



1D/2D/3D Modelling suite for integral water solutions

DELFT3D FLEXIBLE MESH SUITE

Deltires systems

D-Flow Flexible Mesh

User Manual

Deltires
Enabling Delta Life 

D-Flow Flexible Mesh

D-Flow FM in Delta Shell

User Manual

Released for:
Delft3D FM Suite 2020

Version: 1.5.0
SVN Revision: 64819

December 6, 2019

D-Flow Flexible Mesh, User Manual

Published and printed by:

Deltares
Boussinesqweg 1
2629 HV Delft
P.O. 177
2600 MH Delft
The Netherlands

telephone: +31 88 335 82 73
fax: +31 88 335 85 82
e-mail: info@deltares.nl
www: <https://www.deltares.nl>

For sales contact:

telephone: +31 88 335 81 88
fax: +31 88 335 81 11
e-mail: software@deltares.nl
www: <https://www.deltares.nl/software>

For support contact:

telephone: +31 88 335 81 00
fax: +31 88 335 81 11
e-mail: software.support@deltares.nl
www: <https://www.deltares.nl/software>

Copyright © 2019 Deltares

All rights reserved. No part of this document may be reproduced in any form by print, photo print, photo copy, microfilm or any other means, without written permission from the publisher: Deltares.

Contents

List of Figures	xiii
List of Tables	xxi
1 A guide to this manual	3
1.1 Introduction	3
1.2 Overview	3
1.3 Manual version and revisions	4
1.4 Typographical conventions	4
1.5 Changes with respect to previous versions	5
2 Introduction to D-Flow Flexible Mesh	7
2.1 Areas of application	7
2.2 Standard features	7
2.3 Special features	8
2.4 Important differences compared to Delft3D-FLOW	8
2.5 Coupling to other modules	9
2.6 Installation	9
2.6.1 Installation of Delta Shell	9
2.6.2 Installation of the computational core	9
2.7 Examples	10
3 Getting started	11
3.1 Introduction	11
3.2 Overview of D-Flow FM GUI	11
3.2.1 Project window	12
3.2.2 The central window	13
3.2.3 Settings window	14
3.2.4 Map window	14
3.2.5 Messages window	15
3.2.6 Time Navigator window	15
3.3 Dockable views	15
3.3.1 Docking tabs separately	15
3.3.2 Multiple tabs	16
3.4 Ribbons and toolbars	16
3.4.1 Ribbons (shortcut keys)	17
3.4.2 File	17
3.4.3 Home	18
3.4.4 View	18
3.4.5 Tools	19
3.4.6 Map	19
3.4.7 Scripting	20
3.4.8 Shortcuts	21
3.4.9 Quick access toolbar	21
3.5 Important differences compared to Delft3D-FLOW GUI	22
3.5.1 Project vs model	22
3.5.2 Load/save vs import/export	22
3.5.3 Working from the map	22
3.5.4 Coordinate conversion	22
3.5.5 Model area	23
3.5.6 Integrated models (model couplings)	23
3.5.7 Ribbons (shortcut keys)	24
3.5.8 Context menus	24



3.5.9	Scripting	24
4	Tutorial	25
4.1	Introduction: Basic grid concepts	25
4.2	Tutorial Hydrodynamics	26
4.2.1	Outline of the tutorials	27
4.2.2	Tutorial 1: Creating a curvilinear grid	27
4.2.3	Tutorial 2: Creating a triangular grid	33
4.2.4	Tutorial 3: Coupling multiple separate grids	35
4.2.5	Tutorial 4: Inserting a bed level	36
4.2.6	Tutorial 5: Imposing boundary conditions	39
4.2.7	Tutorial 6: Defining output locations	42
4.2.8	Tutorial 7: Defining computational parameters	43
4.2.9	Tutorial 8: Running a model simulation	45
4.2.10	Tutorial 9: Viewing the output of a model simulation	46
5	All about the modelling process	51
5.1	Introduction	51
5.2	mdu-file and attribute files	51
5.3	Filenames and conventions	52
5.4	Setting up a D-Flow FM model	52
5.4.1	General	53
5.4.1.1	Vertical layer specification	53
5.4.1.2	Model coordinate system	55
5.4.1.3	Angle of latitude	55
5.4.2	Area	56
5.4.2.1	Grid-snapped features	56
5.4.2.2	Observation points	58
5.4.2.3	Observation cross-sections	59
5.4.2.4	Thin dams	60
5.4.2.5	Fixed weirs	61
5.4.2.6	Land boundaries	63
5.4.2.7	Dry points and Dry areas	64
5.4.2.8	Pumps	66
5.4.2.9	Weirs	67
5.4.2.10	Gates	68
5.4.2.11	Bridge Pillars	69
5.4.3	Computational grid	70
5.4.4	Bed level	70
5.4.5	Time frame	71
5.4.6	Processes	73
5.4.7	Initial conditions	73
5.4.8	Boundary conditions	75
5.4.8.1	Specification of boundary locations (support points)	75
5.4.8.2	Boundary data editor (forcing)	77
5.4.8.3	Import/export boundary conditions from the <i>Project</i> window	90
5.4.8.4	Overview of boundary conditions in attribute table (non-editable)	92
5.4.9	Physical parameters	93
5.4.9.1	Constants	93
5.4.9.2	Roughness	93
5.4.9.3	Viscosity	95
5.4.9.4	Wind	96
5.4.9.5	Heat Flux model	96
5.4.9.6	Tidal forces	96

5.4.10 Sources and sinks	96
5.4.11 Numerical parameters	98
5.4.12 Output parameters	98
5.4.13 Miscellaneous	102
5.5 Save project, MDU file and attribute files	103
6 Running a model	105
6.1 Introduction	105
6.1.1 Commandline executables	105
6.2 Parallel calculations using MPI	105
6.2.1 Introduction	105
6.2.2 Partitioning the model	106
6.2.2.1 More about the mesh partitioning	107
6.2.3 Partitioning the MDU file	109
6.2.3.1 Remaining model input	110
6.2.4 Running a parallel job	110
6.2.5 Visualizing the results of a parallel run	110
6.2.5.1 Plotting all partitioned map files with Delft3D-QUICKPLOT	111
6.2.5.2 Merging multiple map files into one	111
6.3 Running a scenario using Delta Shell	112
6.4 Running a scenario using a batch script	113
6.4.1 Running the partitioner or dfmoutput on Windows	114
6.4.2 Running the partitioner or dfmoutput on Linux	114
6.5 Run time	115
6.5.1 Multi-core performance improvements by OpenMP	115
6.6 Files and file sizes	116
6.6.1 History file	116
6.6.2 Map file	116
6.6.3 Restart file	117
6.7 Command-line arguments	117
6.8 Restart a simulation	118
6.9 Frequently asked questions	119
7 Visualize results	121
7.1 Introduction	121
7.2 Visualization with Delta Shell	121
7.3 Visualization with Delft3D-QUICKPLOT	122
7.4 Visualization with Muppet	123
7.5 Visualization with Matlab	123
7.6 Visualization with Python	123
8 Hydrodynamics	125
8.1 Introduction	125
8.2 General background	125
8.2.1 Range of applications of D-Flow FM	125
8.2.2 Physical processes	126
8.2.3 Assumptions underlying D-Flow FM	127
8.3 Hydrodynamic processes	128
8.3.1 Topological conventions	129
8.3.2 Conservation of mass and momentum	132
8.3.2.1 Continuity equation	132
8.3.2.2 Momentum equations in horizontal direction	133
8.3.2.3 Vertical velocities	133
8.3.3 The hydrostatic pressure assumption	133

8.3.4	The Coriolis force	134
8.3.5	Diffusion of momentum	134
8.4	Hydrodynamics boundary conditions	135
8.4.1	Open boundary conditions	136
8.4.1.1	The location of support points	136
8.4.1.2	Physical information	137
8.4.1.3	Example	142
8.4.1.4	Miscellaneous	144
8.4.2	Vertical boundary conditions	145
8.4.2.1	Bed friction and conveyance	146
8.4.2.2	Conveyance in 2D	147
8.4.3	Shear-stresses at closed boundaries	149
8.5	Artificial mixing due to sigma-coordinates	149
8.6	Secondary flow	153
8.6.1	Definition	154
8.6.2	Depth-averaged continuity equation	155
8.6.3	Momentum equations in horizontal direction	155
8.6.4	Effect of secondary flow on depth-averaged momentum equations	155
8.6.5	The depth averaged transport equation for the spiral motion intensity	156
8.7	Drying and flooding	157
8.7.1	Definitions	157
8.7.1.1	Piecewise constant approach for the bed level	158
8.7.1.2	Piecewise linear approach for the bed levels	159
8.7.1.3	Hybrid bed level approach	160
8.7.2	Specification in Delta Shell	160
8.8	Intakes, outfalls and coupled intake-outfalls	161
8.9	Equations of state for the density	163
8.10	Tide generating forces	164
8.11	Advection at open boundaries	166
9	Transport of matter	167
9.1	Introduction	167
9.2	Some words about suspended sediment transport	168
9.3	Transport processes	168
9.3.1	Advection	168
9.3.2	Diffusion	169
9.3.3	Sources and sinks	170
9.3.4	Forester filter	171
9.4	Transport boundary and initial conditions	171
9.4.1	Open boundary conditions	171
9.4.2	Closed boundary conditions	172
9.4.3	Vertical boundary conditions	172
9.4.4	Thatcher-Harleman boundary conditions	172
9.4.5	Initial conditions	172
10	3D Modelling	175
10.1	Spurious oscillations in vertical velocity	176
10.2	Anti creep	178
10.3	Turbulence modelling	178
10.3.1	k-epsilon turbulence model	181
10.3.2	k-tau turbulence model	183
11	Heat transport	185
11.1	Heat balance	188

11.2	Solar radiation	188
11.3	Atmospheric radiation (long wave radiation)	190
11.4	Back radiation (long wave radiation)	191
11.5	Effective back radiation	191
11.6	Evaporative heat flux	191
11.7	Convective heat flux	193
12	Wind	195
12.1	Definitions	195
12.1.1	Nautical convention	195
12.1.2	Drag coefficient	195
12.2	File formats	198
12.2.1	Defined on the computational grid	199
12.2.1.1	Specification of uniform wind through velocity components .	199
12.2.1.2	Specification of uniform wind through magnitude and direction	200
12.2.2	Defined on an equidistant grid	201
12.2.3	Defined on a curvilinear grid	202
12.2.4	Space and time varying Charnock coefficients	204
12.2.5	Defined on a spiderweb grid	204
12.2.6	Combination of several wind specifications	206
12.3	Masking of points in the wind grid from interpolation ('land-sea mask')	208
13	Hydraulic structures	211
13.1	Introduction	211
13.2	Structures	211
13.2.1	Fixed weirs	212
13.2.2	(adjustable) Weirs	212
13.2.3	Gates	216
13.2.4	Pumps	217
13.2.5	Thin dams	217
14	Bedforms and vegetation	219
14.1	Bedform heights	219
14.2	Trachytopes	219
14.2.1	Trachytype classes	219
14.2.2	Averaging and accumulation of trachytopes	226
14.3	(Rigid) three-dimensional vegetation model	227
15	Calibration factor	229
16	Coupling with D-Waves (SWAN)	231
16.1	Getting started	231
16.1.1	Input D-Flow FM	232
16.1.2	Input D-Waves	232
16.1.3	Input dimr	233
16.1.4	Online process order	235
16.1.5	Related files	236
16.2	Forcing by radiation stress gradients	237
16.3	Stokes drift and mass flux	238
16.4	Streaming	239
16.5	Enhancement of the bed shear-stress by waves	239
17	Coupling with D-RTC (FBC-Tools)	245
17.1	Introduction	245
17.2	Getting started	245

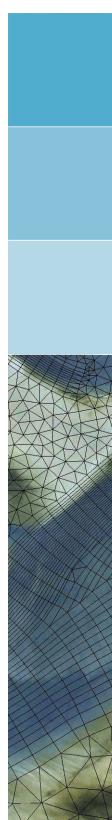
17.2.1	User interface: the first steps	245
17.2.2	Input D-Flow FM	246
17.2.3	Input D-RTC	247
17.2.4	Input dimr	247
17.2.5	Online process order	247
17.3	Technical background	247
17.3.1	BMI interface to interact with the D-Flow FM model state	247
17.3.2	DIMR and the BMI	247
17.3.3	BMI access to hydraulic structure data in D-Flow FM	248
17.3.4	BMI access to observations in D-Flow FM	249
17.3.5	BMI access to forcings in D-Flow FM	249
18	Coupling with D-Water Quality (Delwaq)	251
18.1	Introduction	251
18.2	Offline versus online coupling	251
18.3	Creating output for D-Water Quality	251
18.4	Current limitations	252
19	Morphology	253
20	Water Quality Processes	255
20.1	Introduction	255
20.2	Definition of the water quality system	256
20.2.1	Substances	257
20.2.2	Processes	257
20.2.3	Dry segments	259
20.3	Definition of processes parameters	259
20.3.1	Constant parameter input	259
20.3.2	Spatial parameter input	260
20.3.3	Temporal parameter input	260
20.3.4	Spatial and temporal parameter input	261
20.4	Initial conditions, boundary conditions and sources and sinks	262
20.4.1	Initial condition	262
20.4.2	Boundary conditions	262
20.4.3	Sources and sinks	263
20.5	Output options	263
20.5.1	Map and history output	263
20.5.2	Mass balance areas	264
20.6	Running D-Flow FM with processes	265
20.7	Output	265
20.8	Known issues and limitations	265
21	Calibration and data assimilation	267
21.1	Introduction	267
21.2	Getting started with OpenDA	267
21.3	The OpenDA black box model wrapper for D-Flow FM	268
21.4	OpenDA configuration	268
21.4.1	Main configuration file and the directory structure	268
21.4.2	The algorithm configuration	270
21.4.3	The stochObserver configuration	270
21.4.3.1	NoosTimeSeriesStochObserver	270
21.4.3.2	IoObjectStochObserver	271
21.4.4	The stochModel configuration	272
21.4.5	D-Flow FM files and the OpenDA dataObjects configuration	273
21.4.5.1	Start and end time in the model definition file (.mdu)	273

21.4.5.2 External forcings (.xyz)	273
21.4.5.3 Boundary time series (.tim)	275
21.4.5.4 Meteorological boundary conditions (<*.amu>, <*.amv>, <*.amp>)	275
21.4.5.5 Result time series (<*_his.nc>)	276
21.4.5.6 Restart file (<*_map.nc>)	276
21.4.5.7 Calibration factor definition file (<*.cll>)	276
21.4.5.8 Trachytopes roughness definition file (<*.ttt>)	278
21.5 Generating noise	279
21.6 Examples of the application of OpenDA for D-Flow FM	281
21.6.1 Example 1: Calibration of the roughness parameter	281
21.6.2 Example 2: EnKF with uncertainty in the tidal components	283
21.6.3 Example 3: EnKF with uncertainty in the inflow velocity	284
21.6.4 Example 4: EnKF with uncertainty in the inflow condition for salt	284
21.6.5 Example 5: EnKF with uncertainty on the wind direction	285
21.6.6 Example 6: EnKF with the DCSM v5 model and uncertainty on the wind direction	285
References	287
A The master definition file	291
B The net file for flexible meshes	297
B.1 2D grids in NetCDF UGRID files	297
C Attribute files	299
C.1 Introduction	299
C.2 Polyline/polygon file	299
C.3 Sample file	300
C.4 Time series file (ASCII)	301
C.5 The external forcings file	301
C.5.1 Old style external forcings	301
C.5.2 New style external forcing (boundary conditions only)	302
C.5.2.1 Boundary definitions	302
C.5.3 Accepted quantity names	303
C.6 Trachytopes	304
C.6.1 Area Roughness on Links (ARL-file)	305
C.6.1.1 Example	306
C.6.1.2 Conversion from Delft3D 4 input files	306
C.6.2 Trachytype Definition file (TTD-file)	306
C.6.2.1 General format	306
C.6.2.2 Example	307
C.6.2.3 Discharge dependent format	307
C.6.2.4 Water level dependent format	307
C.6.2.5 Supported roughness formulations	308
C.7 Fixed weirs	309
C.7.0.1 Example	309
C.8 Calibration Factors	310
C.8.1 Calibration factor definition file (CLD-file)	310
C.8.1.1 Header of the CLD-file	310
C.8.1.2 Constant values	310
C.8.1.3 Discharge dependent format	310
C.8.1.4 Water level dependent format	311
C.8.1.5 Example	311
C.8.2 Calibration Class Area on Links (CLL-file)	312

C.8.2.1	Header of the CLL-file	312
C.8.2.2	Body of the CLL-file	312
C.8.2.3	Example	312
C.9	Sources and sinks	312
C.10	Dry points and areas	313
C.11	Structure INI file	313
C.12	Space varying wind and pressure	314
C.12.1	Meteo on equidistant grids	314
C.12.2	Curvilinear data	317
C.12.3	Spiderweb data	320
C.12.4	Meteo on a NetCDF file	324
C.13	Fourier analysis	324
D	Initial conditions and spatially varying input	327
D.1	Introduction	327
D.2	Supported quantities	327
D.2.1	Accepted quantity names in initial field files	327
D.2.2	Water levels	327
D.2.3	Initial velocities	327
D.2.4	Salinity	327
D.2.5	Temperature	327
D.2.6	Tracers	327
D.2.7	Sediment	328
D.2.8	Physical coefficients	328
D.3	Supported file formats	328
D.3.1	Inside-polygon option	328
D.3.2	Sample file	329
D.3.3	Vertical profile file	329
D.3.4	Map file	329
D.3.5	Restart file	329
E	Boundary conditions specification	331
E.1	Supported boundary types	331
E.1.1	Astronomic boundary conditions	331
E.1.2	Astronomic correction factors	331
E.1.3	Harmonic flow boundary conditions	331
E.1.4	QH-relation flow boundary conditions	331
E.1.5	Time-series flow boundary conditions	332
E.1.6	Time-series transport boundary conditions	332
E.1.7	Time-series for the heat model parameters	332
E.2	Boundary signal file formats	332
E.2.1	The <cmp> format	332
E.2.2	The <qh> format	333
E.2.3	The <bc> format	333
E.2.3.1	Name in a bc block	334
E.2.3.2	Function in a bc block	335
E.2.4	The NetCDF-format for point-location time-series	336
F	Output files	339
F.1	Diagnostics file	339
F.2	Demanding output	340
F.2.1	The MDU-file	340
F.2.2	Observation points	340
F.2.2.1	The observation point file with extension <*.xyn>	340

F.2.3	Moving observation points	340
F.2.4	Observation cross sections	341
	F.2.4.1 Observation cross section file with extension <*_crs.pli>	341
F.3	NetCDF output files	341
F.3.1	Timeseries as NetCDF his-file	342
	F.3.1.1 Dimensions	342
	F.3.1.2 Location variables	342
	F.3.1.3 Variables on stations	343
	F.3.1.4 Variables on cross sections	343
	F.3.1.5 Mass balance output	343
	F.3.1.6 Variables on sources/sinks	344
	F.3.1.7 Hydraulic structure output	344
F.3.2	Spatial fields as NetCDF map-file	344
F.3.3	Class map files as NetCDF clm-file	347
F.3.4	Restart files as NetCDF rst-file	347
F.4	Shapefiles	347
F.5	Post processing on his-file	348
F.5.1	Max25 function	348
F.5.2	Maximum based on a generic filter	348
G	Model generation	349
G.1	Introduction	349
G.2	Grid generation	349
	G.2.1 Uniform Cartesian grid	349
	G.2.2 Locally refined Cartesian grid	349
H	Spatial editor	351
H.1	Introduction	351
H.2	General	351
	H.2.1 Overview of spatial editor	351
	H.2.2 Import/export dataset	352
	H.2.3 Activate (spatial) model quantity	353
	H.2.4 Colorscale	354
	H.2.5 Render mode	355
	H.2.6 Context menu	356
H.3	Quantity selection	357
H.4	Geometry operations	358
	H.4.1 Polygons	358
	H.4.2 Lines	359
	H.4.3 Points	360
H.5	Spatial operations	360
	H.5.1 Import point cloud	361
	H.5.2 Crop	361
	H.5.3 Delete	362
	H.5.4 Set value	363
	H.5.5 Contour	364
	H.5.6 Copy to samples	366
	H.5.7 Copy to spatial data	367
	H.5.8 Merge spatial data	368
	H.5.9 Gradient	369
	H.5.10 Interpolate	370
	H.5.11 Smoothing	373
	H.5.12 Overwrite (single) value	375
H.6	Operation stack	375

H.6.1	Stack workflow	375
H.6.2	Edit operation properties	376
H.6.3	Enable/disable operations	376
H.6.4	Delete operations	379
H.6.5	Refresh stack	380
H.6.6	Quick links	380
H.6.7	Import/export	381
Index		383



List of Figures

3.1	Start-up lay-out Delta Shell	11
3.2	Project window of D-Flow FM plugin	13
3.3	The central window with a map and contents of the D-Flow FM model	13
3.4	The model Settings window with General settings tab enabled	14
3.5	Map window controlling map contents	14
3.6	Log of messages, warnings and errors in message window	15
3.7	Time Navigator window in Delta Shell	15
3.8	Docking windows on two screens within the Delta Shell framework.	15
3.9	Bringing the Time Navigator window to the front	16
3.10	Docking the Time Navigator window.	16
3.11	Auto hide the Properties window	16
3.12	Perform operations using the shortcut keys	17
3.13	The <i>File</i> ribbon.	17
3.14	The Delta Shell options dialog.	18
3.15	The <i>Home</i> ribbon.	18
3.16	The <i>View</i> ribbon.	18
3.17	The <i>Tools</i> ribbon contains just the <i>Data</i> item.	19
3.18	The <i>Map</i> ribbon.	19
3.19	The scripting <i>ribbon</i> within Delta Shell.	20
3.20	The quick access toolbar.	21
3.21	Set map coordinate system using right mouse button	23
3.22	Select a coordinate system using the quick search bar	23
3.23	Perform operations using the shortcut keys	24
4.1	Topology and definitions for a grid as used in D-Flow FM.	25
4.2	Perfect orthogonality and nearly perfect smoothness along the edge connecting two triangles. Black lines/dots are network links/nodes, blue lines/dots are flow links/nodes.	26
4.3	Poor grid properties due to the violation of either smoothness or orthogonality at the edge connecting two triangles	26
4.4	Start RGFGRID by a double-click on <i>Grid</i>	28
4.5	Splines in Tutorial01	28
4.6	Settings for the <i>Grow Grid from Splines</i> procedure.	29
4.7	Generated curvilinear grid after the new <i>Grow Grid from Splines</i> procedure.	30
4.8	Orthogonality of the generated curvilinear grid after the <i>Grow Grid from Splines</i> procedure.	31
4.9	Importing a land boundary	32
4.10	After closing RGFGRID the grid is visible in the Delta Shell GUI.	32
4.11	Polygon that envelopes the area in which an unstructured grid is aimed to be established.	33
4.12	Unstructured grid, after having refined the polygon.	34
4.13	Connection of the river grid and the unstructured grid. The red lines show the inserted grid lines used to couple the two grids manually.	35
4.14	Orthogonality of the integrated grid, containing the curvilinear part, the triangular part and the coupling between the two grids.	36
4.15	Map-ribbon with the <i>Spatial Operations</i> menu.	37
4.16	Activate import samples 1.	37
4.17	Interpolated bed levels values at the grid (estuary).	38
4.18	Interpolated bed levels values at the grid (harbour).	38
4.19	Location of the two open boundaries at the sea and river side.	40
4.20	Selection of Boundary01.	40
4.21	Boundary conditions seaside.	41

4.22	Boundary condition riverside.	41
4.23	Overview cross sections and observation points.	43
4.24	The time frame of the simulation.	44
4.25	Imposed initial conditions for the simulation.	44
4.26	Optional output parameters for the computation.	44
4.27	Menu for saving a project.	45
4.28	View of Delta Shell when running a model.	46
4.29	View of Delta Shell, available to select a location for timeseries in.	47
4.30	View of Delta Shell, time-series for observation point "water level at Borsele".	47
4.31	WMS layer icon at the top of the map-tree viewer.	47
4.32	View of Delta Shell in combination with OpenStreetMap.	48
4.33	Select <i>waterlevel(s1)</i> from the map tree	48
4.34	Layer properties editor	49
5.1	The Add Model drop-down menu	53
5.2	The <i>Select model ...</i> window	53
5.3	Overview of general tab	53
5.4	Vertical layer specification window (σ -model is β -functionality)	54
5.5	Coordinate system wizard	55
5.6	Overview of geographical features	56
5.7	Overview of map ribbon	56
5.8	Example of expanded <i>Estimated Grid-snapped features</i> attribute in the <i>Map</i> window	57
5.9	Example of grid snapped features displayed on the central map	58
5.10	Geographical and grid snapped representation of an observation point	58
5.11	Attribute table with observation points	59
5.12	Geographical and grid snapped representation of a cross section	59
5.13	Attribute table with observation cross sections	60
5.14	Geographical and grid snapped representation of a thin dam	60
5.15	Attribute table with thin dams	61
5.16	Geographical and grid snapped representation of a fixed weir	61
5.17	Schematic representation of a fixed weir	61
5.18	Attribute table with fixed weirs	62
5.19	Fixed weir editor	62
5.20	Geographical representation of a land boundary	63
5.21	Attribute table with land boundaries	63
5.22	Geographical and grid snapped representation of several dry points	64
5.23	Attribute table with dry points	65
5.24	Geographical and grid snapped representation of a dry area	65
5.25	Attribute table with dry areas	66
5.26	Polygon for pump (a) and adjustment of physical properties (b).	66
5.27	Selection of the pumps	67
5.28	Polygon for adjustable weir (a) and adjustment of geometrical and temporal conditions (b).	67
5.29	Time series for crest level.	68
5.30	Time series for crest level.	68
5.31	Polygon for gate (a) and adjustment of geometrical and temporal conditions (b).	69
5.32	Selection of a bridge	69
5.33	Adjustment of the properties of the pillars	70
5.34	Bed level activated in the spatial editor	71
5.35	Overview time frame tab	71
5.36	Relation between Reference Date and the simulation start and stop time for astronomic- and harmonic-series as used in the simulation. Time-series should cover the simulation time.	72

5.37 Overview processes tab	73
5.38 Initial conditions in the <i>Project</i> window	73
5.39 The ‘Initial Conditions’ tab where you can specify the uniform values and the layer distributions of the active physical quantities.	74
5.40 Initial water levels activated in the spatial editor	74
5.41 Selecting 3 dimensional initial fields from the dropdown box in the ‘Map’ ribbon to edit them in the spatial editor	74
5.42 Restart files in output states folder	75
5.43 Restart file in initial conditions attribute	75
5.44 Adding a boundary support point on a polyline in the central map	76
5.45 Polyline added in <i>Project</i> window under ‘Boundary Conditions’	76
5.46 Geometry edit options in <i>Map</i> ribbon	76
5.47 Edit name of polyline/boundary in Boundaries tab	77
5.48 Overview of the boundary data editor	78
5.49 Process and quantity selection in the boundary data editor	79
5.50 Activate a support point	80
5.51 Specification of time series in the boundary data editor (left panel)	80
5.52 Window for generating series of time points	81
5.53 Csv import wizard: csv file selection	81
5.54 Clipboard/csv import wizard: specification of how data should be parsed into columns	82
5.55 Clipboard/csv import wizard: specification of how values should be parsed and columns should be mapped	83
5.56 Window for entering input to download boundary data from WPS	84
5.57 Specification of harmonic components in boundary data editor	85
5.58 Selection of astronomical components from list (after pressing ‘select components’)	86
5.59 Suggestions for astronomical components in list	87
5.60 Editing harmonic/astronomic components and their corrections	87
5.61 Specification of a Q-h relationship	88
5.62 Selection of vertically uniform or varying boundary conditions in case of a 3D model	88
5.63 Overview of the layer view component of the boudary conditions editor	89
5.64 Specification of boundary forcing data (in this example for salinity) at 3 positions in the vertical	89
5.65 Example of active and total signal for multiple water level data series on one support point	89
5.66 Importing or exporting boundary features — both polylines <*.pli> and forcing <*.bc> — from the <i>Project</i> window using the right mouse button	90
5.67 Import or export a <*.pli>-file as is or with coordinate transformation.	90
5.68 Import or export a <*.pli>-file as is or with coordinate transformation.	91
5.69 Import or export a *.pli file as is or with coordinate transformation.	92
5.70 Overview of all boundary conditions in attribute table	93
5.71 The physical parameters in the <i>Project</i> window	93
5.72 The section of the ‘Physical Parameters’ tab where you can specify roughness related parameters and formulations.	94
5.73 Roughness activated in the spatial editor to create/edit a spatially varying field	94
5.74 The section of the ‘Physical Parameters’ tab where you can specify (uniform) values for the horizontal and vertical eddy viscosity and diffusivity.	95
5.75 Viscosity activated in the spatial editor to create/edit a spatially varying field	95
5.76 Overview of parameters in sub-tab Wind	96
5.77 Activate the sources and sinks editing icon in the <i>Map</i> ribbon	97
5.78 Add sources and sinks in the central map using the ‘Sources and sinks’ icon.	97
5.79 Sources and sinks appearing in the <i>Project</i> window	98

5.80	Specifying time series for sources and sinks in the sources and sinks editor	98
5.81	<i>Output Parameters</i> tab	98
5.82	Overview <i>Output Parameters</i> tab	100
5.83	Model/data import drop-down menu	103
5.84	Model/data import wizard	103
6.1	Partitioning exporter dialog	108
6.2	Domain selector in Delft3D-QUICKPLOT for partitioned map files.	111
6.3	Selecting the model you want to run in the <i>Project</i> window	112
6.4	Group Run in Home ribbon	113
6.5	Run console Delta Shell	113
7.1	Example of setting output (in)visible in the <i>Map</i> window	121
7.2	Useful map visualization options in the Delft3D-QUICKPLOT	122
7.3	Example of the Muppet visualization of the D-Flow FM map output file	123
8.1	Example of σ -model (left) and Z-model (right).	129
8.2	Flexible mesh topology	130
8.3	Two conventional definitions of the cell center of a triangle: the <i>circumcenter</i> and the <i>mass center</i>	131
8.4	Perfect orthogonality and nearly perfect smoothness along the edge connecting two triangles. Black lines/dots are network links/nodes, blue lines/dots are flow links/nodes.	131
8.5	Poor mesh properties due to violating either the smoothness or the orthogonality at the edge connecting two triangles	132
8.6	Input for map projection for specifying Coriolis parameter on the grid.	134
8.7	Input parameters for horizontal and vertical eddy viscosities.	135
8.8	Virtual boundary 'cells' near the shaded boundary	136
8.9	Delta-Shell view of a simple channel covered by a straightforward Cartesian grid. Boundary conditions are prescribed at the left hand side and the right hand size of the domain.	142
8.10	Bed representation with uniform depth levels (a), and locally sloping bed (b).	147
8.11	A schematic view of the linear variation over the width for calculating the flow parameters.	148
8.12	Example of a hydrostatic consistent and inconsistent grid; (a) $H\delta\sigma > \sigma \frac{\partial H}{\partial x} \delta x$, (b) $H\delta\sigma < \sigma \frac{\partial H}{\partial x} \delta x$	150
8.13	Finite Volume for diffusive fluxes and pressure gradients	151
8.14	Left and right approximation of a strict horizontal gradient	151
8.15	Vertical profile secondary flow (v) in river bend and direction bed stress	154
8.16	Definition of the water levels, the bed levels and the velocities in case of two adjacent triangular cells.	158
8.17	Specification of the conveyance option in Delta Shell.	160
8.18	Specification of the bed level treatment type in Delta Shell.	160
8.19	Specification of the hybrid bed options (with keywords <code>blminabove</code> and <code>blmeanbelow</code>).	161
10.1	Vertical grid concepts: the σ model (left) and z-coordinate model (right)	175
10.2	Illustration of spurious oscillations near open boundaries	177
10.3	Illustration of unphysical oscillations	177
10.4	Illustration of monitor for occurrence of spurious oscillations	178
11.1	Overview of the heat exchange mechanisms at the surface	185
11.2	Co-ordinate system position Sun δ : declination; θ : latitude; wt : angular speed	190

12.1	Nautical conventions for the wind.	195
12.2	Prescription of the dependency of the wind drag coefficient C_d on the wind speed is achieved by means of at least 1 point, with a maximum of 3 points.	196
12.3	Grid definition of the spiderweb grid for cyclone winds.	204
13.1	Selection of structures (and other items) in the toolbar.	211
13.2	Input for simple weir	213
13.3	General structure, side view	213
13.4	General structure, top view	214
13.5	Input for a general structure.	215
13.6	Input for a gate	216
16.1	Schematic view of non-linear interaction of wave and current bed shear-stresses (from Soulsby <i>et al.</i> (1993b, Figure 16, p. 89))	240
16.2	Inter-comparison of eight models for prediction of mean and maximum bed shear-stress due to waves and currents (from Soulsby <i>et al.</i> (1993b, Figure 17, p. 90))	242
17.1	An <i>Integrated model</i> in the Project window	245
17.2	Example of a <i>Control flow</i> in D-RTC	246
20.1	General overview of substances included in D-Water Quality	258
21.1	Visualisation of the EnKF computation results from OpenDA for a certain observation point	284
C.1	Illustration of the data to grid conversion for meteo input on a separate curvilinear grid	319
C.2	Wind definition according to Nautical convention	321
C.3	Spiderweb grid definition	322
H.1	Overview of spatial editor functionality in Map ribbon	351
H.2	Importing a point cloud into the project using the context menu on "project" in the project tree	352
H.3	Importing a point cloud into the project using the "Import" drop-down menu in the Spatial Operations ribbon	353
H.4	Activate the imported point cloud in the spatial editor by double clicking it in the project tree	353
H.5	Activate the imported point cloud in the spatial editor by selecting it from the dropdown box in the Map ribbon	353
H.6	Legend button in Map ribbon	354
H.7	Activate the colorscale using the Legend in the map ribbon	354
H.8	Edit the colorscale properties using the context menu on the active layer in the Map Tree	355
H.9	Select the rendermode for the active layer in the property grid.	356
H.10	Example of a coverage rendered as colored numbers.	356
H.11	Selecting a smoothing operation for a polygon geometry from the context menu (using context menu)	357
H.12	Activating a spatial quantity by double clicking it in the project tree (in this example 'Initial Water Level')	357
H.13	Activating a spatial quantity by selecting it from the dropdown box in the 'Spatial Operations' ribbon	358
H.14	Overview of the available geometry operations in the 'Spatial Operations' ribbon	358
H.15	Activating the polygon operation and drawing polygons in the central map.	359
H.16	Activating the line operation and drawing lines in the central map.	359

H.17	Activating the ‘Add points’ operation, drawing them in the central map and assigning a value to them.	360
H.18	Overview of the available spatial operations in the ‘Spatial Operations’ ribbon	360
H.19	Importing a point cloud using the ‘Import’ operation from the ‘Spatial Operations’ ribbon	361
H.20	Option to perform a coordinate transformation on the imported point cloud	361
H.21	Appearance of import point cloud operation in the operations stack	361
H.22	Performing a crop operation on a point cloud with a polygon using ‘Crop’ from the ‘Spatial Operations’ ribbon	362
H.23	Appearance of crop operation in the operations stack	362
H.24	Performing an delete operation on a point cloud with a polygon using ‘Delete’ from the ‘Spatial Operations’ ribbon	363
H.25	Appearance of delete operation in the operations stack	363
H.26	Performing a set value operation (e.g. overwrite) on a point cloud with a polygon using ‘Set Value’ from the ‘Spatial Operations’ ribbon	364
H.27	Appearance of set value operation in the operations stack	364
H.28	Import a nautical chart as a georeferenced tiff file	365
H.29	Set the right map coordinate system for the geotiff	365
H.30	Performing a contour operation on a nautical chart using lines to define the contours and ‘Contour’ from the ‘Spatial Operations’ ribbon	365
H.31	Bring the sample set to the front if it appears behind the nautical chart	366
H.32	Appearance of contour operation in the operations stack	366
H.33	Applying the copy to samples operation	366
H.34	Copy to samples operation result	367
H.35	Applying the copy spatial data operation	367
H.36	Copy spatial data operation result	367
H.37	Activating the merge spatial data tool from the ribbon	368
H.38	The merge operation requests a pointwise combination method	368
H.39	Resulting grid coverage	369
H.40	Performing a gradient operation on a point cloud with a polygon using ‘Gradient’ from the ‘Spatial Operations’ ribbon	369
H.41	Appearance of gradient operation in the operations stack	370
H.42	Interpolation Operation options	370
H.43	Averaging options	371
H.44	Performing an interpolation operation on a single sample set (without using a polygon) using ‘Interpolate’ from the ‘Spatial Operations’ ribbon	371
H.45	Appearance of interpolation of ‘set1’ to the coverage ‘bed level’ in the operations stack	372
H.46	Performing an interpolation operation on multiple sample sets (without using a polygon) using ‘Interpolate’ from the ‘Spatial Operations’ ribbon	372
H.47	Appearance of interpolation of ‘set1’ and ‘set2’ to the coverage ‘bed level’ in the operations stack	373
H.48	Performing a smoothing operation on a point cloud with a polygon using ‘Smoothing’ from the ‘Spatial Operations’ ribbon	373
H.49	Appearance of smoothing operation in the operations stack	374
H.50	The cursor for the overwrite operation showing the value of the closest coverage point	374
H.51	Performing an overwrite operation on a coverage point using ‘Single Value’ from the ‘Spatial Operations’ ribbon	375
H.52	Appearance of overwrite operation in the operations stack	375
H.53	The ‘Operations’ panel with the operations stack. In this example ‘bed level’ is the coverage (e.g. trunk) that is edited. The point clouds ‘set 1’ and ‘set 2’ (e.g. branches) are used to construct the ‘bed level’ coverage.	376

H.54	Input for the operation (top panel), mask for the operation (middle panel) and output of the operation (bottom panel)	377
H.56	Disabling an operation using the boxed cross icon in the stack menu. The operation will become grey. Note the exlamation marks marking the stack 'out of sync'.	377
H.55	Editing the value or 'Pointwise operation' of a 'Set Value' operation using the properties panel	378
H.57	Removing an operation from the stack using the cross icon in the stack menu	379
H.58	Removing an operation from the stack using the context menu on the selected operation	379
H.59	Refresh the stack using the 'Refresh' button so that all operation are (re-) evaluated	380
H.60	Quick link to the original dataset before performing any spatial operations	380
H.61	Quick link to the output after performing all (enabled) operations	381

List of Tables

3.1	Functions and their descriptions within the scripting <i>ribbon</i> of Delta Shell.	20
3.2	Shortcut keys within the scripting editor of Delta Shell.	21
3.2	Shortcut keys within the scripting editor of Delta Shell.	21
5.1	Overview and description of numerical parameters	99
5.2	Input and output parameters of the example	102
5.3	Time (after <i>Reference Date</i> in seconds) of output files	102
5.4	Overview and description miscellaneous parameters	102
16.1	Fitting coefficients for wave/current boundary layer model	241
17.1	Supported compound variable fields for hydraulic structures in D-Flow FM via BMI.	248
17.2	Supported compound variable fields for observations in D-Flow FM via BMI.	249
17.3	Supported compound variable fields for forcings in D-Flow FM via BMI.	249
20.1	File base D-Water Quality versus D-Water Quality process in D-Flow FM.	255
20.2	Various types of parameter inputs for water quality models in D-Flow FM.	259
20.3	Data form D-Flow FM that is available for water quality processes	261
21.1	Directory structure of the OpenDA Ensemble Kalman filtering configuration for the simple Waal D-Flow FM model.	269
21.2	D-Flow FM files that can be manipulated and the corresponding OpenDA class names to be used in the <code>dflowfmWrapper.xml</code> file.	273
A.1	Standard MDU-file with default settings.	291
C.1	Boundary definitions in new style external forcing file.	302
C.2	List of accepted external forcing quantity names.	303
E.1	Description of <code><bc></code> format.	333
F.1	Features and MDU settings for generating shapefiles	347

1 A guide to this manual

1.1 Introduction

This User Manual describes the hydrodynamic module D-Flow Flexible Mesh (D-Flow FM) which is part of the Delft3D Flexible Mesh Suite or D-HYDRO Suite.

This module is part of several Modelling suites, released by Deltares as Deltares Systems or Dutch Delta Systems. These modelling suites are based on the Delta Shell framework. The framework enables to develop a range of modeling suites, each distinguished by the components and — most significantly — the (numerical) modules, which are plugged in. The modules which are compliant with the Delta Shell framework are released as *D-Name of the module*, for example: D-Flow Flexible Mesh, D-Waves, D-Water Quality, D-Real Time Control and D-Rainfall Runoff.

Therefore, this User Manual is shipped with several modelling suites. On the *Start Page* links are provided to all relevant User Manuals (and Technical Reference Manuals) for that modelling suite. It will be clear that the Delta Shell User Manual is shipped with all these modelling suites. Other user manuals can be referenced. In that case, you need to open the specific user manual from the *Start Page* in the central window. Some texts are shared in different user manuals, in order to improve the readability.

1.2 Overview

To make this manual more accessible we will briefly describe the contents of each chapter.

If this is your first time to start working with D-Flow FM we suggest you to read [Chapter 3, Getting started](#) and practice the tutorial of [Chapter 4](#). These chapters explain the user interface and guide you through the modelling process resulting in your first simulation.

[Chapter 2: Introduction to D-Flow Flexible Mesh](#), provides specifications of D-Flow FM, such as the areas of application, the standard and specific features provided, coupling to other modules and utilities.

[Chapter 3: Getting started](#), gives an overview of the basic features of the D-Flow FM GUI and will guide you through the main steps to set up a basic D-Flow FM model.

[Chapter 4: Tutorial](#), gives you hands-on experience in using the D-Flow FM GUI to define the input of a simple problem, in validating this input, in executing the simulation and in inspecting the results.

[Chapter 5: All about the modelling process](#), provides practical information on the GUI, setting up a model with all its parameters and tuning the model.

[Chapter 6: Running a model](#), discusses how to validate and execute a model run. Either in the GUI, or in batch mode and/or in parallel using MPI. It also provides some information on run times and file sizes.

[Chapter 7: Visualize results](#), explains in short the visualization of results within the GUI. It introduces the programs Quickplot and Muppet to visualize or animate the simulation results, and Matlab for general post-processing.

[Chapter 8: Hydrodynamics](#), gives some background information on the conceptual model of the D-Flow FM module.

Chapter 9: [Transport of matter](#), discusses the modeled transport processes, their governing equations, boundary and initial conditions and user-relevant numerical and physical settings.

Chapter 10: [3D Modelling](#) provides a detailed insight into the modelling of turbulence. **3D modelling is a β -functionality.**

Chapter 11: [Heat transport](#), provides a detailed insight into (the modelling of) heat transport.

Chapter 12: [Wind](#), gives background information of how wind fields should be imposed, the relevant definitions and the supported file formats.

Chapter 13: [Hydraulic structures](#), gives background information of the available hydraulic structures in D-Flow FM, the relevant definitions and the supported file formats.

Chapter 16: [Coupling with D-Waves \(SWAN\)](#), provides guidance on the integrated modelling of hydrodynamics (D-Flow FM) and waves (D-Waves).

Chapter 17: [Coupling with D-RTC \(FBC-Tools\)](#), provides guidance on the integrated modelling of hydrodynamics (D-Flow FM) and real time control of hydraulic structures (D-RTC).

Chapter 18: [Coupling with D-Water Quality \(Delwaq\)](#), provides guidance on the integrated modelling of hydrodynamics (D-Flow FM) and water quality (D-Water Quality).

Chapter 21: [Calibration and data assimilation](#), describes how the OpenDA toolbox could be deployed to apply calibration and data assimilation.

1.3 Manual version and revisions

This manual applies to:

- ◊ the D-HYDRO Suite, version 2020
- ◊ the Delft3D Flexible Mesh Suite, version 2020
- ◊ SOBEK 3, version 3.7.17 (and higher)

1.4 Typographical conventions

Throughout this manual, the following conventions help you to distinguish between different elements of text.

Example	Description
Module Project	Title of a window or a sub-window are given in bold . Sub-windows are displayed in the Module window and cannot be moved. Windows can be moved independently from the Module window, such as the Visualisation Area window.

Example	Description
Save	<p>Item from a menu, title of a push button or the name of a user interface input field.</p> <p>Upon selecting this item (click or in some cases double click with the left mouse button on it) a related action will be executed; in most cases it will result in displaying some other (sub-)window.</p> <p>In case of an input field you are supposed to enter input data of the required format and in the required domain.</p>
<\tutorial\wave\swan-curvi> <siu.mdw>	<p>Directory names, filenames, and path names are expressed between angle brackets, <>. For the Linux and UNIX environment a forward slash (/) is used instead of the backward slash (\) for PCs.</p>
“27 08 1999”	<p>Data to be typed by you into the input fields are displayed between double quotes.</p> <p>Selections of menu items, option boxes etc. are described as such: for instance ‘select Save and go to the next window’.</p>
delft3d-menu	<p>Commands to be typed by you are given in the font Courier New, 10 points.</p>
	<p>In this User manual, user actions are indicated with this arrow.</p>
[m s ⁻¹] [-]	<p>Units are given between square brackets when used next to the formulae. Leaving them out might result in misinterpretation.</p>

Command prompts and terminal output are shown in framed boxes with typewriter font:

```
> ./dflowfm --version
Deltares, D-Flow FM Version 1.1.149.41663, Sep 02 2015, 10:40:42
Compiled with support for:
IntGUI: no
OpenGL: no
OpenMP: yes
MPI    : yes
PETSc : yes
METIS : yes
```

1.5 Changes with respect to previous versions

Several descriptions of β -functionality are added or marked as β -functionality. Screencasts are updated. [Chapter 4: Tutorial](#) is restructured and a tutorial for Morphology is added.

2 Introduction to D-Flow Flexible Mesh

Note: The 3D modelling is a β -functionality.



D-Flow Flexible Mesh (D-Flow FM) is a hydrodynamic simulation program developed by Deltares. It is part of Deltares' unique, fully integrated computer software suite for a multi-disciplinary approach and 1D, 2D and 3D computations for coastal, river and estuarine areas, named Delft3D Flexible Mesh Suite or D-HYDRO Suite. It can carry out simulations of hydrodynamic flow, waves, water quality and ecology.

It has been designed for experts and non-experts alike. The Delft3D Flexible Mesh Suite is composed of several modules, grouped around a mutual interface, while being capable to interact with one another. D-Flow FM, which this manual is about, is one of these modules. D-Flow FM is a multi-dimensional (1D, 2D and 3D) hydrodynamic (and transport) simulation program which calculates non-steady flow and transport phenomena that result from tidal and meteorological forcing on structured and unstructured, boundary fitted grids. The term Flexible Mesh in the name refers to the flexible combination of unstructured grids consisting of triangles, quadrangles, pentagons and hexagons. In 3D simulations the vertical grid is using the σ co-ordinate approach. As an alternative a fixed z layers approach is also possible. The 2D functionality in D-Flow FM has been released, while the functionality for 3D and 1D is in development.



2.1 Areas of application

- ◊ Tide and wind-driven flows (i.e., storm surges).
- ◊ Stratified and density driven flows.
- ◊ River flow simulations.
- ◊ Rural channel networks.
- ◊ Rainfall runoff in urban environments.
- ◊ Simulation of tsunamis, hydraulic jumps, bores and flood waves.
- ◊ Fresh-water river discharges in bays.
- ◊ Salt intrusion.
- ◊ Cooling water intakes and waste water outlets.
- ◊ Transport of dissolved material and pollutants.

2.2 Standard features

- ◊ Tidal forcing.
- ◊ The effect of the Earth's rotation (Coriolis force).
- ◊ Density driven flows (pressure gradients terms in the momentum equations).
- ◊ Advection-diffusion solver included to compute density gradients.
- ◊ Space and time varying wind and atmospheric pressure.
- ◊ Advanced turbulence models to account for the vertical turbulent viscosity and diffusivity based on the eddy viscosity concept. Four options are provided: 1) constant, 2) algebraic, 3) $k-\varepsilon$ and 4) $k-\tau$ model.
- ◊ Time varying sources and sinks (e.g., river discharges).
- ◊ Simulation of the thermal discharge, effluent discharge and the intake of cooling water at any location and any depth.
- ◊ Robust simulation of drying and flooding of inter-tidal flats and river winter beds.

2.3 Special features

- ◊ Built-in automatic switch converting 2D bottom-stress coefficient to 3D coefficient.
- ◊ Built-in anti-creep correction to suppress artificial vertical diffusion and artificial flow due to σ -grids.
- ◊ Heat exchange through the free water surface.
- ◊ Wave induced stresses and mass fluxes.
- ◊ Influence of waves on the bed shear stress.
- ◊ Optional facility to calculate the intensity of the spiral motion phenomenon in the flow (e.g., in river bends) which is especially important in sedimentation and erosion studies (for depth averaged — 2DH — computations only).
- ◊ Non-linear iterations in the solver can be enabled for accurate flooding results.
- ◊ Optional facility for tidal analysis of output parameters.
- ◊ Optional facility for special structures such as pumping stations, bridges, fixed weirs and controllable barriers (1D, 2D and 3D)
- ◊ Default advection scheme suitable for various flow regimes, from bore propagation to eddy shedding.
- ◊ Domain partitioning for parallelized runs on MPI-based High Performance Computing clusters.

2.4 Important differences compared to Delft3D-FLOW

The most noticeable difference between Delft3D-FLOW and D-Flow FM is the use of unstructured grids. Large regions with quadrangles can be coupled with much greater freedom than before, using triangles, pentagons and hexagons. Grid refinement (and coarsening) without DD-coupling is now possible in one and the same model grid. In future, 1D networks will be coupled to 2D grids, either adjacent to each other or the 1D network overlying the 2D grid. Finally, many of Delft3D-FLOW's grid restrictions are now gone: since there are no true grid 'rows' and 'columns' anymore, rows of grid cells may be coupled to columns, in any direction and at any position.

In addition to the unstructured grid files, all geometric model input is now specified in geographical coordinates, either in Cartesian or spherical coordinates (x , y or longitude, latitude). This is different from Delft3D-FLOW which required model input in grid indices. This so-called model-independent coordinates input allows for easy change of a model grid, after which the remaining model input can remain the same.

The new Delta Shell graphical user interface provides a much more powerful and integrated environment for setting up D-Flow FM models and inspecting model input such as time-dependent forcings (boundary conditions and barrier control). Another improvement within Delta Shell is the use of scripting for running and live interaction with a model.

Coupled running of D-Flow FM with other modules has been extended with real time control of hydraulic structures, as listed in the following section.

Like Delft3D-FLOW, D-Flow FM implements a finite volume solver on a staggered grid. However, since there is no concept of grid 'rows' and 'columns', there is also no ADI-solver possible. The continuity equation is solved implicitly for all points in a single combined system. Time integration is done explicitly for part of the advection term, and the resulting dynamic time-step limitation is automatically set based on the Courant criterium. The possible performance penalty that may result from this approach can often be remedied by refining and coarsening the computational grid at the right locations.

In D-Flow FM, the advection scheme is suitable for both sub-critical and critical flows. The scheme is 'shock proof', is capable of reproducing correct bore propagation velocities.

2.5 Coupling to other modules

The hydrodynamic conditions (velocities, water elevations, density, salinity, vertical eddy viscosity and vertical eddy diffusivity) calculated in the D-Flow FM module are used as input to the other modules of the Delft3D Flexible Mesh Suite, which are:

Module	Description
D-Waves (SWAN)	short wave propagation, see also chapter 16
D-Water Quality (Delwaq)	far-field water quality, see also chapter 18
D-Real Time Control	flow-triggered control of hydrodynamic structures

Module couplings that are not yet available are summarized below:

Module	Description
D-WAQ PART	mid-field water quality and particle tracking
Delft3D-SED	cohesive and non-cohesive sediment transport

For using D-Flow FM the following utilities are important:

Module	Description
Delta Shell	for complete model set-up and model runs, see chapter 3 and chapter 5
RGFGRID	for generating curvilinear and unstructured grids
Delft3D-QUICKPLOT	for visualisation and animation of simulation results
OpenEarthTools	set of MATLAB scripts for post-processing of output files, see http://www.openearth.eu
DFMOUTPUT	<ul style="list-style-type: none"> ◊ for merging partitioned map files into one, see Section 6.2.5. ◊ output of derived properties, such as the maximum value of a time series, see section F.5.

For details on using these utility programs you are referred to the respective User Manuals.

2.6 Installation

Separate installation are provided for Delta Shell and the computational core.

2.6.1 Installation of Delta Shell

Delta Shell is only available for Windows operating systems. You can either install the msi-version or copy the zip-version. For the msi-version first follow the steps in the installation program. Consequently, start the application from *Start → All Programs → Deltares* or by double-clicking the short-cut on your desktop. For the zip-version you don't have to install anything. First unpack the zip, consequently go to bin and double-click <DeltaShell.Gui.x64.exe> to start the application.

2.6.2 Installation of the computational core

For the installation of the computational core a separate Installation Manual is provided.

2.7 Examples

An extensive set of example models is available as part of this D-Flow FM release. Throughout this User Manual, references to these testcases are made via the directory names as follows:

- ◊ <f05_boundary_conditions/c019_waterlevel_bc_cmp_varying/>
- ◊ <f17_sources_sinks/c020_sourcesink_3D/>

3 Getting started

3.1 Introduction

The D-Flow FM plugin is part of the Delta Shell framework. For an introduction to the general look-and-feel and functionalities of the Delta Shell framework you are referred to the Delta Shell User Manual. This chapter gives an overview of the basic features of the D-Flow FM plugin and will guide you through the main steps to set up a D-Flow FM model. For a more detailed description of the GUI features you are referred to [chapter 5](#). For technical documentation you are referred to [D-Flow FM TRM \(2015\)](#).

3.2 Overview of D-Flow FM GUI

When you start the application for the first time the lay-out might look like [Figure 3.1](#). The basic lay-out consists of the following items:

- ◊ **Ribbon - top**
- ◊ **Project** window - up left
- ◊ The central window, containing the *Start Page*
- ◊ **Messages and Time Navigator** window - down centre
- ◊ **Toolbox, Chart, Region, Map and Operations** window - to the right
- ◊ **Properties** window - down left

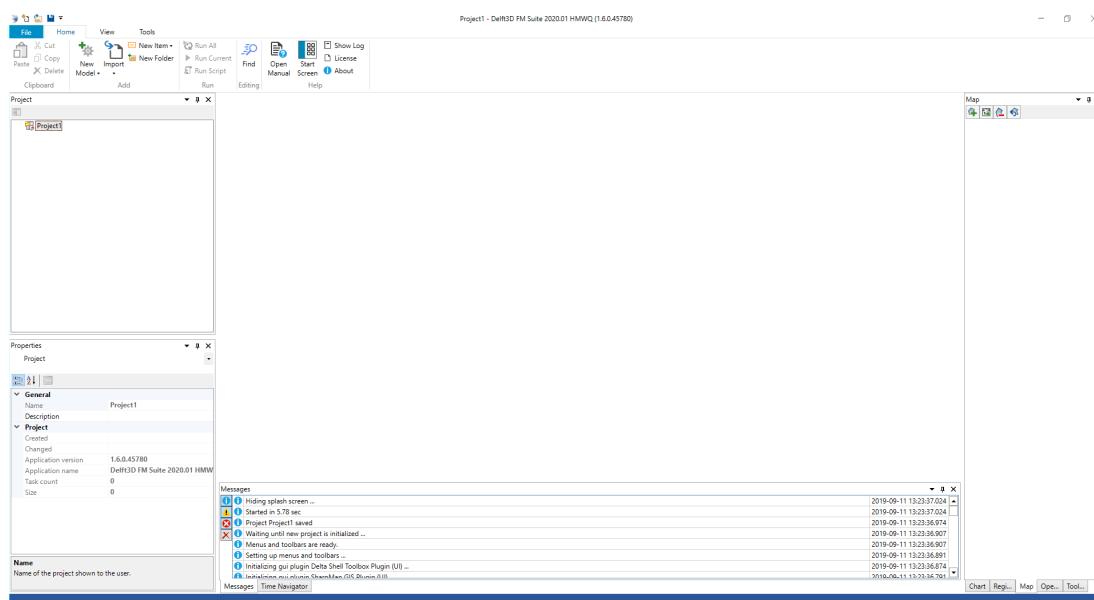


Figure 3.1: Start-up lay-out Delta Shell

All the windows can be customized/hidden according to your own preferences.

These settings will be automatically saved for the next time you start the application.

The most important windows for the D-Flow FM plugin are the **Project**, **Map**, **Messages**, **Time Navigator** and **Properties** windows. The central window displays the **Start Page**. Here the *Map view* or specific editors will be displayed.

The contents of these windows are briefly discussed in the subsections below.

3.2.1 Project window

After adding or importing a D-Flow FM model (see [section 3.5.1](#) and [section 3.5.2](#)), the *Project* window will be extended with D-Flow FM specific features (see [Figure 3.2](#)). The *Project* window provides you with the basic steps to set up a D-Flow FM model.

The *Project* window consists of the following features:

<i>General</i>	general model information such as depth layer specification, model coordinate system and angle of latitude and longitude
<i>Area</i>	geographical (GIS based) features, such as observation points, structures, dry points and land boundaries
<i>Grid</i>	computational grid
<i>Bed Level</i>	model bed level
<i>Time Frame</i>	model time frame and time step
<i>Processes</i>	active physical processes in the model such as salinity, temperature, wind and tide generating forces
<i>Initial Conditions</i>	initial conditions for water levels and other physical processes
<i>Boundary Conditions</i>	model boundaries and boundary condition specification
<i>Physical Parameters</i>	physical settings for processes such as roughness, viscosity, wind and temperature
<i>Sources and Sinks</i>	location and time series specification for point sources and sinks
<i>Numerical Parameters</i>	numerical simulation settings
<i>Output Parameters</i>	output specification
<i>Output</i>	output after running the simulation

Upon clicking the items in the *Project* window the corresponding tab (in case of non-geographic model settings), attribute table (in case of geographic model settings) or editor view (in case of advanced editing options) will open in the central window. Using the right mouse button gives options such as importing/exporting model data.

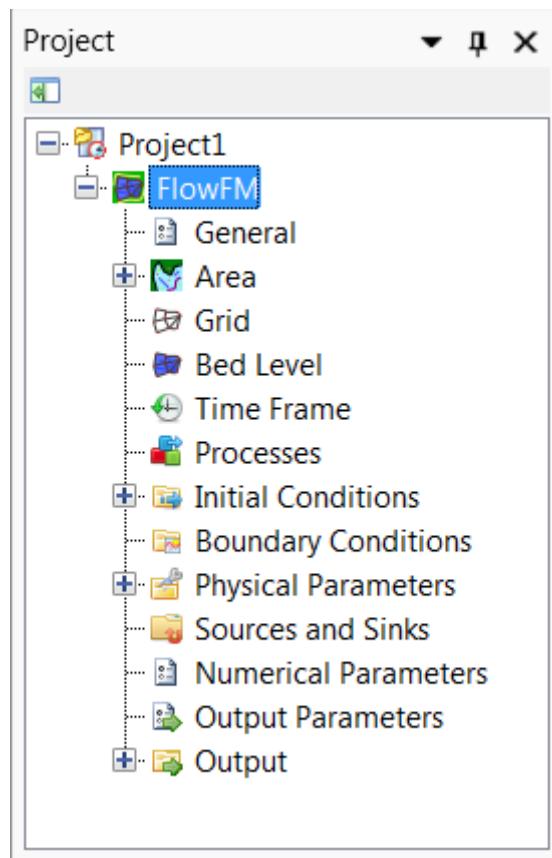


Figure 3.2: Project window of D-Flow FM plugin

3.2.2 The central window

The central window shows the contents of the map or the editor you are working with. Multiple maps and editors may be docked at the central window location; each accessible as a tabs along the top (see [Figure 3.3](#)). The map is used to edit geographic model data, the editor for the overall model settings. Moreover, the contents of the central window can also be a specific editor such as the time point editor or the boundary condition editor. Each of these editors will open as a separate tab.

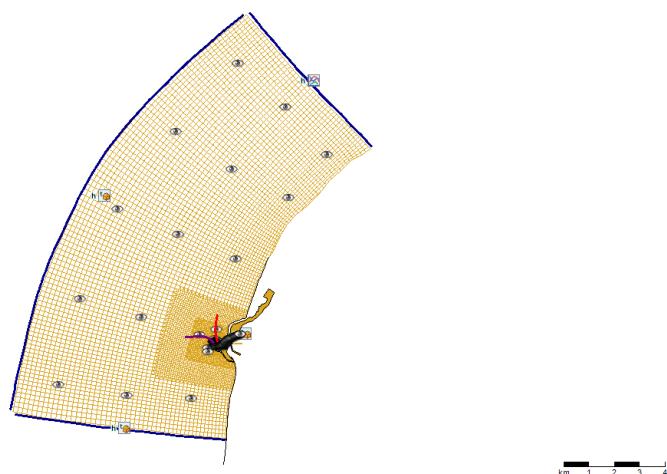


Figure 3.3: The central window with a map and contents of the D-Flow FM model

3.2.3 Settings window

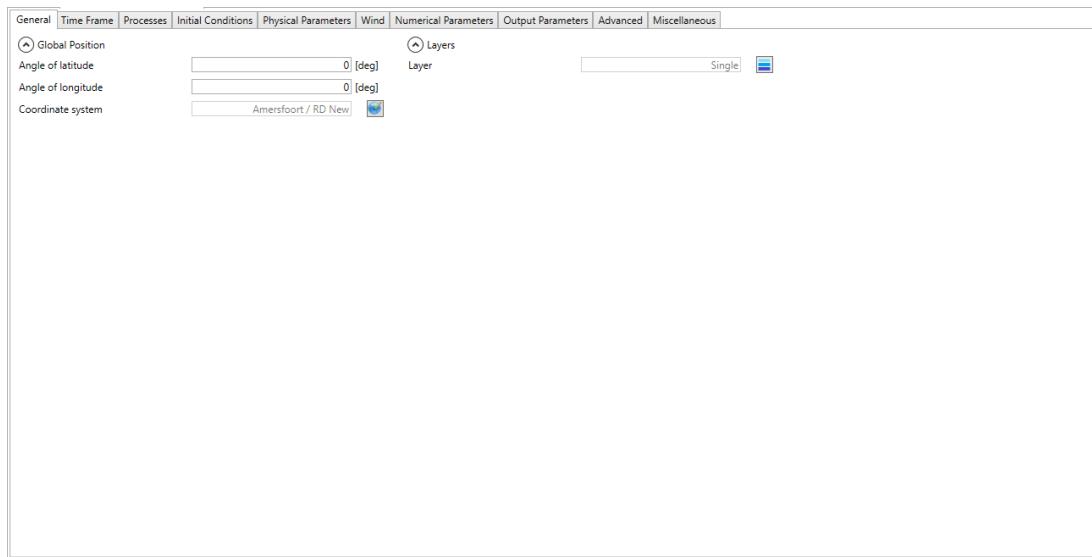


Figure 3.4: The model **Settings** window with General settings tab enabled

3.2.4 Map window

The **Map** window allows the user to control the visibility of the contents of the central map using checkboxes. Furthermore, the user can add (wms) layers, such as satellite imagery or open street maps (see [Figure 3.5](#)).



Note: Please note that the map usually has a different coordinate system than the model. In rendering the model attributes they are transformed to the map coordinate system (for visual inspection on the map), but the model will be saved in the model coordinate system.

