IUT

INSTITUTE OF ENVIRONMENTAL ENGINEERING
faculty of civil engineering
university of innsbruck

CITY DRAIN

INTEGRATED URBAN DRAINAGE

USER MANUAL

Authors:
S. Achleitner and W. Rauch

tirol

STADT INNSBRUCK

Supported by the Fund for Scientific Research established by the state of Tyrol/Austria.
(Wissenschaftsfond des Landes Tirol).

IMPRINT

CITY DRAIN 1.0 – an open source Matlab/Simulink library for integrated simulation of urban drainage systems. (2005)

Software requirements:	MATLAB Release 12 (or higher)
Program Language:	Matlab / Simulink
Program Size:	~17 MB
Availability:	*http://umwelttechnik.uibk.ac.at/*

The software is a freeware and may be downloaded at *http://umwelttechnik.uibk.ac.at/*. A copy may as well be obtained by contacting the Institute of Environmental Engineering.

Authors:

Dipl.-Ing. **Stefan Achleitner**
Univ.-Prof. Dipl.-Ing. Dr. techn. **Wolfgang Rauch**

IUT - Institute of Environmental Engineering
faculty of civil engineering, university of innsbruck

Technikerstraße 13, 6020 Innsbruck, Austria
home: umwelttechnik.uibk.ac.at
mail: umwelttechnik@uibk.ac.at

Acknowledgements:

The authors gratefully acknowledge the support granted by the Fund for Scientific Research established by the state of Tyrol/Austria. (Wissenschaftsfond des Landes Tirol).

Further we would like to thank Sara De Toffol and Heiko Kinzel for their help in programming and testing of CITY DRAIN 1.0.

Disclaimer

CITY DRAIN 1.0
an open source Matlab/Simulink library for integrated simulation of urban drainage systems.

Copyright (C) 2005 , Stefan Achleitner and Wolfgang Rauch
Institute of Environmental Engineering, University of Innsbruck

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble
The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Lesser General Public License instead.) You can apply it to your programs, too.
When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.
To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.
For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.
We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.
Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.
Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.
The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".
Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.
You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:
a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program

itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)
These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.
Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.
In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)
The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.
If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.
If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.
It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.
This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.
Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
END OF TERMS AND CONDITIONS

CONTENT

1 INTRODUCTION

1.1 General Purpose of Integrated Modelling

The aspect of improving ambient water quality, based on the overall management of river basins gained importance during the last years (Blöch, 1999). The emphasis is being put on the improvement of the receiving water quality as well as on the overall management of river basins as requested also by the European water framework directive (WFD). Both aspects require a change in the design procedures for urban drainage systems. Reason is that the application of design rules based on emission criteria does not necessarily lead to an improvement of the water quality – at least they are limited to a certain extend (Lau et al., 2002; Lijklema, 1995). Thus a shift is recently experienced from end of pipe design criteria to ambient water quality approaches (Achleitner et al., 2005). For the application in practice software tools are required that are capable of modelling urban drainage systems (including the receiving water) in an integrated manner. Rainfall as the elementary input source is of irregular occurrence in intensity and duration, which leads to the need of long term simulations for being capable of a systems performance.

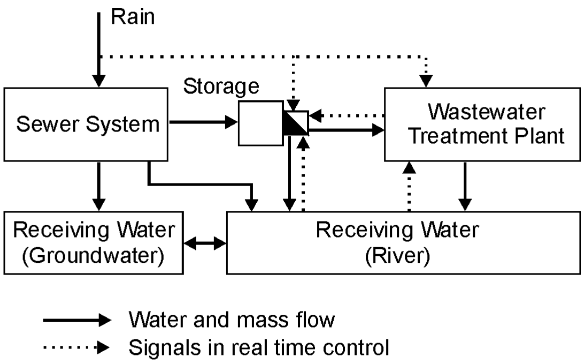


Fig. 1. Schematic on the main elements and information flow in an integrated model (redrawn from (Rauch et al., 2002))

The schematic in Fig. 1 illustrates the main elements and information flow occurring.

1.2 Dominant processes and complexity of models

1.2.1 Principals

A software for integrated modelling may incorporate a variety of models covering hydraulics, mass transport, processes for conversion of matter etc. within the subsystems. Main objective is the prediction of the system performance including the receiving water quality. For choosing the appropriate models it is therefore vital to characterise the impacts onto the receiving water with regard to their type (hydraulic, chemical, bio-chemical,…) and duration (e.g. acute, delayed, accumulating).

Regarding the time scale for modelling not only the dynamics of the relevant processes in the drainage system itself are to be considered but also the duration of the impacts (and associated processes) in the receiving waters. E.g. acute pollution occurs instantly and requires short term modelling whereas accumulative effects in the receiving water can only be covered within a long term simulation effort. But also the stochastic nature of rainfall as the source of impacts in an urban catchment needs to be considered. Single rain events are often source for acute effects in the receiving water such as hydraulic stress or pollutants entering the receiving water. The assessment of those is based on an evaluation of frequency, magnitude and duration of the impact (see. e.g. Harremoës and Rauch (1996)) and thus requires a statistical interpretation. This again is possible only within the framework of long term simulation studies.

Overall the computation in CITY DRAIN © is based on an fixed discrete time steps approach where each subsystem uses the same time increments, usually being predetermined by the timely resolution of the rain data used. Models implement for hydraulics and mass transport are formulated for discrete time steps Δt.

1.2.2 Computational aspects for hydraulics

Flow of water in both sewers and rivers is described by the continuity and momentum equations. The latter is known as the Navier-Stokes or Reynolds equation. The actual form of a hydrodynamic model depends on assumptions made on characterizing turbulence but for water quality purposes mostly the well-known, cross-sectionally integrated (1D) Saint Venant equations or approximations to these equations are used. Different levels of simplifications of the momentum equation are known for describing unsteady flow. Most simple approximation is the kinematic wave model being valid where backwater effects are negligible. All hydrodynamic equations have in common that they are demanding from a computational point of view. There for a variety of simpler conceptual models where developed (frequently denoted as hydrological models). These as well respect conservation of mass but use conceptual relations instead of momentum equations. The rapid simulation with conceptual models puts them in favour to hydrodynamic models regarding computational effort. Effects such as pressurized flow or backwater effects cannot be covered. For allowing long term simulations the blocks implemented in CITY DRAIN are based on purpose on simple conceptual models for hydraulics.

1.2.3 Computational aspects for transport and conversion of matter

For limiting the effort of modelling only relevant pollutants and processes need to be considered. Neglecting issues of secondary importance is required to avoid unnecessary complexity of models. Transport models describe in principle only the flow of soluble and conservative matter through the system. Effects such as physical or biological conversion processes (sedimentation, degradation, etc) are considered by extension of the transport equations.

1.3 Why realising CITY DRAIN in Matlab/Simulink

Basic idea was to create an open source toolbox for integrated modelling of urban drainage systems. For the use in the daily engineering work such software tools are required to be simple in handling and to provide a certain flexibility to be adjustable for different scenarios. Different subsystems should be freely arrangable and connectible to each for describing an integrated urban drainage system and the fluxes of water and matter.

The principle of block-wise modelling of integrated systems in CITY DRAIN has been developed in a Matlab/Simulink© environment. The platform is widely used for all different kinds of dynamic simulations and was found suitable as hosting environment for the CITY DRAIN© software. On the one hand the platform is tailored for dynamic and time dependent simulations, on the other hand a graphical user interface is already provided.

The user interface is block oriented for convenient usage and creation of coupled models. Blocks are connected to each other providing information flow between each other. Besides using pre-existing blocks provided by Simulink the creation of own blocks is supported. Creation of own routines is done by coding in either m-functions, s-function or C++. For simulation either continuous or sampled (discrete) time may be used. Results can be visualized directly in Simulink. Alternatively results may be stored in the Matlab workspace for visualisation or further analysis.

2 FIRST STEPS

2.1 Installation of City Drain

City Drain requires two simple steps prior being available within Matlab/Simulink environment. Following files are part of the City Drain software:

CityDrain01.zip
CD1_startup.m
CD1_A_UserManual.pdf
CD1_B_Tutorial.pdf

The file (*CityDrain01.zip*) contains the software library and all associated. Data is to be unzipped and saved preferably in the operating system's programs directory.

C:\Programme\CityDrain01	for German operating system
C:\Program Files\CityDrain01	for English operating system

All functions of City Drain are provided with the prefix *"CD1_"* to avoid conflicts with other Matlab libraries or functions used. For convenient use of City Drain 1.0 it is required to include the *CityDrain01* directory (and all subdirectories) in the Matlab paths. Therefore the Matlab "startup.m" file is extended for automatic adding of City Drain directory to the Matlab path.

File:
C:\Programme\MATLAB6p5\work\startup.m

In case there is no startup.m file created in your Matlab, please create a new startup-file. Following code to be added can be found in *CD1_startup.m.* The user may modify the path of City Drain included in the code (bold printed).

```
% Path setting for CITY DRAIN 1.0
% IUT Institute of Environmental Engineering

cd01path='C:\Program Files\CityDrain01';
cd01path_full=genpath(cd01path);
k=strcmp(cd01path_full,'');
disp('Matlab-path for CITY DRAIN 1.0:');

if k==1
     disp('HAS NOT BEEN SET !!');
     disp('Please check in startup.m if path is set correctly.');
     disp(' ');disp(' ');

else
     disp(cd01path);
     path(path,cd01path_full);
     disp(' ');disp(' ');
end

clear('cd01path');clear('cd01path_full');clear('k');
```

2.2 The City Drain Library

To open the City Drain block library type

```
> citydrain
```

in the Matlab command window. Alternative, the Library can be opened via *"File/Open…"* :

```
C:\Programs\CityDrain01\CD1_CityDrain_Library.mdl
```

