP

6.4.7. Defined in treewalk.c

int	Counter	counter used in displaying the progress window

6.4.8. Defined in usriface.c

int	DisplayStatus	switch used in the progress window display
int	Progress	simulation progress (% finished)
int	ElapsHours	hours elapsed since the start of the simulation
int	ElapsMinutes	+ minutes elapsed since the start of the simulation
int	ElapsSeconds	+ seconds elapsed since the start of the simulation
int	Hours	estimated hours to go
int	Minutes	+ estimated minutes to go
int	Seconds	+ estimated seconds to go

6.5. Code Portability

The entire code for the GREAT-ER simulator was written in standard ANSI C. It can be compiled on UNIX platforms as well as Windows NT platforms. Both under Microsoft Windows NT 4.0 and under UNIX, the code was compiled using the public domain GNU C Compiler (GCC) (Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111, USA). Note that when the compiler switch -DUNIX is used, the Windows NT specific graphical user interface routines will not be compiled. The Windows NT version of the GCC compiler, RSXNT, can be downloaded from ftp.uni-bielefeld.de.

6.6. Performance Test

A simplified test model was implemented in the simulator. A (hypothetical) input data set was generated, containing 16,000 segments, each with an associated discharge point. In the Monte Carlo simulation, 1,000 ‘shots’ were calculated. Such a simulation was assumed to represent a realistic worst case regarding river system size. The internal memory requirement (RAM) for this simulation was found to be approximately 11 Mb.

The test simulation run was performed without any problem on a low-end Windows NT workstation (Pentium 150 MHz processor). The required calculation time was approximately 1 hours and 30 minutes.

7. References

Knuth, D.A. (1981). The art of computer programming. Second edition. Volume 2. Seminumerical algorithms. Addison-Wesley, Reading (Massachusetts), 701 p.

NRA (1995). SIMCAT 4.13. A guide for users. Environment Agency, UK, 112 p.

Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery B.P. (1992). Numerical Recipes in . The art of scientific computing. Second edition. Cambridge University Press, Cambridge, 1020 p.