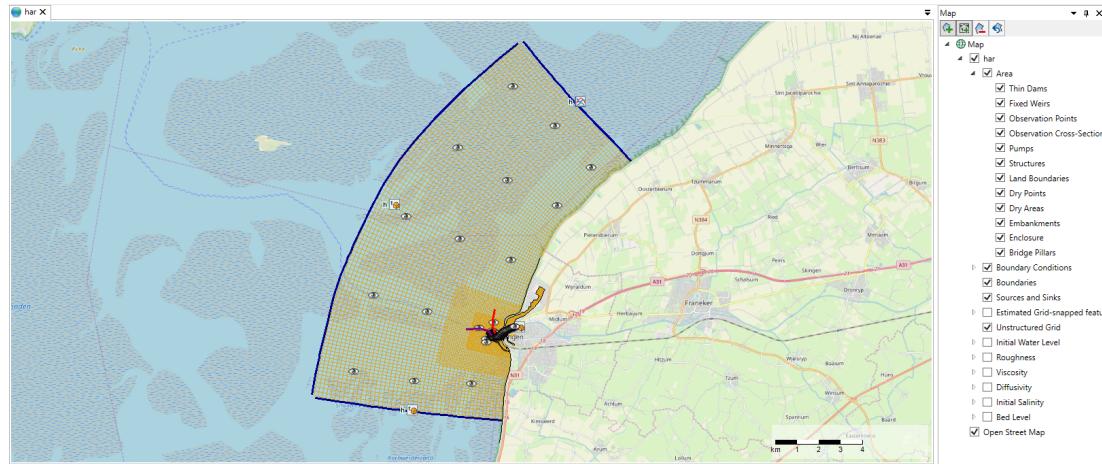


Figure 3.5: Map window controlling map contents

3.2.5 Messages window

The **Message** window (Figure 3.6) provides a log of information on the recent activities in Delta Shell. It also provides warning and error messages.



Figure 3.6: Log of messages, warnings and errors in message window

3.2.6 Time Navigator window

The **Time Navigator** (Figure 3.7) can be used to step through time dependent model output and other time dependent geographic features on the map.

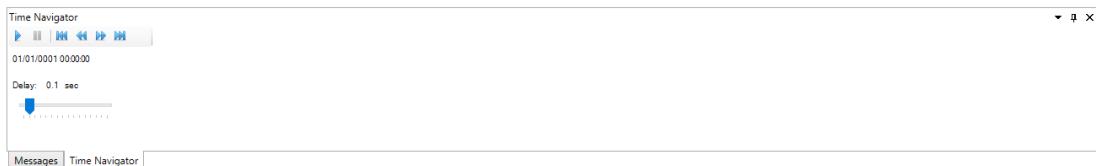


Figure 3.7: Time Navigator window in Delta Shell

3.3 Dockable views

The Delta Shell framework offers lots of freedom to customize dockable views, which are discussed in this section.

3.3.1 Docking tabs separately

Within the Delta Shell framework the user can dock the separate windows according to personal preferences. These preferences are then saved for future use of the framework. An example of such preferences is presented in Figure 3.8, where windows have been docked on two screens.

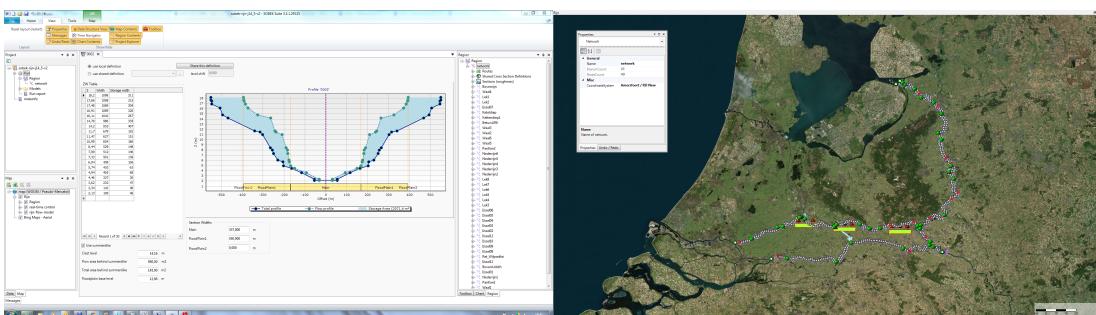


Figure 3.8: Docking windows on two screens within the Delta Shell framework.

3.3.2 Multiple tabs

In case two windows are docked in one view, the underlying window (tab) can be brought to the front by simply selecting the tab, as is shown here.

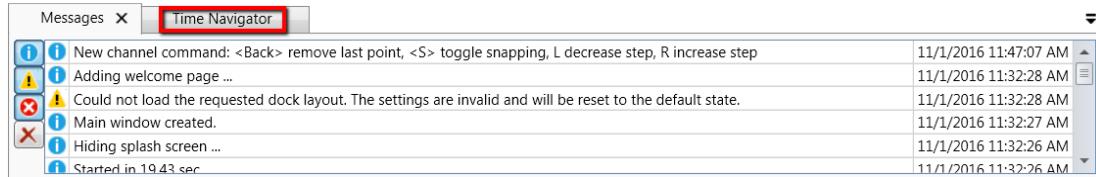


Figure 3.9: Bringing the **Time Navigator** window to the front

By dragging dockable windows with the left mouse button and dropping the window left, right, above or below another one the graphical user interface can be customized according to personal preferences. Here an example of the **Time Navigator** window being docked to the right of the **Messages** window.

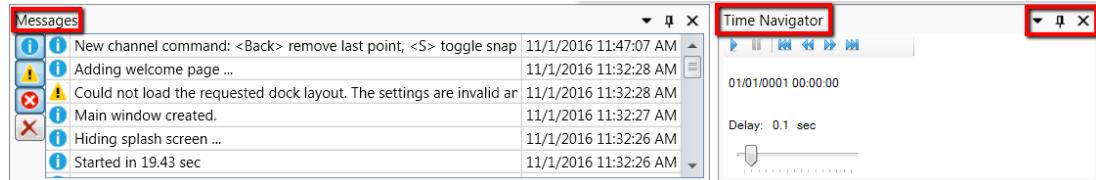


Figure 3.10: Docking the **Time Navigator** window.

Additional features are the possibility to remove or (auto) hide the window (top right in Figure 3.10). In case of removal, the window can be retrieved by two left mouse-clicks on *Time Navigator* in the *View* ribbon. Hiding the **Properties** window results in:

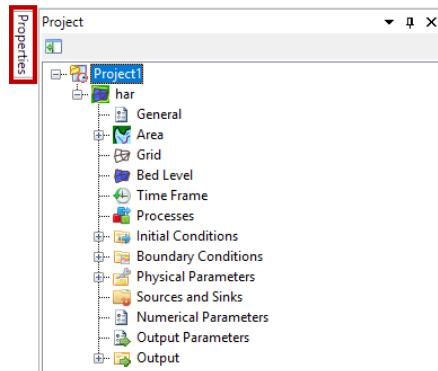


Figure 3.11: Auto hide the **Properties** window

3.4 Ribbons and toolbars

The user can access the toolbars arranged in *ribbons*. Model plug-ins can have their own model specific *ribbon*. The *ribbon* may be auto collapsed by activating the *Collapse the Ribbon* button when right-mouse-clicking on the *ribbon*.

3.4.1 Ribbons (shortcut keys)

Delta Shell makes use of ribbons, just like Microsoft Office. You can use these ribbons for most of the operations. With the ribbons comes shortcut key functionality, providing shortcuts to perform operations. If you press Alt, you will see the letters and numbers to access the ribbons and the ribbon contents (i.e. operations). For example, Alt + H will lead you to the Home-ribbon (Figure 3.12).

Note: Implementation of the shortcut key functionality is still work in progress.

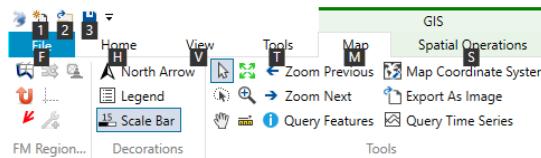


Figure 3.12: Perform operations using the shortcut keys

3.4.2 File

The left-most *ribbon* is the *File* ribbon. It has menu-items comparable to most Microsoft applications. Furthermore, it offers users save and open functionality, as well as the *Info* and *Options* dialogs, as shown in Figure 3.13 and Figure 3.14.

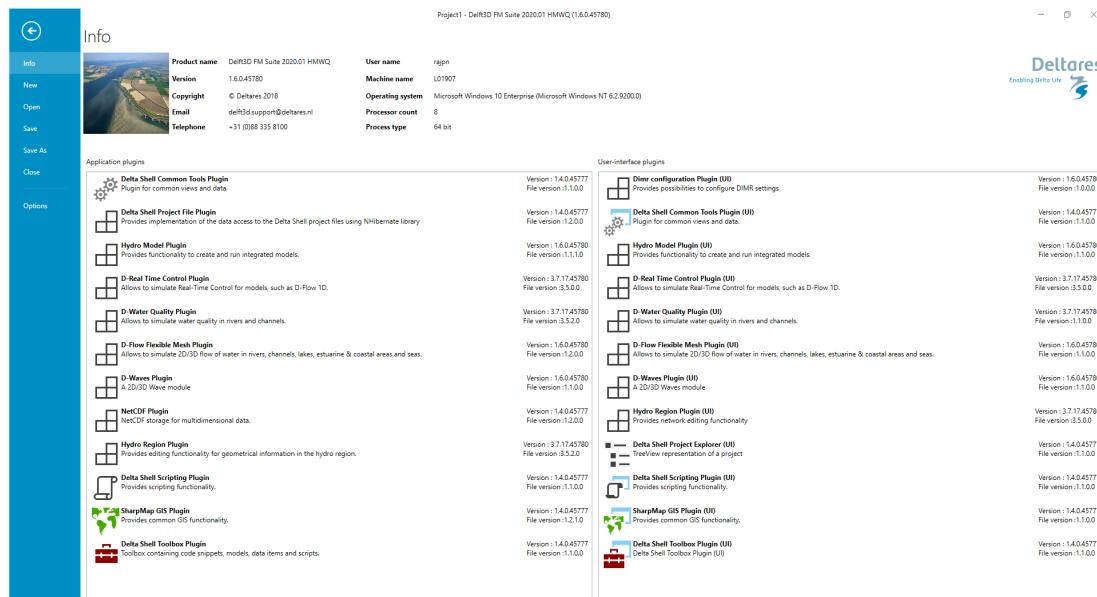


Figure 3.13: The File ribbon.

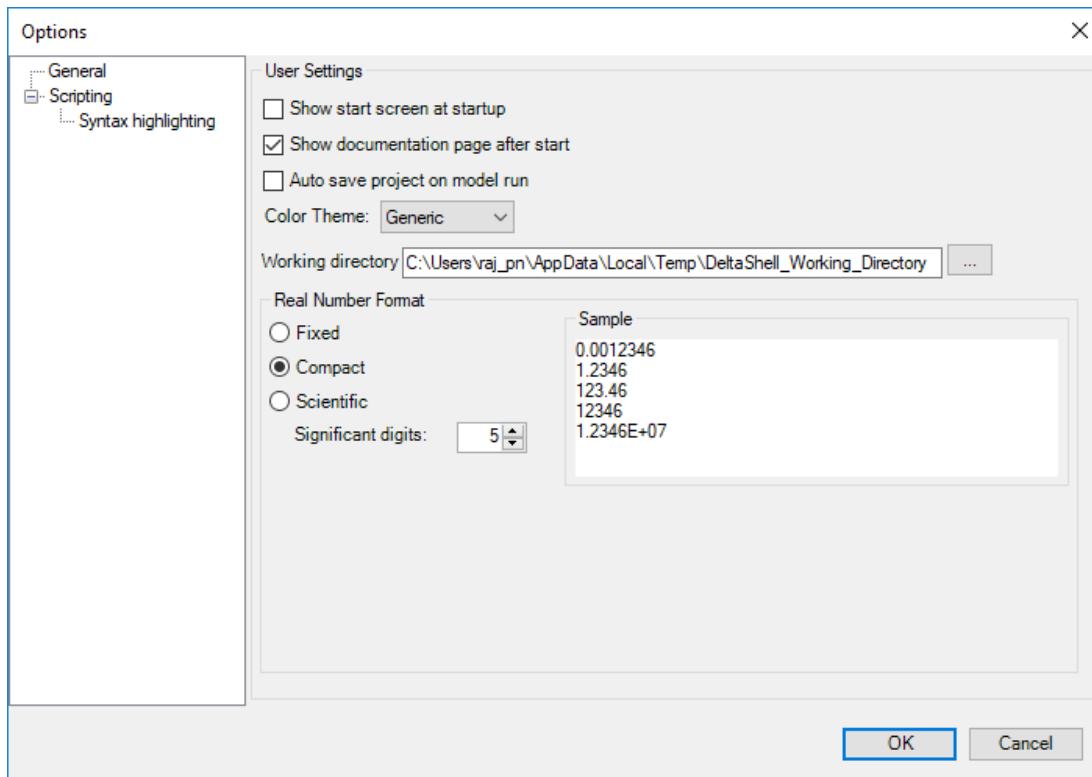


Figure 3.14: The Delta Shell options dialog.

3.4.3 Home

The second *ribbon* is the *Home* ribbon (Figure 3.15). It harbours some general features for clipboard actions, addition of items, running models, finding items within projects or views, and help functionality.

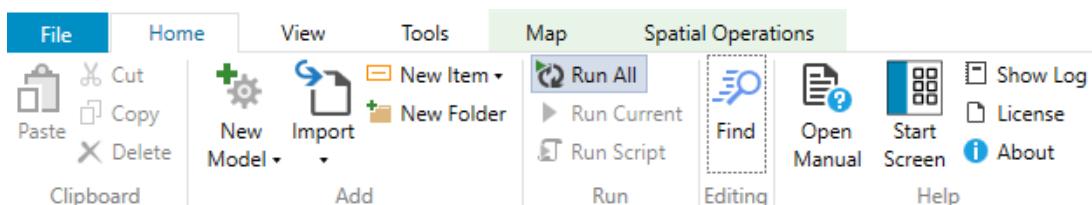


Figure 3.15: The Home ribbon.

3.4.4 View

The third *ribbon* is the *View* ribbon (Figure 3.16). Here, the user can show or hide windows.

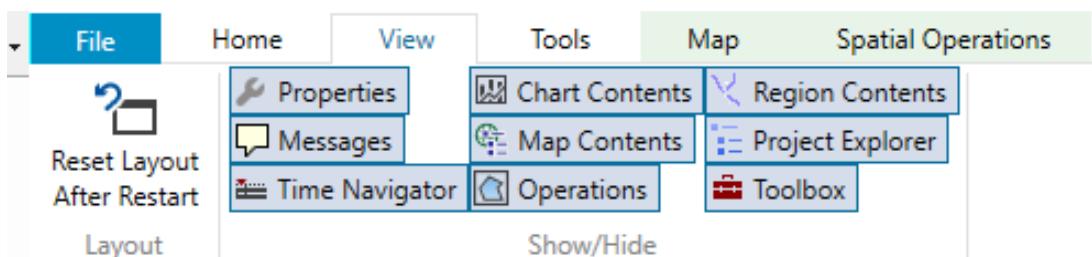


Figure 3.16: The View ribbon.

3.4.5 Tools

The fourth *ribbon* is the *Tools* ribbon (Figure 3.17). By default, it contains only the *Open Case Analysis View* tool. Some model plug-ins offer the installation of extra tools that may be installed. These are documented within the user documentation of those model plug-ins.

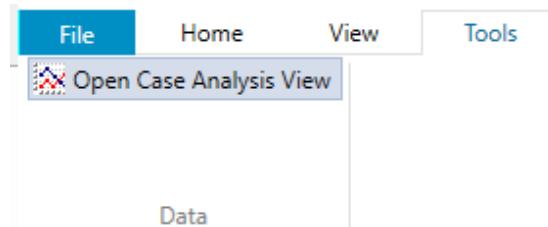


Figure 3.17: The Tools ribbon contains just the Data item.

3.4.6 Map

The last *ribbon* is the *Map* ribbon (Figure 3.18).



Figure 3.18: The Map ribbon.

This will be used heavily, while it harbours all *Geospatial* functions, like:

- ◊ *Decorations* for the map
 - North arrow
 - Scale bar
 - Legend
 - ...
- ◊ *Tools* to customize the map view
 - Select a single item
 - Select multiple items by drawing a curve
 - Pan
 - Zoom to Extents
 - Zoom by drawing a rectangle
 - Zoom to Measure distance
 - ...
- ◊ *Edit* polygons, for example within a network, basin, or waterbody
 - Move geometry point(s)
 - Add geometry point(s)
 - Remove geometry point(s)
- ◊ Addition of *Area* elements, such as
 - Add 2D Cross-section
 - Add 2D Structure
 - Add 2D Pump
 - ...

Still, all functions of the category can be activated as they will appear in the drop-down panel.

3.4.7 Scripting

When you open the scripting editor in Delta Shell, a Scripting *ribbon* category will appear. This ribbon has the following additional options (see also Figure 3.19), which are described in Table 3.1:



Figure 3.19: The scripting ribbon within Delta Shell.

Table 3.1: Functions and their descriptions within the scripting ribbon of Delta Shell.

Function	Description
Run script	Executes the selected text. If no text is selected then it will execute the entire script
Clear cached variables	Clears all variables and loaded libraries from memory
Debugging	Enables/Disables the debug option. When enabled you can add breakpoint to the code (using F9 or clicking in the margin) and the code will stop at this point before executing the statement (use F10 (step over) or F11 (step into) for a more step by step approach)
Python variables	Show or hide python variables (like <code>_var_</code>) in code completion
Insert spaces/tabs	Determines if spaces or tab characters are added when pressing tab
Tab size	Sets the number of spaces that are considered equal to a tab character
Save before run	Saves the changes to the file before running
Create region	Creates a new region surrounding the selected text
Comment selection	Comments out the selected text
Convert to space indenting	Converts all tab characters in the script to spaces. The number of spaces is determined by Tab size
Convert to tab indenting	Converts all x number of space characters (determined by Tab size) in the script to tabs
Python (documentation)	Opens a link to the python website, showing you the python syntax and standard libraries
Delta Shell (documentation)	Opens a link to the Delta Shell documentation website (generated documentation of the Delta Shell api)

Table 3.2: Shortcut keys within the scripting editor of Delta Shell.

Shortcut	Function
----------	----------

3.4.8 Shortcuts

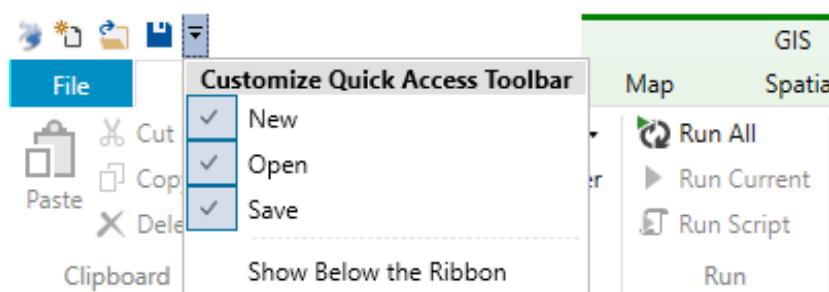
The shortcut keys of the scripting editor within Delta Shell are documented in Table 3.2.

Table 3.2: Shortcut keys within the scripting editor of Delta Shell.

Shortcut	Function
Ctrl + Enter	Run selection (or entire script with no selection)
Ctrl + Shift + Enter	Run current region (region where the cursor is in)
Ctrl + X	Cut selection
Ctrl + C	Copy selection
Ctrl + V	Paste selection
Ctrl + S	Save script
Ctrl + -	Collapse all regions
Ctrl + +	Expand all regions
Ctrl + "	Comment or Uncomment current selection
Ctrl + W	Add selection as watch variable
Ctrl + H	Highlight current selection in script (press esc to cancel)
F9	Add/remove breakpoint (In debug mode only)
F5	Continue running (In debug mode only — when on breakpoint)
Shift + F5	Stop running (In debug mode only — when on breakpoint)
F10	Step over current line and break on next line (In debug mode only - when on breakpoint)
F11	Step into current line if possible, otherwise go to next line (In debug mode only — when on breakpoint). This is used to debug functions declared in the same script (that have already runned)

3.4.9 Quick access toolbar

Note: The user can make frequently used functions available by a single mouse-click in the *Quick Access Toolbar*, the top-most part of the application-window. Do this by right-mouse-clicking a ribbon item and selecting *Add to Quick Access Toolbar*.

**Figure 3.20:** The quick access toolbar.

3.5 Important differences compared to Delft3D-FLOW GUI

The differences between the former Delft3D-FLOW GUI and the D-Flow FM plugin in Delta Shell in lay-out and functionality are numerous. Here, we address only the most important differences in the workflow.

3.5.1 Project vs model

The entity “project” is new in the Delta Shell GUI. In the hierarchy the entity “project” is on a higher level than the entity “model”. A project can contain multiple models, which can either run standalone or coupled. The user can run all models in the project at once (on project level) or each model separately (on model level). When the user saves the project, the project settings will be saved in a `<*.dsproj>` configuration file and the project data in a `<*.dsproj_data>` folder. The `<*.dsproj_data>` folder contains folders with all input and output files for the models within the project. There is no model intelligence in the `<*.dsproj>` configuration file, meaning that the models can also be run outside the GUI from the `<*.dsproj_data>` folder.

3.5.2 Load/save vs import/export

The user can load an existing Delta Shell project, make changes in the GUI and, consequently, save all the project data. Loading and saving means working on the original project data, i.e. the changes made by the user overwrite the original project data. Alternatively, use *Save As* to keep the original project data and save the changes project data at another location (or with another name).

Import/export functionality can be used to copy data from another location into the project (import) or, vice versa, to copy data from the project to another location (export). Import/export is literally copying, e.g.:

- ◊ import: changes on the imported data will only affect the data in the project and not the source data (upon saving the project)
- ◊ export: the model data is copied to another location “as is”, changes made afterwards will only affect the data in the project not the exported data (upon saving the project)

3.5.3 Working from the map

One of the most important differences with the former GUI is the central map. The central map is comparable with the former “visualization area”, but with much more functionality and flexibility. The map helps you to see what you are doing and inspect the model at all times. You can use the *Region* and *Map* ribbons to add/edit model features in the map.

3.5.4 Coordinate conversion

With the map as a central feature, functionality to convert model and map coordinates is an indispensable feature. In the *General* tab you can set the model coordinate system. In the map tree you can set the map coordinate system using the right mouse button ([Figure 3.21](#)). The coordinate systems are subdivided in geographic and projected systems. Use the quick search bar to find the coordinate system you need either by name or EPSG code ([Figure 3.22](#)).

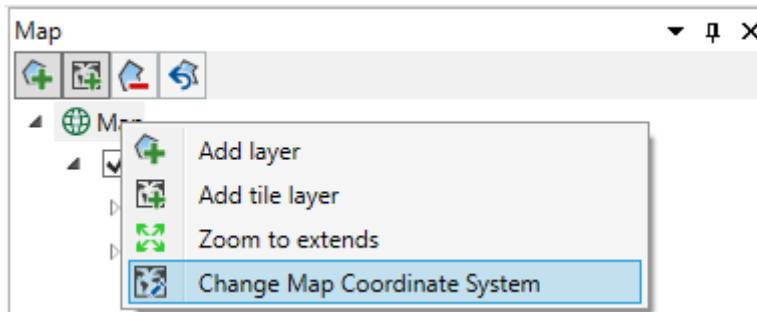


Figure 3.21: Set map coordinate system using right mouse button

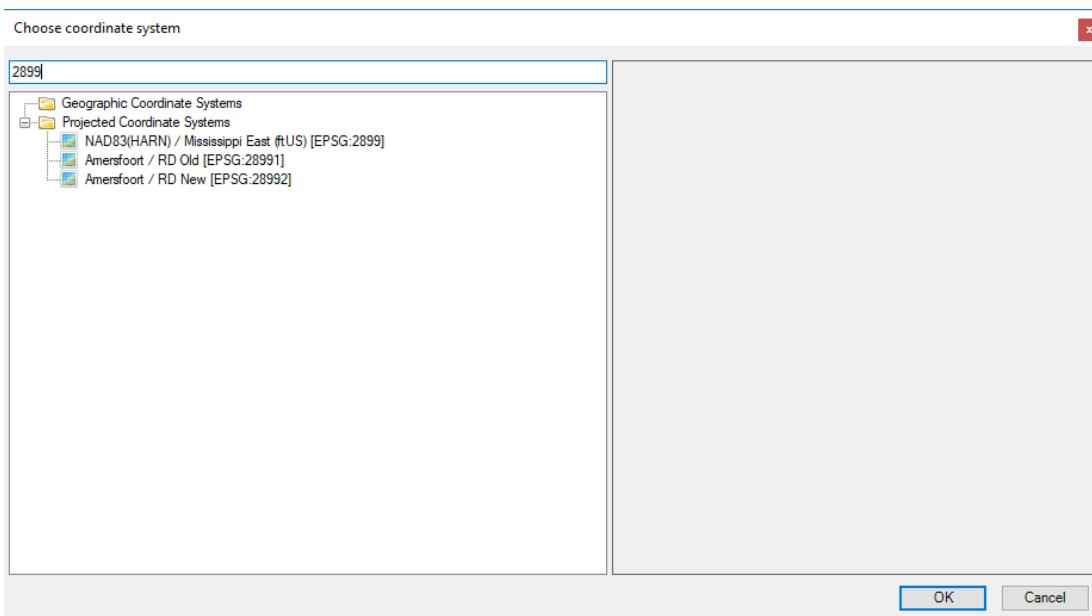


Figure 3.22: Select a coordinate system using the quick search bar

3.5.5 Model area

The model area contains geographical features, such as observation points & curves and obstacles. In contrast to the former GUI, these features can even exist without a grid or outside the grid and they are not based on grid coordinates, implying that their location remains the same when the grid is changed (for example by (de-)refining).

Finally, for the computations, the SWAN computational core interpolates the features to the grid. In the future we would like to show to which grid points the features are snapped before running the computation. However, this requires some updates in the SWAN computational core.

3.5.6 Integrated models (model couplings)

The Delta Shell framework implements the concept of an *Integrated model* in order to couple different models, such as: hydrodynamics coupled with the controlling of structures, waves, morphology and/or water quality.

Two types of coupling are distinguished: *offline* and *online* coupling. In case of an *Integrated model* with *offline* coupling, the entire hydrodynamic simulation is done first, i.e., *separately*

from the second simulation. The file-based hydrodynamic output serves as input for the second simulation. As such, the hydrodynamic flow drives the controlling of structures or the simulation of waves or water quality. In this case there is no feedback on the hydrodynamic simulation. For many applications, this is good practice.

An *online* coupling, on the other hand, exchanges data *every time* after computing a specified time interval. This tight coupling allows for direct feedback of the various processes on one another. This is crucial for controlling structures.



Note: *Offline* is also referred to as *sequential* coupling and *online* as *parallel* coupling.

3.5.7 Ribbons (shortcut keys)

Delta Shell makes use of ribbons, just like Microsoft Office. You can use these ribbons for most of the operations. With the ribbons comes shortcut key functionality, providing shortcuts to perform operations. If you press Alt, you will see the letters and numbers to access the ribbons and the ribbon contents (i.e. operations). For example, Alt + H will lead you to the Home-ribbon (Figure 3.23).



Note: Implementation of the shortcut key functionality is still work in progress.

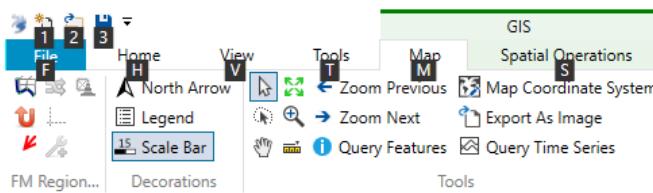


Figure 3.23: Perform operations using the shortcut keys

3.5.8 Context menus

Context menus are the menus that pop up using the right mouse button. These context menus provide you with some handy functionality and shortcuts specific for the selected item. The functionality is available in all Delta Shell windows and context dependent. You can best try it yourself to explore the possibilities.

3.5.9 Scripting

Delta Shell has a direct link with scripting in Iron Python (NB: this is not the same as C-Python). This means that you can get and set data, views and model files by means of scripting instead of having to do it all manually. Scripting can be a very powerful tool to automate certain steps of your model setup or to add new functionality to the GUI. You can add a new script by adding a new item, either in the *Home*-ribbon or through the right mouse button.

4 Tutorial

This chapter includes a number of basic tutorials on setting up and running a hydrodynamic simulation (see section 4.2). However, before you start with the tutorials, it's important to learn some **basic grid concepts**.

Note: in this chapter, read for Delta Shell either D-HYDRO Suite or Delft3D Flexible Mesh Suite.

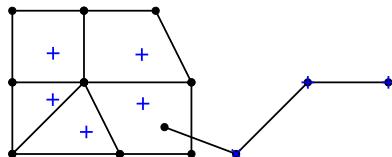


4.1 Introduction: Basic grid concepts

In D-Flow FM, grids (sometimes denoted as 'networks') consist of net cells and are described by **net nodes** (corners of a cell), **net links** (edges of a cell, connecting net nodes), **flow nodes** (the cell circumcentre) and **flow links** (a line segment connecting two flow nodes).

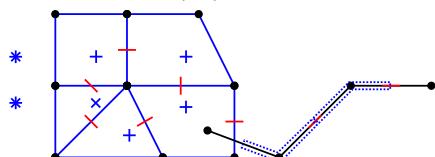
This grid topology is illustrated in Figure 4.1. Important properties of the mesh are the *orthogonality* and *smoothness*. The *orthogonality* is defined as the cosine of the angle φ between a flowlink and a netlink. Ideally 0, angle $\varphi = 90^\circ$. The *smoothness* of a mesh is defined as the ratio of the areas of two adjacent cells. Ideally 1, the areas of the cells are equal to each other. A nearly ideal setup is shown in Figure 4.2.

1. Net (domain discretization)



- net node (1..NUMK)
- ◆ net link (1D) (1..NUML1D)
- net link (2D) (NUML1D+1..NUML)

2. Flow data (2D)



- netcell/flow node (2D) (1..NDX2D=NUMP)
- ◆ netcell/flow node (1D) (NDX2D+1..NDXI)
- * boundary flow node (NDXI+1..NDX)
- + netcell circumcenter (1..LNX1D)
- netcell link (LNX1D..LNXI)
- (LNXI+1..LNX)

Figure 4.1: Topology and definitions for a grid as used in D-Flow FM.



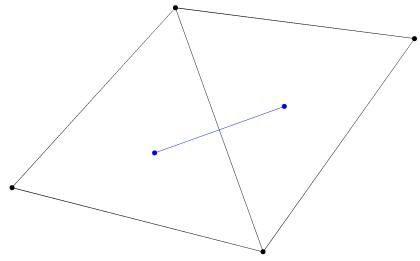
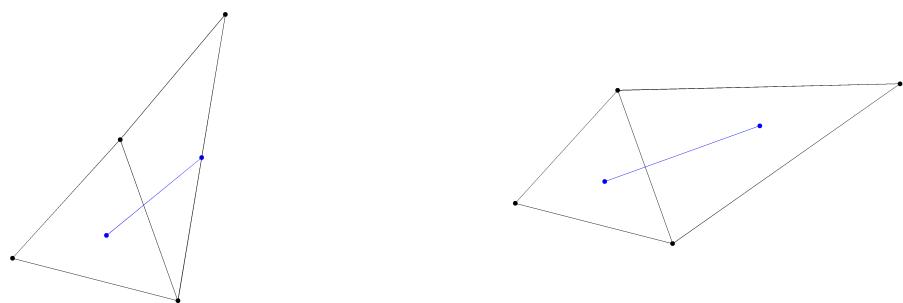


Figure 4.2: Perfect orthogonality and nearly perfect smoothness along the edge connecting two triangles. Black lines/dots are network links/nodes, blue lines/dots are flow links/nodes.

It is rather easy to generate grids that violate the orthogonality and smoothness requirements. In Figure 4.3, two different setups of two grid cells are shown with different grid properties. Figure 4.3a shows how orthogonality can be deteriorated by skewing the right triangle with respect to the left triangle. While having the same area (perfect smoothness), the mutually oblique orientation results in poor orthogonality. In this particular case the circumcentre of the right triangle is outside the area of the triangle because it is an obtuse triangle, which is bad for computations with D-Flow FM. The opposite is shown in Figure 4.3b in which the right triangle has strongly been elongated, disturbing the smoothness property. However, the orthogonality is perfect (both triangles are acute). Nonetheless, both grids need to be improved to assure accurate model results.



(a) Perfect smoothness, but poor orthogonality.

(b) Perfect orthogonality, but poor smoothness

Figure 4.3: Poor grid properties due to the violation of either smoothness or orthogonality at the edge connecting two triangles.
Black lines/dots are network links/nodes,
blue lines/dots are flow links/nodes.

4.2 Tutorial Hydrodynamics

4.2.1 Outline of the tutorials

In this three-hours tutorial you will experience the basic functionality of D-Flow FM. On the one hand you will learn how to capture a certain area with a computational grid. On the other hand you will set up and run a computation yourself. This involves for example inserting a bed level, imposing boundary conditions, running a D-Flow FM model and postprocessing with a D-Flow FM output file. This manual contains nine tutorials (with indication of the estimated time):

- ◊ Tutorial 1 ([section 4.2.2](#)): Creating a curvilinear grid (25 minutes).
- ◊ Tutorial 2 ([section 4.2.3](#)): Creating a triangular grid (20 minutes).
- ◊ Tutorial 3 ([section 4.2.4](#)): Coupling multiple distinct grids (15 minutes).
- ◊ Tutorial 4 ([section 4.2.5](#)): Inserting a bed level (15 minutes).
- ◊ Tutorial 5 ([section 4.2.6](#)): Imposing boundary conditions (25 minutes).
- ◊ Tutorial 6 ([section 4.2.7](#)): Defining output locations (15 minutes).
- ◊ Tutorial 7 ([section 4.2.8](#)): Defining computational parameters (10 minutes).
- ◊ Tutorial 8 ([section 4.2.9](#)): Running a model simulation (20 minutes).
- ◊ Tutorial 9 ([section 4.2.10](#)): Viewing the output of a model simulation (15 minutes).

The necessary input files for a tutorial can be found in the folders *tutorial01* through *tutorial09*. When saving your model data while working through these tutorials, be sure not to overwrite the tutorial data. Ideally, you should create a separate working folder somewhere else on your computer to save your work. The folder *finished* in each tutorial folder contains the output generated by Deltares as a reference.

We will use the Western Scheldt and the harbour of Antwerp as an example case for generating a grid and for setting up and running a computation.

4.2.2 Tutorial 1: Creating a curvilinear grid

Generating grids is done with the RGFGRID tool. Because this is an iterative process and the Undo functionality in RGFGRID is still rather limited, it is important to frequently save intermediate RGFGRID results. By clicking on Esc it is sometimes possible to undo the last action. RGFGRID is a separate tool that communicates with the DeltaShell GUI through a temporary folder whose path is shown in the top menu bar. In order to transfer the generated grid to the DeltaShell GUI, be sure to save your data to this path when leaving RGFGRID.

To start RGFGRID in the Delta Shell Graphical User Interface the following steps have to be carried out:

- Double click on the Delta Shell icon on the desktop of your computer.
- Click on *New Model* in the main toolbar at the top of the Delta Shell GUI.
- A new window pops up in which you have to select *Flow Flexible Mesh Model* → *OK*.
- Double click on *Grid*, see [Figure 4.4](#). RGFGRID will now start.

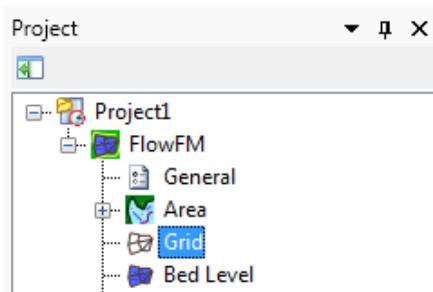


Figure 4.4: Start RGFGRID by a double-click on Grid.

- Choose from the main menubar *Coordinate System* → *Cartesian Coordinates*. This is the default setting, which means that you do not have to change anything.

A curvilinear grid is iteratively generated by first drawing splines. The approach is as follows:

- Import a land boundary via *File* → *Attribute Files* → *Open Land Boundaries* and select the file <tutorial01\scheldtriver.lbd>
- Select *Edit* → *Splines* → *New* and draw a spline roughly following the left boundary of the river (use the leftmousebutton). Finalise the spline by one rightmouseclick. Now we want to bring (attach) the spline close to the boundary.
- Choose the option *Edit* → *Splines* → *Attach to Land Boundary*. Select the two outer points delimiting the spline to be snapped. The vertices which will be snapped are now marked by a square. Press the rightmousebutton to perform the snapping.
- A window appears for extra snapping: Press *Yes* several times. You will see the spline evolve towards the land boundary. Press *No* when you are satisfied with the result.
- Repeat the previous three steps for the righthandside spline.
- Draw a third spline along the river axis; see Figure 4.5.

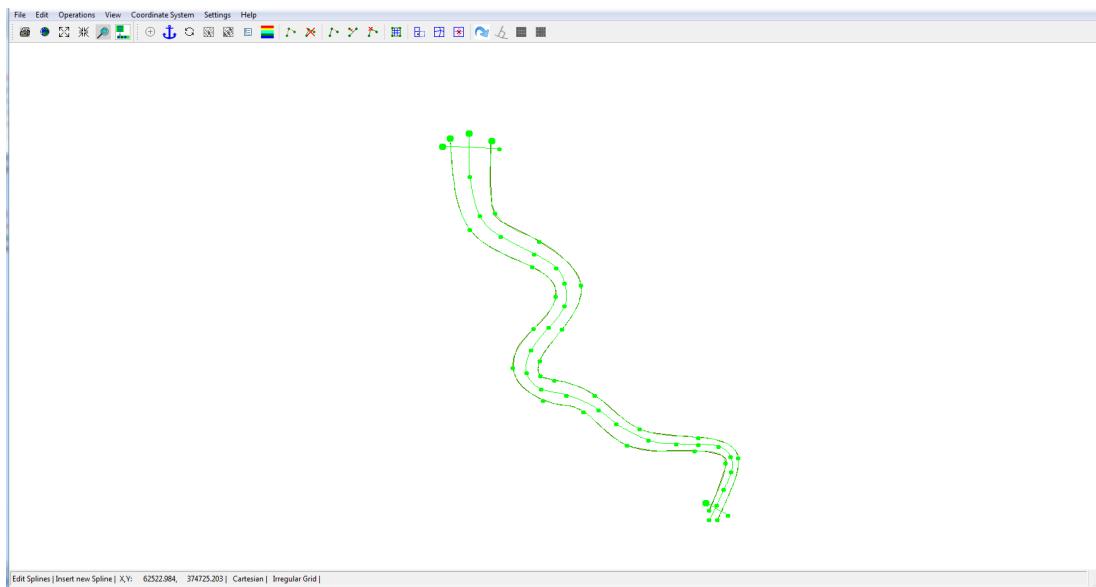


Figure 4.5: Splines in Tutorial01

- Draw two cross-splines, each containing exactly two vertices: one spline at the North side and one spline at the South side of the river (see Figure 4.7). The channel is now enveloped by four splines and there is an extra spline along the river axis. Now we can grow a grid from these splines.

- Choose *Settings* → *Grow Grid from Splines*. . . . You will be able to set several settings for this process. Take over the values shown in Figure 4.6.

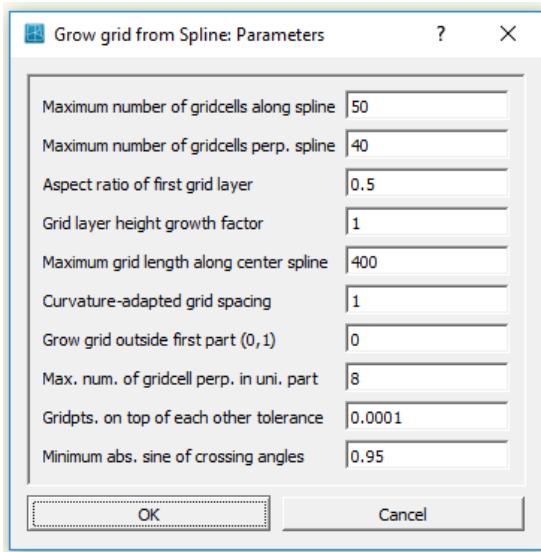


Figure 4.6: Settings for the Grow Grid from Splines procedure.

A brief explanation:

- ◊ Using the parameter *Max. num. of gridcells perp. in uni. part*, the user can give an indication of the number of cells across the width between the longitudinal splines.
 - ◊ By using the parameters *Maximum grid length along center spline*, the user can give an indication of the length of the cells in longitudinal direction. Based on the value of the parameter *Aspect ratio of first grid layer*, the algorithm establishes a suitable grid, under the restrictions of the prevailing maximum numbers of gridcells (first two entries).
 - ◊ The option *Grid layer height growth factor* enables the user to demand for a non-equidistant grid in cross-sectional direction. The value represents the width-ratio of two adjacent cells. Using the option *Grow grid outside first part (0/1)*, one can extend a grid outside the longitudinal splines, for instance to capture winterbed regions.
- After entering the values of Figure 4.6, choose *Operations* → *Grow Grid from Splines*. This will deliver the grid as shown in Figure 4.7.

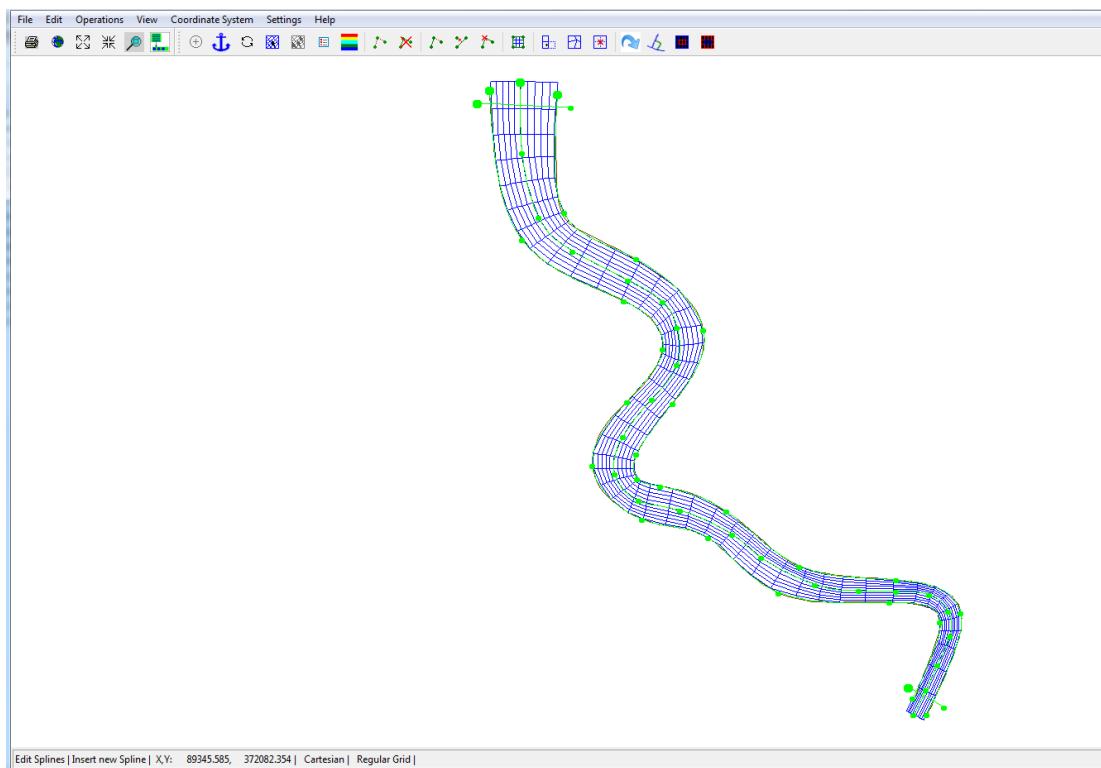


Figure 4.7: Generated curvilinear grid after the new Grow Grid from Splines procedure.

- To be able to further improve the grid, choose *Operations* → *Convert grid* → *Regular to Irregular*. Strictly, the grid is now not curvilinear anymore, but unstructured.
- Choose *View* → *Grid Property Style* → *Coloured Edge* and then *View* → *Grid Property* → *Orthogonality*. The result is shown in [Figure 4.8](#). We remark that no orthogonalisation iterations have been performed yet. Try to reach the level of orthogonality as shown in [Figure 4.8](#).

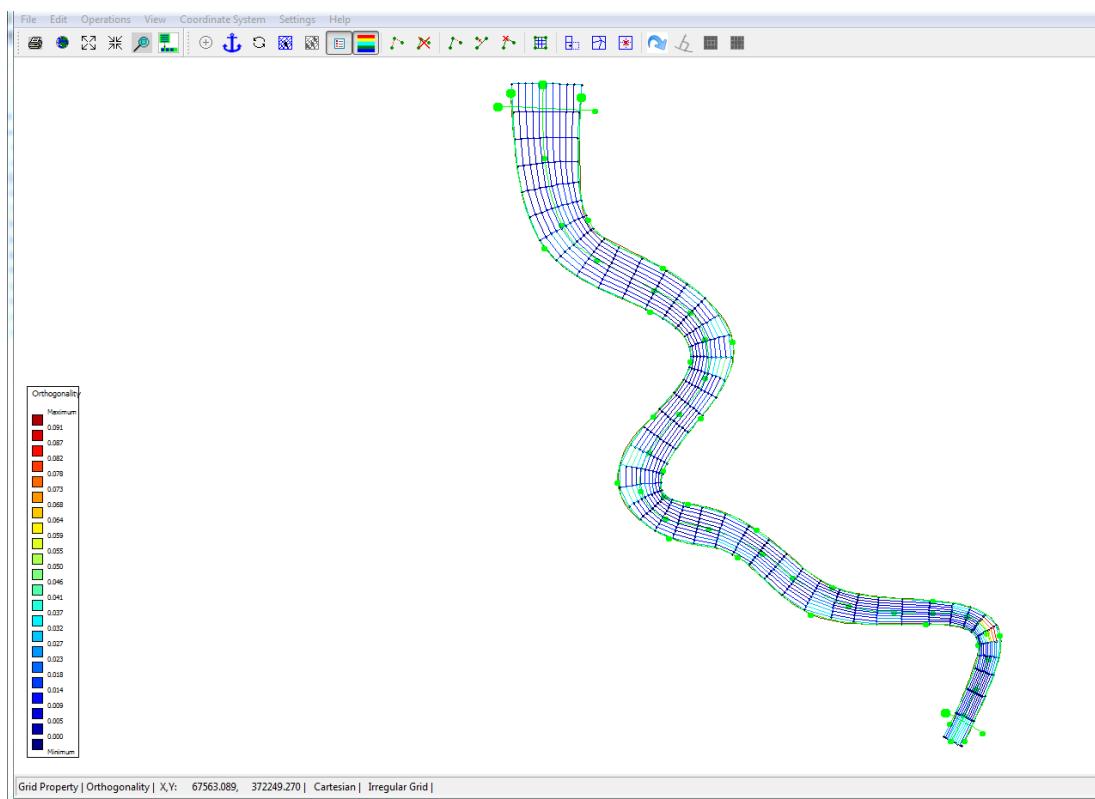


Figure 4.8: Orthogonality of the generated curvilinear grid after the Grow Grid from Splines procedure.

To save the grid and close RGFGRID the following steps have to be carried out:

- Choose from the menubar *File* → *Export* → *UGRID (D-Flow FM)*... and save the grid **in your working folder** as <tutorial01\scheldt01_net.nc>. The grid has now been saved.
- Choose *File* → *Attribute Files* → *Save splines with Intermediate Points* and save the splines **in your working folder** as <tutorial01\scheldt01.spl>. The splines have now also been saved.
- Choose *File* → *Save Project*. Press Cancel when a file selection box asks to save the land boundary.
- Close RGFGRID by choosing *File* → *Exit*

Folder <tutorial01\finished> contains splines and a grid for this tutorial exercise that has been prepared by Deltares.

Now you are back in the Delta Shell GUI. As you can see only the grid has been imported from RGFGRID. If you want to see the land boundary here too, you need to import it:

- Open Area under *Project1* → *FlowFM* → *Area*.
- Import the land boundary from <tutorial01\scheldtriver ldb> by choosing *Import* via the right mouse button on *Project1* → *FlowFM* → *Area* → *Land Boundaries* (see Figure 4.9).
- Do not apply a coordinate transformation by pressing *OK*.

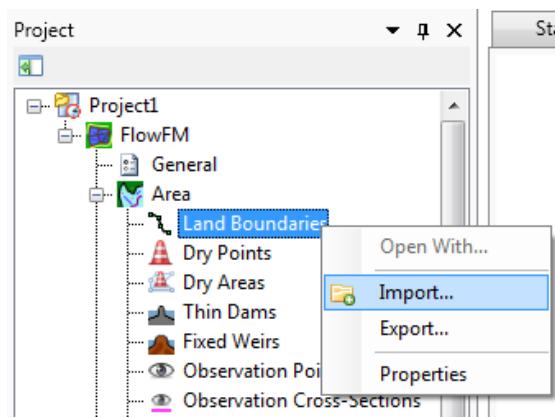


Figure 4.9: Importing a land boundary

Now the grid and land boundary are both visible in the central map of the Delta Shell GUI, see Figure 4.10.

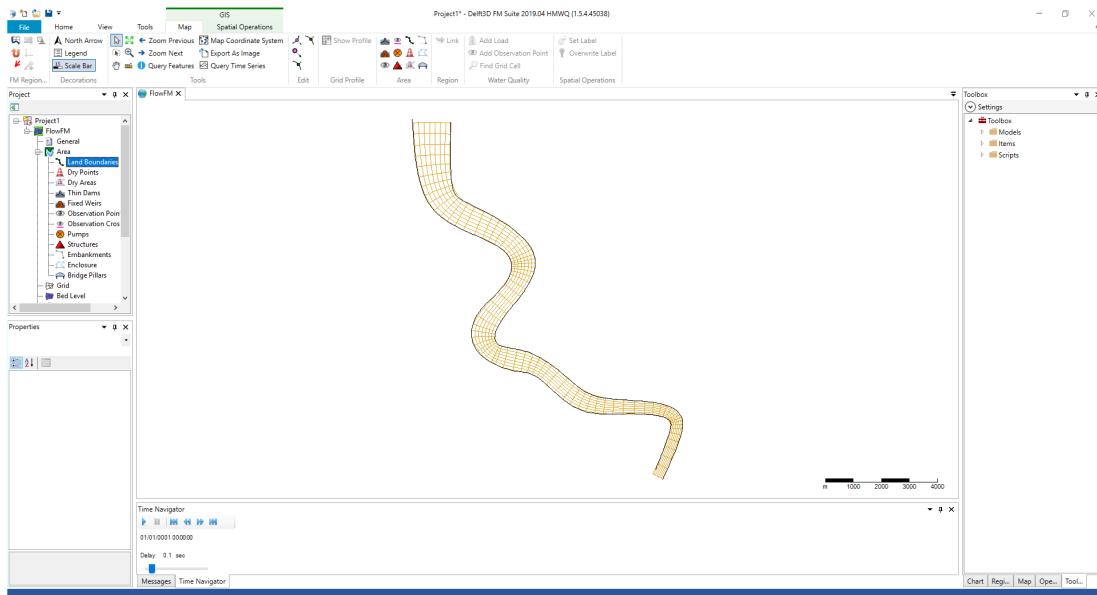


Figure 4.10: After closing RGFGRID the grid is visible in the Delta Shell GUI.

Tutorial 1 has now finished.

- To close Delta Shell, click on *File* → *Exit*. Then, the following question arises: "Save changes to the project? Project?" Click on *No*. Now, the current D-Flow FM model has now been closed.

4.2.3 Tutorial 2: Creating a triangular grid

We will continue with the grid created in the previous tutorial for the Scheldt river. The river is separated from the harbour, west of the river, by a sluice. The small area between the sluice and the Scheldt will benefit from an unstructured grid because of its irregular geometry. The unstructured grid for this irregular geometry is created first in this section.

The following steps have to be carried out:

- Start the the Delta Shell GUI and create a new Flow Flexible Mesh Model as described in [section 4.2.2](#).
- Import the land boundary from <tutorial02\scheldtharbour.ldb> by choosing *Import* via the right mouse button on *Project1* → *FlowFM* → *Area* → *Land Boundaries* (see [Figure 4.9](#)). Do not apply a coordinate transformation, press *OK*.
- Import the grid file by choosing *Import* via the right mouse button on *Project1* → *FlowFM* → *Grid*. Choose the grid file <scheldt02_net.nc> in the folder <tutorial02>.
- Start RGFGRID by double clicking on *Grid*, see [Figure 4.4](#). Set the Coordinate System (if necessary), as has been explained in [section 4.2.2](#) ([Figure 4.4](#)).
- Choose from the menubar *Edit* → *Polygons* → *New*. The intention is to mark the area of interest (i.e. the area for which we want to generate the grid) by means of a polygon, see [Figure 4.11](#).
- Start drawing the polygon at a distance from the curvilinear grid that is similar to the width of the curvilinear grid cells (since we want to have a smooth transition in grid resolution). Specify the second point at a relatively small distance from the first one. This distance is later used as the first indication of the size of the triangular gridcells to be placed.
- Mark the characteristic locations of the area (follow the landboundary) and place the final points at the same distance away from the river grid as the first point. The distance between the last two points should be similar to the distance between the first two points. The result is shown in [Figure 4.11](#).
- Choose *Edit* → *Polygons* → *Refine (linear)* and click on the first and last points of the polyline (in [Figure 4.11](#) we would select point 1 first and then point 15). Now, the polygon is divided into a finer set of line elements (based on the length of the first and last segment of the polyline).

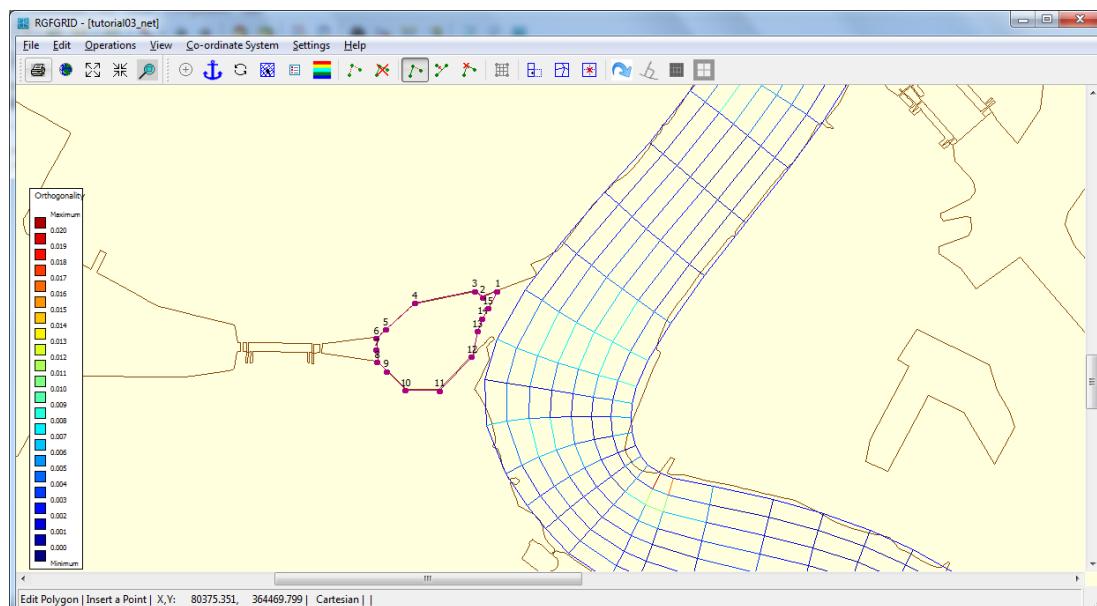


Figure 4.11: Polygon that envelopes the area in which an unstructured grid is aimed to be established.

Remarks:

- ◊ The distance between the points of the polygon is derived from the distance of the two polyline segments at both sides of the *selected* segment. The length of the polyline segments varies linearly from the segment length at the one side of the selected segment towards the segment length at the other side of the selected segment.
- ◊ You can play around to see how this works. If needed, you can add extra polyline points by choosing *Edit* → *Polygon* → *Insert point*. Choose *Edit* → *Polygon* → *Move point* if a point move would make sense.
- ◊ You can snap the refined polygon to the landboundary through *Edit* → *Polygons* → *Attach to Land Boundary*. Select two points for this.

Now we continue with this tutorial by carrying out the following steps:

- Choose *Operations* → *Domain* → *New* to indicate that the new to be generated grid should be created next to the already existing one (and not replace it).
- Choose *Operations* → *Grow Grid from Polygons*. The result is shown in Figure 4.12.
- Improve the orthogonality through *Operations* → *Orthogonalise grid*. The default settings are set to also improve the smoothness.

Since the solver only supports one grid, we need to merge the newly created grid with the original grid of the river. The newly created grid is currently selected. Next:

- Choose *Edit* → *Multi Select* and click on the river Scheldt grid to select it as well. After selection both grids should be coloured blue with black dots on the nodes.
- Choose *Operations* → *Grid* → *Merge Grids* to merge the nodes of the (two) highlighted grids. Now, the grids have been merged. However, nothing will visually change.

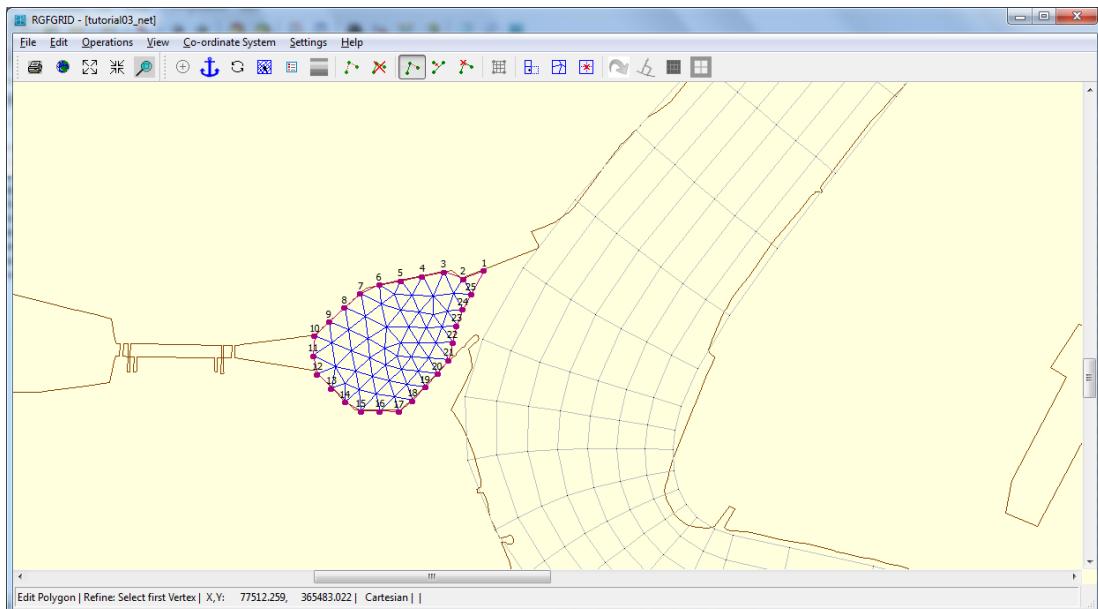


Figure 4.12: Unstructured grid, after having refined the polygon.

To save the grid and go back to the Delta Shell GUI:

- Choose from the menubar *File* → *Export* → *UGRID (D-Flow FM)*... and save the grid **in your working folder** as <tutorial02\scheldt02_net.nc>. Both grids grids have now been merged into one <.nc> file.
- Choose *File* → *Save Project*.
- Close RGFGRID by choosing *File* → *Exit*

Tutorial 2 has now finished.

➤ Close the Delta Shell GUI by choosing *File* → *Exit*. You don't have to save the project.

Folder <tutorial02\finished> contains splines and a grid for this tutorial exercise that has been prepared by Deltaires.

4.2.4 Tutorial 3: Coupling multiple separate grids

From the previous tutorial we have ended up with two separate grids that are not connected yet. Obviously, these two grids should properly be integrated into one single grid. Before we can couple the two grids, we should first make sure that the typical grid size is of the same order of magnitude for both grids at the location where the connection is to be laid. Hence, basically two operations are to be done:

- ◊ Split the grid cells in the Scheldt river along this length so that the cell sizes of both grids match.
- ◊ Merge the two grids and put connections in between.

The splitting can be established as follows:

- Start the the Delta Shell GUI and create a new Flow Flexible Mesh Model as described in [section 4.2.2](#).
- Import the land boundary file and the grid file from the folder <tutorial03> as described in [section 4.2.3](#).
- Start RGFGGRID and set the Coordinate System (if necessary), as has been explained in [section 4.2.2 \(Figure 4.4\)](#).
- Choose *Edit* → *Irregular Grid* → *Split Row or Column*.
- Select the locations where the grid lines should be split by clicking on the left boundary of the Scheldt river.
- Try to achieve the picture shown in [Figure 4.13](#), in particular the typical grid size in the curved area. *N.B. Ignore the red lines in this picture for now!*

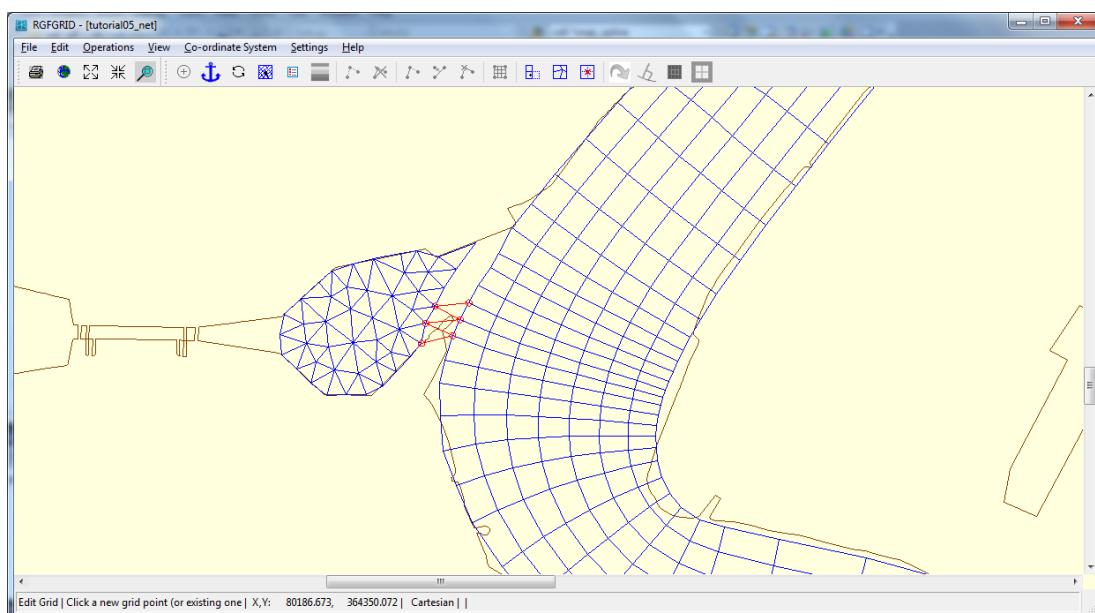


Figure 4.13: Connection of the river grid and the unstructured grid. The red lines show the inserted grid lines used to couple the two grids manually.

The edges of the two the grid segments can be connected as follows:

- Since the two visually separated grids have already been merged into one logical grid in the previous tutorial, we can now physically connect them by laying new gridlines between them. Therefore, select *Edit* → *Irregular Grid* → *Insert Node*. Insert new gridlines in a zigzag-like style: see the red grid lines in [Figure 4.13](#). Now, you will benefit from the (more or less) equal resolutions in the river and unstructured regions.
- Finally, you will end up with a picture like shown in [Figure 4.14](#). You can visualize the orthogonality through *View* → *Grid Property Style* → *Coloured Edge* and *View* → *Grid property* → *Orthogonality*.