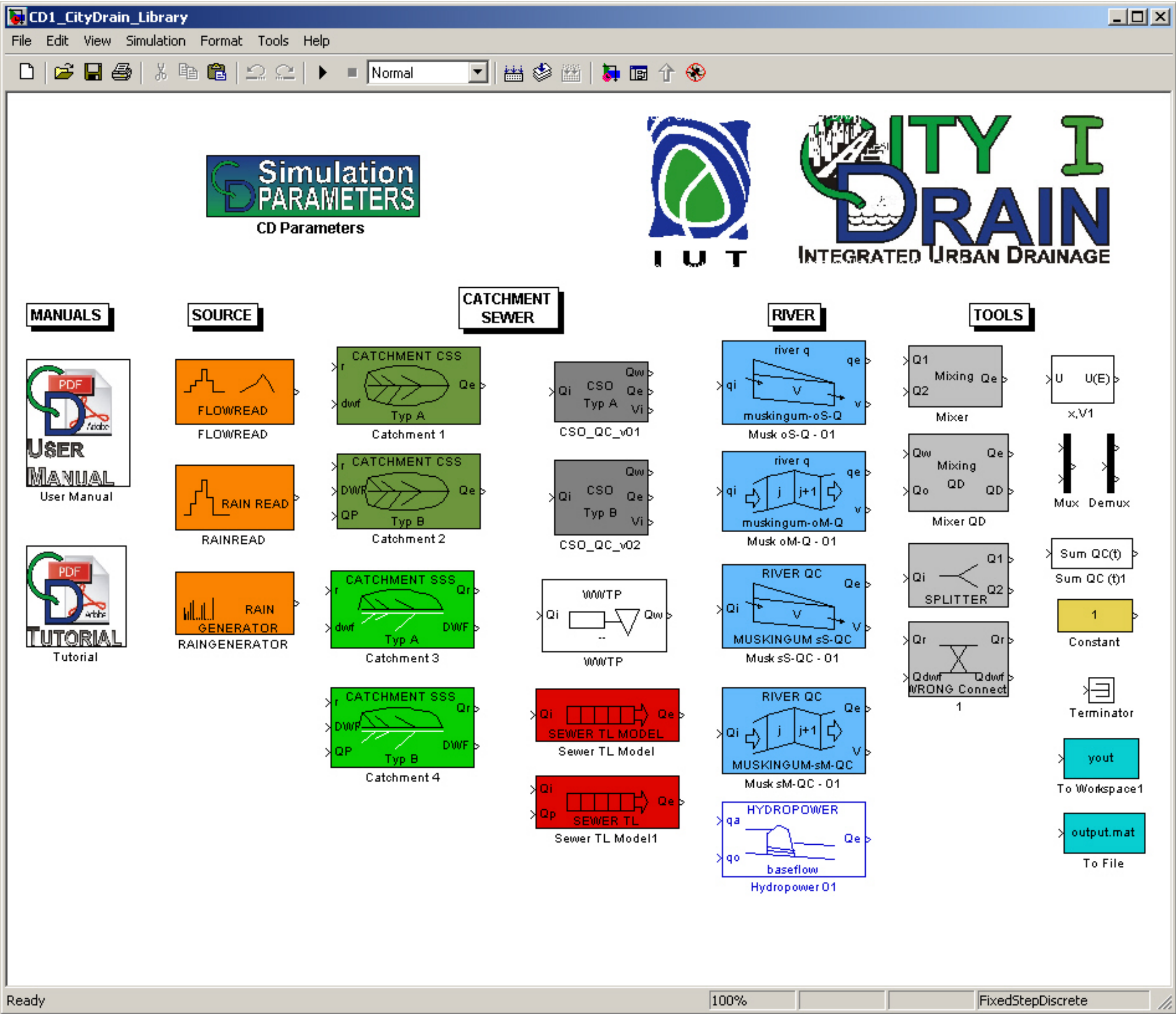


Fig. 2. City Drain 1.0 Block Library (*CD1_CityDrain_Library.mdl*)

The library contains blocks in 5 sections. Core block required for every simulation is the "CD Parameters" blocks organizing global setting for each simulation.

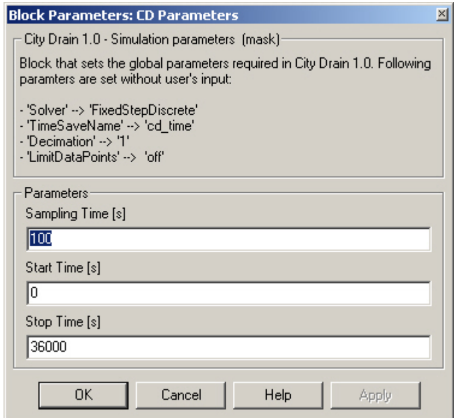
This manual as well as the Tutorial may be opened via double click on the "Manual Blocks". The remaining blocks represent different parts of the urban drainage system and are described in detail in this manual. How to create a new scenario, perform simulations and cope with simulation results is shown in the Tutorial Manual.

3 SIMULATION PARAMETERS AND UNIT CONVENTIONS

CITY DRAIN and the library blocks implemented are designed to work within a discrete time scheme. Constant and discrete time steps are used within a simulation where simulation time and size of time steps are to be chosen by the used.

Core element of every CITY DRAIN simulation is the block **"CD - Simulation Parameters"**

CD Parameters



This block ensures that simulation parameters in the Matlab/Simulink © are defined correctly. User input is required for the

- sampling time Δt,
- start time t_0 and
- stop time t_E.

of the simulation. The sampling time defined is utilized within all CITY DRAIN blocks provided, thus is being globally used. Hidden settings (without required user input) are made for

- 'Solver' ....'FixedStepDiscrete'
- 'TimeSaveName'........'cd_time'
- 'Decimation' ....'1'
- 'LimitDataPoints' ....'off'

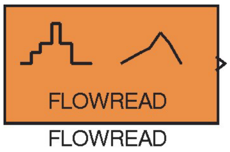
THE BLOCK "CD – SIMULATION PARAMETERS" IS TO BE INCLUDED WITHIN EACH SIMULATION TO ENSURE CORRECT SETTING OF VARIABLES.

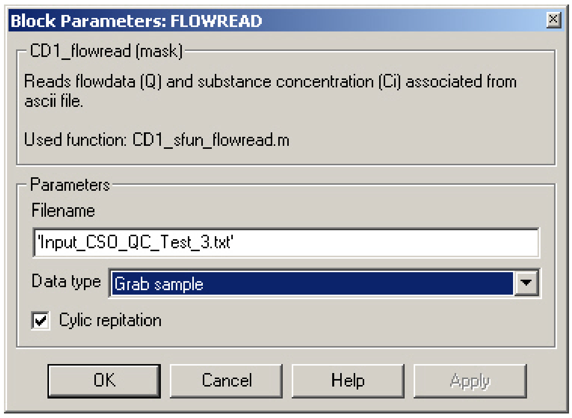
Convention regarding units in City Drain are as followed:

Q [m^3/s]....... Flow [cubic meter per second]
V [m] .... Volume [cubic meter per second]
L [m] .... Length [meters]
t, Δt [s] .... Time [seconds]
C [g/m^3] ...... Concentrations [gram per cubic meter]
M [g] .... Mass [grams]

4 SOURCE BLOCKS

4.1 Flowread





Function:

Reads flow data from ASCII files containing time t, flow Q and concentrations C as input. Data is to be provided column wise. First row in the file allows to hold an alpha-numeric descriptor for column data.
The time is to be provided in [sec] starting with t=0. Sampling time $\Delta t_{CITYDRAIN}$ in the simulation must not necessarily be same as the sampling time given in the raw data. An automatic interpolation of data is provided.
In case the data set ends before the end of simulation, values are set to zero.
Cyclic repetition of data is optionally provided which may be used for e.g. the repetition of daily flow dynamics within long term simulations. Requirement for cyclic repetition is that the first data set Q(t=0), C(t=0) and the last data set $Q(t=t_{max})$, $C(t=t_{max})$ are equal.

Data provided may either represent grab samples (measurement at specific point of time) or composite samples (values representing the mean concentration / flow over time).

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_flowread.m*.
Sampling time used is obtained from the global setting of the sampling time.

Input:

none

Output:

Q	Flow [m³/s].
Ci	Pollutant concentrations [g/m³] The number of pollutant concentrations is automatically inherited from the ASCII file storing the raw data.

Parameters:

Source file containing the hydrograph Q(t) and pollutograph C(t) (optional).

Data type

“Grab sample”….Flows Q and concentrations C given are measured values corresponding to the specific point of time.
“Composite sample”….Flows Q and concentrations C given represent mean values corresponding to the past sampling period.

Initial Conditions:

No initial condition required

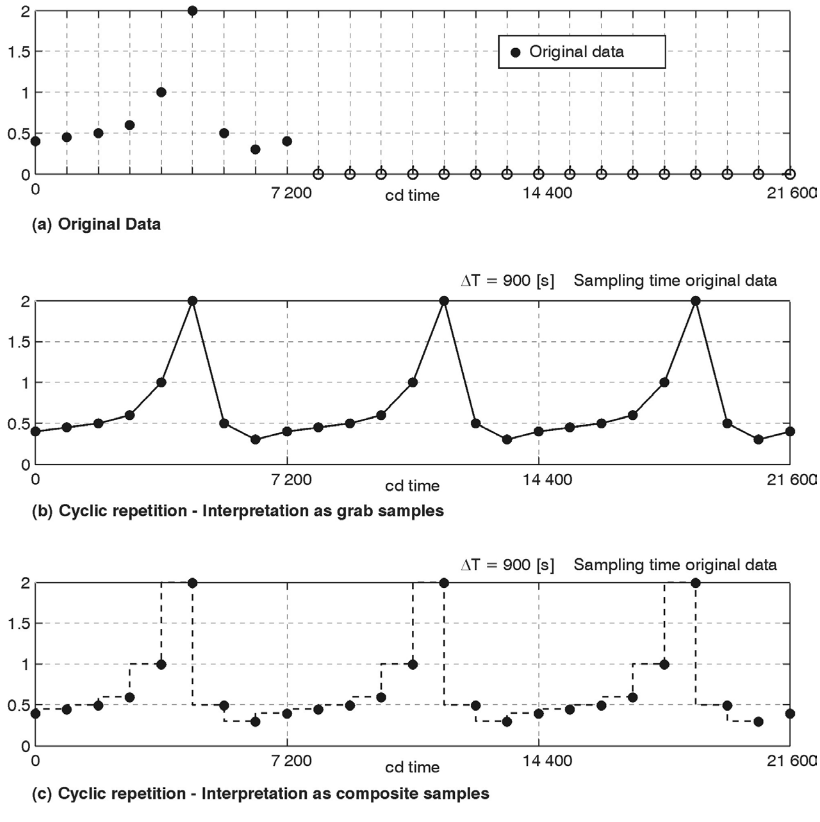
Format / Example Usage:

In the following the output generated by flow read is shown having an example input file containing 15 min values (ΔT= 900 s) of flow and pollutant concentration. Tabulators are used as delimiters.

Example input file (*Input_Flowread_Example.txt*):

```
t       q       C1
0       0.40    0.16
900     0.45    0.20
1800    0.50    0.25
2700    0.60    0.36
3600    1.00    1.00
4500    2.00    4.00
5400    0.50    0.25
6300    0.30    0.09
7200    0.40    0.16
```

The example input is prepared for usage with the cyclic repetition option. There for the first and last entry have to be equal.



(a) Original Data

(b) Cyclic repetition - Interpretation as grab samples

(c) Cyclic repetition - Interpretation as composite samples

Fig. 3. Data read with cyclic repetition having equal sampling time in data (ΔT) and simulation Δt.

Fig. 3 (a) shows a plot of the raw input data having a sampling time of ΔT=900 [s]. Due to cyclic repetition being applied, the last entry of data is substituted as default by the first data entry. The user is requested to provide data used for cyclic repetition having equal data at the first and last entry.

<u>Interpretation of raw data</u>

The raw data read may be interpreted as

- grab samples (Fig. 3(b)) or as
- composite samples (Fig. 3(c))

For grab sample data, the distribution of flow and concentrations is assumed to be linear over ΔT between data points. In the case of composite samples the data read represents mean flows / concentration over the past time step ΔT.

Transfer from raw data to City Drain output data

The output generated from this block is always of type "composite sample" regardless what type of raw data was used. The values represent the mean flow / mean concentration over the last time step $\Delta t_{CITYDRAIN}$.