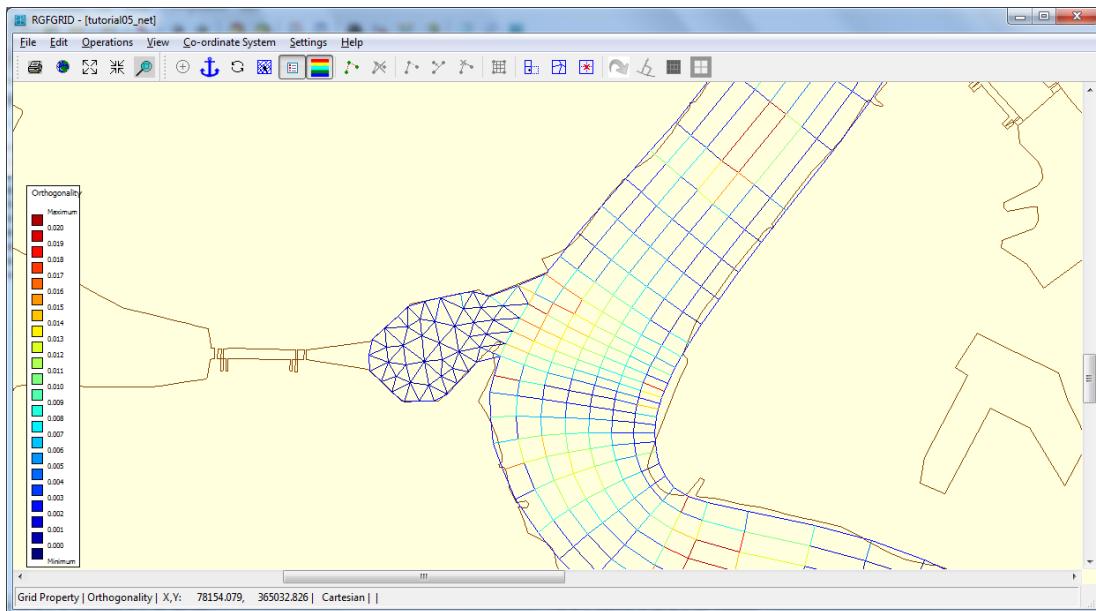


Figure 4.14: Orthogonality of the integrated grid, containing the curvilinear part, the triangular part and the coupling between the two grids.

To save the grid and close RGFGRID is done as follows:

- *File* → *Export* → *UGRID (D-Flow FM)...* and save the grid to your working folder as <tutorial03\scheldt03_net.nc>. The grid has now been saved.
- Choose *File* → *Save Project*.
- Close RGFGRID by clicking on *Exit*.

Tutorial 3 has now finished.

- Close the Delta Shell GUI by choosing *File* → *Exit*. You don't have to save the project.

4.2.5 Tutorial 4: Inserting a bed level

In previous tutorial exercises we focused on the grid generation. The next step is to couple the bed level data to the grid. The grid for this tutorial exercise is available in file <westernscheldt04_net.nc> in the directory <tutorial04>. A harbour has been added to the grid, as well as the Westernscheldt part at the North side. Bed levels are available in the file <westernscheldt_bed_level.xyz>.

The file with bed levels should first be projected onto the unstructured grid by means of interpolation. To this purpose, the following actions should be carried out:

- Start Delta Shell, create a new Flow Flexible Mesh model, import the landboudary <sealand.lbd> and the grid <westernscheldt04_net.nc> from directory <tutorial04> as explained in the previous tutorial exercises.
- Import a sample file containing the measured bed levels: in the drop down menu at the top of the screen (See [Figure 4.15](#)), select the *Spatial Operators* menu, choose layer *Bed Level*, click on *Import from file* and select the file <westernscheldt_bed_level.xyz> from directory <tutorial04>, *Import without transformation (as-is)* → *OK*. This file contains measured bed level data for the Westernscheldt, from the North Sea up to Antwerp

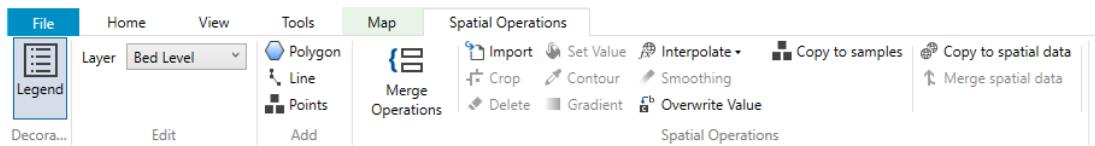


Figure 4.15: Map-ribbon with the Spatial Operations menu.

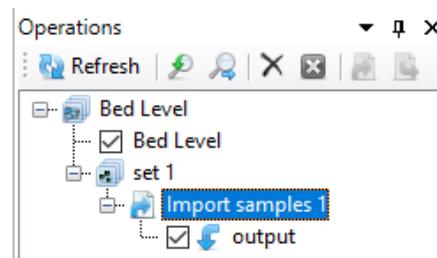


Figure 4.16: Activate import samples 1.

Now we are going to project the bed level data onto the nodes:

- Click on *Operations* and click on "Import samples 1"; see [Figure 4.16](#).
- Click *Interpolate multiple sets* (see [Figure 4.15](#)).
- The window **Interpolation operation** appears. Select the *Overwrite* option for *Pointwise operation*, then press *OK*. The result will look like [Figure 4.17](#). If you do not see a color



bar, click *Legend* in the *Decorations* menu (see [Figure 4.15](#)) and switch off items in the *Map* pane to bring the color bar into view.

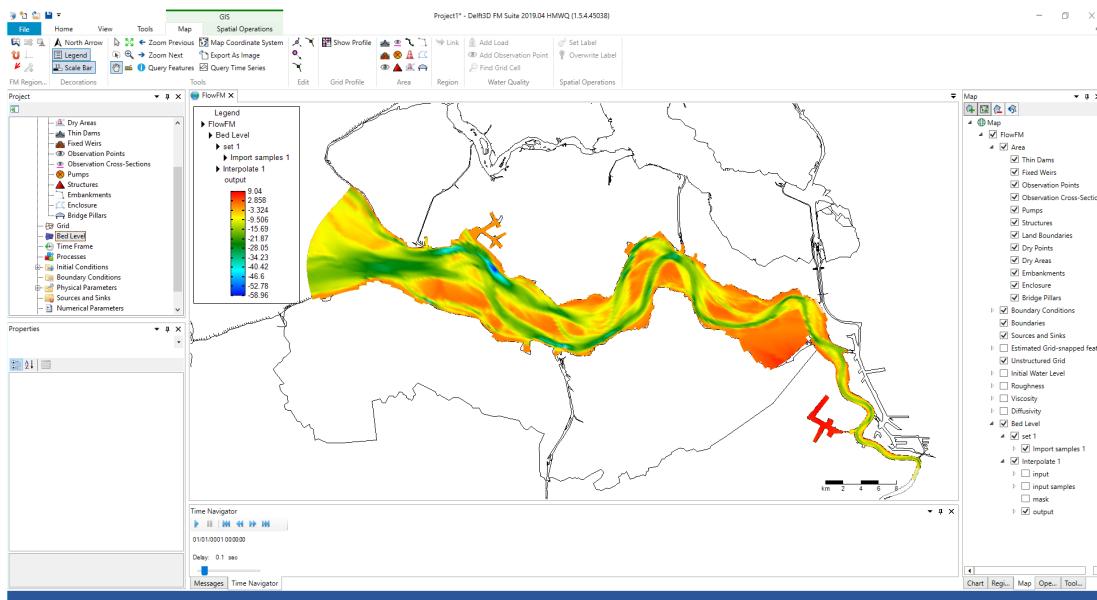


Figure 4.17: Interpolated bed levels values at the grid (estuary).

- Have a look at the sluice and the docks of the harbour. You can see that the bed level in this area is unrealistically high. This unrealistic value has been assigned because of undesired extrapolation, since no bed level data has been available in this area. To repair this, first draw a polygon that envelopes the sluice and the docks. Do this by clicking **Polygon Draw polygon** (see Figure 4.15). Double-click to finish the polygon. See Figure 4.18.
- Choose **Set value** (see Figure 4.15): specify an appropriate value, for instance “-10” m. The result will look like as shown in Figure 4.18.

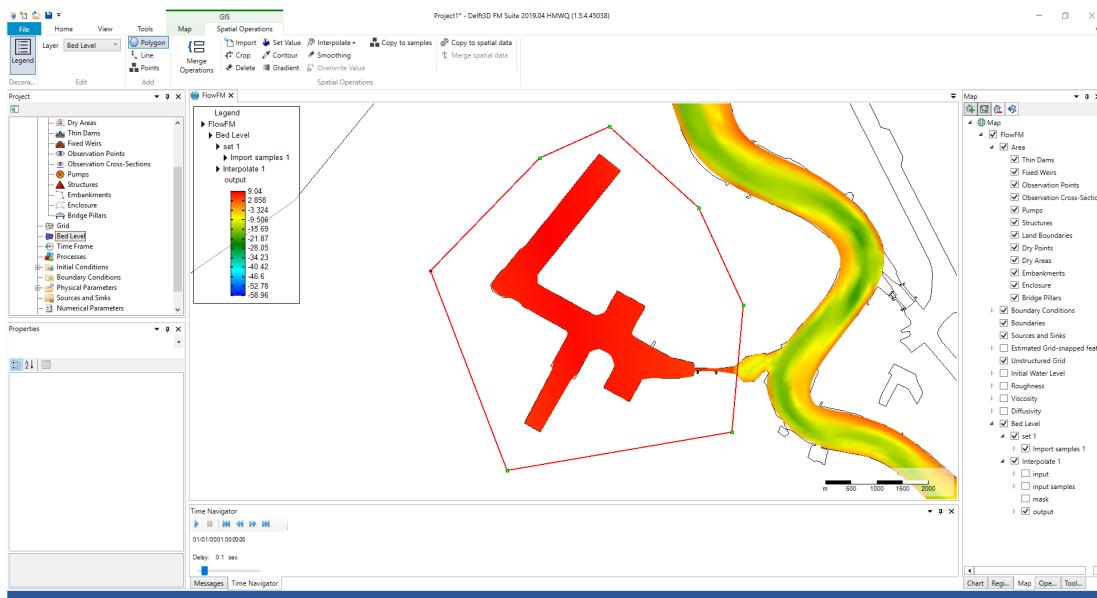


Figure 4.18: Interpolated bed levels values at the grid (harbour).

*N.B. The mesh file (<*_net.nc>) will contain the grid together with the bed level data in the nodes.*

To save the mesh and bed level, and save the model within the Delta Shell GUI the following should be done:

- To save the mesh file you have to rightmouseclick on *Grid* → *Export* in the project tree. Choose *Grid exporter* and save the file in the folder <tutorial04> of **your working environment** as <westernscheldt04_net.nc>. The grid including the bed level have now been saved.
- To close Delta Shell click on *File* → *Exit*. Then, the following question arises: "Save changes to the project: Project?" Click on *No*.

Folder <tutorial04\finished> contains a bed level for this tutorial exercise that has been prepared by Deltaires.

4.2.6 Tutorial 5: Imposing boundary conditions

Along both the sea boundary and the Scheldt river boundary, appropriate boundary conditions have to be imposed. The boundary conditions can be prescribed in many ways, for instance as water levels, velocities (tangential and normal direction), discharge and Riemann. In this tutorial exercise we will impose boundary conditions for the Western Scheldt in the following way:

- ◊ At the sea boundary: a periodic water level boundary, described as a harmonic signal with a mean 0.5 m+NAP, an amplitude 2.5 m and a frequency of $28.984 \text{ } \text{h}^{-1}$,
- ◊ At the river boundary: a constant discharge boundary, described by a constant $2500 \text{ m}^3 \text{ s}^{-1}$.

Let us start with a new D-Flow FM model and import the network and land boundaries:

- Start Delta Shell, create a new Flow Flexible Mesh model, import the boundary <sealand.lbd> and the grid <westernscheldt05_net.nc> from directory <tutorial05> as explained in the previous tutorial exercises.

In order to define boundary conditions one has to follow this strategy:

- 1 draw a polyline along the boundary;
- 2 fill a file with essential information;
- 3 link the boundary files with the model setup.

The boundary conditions can be imposed as follows:

- In the *FM Region...* (in the top ribbon, see [Figure 4.19](#)) choose  *Add a (2D) flow boundary* and insert a polyline along the sea boundary. Doubleclick to finish the polyline. In the Project bar at the left, under *Boundary Conditions*, this boundary has now been added. [Figure 4.19](#) shows how this will look like.
- Tip:** to remove a wrongly placed point in the polyline, use *Remove point from geometry* (top of screen, above *Edit*, see [Figure 4.19](#)). It is also possible to replace or add a point.

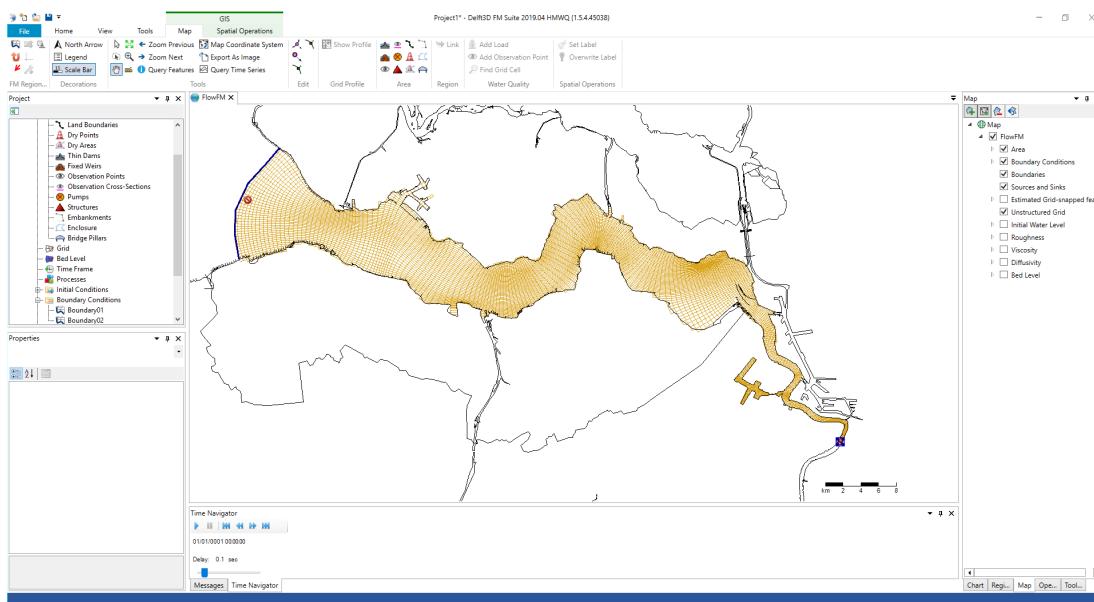


Figure 4.19: Location of the two open boundaries at the sea and river side.

- Draw another polyline at the river entrance south of Antwerp, see the right bottom corner of [Figure 4.19](#)).
- To impose boundary conditions at the seaboundary, double click at *Boundary01*, see [Figure 4.20](#).

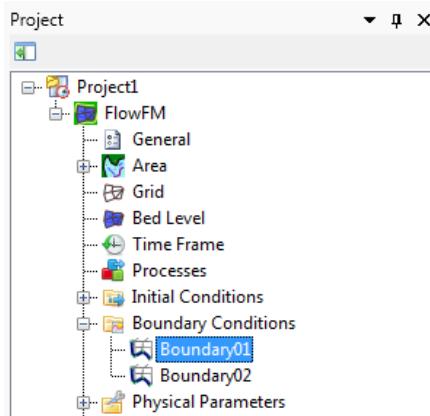


Figure 4.20: Selection of Boundary01.

- Choose *Water level*, press + and choose *Forcing type: Astronomical*.
- Selected the first boundary support point *Boundary01_0001* by clicking on the plus-sign behind it, repeat this for the last boundary support point.
- Choose *Select components...* (at the lefthandside of the screen).
- Choose the components *A0* and *M2*, and press *OK*,
- Select *All support points* and press *OK*. All support points will now have components *A0* and *M2* defined.
- Fill in the amplitudes of the components in the table, see [Figure 4.21](#).

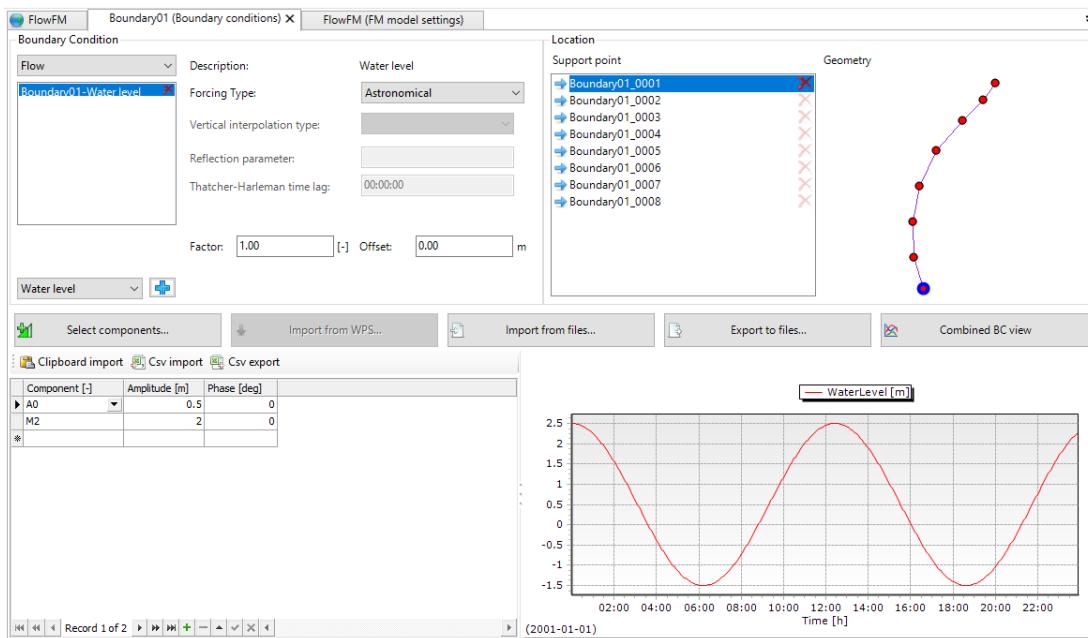


Figure 4.21: Boundary conditions seaside.

We have now prescribed an astronomical water level signal, consisting of two components. The first component has an amplitude equal to 0.5 m with a frequency of 0 degrees/hour, i.e. a constant value of 0.5 m+NAP. The second component has an amplitude of 2.5 m with a frequency of 28.98 degrees/hour. Basically, the signal $h(t) = 0.5 + 2.5 \cos(28.98 t)$ has been prescribed now, with h in meters w.r.t. the reference level (in this case: NAP) and t in hours.

- To impose boundary conditions at the riverboundary, double click at *Boundary02*, Choose *Discharge* and press **+**. Choose *Forcing type: Time Series*.
- To impose a contant discharge of $2500 \text{ m}^3 \text{ s}^{-1}$, fill in the table as in Figure 4.22. Make sure you insert the correct times!

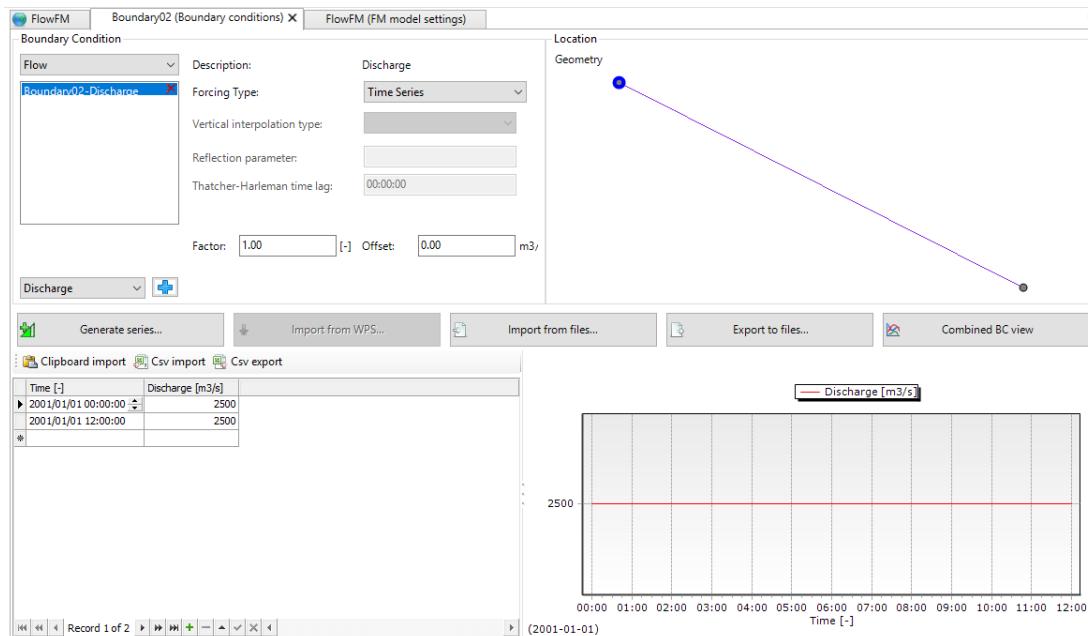


Figure 4.22: Boundary condition riverside.

To be able to load this D-Flow FM model in the future, it is necessary to save the model:

- In the project tree, rightclick on *FlowFM* and select *Rename*. Rename the model to “westernscheldt05”.
- Save the model in folder <tutorial05> **of your working environment**: in the project tree rightclick on *westernscheldt05*. Choose *Export* select *Flow Flexible Mesh model*, press *OK* and save the model as <westernscheldt05.mdu>.

The model can now be loaded in the next tutorial exercise without the need of inserting all network files or land and flow boundaries individually again.

- Now close the D-Flow FM model by rightmouseclicking In the project bar on *westernscheldt05* → *Delete* → *OK*.
- Close the Delta Shell GUI by choosing *File* → *Exit*. You don't have to save the project.

4.2.7 Tutorial 6: Defining output locations

Often one would like to monitor computational results at certain specific cross sections or locations (i.e. 'observation points'). In this tutorial exercise we will add cross sections and observation points in a D-Flow FM model.

We will start with importing an existing D-Flow FM model without cross sections and observation points:

- Start Delta Shell and import a D-Flow FM Model by choosing (in the top ribbon) *Import* → *Flow Flexible Mesh Model* and press *OK*. Then choose <westernscheldt06.mdu> from the folder <tutorial06> and click on *Open*.

Inserting cross sections and observation points is rather straightforward. The following actions are necessary:

Inserting cross sections:

- Choose  *Add new observation cross section (2D)* (see the *Area* menu in the ribbon).
- Draw as many cross sections as you like. Finalize each cross section by doubleclicking the left mouse button.
- To see how many cross sections have been made, double click on *Area* → *Observation Cross-Sections* in the project tree, see [Figure 4.23](#).

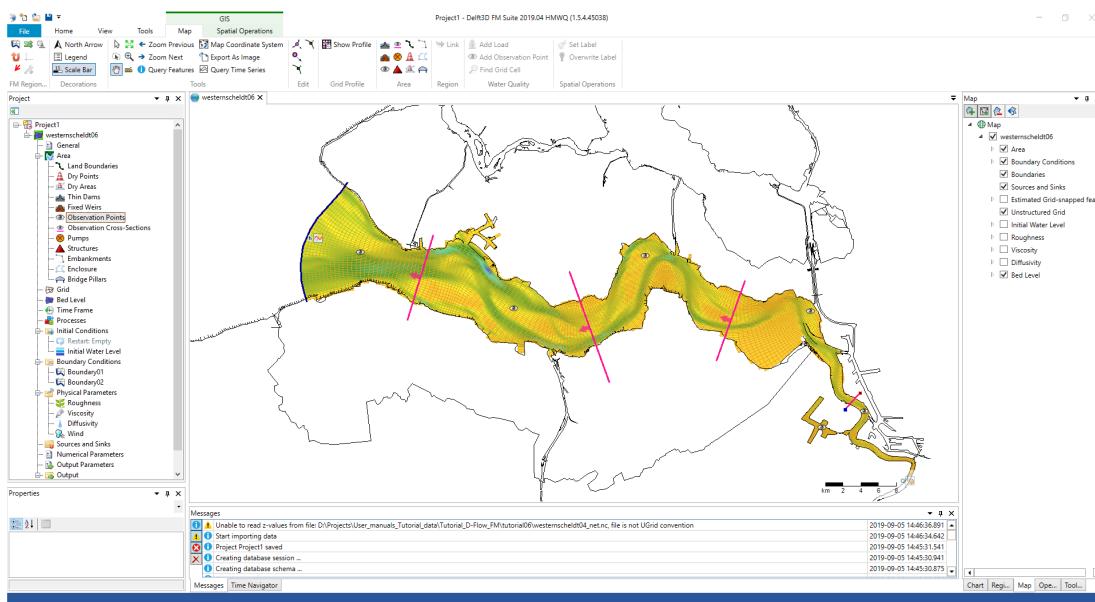


Figure 4.23: Overview cross sections and observation points.

Now we will add observation points:

- Choose *Add new observation point*. Add as many observation points as you like.
- To see how many observation points have been made, double click on *Area* → *Observation Points* in the project tree.
- Save the current FM model by clicking the right mouse in the project tree on *westernscheldt06*.
- Choose *Export*, choose *Flow Flexible Mesh Model* and press *OK*. Then save the model with name “westernscheldt06” in the folder <tutorial06> **of your working environment**. The observation points and cross sections have now been saved in the new <mdu>-file.
- **Tip:** It is also possible to save only the observations points. For instance to a different folder. To try this, click the right mouse on *Observation points* → *Export* → *Observation points to <xyn>-file*. Then, choose a filename and a folder.
- Close the current Flexible Mesh model by clicking the right mouse in the project tree on *westernscheldt06* → *Delete* → *OK*.
- Close the Delta Shell GUI by choosing *File* → *Exit*. You don't have to save the project..

4.2.8 Tutorial 7: Defining computational parameters

Before we can run a model, all computational parameters need to be set. This involves for example the *Time Frame*, *Initial Conditions* and the *Output Parameters*. Most parameters can be set in parameter screens accessible through the project tree. All other parameters are set by default, and will not be considered in this tutorial exercise.

We will start by importing an existing D-Flow FM model:

- Start Delta Shell and import the D-Flow FM Model from the folder <tutorial07>, by choosing *Home* → *Import* → *Flow Flexible Mesh Model* → *OK*. Choose <westernscheldt07.mdu> and press *Open*.

Several parameters will be set. We will start with the *Time Frame*, see Figure 4.24:

- At the Time Frame section: fill in the parameters shown in Figure 4.24.

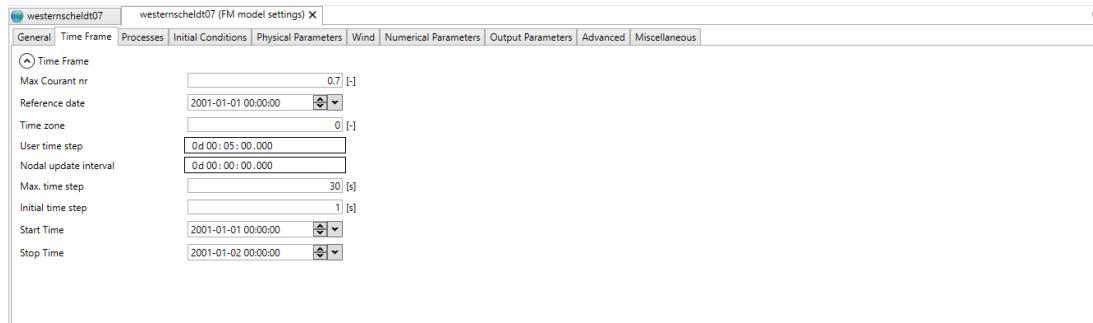


Figure 4.24: The time frame of the simulation.

- At the Initial Conditions section: fill in “0.5” m for the Initial water level, see **Figure 4.25**.

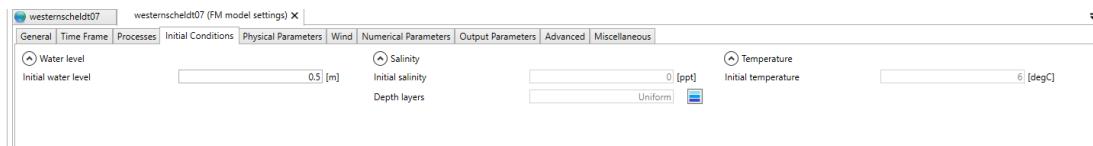


Figure 4.25: Imposed initial conditions for the simulation.

- At the Output Parameters section: fill in the parameters shown in **Figure 4.26**.

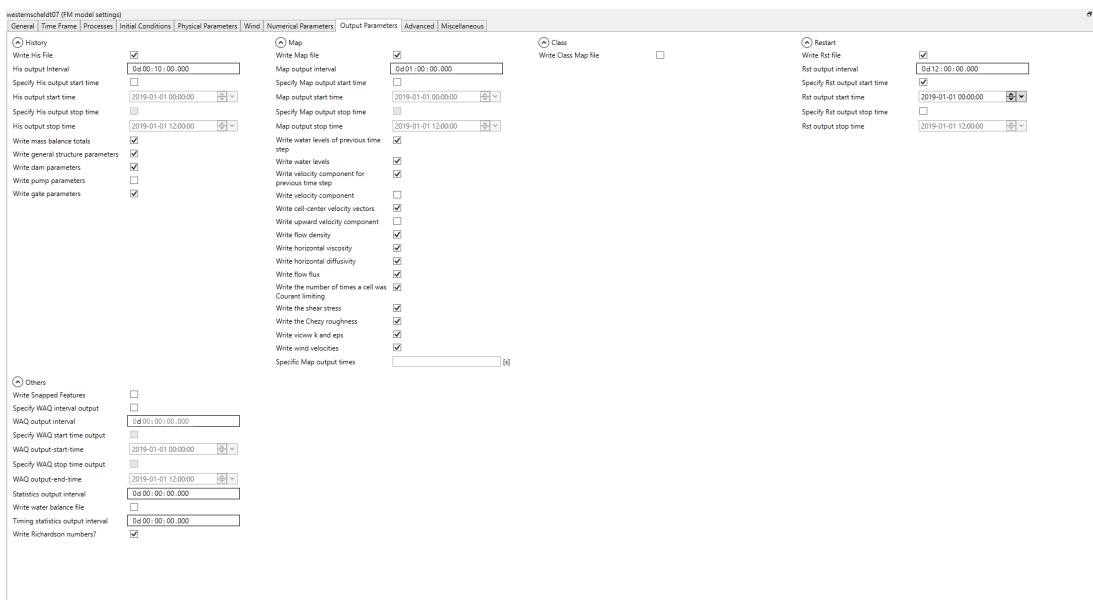


Figure 4.26: Optional output parameters for the computation.

As an explanation, the computation will generate history-files and map-files:

- ◊ In history-files, timeseries are stored at the cross-sections and observation locations, at a frequency specified via the parameter *His Output Interval*.
 - ◊ The map-files collect data over the entire domain, at a frequency specified via the parameter *Map Output Interval*.
- Note:** Be aware that these periods are clipped by the parameter *User time step*, which has been specified under *Time Frame*. That means, if *User time step* > *His Output Interval*, then the period with which his-files are written is given by *User time step*.
- ◊ Restart files will be written when *Write Rst File* is selected.



The period with which these files are written can then be specified at *Rst Output Interval*. This period is not clipped by *User time step*. To start a computation with a restart file is not part of this tutorial.

- Save the current D-Flow FM model in the folder <tutorial07> of **your working environment** by rightmouseclicking in the project tree on *westernscheldt07*, choose *Export* and save the model as *Flow Flexible Mesh model* <westernscheldt07.mdu>.

The computational parameters have now been saved in the <mdu>-file.

- Now close the current FM model (In the project bar, rightmouseclick on *westernscheldt07* → *Delete* → *OK*).

4.2.9 Tutorial 8: Running a model simulation

Up till now we have saved the D-Flow FM model by means of a <mdu>-file. For running computations, it is necessary to save the entire project, which will be done in this tutorial exercise:

- Import the D-Flow FM Model from the folder <tutorial08>, by choosing *Import* → *Flow Flexible Mesh Model* → *OK*. Choose <westernscheldt08.mdu> and *Open*.

For saving the entire project, the following should be done:

- Choose *File* → *Save As* to save <tutorial08.dsproj> in the folder <tutorial08> of **your working environment** and click *Save*.

The project has now been saved in a folder called <tutorial08\tutorial08.dsproj_data> and a project file <tutorial08\tutorial08.dsproj> has been created (See Figure 4.27). Within this folder you will find all input files of the model (some are ASCII), output files of the model (when it has been run) and, if applicable, zip-folders containing the restart files. Note that after saving, the project has been renamed to *tutorial08*

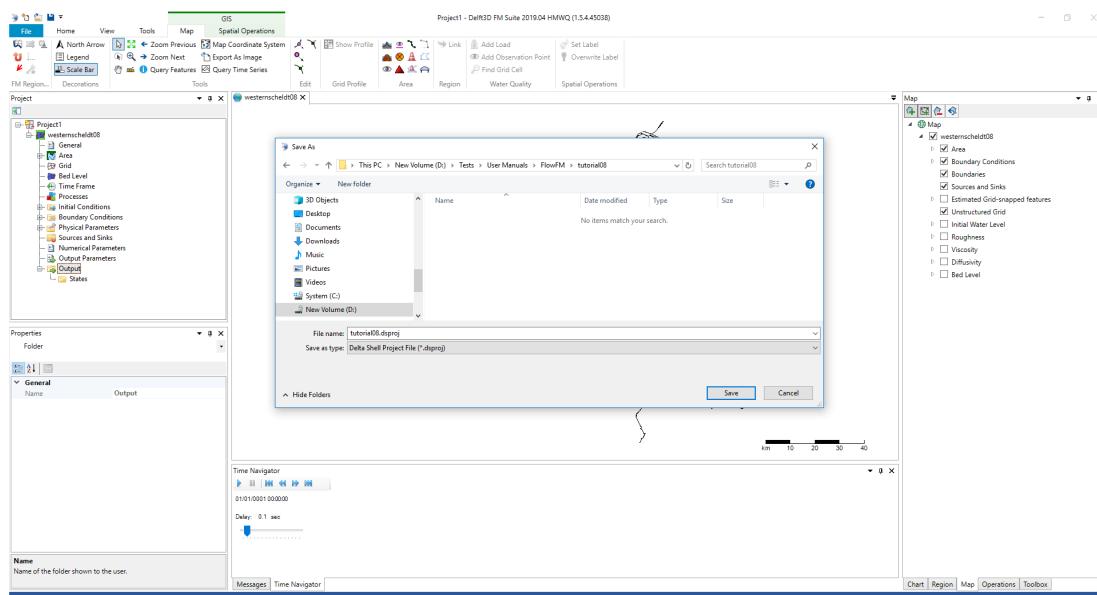


Figure 4.27: Menu for saving a project.

To start a computation, the following needs to be done:

- In the project tree select the model you want to run (which has been named *western-*

schedt08, see Figure 4.28).

- Press the right mouse and choose *Run Model*

The simulation will now start running.

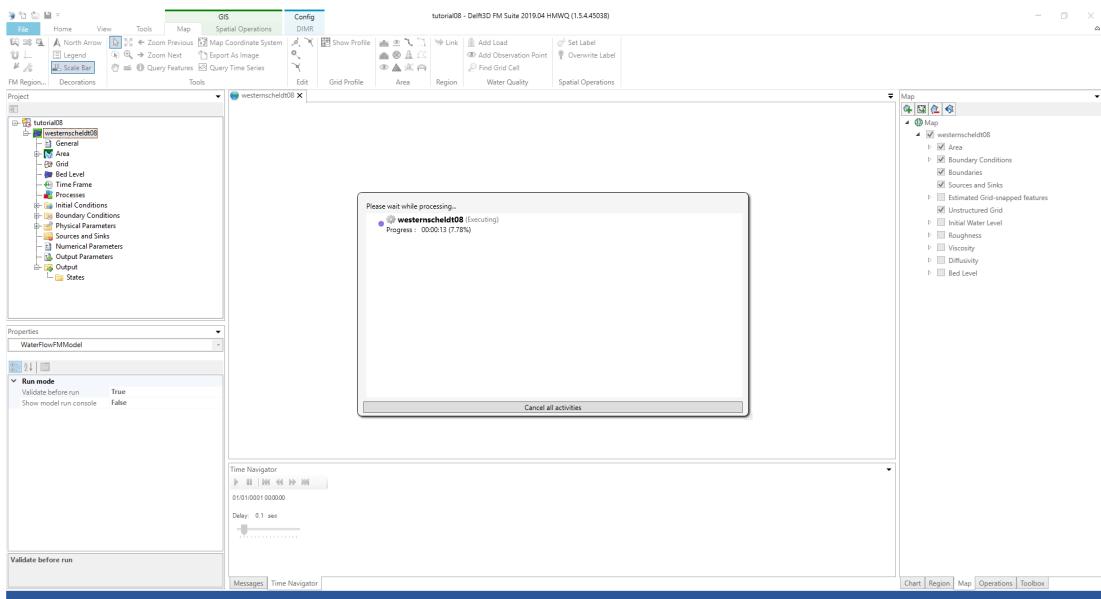


Figure 4.28: View of Delta Shell when running a model.

Now:

- Save the project with generated output in the folder <tutorial08> **of your working environment** (*File* → *Save As* and save the project with name “tutorial08.dsproj”). Press *Yes* to overwrite the project.
- Now close the current D-Flow FM model (In the project bar, rightmouseclick on *westerndscheldt08* → *Delete* → *OK*).

Folder <tutorial08\finished> contains the output for this tutorial exercise that has been prepared by Deltares.

4.2.10 Tutorial 9: Viewing the output of a model simulation

The output of the model can be observed within Delta Shell. At first a project has to be imported:

- Choose *File* → *Open*. Select the file <tutorial9.dsproj> from the folder <tutorial09> and choose *Open*. The project is now activated.
- Doubleclick on *westerndscheldt09*.

To view the output of the history files (observation points and cross-sections):

- Click with leftmousebutton on an observation point (while being in *select* mode. Your mouse looks like an arrow in this mode).
- Choose at the top part of the screen *Query Time Series*, see Figure 4.29 (option available within the Map-ribbon).

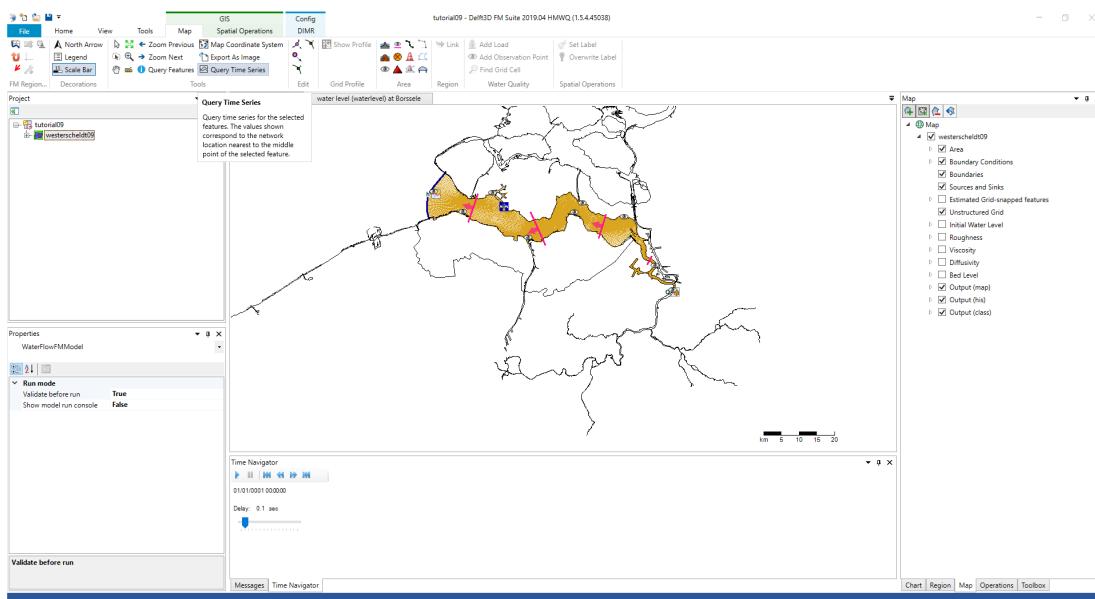


Figure 4.29: View of Delta Shell, available to select a location for timeseries in.

- Choose *Water level (waterlevel)* → OK. You can now observe the water levels in the observation point you selected, see [Figure 4.30](#). You can choose other parameters as well to observe. Also cross-sections can be observed this way.
- Select the Time Navigator in the *View* section of the ribbon. The Time Navigator will appear at the bottom of Delta Shell.

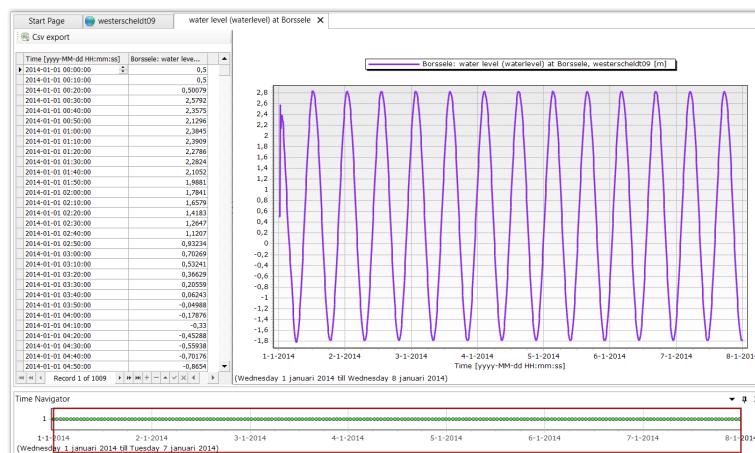


Figure 4.30: View of Delta Shell, time-series for observation point "water level at Borsele".

To view the output of the map files:

- Select the pane 'westerscheldt09'
- Choose in the Map-tree Add New Wms Layer (see [Figure 4.31](#)) and choose <http://openstreetmap.org>.

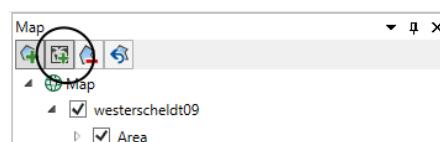


Figure 4.31: WMS layer icon at the top of the map-tree viewer.

Now, the model can be observed with OpenStreetMap in the background. See Figure 4.32.

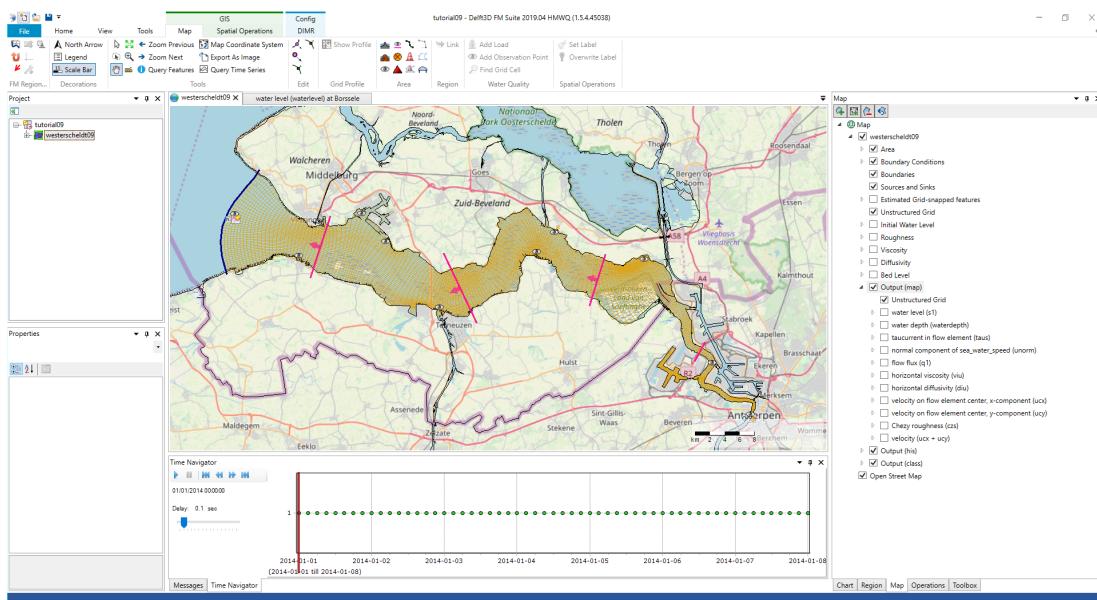


Figure 4.32: View of Delta Shell in combination with OpenStreetMap.

- Open *Output* in the map tree. Check *waterlevel(s1)* to observe water levels, see Figure 4.33.

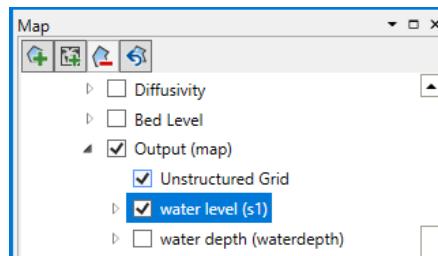


Figure 4.33: Select *waterlevel(s1)* from the map tree

- To adjust the legend, rightmouseclick on *Output(map) → waterlevel (s1)* and select *Properties*. Now the window **Layer Properties Editor** appears in which the legend can be adjusted.
- Set classes to “11”
- Select *Custom range* and set the legend values for minimum and maximum value to “-2.5” to “-0.5”.
- Press *Generate*
- Press *OK*, see Figure 4.34.

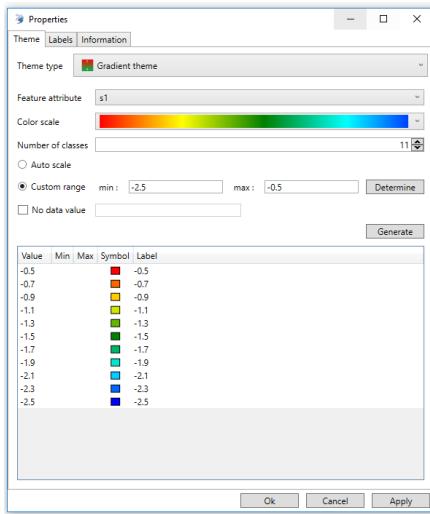


Figure 4.34: Layer properties editor

- Click the play button in the *Time Navigator* to see the water levels change in time. To see the water depths, uncheck the water level in the map tree and check the water depth.
- Now close the current D-Flow FM model (In the project bar, rightmouseclick on *western-scheldt09* → *Delete* → *OK*).

5 All about the modelling process

5.1 Introduction

In order to set up a hydrodynamic model you must prepare an input file. All parameters to be used originate from the physical phenomena being modelled. Also from the numerical techniques being used to solve the equations that describe these phenomena, and finally, from decisions being made to control the simulation and to store its results. Within the range of realistic values, it is likely that the solution is sensitive to the selected parameter values, so a concise description of all parameters is required. This input data is collected into the Master Definition Unstructured file, called a mdu-file.

In section 5.2 we discuss some general aspects of the mdu-file and its attribute files. The sections thereafter describe how the actual modelling process can be done in the GUI.

5.2 mdu-file and attribute files

The Master Definition Unstructured file (mdu-file) is the input file for the hydrodynamic simulation program. It contains all the necessary data required for defining a model and running the simulation program. In the mdu-file you can define attribute files in which relevant data (for some parameters) is stored. This will be particularly the case when parameters contain a large number of data (e.g., time-dependent or space varying data). The mdu-file and all possible user-definable attribute files are listed and described in Appendix A.

Although you are not supposed to work directly on the mdu-file it is useful to have some ideas on it as it reflects the idea of the designer on how to handle large amounts of input data and it might help you to gain a better idea on how to work with this file.

The basic characteristics of an mdu-file are:

- ◊ It is an ASCII file.
- ◊ Each line contains a maximum of 256 characters.
- ◊ Each (set of) input parameter(s) is preceded by a (set of) *keyword(s)*.

The results of all modules are written to platform independent binary (NetCDF-)files, so also these result files you can transfer across hardware platforms without any conversion.

The mdu-file contains several sections, denoted by square brackets, below are the most relevant ones:

- [model] this section contains the program name and its version.
- [geometry] in this section, the main entry comprises the specification of the grid (i.e. the netcdf network file). In addition, thin dams and thin dykes can be specified.
- [numerics] this section contains the settings of specific parts of the flow solver, such as limiters and the iterative solver type.
- [physics] in this field, physical model parameters can be inserted, for instance related to friction modelling and turbulence modelling.
- [wind] the wind section prescribed the dependency of the wind drag coefficient to the wind velocity through 2 or 3 breakpoints. This field also contains pressure information.
- [time] in this section, the start time and the stop time of the simulation are specified in hours, minutes or seconds. The other times specified are specified in seconds.
- [restart] in this section, the restart file can be specified, either as a <*_map.nc>-file or as an <*_rst.nc> file.

[external forcing] this section only contains the name of the external forcings file.
[output] in this section, the writing frequency of output data can be prescribed.

Appendix A contains the full list of MDU sections and keywords.

5.3 Filenames and conventions

Filenames and file extensions hardly have any strict requirements in D-Flow FM, but we do advise to use the suggested file naming patterns below:

file pattern	description
<i>mdu_name.mdu</i>	mdu-file
*_net.nc	Unstructured grid (network) file
*.xyz	Sample file (for spatial fields)
*.ldb	Landboundary file (polyline file format)
*_thd.pli	Thin dam file (polyline file format)
*_fxw.pliz	Fixed weir file (polyline file format with <i>z</i> values)
*_part.pol	Partitioning polygon file (polyline file format)
*.ext	External forcings file
<i>pli_name.pli</i>	Boundary condition location file (polyline file format)
<i>pli_name_000X.tim</i>	Timeseries boundary data file at point #X
<i>pli_name_000X.cmp</i>	Astronomic/harmonical component boundary data file at point #X
*.bc	BC-format boundary data file with polyline and point labels in file
*.xyn or *_obs.ini	Observation station file (see section F.2.2)
*_crs.pli or *_crs.ini	Observation cross-sections file (see section F.2.4)
<i>mdu_name_map.nc</i>	Output map file
<i>mdu_name_his.nc</i>	Output his file
<i>mdu_name_clm.nc</i>	Output class map file (see section F.3.3)
<i>mdu_name.dia</i>	Output diagnostics (log) file

5.4 Setting up a D-Flow FM model

This chapter describes how to set up a D-Flow FM model in an empty Delta Shell project. When you open the GUI, an empty project is automatically created. Starting from scratch, you have to create an empty D-Flow FM model in a Delta Shell project:

- ◊ In the ribbon menu items, go to *Home* and click on *New Model*. In the drop-down menu [Figure 5.1](#) that appears, select "Flow Flexible Mesh Model" to add an empty D-Flow FM model.
- ◊ Or use the right mouse button on the name of your project (project1 by default). In the context menu that appears [Figure 5.2](#), select *Add* and click *New Model*. The *Select model ...* window appears allowing you to add an empty D-Flow FM model.

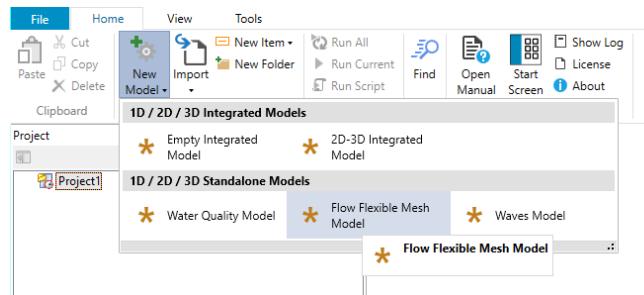


Figure 5.1: The Add Model drop-down menu

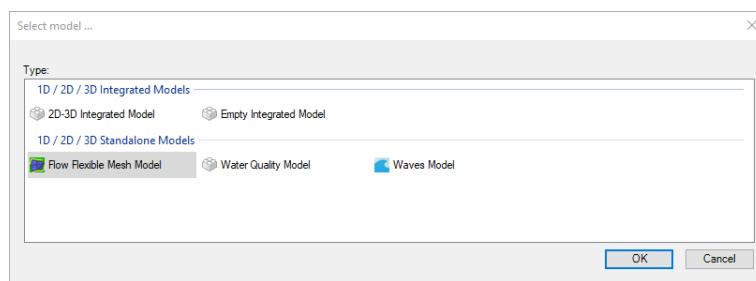


Figure 5.2: The Select model ... window

In the **Project** window, an empty model has appeared (FlowFM by default). Click the plus sign (+) before the name of the model to expand all model attributes in the **Project** window: General, Area, Grid, Bed Level, Time Frame, Processes, Initial Conditions, Boundary Conditions, Physical Parameters, Sources and Sinks, Numerical Parameters, Output Parameters and Output. In the following paragraphs, all model attributes are treated separately.

5.4.1 General

When you double click on *General* in the **Project** window, the tabbed editor for the D-Flow FM appears (Figure 5.3). The general tab contains general model information such as the number of vertical layers, the model coordinate system, the angle of latitude and the angle of longitude.

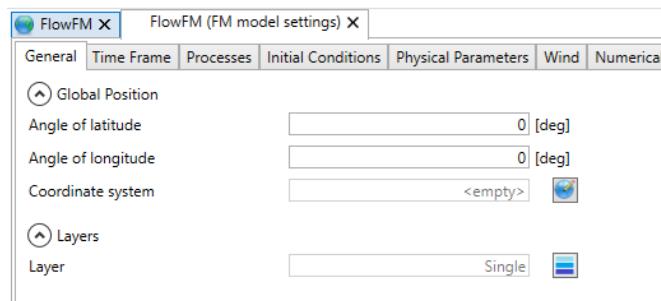


Figure 5.3: Overview of general tab

5.4.1.1 Vertical layer specification

When you click on the blue striped icon next to the vertical layers text box (depth layers), the **Edit depth layers** window appears (Figure 5.4). This window allows you to choose the type of layering and the corresponding number of layers. The drop down menu contains two distinct layering types:

- 1 Single
- 2 Z
- 3 Sigma (β -functionality)



Note: Currently, only the Single and Sigma type layering are presented.

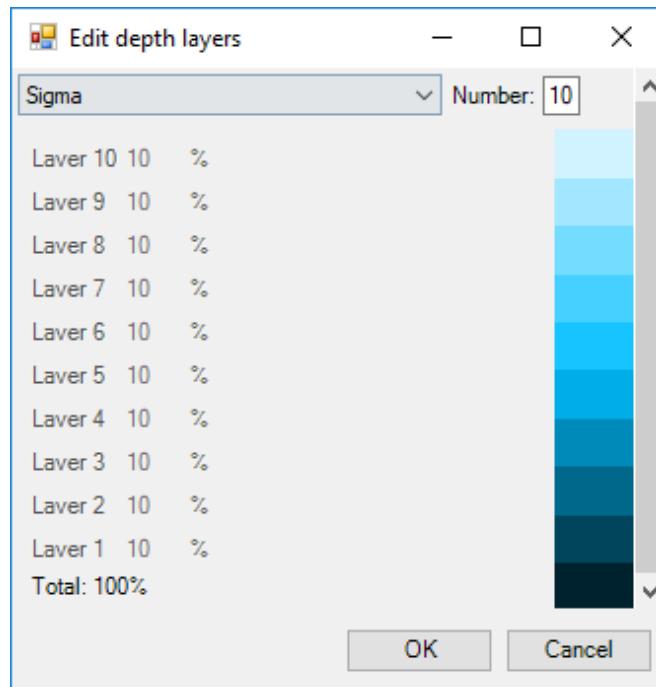


Figure 5.4: Vertical layer specification window (σ -model is β -functionality)

The recommended type of vertical layering differs depending on the model application and the processes that you are interested in. Layers in the σ -model increase or decrease in thickness as the water depth in the model increases or decreases. The relative thickness distribution of the different layers however remains fixed. Layers in the Z -model have a fixed thickness, which does not change as the water depth in the model varies. If the water depth drops below the cumulative thickness of all z -layers, layer(s) will fall dry. When *Single* is chosen, the model contains only 1 vertical layer. (An extensive description of σ - and z -type layering is found in section 8.3).

If your model contains more than 1 layer, the thickness distribution of the vertical layers can be specified. In Delta Shell, user can specify the number of layers and then automatically obtain uniformly distributed layer thicknesses (The total percentage is 100 %). As shown in Figure Figure 5.4, setting 10 layers results in 10 % thickness distribution for each layer. Specifying non-uniformly distributed layer thickness is β -functionality and can be done through the MDU file.

5.4.1.2 Model coordinate system

A very important property of your model is the coordinate system in which it is specified. Within the interface, there is a clear distinction between the coordinate system of your model and all of its attributes and the coordinate system of the central map and all of its items. Both coordinate systems can be set independent from each other. Keep in mind, that the coordinate system of your model is saved and used when you run your model. Only coordinate systems supported by the computational core are supported.

The model coordinate system can be set using the globe icon next to the Coordinate system text box. After clicking this button, the coordinate system wizard is opened (Figure 5.5). This wizard allows you to choose one of many possible coordinate systems and apply it to your model. You can use the search bar to browse the various coordinate systems (searching possible by name and EPSG code). If your model is specified in a certain coordinate system already, it is possible to convert the model coordinate system using the same button. After clicking *OK*, all model attributes are converted to the system of choice. Note that you have to close and re-open all map views for the changes to take effect in these views.

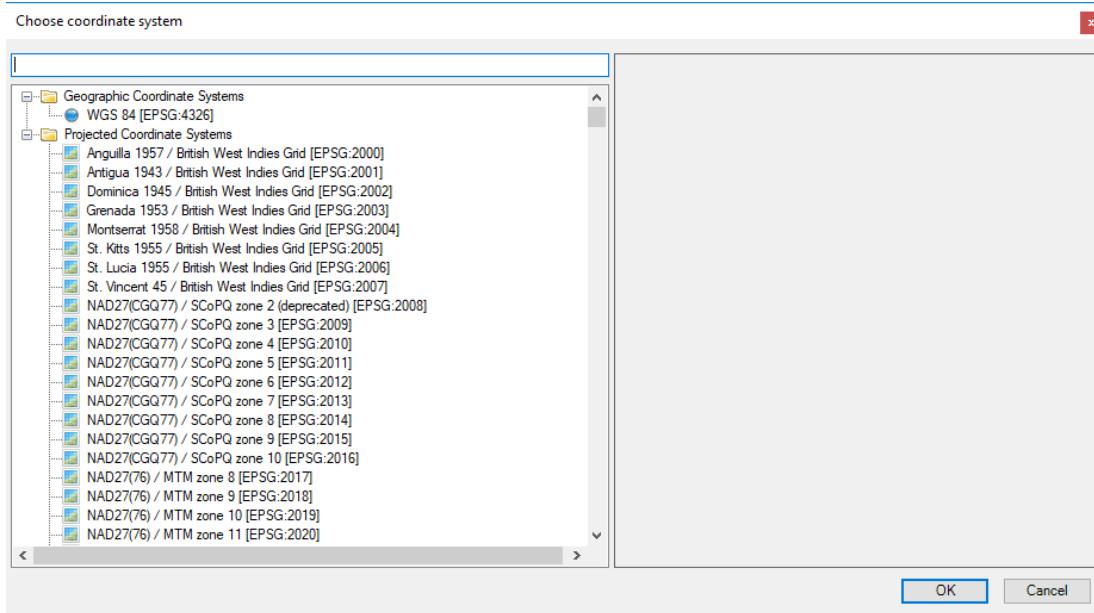


Figure 5.5: Coordinate system wizard

The map coordinate system applies to all items in the map **Project** window. To change the map coordinate system, navigate the menu ribbons to *Map* and click on *Map coordinate system*. Alternatively, right mouse click on *Map* in the map *Project* window and select *Change Map Coordinate System*. The coordinate system wizard appears, allowing you to set the map coordinate system of your choice. When the original model coordinate system differs from the selected map coordinate system, all map items are automatically converted to the specified map coordinate system.

5.4.1.3 Angle of latitude

For a Cartesian grid you have to specify the latitude of the model area; this is used to calculate a fixed Coriolis force for the entire area. For a spherical grid the Coriolis force is calculated from the latitude coordinates in the grid file and thus varies in the latitude direction. Typically, you use spherical co-ordinates for large areas, such as a regional model. When a value of 0 is entered, the Coriolis force is not taken into account.

5.4.2 Area

The model area contains all geographical features, such as the observation points, structures and land boundaries. These features can exist without a grid or outside the grid as they are not based on grid coordinates but xy -coordinates. This means that the location of the model area features remains the same when the grid is changed (for example by (de-)refining). When you expand the model attribute *Area* in the *Project* window, a list of supported geographical features is displayed (Figure 5.6).

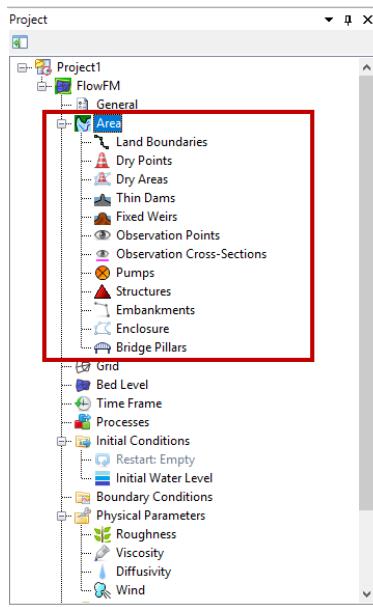


Figure 5.6: Overview of geographical features

Below is Figure 5.7. It displays an overview of the (*Map*) ribbon.

The red boxes to the left (*FM Region 2D / 3D*) and right (*Area*) can be used to *Add ...* the various geographical features to the model: click the corresponding item in the box and use your mouse to indicate the location of the selected feature on the central map. The red box in the middle highlights the buttons available to edit the location of geographical features. The specifics for each feature are discussed separately in the following sections.

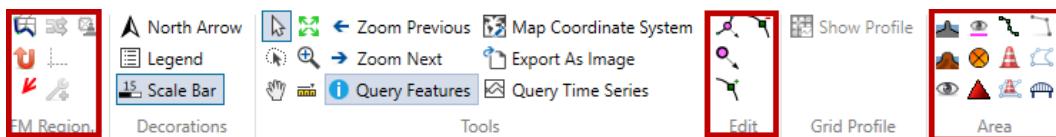


Figure 5.7: Overview of Map ribbon

Importing and exporting of model features is done with the context menu by using your right mouse button on the different features in the *Project* window.

5.4.2.1 Grid-snapped features

All geographical features of your model that are described by x -, and y -coordinates have to be interpolated to your computational grid when you run your model. The computational core of D-Flow FM automatically assigns these features to the corresponding parts of your

grid. The so-called Grid-snapped features are part of the output of a simulation: *shape files*. In section 5.4.12 is described how - in short: select *Write Snapped Features* in the *Output Parameters* tab below the central map.

However, the graphical user interface allows you to inspect the grid-snapped features before a simulation run.

Note: Mark that this way, the grid-snapping is an estimation because it would take too much processing time. In the neighbourhood of *Dry Areas* the grid-snapping will be different, as the *Dry Areas* are not taken into account in this interactive *Estimated Grid-snapped features* mode.

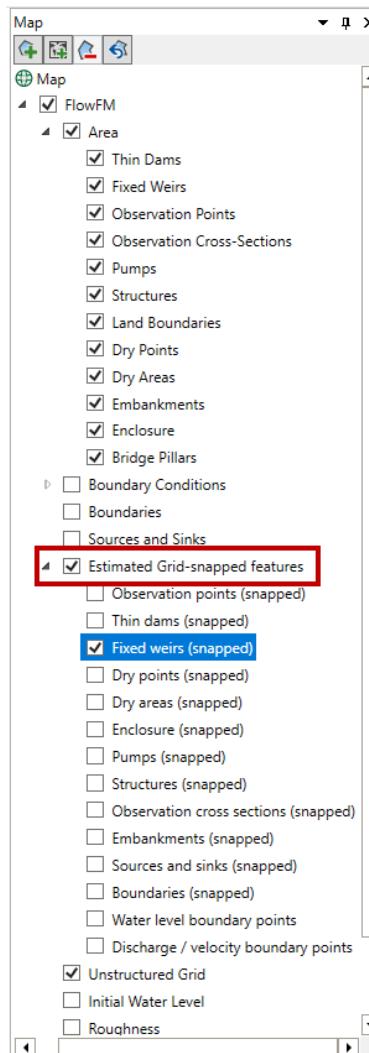


Figure 5.8: Example of expanded Estimated Grid-snapped features attribute in the Map window

Figure 5.8 shows a part of the *Map* window, showing the *Area* and *Estimated Grid-snapped features* attributes. The *x*- and *y*-locations of all spatial model features are shown within the *Area* attribute. You can hide or show any of these attributes by means of clicking the check boxes in front of the attributes. When you enable the Grid-snapped features, all items within the *Area* attribute, as well as all boundaries are interpolated to their corresponding locations on the computational grid. The interpolation is performed instantaneously by the computational core of D-Flow FM, which enables you to directly inspect the numerical interpretation of all

features on the computational grid. Figure 5.9 shows an example of four observation points and one thin dam in the central map, showing both the x - and y -locations of these features as well as their representation on the computational grid.