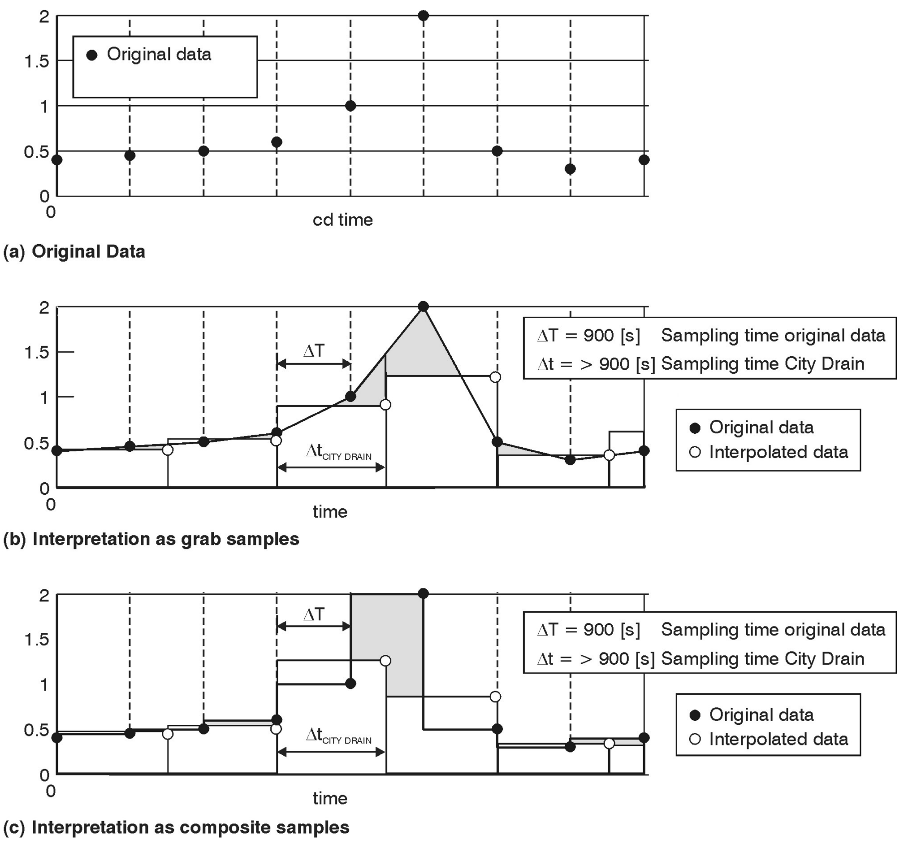


Fig. 4. Interpolation of data when sampling time given in the raw data ΔT and simulation Δt are not equal.

In Fig. 4 (a) a plot of the raw data are shown having time steps ΔT. The example output generated is for sampling times $\Delta t_{CITYDRAIN} > \Delta T$. An internal algorithm is used to account for transferring raw data to output data. Differences in interpolated output is given for raw data being interpreted as either grab sample Fig. 4 (b) or as composite sample Fig. 4 (c) (grey shaded areas).

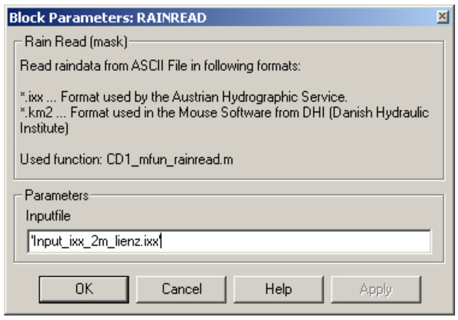
The algorithm is based on the principle of conservation of mass. Volume (V) and mass flux (F) over each time step are integrated and are maintained when transferred to sampling steps used in the simulation ($\Delta t_{CITYDRAIN}$).

$$V = const. = \int q_{RAW}(t)dt = \overline{q} \cdot \Delta t_{CITYDRAIN}$$

$$F = const. = \int q_{RAW}(t) \cdot C_{RAW}(t) \cdot dt = \overline{q} \cdot \overline{C} \cdot \Delta t_{CITYDRAIN}$$

4.2 Rainread

RAINREAD



Function:

Reads rain data from ASCII files having a predefined format. As output rain data is provided following the sampling time of the simulation environment. The dates read are transferred into numerical values of time, where counting of time is started with t=0 at the earliest date/time obtained.
See the formats section for the data types supported.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_rainread.m* .
Sampling time is inherited from the global settings for the current simulation.

Input:

none

Output:

r_R Volume of rain per time step in [mm/ΔT].
tout *Run time is virtually transferred to the Simulink environment*

Parameters:

Inp.File Name of ASCII file (e.g. 'filename.ixx') storing the rain data to be read.

Format *For the type of format no additional user input is required. The appropriate format is defined by the file extension. Following extensions are supported:*

- 'ixx'
- 'km2'
- 'mse'

See section *Formats* for details on the type of rain data formats supported.

Initial Conditions:

No initial condition required

Formats:

Supported formats in *CD1_sfun_rainread.m*

<u>IXX</u> Format used by the Austrian Hydrographic Service. The format uses a line of file for one data entry every 5-minute interval. V_R of time interval t_1 to t_2 is numerically attributed to time t_1 . The format of a line is written as:

DD.MM.YYYY_hh:mm:ss_r_R

with

DD (day), MM (month), YYYY (year)
hh (hour), mm (minute), ss (second)
r_R (Volume of rain [mm])

Example input file:

```
01.01.1991 00:00:00        0.1
01.01.1991 00:05:00        0.1
01.01.1991 00:10:00        0.1
01.01.1991 00:15:00        0.1
01.01.1991 00:20:00        0.1
```

KM2 This format is used as well within the software MOUSE from DHI (Danish Hydraulic Institute). In contrast to the ixx format, where dry rain periods are stored as well, this format produces rather small file sizes.
Rain data is stored as well for discrete time steps but splitted into different rain events. Rain events are defined by the dry (rainless) period in between. By default the dry period separating two rain events is taken as 1 hour which is roughly the time for a drainage system to empty. Thus, two consecutive events do not interact hydraulically in the system.

Each rain event is always introduced by a header and then followed by the rain data itself.

Syntax of the header:

3 YYYYMMDD hhmm 0 N t V.V

3	Internal Code
YYYYMMDD	Date
hhmm	Starting time in hours and minutes
0	Internal code
N	Number of intervals recorded in the event [-]
t	Timestep of one interval [min]
V.V	Total rain volume cumulated in the rain event [mm]

Rain data:
The rain data is stored row wise carrying 10 entries in a row. One line is started by two space. Rain data R is written with one integer digit and three decimal digits stored as rain rate having the unit [10^{-3}mm/s]. Conversion from rain rate R to rain volume r per time step is done by

$$r[mm/\Delta t] = R[10^{-3} mm/s] \cdot 0.001 \cdot \Delta t[s] .$$

Example input file:

```
3 19900425 0820 0 1  5   0.1
  0.333
3 19900426 0955 0 18  5   0.4
  0.250 0.000 0.000 0.000 0.000 0.000 0.000 0.000 0.250 0.000
  0.250 0.000 0.000 0.150 0.100 0.000 0.248 0.002
3 19900426 1320 0 8  5   0.2
  0.150 0.100 0.000 0.000 0.000 0.000 0.000 0.250
3 19900426 1915 0 2  5   0.1
  0.228 0.022
3 19900427 0925 0 11  5   0.6
  0.293 0.002 0.000 0.000 0.293 0.272 0.319 0.298 0.293 0.002
  0.295
```

Since the output generated from the block is a continuous stream of rain data, timely gaps in between the rain events are filled with zeros for the rain volume.

MSE MSE formatted rain data only stores rain events, neglecting dry periods. When reading the data timely gaps are filled by zero values for dry the periods.

The format uses a line per data entry having either 5 or 10 minute intervals. r_R of time interval t_1 to t_2 is numerically attributed to time t_1 . The format of a line is written as:

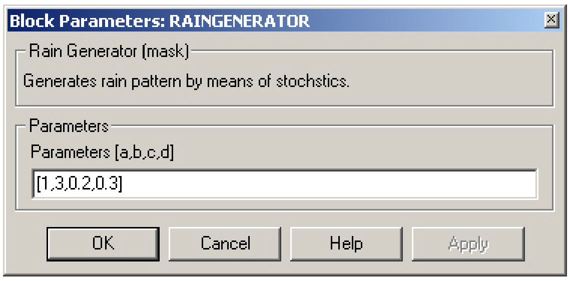
YY_MM_DD_hh_mm_ss_r_R

with
DD (day), MM (month), YY (year)
hh (hour), mm (minute), ss (second)
r_R (Volume of rain [mm])

Example input file:

```
81  1  1 22 40  0  0.0000
81  1  1 22 50  0  0.0000
81  1  1 23  0  0  0.1670
81  1  2  4 30  0  0.1670
81  1  2  4 40  0  0.0000
81  1  2  4 50  0  0.0000
81  1  2  5  0  0  0.0000
```

4.3 Raingenerator



Function:

Generates rain data by means of a simple stochastic algorithm.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_raingenerator.m*.
Sampling time is inherited from the global settings for the current simulation.

Input:

none

Output:

r_R	Volume of rain per time step in [mm].
tout	*Run time is virtually transferred to the Simulink environment*

Parameters:

a, b, c, d Parameters used for calibration of stochastic processes

Initial Conditions:

No initial conditions required

Theory:

The rain series produced is based on a simple stochastic algorithm for generating the main parameters describing a rain series:

T_{DRY}	Duration of next dry period
T_{RAIN}	Duration of next rain event
r_M	Mean rain volume for the next rain event [mm/Δt]

Within a discrete formulation, the durations T_{DRY} and T_{RAIN} are for sake of simplicity given as the number of time steps. At the end of the last rain event (t_0) the durations and the mean rain volume is calculated as:

$$T_{DRY} = -\frac{1}{a} \cdot \log(\theta_a) \cdot \frac{86400}{\Delta t}$$

$$T_{RAIN} = -\frac{1}{b} \cdot \log(\theta_b) \cdot \frac{86400}{\Delta t}$$

$$r_M = -\frac{1}{c} \cdot \log(\theta_c) \cdot \frac{1}{T_{RAIN}}$$

The parameters a, b and c are used for linear scaling of random numbers generated, thus may be used for calibrating the scheme to local real rain series. The log scaled parameters θ_a, θ_b and θ_c are random numbers being uniformly distributed in the interval (0,1).

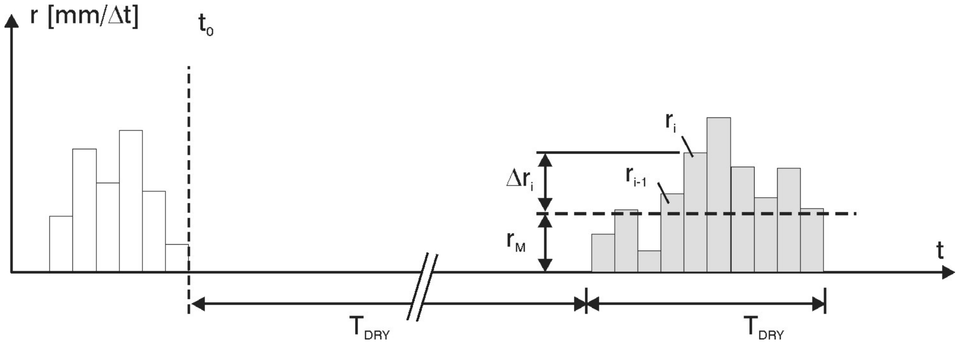


Fig. 5. Schematic for stochastic rain generator

Magnitudes of consecutive rain intensities r_i within a rain event are as well based on a stochastic process. A single rain event is evaluated as deviation Δr_i from the mean rain intensity r_M.

$$r_i = r_M \pm \Delta r_i$$

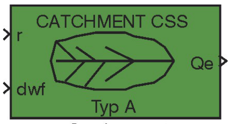
The deviation Δr_i is generated randomly, but using the last deviation Δr_{i-1} generated as an additive term. This is done for including tendencies of increase or decrease and to avoid unnatural jumps in consecutive rain intensities generated.

$$\Delta r_i = \frac{\Delta r_{i-1}}{2} + \Pi \cdot d \cdot r_M$$

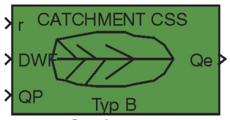
The parameter Π used is a random number being normally distributed with mean 0, variance δ^2=1, and standard deviation δ=1. The constant d is used for linear scaling and required as user input.