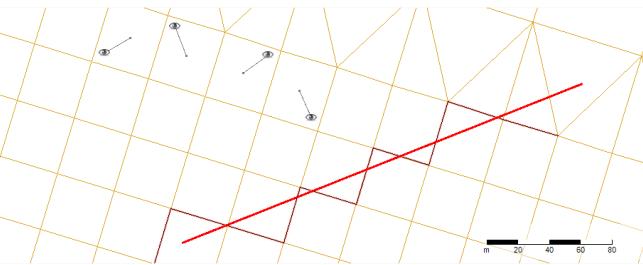


Figure 5.9: Example of grid snapped features displayed on the central map

5.4.2.2 Observation points

Observation points are used to monitor the time-dependent behaviour of one or all computed quantities as a function of time at a specific location, i.e. water elevations, velocities, fluxes, salinity, temperature and concentration of the constituents . Observation points represent an Eulerian viewpoint at the results. (Note: Sediment transport is a β -functionality)



To add an observation point, click the corresponding icon from the *FM Region 2D / 3D* menu in the map ribbon (Figure 5.7). By clicking in the central map, observation points are placed in your model. Selected observation points (first click the *Select* icon from the “Tools” menu in the map ribbon) can be deleted using backspace or directly from the attribute table (explained below). The grid snapped representation (Figure 5.10) is indicated by a line linking the observation points to the closest cell center, indicating that output will be stored of this cell. Importing and exporting of observation points is possible via the context menu of “Observation points” in the *Project* window (right mouse button).

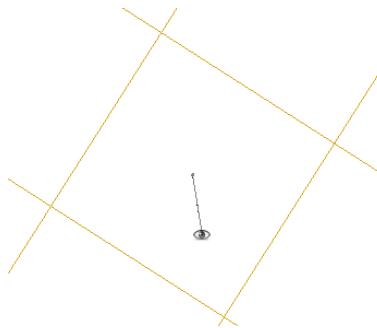


Figure 5.10: Geographical and grid snapped representation of an observation point

When you double click the *Observation points* attribute in the *Project* window, the observation points tab is displayed underneath the central map (Figure 5.11). This tab shows an attribute table with the names, x - and y -locations (in the model coordinate system) of the various observation points within the model. When one of the entries is selected, the corresponding observation point is highlighted in the central map.

Observation Points X				
Name	Group name	X	Y	
ObservationPoint01		315.55	1154.6	
ObservationPoint02		788.67	1293.3	
ObservationPoint03		842.29	1018.9	
ObservationPoint04		448.02	968.46	
ObservationPoint05		583.65	1126.2	

Record 1 of 5 | < << << << >> >> >> + - ▲ ▼ ✓ ✖

Figure 5.11: Attribute table with observation points

5.4.2.3 Observation cross-sections

Cross-sections (Figure 5.12) are used to store the sum of computed fluxes (hydrodynamic), flux rates (hydrodynamic), fluxes of matter (if existing) and transport rates of matter (if existing) sequentially in time at a prescribed interval.

To add a cross section, click the corresponding icon from the *FM Region 2D / 3D* menu in the map ribbon (Figure 5.7). By clicking in the central map, cross section points are added. Note that a cross section consists of a minimum of two points, but an arbitrary amount of intermediate points can be added. The last point is indicated by double clicking the left mouse button. The distance in meters (independent of the local coordinate system) in between the last point and the mouse pointer is indicated in pink; in case more than two points are used, the cumulative length of the entire cross section is shown in black. Once highlighted in the central map, a cross section is deleted with backspace or directly from the attribute table described below. The positive direction through the cross section is indicated by a pink arrow. The direction of this arrow is dependent on the order in which the cross section points are drawn (to change the direction, flip the start and end points). Importing and exporting of cross sections is possible via the context menu of “Observation cross-sections” in the *Project* window (right mouse button).



Figure 5.12: Geographical and grid snapped representation of a cross section

Double clicking the *Observation cross-sections* attribute in the *Project* window enables the *Observation cross-sections* tab underneath the central map view (Figure 5.13). Alternatively, you can double click on any cross section in the map. The attribute table displayed in the tab contains the names of the various cross sections of your model. When one of the entries is selected, the corresponding cross section is highlighted in the central map. A cross section entry can be deleted from the table via the context menu (right mouse button).

Observation Cross-Sections X		
Group name	Name	Δ
	Inlet	
	ObservationCrossSection	
	Outlet	
▶	sluice	

Figure 5.13: Attribute table with observation cross sections

5.4.2.4 Thin dams

Thin dams (Figure 5.14) are infinitely thin objects defined at the velocity points which prohibit flow exchange between the two adjacent computational cells without reducing the total wet surface and the volume of the model. The purpose of a thin dam is to represent small obstacles (e.g. breakwaters, dams) in the model which have sub-grid dimensions, but large enough to influence the local flow pattern. A thin dam is assumed to have an infinite level in the model; no water will ever overflow a thin dam.

To add a thin dam, click the corresponding icon from the *FM Region 2D / 3D* menu in the map ribbon (Figure 5.7). Adding, deleting, importing and exporting of a line feature such as a thin dam is discussed in more detail in section 5.4.2.3 on cross sections.

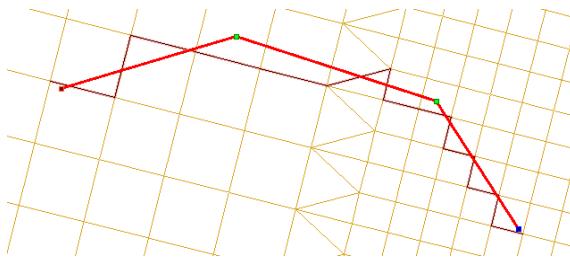


Figure 5.14: Geographical and grid snapped representation of a thin dam

When you double click the *Thin dams* attribute in the *Project* window, the corresponding Thin dams tab appears underneath the central map (Figure 5.15). Alternatively, you can also double click on any thin dam in the central map. Within this tab, an attribute table is shown which displays the names of all thin dams within your model. When one of the entries is selected, the corresponding item is highlighted in the central map. A thin dam entry can be deleted from the table via the context menu (right mouse button).

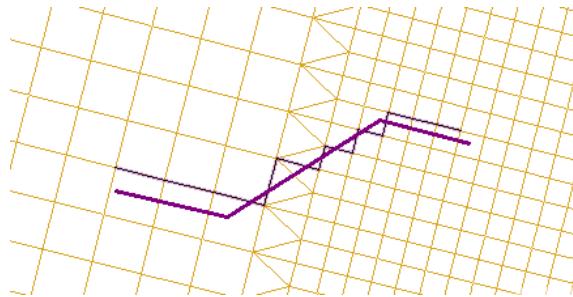
Thin Dams X	
Group name	Name
	ThinDam01
	ThinDam02
▶	ThinDam03

◀◀◀ Record 3 of 3 ▶▶▶ + - ▲▼ × ◀

Figure 5.15: Attribute table with thin dams

5.4.2.5 Fixed weirs

A fixed weir (Figure 5.16) has the same function as a thin dam (section 5.4.2.4). However, unlike a thin dam, a fixed weir can be assigned both xy - and z -values. Furthermore, a fixed weir can be assigned a crest length (a thin dam is infinitely thin). The z -values correspond to the crest level of the fixed weir at the corresponding x - and y -locations; the level can vary in space, but is constant in time (Figure 5.17). Consequently, a fixed weir can overflow if the water level exceeds the crest level of the fixed weir. The level is specified with regard to the same vertical reference level as all other model items with level specifications (e.g. bed level values and initial water levels).

**Figure 5.16:** Geographical and grid snapped representation of a fixed weir**Figure 5.17:** Schematic representation of a fixed weir

To add a fixed weir, click the corresponding icon from the *FM Region 2D / 3D* menu in the map ribbon (Figure 5.7). Adding, deleting, importing and exporting of a line feature such as a fixed weir is discussed in more detail in section 5.4.2.3 on cross sections.

When you double click the *Fixed weirs* attribute in the *Project* window, the corresponding Fixed weirs tab appears underneath the central map (Figure 5.18). Alternatively, you can also

double click on any fixed weir in the central map. Within this tab, an attribute table is shown which displays the names of all fixed weirs within your model. When one of the entries is selected, the corresponding item is highlighted in the central map. A fixed weir entry can be deleted from the table via the context menu (right mouse button).

Fixed Weirs X	
Group name	Name
	FixedWeir01
	FixedWeir02
▶	FixedWeir03

Figure 5.18: Attribute table with fixed weirs

When you double click on a fixed weir in the central map, the fixed weir editor opens in a separate view ([Figure 5.19](#)). On the right, a graphic representation (top view) of the fixed weir is displayed. The support point that is selected in the table is highlighted by means of a blue circle. On the left, a table is displayed showing the following properties of each support point of the fixed weir under consideration:

- ◊ X: x -coordinate of support point
 - ◊ Y: y -coordinate of support point
 - ◊ Crest level: crest level of the support point of support point of a fixed weir
 - ◊ Ground height left: Difference between crest level and toe level at the left side
 - ◊ Ground height right: Difference between crest level and toe level at the right side
 - ◊ Crest width: Width of the crest of a fixed weir in perpendicular direction
 - ◊ Slope left: Slope at left side of support point of a fixed weir
 - ◊ Slope Right: Slope at right side at support point of support point of a fixed weir
 - ◊ Roughness code: Roughness due to vegetation on a support point of a fixed weir

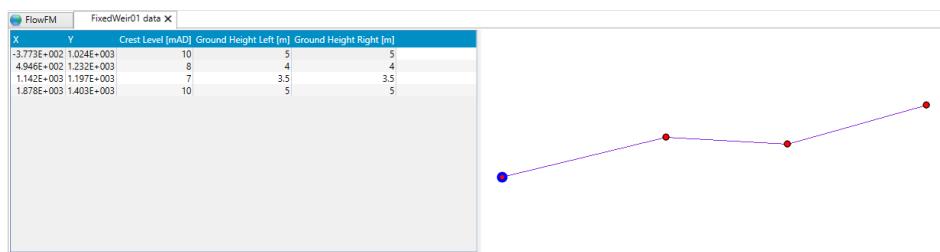


Figure 5.19: Fixed weir editor

5.4.2.6 Land boundaries

A land boundary (Figure 5.20) encloses the main geographic features surrounding your model and indicates the intersection of the water and land masses. When you set up your computational grid in RGFGRID, a land boundary determines the onshore extent of your model. When you open RGFGRID to edit your grid, the land boundary is automatically transferred and displayed. For more details on grid generation, you are referred to the User Manual of RGFGRID (RGFGRID UM, 2016).

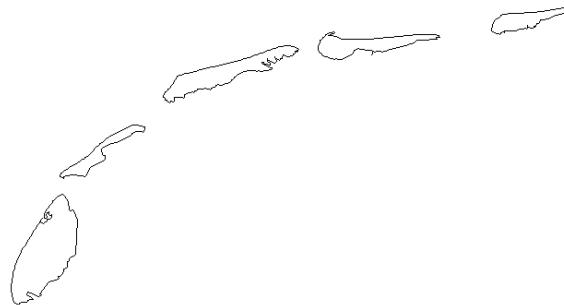


Figure 5.20: Geographical representation of a land boundary

To add a land boundary, click the corresponding icon from the *FM Region 2D / 3D* menu in the map ribbon (Figure 5.7). Adding, deleting, importing and exporting of a line feature such as a land boundary is discussed in more detail in section 5.4.2.3 on cross sections.

When you double click the *Land boundaries* attribute in the *Project* window, the corresponding land boundaries tab appears underneath the central map (Figure 5.21). Alternatively, you can also double click on any land boundary in the central map. Within this tab, an attribute table is shown which displays the names of all land boundaries within your model. When one of the entries is selected, the corresponding item is highlighted in the central map. A land boundary entry can be deleted from the table via the context menu (right mouse button).

Land Boundaries X	
Group name	Name
	Texel
	Vlieland
	Terschelling
	Ameland
▶	Schiermonnikoog

Record 5 of 5

Figure 5.21: Attribute table with land boundaries

5.4.2.7 Dry points and Dry areas

Dry points are grid cells centred around a water level point that are permanently dry during a computation, irrespective of the local water depth and without changing the water depth as seen from the wet points. Dry areas are the same, but provide an easy way of defining many grid points as a single dry area at once.

Note that the flexibility of unstructured grids makes the use of dry points less necessary than with structured grid models, such as Delft3D-FLOW. In the interior of unstructured grids, some or more grid cells can easily be deleted during grid manipulation, e.g., in RGFGRID. Still, a dry points file can be used to explicitly mark locations or regions inside the grid as dry cells.

Dry points in the GUI

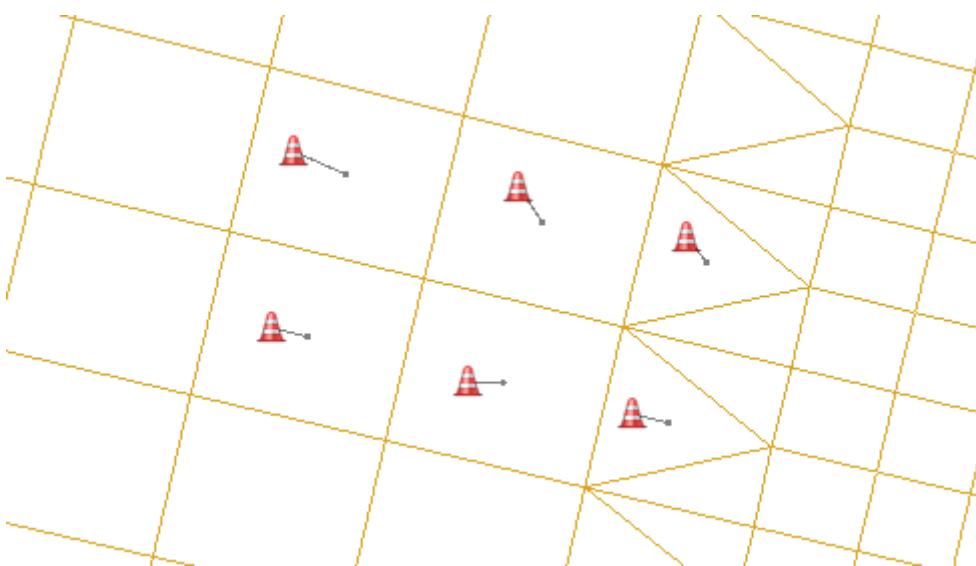


Figure 5.22: Geographical and grid snapped representation of several dry points

To add a dry point, click the corresponding icon from the *FM Region 2D / 3D* menu in the map ribbon ([Figure 5.7](#)). Adding, deleting, importing and exporting of a point feature such as a dry point is discussed in more detail in [section 5.4.2.2](#) on observation points. The grid snapped representation of a dry point ([Figure 5.22](#)) is indicated by a line linking the dry point to the closest cell center.

When you double click the “Dry points” attribute in the *Project* window, the corresponding Dry points tab appears underneath the central map ([Figure 5.23](#)). Alternatively, you can also double click on any dry point in the central map. Within this tab, an attribute table is shown which displays the names of all dry points within your model. When one of the entries is selected, the corresponding item is highlighted in the central map. A dry point entry can be deleted from the table via the context menu (right mouse button).

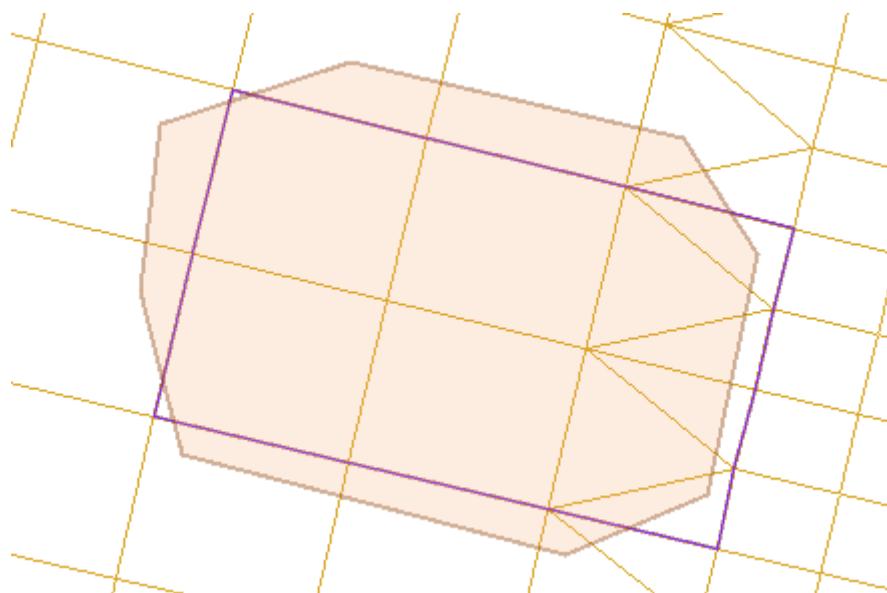
Dry Points X			
	Group name	X	Y
▶		3406	3879.2
◀		2545.3	3860.7
▼		3202.3	3666.3

Record 1 of 3 |◀◀◀◀|◀▶▶▶|+|-|▲|✓|×|◀|

Figure 5.23: Attribute table with dry points

Dry areas in the GUI

Dry areas (Figure 5.24), like dry points, indicate areas that permanently dry during a computation. Instead of adding many separate dry points, you can draw a polygon that encloses all required computational cells. Only cells which centers are strictly inside the polygon are taken into account. The grid snapped representation of the dry area clearly indicates which cells are considered within the dry area.

**Figure 5.24:** Geographical and grid snapped representation of a dry area

When you double click the *Dry areas* attribute in the *Project* window, the corresponding Dry areas tab appears underneath the central map (Figure 5.25). Alternatively, you can also double click on any dry area in the central map. Within this tab, an attribute table is shown which displays the names of all dry areas within your model. When one of the entries is selected, the corresponding item is highlighted in the central map. A dry area entry can be deleted from the table via the context menu (right mouse button).

Dry Areas X		
Name	Group name	
DryArea01		
DryArea02		
DryArea03		

Record 3 of 3

Figure 5.25: Attribute table with dry areas

Dry points file input

Dry points are defined by a sample file <*.xyz>, dry areas are defined by a polygon file <*.pol>. Add the filename to the MDU as below:

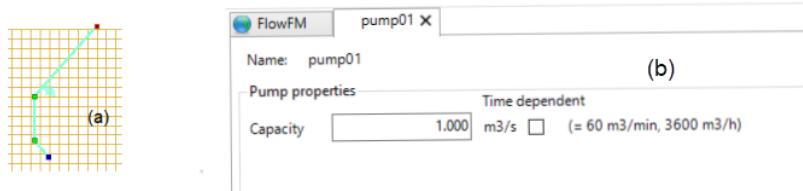
```
[geometry]
# ...
DryPointsFile = <filename.xyz> # Dry points file *.xyz, third column dummy z values,
# or polygon file *.pol.
```

The format of the sample file is defined in [section C.3](#). The format of the polygon file is defined in [section C.2](#). All grid cells that contain a sample point are removed from the model grid, and as a result do not appear in any of the output files. Alternatively, for a polygon file, all grid cells whose mass center lies within the polygon will be removed from the model grid.

5.4.2.8 Pumps

Pumps are a type of structures in D-Flow FM. Unlike the other structures, pumps can force the flow only on one direction. This direction is determined by arrow in D-Flow FM. The direction of pump can be reverted by mouse right-click and selecting "Reverse line(s)".

Like all other structures in D-Flow FM, the pump can be defined by a polygon. The input data of the pumps can be given by selecting and editing the pump polygon (see [Figure 5.26](#)). Right click on the pump polygon and selecting "Delete Selection" leads to deletion of the selected pump. Double clicking the pump polygons (or right click the pump in the list and select "Open view"), it opens a tab for editing the pump properties. The tab includes pump capacity. If the pump capacity is time dependent, it can be given by time series data ([Figure 5.26](#)).

**Figure 5.26:** Polygon for pump (a) and adjustment of physical properties (b).

Right clicking the pumps attribute in the *Project* window opens a pop-down window on which

you can select to import or export pumps. The pumps can be imported as polyline by a <*.pli> file or by a structure file, and they can be exported as a <*.pli> file, structure file, or <shapefile>.

Double clicking the pumps attribute in the *Project* window opens the pumps tab underneath the central map. The attribute table in this tab shows all pumps with their corresponding properties. When one of the pumps is selected, the corresponding item is highlighted in the central map. Double clicking any of the pumps in the central map opens the Structure Editor underneath the central map in which all parameters related to the pump can be set (Figure 5.27).

Name	Long name	Branch	Positive direction	Capacity	Switch-on delivery	Switch-off delivery	Switch-on suction	Switch-off suction	Control on	Group name
pump01			<input checked="" type="checkbox"/>	10	0	0	3	2	Suction and delivery side	
pump02			<input checked="" type="checkbox"/>	12	0	0	3	2	Suction and delivery side	Selected

Figure 5.27: Selection of the pumps

5.4.2.9 Weirs

Unlike the fixed weir, weir (or adjustable weir) can be adjusted based on the user requirements. To set an adjustable weir in the computational domain, you can select the icon *Add new structure (2D)* from the toolbar, and draw a line by mouse. This line includes direction, which defines the sign of total flux passing above the structure (positive flux in the direction of weir, otherwise negative). This direction can be inverted by mouse right-click and selecting *Reverse line(s)*. By double-click on the line, you can define this structure as *Simple weir* and add the geometrical and time-dependent parameters such as *Crest level*, *Crest width*, *Crest level time series* and *Lateral concentration coefficient* (See Figure 5.28).

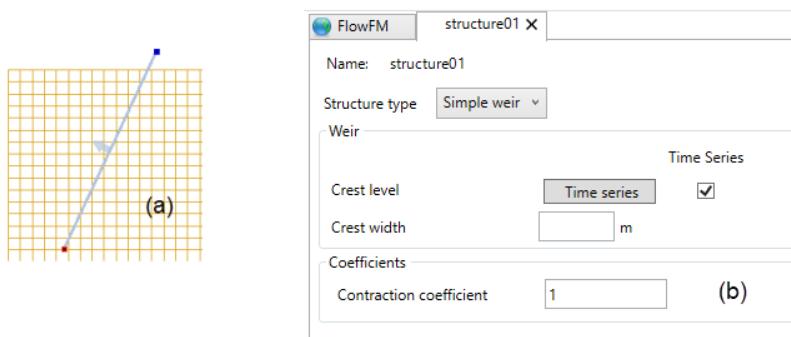


Figure 5.28: Polygon for adjustable weir (a) and adjustment of geometrical and temporal conditions (b).

Moreover, the time series of the crest level can be set in the case the crest level is time dependent. The time dependency diagram can be defined by the help of time series diagram

as shown in [Figure 5.29](#). The time series can also be imported (and exported) from external <csv>-file.

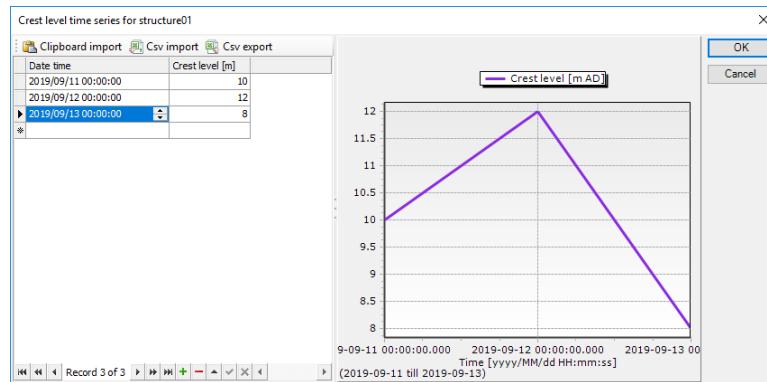


Figure 5.29: Time series for crest level.

The weirs can be deleted, imported and exported. By right clicking on the weir polygon, and selecting "Delete Selection" from the pop-down window, you can delete the selected weir. Right clicking the "Weirs" attribute in the *Project* window opens the "Weirs" tab opens the options for import and export. You can import weirs as polygon (<*.pli> file) or as a structure by structure file. The weirs can also be exported to a polygon file, to a structure data file, or by the help of a shape file.

Double clicking the "Weirs" attribute in the *Project* window opens the "Weirs" tab underneath the central map. The attribute table in this tab shows all weirs with their corresponding properties. When one of the weirs is selected, the corresponding item is highlighted in the central map. Double clicking any of the weirs in the central map opens the Structure Editor as a new map view in which all parameters related to the weir can be set ([Figure 5.30](#)).

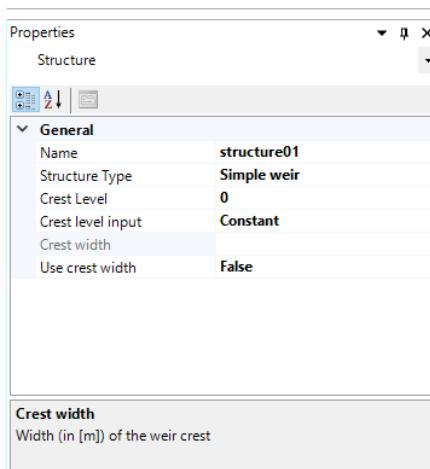


Figure 5.30: Time series for crest level.

5.4.2.10 Gates

In D-Flow FM the gates can be imposed by polygon, and can be edited in a similar way as the other structures (see [Figure 5.31](#)). Like the other structures, mentioned above, the gates can be imported and exported by means of structure file or <*.pli> file.

Figure 5.31 shows the edit tab of the gate properties. The gate can be opened horizontally, as well as vertically.

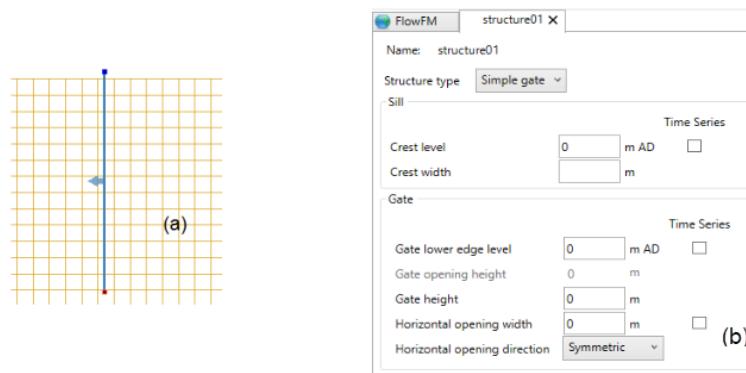


Figure 5.31: Polygon for gate (a) and adjustment of geometrical and temporal conditions (b).

5.4.2.11 Bridge Pillars

Bridge pillars are a type of structures in D-Flow FM that are relevant for the hydrodynamics. In D-Flow FM, a bridge is defined by a polygon on the central map. The user can add a bridge to the model by

- ◊ selecting the *Add new bridge pillar* icon in the *Map* ribbon; and
- ◊ defining the location of the bridge pillars by clicking on the map. The polyline for a bridge is finalized by double-clicking on the last bridge pillar.)
- ◊ repeat the above for another bridge and end by pressing the <ESC> key

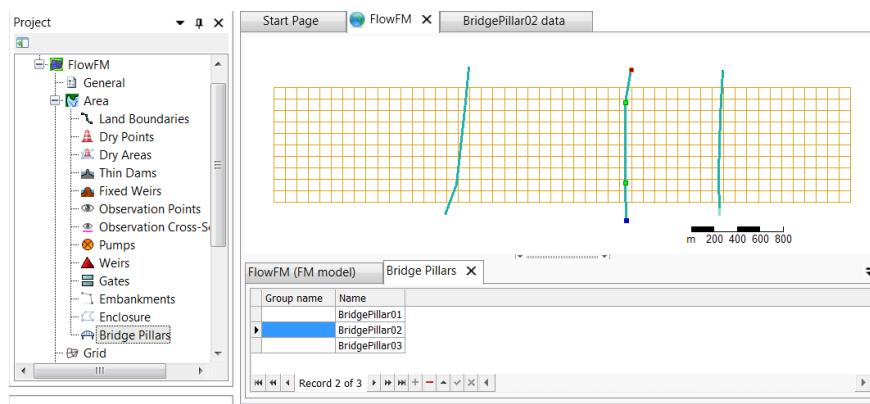


Figure 5.32: Selection of a bridge

Right clicking on the bridge polygon and selecting *Delete Selection* leads to deletion of the selected bridge. After double clicking a bridge polygon or right clicking the bridge in the list and selecting *Open view*, a tab for editing the bridge properties opens (Figure 5.33), with the following parameters:

- ◊ Pillar diameter [m] - (use -999.0 for an auxiliary location)

- ◇ Drag coefficient [-]: default is 1.0



Figure 5.33: Adjustment of the properties of the pillars

Right clicking the *Bridge Pillars* attribute in the *Project* window (Figure 5.32) opens a pop-down window on which you can select to import or export bridges. The bridges can be imported and exported as polyline by a <*.pliz> file or <shapefile>.

As can be seen in Figure 5.33, some of the points of the polygon are relevant pillars with real dimensions and some are not. For example, if a bridge is of curved shape, then additional points can be used to visualize such a bridge, while some of the points of a polyline do not coincide with a pillar. Such points are modelled with a pillar diameter of -999.0.

5.4.3 Computational grid

To set up a grid, click on the *Edit grid* button which opens the program RGFGRID. All features of grid setup in RGFGRID are treated separately in [RGFGRID UM \(2016\)](#). If a land boundary is present in your project, this is exported to RGFGRID automatically. Once you have setup your grid in RGFGRID, click *File → Save Project* and close the program. The grid will now be visible within the central map. Editing of the grid remains possible at any point in time during the setup of your model by means of clicking the *edit grid* button. Any changes you make are always saved after clicking *File → Save Project* and loaded back into the central map.

5.4.4 Bed level

When you double click on *Bed level* in the *Project* window or select *Bed level* from the drop-down box in the spatial editor section of the *Map* ribbon, the spatial editor is activated (Figure 5.34). This editor can be used to generate a bathymetry for your computational grid. How to work with the spatial editor is described in [Appendix H](#). Be aware that the bathymetry in D-Flow FM is defined as the bed level (e.g. positive upward), implying that all bed levels below the reference plane are negative. By default the bed levels are defined on the net nodes.



Note: Please note that, currently, other bed level definition types (e.g. BedlevTypes) are not visually supported by the GUI. If you would like to switch the bed level definitions to another type, you have to set the BedlevType in the *Physical Parameters* tab. However, the bed level locations will not be updated accordingly in the central map.

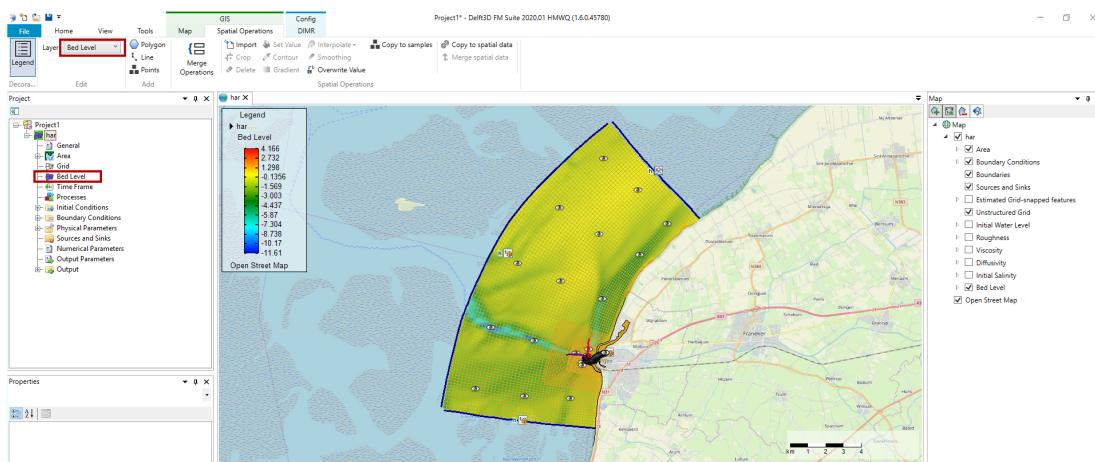


Figure 5.34: Bed level activated in the spatial editor

5.4.5 Time frame

In the settings tab, in the sub-tab time frame (Figure 5.35), you can specify everything related to the time frame in which your model will run.



Figure 5.35: Overview time frame tab

In general, the time frame is defined by a reference date and a start and stop time. The time step size of your model is automatically limited (every time step) based on a Courant condition. In more detail, you must define the following input data:

Max Courant nr

The maximum allowed Courant number, which is used to compute the time step size from the CFL criterium. D-Flow FM uses an explicit advection scheme, therefore a value of 0.7 or lower is advised.

Remark:

- ◊ You should check the influence of the time step on your results at least once.



Reference date

The reference date and time of the simulation. It defines the (arbitrary) $t = 0$ point for all time-series as used in the simulation. In the GUI, time-specifications are always absolute, but in the underlying model input files, these are stored as time values relative to the reference date. Typically, input time-series files are specified in minutes after this $t = 0$ point. See for an illustration Figure 5.36.

The time difference between local time and UTC.

Time zone

The time zone is defined as the time difference (in hours) between the local time (normally used as the time frame for D-Flow FM) and

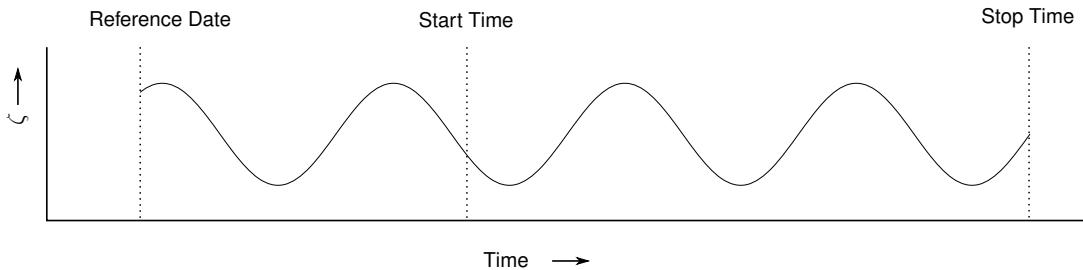


Figure 5.36: Relation between Reference Date and the simulation start and stop time for astronomic- and harmonic-series as used in the simulation. Time-series should cover the simulation time.

Coordinated universal time (UTC). The local Time Zone is used for two processes:

- ◊ To determine the phases in local time of the tidal components when tide generating forces are included in the simulation, see [section 8.10](#).
- ◊ To compare the local time of the simulation with the times at which meteo input is specified, e.g., wind velocities and atmospheric pressure. These can be specified in a different time zone.

User time step

If the *Time Zone* = 0 then the simulation time frame will refer to UTC. The interval that is highest in the hierarchy. It specifies the interval with which the meteorological forcings are updated. The *Max. time step* cannot be larger than the *User time step*, and it will automatically be set back if it is. Also, the output intervals should be a multiple of this *User time step*, see [Appendix F](#). Finally the computational time steps will be fitted to end up exactly at each *User time step*, such that proper equidistant output time series are produced.

Nodal update interval

When using astronomic boundary conditions, the nodal factors can be updated with certain intervals, see [section 8.10](#).

Max. time step

The *Max. time step* is the upper limit for the computational time step. The automatic time step can not be switched off explicitly. (If you want to enforce a fixed time step anyway, set the parameter *Max. time step (s)* to the desired step size, and the parameter *Max. Courant nr.* to an arbitrary high value.)

Initial time step

the initial time step of the model; there is no data available yet during the first time step to compute the time step automatically based on a Courant condition. The computational time step then gradually increases from *Initial time step* to the CFL-number limited time step (assuming that *Initial time step* is relatively small).

The start date and time of the simulation.

Start Time

The stop date and time of the simulation. Always make sure that the model *Stop Time* is larger than the model *Start Time* to avoid errors during your calculation.

Stop Time

5.4.6 Processes

In the processes tab (Figure 5.37) you can specify which processes you want to incorporate into your model. You can choose whether or not to include tidal forcing, salinity, temperature and sediment/morphology by means of check boxes. In addition, you can specify which *Wave model* you want to use. Note that when ticking the sediment/morphology check box, two tabs for setting sediment and morphology parameters appear.

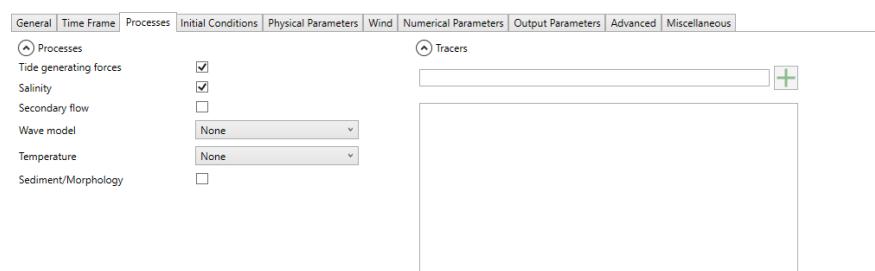


Figure 5.37: Overview processes tab

5.4.7 Initial conditions

When expanding the initial conditions in the *Project* window, all quantities requiring an initial state are shown (Figure 5.38). The number of quantities depends on the activated physical processes in the ‘Processes’ tab (see section 5.4.6). The initial conditions for each quantity can be specified as a uniform value or as a coverage (e.g. a spatially varying field).

The uniform values can be edited in the ‘Initial Conditions’ tab, which opens upon double clicking ‘Initial Conditions’ in the *Project* window (Figure 5.39). In this tab you can also specify the layer distribution for the initial condition specification in case of a 3 dimensional quantity (i.e. salinity). **Note:** Please note that for 3 dimensional initial conditions currently only the option ‘top-bottom’ is supported.



In case of spatially varying initial conditions you can double click the quantity in the *Project* window or select it from the drop-down box in the spatial editor section of the Special Operations ribbon (Figure 5.40). Then the spatial editor is activated, which you can use to edit spatially varying fields. For more information on how to use the spatial editor you are referred to Appendix H. In case of 3 dimensional initial conditions, you can select the layer from the quantity dropdown box in the ‘Map’ ribbon (Figure 5.41).

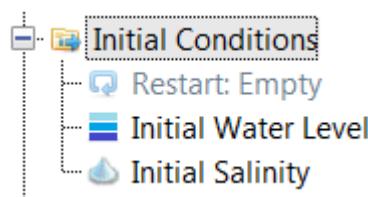


Figure 5.38: Initial conditions in the Project window

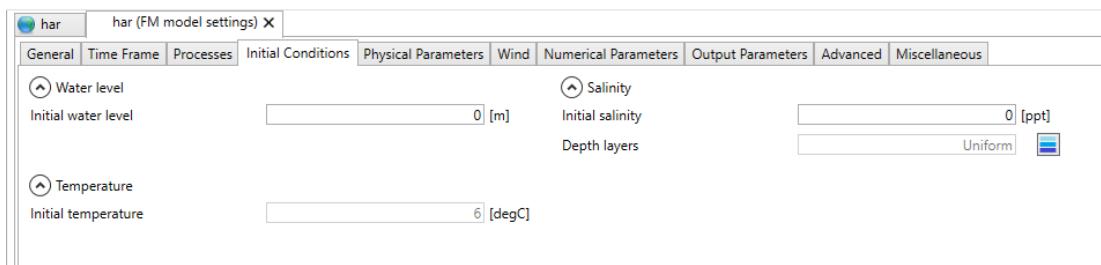


Figure 5.39: The ‘Initial Conditions’ tab where you can specify the uniform values and the layer distributions of the active physical quantities.

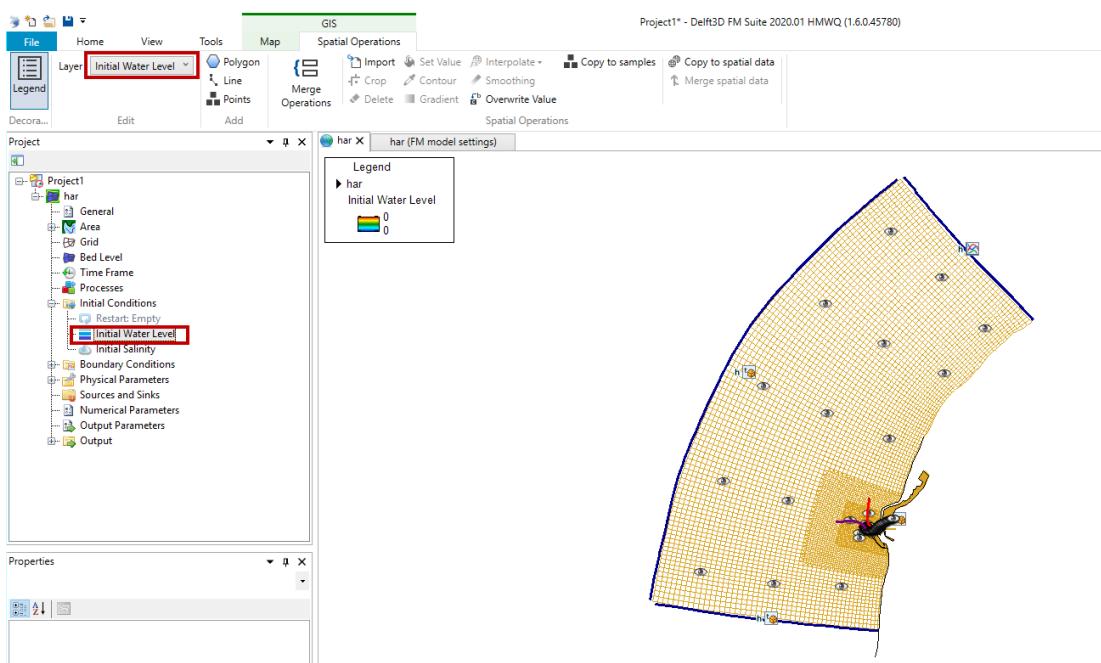


Figure 5.40: Initial water levels activated in the spatial editor

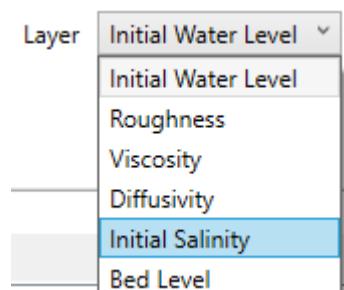


Figure 5.41: Selecting 3 dimensional initial fields from the dropdown box in the ‘Map’ ribbon to edit them in the spatial editor

Instead of defining initial conditions from scratch you can also import fields from a previous computation (using restart files). When you run a model using the Delta Shell GUI which is writing restart files, the restart states will appear in the “Output” folder in the *Project* window (Figure 5.42). For a description of the specification of restart files, please refer to paragraph section 5.4.12. To use a restart file as initial conditions, apply the right mouse button and select “Use as initial state”. The file will now appear under “Initial Conditions” in the *Project*

window (Figure 5.43). To activate the restart file utilize the right mouse button and select “Use restart”. The file will no longer be grey, but is now highlighted in black. The model will now restart from this file. Notice that the simulation still starts at the original *User Start Time*, rather than the time of the restart file. (The restart file only provides initial conditions.)

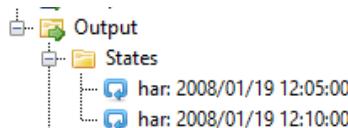


Figure 5.42: Restart files in output states folder

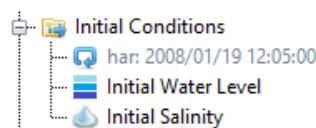


Figure 5.43: Restart file in initial conditions attribute

5.4.8 Boundary conditions

Boundary conditions consist of a location specification ('support points') and a forcing for that location.

Section 5.4.8.1 describes how support points can be specified in Delta Shell. The boundary forcing can be specified in the boundary data editor.

Section 5.4.8.2 describes the functionality of the boundary data editor. Finally, section 5.4.8.4 describes how the user can get an overview of the boundary locations and forcing in the attribute table.

5.4.8.1 Specification of boundary locations (support points)

In D-Flow FM the boundary locations are defined as ‘support points’ on a polyline (<*.pli>). The user can add a boundary polyline (<*.pli>) in the central map by selecting the ‘Add Boundary’ icon in ‘FM Region 2D / 3D’ of the ‘Map’ ribbon (see Figure 5.44). The number of individual mouse clicks determines the number of support points on the polyline. The polyline is closed by a double click. Once the polyline is added it becomes visible in the *Project* window under ‘Boundary Conditions’ (see Figure 5.45). The polyline can be edited by the general edit operations in the ‘Map’ ribbon (i.e. add/delete/move individual geometry points or the complete geometry, see Figure 5.46). The name of the polyline (or ‘boundary’) can be edited in the Boundaries tab, which can be opened by double clicking ‘Boundaries’ in the *Map* window (see Figure 5.47).

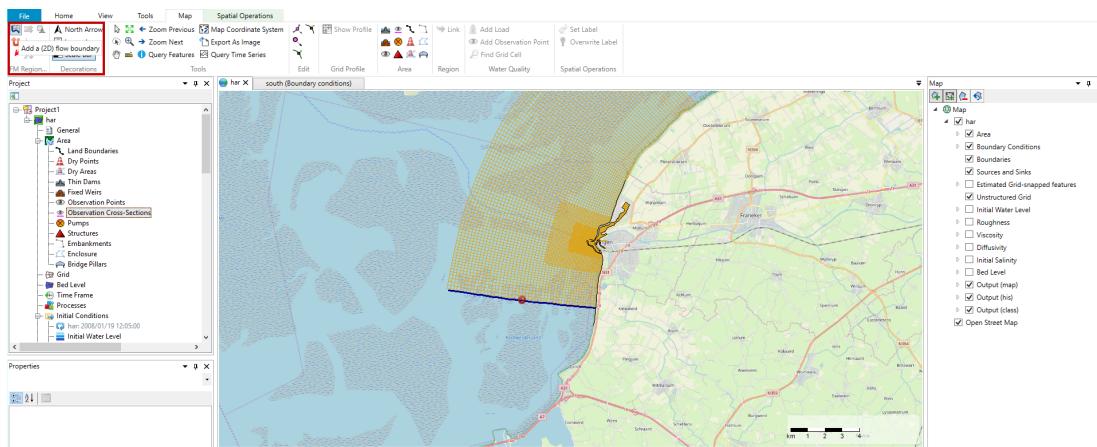


Figure 5.44: Adding a boundary support point on a polyline in the central map. By double clicking on the polyline in the map, the boundary condition editor will open to edit the forcing data on the polyline.

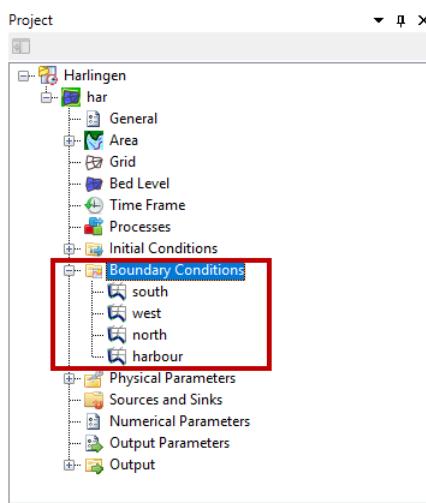


Figure 5.45: Polyline added in Project window under 'Boundary Conditions'. By double clicking on the name of the polyline, the boundary condition editor will open to edit the forcing data on the polyline.

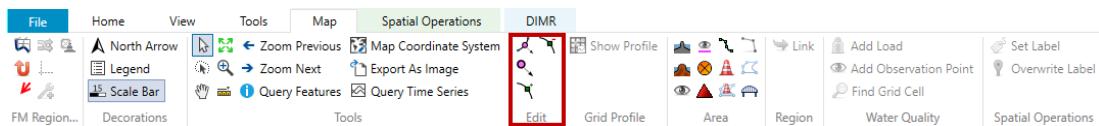


Figure 5.46: Geometry edit options in Map ribbon

Boundaries X	
	Name
▶	south
	west
	north
	harbour

Record 1 of 4

Figure 5.47: Edit name of polyline/boundary in Boundaries tab

5.4.8.2 Boundary data editor (forcing)

The boundary condition editor can be opened either by double clicking on the boundary name in the *Project* window (Figure 5.45) or by double clicking the boundary polyline in the map window (Figure 5.44). An overview of the boundary data editor is given in Figure 5.48. This editor can be used to specify the boundary forcing for different quantities (i.e. water level, velocity, discharge, salinity, etc.) corresponding to different processes (i.e. flow, salinity, temperature, tracers). The user can select the processes and quantities in the upper left corner. In the upper centre panel the user can select the forcing function for the selected quantity (i.e. time series, harmonic components, astronomical components or Q-h relation). The upper right corner contains a list of the support points on the polyline. The geometry view shows the location of the selected support point on the polyline (<*.pli>). In the middle panel are some handy buttons to generate, import and export forcing data. In the lower left panel the user can specify the boundary data for the selected support point. The lower right corner shows the signal of the boundary data. The following sections describe the features of the boundary data editor in more detail.

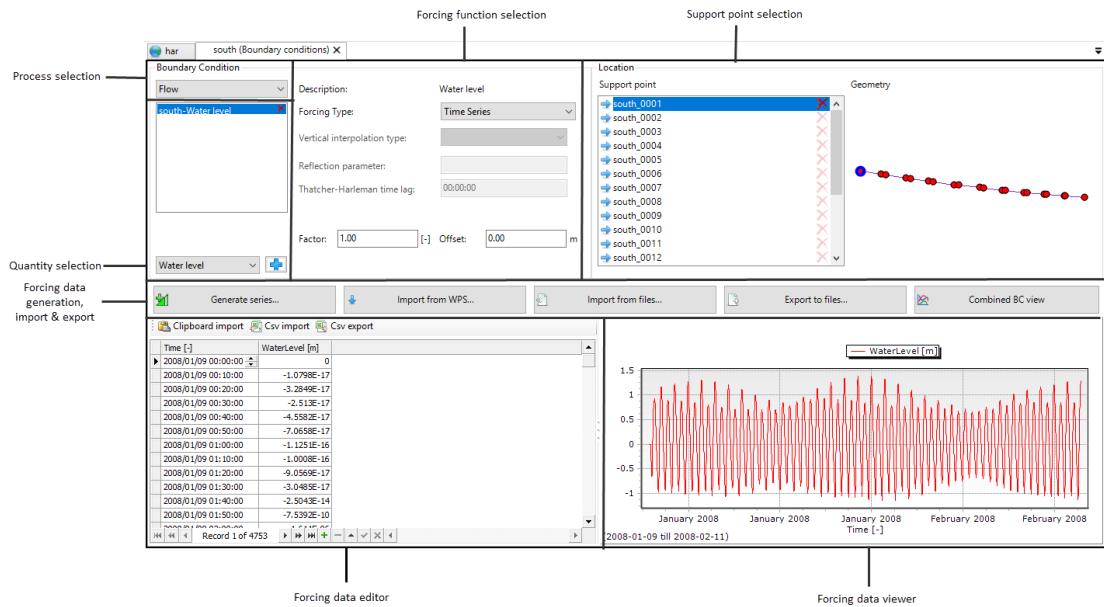


Figure 5.48: Overview of the boundary data editor

Process and quantity selection:



Currently, D-Flow FM supports the processes flow, salinity, temperature and tracers (**Note: tracers are not yet fully editable**). The processes available in the boundary condition editor depend on the selected processes in the processes tab (see [section 5.4.6](#)). After selecting the process from the dropdown box the user can select one of the corresponding quantities, as illustrated in [Figure 5.49](#). For the process flow the user can choose from five principal quantities: water level, velocity, Riemann invariant, Neuman gradient and discharge. All quantities are specified per support point, except for discharges which are specified per polyline (<*.pli>). A support point can have multiple forcing quantities of the same type (i.e. water level, velocity, Riemann, Neumann or discharge). These quantities are added up. This can be relevant to add a surge level to an astronomical water level for example. Furthermore, the user can apply normal and tangential velocities as 'add on' quantities. The following combinations of quantities are allowed:

- ◊ Water level + normal velocity
- ◊ Water level + tangential velocity
- ◊ Water level + normal velocity + tangential velocity
- ◊ Velocity (= normal) + tangential velocity
- ◊ Riemann + tangential velocity

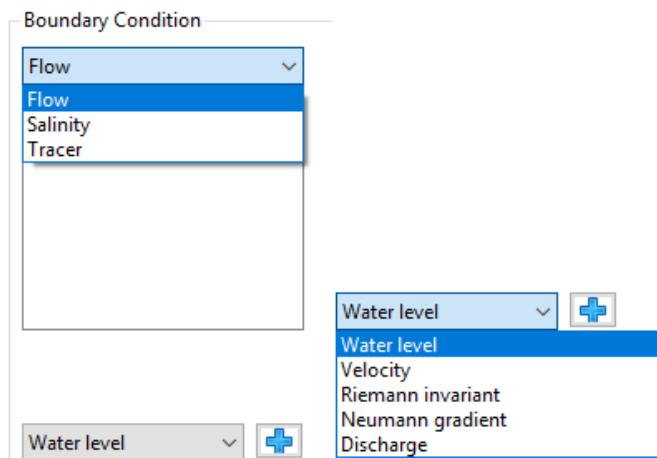


Figure 5.49: Process and quantity selection in the boundary data editor

Forcing function selection:

The user can select one of the following forcing functions:

- ◊ Time series
- ◊ Harmonic components
- ◊ Harmonic components + correction
- ◊ Astronomic components
- ◊ Astronomic components + correction
- ◊ Q-h relation (only for water levels)

The next section describes how data can be added for these types of forcing functions.

Add forcing data to a support point or polyline:

By default all support points on the polyline are deactivated. To add forcing data to a support point the user first needs to activate it by pressing the green 'add'-symbol in the list of support points (see [Figure 5.50](#)). Consequently, the user can specify the forcing data in the lower left panel based on the selected forcing function. Of which a preview is shown in the lower right panel. **Note: Please note that once a support point containing forcing data is made inactive, all data on the support point is lost!**



The user can choose from the forcing functions time series, harmonic components (+ correction), astronomic components (+ correction) and Q-h relation. Examples of the boundary data specifications for these different forcing functions are given below. **Note: Please note that once the user changes the forcing function of a polyline, all data on the polyline is lost!**



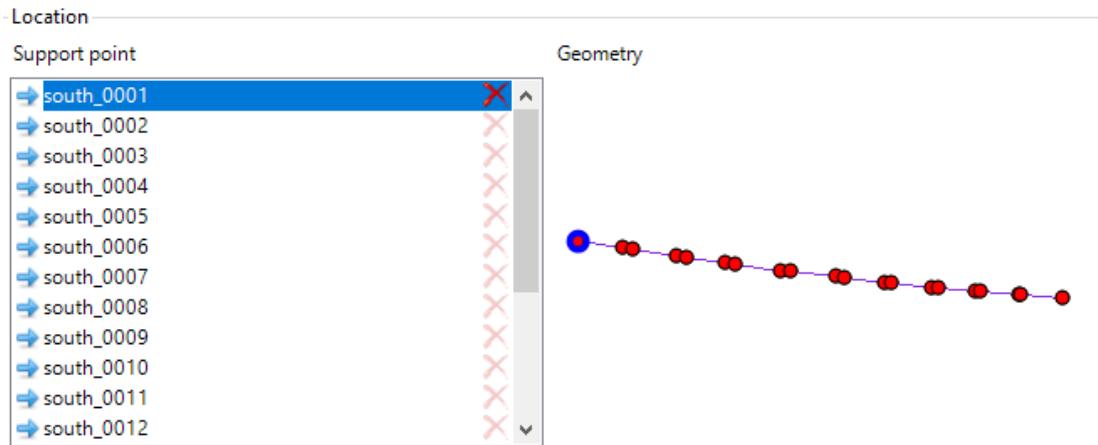


Figure 5.50: Activate a support point

Time series

The time format for time series is yyyy-mm-dd HH:MM:SS. There are multiple ways to specify time series for the selected quantity:

- ◊ Specify the time series step by step in the table (Figure 5.51): the user can add or delete rows with the “plus”- and “minus”-signs below the table.

Time [-]	WaterLevel [m]
2008/02/10 22:40:00	1.0586
2008/02/10 22:50:00	1.1166
2008/02/10 23:00:00	1.1689
2008/02/10 23:10:00	1.2134
2008/02/10 23:20:00	1.2481
2008/02/10 23:30:00	1.2733
2008/02/10 23:40:00	1.2892
2008/02/10 23:50:00	1.2919
2008/02/11 00:00:00	1.2845

Figure 5.51: Specification of time series in the boundary data editor (left panel)

- ◊ Generate time series using the ‘Generate time series’ button (Figure 5.52): the user can specify start time, stop time and time step.

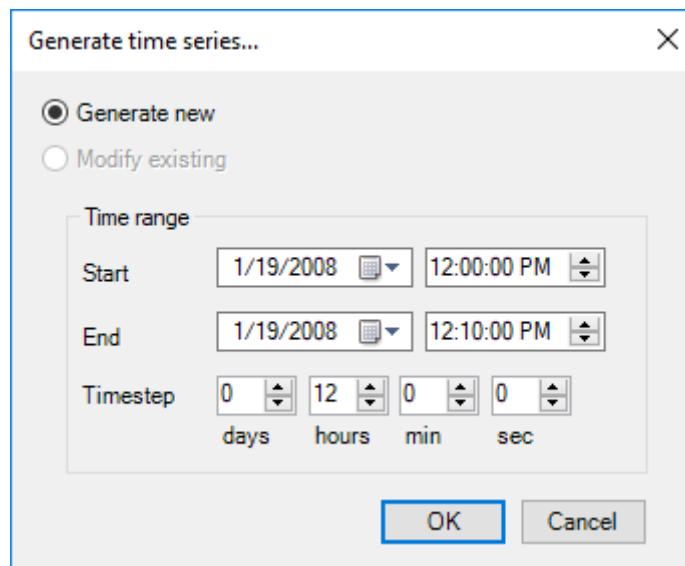


Figure 5.52: Window for generating series of time points

- ◊ Import from csv using the 'Csv import' button: a wizard will open in which the user can (1) select a csv-file (Figure 5.53), (2) specify how data should be parsed into columns (Figure 5.54) and (3) how the values should be parsed and mapped into columns (Figure 5.55).

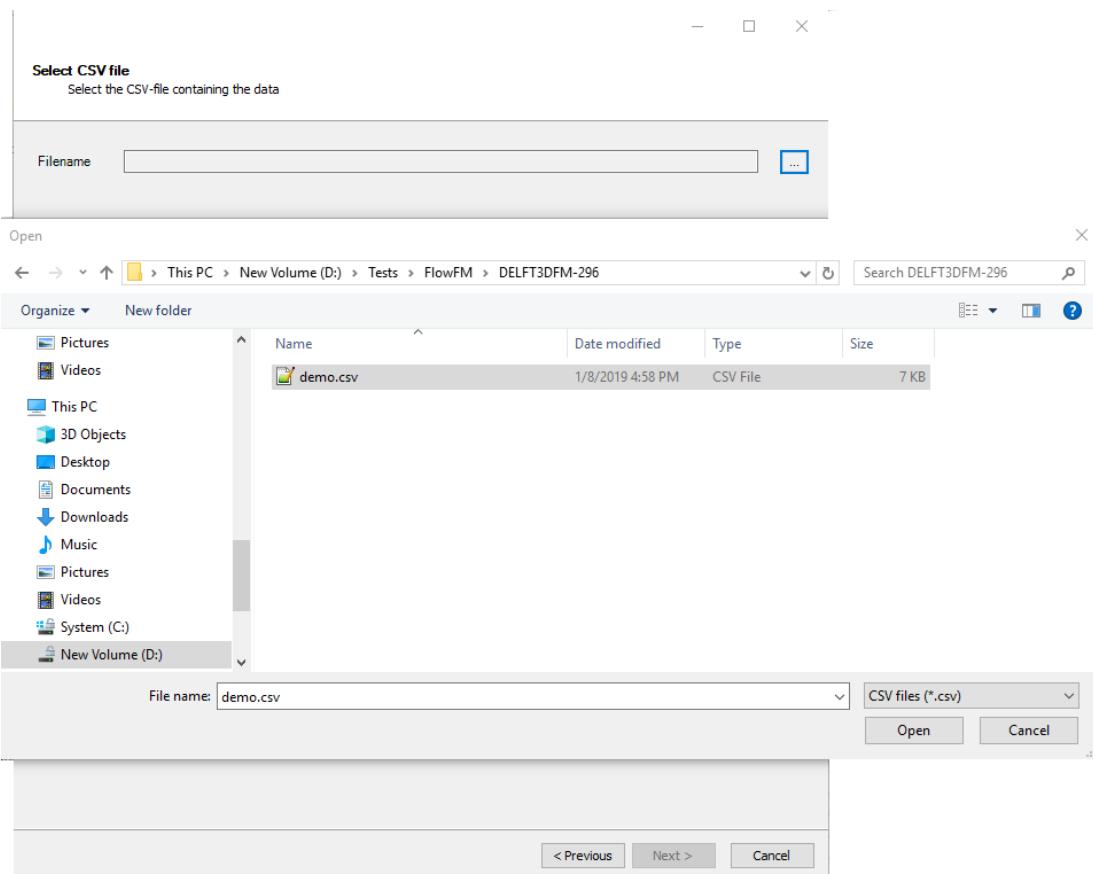


Figure 5.53: Csv import wizard: csv file selection

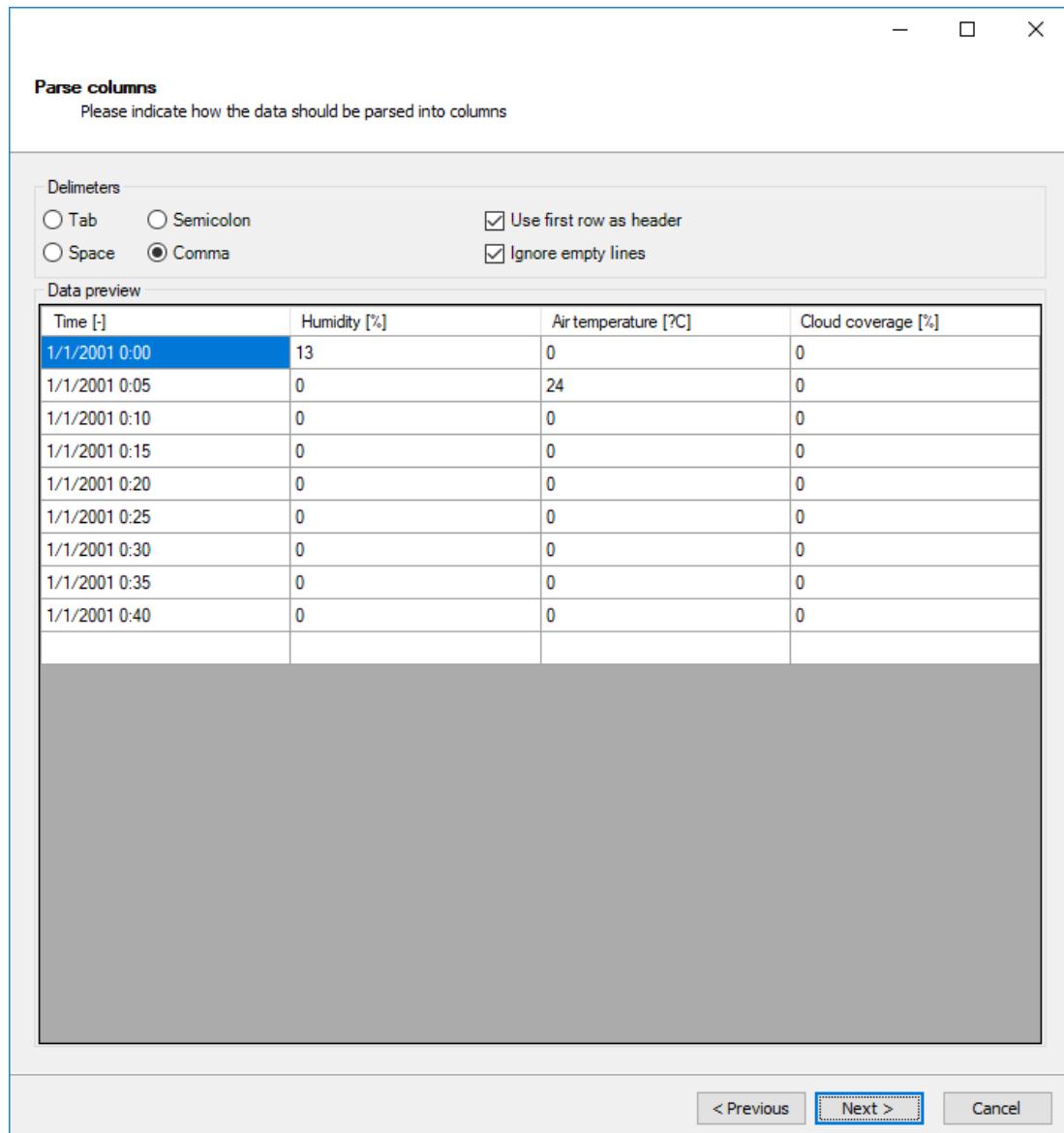


Figure 5.54: Clipboard/csv import wizard: specification of how data should be parsed into columns

Value parsing and column mapping
Please indicate how the values should be parsed and which columns should be used.