5 CATCHMENT BLOCKS

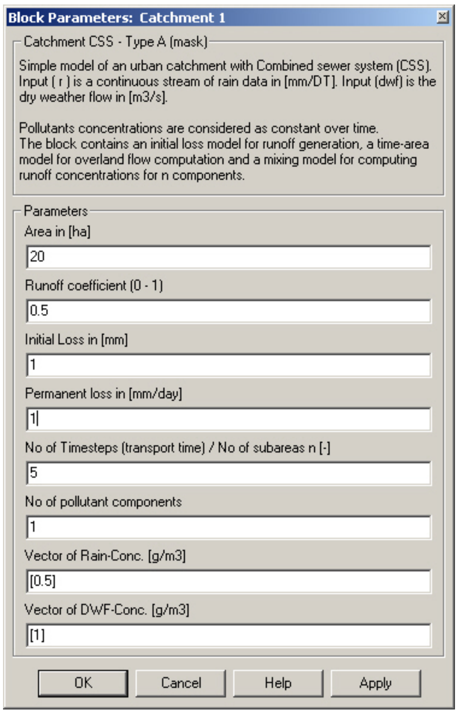
5.1 Catchment CSS (Types A and B)

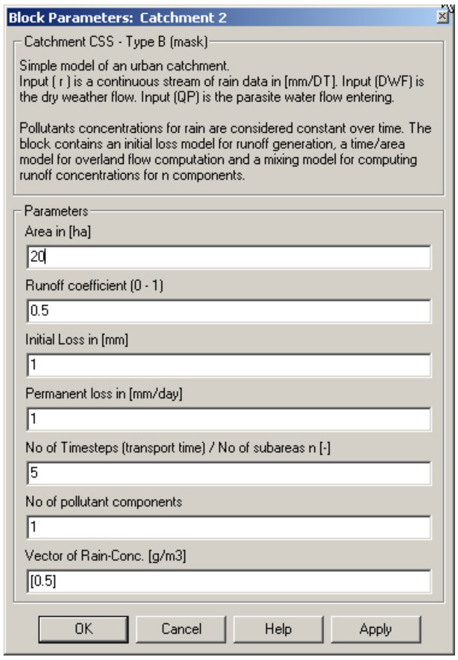


Catchment 1



Catchment 2





Function:

The two catchment CSS blocks (Type A and B) are both designed to simulate a combined sewers system on catchment level. The blocks cope the major drainage-related processes in an urban area and returns for each time step both the current amount and pollutant concentration of the aggregated outflow from the catchment. Storm water runoff and water quality aspects are computed here with a set of consistently simple conceptual models. The main processes that appear on an urban area in this context are

- Runoff generation
- Overland flow
- Dry weather flow generation
- Runoff and wasteflow pollution

In case of Type A, pollutant concentrations for both, rainfall (C_r) and dry weather flow (C_{DWF}), are introduced as constant concentrations using the blocks parameter mask. Rain intensity r_R and dry weather flow rate q_{DWF} are introduces as dynamic inputs. Functioning of Block CSS-Type B is equivalent, except the dynamic input of DWF expecting the full vector of flow and pollutant concentrations associated. Next to dry weather flow (Q_{DWF}) the block accepts an additional dynamic flow input (Q_P). Q_P may be used to simulate parasite water or any other dynamic flow entering the catchment from upstream.

Block Types:

Masked blocks utilizing following underlying S-functions:

S-function	Type A	Type B
CD1_sfun_catchment_lossmodel.m	X	X
CD1_sfun_catchment_tamodel.m	X	X
CD1_sfun_catchment_flowmodel_SW.m	X	X
CD1_sfun_catchment_flowmodel_DWF.m	X	-
CD1_sfun_sewer_tlmodel.m	X	X
CD1_sfun_mixing_QC.m	X	X

Input:

r_R Rain volume per time step [mm/ Δt]. Provided by source block *CD1_rainread*.

Type A only:

q_{DWF} Dynamic flow rate q of dry weather flow [m³/s]

Type B only:

Q_{DWF} Dynamic dry weather flow [q [m³/s] C_1[g/m³] C_2[g/m³] ….]
Q_P Dynamic flow parasite water [q [m³/s] C_1[g/m³] C_2[g/m³] ….]

Output:

Q_e Combined sewer flow at the catchment outlet [q [m³/s] C_1[g/m³] C_2[g/m³] ….].

Parameters:

Types A and B

	Description	used in function
A	Catchment Area A [ha]	CD1_sfun_catchment_flowmodel_SW.m
φ	Runoff coefficient φ [-]. In the range of φ = 0...1.	CD1_sfun_catchment_lossmodel.m
hi	Initial loss [mm]	CD1_sfun_catchment_lossmodel.m
hp	Permanent loss [mm/ day]	CD1_sfun_catchment_lossmodel.m
n_{TA}	Number of sub areas n_{TA} [-]	CD1_sfun_catchment_tamodel.m CD1_sfun_sewer_tlmodel.m
n_P	Number of pollutants n_P [-]	CD1_sfun_catchment_flowmodel_SW.m CD1_sfun_catchment_flowmodel_DWF.m CD1_sfun_sewer_tlmodel.m CD1_sfun_mixing_QC.m
C_{RAIN}	Vector of pollutant concentrations of storm water flow [g/m³]	CD1_sfun_catchment_flowmodel_SW.m

Type A only

	Description	used in function
C_{DWF}	Vector of pollutant concentrations of dry weather flow [g/m³]	CD1_sfun_catchment_flowmodel_DWF.m

Initial Conditions:

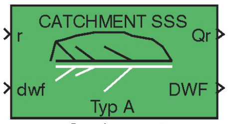
No initial conditions applied

Theory:

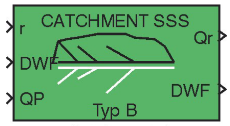
For details of the utilized blocks/s-functions see chapters

5.3.1 Catchment Loss Model
5.3.2 Catchment Time-Area Model
5.3.3 Catchment Flow Model - SW (storm water)
5.3.4 Catchment Flow Model - DWF (dry weather flow)
5.4 Sewer TL Model
7.1 Mixing

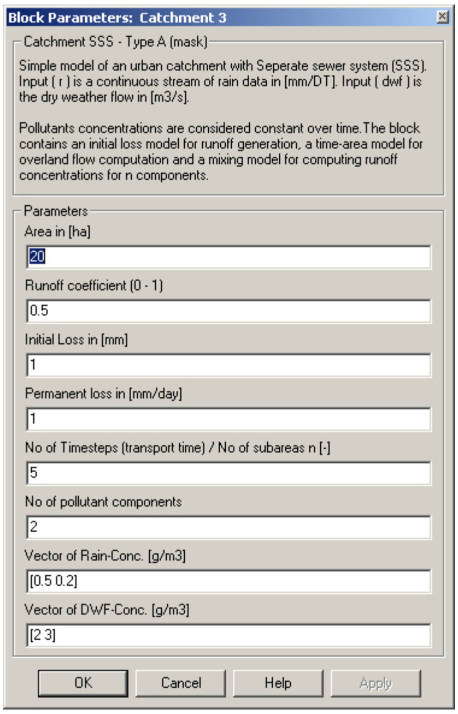
5.2 Catchment SSS (Types A and B)

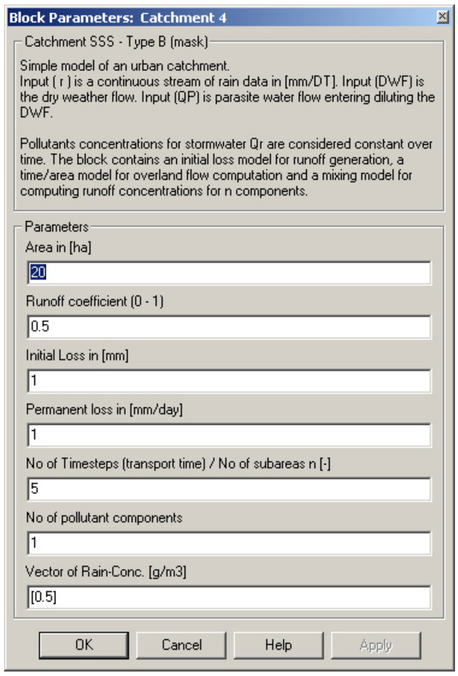


Catchment 3



Catchment 4





Function:

The two catchment SSS blocks (Type A and B) are both designed to simulate a separate sewers system on catchment level. The blocks cope the major drainage-related processes in an urban area and return for each time step both the current aggregated outflow from the catchment and associated pollutant concentration. Storm water runoff and water quality aspects are computed here with a set of consistently simple conceptual models. The main processes that appear on an urban area in this context are

- Runoff generation
- Overland flow
- Dry weather flow generation
- Runoff and wasteflow pollution

In case of Type A, pollutant concentrations for both, rainfall (C_r) and dry weather flow (C_{DWF}), are introduced as constant concentrations using the blocks parameter mask. Rain intensity r_R and dry weather flow rate q_{DWF} are introduces as dynamic inputs. Functioning of block SSS-Type B is equivalent, except the dynamic input for DWF expecting the full vector of flow and pollutant concentrations associated. Next to dry weather flow (Q_{DWF}) the block accepts an additional dynamic flow input (Q_P). Q_P may be used to simulate parasite water or any other dynamic flow entering the catchment from upstream.

Block Types:

Masked blocks utilizing following underlying S-functions:

S-function	Type A	Type B
CD1_sfun_catchment_lossmodel.m	X	X
CD1_sfun_catchment_tamodel.m	X	X
CD1_sfun_catchment_flowmodel_SW.m	X	X
CD1_sfun_catchment_flowmodel_DWF.m	X	-
CD1_sfun_sewer_tlmodel.m	X	X
CD1_sfun_mixing_QC.m	-	X

Input:

r_R Rain volume per time step [mm/ Δt]. Provided by source block *CD1_rainread*.

Type A only:

q_{DWF} Dynamic flow rate q of dry weather flow [m³/s]

Type B only:

Q_{DWF} Dynamic dry weather flow [q [m³/s] C_1[g/m³] C_2[g/m³] ….]

Q_P Dynamic flow parasite water [q [m³/s] C_1[g/m³] C_2[g/m³] ….]

Output:

Q_r Storm water flow at the catchment outlet [q [m³/s] C_1[g/m³] C_2[g/m³] ….].

Q_{DWF} Waste water flow (DWF) at the catchment outlet [q [m³/s] C_1[g/m³] C_2[g/m³] ….].