Culture info

Date time format Apply

Number format

Column selection

Time

WaterLevel

Filtering

Use filter Filter value Apply

Input preview

	Time [-]	Humidity [%]	Air temperature [?C]	Cloud coverage [%]
▶	1/1/2001 0:00	13	0	0
	1/1/2001 0:05	0	24	0
	1/1/2001 0:10	0	0	0
	1/1/2001 0:15	0	0	0
	1/1/2001 0:20	0	0	0
	1/1/2001 0:25	0	0	0
	1/1/2001 0:30	0	0	0

Result preview

	Time	WaterLevel
▶	2001-01-01	13
	2001-01-01 00:05:00.000	0
	2001-01-01 00:10:00.000	0
	2001-01-01 00:15:00.000	0
	2001-01-01 00:20:00.000	0
	2001-01-01 00:25:00.000	0

< Previous Cancel

Figure 5.55: Clipboard/csv import wizard: specification of how values should be parsed and columns should be mapped

- ◊ Import from clipboard using the ‘Clipboard import’ button: a wizard will open in which the user can specify (1) how data should be parsed into columns (Figure 5.54) and (2) how the values should be parsed and mapped into columns (Figure 5.55).
- ◊ Import from Web Processing Service (WPS): with this service the user can download boundary forcing data (for now only water level time series) for a selected support point from an online database (TOPEX/Poseidon 7.2). Upon pressing the button ‘Import from WPS’ a window will pop up as depicted in Figure 5.56. Here, the user can specify the time interval and time step for downloading the data. (**Note: Please note that this service is only available with an internet connection!**)



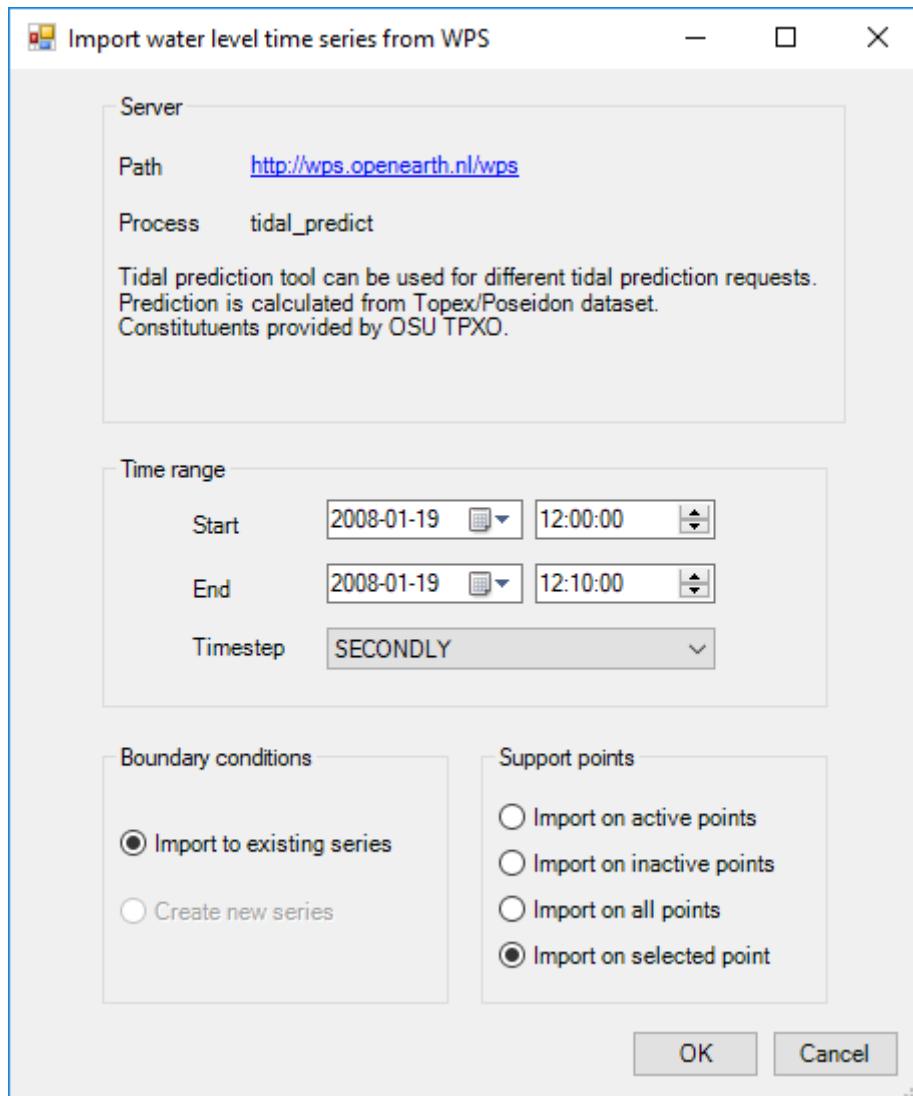


Figure 5.56: Window for entering input to download boundary data from WPS

- ◊ Import from attribute file <*.bc>: with this button the user can import data from existing boundary conditions <*.bc> file. The user has three options for importing:
 - Overwrite where matching (replace): only overwrites the forcing data for matching support points in the <*.bc>-file and GUI input.
 - Overwrite where missing (extend): only overwrites the forcing data for matching support points in the <*.bc>-file and in the GUI input that did not contain data.
 - Overwrite all: overwrites the forcing data for all support points in the GUI (meaning that the forcing data for non-matching support points is emptied).

Harmonic components

The harmonic components are defined by a frequency, amplitude and phase (see [Figure 5.57](#)). By default the forcing data viewer shows the harmonic component for the time frame specified for the model simulation. The options to define the harmonic components are similar to the options for time series:

- ◊ Specify components step by step: the user can add or delete rows with the “plus”- and “minus”-signs below the table.
- ◊ Select (astronomical) components using the ‘Select components’ button ([Figure 5.58](#)): the user can select astronomical components which will be transformed in the corresponding frequencies.
- ◊ Import from csv using the ‘Csv import’ button: a wizard will open in which the user can (1) select a csv-file, (2) specify how data should be parsed into columns and (3) how the values should be parsed and mapped into columns.
- ◊ Import from clipboard using the ‘Clipboard import’ button: a wizard will open in which the user can specify (1) how data should be parsed into columns and (2) how the values should be parsed and mapped into columns.
- ◊ Import from attribute file <*.bc>: with this button the user can import data from existing boundary conditions <*.bc> file. The user has three options for importing:
 - Overwrite where matching (replace): only overwrites the forcing data for matching support points in the bc-file and GUI input.
 - Overwrite where missing (extend): only overwrites the forcing data for matching support points in the bc-file and in the GUI input that did not contain data.
 - Overwrite all: overwrites the forcing data for all support points in the GUI (meaning that the forcing data for non-matching support points is emptied).

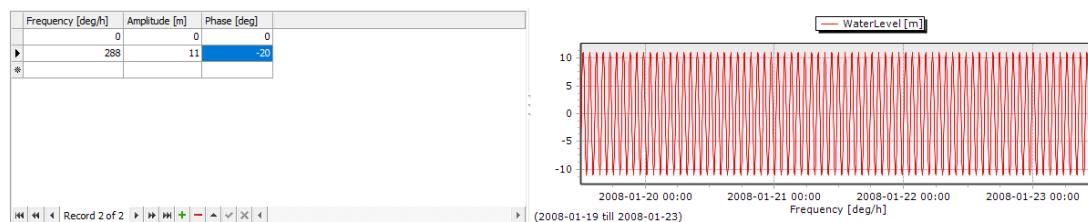


Figure 5.57: Specification of harmonic components in boundary data editor

		Component	Frequency [deg/h]
▶	<input type="checkbox"/>	A0	0.00
	<input type="checkbox"/>	Q1	13.40
	<input type="checkbox"/>	P1	14.96
	<input type="checkbox"/>	O1	13.94
	<input type="checkbox"/>	K1	15.04
	<input type="checkbox"/>	N2	28.44
	<input type="checkbox"/>	M2	28.98
	<input type="checkbox"/>	S2	30.00
	<input type="checkbox"/>	K2	30.08
	<input type="checkbox"/>	M4	57.97
	<input type="checkbox"/>	2(MN)8	114.85
	<input type="checkbox"/>	2(MN)K9	129.89
	<input type="checkbox"/>	2(MN)S6	84.85
	<input type="checkbox"/>	2(MS)8	117.97
	<input type="checkbox"/>	2(MS)K6	87.89
	<input type="checkbox"/>	2(MS)N10	146.41
	<input type="checkbox"/>	2KM(SN)2	30.71
	<input type="checkbox"/>	2KN2S2	28.60

Show periods

Figure 5.58: Selection of astronomical components from list (after pressing ‘select components’)

Astronomic components

Astronomical components are similar to harmonic components, with the exception that the frequency is prescribed. Instead of specifying the frequency the user can select astronomical components by name. Upon editing the component field in the table the user will get suggestions for components in a list (Figure 5.59). The most frequently used components (A0, Q1, P1, O1, K1, N2, M2, S2, K2 and M4) are put on top of the list, the other components are listed in alphabetic order. Instead of defining each component individually, the user can also make a selection of components by pressing the button ‘select components’ (Figure 5.58).

	Component [-]	Amplitude [m]	Phase [deg]
A	A0		
	Q1		
	P1		
	O1		
	K1		
	N2		
	M2		

Figure 5.59: Suggestions for astronomical components in list

Harmonic or astronomic components with corrections

For calibration purposes the user can combine harmonic or astronomic components with corrections. The corrections are defined in terms of an amplitude (multiplication) factor and a phase difference (see [Figure 5.60](#)). This allows the user to keep track of both the original signal and the calibration coefficients. The effects of the corrections on the resulting signal are directly visualized in the forcing data viewer. **Note: Please note that the import functionality is not (yet) working properly for astronomic/harmonic boundary conditions with corrections.**



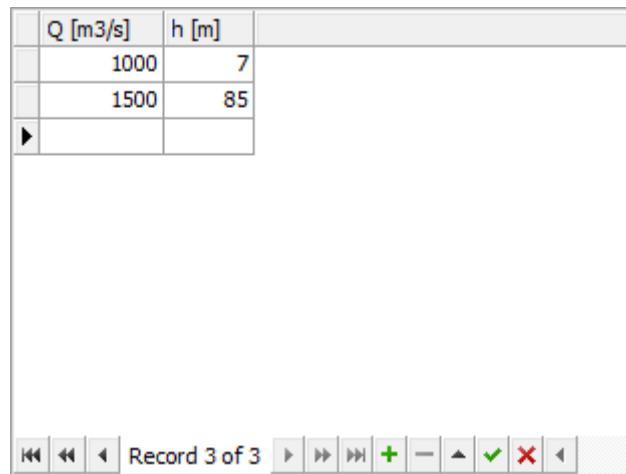
	Component [-]	Amplitude [m]	Phase [deg]	Amplitude corr.	Phase corr. [deg]	
	A0	0.2	0	2	15	
►	M2	1	55	5	30	
*						

Figure 5.60: Editing harmonic/astronomic components and their corrections

Q-h relation (only for water level)

The user can force the boundary with a Q-h relationship, but only for the quantity water level. This is a relationship between discharge and water level (see [Figure 5.61](#)). The relationship can only be prescribed per polyline, not per support point. (**Note:** this functionality has not been tested extensively yet)



**Figure 5.61:** Specification of a Q-h relationship

Exporting boundary conditions

With the *Export to files* button the user can export all boundary forcing data for the given polyline to a <*.bc>-file.

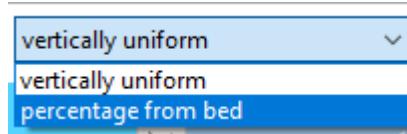
5.4.8.2.1 3D boundary conditions



Note: The 3D-implementation is a β -functionality.

When the model is 3D (i.e. the number of layers is larger than 1), the user can specify 3 dimensional boundary conditions for relevant quantities such as velocity, salinity, temperature and tracer concentration. The user can choose from vertically uniform or vertically varying boundary conditions (Figure 5.62), where the latter are defined as a percentage from the bed. In the boundary condition editor the layer view will appear (Figure 5.63). Here, the user can specify the vertical positions of the boundary conditions and view their position relative to the model layers. Please note that the number and position of vertical boundary conditions does not necessarily have to match the number and/or the exact position of the model layers. The computational core will interpolate the boundary forcing position to the number of model layers.

In case of non-uniform boundary conditions over the vertical, the number of columns in the boundary forcing data editor will increase correspondingly (see Figure 5.64). In this way the user can specify the conditions for all vertical positions in the same table and view the resulting signals in the forcing data viewer.

**Figure 5.62:** Selection of vertically uniform or varying boundary conditions in case of a 3D model

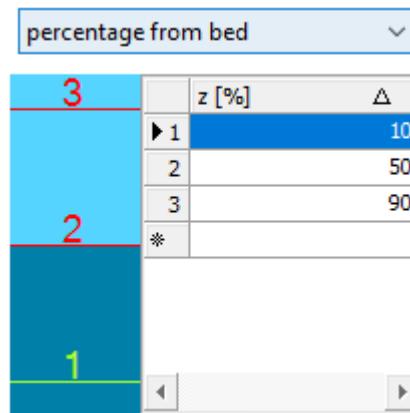


Figure 5.63: Overview of the layer view component of the boundary conditions editor. In the table the user can edit the vertical positions of the boundary conditions as a percentage from the bed. In the view left of the table, the user can see the vertical positions of the boundary conditions (indicated by number corresponding to the table) relative to the model layers.

Time [-]	Salinity(1) [ppt]	Salinity(2) [ppt]	Salinity(3) [ppt]
2008/01/19 12:00:00	30	25	20
2008/02/02 12:00:00	30	25	20
▶			

Figure 5.64: Specification of boundary forcing data (in this example for salinity) at 3 positions in the vertical

View boundary data

All boundary data of the same quantity on a support point can be (pre-)viewed in the boundary data view in the lower-right panel. If multiple signals of the same process have been entered, the viewer will show the active signal in red and the total signal of all datasets in green (see Figure 5.65). In the view the user can zoom-in by dragging a box from top-left to bottom-right and zoom-out by dragging a box from bottom-right to top-left.

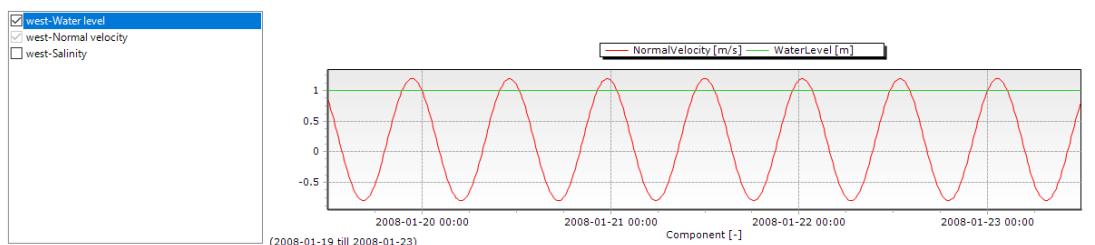


Figure 5.65: Example of active and total signal for multiple water level data series on one support point

To inspect multiple quantities at a support point at the same time (for example water level and normal velocity) the user can use the combined boundary data viewer by pressing the button 'Combined BC view'. **Note: This does not work properly yet.**



5.4.8.3 Import/export boundary conditions from the *Project* window

Apart from import and export functionality per individual boundary polyline in the boundary condition editor, the GUI offers the opportunity to import and export boundary locations (`<*.pli>`) and forcing (`<*.bc>`) on a higher level. Hereto, you have to click the right mouse button on “Boundary Conditions” in the *Project* window and select “Import” or “Export” (Figure 5.66). Imports and exports on “Boundary Conditions” apply to all the boundary conditions whereas import and exports on a boundary polyline apply only to that boundary condition.

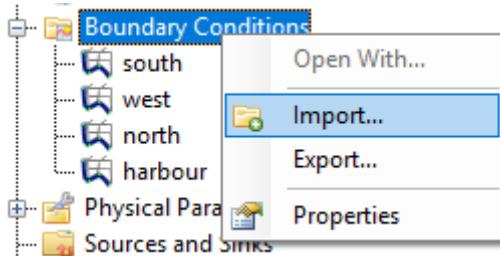


Figure 5.66: Importing or exporting boundary features — both polylines `<*.pli>` and forcing `<*.bc>` — from the *Project* window using the right mouse button

Import and export polylines

Upon importing a `<*.pli>`-file with the same filename and the same polyline name(s) as the existing polyline names in the GUI, the existing polyline(s) will be replaced and all forcing data thereon will be deleted. Upon importing a polyline(s) with a different name(s), the polyline(s) will be added to the *Project* window without any forcing data on it/them. The user will be asked to import the data “as is” or to perform a coordinate transformation before the import (see Figure 5.67).

Alternatively, the user can export created polylines to a `<*.pli>`-file. Upon export the user will be asked to export the data “as is” or to perform a coordinate transformation before the export (see Figure 5.67).

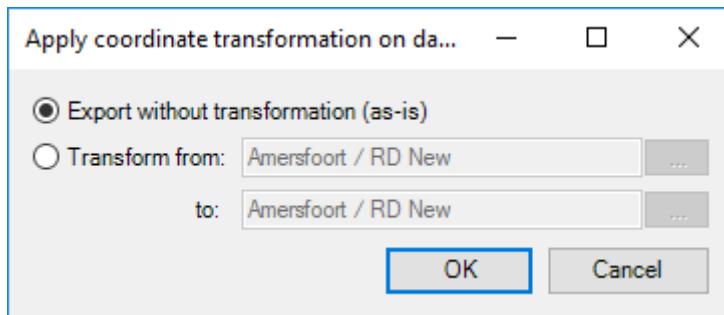


Figure 5.67: Import or export a `<*.pli>`-file as is or with coordinate transformation.

Import and export boundary forcing data

Similar to the polylines, you can import and export boundary forcing data from/to a <*.bc>-file. To import forcing data the existence of a polyline with at least one matching support point is a prerequisite. Upon importing <*.bc> data you can select which quantities and forcing types from the <*.bc>-file should be imported and with which overwrite options (see Figure 5.68). Similarly, you can export boundary forcing data. As an additional exporting feature you can select whether you would like to export: 1) all forcing data into one file, 2) as separate files per boundary, 3) as separate files per process or 4) as separate files per quantity (see Figure 5.69).

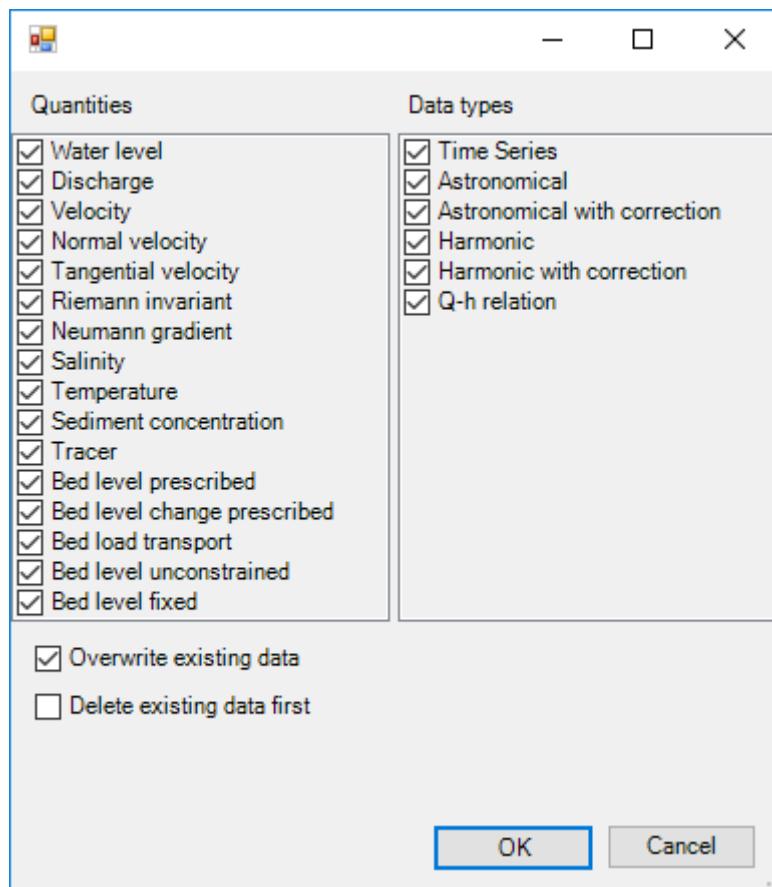


Figure 5.68: Import or export a <*.pli>-file as is or with coordinate transformation.

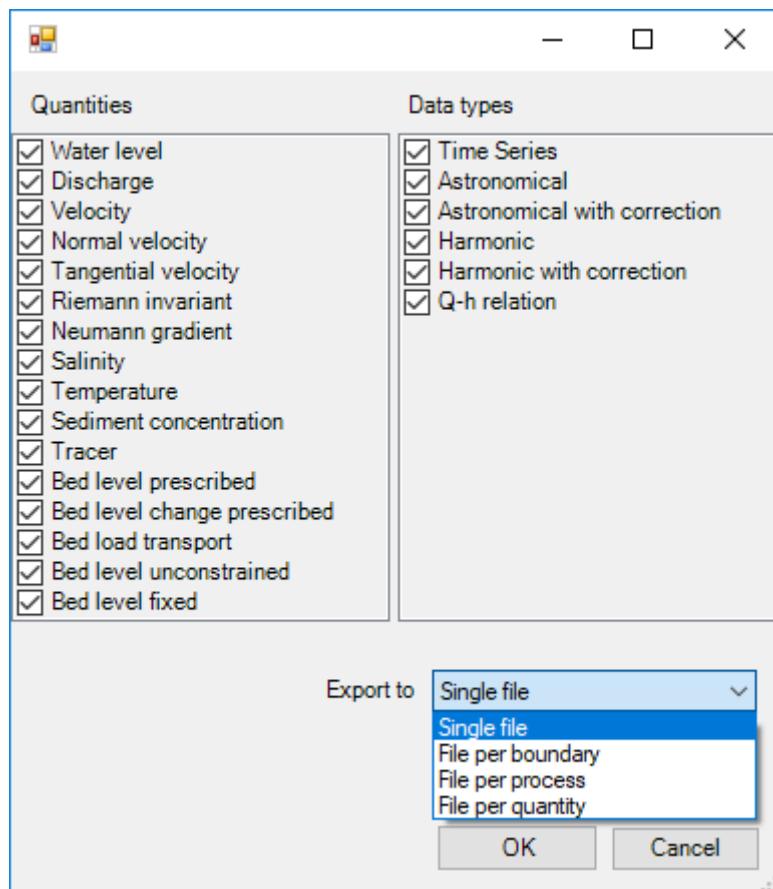


Figure 5.69: Import or export a *.pli file as is or with coordinate transformation.

5.4.8.4 Overview of boundary conditions in attribute table (non-editable)

The attribute table of the boundary conditions gives an overview of all specified boundary polylines and corresponding forcing (see [Figure 5.70](#)). This attribute table can be opened by double clicking ‘Boundary Conditions’ from the project or *Map* window. Most of the features in the attribute table are non-editable, except for the optional (multiplication) factor and offset per quantity per polyline. With these settings you can integrally multiply all data on a polyline with a factor and/or add an offset to it.

Boundary	Quantity	Forcing type	Factor	Offset	
+ south	WaterLevel	Time Series	1	0	
+ south	Salinity	Time Series	1	0	
+ west	WaterLevel	Time Series	1	0	
+ west	NormalVelocity	Astronomical	1	0	
+ west	WaterLevel	Time Series	1	0	
+ west	Salinity	Time Series	1	0	
+ north	WaterLevel	Q-h relation	1	0	
+ north	Salinity	Time Series	1	0	
+ harbour	Salinity	Time Series	1	0	
► + harbour	Discharge	Time Series	1	0	

Figure 5.70: Overview of all boundary conditions in attribute table

5.4.9 Physical parameters

The physical parameters attribute is used to set all physical parameters of your model. When it is expanded in the *Project* window, it shows four attributes; roughness, viscosity, diffusivity and wind.

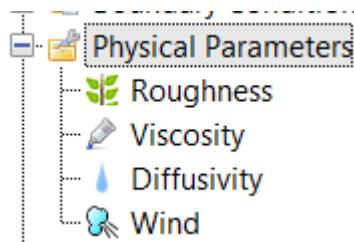


Figure 5.71: The physical parameters in the Project window

5.4.9.1 Constants

In the *Physical Parameters* tab, the user can adjust the value of *Gravity* [m/s^2] under *Others* and *Default water density* [kg/m^3] under *Density*.

5.4.9.2 Roughness

Bed roughness can be specified as a uniform value or as a coverage (e.g. a spatially varying field). The uniform values as well as the roughness formulation (i.e. Chézy, Manning, White-Colebrook or Z_0) can be edited in the ‘Physical Parameters’ tab, which opens upon double clicking ‘Roughness’ in the *Project* window (Figure 5.72). **Note:** The latter is not yet working. In this tab you can also specify the linear friction coefficient, linear friction Umod, wall behaviour (free slip or partial slip) and wall ks for partial slip.



In case of spatially varying bed roughness you can double click the quantity in the *Project* window or select it from the dropdown box in the *Spatial Operations* ribbon (Figure 5.73). Then the spatial editor is activated, which you can use to edit spatially varying fields. For more information on how to use the spatial editor you are referred to Appendix H.

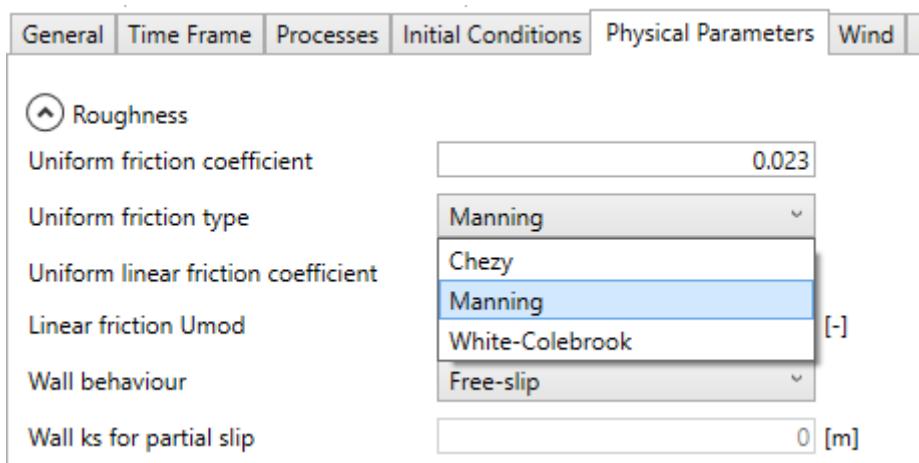


Figure 5.72: The section of the ‘Physical Parameters’ tab where you can specify roughness related parameters and formulations.

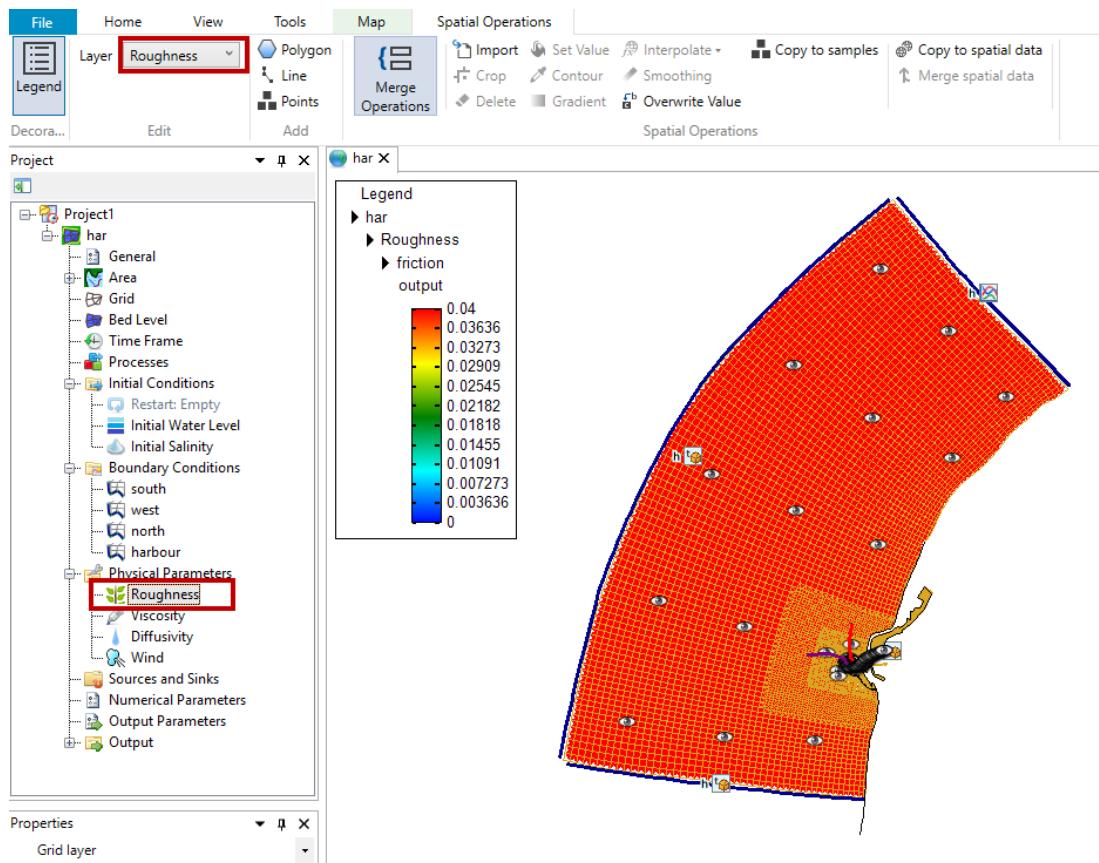


Figure 5.73: Roughness activated in the spatial editor to create/edit a spatially varying field

5.4.9.3 Viscosity

The eddy viscosity can be specified as a uniform value or as a coverage (e.g. a spatially varying field). In the ‘Physical parameters’ tab you can specify the uniform values for the horizontal and vertical (in case of 3D simulations) eddy viscosity and diffusivity (Figure 5.74).

In case of spatially varying viscosity you can double click the quantity in the Project window or select it from the dropdown box in the *Spatial Operations* ribbon (Figure 5.75). Then the spatial editor is activated, which you can use to edit spatially varying fields. For more information on how to use the spatial editor you are referred to Appendix H.

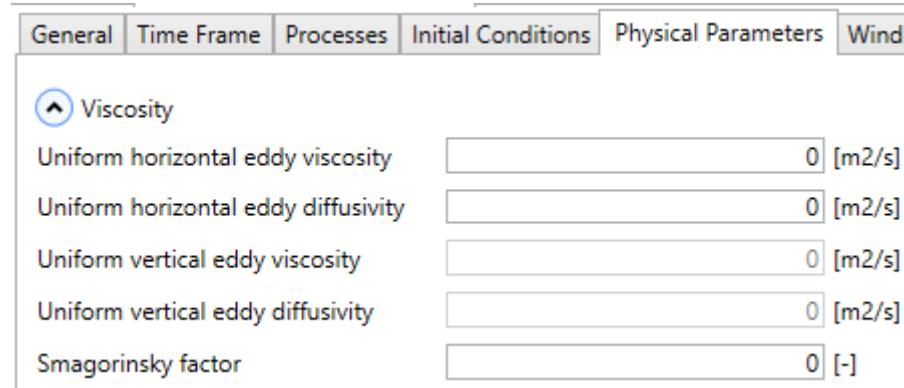


Figure 5.74: The section of the ‘Physical Parameters’ tab where you can specify (uniform) values for the horizontal and vertical eddy viscosity and diffusivity.

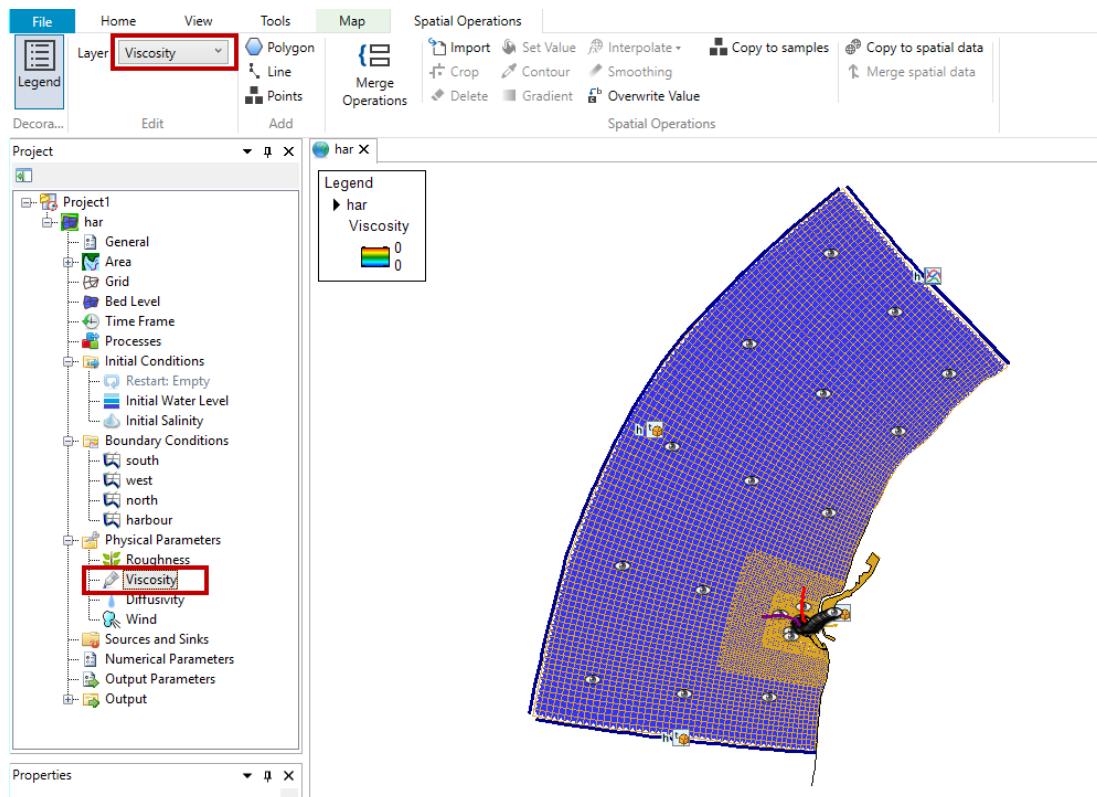


Figure 5.75: Viscosity activated in the spatial editor to create/edit a spatially varying field

5.4.9.4 Wind

All relevant parameters, described in chapter 12 can be adjusted in the following sub-tab.

Parameter	Value	Unit
Wind drag coefficient type	Smith & Banks (2 break points)	
Break points wind drag coefficient	0.00063 0.00723	
Break points wind speed	0 100	
Air density	1.2	[kg/m ³]
Average air pressure on bounds	0	[N/m ²]
Average initial air pressure	0	[N/m ²]

Figure 5.76: Overview of parameters in sub-tab Wind

5.4.9.5 Heat Flux model

The Heat flux model, described in chapter 11, needs only specification of which model to use. See the drop-down selection in Figure 5.37.

5.4.9.6 Tidal forces

The Tide generating forces, described in section 8.10, can be enabled in the Processes tab, see Figure 5.37.

5.4.10 Sources and sinks

Sources and sinks (or: intake/outfall facilities) can be used to add/extract a discharge to/from the model or to redistribute water and constituents (such as temperature and salinity) within the model. Sources and sinks consists of a location (defined by a `<*.pli>`-file) and time series describing the discharges (defined by a `<*.tim>`-file). All the hydrodynamical considerations behind sources and sinks are discussed in section 8.8.

Sources and sinks locations

Sources and sinks can be added to the model using the corresponding icon from the Map ribbon (Figure 5.77). When the sources/sinks icon is active you can add them as polyline elements in the central map using the left mouse button. Each polyline element is closed by double clicking the left mouse button.



Note: Please note that the length of the polyline elements is not taken into account in the handling of sources and sinks (e.g. it is modeled as an instant redistribution of water and constituents without delays and friction losses).

Polyline elements starting outside the model domain and going inward are sources and, vice versa, polyline elements starting inside the model domain and going outward are sinks. Polyline elements starting and ending within the model are intake/outfall type of discharges. The drawing direction determines the direction of the discharge indicated by an arrow (Figure 5.78). In case of a source the direction of the last polyline element determines the direction of flow momentum into the model.

Note: The ‘reverse’ line option — to switch the discharge direction without having to redraw the source/sink — is not implemented.

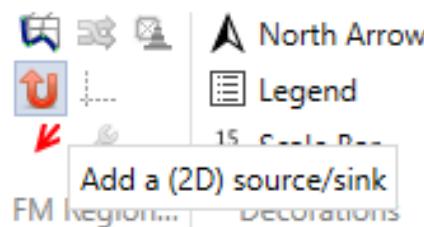


Figure 5.77: Activate the sources and sinks editing icon in the Map ribbon

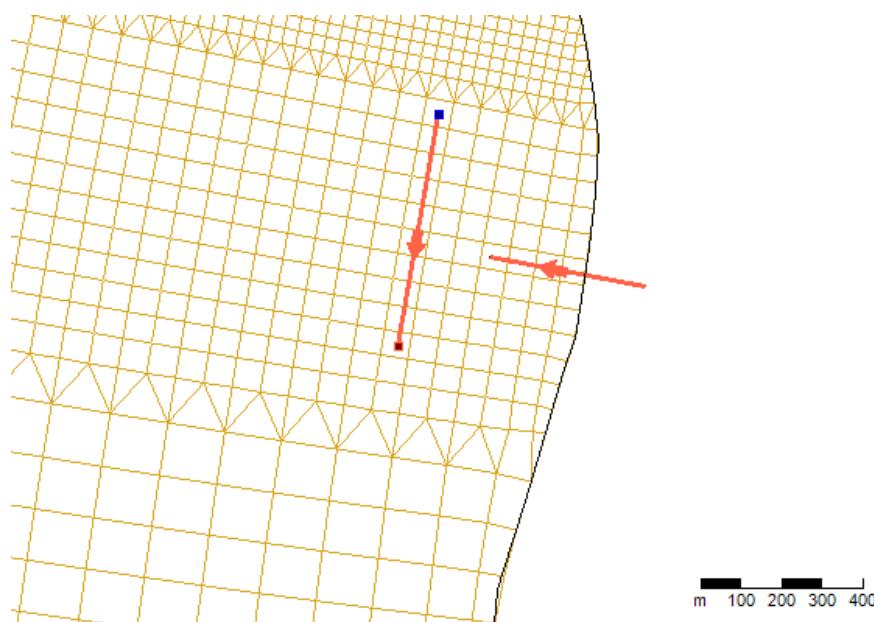


Figure 5.78: Add sources and sinks in the central map using the ‘Sources and sinks’ icon.

Sources and sinks time series

After drawing the sources/sinks locations in the central map, they appear under ‘Sources and sinks’ in the *Project* window (Figure 5.79). Either by double clicking the sources/sinks in the *Project* window or the line element in the central map, you can open the sources and sinks editor (Figure 5.80). In this editor you can specify the discharge time series as well as the corresponding constituents (for example salinity and temperature, depending on the active physical processes). The time series of water discharges are always defined as absolute values. For the time series of constituents the following applies:

- ◊ For sources the constituent time series are absolute values
- ◊ For sinks the constituent time series are determined by the modeled values

- ◇ For sources and sinks (intake/outfall relationships) the constituent time series are the excess values (e.g. on top of the modeled values)

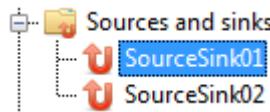


Figure 5.79: Sources and sinks appearing in the Project window

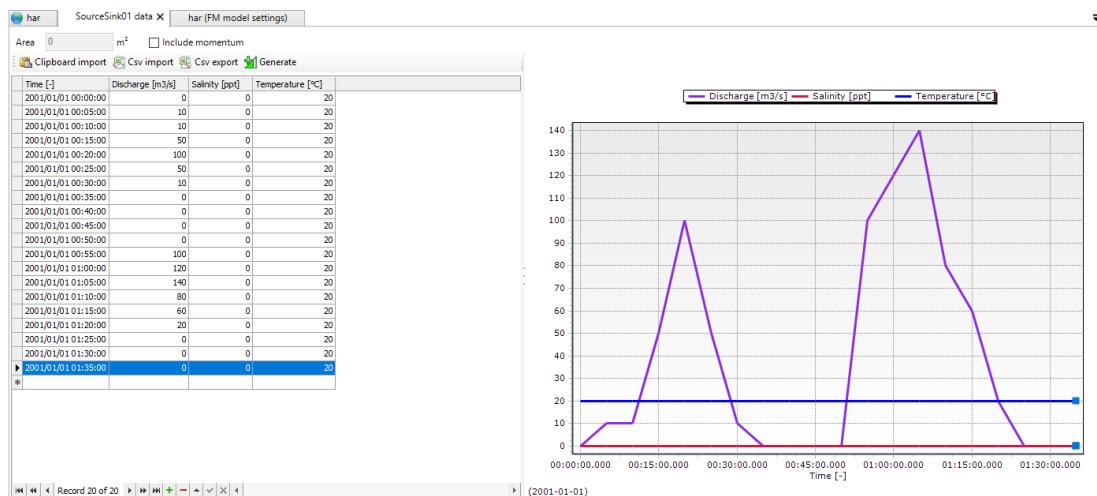


Figure 5.80: Specifying time series for sources and sinks in the sources and sinks editor

5.4.11 Numerical parameters

In the numerical parameters tab, all numerical parameters related to your computation can be set. The parameters that can be set are described in [Table 5.1](#).

5.4.12 Output parameters

Model runs can produce various types of output files. The **Map** window shows the two most-used types: (*map* and *his(tory)* output ([Figure 5.81](#)).

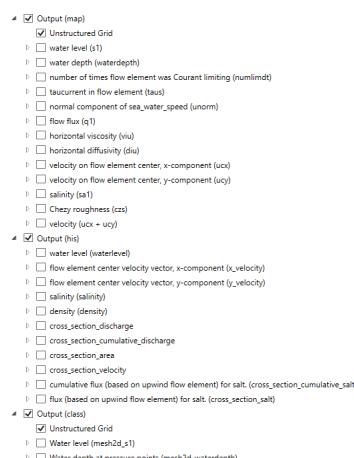


Figure 5.81: Output Parameters tab

Table 5.1: Overview and description of numerical parameters

Parameter	Description
Max Courant number	The size of the time step, Δt , is computed by the computational kernel automatically, each time step by means of the maximum tolerable Courant number. By default, this value is 0.7.
Wave velocity fraction	The wave velocity fraction is related to stability of the computation. By using this fraction, the velocity of the flow is enlarged with the wave velocity times this fraction. The value of this fraction is 0.1 by default.
Advection type	This key depicts the ID of the advection scheme. By default this value is 33.
Water depth limiter type	The limiter type for waterdepth in continuity equation: 0 means no limiter (default), 1 is the minmod method, 2 the Van Leer method, 3 the Koren method and 4 the monotone central method.
Advection velocity limiter type	The limiter type for the cell center advection velocity: 0 means no limiter, 1 the minmod method, 2 the Van Leer method, 3 the Koren method and 4 the monotone central method (default).
Salinity transport limiter type	The limiter type for salinity transport: 0 means no limiter, 1 the minmod method, 2 the Van Leer method, 3 the Koren method and 4 the monotone central method (default).
Solver type	Specification of the Poisson equation type solver for the pressure: 1 = sobekGS_OMP, 2 = sobekGS_OMPthreadsafe, 3 = sobekGS, 4 = sobekGS + Saadilud (default), 5 = parallel/global Saad, 6 = parallel/Petsc, 7 = parallel/GS.
Max degree in Gaussian elimination	The maximum degree in the Gauss elimination, part of the pressure solver. This key has the value 6 by default.
Thin dike scheme	Or: fixed weir scheme. 0: none, 1: compact stencil, 2: whole tile lifted, full subgrid weir + factor.
Thin dike contraction	Or: fixed weir contraction. This is the fixed weir flow width contraction factor, being the flow width = flow width times the fixed weir contraction.
Boundary smoothing time	Fourier smoothing time on waterlevel boundaries (s).
Linear continuity	Default 0; Set to 1 for linearizing $d(H_u)/dx$; link to AdvecType.
Threshold for drop losses	Apply droplosses only if local bed slope is larger than this specific value.
Theta of time integration	Compromise in explicit/implicit time integration.
Downwind cell H on Q boundaries	Specifies the way of the upstream discharge boundary: 0 is original hu on qbnd, 1 is downwind hs on qbnd

The history file contains output on specific locations: time series data on observation points ([section 5.4.2.2](#)), cross-sections ([section 5.4.2.3](#)) and structures (Sections [5.4.2.8 – 5.4.2.10](#)). The map file contains flow quantities on the entire grid at specified time intervals, and can later be used for 2D and 3D visualizations of entire flow fields. The map file can typically turn out much larger than the history file, and it is therefore advised to use larger time intervals for map files than for his files.

Additionally, three more types of output can be requested: restart files, WAQ output, and timing statistics. Restart files are a special type of map file that can later be used as initial states in other runs. Restart files contain several additional flow quantities and are written at specified intervals into one file per each restart time. Typically, one selects a large restart interval in order not to waste disk space. WAQ output files are written by D-Flow FM and are intended to be used as input files to subsequent D-Water Quality runs. More details on water quality modelling can be found in [chapter 18](#). Timing statistics can be produced both in the diagnostics file (via *Statistics output interval*, for viewing basic simulation progress), and in a separate detailed timings file (via *Timing statistics output interval*, for detailed performance analysis).

When double clicking the *Output Parameters* in the *Project* window, the *Output Parameters* tab is highlighted below the central map. All parameters related to the output of your model run are specified here ([Figure 5.82](#)). The most common output parameters to set are the parameters related to the water quality files, history files, map files and restart files.

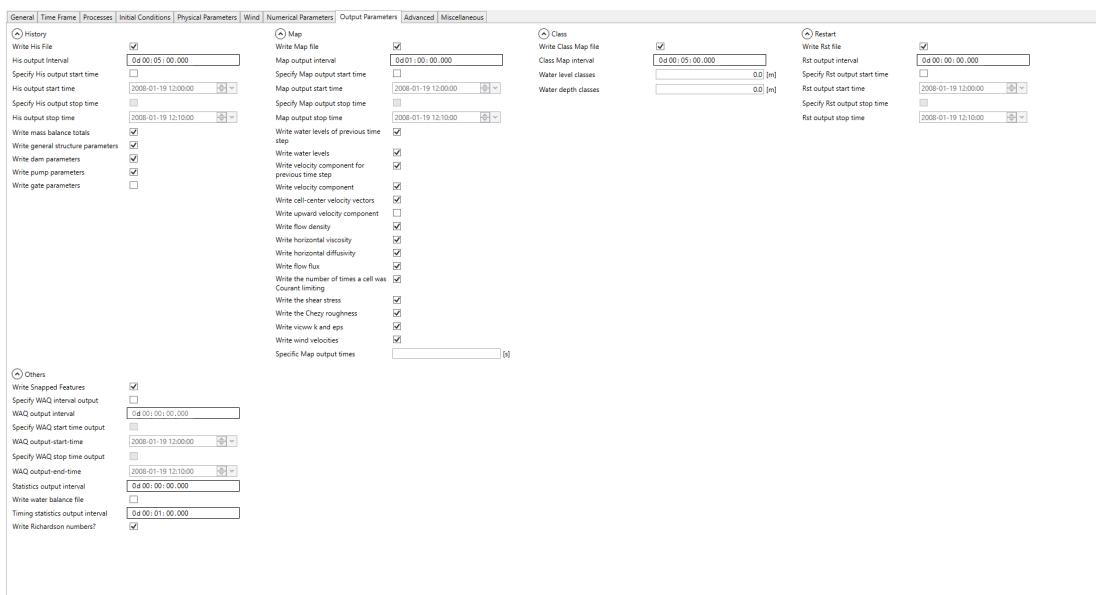


Figure 5.82: Overview Output Parameters tab

For each of the three files (water quality, history, map and restart) the input parameters are specified in the same manner ([Figure 5.82](#)). Taking his output as an example: when *Write His File* is checked, the output of history file is enabled. *His Output Interval* determines the interval at which the output data is stored in the file; the smaller the interval, the more detailed the output and the larger the file. By default, history output is written from the start until the end of the simulation. Optionally, output can be restricted to a certain time window: to specify different output start and/or stop times, check the box next to *Specify His Output Start Time* and/or *Specify His Output Stop Time* and enter the desired times next to the parameters *His Output Start Time* and *His Output Stop Time*.

When *Write Snapped Features* is activated, then shape files with snapped data will be generated for all quantities, such as fixed weirs and thin dams. For example, for fixed weirs a crest height is specified at both end of a fixed weir polyline. In between linear interpolation is applied, which can be checked via these shape files.

Below, the various output options are described in greater detail. *There is a special requirement on the output parameters for His and Restart files that the Output Interval, Output Start Time and Output Stop Time must be integer multiples of User Time Step. Optionally, the interval (Output Stop Time–Output Start Time) should be an integer multiple of User Time Step.*

Write his file

<i>His Output Interval</i>	Time interval for history time series.
<i>His Output Start/Stop time</i>	Restrict history output to a specified time window.
<i>Write mass balance totals</i>	Enable detailed mass balance time series output in the his file.
<i>Write (misc.) structure parameters</i>	Enable time series output across general structures, pumps, weirs and gates in the his file.

Write map file

<i>Map Output Interval</i>	Time interval for map field time series.
<i>Map Output Start/Stop time</i>	Restrict map output to a specified time window.
<i>Specific Map Output Times</i>	File containing specific time values at which to produce additional map output snapshots. If the value is not integer, it is firstly set to the least integer larger than or equal to this value. If the computational time does not hit the specified time value, the output snapshot is chosen to be at the time closest to the specified time value.
<i>Write water levels, etc.</i>	Several optionals for enabling/disabling certain quantities output in the map file.

Write restart file

<i>Restart interval</i>	Time interval for restart files.
<i>Rst Output Start/Stop time</i>	Restrict restart output to a specified time window.

Other output options

<i>WAQ Output Interval</i>	Time interval for D-Water Quality files in <DFM_DELWAQ_mdu_name>*.hyd>, etc.
<i>Simulation statistics output interval</i>	Interval for simulation progress output (on standard out and diagnostics file).
<i>Timing statistics output interval</i>	Interval for detailed timings output into <mdu_name_timings.txt> for expert performance analysis.

Example

One example of input and output parameters (in seconds) is given in [Table 5.2](#). [Table 5.3](#) shows the time (after *Reference Date* in seconds) when output files are generated. We explain this example as follows.

- ◊ The history file has output interval 18 seconds. No parameters are specified for *His Output Start Time* and *His Output Stop Time*, which means that they are automatically set equal to *Start Time* and *Stop Time* of the simulation, respectively.
- ◊ The *Map Output Interval* is 6 seconds, and *Map Output Start Time* is 15 seconds. Since the *Map Output Stop Time* is not given, it is set to equal to *Stop Time*. Moreover, we hope to have output at time given in *Specified Map Output* as 30.5 and 42.1 seconds. These two values are firstly set to 31 and 43 seconds, respectively, in the simulation. Then there will be output snapshots if the computational time hit these two integers. Otherwise, as in this example, the output snapshots will be provided when the computational time hits the time that is greater and the closest to these integers, i.e. at 31.2 and 43.2 (as seen in [Table 5.3](#)).
- ◊ Three parameters are set for the output of the Rst files: the interval, start and stop time.

Input parameters		Output parameters	
<i>Reference Date</i>	2007-11-19 00:00:00	<i>His Output Interval</i>	00:00:18
<i>User Time Step</i>	00:00:00.3	<i>Map Output Interval</i>	00:00:06
<i>Start Time</i>	2007-11-19 00:00:03	<i>Map Output Start Time</i>	2007-11-19 00:00:15
<i>Stop Time</i>	2007-11-19 00:00:51	<i>Specific Map Output</i>	30.5, 42.1
		<i>Rst Output Interval</i>	00:00:09
		<i>Rst Output Start Time</i>	2007-11-19 00:00:12
		<i>Rst Output Stop Time</i>	2007-11-19 00:00:45

Table 5.2: Input and output parameters of the example

His file	3, 21, 39, 51
Map file	3, 15, 21, 27, 31.2, 33, 39, 43.2, 45, 51
Restart file	3, 12, 21, 30, 39, 45

Table 5.3: Time (after Reference Date in seconds) of output files

5.4.13 Miscellaneous

Within the miscellaneous sub-tab, various parameters in relation to waves and equatorial settings can be adjusted. [Table 5.4](#) gives an overview and description of these parameters.

Table 5.4: Overview and description miscellaneous parameters

Parameter	Description
Time step type	Leave at default.
Turbulence model	See chapter 10 .
Turbulence advection	Leave at default.
Water level threshold	Max allowed water level difference between old and new time step in any cell. Run will abort if exceeded. (0 means disabled)
Velocity threshold	Max allowed velocity difference between old and new time step in any cell. Run will abort if exceeded. (0 means disabled)
Dry cell threshold	Flooding threshold at velocity points. Used in wetting and drying.

5.5 Save project, MDU file and attribute files

To save your Delta Shell project, navigate the menu ribbons to *File* and click *Save as*. Choose a location, specify a name and click *Save*. Your project will now be saved in a folder called <name.dsproj_data> and a <name.dsproj> file is written. Within this folder you will find all input ASCII input files of your model, output files of your model (if the model was run using the GUI) and zip folders containing your restart files. Be aware that the output files are stored within a separate folder in which the input files of your model are stored. The output folder on the same level as the folder containing model input files is empty.

To open a project, navigate the menu ribbons to *File* and click *Open*. Select the <*.dsproj>-file of choice and click *Open*.

Importing model or data within a Delta Shell project can be achieved in two ways. Navigate the menu ribbons to *File* and click *Import*. A drop-down menu appears, allowing you to select what you want to import (Figure 5.83).

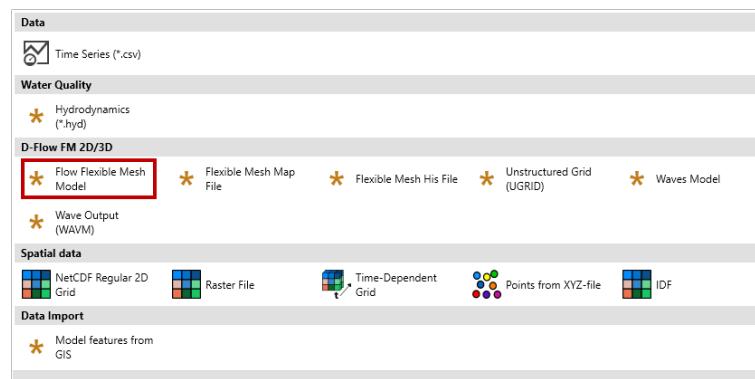


Figure 5.83: Model/data import drop-down menu

Alternatively, you can also right mouse click on the name of your project in the *Project* window and select *Import*. An import wizard appears, allowing you to select what you want to import (Figure 5.84).

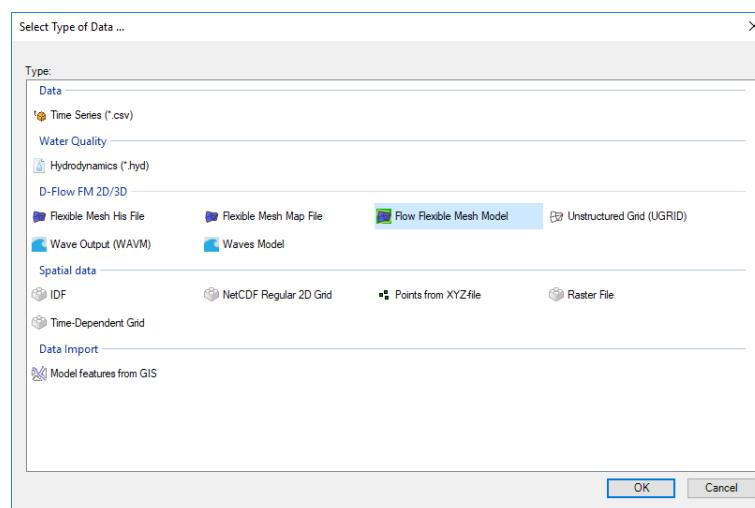


Figure 5.84: Model/data import wizard

Exporting your model can be achieved in the same fashion as importing your model. All model

input files will be written to the folder you select. Be aware that model files exported to a folder in which other model files with the same names are present will be overwritten.

6 Running a model

6.1 Introduction

After defining the input for the D-Flow FM hydrodynamic simulation, the computation can be executed either via Delta Shell (on Windows) or using batch scripts (on Linux and Windows). Via Delta Shell, the status of the computation and possible messages are displayed in a separate messages window. When using a batch script, all messages are written to the diagnostics file ([section F.1](#)) and you can continue working in the current window.

Use a batch script (see [section 6.4](#)) in the following cases:

- 1 Using MPI to run in parallel
- 2 Using some queueing mechanism on a cluster
- 3 Running some unattended simulations, while continuing to work in Delta Shell.

6.1.1 Commandline executables

Note that currently we have two different names of the D-Flow FM executables, for Windows `dflowfm-cli.exe` and for Linux `dflowfm`. There are also some pre- and postprocessing utilities, such as `dfmoutput`. In a standard installation, these executables are located in the following directory:

On Windows:

```
"<Your installation base dir>\dfmsuite\plugins\DeltaShell.Dimr\kernels\x64\dflowfm\bin"
```

On Linux:

```
<Your installation base dir>\bin
```

These executables need dynamic libraries. See [6.4.1](#) and [6.4.2](#) how to make sure these libraries are found and run these executables.

6.2 Parallel calculations using MPI

6.2.1 Introduction

This section describes parallel computing with D-Flow FM based on the *Message Passing Interface* system (MPI). This can be run both on computing clusters with distributed memory as well as shared memory machines with multiple processors and/or multiple CPU cores. The goal of parallelization of D-Flow FM is twofold. We aim for much faster computations on shared- or distributed-memory machines and the ability to model problems that do not fit on a single machine. A less powerful, yet possibly attractive performance improvement is offered by D-Flow FM's OpenMP-based parallelization ([section 6.5.1](#)).

Technical backgrounds on the parallel algorithms in D-Flow FM are described in the Technical Reference Manual [D-Flow FM TRM \(2015\)](#).

Workflow of a parallel run

A parallel run divides the work between multiple processes. To this end we partition the model and let separate processes solve the submodels and generate partitioned output. Given a whole model, the workflow is as follows:

- 1 partition the model (mesh and model definition file),
- 2 submit the parallel job to a queue on a computing cluster, and
- 3 visualize the results from partitioned output files.

6.2.2 Partitioning the model

In D-Flow FM a model is defined by the model definition file, the mesh file and external forcing/boundary condition files, et cetera. The latter are shared by all submodels and do *not* need to be partitioned, they should only be available to all processes. So, partitioning the model concerns partitioning of:

1 the mesh

This is achieved through the graphical user interface or by a command line option. Mesh files for every subdomain will be created;

2 the model definition file

The partitioned model definition files will contain references to the subdomain mesh, and all other information equals its sequential counterpart.

An efficient approach to partition both the mesh and MDU files is via the command line, using

```
> dflowfm --partition:ndomains=n:icgsolver=i <mdu-file>
```

This command reads the name of the mesh file from `mdu-file`, and generates `n` subdomain mesh files by the METIS software package (See [D-Flow FM TRM \(2015\)](#) and references mentioned therein). Then, it creates `n` subdomain MDU files where the parallel Krylov solver `icgsolver` is set to `i`. Here, `i` can be 6, the PETSc solver(recommended for Linux), or, 7, the parallel CG with MILU block preconditioning (recommended for Windows).

For example, to partition the MDU file `<example.mdu>`, which specifies the mesh file as `<example_net.nc>`, to eight subdomain files on a Windows machine, one can use:

```
> dflowfm --partition:ndomains=8:icgsolver=7 example.mdu
```

The mesh file is partitioned into eight mesh files, for each subdomain a mesh file, with names `<example_000j_net.nc>`, $j=0, 1, \dots, 7$. In other words, they are:

```
example_0000_net.nc  example_0003_net.nc  example_0006_net.nc  
example_0001_net.nc  example_0004_net.nc  example_0007_net.nc  
example_0002_net.nc  example_0005_net.nc
```

Then, for each subdomain a mdu-file is created, with names <example_0000.mdu> to <example_0007.mdu>. The different items in, e.g. <example_0000.mdu> with respect to the original mdu-file are:

```
[geometry]
NetFile      = example_0000_net.nc

[numerics]
Icgssolver   = 7
```

If the user wants to manually partition the mesh instead of applying METIS, then he has to provide a polygon file <userpol.pol> which determines the mesh partition. In this situation, following command can partition both the MDU and mesh files:

```
> dfowfm --partition:icgssolver=i <mdu-file> <userpol.pol>
```

6.2.2.1 More about the mesh partitioning

The mesh can be automatically partitioned with the METIS software package, or manually by supplying polygons that define the subdomains. They both produce a cell coloring of the unpartitioned mesh. In each subdomain, the cells are assigned the same color, and augmented with ghost cells. These information are saved in the resulting partitioning mesh file, e.g. <example_NNNN_net.nc>, where _NNNN is a subdomain index.

Regarding partitioning manually with user-supplied partitioning polygons, the partitioning obeys the following rules:

- ◊ if the polygons have a z -value specified, it is considered a subdomain number,
- ◊ if the polygons have no z -value specified, its order determines the corresponding subdomain number,
- ◊ if a cell is not inside at least one polygon, it is assigned to subdomain 0,
- ◊ if a cell is inside only one polygon, it is assigned to the subdomain defined by that polygon,
- ◊ if a cell is inside more than one polygon, it is assigned to the subdomain with the highest number.

In other words, the polygons may be overlapping and the largest subdomain number is taken in the overlapping regions. If the polygons have no z -value, the polygon order determines the corresponding subdomain number, i.e. the first polygon corresponds to subdomain 1 et cetera and there is no polygon defining subdomain 0.

Both types of mesh partitioning are available through Delta Shell or on the command line.

Partitioning the mesh with METIS via Delta Shell

We will firstly focus on the METIS partitioner. Partitioning the mesh from within Delta Shell is achieved by the following steps. In the *Project* window, select the model you want to run by means of clicking on the desired model (Figure 7.1). A right-mouse click will open the context menu, then select *Export....* In the window **Select Type of Data...**, choose *Partition exporter* and the partitioning dialog will appear (Figure 6.1).

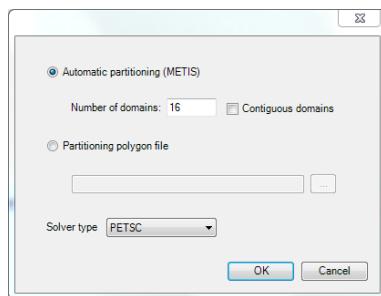


Figure 6.1: Partitioning exporter dialog

Enter the desired amount of subdomains, and typically leave the *contiguous* option switched off and the solver type at its default. After pressing *OK*, a file dialog will appear. Enter the name of the MDU file, *without* any trailing '_000x' partition numbers: these will be added automatically.

Partitioning the mesh with the graphical user interface is achieved by the following steps: You are prompted for Contiguous domains are not necessary for parallel computations in D-Flow FM and often METIS produces contiguous subdomains without enforcing it. Still, some users may want to explicitly enforce contiguous domains. If you want to do so, make sure that your unpartitioned mesh is contiguous. Not being so may cause errors. Note that there is *no* polygon defining subdomain 0. Parts of the mesh not confined by any polygon are assumed to be in subdomain 0. In other words, there are at least $N - 1$ polygons defining N subdomains. In that case, regenerate the cell colors/domain numbers again by selecting *Operations → Generate domain numbers (polygon or METIS)*. Note that you will not be asked to specify the number of subdomains. The cell coloring/domain numbering is now based on the (modified) polygons and not produced by the METIS partitioner. Manual partitioning with user-specified polygons will be explained in the next section. The entered mesh filename is a *basename* that will be used to derive the partitioning filenames, e.g. <example_net.nc> will produce:

Partitioning the mesh from the command line

Apart from using the graphical user interface, it is possible to perform the partitioning on the command line. The two types of partitioning can be carried out via the following commands.

The command that uses METIS partitioner reads:

```
> dflowfm --partition:ndomains=n <meshfile>
```

where `ndomains=n` specifies that `n` subdomains are to be generated. For example, partitioning <example_net.nc> results in subdomain mesh files <example_000j_net.nc>, $j=0, 1, \dots, n-1$.

An advanced command, which enables more options, is:

```
> dflowfm -partition:ndomains=n[:method=0|1][:genpolygon=0|1][:contiguous=0|1]
<meshfile>
```

where the partition method can be chosen via setting `method=0`, the Recursive Bisection method (default), or `method=1`, the multilevel K-Way method. We refer to [Karypis \(2013\)](#) for more details about these two methods. Option `genpolygon` specifies if the command generates a partition polygon file (`genpolygon=1`), or not (`genpolygon=0`, default). Option `contiguous` enforces the contiguous partition when specifying both `contiguous=1` and `method=1`. (Only the K-Way method enables the contiguous partition.) It is not switched on by default. Comparing to the previous command, this advanced command additionally generates a partition polygon file `<example_part.pol>` when `genpolygon=1` is specified.

Note: Backwards compatibility: when using the legacy partitioning script `generate_parallel_mdu.sh` make sure to include the non-default option `genpolygon=1` in the above command, such that the required polygon file is produced. 

Note: When partition a mesh file, e.g. `<example_net.nc>`, by default a separate file `<DFM_interpreted_idomain_example_net.nc>` is generated, which includes partition domain information and cell information. One can switch off generating this file by adding in the MDU file `->[output]->Writepart_domain = 0`. 

To manually partition a mesh, a user-specified polygon file `<userpol.pol>` has to be provided. The corresponding command reads:

```
> dfflowfm --partition <meshfile> <userpol.pol>
```

This generates files the same as before.

6.2.3 Partitioning the MDU file

Having partitioned the mesh, the model definition file needs to be partitioned, as every sub-model requires its own definition file with references to

- ◊ the partitioned mesh file, e.g. `<example_0000_net.nc>`,
- ◊ an appropriate parallel Krylov solver, can be
 - 6: PETSc solver, recommended (for Linux), or
 - 7: parallel CG with MILU block preconditioning (for Windows),
- ◊ a unique snapshot directory,
- ◊ optionally, a partitioned restart file, and
- ◊ optionally, a partitioning polygon file, e.g. `<example_part.pol>`.

The `generate_parallel_mdu.sh` script partitions a sequential model definition file automatically:

```
> generate_parallel_mdu.sh <mdu-file> <nprocs> [partpol-file] <ICGSolver>
```

with

<code>mdu-file</code>	sequential model definition file,
<code>nprocs</code>	number of subdomains/parallel processes,

partpol-file partitioning polygon file (optionally),
Icgssolver parallel Krylov solver, can be 6 or 7.

Note that the partitioned mesh filenames (and partitioned restart filenames) are derived from the mesh filename (and the restart filename) specified in the sequential model definition file.

6.2.3.1 Remaining model input

A parallel run of a D-Flow FM model needs only partitioned <.mdu> and <_net.nc> files. All other model input is the same as for a standalone run, e.g., meteo forcings, boundary conditions, observation stations and more. These are generally copied to the working directory by the parallel job submission script.

6.2.4 Running a parallel job

To run a parallel job with D-Flow FM model on a cluster you have to prepare the submission script. The submission script should be prepared with respect to the specific options that job scheduler on your cluster requires.

The simple example of the D-Flow FM submission script on the cluster with the Grid Engine:

```
#!/bin/bash
#$ -V
#$ -q test
#$ -cwd
#$ -N My_DFlowFM_JOB
#$ -m bea
#$ -M my.email@provider.net

export LD_LIBRARY_PATH=$DFLOWFM/lib:$LD_LIBRARY_PATH
export PATH=$DFLOWFM/bin:$PATH

mpexec -np 4 dflowfm --autostartstop YOUR_MDU_FILE.mdu >out.txt 2>err.txt
```

In this simplified example above we submit the D-Flow FM simulation that was partitioned into 4 domains and is going to use only 1 node. The options used above are:

- V Specify that all environment variables active within the qsub utility be exported to the context of the job.
- q Specify the queue 'test' to be used for this job, if absent default queue is used.
- cwd Execute the job from the current working directory.
- N Specify the name of the job.
- m Specifies which message type should be emailed (b=beginning of job, e=end of job, a=abort of job).
- M Specifies the email address to send the notification.

In order to submit more complicated, e.g. multi-node simulations, additional options that are scheduler depended have to be added.

6.2.5 Visualizing the results of a parallel run

The map and history output files (as introduced in section 5.4.12) deserve special attention in parallel runs.

The history file — with time series for observation points, structures and more — is written only by process #0, and all model-global data has already been aggregated into that single file: <mdu_name_0000_his.nc>.

The map file — with full-grid output of flow fields — is written for each domain separately as <mdu_name_000X_map.nc>. This saves communication overhead during the parallel run. The partitioned map files contain duplicate points, since each file also contains the domain's ghost nodes. For postprocessing these map files, two options are now available:

- 1 Direct plotting of the set of all map files in Delft3D-QUICKPLOT: the partitioned file series will be recognized automatically, and the partition results will be drawn on top of each other. For water levels this gives good results.
- 2 Merging the partitioned map files into a single global map file with the dfmoutput tool. The resulting map file can then be loaded again in Delft3D-QUICKPLOT and other post-processing utilities.

6.2.5.1 Plotting all partitioned map files with Delft3D-QUICKPLOT

When opening one of the partitioned map files into Delft3D-QUICKPLOT, it will automatically detect that the map file is part of a series. An additional select list *Domain* appears, see Figure 6.2. Select either “all partitions”, or a partition of your choice, and proceed with the plotting as normal (section 7.3).

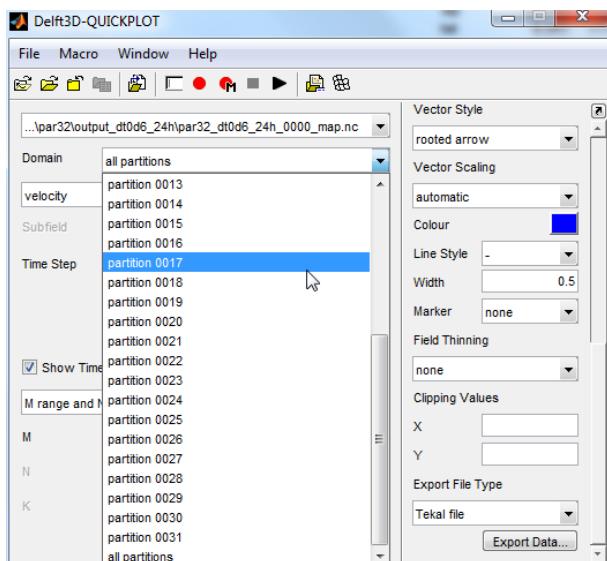


Figure 6.2: Domain selector in Delft3D-QUICKPLOT for partitioned map files.

6.2.5.2 Merging multiple map files into one

The partitioned map files of a parallel model run can be merged into a single global map file with the dfmoutput tool. It cuts off ghost nodes, and concatenates all grid points, taking care of correct global renumbering. Usage:

```
> dfmoutput mapmerge [--infile FILE1 [FILE2 FILE3...]] [--outfile DSTFILE]
```

where FILE1/2/3 are the input files, e.g. <mdu_name_0000_map.nc>, ... ,

<mdu_name_0031_map.nc> and DSTFILE is an optional output file name (the default is <mdu_name_merged_map.nc>).



Remark:

- ◊ Since a restart file is a special type of map file, partitioned restart files can also be merged using the above command.

The built-in help gives a list of more advanced options:

```
> dfmoutput mapmerge --help
Merge multiple map files from parallel run into one.

Optional switches:
--infile FILE1 [FILE2...], -i FILE1 [FILE2...]
    default value
    One or more input files.
--listfile LISTFILE, -F LISTFILE
    Pass contents of LISTFILE as input files.
--outfile DSTFILE, -o DSTFILE
    Write output to file DSTFILE.
Default: <model>_merged_map.nc
--force, -f
    default value .false.
    Force overwriting of existing output file.
--help, -h
    Print this help message
--version, -v
    Print version

Examples:
dfmoutput mapmerge --infile model_0000_map.nc model_0001_map.nc
```

See sections 6.4.1 and 6.4.1 if you have difficulties running dfmoutput .

6.3 Running a scenario using Delta Shell

In the *Project* window, select the model you want to run by means of clicking the first attribute of the desired model (Figure 7.1).

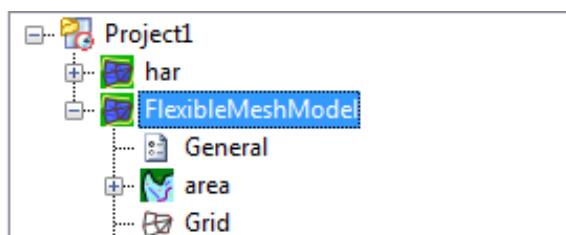


Figure 6.3: Selecting the model you want to run in the Project window

Starting the calculation can be achieved in two ways. You can navigate the menu ribbons; go to “Home” and in the group Run click on the button “Run Current”. To run all models that are opened in the *Project* window, click on “Run All” (Figure 6.4).

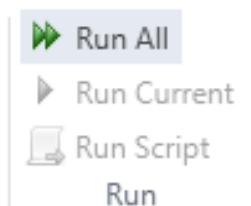


Figure 6.4: Group Run in Home ribbon

Alternatively, you can right mouse click the first attribute in the *Project* window of the model you want to run. Next, click “Run Model” to start the calculation. When you select the first of the model you want to run, the properties window (“View” “Properties”) shows several properties of the model you selected. If you set “ShowModelRunConsole” on true, the model run console of the computational core will be showed during the calculation, providing you with additional information during the model run (Figure 6.5).

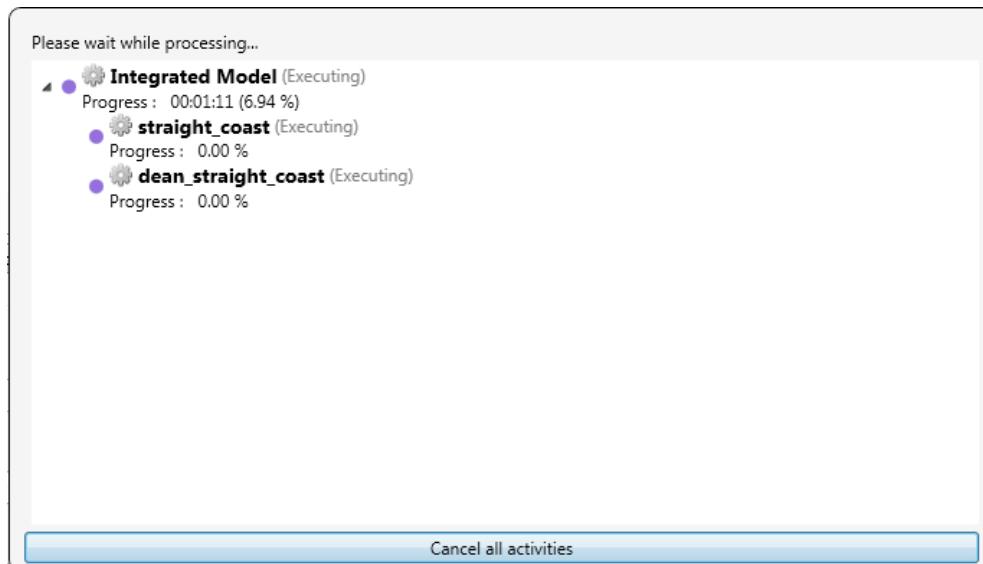


Figure 6.5: Run console Delta Shell

When you cancel the run by clicking “Cancel all activities”, model results are stored up to the point where you cancel the run.

6.4 Running a scenario using a batch script

Separate scripts are needed for Windows (with the extension <*.bat>) and Linux (with the extension <*.sh>). See section 6.7 for the command-line arguments.

In this section we refer to the variable %DFLOWFM% for Windows and \$DFLOWFM for Linux as a variable that stores path to the directory that contains the dflowfm-cli package.

The easiest Windows script (assuming you have downloaded the Windows dflowfm-cli package, assuming you are in the directory of a configured test-case):

```
set PATH=%DFLOWFM%\bin;%PATH%
```

```
dflowfm-cli.exe --autostartstop YOUR_MDU_FILE.mdu
```

The easiest Linux script (assuming you have downloaded the Linux dflowfm-cli package, assuming you are in the directory of a configured test-case):

```
export LD_LIBRARY_PATH=$DFLOWFM/lib:$LD_LIBRARY_PATH
export PATH=$DFLOWFM/bin:$PATH
dflowfm --autostartstop YOUR_MDU_FILE.mdu
```

6.4.1 Running the partitioner or dfmoutput on Windows

In this section the variable %DIMR% refers to the x64 directory of the DIMR installation.

An example script for running the partitioner is:

```
set PATH=%DIMR%\dflowfm\bin;%DIMR%\share\bin;%PATH%
dflowfm-cli.exe --partition %YOUR_PARTITION_OPTIONS%
```

An example script for running dfmoutput is:

```
set PATH=%DIMR%\dflowfm\bin;%DIMR%\share\bin;%PATH%
dfmoutput.exe %YOUR_DFMOUTPUT_OPTIONS%
```

This example script works for mapmerge, max25 and genfilter.

6.4.2 Running the partitioner or dfmoutput on Linux

In this section \$DIMR refers to the lnx64 directory of the DIMR installation.

An example script for running the partitioner is:

```
export LD_LIBRARY_PATH=$DIMR/lib:$LD_LIBRARY_PATH
export PATH=$DIMR/bin:$PATH
dflowfm --partition %YOUR_PARTITION_OPTIONS%
```

An example script for running dfmoutput is:

```
export LD_LIBRARY_PATH=$DIMR/lib:$LD_LIBRARY_PATH
export PATH=$DIMR/bin:$PATH
dfmoutput %YOUR_DFMOUTPUT_OPTIONS%
```

This example script works for mapmerge, max25 and genfilter.

6.5 Run time

The actual run time of a model can vary considerably depending on a variety of factors such as:

- ◊ The problem being solved, characterised by the number of active grid points, the number of layers in the vertical or the number of processes taken into account.
- ◊ The length of the simulation in time and the time step being used.
- ◊ The hardware configuration that is used and the work load of the processor.

For this reason, only some general considerations are given to determine the run time of a hydrodynamic simulation. On a PC or a workstation without separate I/O-processors the CPU time is the sum of the processor time and the I/O time.

The *processor time* required for a simulation is primarily determined by:

- ◊ The model definition, i.e., the number of active grid points and the number and type of the processes taken into account.
- ◊ The length of the simulated period in terms of the number of time steps executed.

The *I/O time* is determined by:

- ◊ The number of times the computed data are written to history, map, restart files and other communication files for water quality or wave model couplings.
- ◊ The number of observation points, cross-sections and the number of output parameters.

The simulation performance is defined as the CPU time per grid point per time step per constituent:

$$\text{simulation performance} = \frac{\text{CPU time}}{\text{Dnt} \cdot \text{Ndx}} \quad [\text{system seconds}]$$

where:

- | | |
|-----|--------------------------------------|
| Dnt | is the number of time steps executed |
| Ndx | is the number of flow nodes |

The simulation performance is written to the diagnostic file at the end of the simulation.

6.5.1 Multi-core performance improvements by OpenMP

D-Flow FM has built-in support for multi-core parallelism using OpenMP¹. This speeds up calculations by employing multiple processor cores in a single (shared-memory) computer, e.g., a modern-day notebook. OpenMP-parallelism in D-Flow FM does not scale as well as MPI-parallelism (section 6.2), but it comes for free (not any change to model input necessary) and can give a welcome performance improvement (approximately double speed on an Intel quadcore CPU). It is strongly advised to limit the number of OpenMP-threads to one less than the number of physical cores in your machine, thus also ignoring any hyperthreading. An example on Linux for an i7 quadcore CPU machine:

¹<http://www.openmp.org>

```
export OMP_NUM_THREADS=3
dflowfm --autostartstop YOUR_MDU_FILE.mdu
```

6.6 Files and file sizes

For estimating the required disk space the following files are important:

- ◊ history file
- ◊ map file
- ◊ restart file

6.6.1 History file

The size of the history file is determined by:

- ◊ The number of monitoring points (observation points + cross-sections): H1.
- ◊ The number of quantities stored: H2.
- ◊ The number of additional process parameters, such as salinity, temperature, constituents and turbulence quantities, taken into account in the simulation: H3.
- ◊ The number of time the history data is written to the history file: H4.

You can estimate the size of a history file (in bytes) from the following equation:

$$\text{size history file} = H1 \cdot (H2 + H3) \cdot H4 \cdot 8 \text{ bytes.}$$

As a first approximation you can use H2 = 8.

Example

For a 2D simulation with density driven currents (salinity and temperature), a simulated period of 12 hrs 30 min, a time integration step of 5 minutes, 30 monitoring points and each time step being stored, the size of the history file will be of the order of 384 kBytes. For the same model but now with 10 layers in the vertical the file size will increase to about 4 MBytes. These estimates show that history files are rather small. Unless the number of monitoring points is excessively large the history files are typically much smaller than the map output files.

6.6.2 Map file

The size of the map file is determined by:

- ◊ The size of the model, i.e. the number of grid cells multiplied by the number of layers ($N_{dx} \cdot K_{max}$): M1n, and the number of flow links (open grid cell edges) multiplied by the number of layers ($L_{nx} \cdot K_{max}$): M1l.
- ◊ The number of quantities stored on grid cells and flow links: M2n, M2l, respectively.
- ◊ The number of process parameters taken into account, such as salinity, temperature, constituents and turbulence quantities: M3.
- ◊ The number of time steps for which the map file is written: M4.



Remark:

- ◊ For a more refined estimate you should distinguish between parameters that depend or not on the number of layers used (such as the water level). For a 3D simulation the

latter quantities can be neglected, for a 2D simulation they must be accounted for. As a first estimate we double the number of quantities M2 in a 2D simulation.

As a first approximation you can use $M2n = 5$, $M2l = 5$ for a 3D simulation and $M2n = 8$, $M2l = 5$ for a 2D simulation.

You can estimate the size of a map file (in bytes) from the following equation:

$$\text{size map file} = [M1n \cdot (M2n + M3) + M1l \cdot M2l] \cdot M4 \cdot 8 \text{ bytes.}$$

Example

For a 2D simulation with 6800 grid cells and 13000 flow links, simulation results stored for a period of 7 days, and the file is written with an interval of 60 minutes the size of the map file will be about 161 MBytes. For larger models the map file can easily become excessively large, as result the map file is less frequently written, for instance every 2 or 3 hours.

6.6.3 Restart file

A restart file is a special type of map file, where only one time snapshot per file is saved (i.e, $M4 = 1$). No grid or flow geometry information is stored in a restart file, except for the flow cell/link information (denoted by M5). Moreover, the restart files obtained after a parallel run contain some necessary information about parallelization (denoted by M6). Similarly to the equation of size map file above, one can write the estimating equation as follows:

$$\text{size rst file} = [M1n \cdot (M2n + M3) + M1l \cdot M2l + M5 + M6] \cdot 8 \text{ bytes,}$$

where $M6 = 0$ for a sequential run.

6.7 Command-line arguments

A complete model schematisation can be run from the command line using the D-Flow FM Command Line Interface (CLI), `dflowfm-cli.exe` (`dflowfm` on Linux). A basic non-interactive run is started by:

```
> dfollowfm-cli --autostartstop MDUFILE
```

In the box below, a full list of command-line options and arguments is shown:

```
> dfollowfm-cli --help
Usage: dfollowfm-cli [OPTIONS] [FILE]...
Options:
  --autostart MDUFILE
    Auto-start the model run, and wait upon completion.

  --autostartstop MDUFILE
    Auto-start the model run, and exit upon completion.

  --noautostart MDUFILE
    Disable any AutoStart option in the MDU file (if any).

  --partition:OPTS [POLFILE] NETFILE
```

```

Partitions the unstructured grid in NETFILE into multiple files.

POLFILE is an optional polygon file which defines the partitions.
Only used when ndomain in OPTS is undefined or 0.

OPTS is a colon-separated list opt1=val1:opt2=val2:...
    ndomains = N      Number of partitions.
    method    = [01]   Partition method: Recursive Bisection(0), K-Way(1).

    genpolygon= [01]  Generate partition polygon(1) or not(0).
    contiguous= [01]  Enforce contiguous grid cells in each domain.
                      Only available when K-Way is enabled (method=1).

-t N, --threads N
    Set maximum number of OpenMP threads.
N must be a positive integer.

--refine:OPTS NETFILE
    Refine the unstructured grid in NETFILE from commandline.
    OPTS is a colon-separated list opt1=val1:opt2=val2:...
        hmin=VAL
        dtmax=VAL
        maxlevel=M
        connect=[01]
        directional=[01]
        outsidecell=[01]

-q, --quiet
    Minimal output: Only (fatal) errors are shown.

--verbose:[level_stdout[:level_dia]], e.g., --verbose:INFO:DEBUG
    Set verbosity level of output on standard out and in diagnostics file.
    where level is in: {DEBUG|INFO|WARNING|ERROR|FATAL}
    Levels are optional, default is INFO on screen, DEBUG in dia file.

-h, --help
    Display this help information and exit.

-v, --version
    Output version information and exit.

```

6.8 Restart a simulation

D-Flow FM allows to restart a simulation, not from the original starting time, but from a user-specified time with all the information at that time. In other words, for a model which has a long spinup time from t_{start} , instead of restarting from t_{start} , one can restart another simulation of the same model from a later time t_{rst} , where $t_{start} \leq t_{rst} < t_{stop}$. And the results for times $t_{rst} \leq t \leq t_{stop}$ are the same as in the original simulation run.

This can be achieved by following steps:

- 1 In order to obtain restart files $<_rst.nc>$, this needs to be enabled in the output parameters (see section 5.4.12 for more details about how to set output parameters).
- 2 Run the original model from t_{start} to t_{stop} , resulting in restart files.
- 3 To restart the simulation from time t_{rst} , in the MDU file modify TStart to t_{rst} , and fill in the name of corresponding restart file in RestartFile of the [restart]-section. Moreover, place the corresponding restart file relative to the directory of this MDU file.

To restart a parallel simulation, one can use any of the following methods:

Method 1 On each subdomain use its own restart file.

Method 2 Merge all the subdomain restart files into a single global restart file (see section 6.2.5.2), and then use it for all the subdomains. This means to put the name of this merged file as `RestartFile` in all subdomain MDU files. This method is supported when restarting with the same, or a different domain partitioning. Moreover, such a merged restart file can also be used to restart a sequential run.

The above two methods require taking care of the name of the restart file in subdomain MDU files. D-Flow FM automatically fills in the correct names, when partition an MDU file to subdomain MDU files (see section 6.2.2), in the following way:

For Method 1 When `<mduname_YYMMDD_HHMMSS_rst.nc>` is filled in `RestartFile` of the original MDU file, partition this MDU file gives subdomain MDU files with `RestartFile` as `<mduname_000X_YYMMDD_HHMMSS_rst.nc>` for each subdomain `<000X>`.

For Method 2 If the string word "merged" is contained in the name of `RestartFile` in the original MDU file, then this file name is kept to all the subdomain MDU files after partitioning.

It is also possible to use a map file as a restart file, following the above approaches. Notice that in this case, one needs to specify the `RestartDateTime` in MDU file as well. However, we strongly recommend to use `<_rst.nc>` file instead of a map file.

D-Flow FM also supports restarting a 3D model simulation with σ -layers. **Note:** The 3D-implementation is a beta functionality.



6.9 Frequently asked questions

This chapter aims to help you with common questions that may arise while using D-Flow FM.

1 Question

My model does not run/crashes. What's wrong?

Answer

The diagnostics file is the starting point for finding out what went wrong. See section F.1 for a detailed description of the contents of this file, and the order of model run output. Globally, ask yourself the following questions:

- ◊ Was the MDU file found?
- ◊ If yes, was the model successfully loaded? Common mistakes are missing boundary or meteorological forcings file.
- ◊ If yes, was the time loop successfully started? Possible errors are non-writable output files.
- ◊ If yes, does the dia file contain any messages from during the time loop? Possible errors are solver convergence errors.
- ◊ If not, did the run end successfully?

2 Question

I get a warning that my network is non-orthogonal. Can I loosen the orthogonality threshold?

Answer

Unfortunately, no. Orthogonality is very important for accuracy: advised orthogonality values for your grid are around 0.01, preferably lower. The current threshold is already very high at a value of 0.5. Use RGFGRID to improve your grid orthogonality (and smoothness).

7 Visualize results

7.1 Introduction

A model run will produce two types of output:

- 1 a graph or history (<*.his>-file) for a specific quantity on a location
- 2 a map (<*.map>-file) for a specific quantity

Both are stored in files within the project, as described in [section 6.6](#).

D-Flow FM provides basic visualization of the model and the model results. Advanced and tailor-made visualization is possible by the export of the his- and/or map-files, and inspection with dedicated visualization applications. Some are provided and described below.

7.2 Visualization with Delta Shell

When your model run has finished, a new folder called “Output” has appeared at the bottom of the attributes of your model in the *Map* window. Inside this folder, all output quantities of the his-, and map-files can be found, as well as a folder called “States”, in which you find all restart files written during the model run. Output folders have also appeared in the *Map* window; “Output (map)” and “Output (his)”. Both the results of your map and his files can be viewed in the central map by means of enabling the desired quantities from the *Map* window ([Figure 7.1](#)).

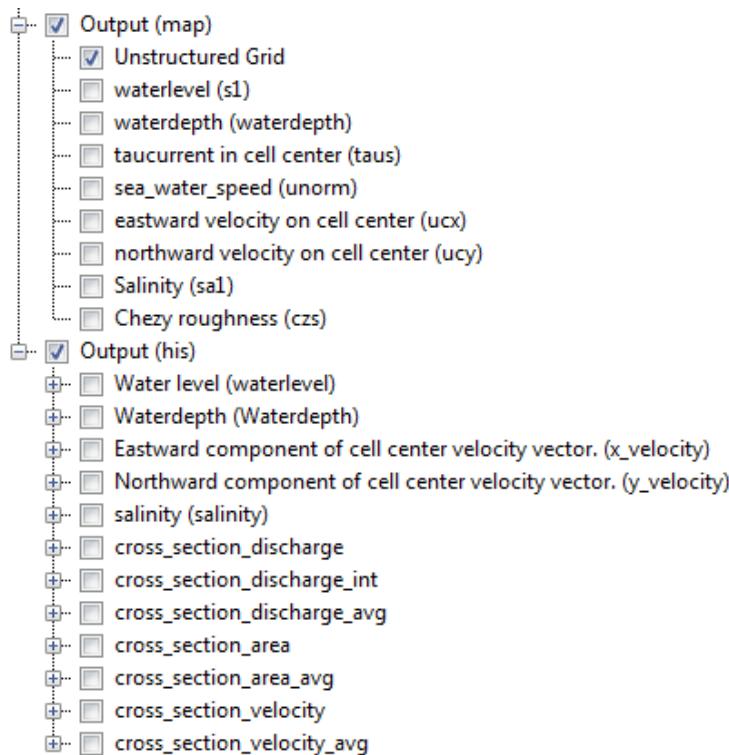


Figure 7.1: Example of setting output (in)visible in the Map window

The time navigator can be used to slide through the different time steps in the output files. If your model is relatively large, the drawing performance of the interface can be improved

dramatically by enabling QuadTree visualizations in the properties window of the layer you want to visualize. To this end, select the layer you want to visualize in the ...

7.3 Visualization with Delft3D-QUICKPLOT

The interface of the Delft3D-QUICKPLOT allows to open the NetCDF output files from a D-Flow FM simulation. In the interface you can select data fields, make a selection of time steps, stations or specify a preferred figure presentation options. After selection of a certain options you can visualize your data by using the “Quick View” button. To add another plot to an existing figure the “Add to Plot” button should be used.

When plotting the map output files from a D-Flow FM simulation, by default the presentation type “markers” will be selected. To smoothly visualize your results it is recommended to change presentation type into “polygons” and then select the option “Fill Polygons” (see the example on the [Figure 7.2](#)).

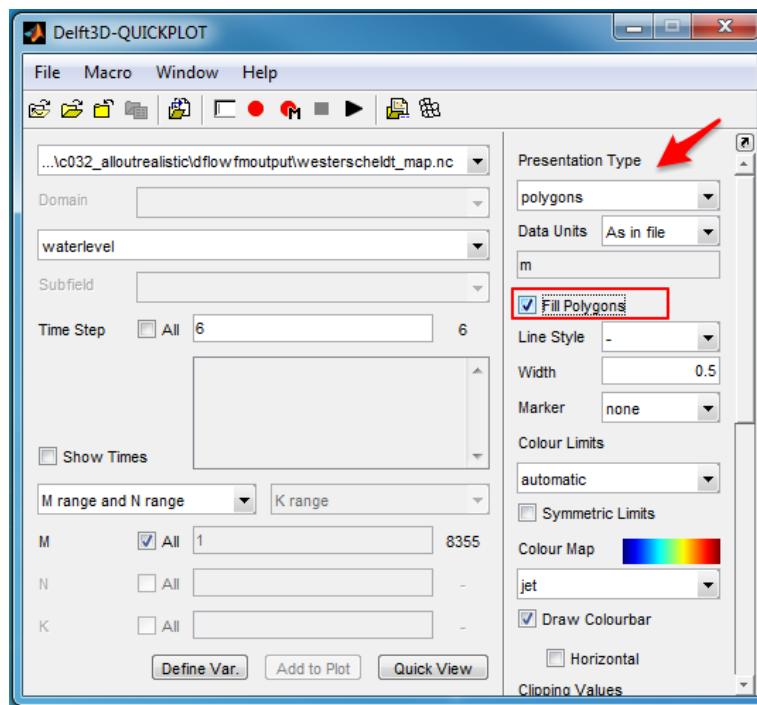


Figure 7.2: Useful map visualization options in the Delft3D-QUICKPLOT

See the Delft3D-QUICKPLOT User Manual for full details and a description of the routines and their use.

7.4 Visualization with Muppet

The interface of the Muppet allows to “Add Dataset” from a NetCDF output file from a D-Flow FM simulation. It is possible to create a timeseries for a selected variable in from the history files. Additionally Muppet offers the option to visualize the map output files from a D-Flow FM simulation (see Figure 7.3).

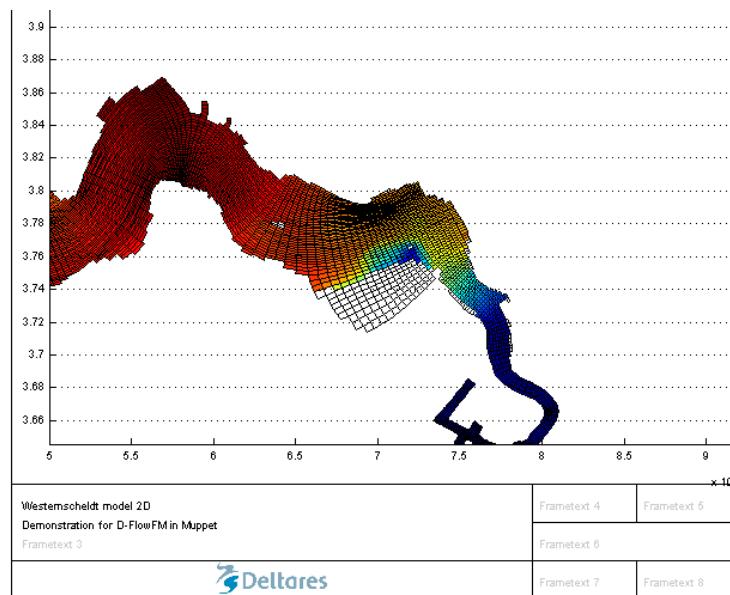


Figure 7.3: Example of the Muppet visualization of the D-Flow FM map output file

In the Muppet interface it is possible to specify figure settings, plot descriptions, labels and many more.

7.5 Visualization with Matlab

In the OpenEarthTools you can find two example scripts that allow you to load and visualize the D-Flow FM output. Note that in the Matlab scripts for the D-Flow FM the UGRID notation is used.

The <plotMap.m> script plots a D-Flow FM unstructured map and optionally the handles h are returned. By modifying the script you can change the plotted variable, layout or the plot style options.

The <plotNet.m> script plots a D-Flow FM unstructured net, optionally the handles h are returned. This script can visualize nodes, links and cell circumcenters of a network. By modifying the script you can choose the preferred plotting settings and variables to plot (by default nodes, edges and faces are plotted).

7.6 Visualization with Python

Currently there are no examples of Python scripts for the D-Flow FM output visualization in the OpenEarthTools. However, if such a need arises it is possible to load netcdf Python libraries, and create your own simple visualization of the variables present in the NetCDF output files.

8 Hydrodynamics

8.1 Introduction

Increasing awareness of environmental issues has focused the attention of scientists and engineers on the problem of predicting the flow and dispersion of contaminants in water systems. Reliable information on water flow, waves, water quality, sediment transport and morphology can be obtained from appropriate mathematical models. In general the first step in such modelling activities concerns the simulation of the flow itself. Whether the problem is related, for example, to the stability of a hydraulic structure, to salt intrusion, to the dispersion of pollutants or to the transport of silt and sediment, flow simulations usually form the basis of the investigations to be carried out.

The *Delft3D Flexible Mesh Suite* is the integrated modelling system of Deltares for the aquatic environment. D-Flow Flexible Mesh, the flow module of this system, provides the hydrodynamic basis for other modules such as water quality, ecology, waves and morphology. For steady and non-steady modelling of the far-field water quality and ecology, it is coupled with the far-field water quality module D-Water Quality. For the interaction between waves and currents the flow module may be coupled with the short-waves model D-Waves. To control structures, the flow module is coupled to the D-Real Time Control module.

D-Flow FM is flexible by using an unstructured grid in the horizontal plane. In the vertical direction D-Flow FM offers two different vertical grid systems: a so-called σ co-ordinate system (σ -model) introduced by [Phillips \(1957\)](#) for ocean models and the Cartesian z -co-ordinate system (Z -model).

This section gives some background information on the conceptual model of the D-Flow FM module. Most of the concepts and algorithms are applicable to both the σ -model and Z -model.

Note: The 3D-implementation is a β -functionality.



8.2 General background

8.2.1 Range of applications of D-Flow FM

The hydrodynamic module D-Flow FM simulates two-dimensional (2DH, depth-averaged) or three-dimensional (3D) unsteady flow and transport phenomena resulting from tidal and/or meteorological forcing, including the effect of density differences due to a non-uniform temperature and salinity distribution (density-driven flow). The flow model can be used to predict the flow in shallow seas, coastal areas, estuaries, lagoons, rivers and lakes. It aims to model flow phenomena of which the horizontal length and time scales are significantly larger than the vertical scales.

If the fluid is vertically homogeneous, a depth-averaged approach is appropriate. D-Flow FM is able to run in two-dimensional mode (one computational layer), which corresponds to solving the depth-averaged equations. Examples in which the two-dimensional, depth-averaged flow equations can be applied are tidal waves, storm surges, tsunamis, harbor oscillations (seiches) and transport of pollutants in vertically well-mixed flow regimes.

Three-dimensional modelling is of particular interest in transport problems where the horizontal flow field shows significant variation in the vertical direction. This variation may be generated by wind forcing, bed stress, Coriolis force, bed topography or density differences. Examples are dispersion of waste or cooling water in lakes and coastal areas, upwelling and

downwelling of nutrients, salt intrusion in estuaries, fresh water river discharges in bays and thermal stratification in lakes and seas.

8.2.2 Physical processes

The numerical hydrodynamic modelling system D-Flow FM solves the unsteady shallow water equations in two (depth-averaged) or in three dimensions. The system of equations consists of the horizontal equations of motion, the continuity equation, and the transport equations for conservative constituents. The equations are formulated in orthogonal curvilinear co-ordinates or in spherical co-ordinates on the globe. In D-Flow FM models with structured grid are considered as a simplified form of an unstructured grid. In Cartesian co-ordinates, the free surface level and bathymetry are related to a flat horizontal plane of reference, whereas in spherical co-ordinates the reference plane follows the Earth curvature.

The flow is forced by tide at the open boundaries, wind stress at the free surface, pressure gradients due to free surface gradients (barotropic) or density gradients (baroclinic). Source and sink terms are included in the equations to model the discharge and withdrawal of water.

The D-Flow FM model includes mathematical formulations that take into account the following physical phenomena:

- ◊ Free surface gradients (barotropic effects).
- ◊ The effect of the Earth rotation (Coriolis force).
- ◊ Water with variable density (equation of state).
- ◊ Horizontal density gradients in the pressure (baroclinic effects).
- ◊ Turbulence induced mass and momentum fluxes (turbulence closure models).
- ◊ Transport of salt, heat and other conservative constituents.
- ◊ Tidal forcing at the open boundaries.
- ◊ Space and time varying wind shear-stress at the water surface.
- ◊ Space varying shear-stress at the bed.
- ◊ Space and time varying atmospheric pressure on the water surface.
- ◊ Time varying sources and sinks (e.g. river discharges).
- ◊ Drying and flooding of tidal flats.
- ◊ Heat exchange through the free surface.
- ◊ Evaporation and precipitation.
- ◊ Tide generating forces.
- ◊ Effect of secondary flow on depth-averaged momentum equations.
- ◊ Lateral shear-stress at wall.
- ◊ Vertical exchange of momentum due to internal waves.
- ◊ Influence of waves on the bed shear-stress (2D and 3D).
- ◊ Wave induced stresses (radiation stress) and mass fluxes.
- ◊ Flow through hydraulic structures.
- ◊ Wind driven flows including tropical cyclone winds.

8.2.3 Assumptions underlying D-Flow FM

In D-Flow FM the 2D (depth-averaged) or 3D non-linear shallow water equations are solved. These equations are derived from the three dimensional Navier-Stokes equations for incompressible free surface flow. The following assumptions and approximations are applied:

- ◊ In the σ co-ordinate system the depth is assumed to be much smaller than the horizontal length scale. For such a small aspect ratio the shallow water assumption is valid, which means that the vertical momentum equation is reduced to the hydrostatic pressure relation. Thus, vertical accelerations are assumed to be small compared to the gravitational acceleration and are therefore not taken into account.
- ◊ The effect of variable density is only taken into account in the pressure term (Boussinesq approximation).
- ◊ In the σ co-ordinate system, the immediate effect of buoyancy on the vertical flow is not considered. In D-Flow FM vertical density differences are taken into account in the horizontal pressure gradients and in the vertical turbulent exchange coefficients. So the application of D-Flow FM is restricted to mid-field and far-field dispersion simulations of discharged water.
- ◊ For a dynamic online coupling between morphological changes and flow the 2D sediment and morphology feature is available.
- ◊ In a Cartesian frame of reference, the effect of the Earth curvature is not taken into account. Furthermore, the Coriolis parameter is assumed to be uniform unless specifically specified otherwise.
- ◊ In spherical co-ordinates the inertial frequency depends on the latitude.
- ◊ At the bed a slip boundary condition is assumed, a quadratic bed stress formulation is applied.
- ◊ The formulation for the enhanced bed shear-stress due to the combination of waves and currents is based on a 2D flow field, generated from the velocity near the bed using a logarithmic approximation.
- ◊ The equations of D-Flow FM are capable of resolving the turbulent scales (large eddy simulation), but usually the hydrodynamic grids are too coarse to resolve the fluctuations. Therefore, the basic equations are Reynolds-averaged introducing so-called Reynolds stresses. These stresses are related to the Reynolds-averaged flow quantities by a turbulence closure model.
- ◊ In D-Flow FM the 3D turbulent eddies are bounded by the water depth. Their contribution to the vertical exchange of horizontal momentum and mass is modelled through a vertical eddy viscosity and eddy diffusivity coefficient (eddy viscosity concept). The coefficients are assumed to be proportional to a velocity scale and a length scale. The coefficients may be specified (constant) or computed by means of an algebraic, $k\cdot\tau$ or $k\cdot\varepsilon$ turbulence model, where k is the turbulent kinetic energy, τ is the turbulent time scale and ε is the dissipation rate of turbulent kinetic energy.
- ◊ In agreement with the aspect ratio for shallow water flow, the production of turbulence is based on the vertical (and not the horizontal) gradients of the horizontal flow. In case of small-scale flow (partial slip along closed boundaries), the horizontal gradients are included in the production term.
- ◊ The boundary conditions for the turbulent kinetic energy and energy dissipation at the free surface and bed assume a logarithmic law of the wall (local equilibrium).
- ◊ The eddy viscosity is an-isotropic. The horizontal eddy viscosity and diffusivity coefficients should combine both the effect of the 3D turbulent eddies and the horizontal motions that cannot be resolved by the horizontal grid. The horizontal eddy viscosity is generally much larger than the vertical eddy viscosity.
- ◊ For large-scale flow simulations, the tangential shear-stress at lateral closed boundaries can be neglected (free slip). In case of small-scale flow partial slip is applied along closed boundaries.
- ◊ For large-scale flow simulations, the horizontal viscosity terms are reduced to a bi-harmonic

operator along co-ordinate lines. In case of small-scale flow the complete Reynold's stress tensor is computed. The shear-stress at the side walls is calculated using a logarithmic law of the wall.

- ◊ In the σ co-ordinate system, D-Flow FM solves the so-called long wave equation. The pressure is hydrostatic and the model is not capable of resolving the scales of short waves. Therefore, the basic equations are averaged in analogy with turbulence introducing so called radiation stresses. These stresses are related to the wave quantities of Delft3D-WAVE.
- ◊ It is assumed that a velocity point is set dry when the actual water depth is below half of a user-defined threshold. If the point is set dry, then the velocity at that point is set to zero. The velocity point is set wet again when the local water depth is above the threshold. The hysteresis between drying and flooding is introduced to prevent drying and flooding in two consecutive time steps.
The drying and flooding procedure leads to a discontinuous movement of the closed boundaries at tidal flats.
- ◊ A continuity cell is set dry when all surrounding velocity points at the grid cell faces are dry or when the actual water depth at the cell centre is below zero (negative volume).
- ◊ The flux of matter through a closed wall and through the bed is zero.
- ◊ Without specification of a temperature model, the heat exchange through the free surface is zero. The heat loss through the bed is always zero.
- ◊ If the total heat flux through the water surface is computed using a temperature excess model the exchange coefficient is a function of temperature and wind speed and is determined according to [Sweers \(1976\)](#). The natural background temperature is assumed constant in space and may vary in time. In the more advanced heat flux formulation the fluxes due to solar radiation, atmospheric and back radiation, convection, and heat loss due to evaporation are modeled separately.
- ◊ The effect of precipitation on the water temperature is accounted for.

8.3 Hydrodynamic processes

D-Flow FM solves the Navier Stokes equations for an incompressible fluid, under the shallow water and the Boussinesq assumptions. In the vertical momentum equation the vertical accelerations are neglected, which leads to the hydrostatic pressure equation. In 3D models the vertical velocities are computed from the continuity equation. The set of partial differential equations in combination with an appropriate set of initial and boundary conditions is solved on an unstructured finite volume grid.

In the horizontal direction D-Flow FM uses orthogonal unstructured grids. Two coordinate references are supported:

- 1 Cartesian co-ordinates
- 2 Spherical co-ordinates

The boundaries of a river, an estuary or a coastal sea are in general curved and are not smoothly represented on a structured grid. The boundary becomes irregular and may introduce significant discretization errors. To reduce these errors unstructured grids are used. The unstructured grids also allow local grid refinement in areas with large horizontal gradients.

In the vertical direction D-Flow FM offers two different vertical grid systems: the σ coordinate system (σ -model) and the Cartesian z -co-ordinate system (Z-model). In the σ model, the vertical grid consists of layers bounded by two σ planes, which are not strictly horizontal but follow the bed topography and the free surface. Because the σ -model is boundary fitted both to the bed and to the moving free surface, a smooth representation of the topography is obtained. The number of layers over the entire horizontal computational area is constant,

irrespective of the local water depth. The distribution of the relative layer thickness is usually non-uniform. This allows for more resolution in the zones of interest such as the near surface area (important for e.g. wind-driven flows, heat exchange with the atmosphere) and the near bed area (sediment transport). Please note that in D-Flow FM, unlike Delft3D-FLOW, the σ coordinate is equal to zero on the bed and 1 on the water surface.

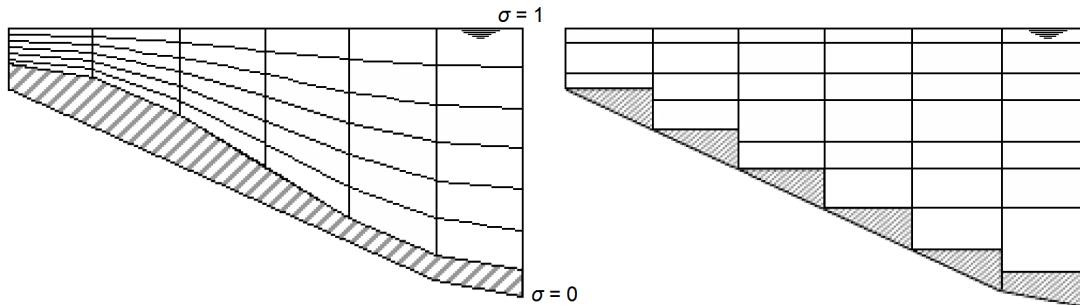


Figure 8.1: Example of σ -model (left) and Z-model (right).

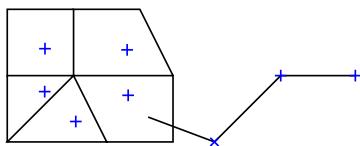
Although the σ -grid is boundary fitted (in the vertical), it will not always have enough resolution around the pycnocline. The co-ordinate lines intersect the density interfaces that may give significant errors in the approximation of strictly horizontal density gradients (Leendertse, 1990; Stelling and Van Kester, 1994). Therefore, Z-model was introduced in D-Flow FM for 3D simulations of weakly forced stratified water systems. The Z-model has horizontal co-ordinate lines that are (nearly) parallel with density interfaces (isopycnals) in regions with steep bed slopes. This is important to reduce artificial mixing of scalar properties such as salinity and temperature. The Z-model is not boundary-fitted in the vertical. The bed (and free surface) is usually not a co-ordinate line and is represented as a staircase (zig-zag boundary).

8.3.1 Topological conventions

A computational cell in a D-Flow FM grid (sometimes referred to as a 'network') consists of corner nodes and edges connecting the corner nodes. Such a grid cell should contain at least three corner nodes and at most six corner nodes. The following topological conventions are used:

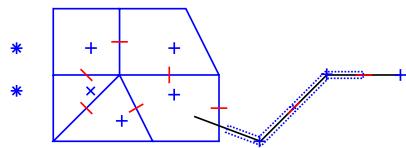
- ◊ netnodes: corners of a cell (triangles, quadrangles, ...),
- ◊ netlinks: edges of a cell, connecting netnodes,
- ◊ flownodes: the cell circumcentre, in case of triangles the exact intersection of the three perpendicular bisectors and hence also the centre of the circumscribing circle,
- ◊ flowlinks: a line segment connecting two flownodes.

1. Net (domain discretization)



net node ++ net link (1D) — net link (2D)	(1..NUMK) (1..NUML1D) (NUML1D+1..NUML)
--	--

2. Flow data (1D+2D)



* boundary flow node + pressure points: 2D flow node circumcenter/1D flow node — flow link	□ netcell/flow node (2D) (1..NDX2D=NUMP) ↗ netcell/flow node (1D) (NDX2D+1..NDX) * boundary flow node (NDXI+1..NDX) + pressure points: 2D flow node circumcenter/1D flow node — flow link 1D internal (1..LNX1D) 2D internal (LNX1D..LNXI) 1D open bnd (LNXI+1..LNX1DBND) 2D open bnd (LNX1DBND+1..LNX)
---	--

Figure 8.2: Flexible mesh topology

This mesh topology is illustrated in Figure 8.2. The 'center' of a cell can be defined in multiple ways. To illustrate this, two conventional cell center definitions for a triangle are highlighted in Figure 8.3. The two displayed cell centers have different properties:

- 1 the circumcenter is the location within the triangle which is the center point of a circle that intersects the triangle at each corner node of the triangle; as a consequence, the orthogonal projection of the center point to each face of the triangle divides each face into two exactly equidistant pieces,
- 2 the mass center (or centroid) is the center of gravity; as a consequence, a line through a corner node and the mass center divides a face into two exactly equidistant pieces under an angle not necessarily equal to 90°.

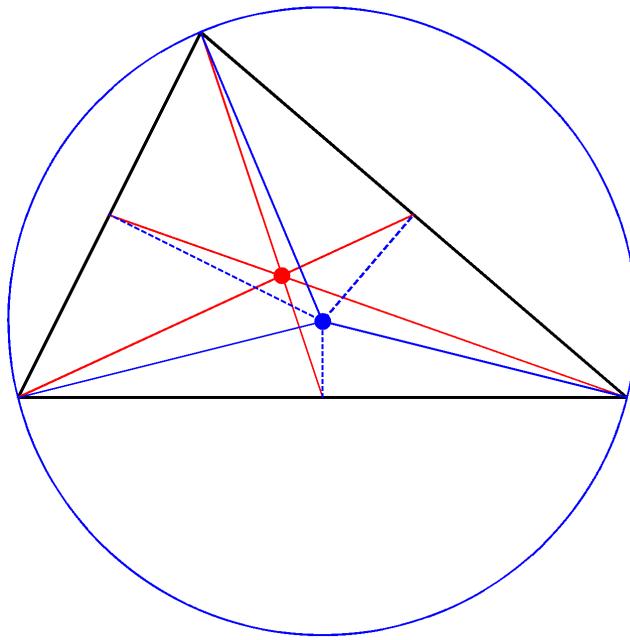


Figure 8.3: Two conventional definitions of the cell center of a triangle: the *circumcenter* and the *mass center*.

D-Flow FM utilizes the **circumcenter** as the basis of the definition of the elementary flow variables 'water level' and 'flow velocity'. The water level is defined at the circumcenter, whereas the face normal flow velocity is defined at the orthogonal projection of the circumcenter onto the cell face, i.e., the midpoint of the cell face.

Important properties of the mesh are the *orthogonality* and *smoothness*. The *orthogonality* is defined as the cosine of the angle φ between a flowlink and a netlink. Ideally 0, angle $\varphi = 90^\circ$. The *smoothness* of a mesh is defined as the ratio of the areas of two adjacent cells. Ideally 1, the areas of the cells are equal to each other. A nearly ideal setup is shown in Figure 8.4.

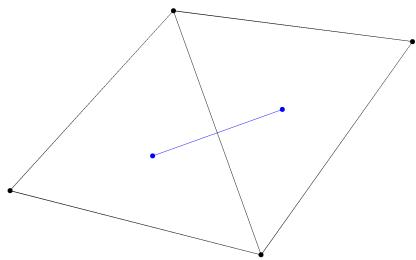


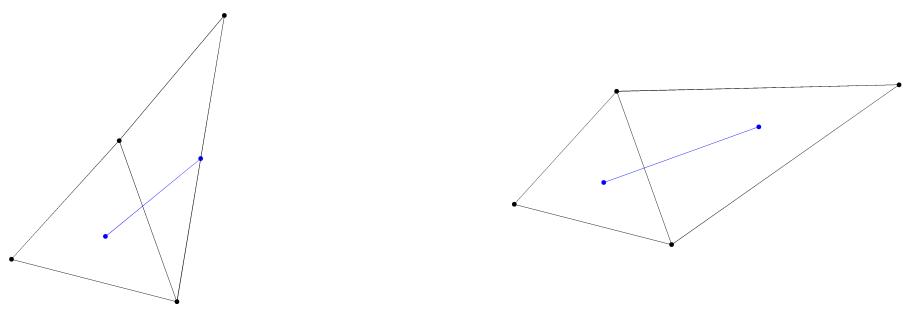
Figure 8.4: Perfect orthogonality and nearly perfect smoothness along the edge connecting two triangles. Black lines/dots are network links/nodes, blue lines/dots are flow links/nodes.

It is quite easy (and therefore dangerous) to generate meshes that violate the orthogonality

and smoothness requirements. In [Figure 8.5](#), two different setups of two gridcells are shown with different mesh properties.

The left picture of [Figure 8.5](#) shows how orthogonality can be deteriorated by skewing the right triangle with respect to the left triangle. While having the same area (perfect smoothness), the mutually oblique orientation results in poor orthogonality. In this particular case, the centre of the circumscribing circle is in principle located outside the right triangle. Such a triangle is denoted as an 'open' triangle, which is bad for computations.

The opposite is shown in the right picture of [Figure 8.5](#) in which the right triangle has strongly been elongated, disturbing the smoothness property. However, the orthogonality is nearly perfect. Nonetheless, both meshes need to be improved to assure accurate model results.



(a) Perfect smoothness, but poor orthogonality.

(b) Perfect orthogonality, but poor smoothness

Figure 8.5: Poor mesh properties due to violating either the smoothness or the orthogonality at the edge connecting two triangles. Black lines/dots are network links/nodes, blue lines/dots are flow links/nodes.

8.3.2 Conservation of mass and momentum

In this section, we will present in detail the governing equations for mass and momentum conservation. These equations are indicated by a continuity equation and momentum equations.

8.3.2.1 Continuity equation

D-Flow FM solves the depth-averaged continuity equation, derived by integrating the continuity equation, for incompressible fluids ($\nabla \cdot \mathbf{u} = 0$) over the total depth, taken into account the kinematic boundary conditions at water surface and bed level, and is given by:

$$\frac{\partial h}{\partial t} + \frac{\partial Uh}{\partial x} + \frac{\partial Vh}{\partial y} = Q \quad (8.1)$$

with U and V the depth averaged velocities. Q is representing the contributions per unit area due to the discharge or withdrawal of water, precipitation and evaporation:

$$Q = \int_0^h (q_{in} - q_{out}) dz + P - E \quad (8.2)$$

with q_{in} and q_{out} the local sources and sinks of water per unit of volume [1/s], respectively, P the non-local source term of precipitation and E non-local sink term due to evaporation. We remark that the intake of, for example, a power plant is a withdrawal of water and should be modelled as a sink. At the free surface there may be a source due to precipitation or a sink due to evaporation.

8.3.2.2 Momentum equations in horizontal direction

The momentum equations in x - and y -direction are given by:

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} + w \frac{\partial u}{\partial z} - fv = -\frac{1}{\rho_0} \frac{\partial P}{\partial x} + F_x + \frac{\partial}{\partial z} \left(\nu_V \frac{\partial u}{\partial z} \right) + M_x \quad (8.3)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + w \frac{\partial v}{\partial z} + fu = -\frac{1}{\rho_0} \frac{\partial P}{\partial y} + F_y + \frac{\partial}{\partial z} \left(\nu_V \frac{\partial v}{\partial z} \right) + M_y \quad (8.4)$$

Where ν_V is the vertical eddy viscosity coefficient. Density variations are neglected, except in the baroclinic pressure terms, $\partial P/\partial x$ and $\partial P/\partial y$ represent the pressure gradients.

The forces F_x and F_y in the momentum equations represent the unbalance of horizontal Reynolds stresses.

M_x and M_y represent the contributions due to external sources or sinks of momentum (external forces by hydraulic structures, discharge or withdrawal of water, wave stresses, etc.).

The effects of surface waves on the flow as modelled in D-Flow FM are described in section 16.2.

8.3.2.3 Vertical velocities

The vertical velocity w in the adapting σ co-ordinate system is computed from the continuity equation:

$$\frac{\partial h}{\partial t} + \frac{\partial uh}{\partial x} + \frac{\partial vh}{\partial y} + \frac{\partial wh}{\partial z} = h (q_{in} - q_{out}) \quad (8.5)$$

At the surface the effect of precipitation and evaporation is taken into account. The vertical velocity w is defined at the iso σ -surfaces. w is the vertical velocity relative to the moving σ -plane. It may be interpreted as the velocity associated with up- or downwelling motions.

8.3.3 The hydrostatic pressure assumption

Under the shallow water assumption, the vertical momentum equation is reduced to a hydrostatic pressure equation. Vertical accelerations due to buoyancy effects and due to sudden variations in the bed topography are not taken into account. So:

$$\frac{\partial P}{\partial z} = -\rho gh \quad (8.6)$$

For water of constant density and taking into account the atmospheric pressure, it includes gradients of the free surface level, called barotropic pressure gradients. The atmospheric pressure is included in the system for storm surge simulations. The atmospheric pressure gradients dominate the external forcing at peak winds during storm events. Space and time varying wind and pressure fields are especially important when simulating storm surges.

In case of a non-uniform density the pressure gradients includes not only barotropic pressure gradient, but also vertical pressure gradient, the so called baroclinic pressure gradient. The baroclinic pressure gradient is the result of variable distribution of density and temperature in the vertical direction.

In the horizontal gradient a vertical derivative is introduced by the σ co-ordinate transformation. In estuaries and coastal seas the vertical grid may deteriorate strongly in case of steep

bed slopes. In order to avoid artificial flow the numerical approximation of the baroclinic pressure terms requires a special numerical approach. The treatment of D-Flow FM to avoid the artificial mixing due to σ co-ordinates are discussed in section 8.5, see also Stelling and Van Kester (1994).

8.3.4 The Coriolis force

The Coriolis parameter f depends on the geographic latitude ϕ and the angular speed of rotation of the earth, Ω ($f = 2\Omega \sin \phi$). For a grid the user should specify the space varying Coriolis parameter, using a suitable projection. This can be done by selecting *Coordinate System* in RGFGRID, and selection of the option for *Spherical Coordinate*. The parameters for translation and rotation can be given as shown in Figure 8.6.

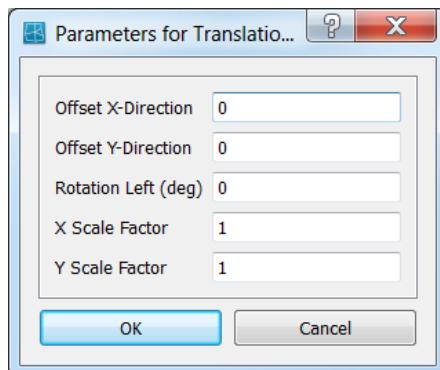


Figure 8.6: Input for map projection for specifying Coriolis parameter on the grid.

8.3.5 Diffusion of momentum

The forces F_x and F_y in the horizontal momentum equations represent the unbalance of horizontal Reynolds stresses. The Reynolds stresses are modelled using the eddy viscosity concept, (for details e.g. Rodi (1984)). This concept expresses the Reynolds stress component as the product between a flow as well as grid-dependent eddy viscosity coefficient and the corresponding components of the mean rate-of-deformation tensor. The meaning and the order of the eddy viscosity coefficients differ for 2D and 3D, for different horizontal and vertical turbulence length scales and fine or coarse grids. In general the eddy viscosity is a function of space and time.

For 3D shallow water flow the stress tensor is an-isotropic. The horizontal eddy viscosity coefficient, ν_H , is much larger than the vertical eddy viscosity ν_V ($\nu_H \gg \nu_V$). The horizontal viscosity coefficient may be a superposition of three parts:

- 1 a part due to "sub-grid scale turbulence",
- 2 a part due to "3D-turbulence" see Uittenbogaard *et al.* (1992) and
- 3 a part due to dispersion for depth-averaged simulations.

In simulations with the depth-averaged momentum and transport equations, the redistribution of momentum and matter due to the vertical variation of the horizontal velocity is denoted as dispersion. In 2D simulations this dispersion is not simulated as the vertical profile of the horizontal velocity is not resolved. Then this dispersive effect may be modelled as the product of a viscosity coefficient and a velocity gradient. The dispersion term may be estimated by the Elder formulation.

If the vertical profile of the horizontal velocity is not close to a logarithmic profile (e.g. due to stratification or due to forcing by wind) then a 3D-model for the transport of matter is recommended.

The horizontal eddy viscosity is mostly associated with the contribution of horizontal turbulent motions and forcing that are not resolved by the horizontal grid ("sub-grid scale turbulence") or by (a priori) the Reynolds-averaged shallow-water equations. in 3D, in the vertical direction, ν_V is referred to as the three-dimensional turbulence and in it is computed following a 3D-turbulence closure model.

Therefore, in addition to all turbulence closure models in D-Flow FM a constant (space and time) background mixing coefficient may be specified by the user, which is a background value for the vertical eddy viscosity in the momentum [Equation \(8.3\)](#) and [Equation \(8.4\)](#) consequently.

The horizontal and vertical eddy viscosities can be set by user defined value under Physical Parameters shown in [Figure 8.7](#).

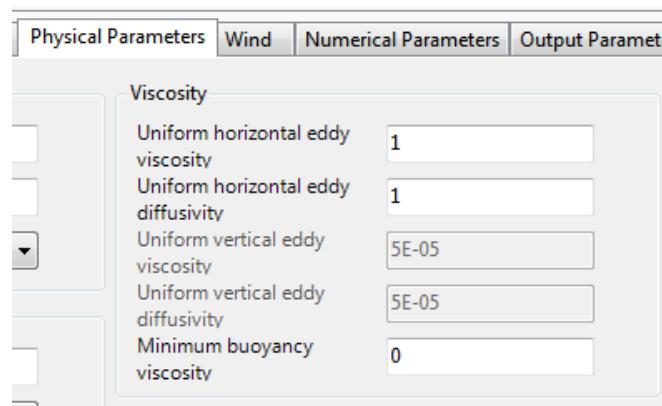


Figure 8.7: Input parameters for horizontal and vertical eddy viscosities.

8.4 Hydrodynamics boundary conditions

In [section 5.4.8](#), the boundary conditions are discussed from the viewpoint of the user interface. In the user interface, the user can specify the locations at which particular boundary conditions are to be imposed. Using [section 5.4.8](#) as a backdrop, the present section discusses the underlying files and fileformats and the way these are interpreted by the computational kernel. Three types of boundary conditions are discussed in this section, namely open boundaries (in [section 8.4.1](#)), vertical boundaries (in [section 8.4.2](#)) and closed boundaries (in [section 8.4.3](#)).

8.4.1 Open boundary conditions

The proper prescription of an open boundary condition along a certain (part of the) rim of the grid can be achieved by considering four elements of the model:

- 1 a polyline file (extension `.pli`), containing the *locations* at which the boundary conditions should be imposed,
- 2 a boundary conditions file (extension `.bc`), containing the key *physical information*, such the time dependent information of the quantity under consideration and the physical nature of the quantity itself,
- 3 an external forcing file (extension `.ext`), in which the connection is laid between the polyline file and the boundary conditons file,
- 4 the prescription of the single external forcing file, containing the connection between location and physical information, in the master definition file (extension `.mdu`).

These four elements are subsequently discussed in the following.

8.4.1.1 The location of support points

The flow engine needs the specification of support points for the boundary conditions (also see [section 5.4.8.1](#)). By default, these support points are a means to construct a series of virtual cell centers along the boundary rim of the grid. [Figure 8.8](#) provides an image of this concept.