Parameters:

Types A and B

	Description	used in function
A	Catchment Area A [ha]	CD1_sfun_catchment_flowmodel_SW.m
φ	Runoff coefficient φ [-]. In the range of φ = 0...1.	CD1_sfun_catchment_lossmodel.m
hi	Initial loss [mm]	CD1_sfun_catchment_lossmodel.m
hp	Permanent loss [mm/ day]	CD1_sfun_catchment_lossmodel.m
n_{TA}	Number of sub areas n_{TA} [-]	CD1_sfun_catchment_tamodel.m CD1_sfun_sewer_tlmodel.m
n_P	Number of pollutants n_P [-]	CD1_sfun_catchment_flowmodel_SW.m CD1_sfun_catchment_flowmodel_DWF.m CD1_sfun_sewer_tlmodel.m CD1_sfun_mixing_QC.m
C_{RAIN}	Vector of pollutant concentrations of storm water flow [g/m³]	CD1_sfun_catchment_flowmodel_SW.m

Type A only

	Description	used in function
C_{DWF}	Vector of pollutant concentrations of dry weather flow [g/m³]	CD1_sfun_catchment_flowmodel_DWF.m

Initial Conditions:

No initial conditions applied

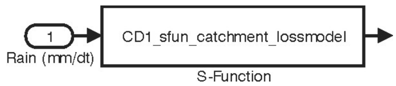
Theory:

For details of the utilized blocks/s-functions see chapters

5.3.1 Catchment Loss Model
5.3.2 Catchment Time-Area Model
5.3.3 Catchment Flow Model - SW (storm water)
5.3.4 Catchment Flow Model - DWF (dry weather flow)
5.4 Sewer TL Model
7.1 Mixing

5.3 Catchment – Underlying Blocks (CSS and SSS)

5.3.1 Catchment Loss Model



Function:

Applies initial and permanent loss to a given precipitation, responsible for the generation of effective runoff height.

Block Type:

Discrete Block utilizing S-function (*CD1_sfun_catchment_lossmodel.m*)

Input:

r_R Rain volume per time step [mm/ Δt]. Provided by e.g. source block *CD1_rainread*.

Output:

h_e Effective runoff per time step [mm/ Δt].

Parameters:

S function parameters [hi , hp , φ]

hi Initial loss [mm]
hp Permanent loss [mm/ Δt]
φ Runoff coefficient φ [-]. In the range of φ = 0...1.

Initial Conditions:

No initial conditions to be applied

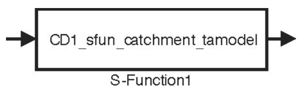
Theory:

The underlying model is a virtual basin having the volume (height) of h_I. The initial loss is therefore represented by a filling of the basin, where the effective runoff h_e is computed as spilled volume per time step Δt .

$$h_e = \left(r_R - h_I\right) \cdot \varphi \geq 0$$

Permanent loss per time step h_P is taken into account only during dry weather periods, used for emptying the virtual basin.

5.3.2 Catchment Time-Area Model



Function:

Applies the time area method (flood routing method) on a catchment scale.

Block Type:

Discrete Block utilizing S-function (*CD1_sfun_catchment_tamodel.m*)

Input:

h_e Effective runoff height per time step [mm/ Δt].

Output:

$h_{e,OUT}$ Effective runoff per time step [mm/ Δt] at the catchment outlet.

Parameters:

S function parameters [n_{TA}]

n_{TA} Number of sub-areas applied in the model [-].

Initial Conditions:

No initial conditions to be applied

Theory:

The applied time area model is simplified with respect to the areas considered. In the general formulation of the time area method, areas of different sizes can be used, where herein areas are considered equally distributed. The number of sub-areas (n_{TA}) is kept variable in order to account for the longitudinal expansion of the catchment. The number of sub-areas further represent the flow time T_F of storm water in relation of the time steps used in the simulation (Δt):

$$T_F = n_{TA} \cdot \Delta t$$

The dynamic output of effective runoff height $h_{i,e,OUT}$ [mm/Δt] is calculated relating to the total catchment area A_{TOT}. For a time step i the contributing rain to the actual runoff would be

$$h_{i,e,OUT} = \begin{bmatrix} r_i & r_{i-1} & \cdots & r_{i-(j-1)} \end{bmatrix} \times \begin{bmatrix} 1/n_{TA} \\ 1/n_{TA} \\ \vdots \\ 1/n_{TA} \end{bmatrix} = \sum_{j=1}^{n} \frac{1}{n_{TA}} \cdot r_{i-(j-1)}$$

with

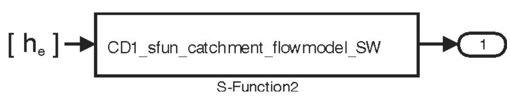
n_{TA} Number of sub catchments

j Control Variable running from j=1...n_{TA}

The actual runoff rate $q_{e,OUT}$ [m³/s] is calculated in block *CD1_catchment_flowmodel_SW.m*.

$$q_{e,OUT} = h_{e,OUT} \cdot A_{TOT}$$

5.3.3 Catchment Flow Model - SW (storm water)



Function:

Block for generating outflow Q_e from a given effective runoff height h_e [mm / Δt]. Pollutant concentrations – given as constant parameters - are added to the flow.

Block Type:

Discrete Block utilizing S-function (*CD1_sfun_catchment_flowmodel_SW.m*)

Input:

h_e Effective runoff height per time step [mm/ Δt] at the catchment outlet.

Output:

Q_e Stormwater runoff in [m³/s] at the catchment outlet, including concentrations of substances carried. Concentrations applied are introduced as constant parameter C_{RAIN}.

Parameters:

S function parameters [A, $n_{P'}$, C_{RAIN}, t_{SAMP}]

A	Catchment Area [m²].
n_P	Number of pollutants carried n_P [-]
C_{RAIN}	Vector of pollutant concentrations of storm water flow [g/m³]
Δt	Sampling rate (time steps) in [s]. Δt is inherited from the global setting of sampling times.

Initial Conditions:

No initial conditions to be applied

Theory:

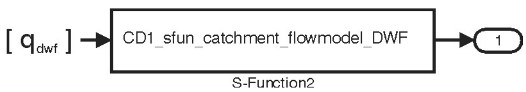
The S-function is responsible for creating a flow vector representing the storm water flow at the catchment outlet only. The function covers the transfer from dynamic rain height h_e in terms of [mm/Δt] to flow rate q_e [m³/s] using the catchment area A. The resulting outflow rate from the catchment is calculated as

$$q_e[m^3/s] = \frac{h_e[mm/\Delta t]}{1000 \cdot \Delta t[s]} \cdot A[m^2]$$

The number of substances carried n_P must meet the length of the vector holding the pollutant concentrations (C_{RAIN}). The resulting output vector Q_e has the form of

$$Q_e = \begin{bmatrix} q_e & C_{RAIN,1} & \cdots & C_{RAIN,K} & \cdots & C_{RAIN,n_P} \end{bmatrix}$$

5.3.4 *Catchment Flow Model - DWF (dry weather flow)*



Function:

Block for generating a vector of waste water flow Q_{DWF} using a dynamic flow rate q_{dwf} [m³/s] and constant pollutant concentrations given as constant parameter C_{DWF}.

Block Type:

Discrete Block utilizing S-function (*CD1_sfun_catchment_flowmodel_DWF.m*)

Input:

q_{DWF} Flow rate of dry weather flow [m³/s] at the catchment outlet.

Output:

Q_{DWF} Dry weather flow at the catchment outlet, including concentrations of substances carried.

Parameters:

S function parameters: [$n_{P'}$, C_{DWF}, Δt]

n_P	Number of pollutants n_P [-]
C_{DWF}	Vector of pollutant concentrations of storm water flow [g/m³]

Δt Sampling rate (time steps) in [s]. Δt is inherited from the global setting of sampling times.

Initial Conditions:

No initial conditions to be applied

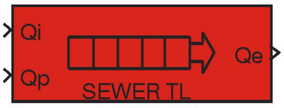
Theory:

The S-function is responsible for creating a flow vector representing the dry water flow at the catchment outlet. Dynamic storm water flow rate q_{dwf} [m³/s] is combined with constant concentrations C_{DWF}. The resulting output vector Q_{DWF} has the form of

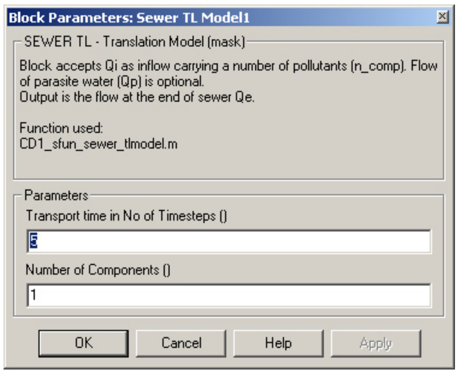
$$Q_{DWF} = \begin{bmatrix} q_{dwf} & C_{DWF,1} & \cdots & C_{DWF,K} & \cdots & C_{DWF,n_P} \end{bmatrix}$$

The number of substances carried n_P must meet the length of the vector holding the pollutant concentrations (C_{DWF}).

5.4 Sewer TL Model



Sewer TL Model1



Function:

Block for flow / pollutant routing in a sewer by means of simple translation of water and matter. The block may be used for sewer stretches not subjected to surface runoff. Upstream inflow is required as dynamic input Q_i. Optional parasite water Q_P infiltrating along the sewer stretch may be added as dynamic input.

Block Type:

Discrete Block utilizing S-function (*CD1_sfun_sewer_tlmodel.m*)

Input:

Q_i Inflow to be routed in the sewer.
Q_P Parasite water inflow to be routed in the sewer (optional).

Output:

Q_e Effluent flow at the sewer outlet.

Parameters:

S function parameters: [$n_{P'}$, C_{DWF}, Δt]

n_P Number of pollutants n_P [-]
n_T Transport time defined as number of time steps.
Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

Initial Conditions:

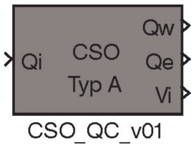
No initial conditions to be applied.

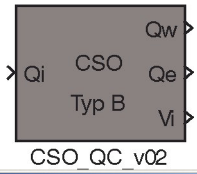
Theory:

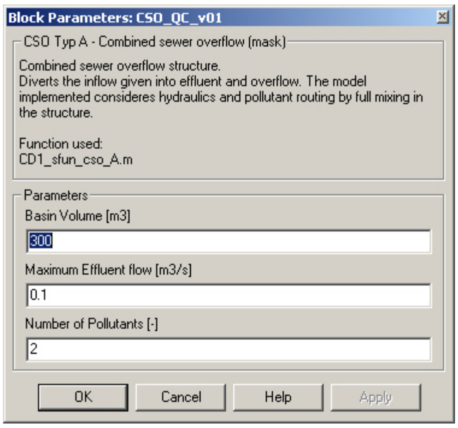
Routing is done by means of simple translation of water and matter. The flow time T_F in the sewer is given by the number of time steps n_T.

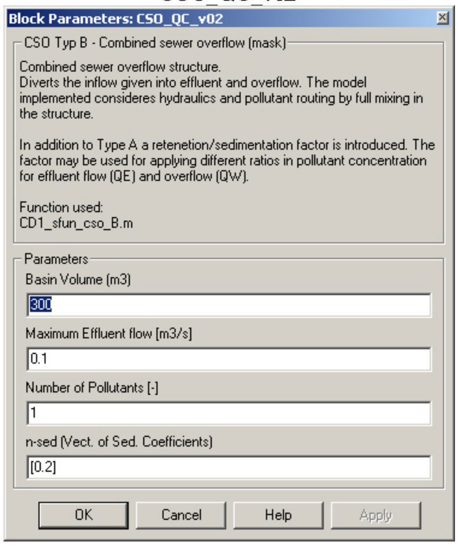
$$n_T \cong T_F / \Delta t$$

5.5 CSO (Types A and B)









Function:

Simulation of an overflow structure for either combined or separate sewer. Inflow is routed downstream via the effluent outlet (Q_E) which is limited by a maximum effluent flow rate $Q_{E,MAX}$ [m³/s]. Flow exceeding the structures storage capacity (V_{MAX}) is routed via the CSO overflow (Q_W [m³/s]). With the model implemented hydraulic and pollutants routing is based on instant and ideal mixing in the CSO structure.

CSO Type B contains, in contrast to Type A, a simplified modelling option of sedimentation. Using a linear sedimentation coefficient η_{sed} in CSO type B allows modelling of lower pollutant concentrations present in the overflow (Q_W) than in the effluent flow (Q_W).

Block Type:

Discrete Block utilizing S-functions:

CD1_sfun_cso_A.m
CD1_sfun_cso_B.m

Input:

Q_i Inflow to CSO structure.

Output:

Q_e Effluent flow at the CSO structure.
Q_w Overflow generated when CSO storage volume is exceeded.

Parameters:

S function parameters:

CSO Type A [V_{MAX}, $Q_{e,MAX}$, $n_{P'}$, Δt]
CSO Type B [V_{MAX}, $Q_{e,MAX}$, $n_{P'}$, η_{sed}, Δt]

V_{MAX} Basin volume (storage volume) [m³]

$Q_{E,MAX}$ Maximum Effluent flow [m³/s]
n_P Number of pollutants carried [-].
η_{sed} Sedimentation coefficient
Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

Initial Conditions:

No initial conditions to be applied. The structure is considered being empty at the beginning of the simulations.

Theory:

The model is based on the discrete formulation of mass balance equation.

$$\frac{V_i - V_{i-1}}{\Delta t} = Q_{I,i} - Q_{E,i} - Q_{W,i}$$

Boundary conditions introduced are the storage volume of the structure $V_{E,MAX}$ (maximum Volume) and the maximum excess flow $Q_{E,MAX}$ routed downstream.
Pollutants routing is based on instant and full mixing of inflow (Qi,Ci) and mater in the Tank (V,C). Concentrations in the outflows (Q_w, C_{Qw} and Q_e, C_{Qe}) are equal to concentrations given in the tank.
For CSO Type B, settling of matter is enabled where the concentrations in the overflow (Cw) are reduced using a sedimentation (settling) coefficient η_{SED}.

$$C_{QW,i} = \frac{C_{QI,i} \cdot Q_{I,i} \cdot \Delta t + C_{V,i-1} \cdot V_{i-1}}{Q_{I,i} \cdot \Delta t + V_{i-1}} \cdot \left(1 - \eta_{SED}\right)$$

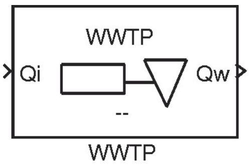
Concentrations given in the effluent flow Q_E and in the CSO structure are consider equal. According to the mass balance of matter the effluent concentrations denote therefore as:

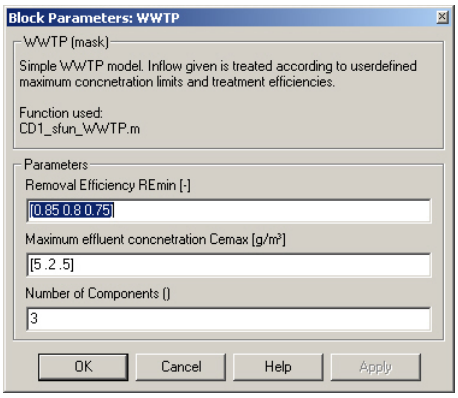
$$C_{V,i} = \frac{C_{QI,i} \cdot Q_{I,i} \cdot \Delta t + C_{V,i-1} \cdot V_{i-1}}{Q_{E,i} \cdot \Delta t + V_i} \cdot \left(1 - \frac{Q_{W,i} \cdot \Delta t}{Q_{I,i} \cdot \Delta t + V_{i-1}} \cdot (1 - \eta_{SED})\right)$$

$$C_{QE,i} = C_{V,i}$$

In CSO Type A no sedimentation is considered. This equals a sedimentation coefficient $\eta_{SED} = 0$.
Index i in all equations represents the time step.

5.6 WWTP





Function:

Models a WWTP that is consideration to fulfill predefined requirements of

- removal efficiencies $R_{E,MIN}$ [-] and
- maximum effluent concentrations $C_{E,MAX}$ [g/m³].

Thus emission standards are considered to be fulfilled, regardless the hydraulic or pollutant load associated. In the model no process are considered.

Block Type:

Discrete Block utilizing S-function (*CD1_sfun_WWTP.m*)

Input:

Q_i Inflow to WWTP.

Output:

Q_w Outflow from the WWTP.

Parameters:

S function parameters: [$n_{P'}$, $C_{E,MAX}$, $R_{E,MIN}$]

$R_{E,MIN}$ Required removal efficiencies [-]
$C_{E,MAX}$ Maximum effluent concentration [g/m³]
n_P Number of pollutants carried [-].

Initial Conditions:

No initial conditions to be applied.

Theory:

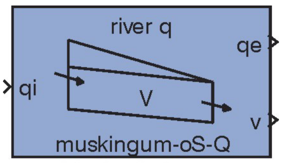
Based on user defined removal efficiencies $R_{E,MIN}$ [-] and maximum effluent concentrations $C_{E,MAX}$ [g/m³], concentrations of the inflow are reduced and assigned to the effluent flow. For each substance the more stringent of the defined prerequisites is applied.

$$C_{W,i} = \min\left[C_{E,i,MAX} \quad ; \quad C_{I,i} \cdot R_{E,MIN}\right]$$

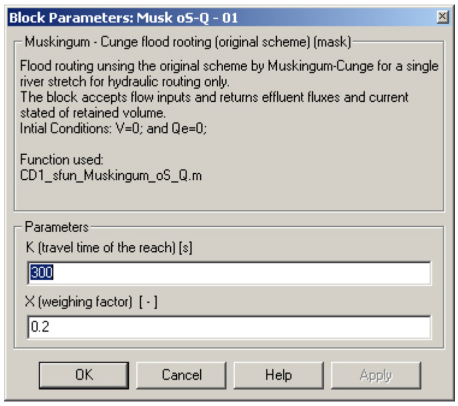
Thus, it is assumed that the WWTP meets the prerequisites defined, regardless the magnitude of either hydraulic or pollutant loading applied.

6 RIVER (FLOOD ROUTING) BLOCKS

6.1 Muskingum oS - Q



Musk oS-Q - 01



Function:

Muskingum oS-Q describes the flood routing by the Muskingum Method (Roberson *et al.*, 1995) for a single stretch. **The block considers hydraulics only**, where contaminant transport is not included.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_Muskingum_oS_Q.m*

Input:

q_I Upstream inflow in the stretch in [m³/s].

Output:

q_E Downstream outflow from the stretch in [m³/s].
v Current Volume stored in the reach [m³].

Parameters:

K Muskingum parameter describing the time required for a discharge wave travelling through the reach [s].
X Dimensionless weighting factor that relates to the amount of wedge storage [-]. Usually in the range of 0 (linear reservoir storage) and 0.5. (Typical value = 0,2).

S-function parameters: [Δt, K, X, C_0,C_1,C_2]

C_0,C_1,C_2 Muskingum constants calculated within the block mask (no user input required). See theory section below.
Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

Initial Conditions:

$Q_E = 0$; $V = 0$; $Q_I = 0$;

Theory:

The outflow from a reach is calculated as a function of $Q_{E,I}$, *$Q_{I,i}$ and* $Q_{E,I+1}$. Index *i* denotes the time step.

$$Q_{E,i+1} = C_0 \cdot Q_{I,i+1} + C_1 \cdot Q_{I,i} + C_2 \cdot Q_{E,i}$$

Muskingum constants C_0,C_1 and C_2 are constant over time and calculated as followed:

$$C_0 = \frac{0{,}5 \cdot \Delta t - K \cdot X}{K \cdot (1 - X) + 0{,}5 \cdot \Delta t}\;;\quad C_1 = \frac{0{,}5 \cdot \Delta t + K \cdot X}{K \cdot (1 - X) + 0{,}5 \cdot \Delta t}\;;\quad C_2 = \frac{K \cdot (1 - X) - 0{,}5 \cdot \Delta t}{K \cdot (1 - X) + 0{,}5 \cdot \Delta t}$$

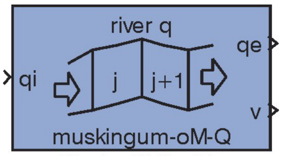
The actual volume stored in the river reach denotes as:

$$V_{i+1} = K \cdot \left[X \cdot Q_{I,i+1} + (1 - X) \cdot Q_{E,i+1}) \right]$$

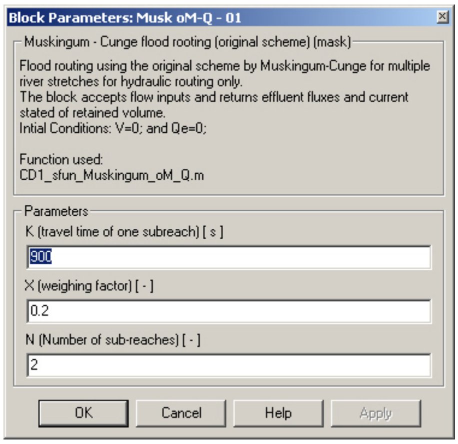
The numerical stability of the Muskingum equation is given when fulfilling the following relation of K, X and Δt:

$$\frac{1}{2 \cdot (1 - X)} \leq \frac{K}{\Delta t} \leq \frac{1}{2 \cdot X}$$

6.2 Muskingum oM - Q



Musk oM-Q - 01



Function:

Muskingum oM-Q describes the flood routing by the Muskingum Method (Roberson *et al.*, 1995) for a river stretch comprised holding multiple subreaches. **The block considers hydraulics only**, where contaminant transport is not included.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_Muskingum_oM_Q.m*

Input:

q_I Upstream inflow in the stretch in [m³/s].

Output:

q_E Downstream outflow from the stretch in [m³/s]. Outflow from the last (N^{th}) sub reach.

v Vector of current volumes stored in sub reaches [m³]. The length of the Vector depends on the number of sub reaches (N) considered.
[V1, V2,… V_{N-1}, V_N]

Parameters:

K Muskingum parameter describing the time required for a discharge wave travelling through the reach [s]. K applies to one subreach and does not cover travelling time for all sub reaches.

X Dimensionless weighting factor that relates to the amount of wedge storage [-]. Usually in the range of 0 (linear reservoir storage) and 0.5. (Typical value = 0,2).

N Number of subreaches

S-function parameters: [Δt, N, K, X, C_0,C_1,C_2]

C_0,C_1,C_2 Muskingum constants calculated within the block mask (no user input required). Constants apply to one subreach !! See theory section below.

Δt *Sampling rate Δt [s] is inherited from global setting of simulation parameters.*

Initial Conditions:

Volumes in and flows rate between subreaches are considered to be zero for the initial conditions.

Theory:

Flows from a sub reaches (j) are driven by the outflow from their upstream sub reach (j-1) where *j* denotes the reach number and *i* denotes the time step.

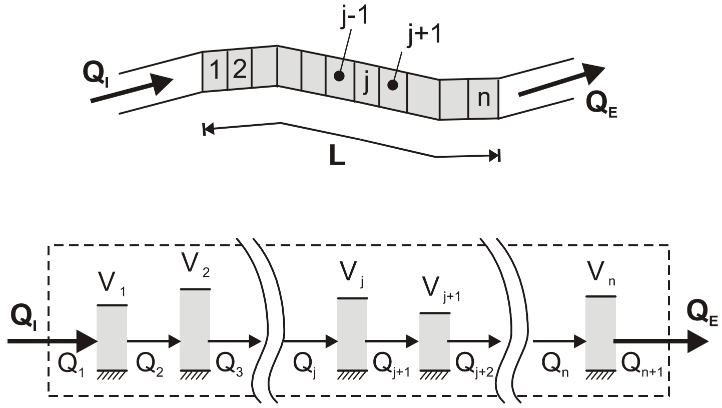


Fig. 6. Schematic on nomenclature for multiple subreaches

$$Q_{i+1}^{j+1} = C_0 \cdot Q_{i+1}^{j} + C_1 \cdot Q_i^j + C_2 \cdot Q_i^{j+1}$$

$$C_0 = \frac{0,5 \cdot \Delta t - K \cdot X}{K \cdot (1-X) + 0,5 \cdot \Delta t}\ ; \quad C_1 = \frac{0,5 \cdot \Delta t + K \cdot X}{K \cdot (1-X) + 0,5 \cdot \Delta t}\ ; \quad C_2 = \frac{K \cdot (1-X) - 0,5 \cdot \Delta t}{K \cdot (1-X) + 0,5 \cdot \Delta t}$$

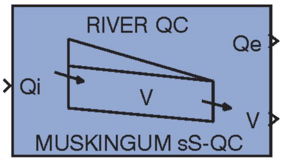
Current Volume stored for a sub reach *j* is calculated as:

$$V_{i+1}^{j} = K \cdot \left[X \cdot Q_{i+1}^{j} + (1-X) \cdot Q_{i+1}^{j+1}) \right]$$

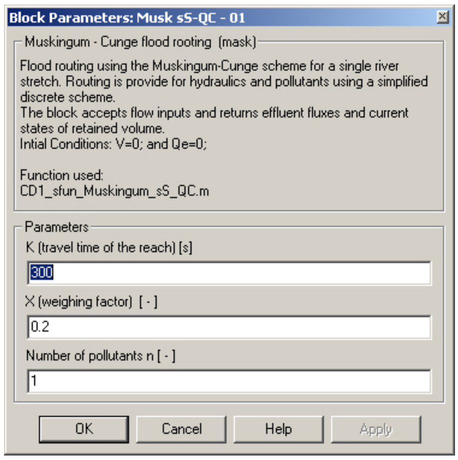
Again, for numerical stability the following relation between K, X and the discrete sampling time Δt must be fulfilled:

$$\frac{1}{2 \cdot (1-X)} \leq \frac{K}{\Delta t} \leq \frac{1}{2 \cdot X}$$

6.3 Muskingum sS - QC



Musk sS-QC - 01



Function:

Muskingum sS-QC describes the flood routing by the Muskingum Method. Formulation is based on a simplified discrete scheme compared to the original scheme. Instead of considering the discharges Q at instant times they are considered as mean discharges over the last discrete period of time. This is feasible when recalling that the measured precipitation represents cumulated (mean) quantities of rainfall for discrete time periods.
The block provides hydraulic routing as well as routing of pollutants.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_Muskingum_sS_QC.m*

Input:

Q_I Upstream inflow in the stretch in [m³/s].

Output:

Q_E Downstream outflow from the stretch. [q_E, $C_{E,1}$, $C_{E,2}$, ….,$C_{E,np}$].
V Vector of current volumes and pollutant concentrations stored in the reach. [V, C_1, C_2,…, C_{np}]

Parameters:

K Muskingum parameter describing the time required for a discharge wave travelling through the reach [s].
X Dimensionless weighting factor that relates to the amount of wedge storage [-]. Usually in the range of 0 (linear reservoir storage) and 0.5. (Typical value = 0,2).
n_P Number of pollutants carried

S-function parameters: [Δt, K, X, C_A,C_B,n_P]

C_A,C_B Muskingum constants calculated within the block mask (no user input required). See theory section below.
Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

Initial Conditions:

Flows and concentrations in the reach are initially considered being zero.

Theory:

The outflow from a reach is calculated as a function of $Q_{I,i}$ *and* V_{i-1}. Index *i* denotes the corresponding time step, where flow rates *Qi* represent mean values of flow of the past time step $\Delta t_i = t_i - t_{i-1}$.

$$Q_{E,i} = \frac{Q_{I,i} \cdot C_A + V_{i-1}}{C_B} \quad \text{with}$$

$$C_A = \frac{\Delta t}{2} - K \cdot X$$

$$C_B = \frac{\Delta t}{2} + K \cdot (1 - X)$$

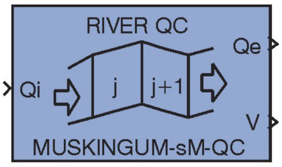
The actual volume stored in the river reach denotes as:

$$V_i = \left(Q_{I,i} - Q_{E,i}\right) \cdot \Delta t + V_{i-1}$$

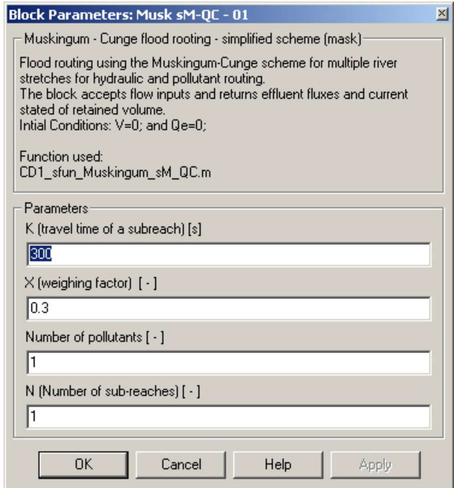
Again, for numerical stability criteria defining the relation between K, X and the discrete sampling time Δt must be fulfilled:

$$1 \le \frac{K}{\Delta t} \le \frac{1}{2X}$$

6.4 Muskingum sM - QC



Musk sM-QC - 01



Function:

Muskingum sM-QC describes the flood routing by the Muskingum Method. Formulation is based on a simplified discrete scheme equivalent to the formulation used in Muskingum sS-QC. Instead of considering the discharges Q at instant times they are considered as mean discharges over the last discrete period of time.

The block provides hydraulic routing as well as routing of pollutants using a user defined number of multiple sub reaches.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_Muskingum_sM_QC.m*

Input:

Q_I Upstream inflow in the river stretch being the flow into the first sub reach. $[q_I, C_{I,1}, C_{I,2}, \ldots, C_{I,np}]$

Output:

Q_E Downstream outflow from the river stretch $[q_E, C_{E,1}, C_{E,2}, \ldots, C_{E,np}]$.

V Vector of current volumes and pollutant concentrations stored in each of the N sub reaches of the river stretch.
$[V^1, C_1^{\,1}, C_2^{\,1}, \ldots, C_{np}^{\,1}, \ldots, V^j, C_1^{\,j}, C_2^{\,j}, \ldots, C_{np}^{\,j}, \ldots, V^N, C_1^{\,N}, C_2^{\,N}, \ldots, C_{np}^{\,N}]$

Parameters:

N Number of subreaches
K Muskingum parameter describing the time required for a discharge wave travelling through the reach [s]. **K applies to one subreach** and does not cover travelling time for all sub reaches.
X Dimensionless weighting factor that relates to the amount of wedge storage [-] in the range of 0 (linear reservoir storage) and 0.5. (Typical value = 0,2).
n_P Number of pollutants carried

S-function parameters: $[\Delta t, K, X, C_A, C_B, n_P, N]$

C_A, C_B Muskingum constants calculated within the block mask (no user input required). See theory section below.
Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

Initial Conditions:

Flows and concentrations in the reach are initially considered being zero.

Theory:

The outflow from a sub reach Q^{j+1} is calculated as a function of the inflow Q^j_i *and stored volume* V^j_{i-1}. Index *i* denotes the corresponding time step, where flow rates *Qi* represent mean values of flow of the past time step $\Delta t_i = t_i - t_{i-1}$. Index j denotes the number of the sub reach.

$$Q_i^{j+1} = \frac{Q_i^j \cdot C_A + V_{i-1}^j}{C_B} \quad \text{with}$$

$$C_A = \frac{\Delta t}{2} - K \cdot X$$

$$C_B = \frac{\Delta t}{2} + K \cdot (1 - X)$$

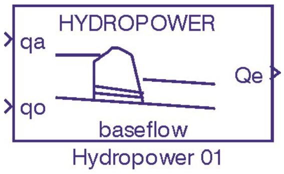
The actual volume stored in a sub reach denotes as

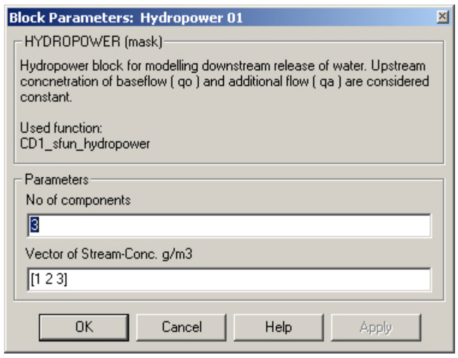
$$V_i^j = \left(Q_i^j - Q_i^{j+1}\right) \cdot \Delta t + V_{i-1}^j$$

Again, for numerical stability criteria defining the relation between K, X and the discrete sampling time Δt must be fulfilled:

$$1 \le \frac{K}{\Delta t} \le \frac{1}{2X}$$

6.5 Hydropower





Function:

The block simulates river discharges from retaining structure such as hydropower intakes. The flow discharged downstream comprises of flow rates for baseflow q_o and additional discharge q_a. Background concentrations associated are introduced as constants in the block mask.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_hydropower.m*

Input:

q_o Flowrate of baseflow [m³/s]
q_a Flowrate of additional flow [m³/s]

Output:

Q_E Outflow from the Hydropower block [q_E, $C_{E,1}$, $C_{E,2}$, ….,$C_{E,np}$].

Parameters:

n_P Number of pollutants carried
C_{STREAM} Constant pollutant background concetration [C_1, C_2, ….,C_{np}].

S-function parameters: [n_P, C_{STREAM} , Δt]

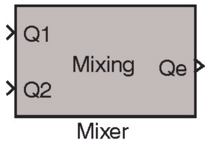
Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

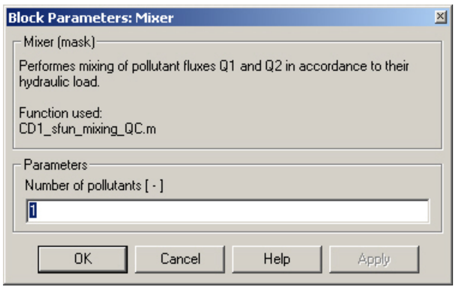
Initial Conditions:

No initial conditions required.

7 TOOLS

7.1 Mixing QC





Function:

The block is designed to combine flows from two sources by means of simple mixing.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_mixing_QC.m*

Input:

Q_1	Dynamic input of flow and pollutant concentrations associated $[q_1, C_{1,1}, C_{1,2}, \ldots, C_{1,np}]$
Q_2	Dynamic input of flow and pollutant concentrations associated $[q_2, C_{2,1}, C_{2,2}, \ldots, C_{2,np}]$

Output:

Q_E	Outflow from the block $[q_E, C_{E,1}, C_{E,2}, \ldots, C_{E,np}]$.

Parameters:

n_P Number of pollutants carried

S-function parameters: $[n_P, \Delta t]$

Δt Sampling rate Δt [s] is inherited from global setting of simulation parameters.

Initial Conditions:

No initial conditions required.

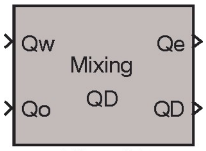
Theory:

Equation for mixing of substances

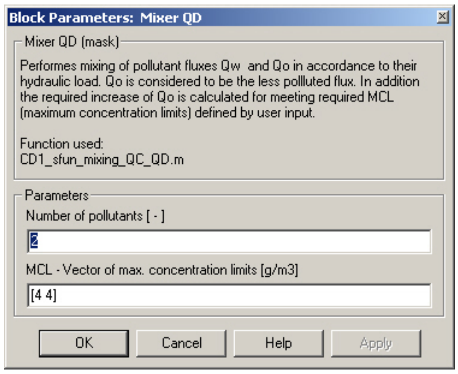
$$C_E = \frac{C_1 \cdot q_1 + C_2 \cdot q_2}{q_E} \text{ with}$$

$$q_E = q_1 + q_2$$

7.2 Mixing QC-QD



Mixer QD



Function:

The block is designed to combine flows from two sources by means of simple mixing. Two inflows Q_0 and Q_W are considered where Q_0 is considered being the less polluted flow. In extend to the block Mixing-QC the block evaluates the additional flow rates q_D required to meet user defined maximum concentration limits C_{MCL}.

A standard application of the block is to combine upstream base flow in a river (Q_0) and combined sewer overflow (Q_W) entering the river.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_mixing_QC_QD.m*

Input:

Q_w Dynamic input of flow and pollutant concentrations associated $[q_W, C_{W,1}, C_{W,2}, \ldots, C_{W,np}]$

Q_0 Dynamic input of flow and pollutant concentrations associated $[q_0, C_{0,1}, C_{0,2}, \ldots, C_{0,np}]$

Output:

Q_E Outflow from the block $[q_E, C_{E,1}, C_{E,2}, \ldots, C_{E,np}]$.

Q_D Vector of flow rate demands for each pollutant concentration respectively. $[q_{D1}, q_{D2}, q_{D3}, q_{D4}, \ldots q_{D,np}, q_{D,MAX}]$. The maximum flow rate demand out of all is stored in the vectors last entry.

Parameters:

n_P Number of pollutants carried

C_{MCL} Maximum concentrations limits defined by the user $[C_{MCL,1}, C_{MCL,2}, \ldots, C_{MCL,np}]$

S-function parameters: $[n_P, \Delta t, C_{MCL}]$

Δt *Sampling rate Δt [s] is inherited from global setting of simulation parameters.*

Initial Conditions:

No initial conditions required.

Theory:

Mixing of substances is done equivalent as in the block *Mixing-QC*.

$$C_E = \frac{C_1 \cdot q_1 + C_2 \cdot q_2}{q_E} \text{ with}$$

$$q_E = q_1 + q_2$$

Outflow concentrations are checked for compliance with user defined maximum concentration limits (C_{MCL}). In case of non-compliance the additional upstream flow (q_0) for sufficient dilution is returned. Based on the extended mixing formula

$$C'_E = \frac{q_W \cdot C_W + (q_0 + q_D) \cdot C_0}{q_W + q_0 + q_D} \leq C_{MCL}$$

the additional upstream demand of flow rate denotes as

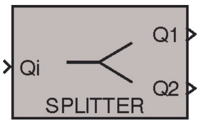
$$q_D \geq \frac{q_W \cdot (C_{MCL} - C_W) + q_0 \cdot (C_{MCL} - C_0)}{(C_0 - C_{MCL})}$$

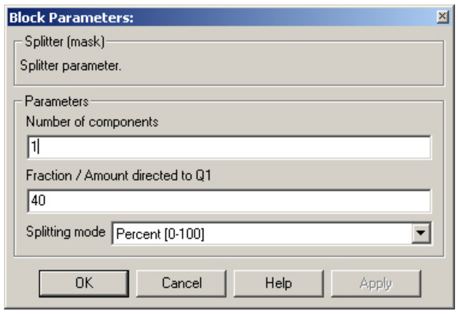
For having different substances considered, the minimum demanding flow rate is returned for each substance as well as the maximum out of these:

Q_D= [q_{D1}, q_{D2}, q_{D3}, q_{D4},… $q_{D,np}$, $q_{D,MAX}$]. with

$q_{D,MAX}$ = MAX {q_{D1}, q_{D2}, q_{D3}, q_{D4},… $q_{D,np}$}

7.3 Splitter





Function:

The block is designed to split flows according to a fixed flows rate of factor. Thus, the block may be used to model a pump operating with a fixed flow rate using the splitting mode *{ m³/s }* Alternative structures that divert the inflow into two flows at a specific ration may be modelled using either the splitting mode *{ Percent (0-100) }* or *{ Fraction (0-1) }*.

Block Type:

Discrete Block utilizing S-function: *CD1_sfun_splitter.m*

Input:

Q_I Dynamic input of flow and pollutant concentrations associated $[q_I, C_{I,1}, C_{I,2}, \ldots, C_{I,np}]$

Output:

Q_1 Outflow from the block; Fraction according to user input defined in the mask $[q_1, C_{I,1}, C_{I,2}, \ldots, C_{I,np}]$.

Q_2 Outflow from the block; Remaining fraction $q_I = q_I - q_1$ $[q_2, C_{I,1}, C_{I,2}, \ldots, C_{I,np}]$.

Parameters:

n_P Number of pollutants carried

f Fraction / amounted directed to Q_1

mode Splitting mode to be used

{ m³/s }	Fixed flow rate diverted to output port Q_1
{ Percent (0-100) }	Fixed percentage of flow diverted to output port Q_1
{ Fraction (0-1) }	Fixed fraction of flow diverted to output port Q_1

S-function parameters: $[n_P, f, \text{mode}, \Delta t]$

Δt *Sampling rate Δt [s] is inherited from global setting of simulation parameters.*

Initial Conditions:

No initial conditions required.

Theory:

Splitting of flows is done according to the splitting mode selected where the flow q_1 is calculated as:

<u>Mode { m³/s }</u>

$$f \le q_I \quad \rightarrow \quad q_1 = f \text{ and } q_2 = q_I - q_1$$

$$f \ge q_I \quad \rightarrow \quad q_1 = q_I \text{ and } q_2 = 0$$

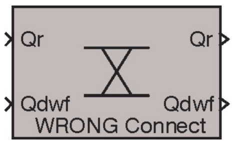
Mode { Percent (0-100) }

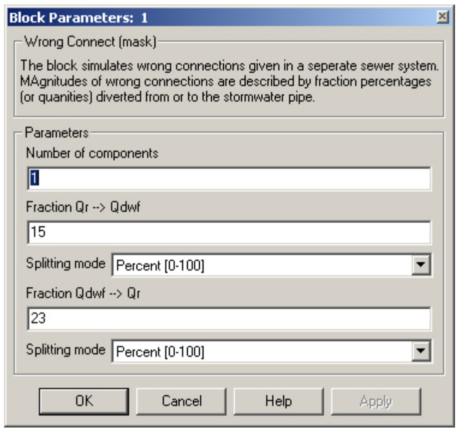
$$q_1 = \frac{f}{100} \cdot q_I \text{ and } q_2 = \left(1 - \frac{f}{100}\right) \cdot q_I$$

Mode { Fraction (0-1) }

$$q_1 = f \cdot q_I \text{ and } q_2 = (1 - f) \cdot q_I$$

7.4 Wrong Connect





Function:

The block is designed to model wrong connections that may occur in a separate sewer system. Thereby single households may be wrongly connected to the storm water sewer. On the other hand, storm water inlets may be connected to the waster water sewer.
Modelling of such wrong connections is done on a catchment level. It is not the number of wrong connections being of interest, but the overall fraction or quantity of flow in the catchment that is diverted from the storm to the waste sewer (and vice versa).

Block Type:

Discrete Block utilizing S-functions:

CD1_sfun_splitter.m
CD1_sfun_mixing_QC.m

Input:

Q_r Dynamic input of flow and pollutant concentrations representing storm water flow in the catchment. $[q_r, C_{r,1}, C_{r,2}, \ldots, C_{r,np}]$

Q_{dwf} Dynamic input of flow and pollutant concentrations representing dry weather flow in the catchment. $[q_{dwf}, C_{dwf,1}, C_{dwf,2}, \ldots, C_{dwf,np}]$

Output:

Q_r Flow and pollutant concentrations in the storm water pipe including wrong connections. $[q_r, C_{r,1}, C_{r,2}, \ldots, C_{r,np}]$

Q_{dwf} Flow and pollutant concentrations in the waste water pipe including wrong connections. $[q_{dwf}, C_{dwf,1}, C_{dwf,2}, \ldots, C_{dwf,np}]$

Parameters:

n_P Number of pollutants carried
$f_{r\text{-}dwf}$ Fraction / Amounted directed from the storm to the waste water pipe
$m_{r\text{-}dwf}$ Splitting mode to be used (see block Splitter for details)
$f_{dwf\text{-}r}$ Fraction / Amounted directed from the waste to the storm water pipe
$m_{dwf\text{-}r}$ Splitting mode to be used (see block Splitter for details)

Initial Conditions:

No initial conditions required.

Theory:

See blocks

Splitter: *CD1_sfun_splitter.m*
Mixer-QC: *CD1_sfun_mixing_QC.m*

for details.

8 LITERATURE

Achleitner S., DeToffol S., Engelhard C. and Rauch W. (2005). The European Water Framework Directive: Water Quality Classification and Implications to Engineering Planning. *Environmental Management*, **35** (1)

Blöch H. (1999). The European Water Framework Directive: Taking European water policy into the next millennium. *Water Science and Technology*, **40** (10),67-71.

Harremoës P. and Rauch W. (1996). Integrated design and analysis of drainage systems, including sewers, treatment plant and receiving waters. *Journal of Hydraulic Research*, **34** (6)

Lau J., Butler D. and Schütze M. (2002). Is combined sewer overflow spill frequency/volume a good indicator of receiving water quality impact? *Urban Water*, **4**,181–189.

Lijklema L. (1995). Water Quality Standards: Sense and Nonsense. *Water Science and Technology*, **31** (8),321-327.

Rauch W., Bertrand-Krajewski J. L., Krebs P., Mark O., Schilling W., Schütze M. and Vanrolleghem P. A. (2002). Deterministic modelling of integrated urban drainage systems. *Water Science and Technology*, **45** (3),81-94.

Roberson J. A., Cassidy J. and Chaudhry M. H. (1995). *Hydraulic Engineering*. John Wiley Sons, Inc., New York.