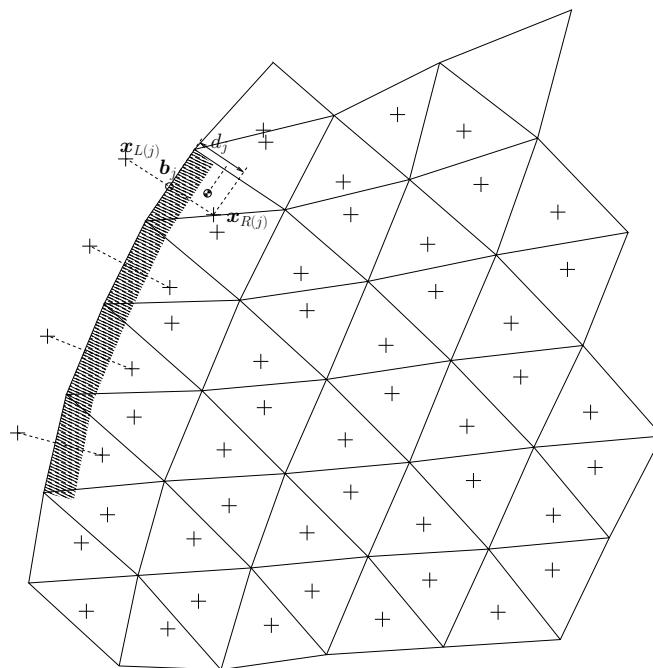


Figure 8.8: Virtual boundary ‘cells’ near the shaded boundary; $x_{L(j)}$ is the virtual ‘cell’ center near boundary face j ; $x_{R(j)}$ is the inner-cell center; b_j is the point on face j that is nearest to the inner-cell center

The support points are stored as one single polyline per boundary condition, marking the rim along which the boundary conditions should hold. The polyline should be drawn in the vicinity of the rim. The user can specify the size of this ‘vicinity’ by means of the MDU-file keyword `OpenBoundaryTolerance`. The keyword specifies the search tolerance factor between the boundary polyline and the grid cells. The unit of this keyword is the cell size unit (i.e., not

metres). By default, this value is 3, which loosely means that in the vicinity of $3\Delta x$ of the grid rim is searched of a boundary condition polyline.

The actual location of a specific boundary location point can be computed in three different ways, dependent on the user's choice for the keyword `izbndpos` in the MDU-file:

- 1 `izbndpos = 0`: construction of the boundary condition point by means of orthogonal mirroring of the closest cell center,
- 2 `izbndpos = 1`: construction of the boundary condition point as the orthogonal projection of the closest cell center onto the grid rim,
- 3 `izbndpos = 2`: construction of the boundary condition point as the actual location of the support points spanning the polyline.

Only the option `izbndpos = 0` is discussed; the other two options are not yet fully operational. The mirroring of the closest cell center is conducted as follows. First, the orthogonal distance from the boundary cell center to the actual rim of the grid is computed: d_j in Figure 8.8. Second, the cell center of the outside virtual cell is defined at a distance d_j outside of the grid. However, the flat area of the cell, say A , at the rim can give rise to an adaptation of this distance. If $\frac{1}{2}\sqrt{A} > d_j$, then the center of the virtual cell is located at a distance $\frac{1}{2}\sqrt{A}$ away from the grid.

8.4.1.2 Physical information

In the `bc`-file, multiple types of boundary conditions can be prescribed. In this section, the boundary conditions for the flow motion are briefly reflected on.

Water level

A water level signal is applied at the cell center of the virtual cell outside the grid (ghost cells or mirror cells). Water levels can be imposed as a timeseries or as a harmonic signal. In case of a harmonic signal, the period of the signal can be given as an astronomic component acronym.

If a timeseries is applied to the first support point of a polyline named `arbitraryname`, prescribed in some polyline file, then the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = timeseries
Time-interpolation = linear
Quantity      = time
Unit          = minutes since YYYY-MM-DD
Quantity      = waterlevelbnd
Unit          = m
[two-column data]
```

The data is to be inserted as a two-column array containing the time (in minutes) and the water level itself (in meters w.r.t. the reference level). In case of harmonic components, the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
```

```

Function          = harmonic
Quantity         = harmonic component
Unit             = minutes
Quantity         = waterlevelbnd amplitude
Unit             = m
Quantity         = waterlevelbnd phase
Unit             = degrees
[three-column data]

```

The data is to be inserted as a three-column array containing the period (in minutes), the amplitude (in meters) and the phase (in degrees). If the period is T minutes, the amplitude is A meters and the phase is φ degrees, then the signal that is applied reads

$$h(t) = B + A \cos(2\pi t/T - \varphi). \quad (8.7)$$

The parameter B can be prescribed by an additional signal with a period specified equal to 0 minutes. As an example, the data series:

```

0.0  0.5  0.0
745.0 2.0  0.0

```

represents the signal $h(t) = 0.5 + 2.0 \cos(2\pi t/745)$, with the time t in minutes.

Discharge

A discharge boundary condition is applied at the face-center of the virtual boundary cell. For this, the face-normal velocity is used in combination with the face-center water depth. The face-based water depth in the evaluation of the flow area can optionally be set to a downwind approximation (for an inflowing discharge boundary) with the option `jbasqbnndownwindhs = 1` (default value).

Discharges can be imposed as a timeseries or as a harmonic signal. In case of a harmonic signal, the period of the signal can be given as an astronomic component acronym.

If a timeseries is applied to the first support point of a polyline named `arbitraryname`, prescribed in some polyline file, then the header of the `bc`-file is:

```

[forcing]
Name          = arbitraryname_0001
Function      = timeseries
Time-interpolation = linear
Quantity      = time
Unit          = minutes since YYYY-MM-DD
Quantity      = dischargebnd
Unit          = m3/s
[two-column data]

```

The data is to be inserted as a two-column array containing the time (in minutes) and the water level itself (in meters w.r.t. the reference level). In case of harmonic components, the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = harmonic
Quantity      = harmonic component
Unit          = minutes
Quantity      = dischargebnd amplitude
Unit          = m3/s
Quantity      = dischargebnd phase
Unit          = degrees
[three-column data]
```

The data is to be inserted as a three-column array containing the period (in minutes), the amplitude (in cubic meters per second) and the phase (in degrees).

Velocity

A velocity boundary condition is applied at the face-center of the virtual boundary cell. Values provided are interpreted as face-normal velocities. Velocities can be imposed as a timeseries or as a harmonic signal. In case of a harmonic signal, the period of the signal can be given as an astronomic component acronym.

If a timeseries is applied to a the first support point of a polyline named `arbitraryname`, prescribed in some polyline file, then the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = timeseries
Time-interpolation = linear
Quantity      = time
Unit          = minutes since YYYY-MM-DD
Quantity      = velocitybnd
Unit          = m/s
[two-column data]
```

The data is to be inserted as a two-column array containing the time (in minutes) and the water level itself (in meters w.r.t. the reference level). In case of harmonic components, the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = harmonic
Quantity      = harmonic component
Unit          = minutes
Quantity      = velocitybnd amplitude
Unit          = m/s
Quantity      = velocitybnd phase
Unit          = degrees
[three-column data]
```

The data is to be inserted as a three-column array containing the period (in minutes), the amplitude (in meters per second) and the phase (in degrees).

Water level gradient

Besides the option to prescribe actual water levels as a boundary condition, D-Flow FM facilitates the prescription of water level *gradients*. Such a Neumann-type boundary condition for the water level can be assigned through the keyword `neumannbnd`. The value of the water level gradient is applied at the face-center.

Water level gradients can be imposed as a timeseries or as a harmonic signal. In case of a harmonic signal, the period of the signal can be given as an astronomic component acronym. If a timeseries is applied to the first support point of a polyline named `arbitraryname`, prescribed in some polyline file, then the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = timeseries
Time-interpolation = linear
Quantity       = time
Unit           = minutes since YYYY-MM-DD
Quantity       = neumannbnd
Unit           = -
[two-column data]
```

The data is to be inserted as a two-column array containing the time (in minutes) and the water level itself (in meters w.r.t. the reference level). In case of harmonic components, the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = harmonic
Quantity       = harmonic component
Unit           = minutes
Quantity       = neumannbnd amplitude
Unit           = -
Quantity       = neumannbnd phase
Unit           = degrees
[three-column data]
```

The data is to be inserted as a three-column array containing the period (in minutes), the amplitude (in meters per second) and the phase (in degrees). The water level gradient is interpreted positive in the outward-normal direction.

Riemann invariant

At a Riemann boundary we do not allow any outgoing perturbation with respect to some reference boundary state to reflect back from the boundary. This is achieved by prescribing the incoming Riemann invariant. Using the convention of a positive inward normal at the boundary, this can be put as $u_b + 2\sqrt{gH_b}$, with u_b the velocity at the boundary and H_b the total water depth at the boundary. In this expression, we take the boundary values u_b and H_b as the reference boundary state. While applying Riemann boundaries, directional effects are disregarded.

Using linearization and the assumption of a flow field that is initially at rest, the Riemann boundary is rewritten such that it takes the form:

$$\zeta = 2\zeta_b - \sqrt{\frac{H}{g}}u - \zeta_0 \quad (8.8)$$

with ζ the surface level elevation, H the total water depth, g the gravitational acceleration, u the velocity (positive inward) and ζ_0 the initial surface level elevation. Instead of prescribing a combination of the velocity and the water level, we prefer to prescribe only the water level at the boundary, i.e. ζ_b . The value for ζ_b is supposed to be provided by the user (as well as, obviously, the initial surface level elevation ζ_0).

A water level for Riemann boundary can be imposed as a timeseries or as a harmonic signal. In case of a harmonic signal, the period of the signal can be given as an astronomic component acronym. If a timeseries is applied to a the first support point of a polyline named `arbitraryname`, prescribed in some polyline file, then the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = timeseries
Time-interpolation = linear
Quantity      = time
Unit          = minutes since YYYY-MM-DD
Quantity      = riemannbnd
Unit          = m
[two-column data]
```

The data is to be inserted as a two-column array containing the time (in minutes) and the water level for Riemann boundary (in meters). In case of harmonic components, the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = harmonic
Quantity      = harmonic component
Unit          = minutes
Quantity      = riemannbnd amplitude
Unit          = m
Quantity      = riemannbnd phase
Unit          = degrees
[three-column data]
```

The data is to be inserted as a three-column array containing the period (in minutes), the amplitude (in meters per second) and the phase (in degrees).

Suppose, as an example, that we have:

- ◊ a flow domain with a bathymetry at a bed level equal to 0 m w.r.t. the reference level,
- ◊ an initial water level equal to 10 m w.r.t. the reference level,
- ◊ a local initial disturbance of the initial water level,
- ◊ and an initial flow field at rest,

and that we aim to prevent reflections at the boundary. In that case, it follows from [Equation \(8.8\)](#) that $\zeta_b = 10$ m w.r.t. the reference level is to be prescribed. Remark that this application only holds for small disturbances of ζ from ζ_b .

Discharge-water level dependency

If a relation between the discharge and the local water level is known on beforehand, then this relation can be provided to the flow model as a table by means of the keyword `qhtable`. This table, provides the water level ζ as a function of the computed discharge Q .

If a Qh-table is applied to a the first support point of a polyline named `arbitraryname`, prescribed in some polyline file, then the header of the `bc`-file is:

```
[forcing]
Name          = arbitraryname_0001
Function      = qhtable
Quantity      = qhbnd discharge
Unit          = m3/s
Quantity      = qhbnd waterlevel
Unit          = m
[two-column data]
```

The data is to be inserted as a two-column array containing the discharge (in cubic meters per second) and the water *level* (in meters w.r.t. the reference level).

The user is able to apply a weighting parameter to compromise between the water level computed by the Qh-relation and the water level computed at the previous time step level. This parameter is accessible as the keyword `Qhrelax` in the MDU-file. By default, this parameter is 0.01. As a consequence, the newly computed water level consists of the Qh-relation result for 1 % and for 99 % of the previous time step water level.

8.4.1.3 Example

Recalling the four elementary actions from the beginning of [section 8.4.1](#), the administration in files can be illustrated by means of an example. Suppose, we have a channel that is geometrically modelled by means of the grid shown in [Figure 8.9](#). The domain is 10000 m long and 500 m wide. The bottom left corner coincides with the origin of the geometrical frame of reference.

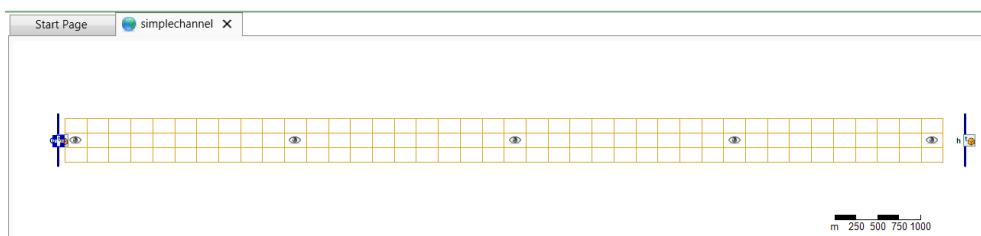


Figure 8.9: Delta-Shell view of a simple channel covered by a straightforward Cartesian grid. Boundary conditions are prescribed at the left hand side and the right hand side of the domain.

The domain shown in [Figure 8.9](#) contains two boundaries: on the left hand, a discharge

boundary condition is present to prevent flow entering the domain, on the right hand, a water level boundary condition is set. Having two boundaries, we have two polyline files, in this example: `left.pli` and `right.pli`.

The contents of `left.pli` are:

```
boundaryleft
 2      2
-80    -50
-80    550
```

whereas the contents of `right.pli` are:

```
boundaryright
 2      2
10250   -50
10250   550
```

Notice that both polyline files contain the name of the polyline, namely `boundaryleft` and `boundaryright`, respectively. In this example, both the polylines contain *two* support points. For each of these support points, timeseries can be prescribed. In this example, we restrict ourselves to homogeneous boundary conditions. This means that we have to prescribe physical information for the first support points, i.e. for `boundaryleft_0001` and `boundaryright_0001`.

The physical information should be provided in the `bc`-file. In this example, we use the following `bc`-file:

```
[forcing]
Name          = boundaryleft_0001
Function      = timeseries
Time-interpolation = linear
Quantity      = time
Unit          = minutes since 2001-01-01
Quantity      = dischargebnd
Unit          = m3/s
 0.000000  2500.0
120.000000 3000.0
240.000000 2500.0
360.000000 2000.0
480.000000 2500.0
600.000000 3000.0
720.000000 2500.0
840.000000 2000.0
960.000000 2500.0
1080.000000 3000.0
1200.000000 2500.0
1320.000000 2000.0
1440.000000 2500.0

[forcing]
Name          = boundaryright_0001
```

```

Function      = timeseries
Time-interpolation = linear
Quantity      = time
Unit          = minutes since 2001-01-01
Quantity      = waterlevelbnd
Unit          = m
0.000000    2.50
1440.000000  2.50

```

This file couples the timeseries for the discharge to the support point named `boundaryleft_0001`. Likewise, the water level boundary, being constant in time, is coupled to the support point named `boundaryright_0001`.

The final specification of the boundary conditions is wrapped up in the external forcing file, with extension `.ext`. In this file, the connection is laid between the quantity of the boundary condition, the name of the polyline file (in which the name of the polyline itself is given) and the forcing file (in which the physical information is provided). In our example, the file `simplechannel.ext` has the following contents:

```

[boundary]
quantity      = dischargebnd
locationfile  = left.pli
forcingfile   = simplechannel.bc

[boundary]
quantity      = waterlevelbnd
locationfile  = right.pli
forcingfile   = simplechannel.bc

```

The final step is to let the model know that the external forcings file `simplechannel.ext` is the one that should be used. This is to be achieved in the MDU-file:

```

[external forcing]
ExtForceFile           =
ExtForceFileNew         =
= simplechannel.ext

```

Notice that the name is specified at the keyword `ExtForceFileNew`. The other keyword, `ExtForceFile`, should be kept empty, unless *deprecated* `.tim`-files and/or `.cmp`-files are used to prescribe the physical information for the boundaries.

8.4.1.4 Miscellaneous

In the previous sections, the most essential information on the application of boundary conditions is described. Some remaining aspects are discussed in this section.

Artificial boundary layers

Advection terms at the offshore boundary may generate an artificial boundary layer along the boundary. The advection terms containing normal gradients have to be switched off. This is

done by utilizing the keyword `jacstbnd` in the MDU-file. By default this keyword `jacstbnd` = 0, keeping the functionality inactive. The keyword can be set to `jacstbnd` = 1 to do otherwise.

Smoothing parameter boundary conditions

The solution of the shallow water equations is uniquely determined by a set of initial and boundary conditions. The boundary conditions represent the external forcing and determine the steady state solution. The deviation between the initial condition and the steady state solution generates a transient (mass spring system analogy).

In D-Flow FM, the initial conditions for the water level and velocities are obtained from:

- ◊ The results of a previous run (warm start).
- ◊ User-prescribed (space varying or uniform) input fields (cold start).

The initial values are usually inconsistent with the boundary conditions at the start time of the simulation. This will generate a transient solution consisting of waves with eigen frequencies of the model domain. These waves may be reflected at the boundaries and generate a standing wave system. The waves should be dissipated completely by bottom friction and viscosity terms or leave the domain through the open boundaries. The damping of the transient solution determines the spin-up time of the numerical model.

To reduce the amplitude of the transient wave and the spin-up time of a model, D-Flow FM has an option to switch on the boundary forcing gradually by use of a smoothing period (parameter T_{smo}). With $F_i(t)$ the initial value at the boundary, $F_b(t)$ the boundary condition signal and $F_b^{smo}(t)$ the boundary condition after smoothing, the boundary forcing is given by:

$$F_b^{smo}(t) = \alpha F_i(t) + (1 - \alpha) F_b(t), \quad (8.9)$$

with:

$$\alpha = \frac{T_{smo} - t}{T_{smo}} \quad (8.10)$$

if $t < T_{smo}$. In case $t \geq T_{smo}$, then the smoothing parameter is set to zero: $\alpha = 0$.

Smoothing is possible both for a warm and a cold start. If the initial conditions are consistent with the boundary conditions at the start time of the simulation then the smoothing time should be set to zero.

Secondary boundary conditions

In section 8.4.1.2, several types of boundary conditions are discussed. In addition, two types of boundary conditions can be applied on top of the canonical types: *normal* velocities and *tangential* velocities. The associated keyword are `normalvelocitybnd` and `tangentialvelocitybnd`, respectively.

8.4.2 Vertical boundary conditions

The vertical boundary conditions close the system of shallow water equations at the bed level and the free surface level. Without precipitation, evaporation or ground water flow, there is by definition no flow through the bottom or top surface of the water column. The vertical boundary conditions for the continuity equation (8.1) are most easily expressed when the σ co-ordinate is used for the vertical direction. In that case the free surface ($\sigma = 1$, or $z = \zeta$) and the

bottom ($\sigma = 0$, or $z = z_b$) are σ co-ordinate surfaces. ω is the vertical velocity relative to the σ -plane. The impermeability of the surface and the bottom is taken into account by prescribing the following kinematic conditions:

$$\omega|_{\sigma=1} = 0 \text{ and } \omega|_{\sigma=0} = 0 \quad (8.11)$$

8.4.2.1 Bed friction and conveyance

The boundary conditions for the momentum equations (8.3) and (8.4) at the bed ($z = z_b$) are:

$$\frac{\nu_V}{H} \frac{\partial u}{\partial \sigma} \Big|_{\sigma=0} = \frac{1}{\rho_0} \tau_{bx} \quad (8.12)$$

and

$$\frac{\nu_V}{H} \frac{\partial v}{\partial \sigma} \Big|_{\sigma=0} = \frac{1}{\rho_0} \tau_{by} \quad (8.13)$$

with τ_{bx} and τ_{by} the components of the bed stress in x - and y -direction, respectively. These equations hold in 2D and 3D parts of the model, in 1D parts the equation reduces to a single component along the channel.

For 2D depth-averaged flow the shear-stress at the bed induced by a turbulent flow is assumed to be given by a quadratic friction law:

$$\tau_b = \frac{\rho_0 g \mathbf{U} |\mathbf{U}|}{C_{2D}^2}, \quad (8.14)$$

where $|\mathbf{U}|$ is the magnitude of the depth-averaged horizontal velocity. In D-Flow FM we support the following equations to compute the C_{2D} coefficient. Since these equations can also be applied in 1D, we are using the 1D formulation based on hydraulic radius R which is typically equivalent to the water depth H in 2D. The numbers and strings following the name of the roughness formulation indicate the number and name by which the formulations are selected in the input files.

- ◊ **Chézy formulation:** 0, Chezy

$$C_{2D} = \text{Chézy coefficient [m}^{1/2}/\text{s}. \quad (8.15)$$

- ◊ **Manning's formulation:** 1, Manning

As formulated by [Manning \(1891\)](#), and also known as Gauckler–Manning–Strickler formulation.

$$C_{2D} = \frac{\sqrt[6]{R}}{n} \quad (8.16)$$

where:

$$\begin{aligned} R &\text{ is the hydraulic radius} \\ n &\text{ is the Manning coefficient [m}^{-1/3}\text{s].} \end{aligned}$$

- ◊ **Log law of the Wall:** 2, WallLawNikuradse

$$C_{2D} = \frac{\sqrt{g}}{\kappa} \ln \left(1 + \frac{30R}{ek_n} \right) \quad (8.17)$$

where:

- R is the hydraulic radius
 k_n is the Nikuradse roughness length [m].
- ◇ **White-Colebrook's formulation:** 3, WhiteColebrook
As formulated by Colebrook and White (1937) and Colebrook (1939).

$$C_{2D} = 18^{10} \log \left(\frac{12R}{k_n} \right) \quad (8.18)$$

where:

- R is the hydraulic radius
 k_n is the Nikuradse roughness length [m].

Alluvial bed properties and land use data can be used to determine the roughness and flow resistance. Specific functionality has been implemented to make the translation of bed and land use properties into roughness and flow resistance characteristics. See section 14.2 for details.

8.4.2.2 Conveyance in 2D

Bed friction often plays a major role in the discharge capacity and expected maximum water levels of channels and gullies. If we model a trapezoidal channel with sloping sidewalls on a grid with 6 grid cells across the width of the channel, a typical cross section using the standard bed representation with uniform depth appears per cell (Figure 8.10a). In D-Flow FM, we allow for representation of a locally sloping bed as shown in Figure 8.10b.

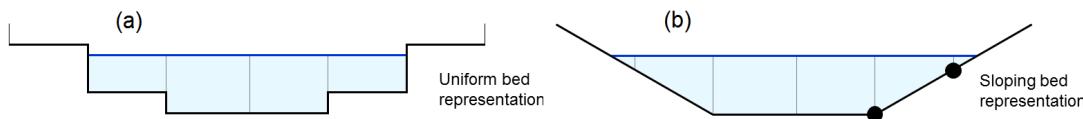


Figure 8.10: Bed representation with uniform depth levels (a), and locally sloping bed (b).

The most left and the most right cell are not yet wet in the uniform bed representation. In the sloping bed representation, these outer cells are partly wet, yielding a more accurate description of the total wet cross sectional area. The user can select whether to apply this more accurate description or not. This can only be done only in combination with a net-node based bathymetry description, i.e. using the keyword `Ibedlevtype = 3` in the `.mdu` file.

For the bed friction in 2D models, one implicitly assumes a fully developed vertical velocity profile, using a logarithmic function of the water depth for the White-Colebrook bed friction formulation, or a one-sixth power function of the water depth for the Manning formulation. In a sloping cell, the local water depth is varying over the width of the cell. In the deeper part flow velocities will be higher than in the shallower part. Bed stresses will vary over the width of a cell with water depth and with the velocity. Integrating the stresses over the width of a cell one can derive the resulting total stress.

The bed friction term is not only a function of the normal velocity component in the direction of the flow link itself, but also depends on the tangential velocity component and with that on the total velocity. For each of the four components water depth, normal velocity, total velocity and Chézy parameter, we assume a linear variation over the width (Figure 8.11).

If we have in the `.mdu` file `Conveyance2D = 3`, the 2D analytic conveyance description using these four linearly varying components is applied. This option shows best grid conver-

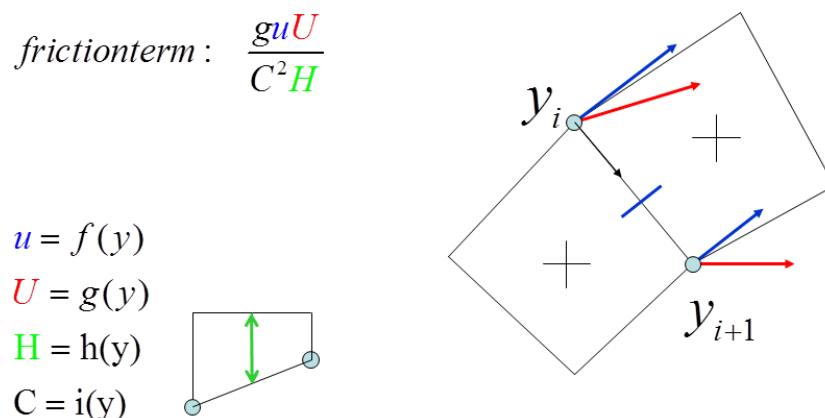


Figure 8.11: A schematic view of the linear variation over the width for calculating the flow parameters.

gence behaviour. Good grid convergence implies that the converged answer can be achieved on a coarser grid, thus saving computational costs.

If one sets `Conveyance2D = 2`, the tangential velocity component is assumed zero. This method is only applicable on curvilinear grids that are aligned with the flow direction.

If one sets `Conveyance2D = 1`, both the normal and tangential velocity component are assumed constant over the width. Effectively one obtains the so called 'lumped' bed friction approach, with hydraulic radius $R = A/P$, A being the wet cross sectional area and P the wet perimeter. This method works equally well as methods 2 and 1, provided that there is sufficient resolution of a gully in the grid. It is found that when a gully is resolved by more than about 10 or 12 cells, it provides almost identical answers as method 2, while saving some 10 % computational overhead compared to method 2.

Setting `Conveyance2D = -1`, the hydraulic radius is based on a uniform bed level at the velocity point taken as the average bed level of the two surrounding net nodes (depth points in WAQUA terms). This option is identical to the combination of WAQUA option `dpuopt = mean i.c.m.` `dpsopt = max`.

Setting `Conveyance2D = 0`, the hydraulic radius is based on a uniform bed level at the velocity point taken as the average bed level of the two surrounding water level points. This option is not advisable because cell bed levels are taken as the minimum value of the bed levels of attached link. So a min-max operator is invoked, which is not suitable for accuracy.

8.4.3 Shear-stresses at closed boundaries

A closed boundary is situated at the transition between land and water. At a closed boundary, two boundary conditions have to be prescribed. One boundary condition has to do with the flow normal to the boundary and the other one with the shear-stress along the boundary. A closed sidewall is always considered as impermeable. For the shear stress along the boundary, the possible conditions to be prescribed are free-slip (zero tangential shear-stress), partial-slip and full-slip.

For large-scale simulations, the influence of the shear-stresses along closed boundaries can be neglected. Free slip is then applied for all closed boundaries. For simulations of smallscale flow (e.g. laboratory scale), the influence of the side walls on the flow may no longer be neglected. This option has been implemented for closed boundaries and dry points *but not for thin dams*. The reason is that the shear stress at a thin dam is dependent on the side (double valued).

Along the side walls, the tangential shear stress is calculated based on a logarithmic law of the wall. The friction velocity u_* is determined by the logarithmic law for a rough wall, with side wall roughness y_0 and the velocity in the first grid point near the side wall. Let Δy_s be the grid size normal to the sidewall, then:

$$|\vec{u}_{\text{sidewall}}| = \frac{u_*}{\kappa} \ln \left(1 + \frac{\Delta y_s}{2y_0} \right) \quad (8.19)$$

The choice for the way tangential shear stress are computed can be inserted through the key `irov` in the MDU-file (under `[physics]`):

- ◊ the case `irov = 0` treats the side walls as free-slip walls (default),
- ◊ the case `irov = 1` treats the side walls as partial-slip walls,
- ◊ the case `irov = 2` treats the side walls as no-slip walls.

If the choice for partial-slip walls is made, then a specific wall roughness value should be prescribed in the MDU-file. For this, the keyword `wall_ks` should be used (under `[physics]` in the MDU-file). The computational engine interprets this value as Nikuradse roughness k_s . The value for y_0 is computed as $y_0 = k_s/30$.

8.5 Artificial mixing due to sigma-coordinates

The σ -transformation is boundary-fitted in the vertical. The bottom boundary and free surface are represented smoothly. The water column is divided into the same number of layers independent of the water depth. In a σ -model, the vertical resolution increases automatically in shallow areas.

For steep bottom slopes combined with vertical stratification, σ -transformed grids introduce numerical problems for the accurate approximation of horizontal gradients both in the baroclinic pressure term and in the horizontal diffusion term. Due to truncation errors artificial vertical mixing and artificial flow may occur, [Leendertse \(1990\)](#) and [Stelling and Van Kester \(1994\)](#). This artificial vertical transport is sometimes called "creep".

Let z_b be the position of the bed and H the total water depth. If we consider the transformation from Cartesian co-ordinates to σ co-ordinates, defined by:

$$x = x^*, \quad y = y^*, \quad \sigma = \frac{z - z_b}{H} \quad (H = \zeta - z_b) \quad (8.20)$$

as result $\sigma = 0$ at the bed level and $\sigma = 1$ at the free surface. The horizontal pressure gradient reads:

$$\frac{\partial p}{\partial x} = \frac{\partial p^*}{\partial x^*} \frac{\partial x^*}{\partial x} + \frac{\partial p^*}{\partial \sigma} \frac{\partial \sigma}{\partial x} = \frac{\partial p^*}{\partial x^*} - \frac{1}{H} \left(\frac{\partial z_b}{\partial x} + \sigma \frac{\partial H}{\partial x} \right) \frac{\partial p^*}{\partial \sigma}. \quad (8.21)$$

In case of vertical stratification near steep bottom slopes, small pressure gradients at the left-hand side may be the sum of relatively large terms with opposite sign at the right-hand side.

Small truncation errors in the approximation of both terms result in a relatively large error in the pressure gradient.

This artificial forcing produces artificial flow.

The truncation errors depend on the grid sizes Δx and Δz .

Observations of this kind has led to the notion of "hydrostatic consistency", see also [Figure 8.12](#). In the notation used by [Haney \(1991\)](#) this consistency relation is given by:

$$\left| \frac{\sigma}{H} \frac{\partial H}{\partial x} \right| < \left| \frac{\partial \sigma}{\partial x} \right| \quad (8.22)$$

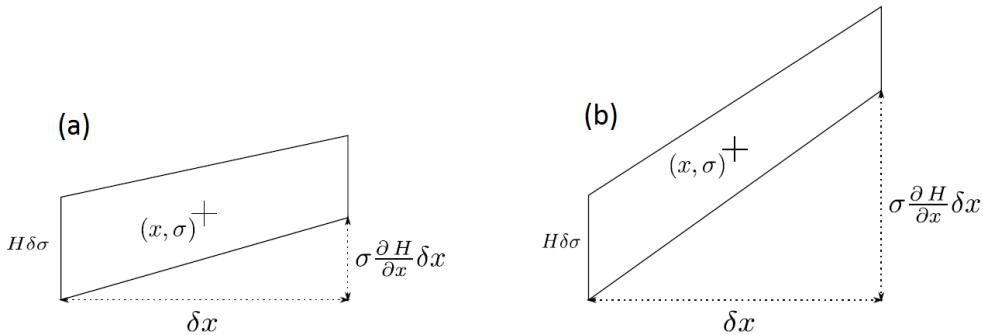


Figure 8.12: Example of a hydrostatic consistent and inconsistent grid; (a) $H\delta\sigma > \sigma \frac{\partial H}{\partial x} \delta x$, (b) $H\delta\sigma < \sigma \frac{\partial H}{\partial x} \delta x$

From this equation, it can be seen that by increasing the number of σ -levels the consistency condition will eventually be violated.

Similarly, for the horizontal diffusion term, the transformation from Cartesian co-ordinates to σ co-ordinates leads to various cross derivatives. For example, the transformation of a simple second order derivative leads to:

$$\frac{\partial^2 c}{\partial x^2} = \frac{\partial^2 c^*}{\partial x^{*2}} + \left(\frac{\partial \sigma}{\partial x} \right)^2 - \frac{\partial^2 c^*}{\partial \sigma^2} + 2 \frac{\partial \sigma}{\partial x} - \frac{\partial^2 c^*}{\partial x^* \partial \sigma} + \frac{\partial^2 \sigma}{\partial x^2} - \frac{\partial c^*}{\partial \sigma} \quad (8.23)$$

For such a combination of terms it is difficult to find a numerical approximation that is stable and positive, see [Huang and Spaulding \(1996\)](#). Near steep bottom slopes or near tidal flats where the total depth becomes very small, truncations errors in the approximation of the horizontal diffusive fluxes in σ -co-ordinates are likely to become very large, similar to the horizontal pressure gradient.

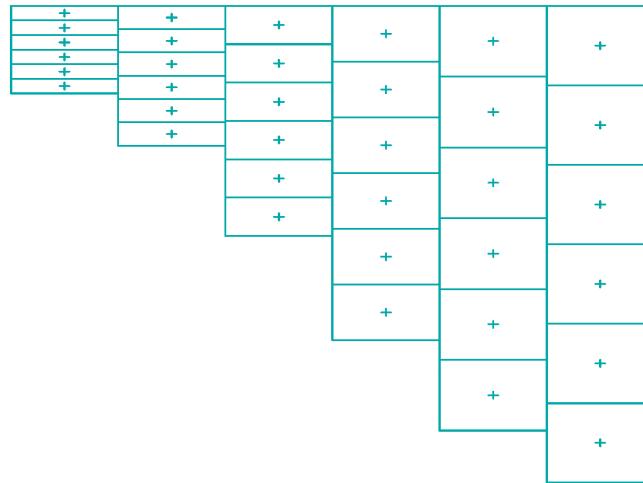


Figure 8.13: Finite Volume for diffusive fluxes and pressure gradients

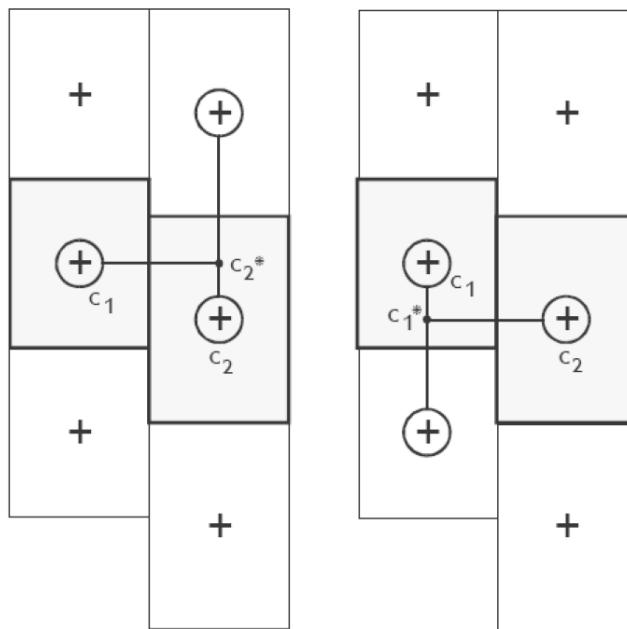


Figure 8.14: Left and right approximation of a strict horizontal gradient

In D-Flow FM the stress tensor is redefined in the σ co-ordinate system assuming that the horizontal length scale is much larger than the water depth (Blumberg and Mellor, 1985) and that the flow is of boundary-layer type. The horizontal gradients are taken along σ -planes. This approach guarantees a positive definite operator, also on the numerical grid (Beckers *et al.*, 1998).

If the same approach is used for the horizontal diffusion operator in the transport equation:

$$\frac{\partial^2 c}{\partial x^2} \approx \frac{\partial^2 c^*}{\partial x^{*2}} \quad (8.24)$$

Horizontal diffusion will lead to vertical transport of matter through vertical stratification interfaces (pycnocline) which is nonphysical. A more accurate, strict horizontal discretization is needed.

In D-Flow FM an option is available that minimises artificial vertical diffusion and artificial flow due to truncation errors. A method has been implemented which gives a consistent, stable and monotonic approximation of both the horizontal pressure gradient and the horizontal diffusion term, even when the hydrostatic consistency condition equation is not fulfilled. This "anti-creep" option is based upon a Finite Volume approach; see [Figure 8.13](#). The horizontal diffusive fluxes and baroclinic pressure gradients are approximated in Cartesian co-ordinates by defining rectangular finite volumes around the σ -co-ordinate grid points. Since these boxes are not nicely connected to each other, see [Figure 8.14](#), an interpolation in z co-ordinates is required to compute the fluxes at the interfaces.

Since the centres of the finite volumes on the left-hand side and right-hand side of a vertical interval are not at the same vertical level, a z -interpolation of the scalar concentration c is needed to compute strictly horizontal derivatives. The values obtained from this interpolation are indicated by c_1^* and c_2^* respectively in [Figure 8.14](#). ([Stelling and Van Kester, 1994](#)) apply a non-linear filter to combine the two consistent approximations of the horizontal gradient,

$$\begin{aligned}
 s_1 &= (c_2^* - c_1) / \Delta x \text{ and } s_2 = (c_2 - c_1^*) / \Delta x \\
 \text{If } s_1 \times s_2 < 0 \text{ Then} \\
 \frac{\Delta c}{\Delta x} &= 0 \\
 \text{Else} \\
 \frac{\Delta c}{\Delta x} &= \text{sign}(s_1) \times \min(|s_1|, |s_2|) \\
 \text{Endif}
 \end{aligned} \tag{8.25}$$

If an interval has only grid boxes at one side, the derivative is directly set to zero. The horizontal fluxes are summed for each control volume to compute the diffusive transport. The integration of the horizontal diffusion term is explicit with time step limitation:

$$\Delta t \leq \frac{1}{D_H} \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) \tag{8.26}$$

The derivatives are used in the integral for the baroclinic pressure force in the momentum equation:

$$P_x(x, z) = \frac{1}{\rho_0} \int_z^\zeta g \frac{\partial \rho(x, s)}{\partial x} ds \tag{8.27}$$

Originally, this approach was implemented in Delft3D-FLOW. [Slørdal \(1997\)](#) stated that the above approximation may sometimes produce errors of the same sign which leads to a systematic underestimation of the baroclinic pressure term. This underestimation can be ascribed to the non-linear filter, which selects the minimum of the two gradients under consideration. This limiter is fully analogous to the min-mod limiter used for the construction of monotone advection schemes ([Hirsch, 1990](#)). Since the same approximation of the horizontal gradient is used for the horizontal diffusion flux, it is important to ensure that the difference operator is positive definite in order to get physically realistic solutions. The maximum and minimum of a variable being transported by diffusion do not increase or decrease (min-max principle). By taking the minimum of the gradients, [Stelling and Van Kester \(1994\)](#) show that, the min-max principle is fulfilled. [Beckers et al. \(1998\)](#) show that any nine-point consistent linear discretization of the horizontal diffusion on the σ -grid does not fulfil the min-max principle. From numerical tests [Slørdal \(1997\)](#) concluded that the underestimation is reduced by increasing the vertical resolution, but is sometimes enhanced by increasing the horizontal resolution.

Let s_4 be a consistent approximation of the horizontal gradient $s_4 = (s_1 + s_2)/2$. [Slørdal \(1997\)](#) suggested to take s_4 as approximation of the horizontal gradient. He calls his approach the "modified Stelling and Van Kester scheme". It is equivalent to linear interpolation at a certain z -level before taking the gradient. It is more accurate than taking the minimum of the absolute value of the two slopes s_1 and s_2 but it does not fulfil the min-max principle for the diffusion operator. It may introduce wiggles and a small persistent artificial vertical diffusion (except for linear vertical density distributions). Due to the related artificial mixing, stratification may disappear entirely for long term simulations, unless the flow is dominated by the open boundary conditions.

By introducing an additional approximation of the horizontal gradient in the filter algorithm defined by $s_3 = (c_2 - c_1)/\Delta x$, the stringent conditions of the minimum operator can be relaxed somewhat. The drawback of underestimation of the baroclinic pressure force reported by [Slørdal \(1997\)](#) can be minimised without loosing that the method fulfils the min-max principle. This third gradient s_3 , which is consistent for $\min(|s_1|, |s_2|) < s_3 < \max(|s_1|, |s_2|)$, has point-to-point transfer properties and therefore leads to a positive scheme for sufficiently small time steps. The following non-linear approach presently available in D-Flow FM is both consistent and assures the min-max principle:

```
If    $s_1 \times s_2 < 0$  Then
     $\frac{\Delta c}{\Delta x} = 0$ 
Elseif   $|s_4| < |s_3|$  Then
     $\frac{\Delta c}{\Delta x} = s_4$ 
Elseif   $\min(|s_1|, |s_2|) < |s_3| < \max(|s_1|, |s_2|)$  Then
     $\frac{\Delta c}{\Delta x} = s_3$ 
Else
     $\frac{\Delta c}{\Delta x} = \text{sign}(s_1) \min(|s_1|, |s_2|)$ 
Endif
```

(8.28)

The method requires a binary search to find the indices of neighbouring grid boxes, which is time consuming. The increase in computation time is about 30 %.

If the streamlines are strictly horizontal, transport of matter discretised on a σ co-ordinate grid may still generate some numerical vertical diffusion by the discretisation of the advection terms.

8.6 Secondary flow

The flow in a river bend is basically three-dimensional. The velocity has a component in the plane perpendicular to the river axis. This component is directed to the inner bend near the riverbed and directed to the outer bend near the water surface, see Figure 8.15.

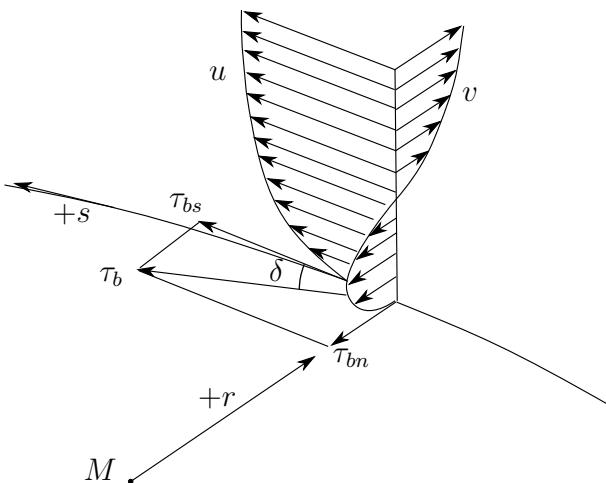


Figure 8.15: Vertical profile secondary flow (v) in river bend and direction bed stress

This so-called "secondary flow" (spiral motion) is of importance for the calculation of changes of the riverbed in morphological models and the dispersion of matter. In a 3D model the secondary flow is resolved on the vertical grid, but in 2D depth-averaged simulations the secondary flow has to be determined indirectly using a secondary flow model. It strongly varies over the vertical but its magnitude is small compared to the characteristic horizontal flow velocity.

8.6.1 Definition

The secondary flow will be defined here as the velocity component $v(\sigma)$ normal to the depth-averaged main flow. The spiral motion intensity of the secondary flow I is a measure for the magnitude of this velocity component along the vertical:

$$I = \int_0^1 |v(\sigma)| d\sigma \quad (8.29)$$

A vertical distribution for a river bend is given in Figure 8.15. The spiral motion intensity I leads to a deviation of the direction of the bed shear stress from the direction of the depth-averaged flow and thus affects the bedload transport direction. This effect can be taken into account in morphological simulations.

The component of the bed shear stress normal to the depth-averaged flow direction τ_{br} reads:

$$\tau_{br} = -2\rho\alpha^2 \left(1 - \frac{\alpha}{2}\right) |\mathbf{u}| I \quad (8.30)$$

where α is defined in Equation (8.41) and $|\mathbf{u}|$ is the magnitude of the depth-averaged velocity. To take into account the effect of the secondary flow on the depth-averaged flow, the depth-averaged shallow water equations have to be extended with:

- ◊ An additional advection-diffusion equation to account for the generation and adaptation of the spiral motion intensity.
- ◊ Additional terms in the momentum equations to account for the horizontal effective shear stresses originating from the secondary flow.

8.6.2 Depth-averaged continuity equation

The depth-averaged continuity equation is given by:

$$\frac{\partial h}{\partial t} + \frac{\partial hu}{\partial x} + \frac{\partial hv}{\partial y} = Q \quad (8.31)$$

where u and v indicate the depth-averaged velocities along Cartesian axis.

8.6.3 Momentum equations in horizontal direction

The momentum equations in x - and y -direction are given by:

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} - fu = -\frac{1}{\rho_0} P_x - \frac{gu\sqrt{u^2 + v^2}}{C_{2D}^2 h} + F_x + F_{sx} + M_x \quad (8.32)$$

$$\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} + fv = -\frac{1}{\rho_0} P_v - \frac{gv\sqrt{u^2 + v^2}}{C_{2D}^2 h} + F_y + F_{sy} + M_y \quad (8.33)$$

The fourth term in the right-hand side represents the effect of the secondary flow on the depth-averaged velocities (shear stresses by depth-averaging the non-linear acceleration terms).

8.6.4 Effect of secondary flow on depth-averaged momentum equations

To account for the effect of the secondary flow on the depth-averaged flow, the momentum equations have to be extended with additional shear stresses. To close the equations these stresses are coupled to parameters of the depth-averaged flow field. The main flow is assumed to have a logarithmic velocity profile and the secondary flow originates from a multiplication of a universal function with the spiral motion intensity, see Kalkwijk and Booij (1986). Depth averaging of the 3D equations leads to correction terms in the depth-averaged momentum equations for the effect of spiral motion:

$$F_{sx} = \frac{1}{h} \left(\frac{\partial h T_{xx}}{\partial x} + \frac{\partial h T_{xy}}{\partial y} \right) \quad (8.34)$$

$$F_{sy} = \frac{1}{h} \left(\frac{\partial h T_{xy}}{\partial x} + \frac{\partial h T_{yy}}{\partial y} \right) \quad (8.35)$$

with the shear-stresses, resulting from the secondary flow, modelled as:

$$T_{xx} = -2\beta uv \quad (8.36)$$

$$T_{xy} = T_{yx} = \beta(u^2 - v^2) \quad (8.37)$$

$$T_{yy} = 2\beta uv \quad (8.38)$$

and:

$$\beta = \beta^* \frac{h}{R_s^*} \quad (8.39)$$

$$\beta^* = \beta_c (5\alpha - 15.6\alpha^2 + 37.5\alpha^3) \quad (8.40)$$

$\beta_c \in [0, 1]$, correction coefficient specified by you,

$$\alpha = \frac{\sqrt{g}}{\kappa C_{2D}} < \frac{1}{2} \quad (8.41)$$

with R_s^* the effective radius of curvature of a 2D streamline to be derived from the intensity of the spiral motion and κ the Von Kármán constant. The spiral motion intensity is computed by Equation (8.42). The limitation on α , Equation (8.41), is set to ensure that the length scale L_a in Equation (8.49) is always positive. For $\beta_c = 0$, the depth-averaged flow is not influenced by the secondary flow.

Remark:

- ◊ Equation (8.41) effectively means a lower limit on C_{2D} .

8.6.5 The depth averaged transport equation for the spiral motion intensity

The variation of the spiral motion intensity I in space and time, is described by a depth-averaged advection-diffusion equation:

$$\frac{\partial hI}{\partial t} + \frac{\partial uhI}{\partial x} + \frac{\partial vhI}{\partial y} = h \frac{\partial}{\partial x} \left(D_H \frac{\partial I}{\partial x} \right) + h \frac{\partial}{\partial x} \left(D_H \frac{\partial I}{\partial y} \right) + hS \quad (8.42)$$

with:

$$S = -\frac{I - I_e}{T_a} \quad (8.43)$$

$$I_e = I_{be} - I_{ce} \quad (8.44)$$

$$I_{be} = \frac{h}{R_s} |\mathbf{u}| \quad (8.45)$$

$$I_{ce} = f \frac{h}{2} \quad (8.46)$$

$$|\mathbf{u}| = \sqrt{u^2 + v^2} \quad (8.47)$$

$$T_a = \frac{L_a}{|\mathbf{u}|} \quad (8.48)$$

$$L_a = \frac{(1 - 2\alpha)h}{2\kappa^2\alpha} \quad (8.49)$$

and R_s the radius of curvature of the stream-line defined by:

$$\frac{u_s}{R_s} = -\frac{\partial u_r}{\partial s} \quad (8.50)$$

with u_s and u_r the components along and perpendicular to the streamline. The effective radius of curvature to be used for the evaluation of the coefficient β , Equation (8.40), reads:

$$R_s^* = \frac{h |\mathbf{u}|}{I} \quad (8.51)$$

To guarantee stability the effective radius of curvature is bounded by the following empirical relation:

$$R_s^* \geq 10h \quad (8.52)$$

The above formulas account for two sources of secondary flow:

- ◊ The centrifugal force in case of curved streamlines, I_{be} .
- ◊ The effect of the Coriolis force, I_{ce} .

8.7 Drying and flooding

Estuaries and coastal embayments contain large, shallow, and relatively flat areas separated and interleaved by deeper channels and creeks. When water levels are high, the entire area is covered by water. But as tide falls, the shallow areas are exposed, and ultimately the flow is confined only to the deeper channels. The dry tidal flats may occupy a substantial fraction of the total surface area. The accurate reproduction of covering or uncovering of the tidal flats is an important feature of numerical flow models based on the shallow water equations.

Many rivers have compound channels, consisting of a main channel that always carries flow (the summer bed) and one or two flood plains which only convey flow during extreme river discharges (the winter bed). The summer bed is surrounded by low dykes, which could be overtapped if the river discharge increases. The winter-bed is surrounded by much higher dykes, which are designed to protect the polders against water levels due extreme river discharges. The flooding of the flood plains increases the drainage capacity of the river and reduces the local water level gradients.

In a numerical model, the process of drying and flooding is represented by removing grid points from the flow domain that become *dry* when the tide falls and by adding grid points that become *wet* when the tide rises. Drying and flooding is constrained to follow the sides of grid cells. In this section, we specify the algorithms which have been used to determine the moment when a grid cell (water level point) or cell boundary (velocity point) becomes dry or wet. Drying and flooding gives a discontinuous movement of the closed boundaries and may generate small oscillations in water levels and velocities. The oscillations introduced by the drying and flooding algorithm are small if the grid sizes are small and the bottom has smooth gradients.

Essential elements of the wetting and drying algorithm are the definition of the water level, the definition of the bed level and the criteria for setting a velocity and/or water level point wet or dry. In the following subsections, these three items will be discussed.

8.7.1 Definitions

In [section 8.3.1](#), the locations of the primary flow variables (water level and flow velocity) have been described. Through [Figure 8.3](#) it has been clarified that the water level is computed at the location of the circumcenter (cell center), whereas the face normal flow velocity is computed at the midpoint of each cell face (face center).

For the computation of these two primary variables, the level of the bed must be known both at the cell center and at the face center. The user can specify the way these values are interpreted from the available bathymetry by means of the MDU-file keywords `bedlevtyp` and `conveyance2D`. For a proper understanding of the possibilities of D-Flow FM, [Figure 8.16](#) is provided with a three-dimensional representation of two adjacent triangular cells.

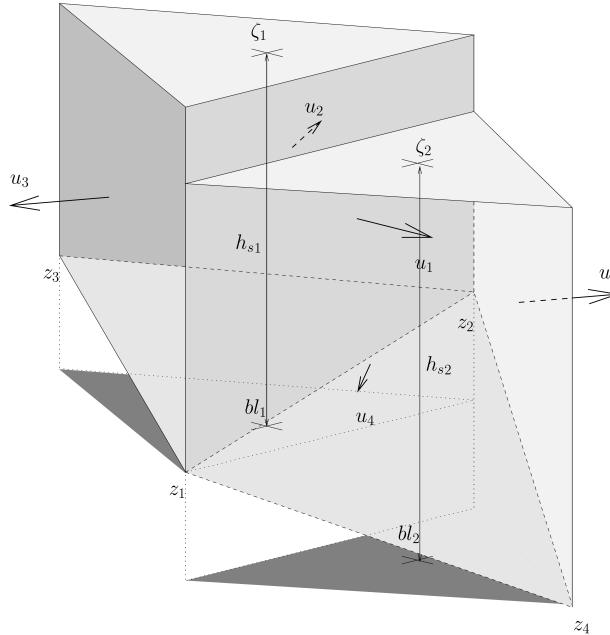


Figure 8.16: Definition of the water levels, the bed levels and the velocities in case of two adjacent triangular cells.

To the keyword `bedlevtyp`, the values 1, 2, 3, 4, 5 and 6 can be assigned. For the keyword `conveyance2D`, the values -1, 0, 1, 2 and 3 are available.

The most common case is the definition of the bed levels at the corner nodes of the cell and a choice for the keyword `bedlevtyp = 3`. Depending on the choice for `conveyance2D`, the face center bed levels and cell center bed levels are computed.

8.7.1.1 Piecewise constant approach for the bed level

In principle, the bed levels are considered as piecewise constant in space. This approach is followed if `conveyance2D < 1`. The keyword `bedlevtyp` can be used in combination with the piecewise constant approach as highlighted in the table below. Note that the bed level is defined with respect to the reference level (not to be confused with a water depth given as a positive value downwards). With `conveyance2D < 1`, we have:

Keyword	Value	Cell center bed level	Face center bed level
bedlevtyp	1	user specified	the highest cell center bed level considering the two cells next to the face
bedlevtyp	2	the lowest face center bed level considering all the faces of the cell	user specified
bedlevtyp	3	the lowest face center bed level considering all the faces of the cell	the mean corner bed level considering the two corner nodes the face is connecting
bedlevtyp	4	the lowest face center bed level considering all the faces of the cell	the minimum corner bed level considering the two corner nodes the face is connecting
bedlevtyp	5	the lowest face center bed level considering all the faces of the cell	the maximum corner bed level considering the two corner nodes the face is connecting
bedlevtyp	6	the mean corner bed level considering all the corners of the cell	the highest cell center bed level considering the two cells next to the face

The former Delft3D-FLOW version, running on curvilinear meshes, has utilized the piecewise constant approach for bed levels as well. The treatment of the bed level itself is, however, different from D-Flow FM. In Delft3D-FLOW, the user can distinctly specify the treatment type for the cell center bed level and the face center bed level. In D-Flow FM, the choice for the one implies the choice for the other. A strict, *exact* match of settings for which Delft3D-FLOW and D-Flow FM treat the bed similarly, is not facilitated. Hence, the user himself should take care in comparing the Delft3D-FLOW results and D-Flow FM results when it comes to the bed level treatment settings.

8.7.1.2 Piecewise linear approach for the bed levels

A piecewise linear bed level approach can be chosen for through setting `conveyance2D` ≥ 1 . In this case, only the approach for `bedlevtyp` equal to 3, 4 and 5 is affected. With `conveyance2D` ≥ 1 , we have:

Keyword	Value	Cell center bed level	Face center bed level
bedlevtyp	3, 4, 5	the lowest corner node bed level considering all the nodes of the cell	linearly varying from the bed level at the one corner node to the bed level at the other corner node

Notice that the choice for either `bedlevtyp` equal to 3, 4 or 5 does not imply different bed level treatment approaches. For the case `conveyance2D` ≥ 1 , the bed is assumed linearly varying within a face only to compute the wet cross-sectional area of the vertical face; it should, however, be remarked that for the computation of the water column volume in a cell, this linear variation of the bed is *not* taken into account.

8.7.1.3 Hybrid bed level approach

In addition to the previously described bed level treatment approaches, D-Flow FM facilitates a hybrid approach for the computation of the cell center bed level by means of the keywords `blminabove` and `blmeanbelow`. In this approach, the cell center bed level is computed as the mean of the associated face center bed levels, be it only below the user specified level `blmeanbelow`. For levels above the user specified level `blminabove`, the minimum value of the associated face center bed levels is used. In between these two user specified levels, the bed levels are constructed by means of a linear interpolation between the two approaches' results.

8.7.2 Specification in Delta Shell

The specification of the treatment of the bed level locations can be achieved through the tab fields 'Numerical Parameters' and 'Physical Parameters'.

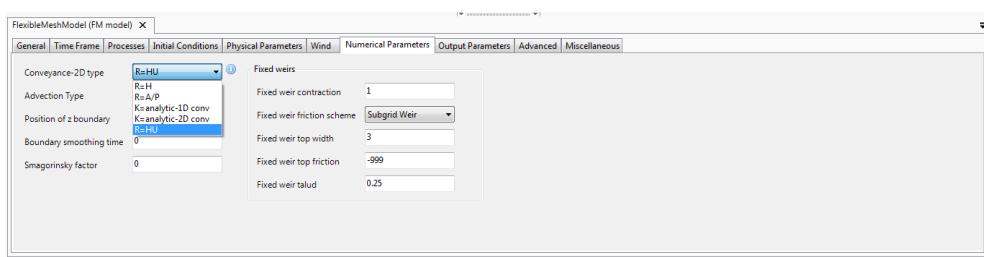


Figure 8.17: Specification of the conveyance option in Delta Shell.

The tab field 'Numerical Parameters' contains the choice for the conveyance options: see Figure 8.17).

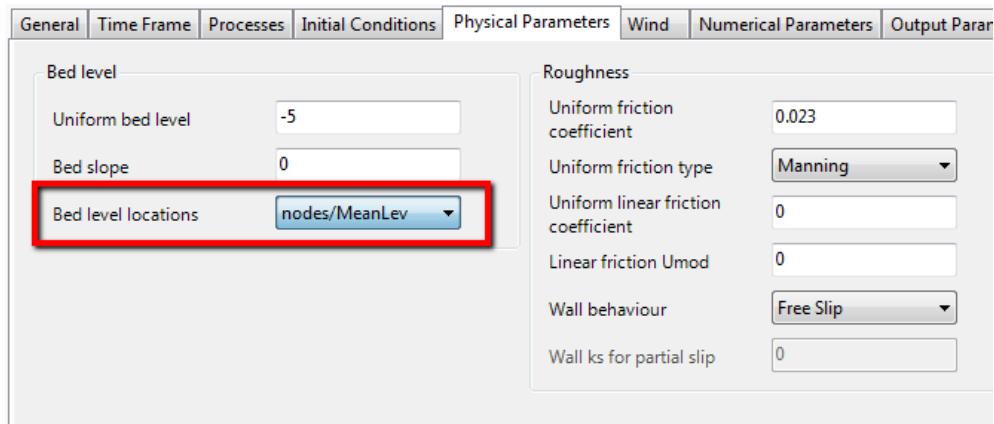


Figure 8.18: Specification of the bed level treatment type in Delta Shell.

The tab field 'Physical Parameters' contains the choice for the Bed level locations. Five options are facilitated: `bedlevtyp` numbers 1 up to 5; the option 6 is not facilitated in the user interface.

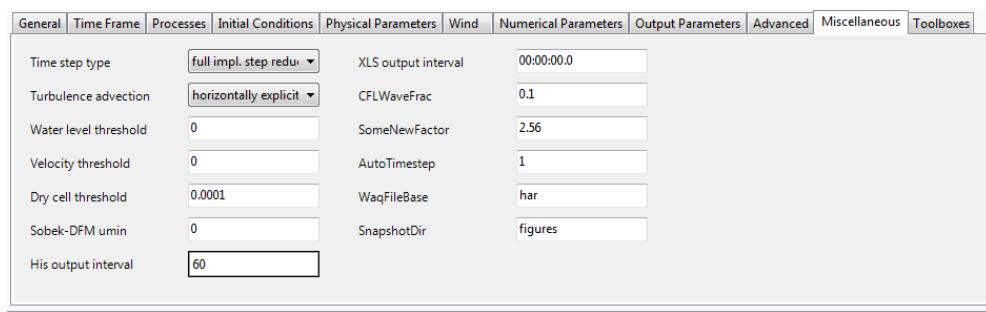


Figure 8.19: Specification of the hybrid bed options (with keywords `blminabove` and `blmeanbelow`).

By means of the tab field ‘Miscellaneous’, the hybrid options with the keywords `blminabove` and `blmeanbelow` can be enabled.

8.8 Intakes, outfalls and coupled intake-outfalls

Many engineering studies concern the design of intakes and outfalls. For instance, studies about positioning of waste water diffusers or coupled intake-outfall design for cooling water at power plants. Intakes and outfalls can be modeled using sinks and sources. A source is a point in the model where a discharge Q in [m^3/s] is prescribed by a time series ASCII file ([section C.4](#)) with at least two columns and, depending on the model settings, possibly more. For each optional constituent that is modelled, an extra column is required. Best known constituents are salinity and temperature and the values in the time series file should be specified as salinity increase in [ppt] and temperature increase in [$^\circ\text{C}$].

- 1 When salinity and temperature are not used in the computation:
Model expects two columns, where the first column is the time in minutes, the second is the discharge in [m^3/s].
- 2 When either salinity or temperature is used in the computation:
Model expects three columns, where the first column is the time in minutes, the second is the discharge in [m^3/s], the third column contains either the salinity (increase) in [ppt] or the temperature (increase) in [$^\circ\text{C}$]. Note that third column can be either salinity or temperature depending which constituent is used in the model and which is not.
- 3 In general, when other constituents are also modelled (e.g. one or more sediment fractions, passive tracers), the column order is as follows: Time, Discharge, Salinity?, Temperature?, (Sed.frac 1, ..., Sed.frac n)?, Spiral flow intensity?, (Tracer 1, ..., Tracer n)?.

The location of the source or sink is specified in a polyline file ([section C.2](#)), containing a polyline with either multiple points or just one point. If two or more points are specified, a coupled source sink pair is made, the first point is the FROM (or sink) point, the last point is the TO (or source) point. Three variants may occur:

- 1 Sink point side lies inside a grid cell A, source point also lies inside a grid cell B (A may equal B, but that is rarely useful): water is extracted from cell A and transported to B.
- 2 Sink point lies outside of the grid, source point lies in a grid cell B: water is discharged into B, a bit like an inflow discharge boundary condition.
- 3 Sink side lies inside a grid cell A, source side lies outside of the grid: water is extracted from A, a bit like an outflow discharge boundary condition.

Specifying a negative discharge value effectively interchanges the role of the source and sink

points. If only one point is in the `<*.pli>` file, it is assumed to be a source point. Specifying a negative discharge turns the source into a sink.

For 3D computations, the polyline file should have a third column with z -values. It is good practice to change the `<*.pli>` file into a `<*.pliz>` file. The z -values are used to determine in which vertical grid cell the source and/or sink lie. The layer number can vary in time in sigma models.

In the case of a coupled pair of sink source points (variant 1), the third and fourth column with the salinity and temperature specification are interpreted as delta salinity and delta temperature. So the values at the source point become the values at the sink point plus the specified delta values.

The sources and sinks are specified in the `<*.ext>` file in a way similar to the boundary conditions:

```
QUANTITY=discharge_salinity_temperature_sorsin
FILENAME=chan1_westeast.pli # A file 'chan1_westeast.tim', with same basename
                           # as the polyline should be present
FILETYPE=9
METHOD =1
OPERAND =O
AREA   =1.5
```

The specified area in the last line of this file determines (together with the specified discharge) the amount of momentum uQ that is released at the source point. This is currently only implemented in advection scheme 33 (cf. [D-Flow FM TRM \(2015\)](#) on Momentum advection). Omitting this line or specifying a zero area switches off the release of momentum at the source point. The direction of the discharged momentum is in direction of the last two points of the polyline. So a one point polyline is momentumless. Both in 2D and 3D, the momentum can only be directed in horizontal direction.

Sources and sinks are treated explicitly in the numerical scheme. This implies that the actual discharged or extracted amounts of water are limited by the velocity Courant condition $Q\Delta t/V < 1$. Not doing so could lead to severe timestep restrictions. Consider for instance a specified extraction in case the extraction point has fallen dry. In that case, a real pump would not be able to extract water and the specified extraction is more likely an input error than an actual description of a physically feasible situation. So, we limit the specified discharges to the local velocity Courant restriction. By specifying observation cross-sections ([section 5.4.2.3](#)), one can compare prescribed discharges to the discharges that were actually realised in the model. In case of large differences, the discharge should probably be distributed over a larger number of gridcells, or the extraction channel that feeds the extraction point should be dredged.

When modelling freshwater inflow from a river into a sea, the vertical distribution of the discharge that one should specify depends on the amount of detail available for modelling the saline water - fresh water interface. If the river is modeled with sufficient detail, and the river is well mixed at the upstream model boundary, the mixing process can be part of the modelling and the river can be discharged over the whole vertical. If the mixing process cannot be resolved in the model, for instance if the river is modeled as a point discharge adjacent to a closed boundary, make sure that the whole river is discharged into the top layer or in the top layers, depending on the estimated thickness of the fresh water plume at the discharge cell.

If one would have distributed the river discharge over the whole vertical, the size of the fresh water plume would be underestimated because of too much mixing at the river discharge cell.

For example input files see example directories:

- ◊ <f17_sources_sinks/c010_sourcesink_2D/>
- ◊ <f17_sources_sinks/c020_sourcesink_3D/>

8.9 Equations of state for the density

The density of water ρ is a function of salinity (s) and temperature (t).

In D-Flow FM we copied the implementation from Delft3D-FLOW of the formulation derived by [Eckart \(1958\)](#) that is based on a limited number of measurements dating from 1910 (only two salinities at 5 temperatures). In the original equation the pressure is present, but at low pressures the effect on density can be neglected.

Eckart formulation

The Eckart formulation is given by ([Eckart, 1958](#)):

Range: $0 < T < 40^\circ\text{C}$, $0 < s < 40 \text{ ppt}$

$$\rho = \frac{1000P_0}{\lambda + \alpha_0 P_0}, \quad (8.53)$$

where:

$$\lambda = 1779.5 + 11.25T - 0.0745T^2 - (3.80 + 0.01T)s, \quad (8.54)$$

$$\alpha_0 = 0.6980, \quad (8.55)$$

$$P_0 = 5890 + 38T - 0.375T^2 + 3s. \quad (8.56)$$

with the salinity s in [ppt] and the water temperature T in [$^\circ\text{C}$].

The keyword that selects the equation of state in the mdu file is called Idensform. The influence of salinity and or temperature on water motion through the baroclinic pressure can be switched off by setting Idensform=0

UNESCO formulation

The UNESCO formulation is given by ([UNESCO, 1981a](#)):

Range: $0 < T < 40^\circ\text{C}$, $0.5 < s < 43 \text{ ppt}$

$$\rho = \rho_0 + As + Bs^{3/2} + Cs^2 \quad (8.57)$$

where

$$\begin{aligned} \rho_0 = & 999.842\,594 + 6.793\,952 \times 10^{-2}T - 9.095\,290 \times 10^{-3}T^2 + \\ & + 1.001\,685 \times 10^{-4}T^3 - 1.120\,083 \times 10^{-6}T^4 + 6.536\,332 \times 10^{-9}T^5 \end{aligned} \quad (8.58)$$

$$\begin{aligned} A = & 8.244\,93 \times 10^{-1} - 4.089\,9 \times 10^{-3}T + 7.643\,8 \times 10^{-5}T^2 + \\ & - 8.246\,7 \times 10^{-7}T^3 + 5.387\,5 \times 10^{-9}T^4 \end{aligned} \quad (8.59)$$

$$B = -5.724\,66 \times 10^{-3} + 1.022\,7 \times 10^{-4}T - 1.654\,6 \times 10^{-6}T^2 \quad (8.60)$$

$$C = 4.831\,4 \times 10^{-4} \quad (8.61)$$

with the salinity s in [ppt] and the water temperature T in [$^{\circ}$ C].



Note: The UNESCO formulation is set as default.



Remarks:

- ◊ Equation (8.59) is known as the International Equation of State for Seawater (EOS80) and is based on 467 data points. The standard error of the equation is 3.6×10^{-3} kg/m³ (Millero and Poisson, 1981).
- ◊ The Practical Salinity Scale (UNESCO, 1981b, Par. 3.2) and the International Equation of State are meant for use in all oceanic waters. However, these equations should be used with caution in waters that have a chemical composition different from standard seawater. In such waters, densities derived with the methods based on practical salinity measurements and the International Equation of State may deviate measurably from the true densities. However, in water masses different in composition from standard seawater the *differences* in densities derived by the new equations involve only very small errors.

Recommendation

The UNESCO formulae serve as an international standard. Further, the UNESCO formulae show the correct temperature of 4 degrees Celsius where fresh water has its maximum density. The latter is of importance for thermal stratification in deep lakes in moderate climate zones. Therefore we recommend the application of the UNESCO formulae and to select the Eckart formulation only for consistence with previous projects in which it has been used. At this moment the default is UNESCO.

Nevertheless, the UNESCO formulae have their limitations as they are based on the general properties of seawater mixed with fresh water. Particularly in cases where marginal density differences play a role, typically lakes, a variable mineral content of the water may create density differences not detected by just temperature and salinity. For such dedicated cases, you are warranted to check the accuracy of the UNESCO formulae against experimental density-relations derived from the (lake) water. In case of deviations or other constituents determining the water density, we then recommend to contact the Helpdesk for further assistance such as the implementation of a more dedicated density formulation.

8.10 Tide generating forces

Numerical models of tidal motion in coastal seas generally do not account for the direct local influence of the tide generating forces. The amount of water mass in these models is relatively small and the effect of these forces on the flow can be neglected. In that case, tidal motion can often be reproduced with sufficient accuracy just by prescribing the tidal forcing along open model boundaries.

In models covering larger seas or oceans, the contribution of the gravitational forces on the water motion increases considerably and can no longer be neglected. In a global model, open boundaries are absent and the tidal motion can only be induced by including tide generating forces. Because tidal forces only play a significant role in the larger models, and because the exact position of each computational point on earth must be known, we have implemented these forces only in combination with spherical coordinates. By default, tide generating forces is switched on in spherical models. You can switch it off by setting in the mdu file:
Tidalforcing = 0

The tide generating forces originate from the Newtonian gravitational forces of the terrestrial system (Sun, Moon and Earth) on the water mass. The equilibrium tide is the tide that would

result from the gravitational forces if a solid earth would be completely covered by an ocean of about 21.5 km deep, such that the wave propagation speed would match the speed of the celestial forces over the ocean surface. The interacting frequencies that result can be grouped as diurnal, semi diurnal and long periodic. The total number of tide generating frequencies that is evaluated in the computation is a tradeoff between required accuracy and computational cost. Per default, we use a set of 60 frequencies, similar to TRIWAQ.

A smaller set of 11 main frequencies as used in Delft3D-FLOW can also be selected by setting in the MDU file :

```
Doodsonstart = 57.555
Doodsonstop  = 275.555
Doodsoneps   = 0.030
```

The full set of 484 components can be selected by specifying:

```
Doodsonstart = 55.565
Doodsonstop  = 375.575
Doodsoneps   = 0.000
```

For the set of 60 components these numbers are:

```
Doodsonstart = 55.565
Doodsonstop  = 375.575
Doodsoneps   = 0.030
```

The earth itself is deformed by the celestial forces, this is called the solid earth tide. An estimate of this influence is based on the work of [Love \(1927\)](#), is included in the total tide generating potential.

We have gratefully applied the Fortran subroutines written by E.J.O. Schrama. Details on the implementation of tide generating forces can be found in his lecture notes [Schrama \(2007\)](#).

In order to save computation cost, the tidal potential is not computed for each computational point, but on a grid with a resolution of 1 degree in both the South-North and West-East direction.

The tidal and surge motions of the seas and ocean also deform the earth , which is called tidal loading. Next to that, the water at some point is attracted by all moving water elsewhere on the globe. This is called self attraction. The combined influence of these two processes is implemented in a beta version.

8.11 Advection at open boundaries

In hydrodynamic models, one often applies high resolution grids in limited areas, for instance to investigate the influence of a dam design on local flow patterns in a coastal area. Then, the challenge is to prescribe the overall hydrodynamics in a detailed model. Tidal motion is important and therefore water levels have to be prescribed. If an overall model with good reproduction of water levels and velocities is available, then it is common practice to start with water levels from the overall model on the open alongshore boundary and on both cross-shore boundaries of the detail model. Then, often the reproduction of water levels is accurate, but the reproduction of flow velocities might be less accurate. In particular, the residual flow velocities might be inaccurate. Obtaining a good velocity reproduction by prescribing only water levels often only works for the larger models (e.g. > 30 km) so that bed friction, pressure gradient and acceleration can balance each other. In smaller models, however, this might not be the case. Even small water level differences on the open boundaries may introduce unrealistic high or low velocities. A remedy might be to apply velocity boundaries or other types of boundary conditions (Neumann-type, ...) on the cross-shore boundaries. However, this often reduces reproduction quality of water levels or might introduce strange flow patterns. Also the existence of stratification may also lead to unrealistic currents near open boundaries.

Therefore, in D-Flow FM the keyword `Zerozbndinflowadvection` can be activated in order to suppress unphysical currents near open boundaries. This applies for all water level open boundaries. This keyword that has to be specified in [numerics] and the options are `Zerozbndinflowadvection=0,1` or `2`. A value of `0` results into the default Neumann approach for currents, yielding that the gradient in velocity at the open boundary is zero. This approach is the default and applied when keyword `Zerozbndinflowadvection` is not specified. Applying a value of `1` forces cell center velocities to zero on the open boundary at inflow for all water level boundaries. A value of `2` forces cell center velocities to zero on the open boundary for both inflow and outflow for all water level boundaries.

9 Transport of matter

9.1 Introduction

In D-Flow FM, transport is formulated as

$$\frac{d}{dt} \int_{V(t)} c dV + \int_{\partial V(t)} c(\mathbf{u} - \mathbf{v}) \cdot \mathbf{n} dS = \int_{\partial V(t)} (K \nabla c) \cdot \mathbf{n} dS + \int_{V(t)} s dV, \quad (9.1)$$

where $V(t)$ is a three-dimensional control volume, c is a concentration, \mathbf{u} the flow velocity field, \mathbf{v} the velocity of the (vertically) moving control volume, K is a diagonal matrix $K = \text{diag}(\kappa, \kappa, \kappa_z)$ with diffusion coefficients and s a source term. For two-dimensional (depth-averaged) flow, we obtain

$$\frac{\partial hc}{\partial t} + \nabla \cdot (h \mathbf{u} c) = \nabla \cdot (h \kappa \nabla c) + hs, \quad (9.2)$$

where h is the water depth. In case of three-dimensional (layer-averaged) flow, with Δz a layer thickness from $z_0(x, y, t)$ to $z_1(x, y, t)$, we obtain

$$\begin{aligned} \frac{\partial \Delta z c}{\partial t} + \nabla \cdot (\Delta z \mathbf{u} c) + [\omega_{z_1} c]_{z=z_1} - [\omega_{z_0} c]_{z=z_0} &= \nabla \cdot (\Delta z \kappa \nabla c) + \\ \left[\kappa_z \frac{\partial c}{\partial z} - \kappa \nabla z_1 \cdot \nabla c \right]_{z=z_1} - \left[\kappa_z \frac{\partial c}{\partial z} - \kappa \nabla z_0 \cdot \nabla c \right]_{z=z_0} &+ \Delta z s, \end{aligned} \quad (9.3)$$

where by \mathbf{u} and ∇ still the horizontal components are meant, i.e. $\mathbf{u} = (u, v)^T$ and $\nabla = \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial y} \right)^T$ and κ_z is the vertical diffusion coefficient. Furthermore ω_{z_0} and ω_{z_1} are the velocity component normal and relative to the moving $z = z_0$ and $z = z_1$ layer interfaces respectively. Note that taking $c = 1$ yields

$$\omega_{z_1} + \frac{\partial z_1}{\partial t} = \omega_{z_0} - \nabla \cdot (\Delta z \mathbf{u}) + \frac{\partial z_0}{\partial t}, \quad (9.4)$$

which, combined with a zero-flux condition at the bed, recursively defines ω_{z_0} and ω_{z_1} for all layers.

We apply Equation (9.2) and Equation (9.3) to transport of

- ◊ salinity,
- ◊ temperature,
- ◊ suspended sediment,
- ◊ tracers,
- ◊ other, intentionally not mentioned,

and ultimately to the water itself (the continuity equation) to obtain the relative layer interface velocities as expressed by Equation (9.4).

In the following we will highlight various physical and numerical settings in D-Flow FM. First and foremost, it is important to understand that there are *two* numerical implementations available, that differ slightly. These are selected with the keyword `transportmethod` in the mdu-file:

```
transportmethod = 0
```

or

```
transportmethod = 1           # default.
```

The difference mainly concerns the treatment of vertical advection as explained in the next section. Note that tracers are only available with `transportmethod '1'`.

9.2 Some words about suspended sediment transport

In D-Flow FM *two* sediment models are available.

The first is a genuine D-Flow FM implementation, while the second is adopted from Delft3D-SED and is only available in combination with `transportmethod '1'`.

For a description of the latter, see [D-Morphology UM \(2019\)](#).

We will not discuss the specifics of either of the two sediment models further and confine ourselves to mentioning the relevant settings that concern suspended sediment transport.

9.3 Transport processes

Looking at the equations that govern transport of matter, we can identify three processes, namely advection, diffusion and sources/sinks. The next sections will describe the relevant user settings.

9.3.1 Advection

Advection of matter is expressed by the term $\nabla \cdot (huc)$ in [Equation \(9.2\)](#) in the case of two-dimensional modelling.

In three dimensions, we make a distinction between horizontal advection $\nabla \cdot (\Delta z \mathbf{u}c)$ and vertical advection $[\omega_{z_1} c]_{z=z_1} - [\omega_{z_0} c]_{z=z_0}$ in [Equation \(9.3\)](#). A higher-order numerical approximation of these terms can be obtained by setting their "limiter type" to an appropriate value. Setting the limiter type to "0" reduces the numerical approximation to a low-order upwind method (i.e. severe limiting).

Although not restricted to the advection of salinity only, the choices for *all* matter are set with keyword `limtypsa` in the mdu-file, for example

```
[numerics]
limtypsa = 0           # first-order upwind, or
```

or

```
[numerics]
limtypsa = 4           # MC limiter
```

For `transportmethod=1`, only the monotonized central (MC) limiter is available and used when `limtypsa>0`, for example 1 or 2 or 6, but not -1. For `transportmethod=0` other limiters are available as well and the MC limiter is selected by setting `limtypsa` to '4', which is its default value. We will not discuss the non-default limiters.

In three-dimensional modelling horizontal advection is similarly as in two dimension, but now vertical advection can be selected with the keyword `Vertadvtypsal` in the mdu-file, i.e.

```
[numerics]
Vertadvtypsal = 0           # no vertical advection
```

or

```
[numerics]
Vertadvtypsal = 5           # default
```

The various combinations of `transportmethod` and `Vertadvtypsal` have the following meaning:

Vertadvtypsal	transport method		1 time
	0 space	0 time	
0	none	none	none
1	1 st order upwind	forward Euler	salt and temperature: central, other: higher-order limited upwind
2	central	forward Euler	
3	1 st order upwind	θ -method	salt and temperature: θ -method, other: forward Euler
4	central	θ -method	
5	neg. stratification or water depth < chkadvd: 1 st order upwind, otherwise: central	θ -method	temperature: central, other: higher-order limited upwind
6	as 5	as 5	

For the limited higher-order upwind discretization the MC limiter is used if `transportmethod=1` if `limtypsa>0`, regardless of its specific value, and 1st-order upwind if `limtypsa=0`.

9.3.2 Diffusion

Diffusion of matter is expressed by the term $\nabla \cdot (h\kappa \nabla c)$ for two-dimensional modelling and a similar and additional terms in three dimensions, see [Equation \(9.3\)](#).

Clearly, we have horizontal diffusivity κ and vertical diffusivity κ_z . The horizontal diffusivity κ is a summation of

- ◊ the molecular diffusivity κ_l , only for `transportmethod=1`. We use the following values:

$$\kappa_l = \begin{cases} \frac{1}{700}\nu_l, & \text{salt,} \\ \frac{1}{6.7}\nu_l, & \text{temperature,} \\ 0\nu_l, & \text{sediment,} \\ 0\nu_l, & \text{tracers,} \end{cases} \quad (9.5)$$

where $\nu_l = 10^{-6} \text{ m}^2/\text{s}$ is the kinematic viscosity,

- ◊ a background diffusivity, user-specified as

- spatially varying values by `horizontaleddydiffusivitycoefficient` in the ext-file, or

- a uniform value `Dicouuv` in the mdu-file,
- in that order of precedence, and
- ◊ a contribution from turbulent transport expressed as ν_t/σ_t , where ν_t is the eddy viscosity coefficient and σ_t is the turbulent Prandtl-Schmidt number for which the following values are used:

$$\sigma_t = \begin{cases} 0.7, & \text{salt,} \\ 0.7, & \text{temperature,} \\ 1.0, & \text{sediment,} \\ 1.0, & \text{tracers.} \end{cases} \quad (9.6)$$

Horizontal diffusion is turned off when `Dicouuv=0`.

The user has to be aware of the following. The explicit nature of the horizontal diffusion terms in the time-integration method imposes a condition on the time step for numerical stability. Recall that we have a similar condition due to explicit horizontal advection. Although we always decrease the time step to satisfy the advection-related time step criterion, we do not do so for diffusion. Instead, diffusion is limited such that our time step is restricted by advection only, or by a user-specified maximum time step if it is smaller. For further details, consult D-Flow FM TRM (2015).

Be aware that the modelled diffusion may be smaller than anticipated. However, the actual effective diffusion encountered may be (much) larger than anticipated due to numerical diffusion of the advection scheme.

Not considering suspended sediment, and similar to the horizontal diffusivity, the vertical diffusivity κ_z is a summation of

- ◊ the molecular diffusivity κ_l (for both transport methods), see [Equation \(9.5\)](#), not added for transport method '1' if `Dicoww=0`,
- ◊ a background vertical diffusivity, user-specified as a uniform value `Dicoww` in the mdu-file, and
- ◊ a contribution from turbulent transport, similar to its horizontal counterpart, but with the horizontal viscosity coefficient now replaced by the vertical (eddy) viscosity coefficient.

Time integration is performed with a method that does not impose an additional constraint on the time step. Unlike horizontal diffusion which is limited to ensure numerical stability while maintaining the time-step size, vertical diffusion is *not* limited.

Vertical diffusion is turned off when, again not considering suspended sediment:

- ◊ for transport method '0': `Dicoww=0` or `Vertadvtypsal=1` or `Vertadvtypsal=2`,
- ◊ for both transport methods: the water depth is smaller than a threshold 10^{-2} m.

For the vertical diffusivity of suspended sediment, see [D-Morphology UM \(2019\)](#).

9.3.3 Sources and sinks

Sources and sinks may be provided by an entry in the ext-file as follows:

```
quantity=discharge_salinity_temperature_sorsin
```

This simultaneously prescribes sources and sinks of

- ◊ water volume itself (i.e. discharge),
- ◊ salinity, and
- ◊ temperature, and.
- ◊ any other constituents that are transported.

See [section 8.8](#) for more details.

9.3.4 Forester filter

The central vertical advection schemes of transport method '0' may cause nonphysical oscillations, or wiggles, especially near regions of large gradients. The user has the option to suppress salt and temperature wiggles with a filter inspired by the so-called Forester filter. This filter also penalizes physically unstable stratification. The maximum number of iterations in the filter is controlled with keywords `Maxitverticalforestersal` for salt (default 100) and `Maxitverticalforestertem` for temperature (default 0, i.e. no filtering). Note that the filter is unavailable when using transport method '1'.

9.4 Transport boundary and initial conditions

The equations that govern transport of matter are complemented with boundary and initial conditions. We make the distinction between "horizontal" boundaries, that are either "open" or "closed", and vertical boundaries, only relevant for three-dimensional modelling.

9.4.1 Open boundary conditions

At "horizontal" open boundaries the following boundary conditions are applied:

- ◊ salt, temperature and tracers:
 - inflow: user-specified Dirichlet condition,
 - outflow: homogeneous Neumann condition,
- ◊ suspended sediment: see [D-Morphology UM \(2019\)](#).

The user-specified Dirichlet conditions are supplied in the usual manner through the ext-file, i.e.

```
quantity=salinitybnd
```

for salt and

```
quantity=temperaturebnd
```

for temperature. Boundaries conditions for multiple tracers may be defined by appending their name to the `tracerbnd` keyword, for example

```
quantity=tracerbndMY_FIRST_TRACER
```

and

```
quantity=tracerbndMY_SECOND_TRACER
```

9.4.2 Closed boundary conditions

At "horizontal" closed boundaries zero-flux conditions are applied.

9.4.3 Vertical boundary conditions

At the "vertical" boundaries, i.e. at the bed and at the water surface, zero-flux conditions are applied, except for temperature that is, see [chapter 11](#) for further details on that matter.

9.4.4 Thatcher-Harleman boundary conditions

Consider (a part of) an open boundary where the flow reverts from outflowing to inflowing. According to [section 9.4.1](#), at that very moment a Dirichlet condition becomes effective and a user-specified boundary value is prescribed. This value does in general not reflect the true condition at the boundary and causes a discontinuous temporal behaviour. The so-called Thatcher-Harleman boundary condition is intended to regularize this behaviour. The actual value applied at the boundary is smoothly transformed from the last value under outflow conditions to the user-specified value under inflow conditions within a user-specified *return time*, which has to be specified in seconds. This return time is prescribed with the keyword `return_time` in the "new style" external forcings file for boundary condition, for example

```
[boundary]
quantity      = salinitybnd
locationfile  = tfl_01.pli
forcingfile   = tfl.bc
return_time   = 250
```

See [section C.5](#) for more details on this format. Note that the Thatcher-Harleman boundary conditions are only available currently for two-dimensional modelling.

9.4.5 Initial conditions

We will only consider initial conditions for salinity, temperature and tracers. For initial sediment concentrations, see [D-Morphology UM \(2019\)](#).

Initial conditions are specified in three possible ways, namely

- ◊ a horizontally spatially varying field in the usual way through the ext-file,
- ◊ a vertical profile in three dimensions, horizontally uniformly distributed, for salinity and temperature only in the ext-file, and
- ◊ uniform values for salinity and temperature in the mdu-file.

In the ext-file we have

```
quantity      = initialsalinity
```

for the initial salinity, and

```
quantity      = initialsalinitytop
```

for the initial salinity in the top layer in case of three-dimensional modelling. When specified, the initial salinity field is linearly distributed from the "initialsalinity" in the lowest layer to the

"initialsalinitytop" in the top layer. When *not* specified, the initial salinity field is vertically uniformly distributed.

The initial temperature field is prescribed with

```
quantity      = initialtemperature
```

which in three dimensions is vertically uniformly distributed.

A horizontally uniformly distributed vertical profile of salinity and temperature can be prescribed with `initialverticalsalinityprofile` and `initialverticaltemperatureprofile` respectively, for example for salinity

```
QUANTITY=initialverticalsalinityprofile
FILENAME=inisal.pol
FILETYPE=10
METHOD=4
OPERAND=0
```

The polygon file contains (z , salinity) value pairs, where z is the vertical coordinate in meters. For example for linearly varying salinity from 30 to 20 ppt from -10 to 0 m:

```
L1
2 2
-10 30
0 20
```

The initial field of an arbitrary number of tracers are prescribed in the same way, except for the user-specified tracername that is added to the keyword `initialtracer`, similar to the tracer boundary conditions, for example

```
quantity=initialtracerMY_FIRST_TRACER
```

and

```
quantity=initialtracerMY_SECOND_TRACER
```

Default values for salinity and temperature are defined in the mdu-file with `Initialsalinity` and `Initialtemperature` respectively.

10 3D Modelling

Note: 3D modelling is a β -functionality.



Introduction

In many rivers the flow is well mixed and can be described efficiently in 2D using the logarithmic flow profile. In oceans, seas, estuaries and lakes, stratified flows may occur and the flow profile may not be logarithmic anymore. Then, 3D modelling is required. Vertical gradients in velocity, salinity and temperature have to be computed accurately. Especially in stratified systems this might be a challenge. High gradients may occur around pycnoclines, and near the surface and near the bed. Fortunately, there is no need to resolve the bed boundary layer because it is fully developed in most cases. The law of the wall is then well applicable.

The number of layers depends per application. From experience with Delft3D, which is based on structured grids, for many decades, salinity intrusion in estuaries has been successfully computed with a relatively few number of layers. Quite often ten layers have been applied. Nowadays, twenty layers seem to be the standard in 3D modelling.

σ and z-co-ordinate modelling

In the vertical, two approaches are widely used, the so-called σ -co-ordinate (terrain-following) and z-co-ordinate (geopotential) models. A σ -co-ordinate model (Phillips, 1957) has a fixed number of layers everywhere and the layer interfaces move in time with the time varying water level, while a z-co-ordinate model uses layer interfaces at fixed vertical positions. Consequently, the number of layers at a location varies with its water depth, see the figure below.

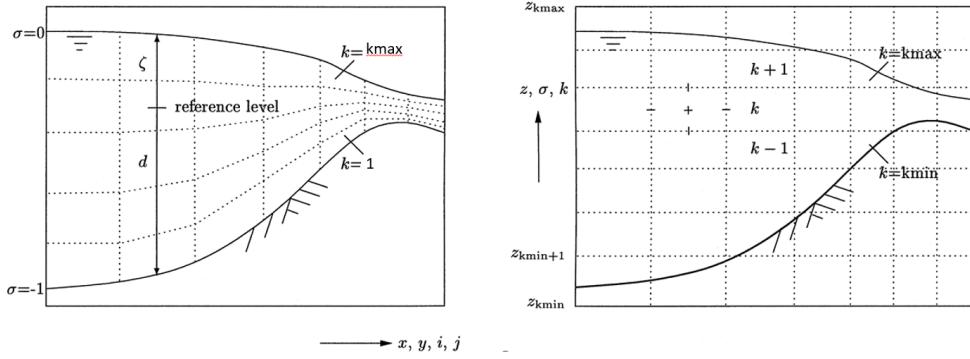


Figure 10.1: Vertical grid concepts: the σ model (left) and z-coordinate model (right)

In D-Flow FM both layer approaches have been implemented. To switch to 3D simulations the keyword `Kmx=N` has to be set with `N` the number of layers in the vertical. When `Kmx` equals 0, then 2D implementation is applied. When `Kmx` equals 1, then a 3D computation with one layer is applied, which is very similar to 2D modelling. The only difference is that the specified 2D bed friction coefficient is first translated to a `z0` value that is used in the computation of `ustar` at the lowest layer of a 3D model using the logarithmic profile assumption.

Via keyword `LayerType` one can choose between σ -co-ordinate (`LayerType=1`) or z-coordinates (`LayerType=2`). In case of a z-co-ordinate model we have the following additional keywords:

`ZlayTop =`

```
ZlayBot =
```

to specify the maximum and the minimum grid layer interface, respectively. Via keyword StretchType non-uniform layers can be specified:

```
1=user defined  
2=exponential  
otherwise=uniform
```

Via keyword StretchCoef the layer thicknesses (in percentages) are prescribed. An example for option 1 might be:

```
StretchCoef=4.0, 5.9, 8.7, 12.7, 18.7, 18.7, 12.7, 8.7, 5.9, 4.0
```

and for option 2:

```
StretchCoef=stretching level, and two coefficients for layers growth
```

Z-layer specification for temperature computations

For temperature computations, the part of the water column just below the lowest thermocline up to the free surface is of interest. Not knowing its position in advance, this top part of the water column can best be resolved by a uniform resolution in the vertical via keyword Dztop. The vertical grid is uniform above a level called Dztopuniabovez and extends to the top layer level called Floorlevtplay. This level is called a floor level because only the floor level of the top layer can be specified. (The ceiling of the top layer is the water level). Below the level Dztopuniabovez the layer size may grow in downward direction. To have layers growing in thickness downward by a factor 1.1, set keyword SigmaGrowthFactor=1.1. The total number of layers is recounted based upon the deepest cell in the model.

10.1 Spurious oscillations in vertical velocity

In D-Flow FM unphysical oscillations can occur in 3D models. These oscillations are primarily manifest as a “checkerboard”-like pattern in the vertical velocities. For some model applications this might lead to unrealistic model results. A high-order horizontal momentum filter has been developed to suppress these spurious oscillations.

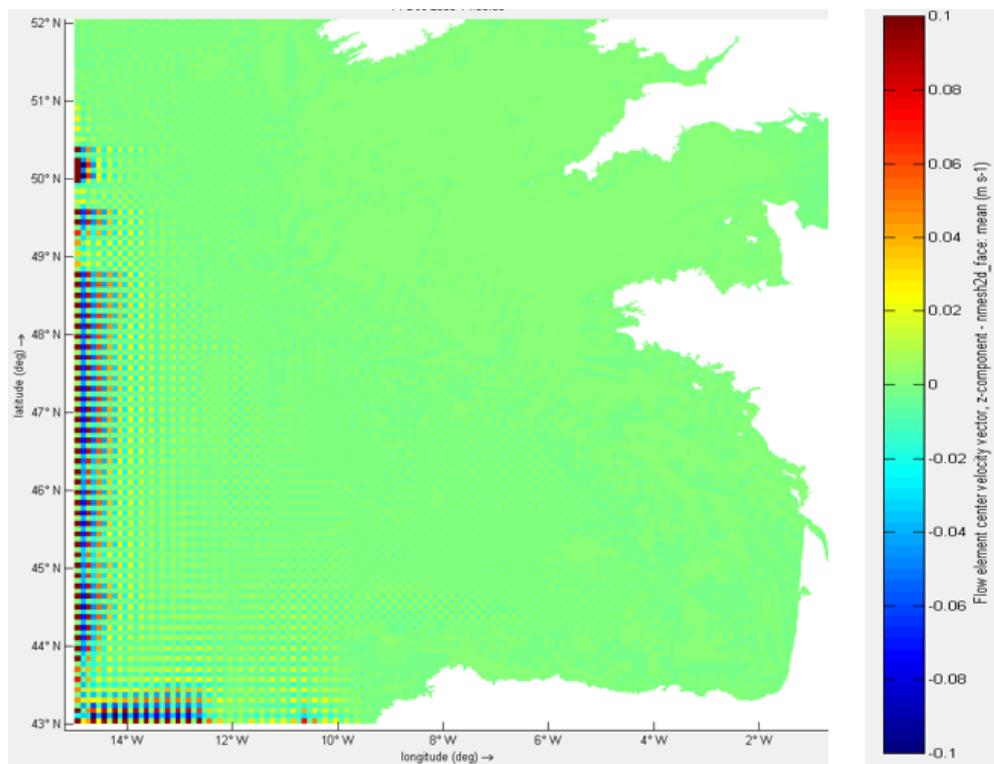


Figure 10.2: Illustration of spurious oscillations near open boundaries

Figure Figure 10.2 illustrates these patterns in the vertical velocities near the southwestern open boundary of the DCSM model without the application of the filter. These patterns disappear with the application of the filter.

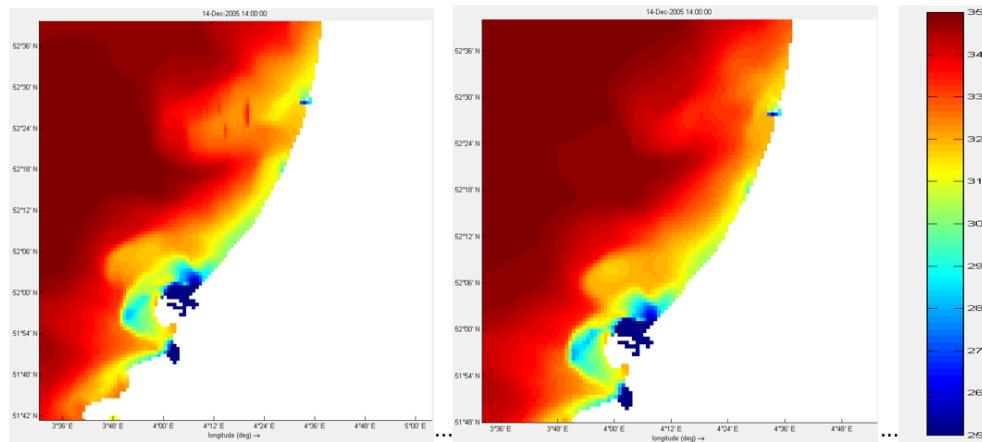


Figure 10.3: Illustration of unphysical oscillations

In Figure Figure 10.3 the fresh water Rhine plume in the model is shown. Unphysical patterns are observed in the northeastern part of the model domain, quite near the fresh water plume (see blue area). When the filter is applied, the unphysical oscillations are no longer visible. This filter can be switched on via keyword `HorizontalMomentumFilter=1`. This filter is only available for σ co-ordinates yet. In near future this filter will also become available for z-co-ordinate modelling.

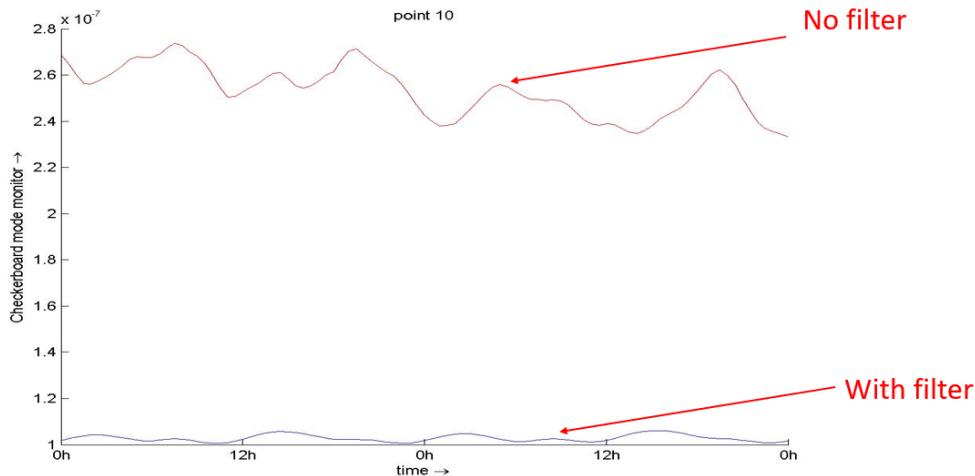


Figure 10.4: Illustration of monitor for occurrence of spurious oscillations

A monitor function is available to check whether spurious oscillations occur in a model application. This filter can be switched on via keyword `CheckerboardMonitor=1`. This is illustrated in Figure 10.4.

10.2 Anti creep

In the σ co-ordinate system, the grid co-ordinate lines intersect the mostly horizontal density interfaces. A well-known disadvantage of σ co-ordinates with standard numerical schemes is that in areas of steep bed topography unphysical vertical mixing may occur (Haney, 1991). This is a result of erroneous computation of horizontal gradients when vertical gradients exist. To cure this, the anti-creep method by Stelling and Van Kester (1994), first implemented in Delft3D, has also been implemented in D-Flow Flexible Mesh. This can be switched on via keyword `AntiCreep = 1`.

In z-coordinates, the gridlines are more parallel to most isopycnals and the anticreep scheme is not required. This is an advantage because the anticreep scheme involves a Minmod limiter that may annihilate the baroclinic pressure term completely for steep gridcells in combination with thin layers. Ranking the vertical schemes in their capability of computing salinity intrusion against a freshwater discharge, we obtain best results using z-coordinates, then sigma anticreep then just sigma. Still, in most models, computed salinity intrusion length is less than the observed one.

10.3 Turbulence modelling

D-Flow FM solves the shallow water equations for incompressible flow. Usually the grid (horizontal and/or vertical) is too coarse and the time step too large to resolve the turbulent scales of motion. The turbulent processes are “sub-grid”. The primitive variables are space- and time-averaged quantities. Filtering the equations leads to the need for appropriate closure assumptions.

For 3D shallow water flow the stress and diffusion tensor are anisotropic. The horizontal eddy viscosity coefficient ν_H and eddy diffusivity coefficient D_H are much larger than the vertical coefficients ν_V and D_V , i.e. $\nu_H \gg \nu_V$ and $D_H \gg D_V$. The horizontal coefficients are assumed to be a superposition of three parts:

-
- 1 a part due to molecular viscosity.
 - 2 a part due to “2D-turbulence”,
 - 3 a part due to “3D-turbulence” see [Uittenbogaard et al. \(1992\)](#) and

The 2D part is associated with the contribution of horizontal motions and forcings that cannot be resolved (“sub-grid scale turbulence”) by the horizontal grid (Reynolds averaged or eddy resolving computations). The 3D part is referred to as the three-dimensional turbulence and is computed following one of the turbulence closure models, described in this section. For 2D depth-averaged simulations, the horizontal eddy viscosity and eddy diffusivity coefficient should also contain a contribution due to the vertical variation of the horizontal flow (Taylor shear dispersion).

The background horizontal viscosity coefficient ν_H^{back} and eddy diffusivity coefficient D_H^{back} (constant or space-varying) can be specified in the Delta Shell GUI and D-Flow FM’s MDU file. In D-Flow FM a sub-grid scale model, HLES for 2D-turbulence is implemented. (see D-Flow FM User Manual). In D-Flow FM we have not yet achieved this, but we implemented a simple horizontal model, the so called Smagorinsky model, so that we can at least cope with possibly very large grid size variations.

The horizontal eddy coefficients are typically an order of magnitude larger than the vertical coefficients determined by the turbulence closure model.

In D-Flow FM, two two-equation turbulence closure models have been implemented to determine the vertical eddy viscosity (ν_V) and vertical eddy diffusivity (D_V):

- 1 $k-\varepsilon$ turbulence closure model.
- 2 $k-\tau$ turbulence closure model (in β -version).

The $k-\tau$ model is a mathematical variant of the $k-\varepsilon$ model, [Dijkstra \(2014\)](#). Both models solve equations for production, dissipation and transport of turbulent kinetic energy k . In the $k-\varepsilon$ model the dissipation rate of turbulent kinetic energy ε , is modeled, whereas in the $k-\tau$ model the dissipation timescale τ is modeled.

Both models are based on the so-called eddy viscosity concept of [Kolmogorov \(1942\)](#) and [Prandtl \(1945\)](#).

A brief description of each of these turbulence closure model will be given further on in this section, for more details we refer to [Uittenbogaard et al. \(1992\)](#). The $k-\varepsilon$ model has been used in tidal simulations by [Baumert and Radach \(1992\)](#) and [Davies and Gerritsen \(1994\)](#), for stratified flow of a river plume by [Postma et al. \(1999\)](#) and for the evolution of a thermocline by [Burchard and Baumert \(1995\)](#).

For strongly stratified flows it is important to introduce suitably chosen constant ambient (background) mixing coefficients, because the mixing coefficients computed by turbulence models with shear production only, reduce to zero. In the latter situation the vertical layers are completely de-coupled (frictionless). Disturbances are hardly damped and also the erosion of the vertical stratification is reduced to molecular diffusion.

Based on our experience with highly stratified flows we suggest applying an ambient or background vertical eddy viscosity in the order of $10^{-4} \text{ m}^2/\text{s}$ for the vertical exchange of momentum. This value corresponds with field measurements in the Rotterdam Waterway, The Netherlands.

Eddy diffusivity

The vertical eddy diffusivity is a scaled form of the eddy viscosity according to:

$$D_{3D} = \frac{\nu_{3D}}{\sigma_c}. \quad (10.1)$$

Parameter σ_c is the Prandtl-Schmidt number. Its numerical value depends on the substance c .

In Delft3D-FLOW the following settings of σ_c are used:

- ◊ In all cases, regardless the turbulence closure model, $\sigma_c = 0.7$ for the transport of heat, salinity, and tracer concentrations. For suspended sediment concentrations in online sediment transport computations, $\sigma_c = 1.0$.
- ◊ For the transport of turbulent kinetic energy k in the $k-L$ model and $k-\varepsilon$ model $\sigma_c = 1.0$, and for the transport of turbulent kinetic energy dissipation ε in the $k-\varepsilon$ model $\sigma_c = 1.3$.

In the mathematical formulation, the fluxes are instantaneously influenced by changes in the vertical gradients of velocity and density. A physical adjustment time of the turbulence to the variations of the vertical gradients, is not taken into account. The fluxes are not a monotone function of the gradients. For the transport equation of heat, for small temperature gradients the heat flux increases when the temperature gradient increases but for large temperature gradients the heat flux decreases because the vertical eddy diffusivity is damped. For large values of the density gradients and small values of the velocity gradients, the vertical diffusion equation becomes mathematically ill-posed [Barenblatt et al. \(1993\)](#), and the computed vertical profiles may become discontinuous (stepwise). The number of “steps” is dependent on the vertical grid.

The numerical scheme for the vertical advection of heat and salt (central differences) may introduce small vertical oscillations. This computational noise may enhance the turbulent mixing. D-Flow FM has a vertical filtering technique to remove this noise and to reduce the undesirable mixing. For more details, see [section 9.3.4](#).

In strongly-stratified flows, the turbulent eddy viscosity at the interface reduces to zero and the vertical mixing reduces to molecular diffusion. To account for the vertical mixing induced by shearing and breaking of short and random internal gravity waves, we suggest to apply an ambient eddy diffusivity in the order of 10^{-4} to 10^{-5} m²/s dependent on the Prandtl-Schmidt number. In Delft3D-FLOW for stable stratified flows, the minimal eddy diffusivity may be based on the Ozmidov length scale L_{oz} , specified by you and the Brunt-Väisälä frequency of internal waves:

$$D_V = \max \left(D_{3D}, 0.2L_{oz}^2 \sqrt{-\frac{g}{\rho} \frac{\partial \rho}{\partial z}} \right). \quad (10.2)$$

This feature is still to be implemented in D-Flow FM

For a detailed description of the turbulence closure models of Delft3D-FLOW we refer to [Rodijns et al. \(1984\)](#) and [Uittenbogaard et al. \(1992\)](#).

10.3.1 $k-\varepsilon$ turbulence model

In the $k-\varepsilon$ turbulence model, transport equations must be solved for both the turbulent kinetic energy k and for the energy dissipation ε . The mixing length L is then determined from ε and k according to:

$$L = c_D \frac{k\sqrt{k}}{\varepsilon}. \quad (10.3)$$

In the transport equations, the following two assumptions are made:

- ◊ The production, buoyancy, and dissipation terms are the dominating terms.
- ◊ The horizontal length scales are larger than the vertical ones (shallow water, boundary layer type of flows).

Because of the first assumption, the conservation of the turbulent quantities is less important and the transport equation is implemented in a non-conservation form.

The transport equations for k and ε are non-linearly coupled by means of their eddy diffusivity D_k , D_ε and the dissipation terms. The transport equations for k and ε are given by:

$$\begin{aligned} \frac{\partial k}{\partial t} + u \frac{\partial k}{\partial x} + v \frac{\partial k}{\partial y} + \frac{\omega}{\zeta - z_b} \frac{\partial k}{\partial \sigma} = \\ + \frac{1}{(\zeta - z_b)^2} \frac{\partial}{\partial \sigma} \left(D_k \frac{\partial k}{\partial \sigma} \right) + P_k + P_{kw} + B_k - \varepsilon, \end{aligned} \quad (10.4)$$

$$\begin{aligned} \frac{\partial \varepsilon}{\partial t} + u \frac{\partial \varepsilon}{\partial x} + v \frac{\partial \varepsilon}{\partial y} + \frac{\omega}{\zeta - z_b} \frac{\partial \varepsilon}{\partial \sigma} = \\ + \frac{1}{(\zeta - z_b)^2} \frac{\partial}{\partial \sigma} \left(D_\varepsilon \frac{\partial \varepsilon}{\partial \sigma} \right) + P_\varepsilon + P_{\varepsilon w} + B_\varepsilon - c_{2\varepsilon} \frac{\varepsilon^2}{k}. \end{aligned} \quad (10.5)$$

with

$$D_k = \frac{\nu_{mol}}{\sigma_{mol}} + \frac{\nu_{3D}}{\sigma_k} \quad \text{and} \quad D_\varepsilon = \frac{\nu_{3D}}{\sigma_\varepsilon} \quad (10.6)$$

In the production term P_k of turbulent kinetic energy, the horizontal gradients of the horizontal velocity and all the gradients of the vertical velocities are neglected. The production term is given by:

$$P_k = \nu_{3D} \frac{1}{(\zeta - z_b)^2} \left[\left(\frac{\partial u}{\partial \sigma} \right)^2 + \left(\frac{\partial v}{\partial \sigma} \right)^2 \right]. \quad (10.7)$$

For small-scale applications (e.g. simulation of laboratory flume), you can switch on a more extended production term P_k of turbulent kinetic energy (option “partial slip”, rough side wall) given by:

$$\begin{aligned} P_k = 2\nu_{3D} \left[\frac{1}{2(\zeta - z_b)^2} \left\{ \left(\frac{\partial u}{\partial \sigma} \right)^2 + \left(\frac{\partial v}{\partial \sigma} \right)^2 \right\} \right] + \\ + 2\nu_{3D} \left[\left(\frac{\partial u}{\partial x} \right)^2 + \frac{1}{2} \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 \right]. \end{aligned} \quad (10.8)$$

In this expression, ν_{3D} is the vertical eddy viscosity, prescribed by [Equation \(10.16\)](#). In [Equation \(10.7\)](#) and [Equation \(10.8\)](#) it has been assumed that the gradients of the vertical velocity w can be neglected with respect to the gradients of the horizontal velocity components u and v . The horizontal and vertical (σ -grid) curvature of the grid has also been neglected.

The turbulent energy production due to wave action is given by P_{kw} , but has not been implemented yet in D-Flow FM: Wave forcing in 3D models is being prepared for an upcoming release.

Near the closed walls the normal derivative of the tangential velocity is determined with the law of the wall:

$$\frac{\partial u}{\partial y} = \frac{u_*}{\kappa y}. \quad (10.9)$$

In stratified flows, turbulent kinetic energy is converted into potential energy. This is represented by a buoyancy flux B_k defined by:

$$B_k = \frac{\nu_{3D}}{\rho \sigma_\rho} \frac{g}{h} \frac{\partial \rho}{\partial \sigma} \quad (10.10)$$

with the Prandtl-Schmidt number $\sigma_\rho = 0.7$ for salinity and temperature and $\sigma_\rho = 1.0$ for suspended sediments.

The production term P_ε and the buoyancy flux B_ε are defined by:

$$P_\varepsilon = c_{1\varepsilon} \frac{\varepsilon}{k} P_k, \quad (10.11)$$

$$B_\varepsilon = c_{1\varepsilon} \frac{\varepsilon}{k} (1 - c_{3\varepsilon}) B_k, \quad (10.12)$$

with L prescribed by [Equation \(10.3\)](#) and the calibration constants by ([Rodi, 1984](#)):

$$c_{1\varepsilon} = 1.44, \quad (10.13)$$

$$c_{2\varepsilon} = 1.92, \quad (10.14)$$

$$c_{3\varepsilon} = \begin{cases} 0.0 & \text{unstable stratification} \\ 1.0 & \text{stable stratification} \end{cases} \quad (10.15)$$

In D-Flow FM in the ε -equation for stable stratification the buoyancy flux is switched off, so $c_{3\varepsilon} = 1.0$ and for unstable stratification the buoyancy flux is switched on $c_{3\varepsilon} = 0.0$.

The energy production and energy dissipation due to waves, the terms P_{kw} and $P_{\varepsilon w}$ in [Equation \(10.4\)](#) and [Equation \(10.5\)](#), have not been implemented yet in D-Flow FM: Wave forcing in 3D models is being prepared for an upcoming release.

The coefficients of the 3D k - ε turbulence closure model as implemented in D-Flow FM are not the same as in the depth-averaged k - ε turbulence closure model ([Rodi, 1984](#)), therefore for depth-averaged simulations, the k - ε turbulence closure model is not available for you.

The vertical eddy viscosity ν_{3D} is determined by:

$$\nu_{3D} = c'_\mu L \sqrt{k} = c_\mu \frac{k^2}{\varepsilon}, \quad (10.16)$$

with:

$$c_\mu = c_D c'_\mu. \quad (10.17)$$

To solve the transport equation, boundary conditions must be specified. A local equilibrium of production and dissipation of kinetic energy is assumed at the bed which leads to the following Dirichlet boundary condition:

$$k|_{\sigma=-1} = \frac{u_{*b}^2}{\sqrt{c_\mu}}. \quad (10.18)$$

The friction velocity u_{*b} at the bed is determined from the magnitude of the velocity in the grid point nearest to the bed, under the assumption of a logarithmic velocity profile. The bed roughness (roughness length) may be enhanced by the presence of wind generated short crested waves.

In case of wind forcing, a similar Dirichlet boundary condition is prescribed for the turbulent kinetic energy k at the free surface:

$$k|_{\sigma=0} = \frac{u_{*s}^2}{\sqrt{c_\mu}}. \quad (10.19)$$

In the absence of wind, the turbulent kinetic energy k at the surface is set to zero.

At open boundaries, the turbulent energy k is computed using the equation for k without horizontal advection. For a logarithmic velocity profile this will approximately lead to the following linear distribution based on the shear-stress at the bed and at the free surface:

$$k(z) = \frac{1}{\sqrt{c_\mu}} \left[u_{*b}^2 \left(1 - \frac{z - z_b}{\zeta - z_b} \right) + u_{*s}^2 \frac{z - z_b}{\zeta - z_b} \right]. \quad (10.20)$$

For ε the bed boundary condition reads:

$$\frac{\partial \varepsilon}{\partial z} = \frac{(\varepsilon_{b+1} - \varepsilon_b)}{\Delta z_b} = \frac{u_*^3}{\kappa \left(\frac{\Delta z_b}{2} + 9z_0 \right)^2} \quad (10.21)$$

The $k-\varepsilon$ turbulence model was successfully applied for the simulation of stratified flow in the Hong Kong waters (Postma *et al.*, 1999) and verified for the seasonal evolution of the thermocline (Burchard and Baumert, 1995).

10.3.2 $k-\tau$ turbulence model

The $k-\tau$ turbulence model is derived by Speziale *et al.* (1992) as a transformation of the ε -equation of the $k-\varepsilon$ turbulence model where the variable τ models a typical time-scale of turbulent eddies.

The $k-\tau$ turbulence model used in D-Flow FM deviates from the equation of Speziale *et al.* (1992) at a few points, to derive their $k-\tau$ model they used a different version of the $k-\varepsilon$ model and the most important difference is that they do not include buoyancy in their model.

The transport equations for k and τ used in D-Flow FM read (see for a derivation Dijkstra (2014)):

$$\frac{\partial k}{\partial t} + u \frac{\partial k}{\partial x} + v \frac{\partial k}{\partial y} + \frac{\omega}{\zeta - z_b} \frac{\partial k}{\partial \sigma} = \frac{1}{(\zeta - z_b)^2} \frac{\partial}{\partial \sigma} \left(D_k \frac{\partial k}{\partial \sigma} \right) + P_k + P_{kw} + B_k - k\tau, \quad (10.22)$$

$$\begin{aligned} \frac{\partial \tau}{\partial t} + u \frac{\partial \tau}{\partial x} + v \frac{\partial \tau}{\partial y} + \frac{\omega}{\zeta - z_b} \frac{\partial \tau}{\partial \sigma} = \\ \frac{1}{(\zeta - z_b)^2} \left\{ \frac{\partial}{\partial \sigma} \left(D_\tau \frac{\partial \tau}{\partial \sigma} \right) + \frac{2}{k} D_\tau \frac{\partial \tau}{\partial \sigma} \frac{\partial k}{\partial \sigma} - \frac{2}{\tau} D_\tau \frac{\partial \tau}{\partial \sigma} \frac{\partial \tau}{\partial \sigma} - \frac{\tau}{k} \frac{\partial}{\partial \sigma} \left(\left(\frac{1}{\sigma_\varepsilon} - \frac{1}{\sigma_k} \right) \right) \frac{\partial k}{\partial \sigma} \right\} \\ - \frac{\tau}{k} (c_{\varepsilon 1} - 1) P_k - \frac{\tau}{k} (c_{\varepsilon 3} - 1) B_k + c_{\varepsilon 2} - 1 \end{aligned} \quad (10.23)$$

with

$$D_k = \frac{\nu_{mol}}{\sigma_{mol}} + \frac{\nu_{3D}}{\sigma_k} \quad \text{and} \quad D_\tau = \frac{\nu_{3D}}{\sigma_\tau}. \quad (10.24)$$

11 Heat transport

This chapter is an almost integral copy of the Delft3D-FLOW manual. The difference is that in Delft3D-FLOW five heat flux models are implemented, whereas in D-Flow FM only two models are implemented. These are the most complete heat flux model, the so called Composite heat flux model (i.e. the Ocean heat flux model nr 5 in Delft3D-FLOW) and the most simple model, the Excess temperature model (model nr 3 in Delft3D-FLOW). In D-Flow FM, the parameter that sets the temperature model is called `Temperaturemodel` in the `mdu`-file. We kept the numbering of Delft3D-FLOW. When specifying `Temperaturemodel=1`, the temperature is taken into account in the transport solver and in the equation of state, but heat fluxes through the water surface are not taken into account. This may be useful when mixing is the primary factor that determines the temperature distribution.

The heat radiation emitted by the sun reaches the earth in the form of electromagnetic waves with wavelengths in the range of 0.15 to 4 μm . In the atmosphere the radiation undergoes scattering, reflection and absorption by air, cloud, dust and particles. On average neither the atmosphere nor the earth accumulates heat, which implies that the absorbed heat is emitted back again. The wavelengths of these emitted radiations are longer (between 4 and 50 μm) due to the lower prevailing temperature in the atmosphere and on Earth. Schematically the radiation process, along with the heat flux mechanisms at the water surface, is shown in Figure 11.1.

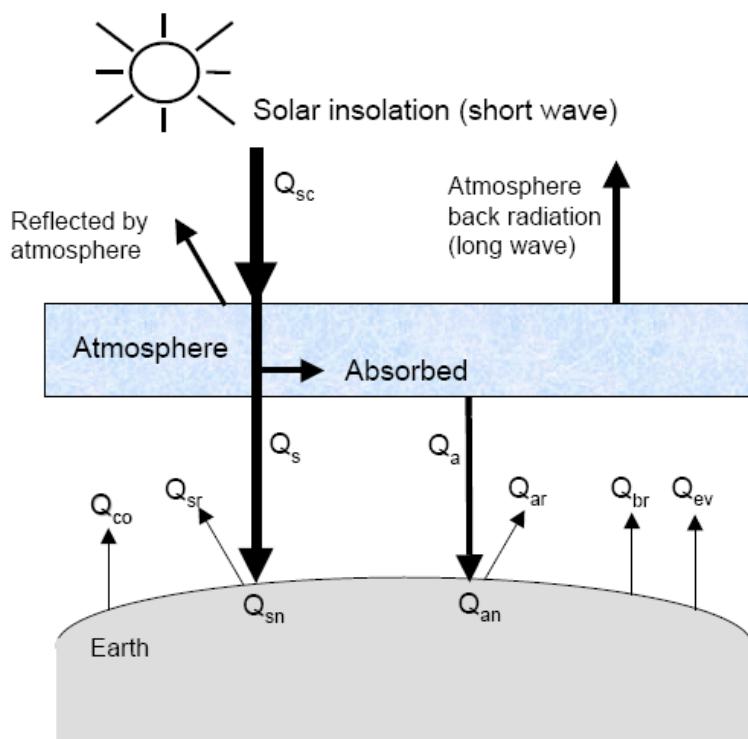


Figure 11.1: Overview of the heat exchange mechanisms at the surface

Legend for Figure 11.1:

Q_{sc}	radiation (flux) for clear sky condition in $[\text{J}/\text{m}^2\text{s}]$
Q_{co}	heat loss due to convection (sensible) in $[\text{J}/\text{m}^2\text{s}]$
Q_{sr}	reflected solar radiation in $[\text{J}/\text{m}^2\text{s}]$
Q_s	solar radiation (short wave radiation) in $[\text{J}/\text{m}^2\text{s}]$
Q_{sn}	net incident solar radiation (short wave), $= Q_s - Q_{sr}$

Q_a	atmospheric radiation (long wave radiation) in [J/m ² s]
Q_{an}	net incident atmospheric radiation (long wave)
Q_{ar}	reflected atmospheric radiation in [J/m ² s]
Q_{br}	back radiation (long wave radiation) in [J/m ² s]
Q_{ev}	heat loss due to evaporation (latent) in [J/m ² s]

In D-Flow FM the heat exchange at the free surface is modeled by taking into account the separate effects of solar (short wave) and atmospheric (long wave) radiation, and heat loss due to back radiation, evaporation and convection. The heat losses due to evaporation and convection are functions of the wind speed. In absence of wind, these terms become zero. However, since water vapor is lighter than air, water may be cooled by evaporation and convection even in a no wind situation. The terms are called Q_{evfree} and Q_{cofree} respectively.

Excess temperature model - heat flux model 3

In the Excess temperature model the heat exchange flux at the air-water interface is computed based upon the prescribed background air temperature, the computed water temperature of the top layer and the prescribed wind speed. This relatively simple model is sometimes used in intake-outfall design studies. It can be applied when the temperature mixing process itself is more relevant than the actual heat loss through the air water interface. The applied heat exchange coefficient is mainly a function of the windspeed and water surface temperature.

The excess temperature model 3 is based on [Sweers \(1976\)](#), the heat exchange flux is represented by a bulk exchange formula:

$$Q_{tot} = -\lambda (T_s - T_{back}), \quad (11.1)$$

with T_s the water temperature at the free surface and T_{back} the natural background temperature, both in °C.

The heat exchange coefficient λ is a function of the surface temperature T_s and the wind speed U_{10} . It is derived by linearization of the exchange fluxes for back radiation, evaporation and convection. The following relation was derived by [Sweers \(1976\)](#):

$$\lambda = 4.48 + 0.049T_s + f(U_{10}) (1.12 + 0.018T_s + 0.00158T_s^2). \quad (11.2)$$

Composite - heat flux model 5

The heat flux model 5 following [Gill \(1982\)](#) and [Lane \(1989\)](#) was calibrated for the North Sea and successfully applied for great lakes.

In the Composite heat flux model, the relative humidity in [%], air temperature in [°C] and cloudiness in [%] are prescribed.

These quantities may be either uniform or specially varying. In the external forcingsfile one may have:

```
QUANTITY =humidity_airtemperature_cloudiness
FILENAME =meteo.hac
FILETYPE =6
METHOD =3
OPERAND =0
```

For example input files see example directories:

```
> f20_heat_flux/**/
```

The effective back radiation and the heat losses due to evaporation and convection are computed by the model. Additionally, when air and water densities and/or temperatures are such that free convection occurs, free convection of latent and sensible heat is computed by the model.

Normally, solar radiation is computed based upon time of day, position on earth and cloudiness. However, if solar radiation was measured, it can also be prescribed (in [W/m²]), e.g.:

```
QUANTITY =humidity_airtemperature_cloudiness_solarradiation
FILENAME =meteo.hacs
FILETYPE =6
METHOD =3
OPERAND =0
```

In both heat flux models, the wind forcing may be uniform or spatially varying.

If wind is uniform, the wind speed and direction are prescribed, wind speed is in m/s, and direction follows nautical convention: 0 means wind coming from North, 90 means wind is coming from East. In the external forcings file specify, e.g.:

```
QUANTITY=windxy
FILENAME=zeg99-10.wnd
FILETYPE=2
METHOD=1
OPERAND=0
```

If wind is spatially varying, the air pressure is also prescribed:

```
QUANTITY =airpressure_windx_windy
FILENAME =CSM_2015.apwxwy
FILETYPE =6
METHOD =3
OPERAND =0
```

Air pressure is in [Pa], wind *x*- and *y*-components are given [m/s].

For the physical background of the heat exchange at the air-water interface and the definitions, we refer to [Sweers \(1976\)](#) for the Excess temperature model (Temperaturemodel=3), and to [Gill \(1982\)](#) and [Lane \(1989\)](#) for the Ocean heat flux model (Temperaturemodel=5).

11.1 Heat balance

The total heat flux through the free surface reads:

$$Q_{tot} = Q_{sn} + Q_{an} - Q_{br} - Q_{ev} - Q_{co} - Q_{evfree} - Q_{cofree}, \quad (11.3)$$

with:

Q_{sn}	net incident solar radiation (short wave)
Q_{an}	net incident atmospheric radiation (long wave)
Q_{br}	back radiation (long wave)
Q_{ev}	evaporative heat flux (latent heat)
Q_{co}	convective heat flux (sensible heat)
Q_{evfree}	evaporative heat flux (free convection latent heat)
Q_{cofree}	convective heat flux (free convection sensible heat).

The subscript n refers to a net contribution. Each of the heat fluxes in Equation (11.3) will be discussed in detail.

The change in temperature in the top layer T_s [$^{\circ}\text{C}$] is given by:

$$\frac{\partial T_s}{\partial t} = \frac{Q_{tot}}{\rho_w c_p \Delta z_s}, \quad (11.4)$$

where Q_{tot} [$\text{J/m}^2\text{s}$] is the total heat flux through the air-water surface, c_p ($= 3930 \text{ J kg}^{-1} \text{ K}$) is the specific heat capacity of sea water, ρ_w is the specific density of water [kg/m^3] and Δz_s [m] is the thickness of the top layer. As in Delft3D-FLOW, the heat exchange at the bed is assumed to be zero. This may lead to over-prediction of the water temperature in shallow areas. Also the effect of precipitation on the water temperature is not taken into account.



Remarks:

- ◊ The temperature T is by default expressed in $^{\circ}\text{C}$. However, in some formulas the absolute temperature \bar{T} in K is used. They are related by:

$$\bar{T} = T + 273.15. \quad (11.5)$$

- ◊ In Equation (11.4) the total incoming heat flux is absorbed with exponential decay as a function of depth. See the parameter Secchi-depth in the mdu-file.

11.2 Solar radiation

The short-wave radiation emitted by the sun that reaches the earth surface under a clear sky condition can be evaluated by means of:

- ◊ Applying Stefan-Boltzmann's law for radiation from a black-body:

$$Q = \sigma \bar{T}^4 \quad (11.6)$$

with σ = Stefan-Boltzmann's constant $= 5.67 \times 10^{-8} \text{ J/(m}^2\text{s K}^4\text{)}$ and \bar{T} the (absolute) temperature in K.

- ◊ Direct measurements.

Not all of the radiation is absorbed at the water surface. A part is transmitted to deeper water. Short waves can penetrate over a distance of 3 to 30 meters, depending on the clarity of the water, while the relatively longer waves are absorbed at the surface. Therefore, it is convenient to separate the incoming solar insolation into two portions:

- 1 βQ_{sn} , the longer wave portion, which is absorbed at the surface and
- 2 $(1 - \beta) Q_{sn}$, the remainder part, which is absorbed in deeper water.

The absorption of heat in the water column is an exponential function of the distance H from the water surface:

$$(1 - \beta) Q_{sn} = \int_0^H e^{-\gamma z} dz \Rightarrow \quad (11.7)$$

$$Q_{sn}(h) = \frac{\gamma e^{-\gamma h}}{1 - e^{-\gamma H}} (1 - \beta) Q_{sn}, \quad (11.8)$$

with:

β	part of Q_{sn} absorbed at the water surface which is a function of the wavelength. The default value of β in D-Flow FM is 0.06.
γ	extinction coefficient (measured) in m^{-1} , also related to the so-called Secchi-depth $\gamma = \frac{1.7}{H_{Secchi}}$
h	distance to the water surface in meters.
H	total water depth.

The incoming energy flux at the water surface depends on the angle (declination) between the incoming radiation and the Earth's surface. This declination depends on the geographical position on the Earth and the local time. The Earth axis is not perpendicular to the line connecting the Sun with Earth. This tilting (angle δ) varies with the time of the year and it leads to a seasonal variation of the radiation flux. At June 21, the declination is maximal, 23.5 degrees. Of course, by the rotation of the Earth the solar radiation also varies during the day. Near twelve o'clock local time, the sun elevation above the horizon is maximal. For an overview of the angles used to determine the solar elevation angle γ , see [Figure 11.2](#).

The temporal and latitude-dependent solar elevation angle γ is estimated by:

$$\sin(\gamma) = \sin(\delta) \sin\left(\frac{\pi\phi}{180}\right) - \cos(\delta) \cos\left(\frac{\pi\phi}{180}\right) \cos(\omega_1 t) \quad (11.9)$$

with:

$$\delta = \frac{23.5\pi}{180} \cos(\omega_0 t - 2.95), \quad (11.10)$$

where ω_0 is the frequency of the annual variation and ω_1 the frequency of the diurnal variation; ϕ is the latitude.

The incoming short-wave solar radiation through a clear sky at ground level Q_{sc} is about 0.76 of the flux incident at the top of the atmosphere ([Gill, 1982](#)):

$$Q_{sc} = \begin{cases} 0.76S \sin(\gamma), & \sin(\gamma) \geq 0, \\ 0.0, & \sin(\gamma) < 0. \end{cases} \quad (11.11)$$

The solar constant $S = 1368 \text{ J}/(\text{m}^2\text{s})$ or W/m^2 . This is the average energy flux at the mean radius of the Earth.

A part of the radiation that reaches the water surface is reflected. The fraction reflected or scattered (surface albedo) is dependent on latitude and season. Cloud cover will reduce the magnitude of the radiation flux that reaches the sea surface. The cloudiness is expressed by

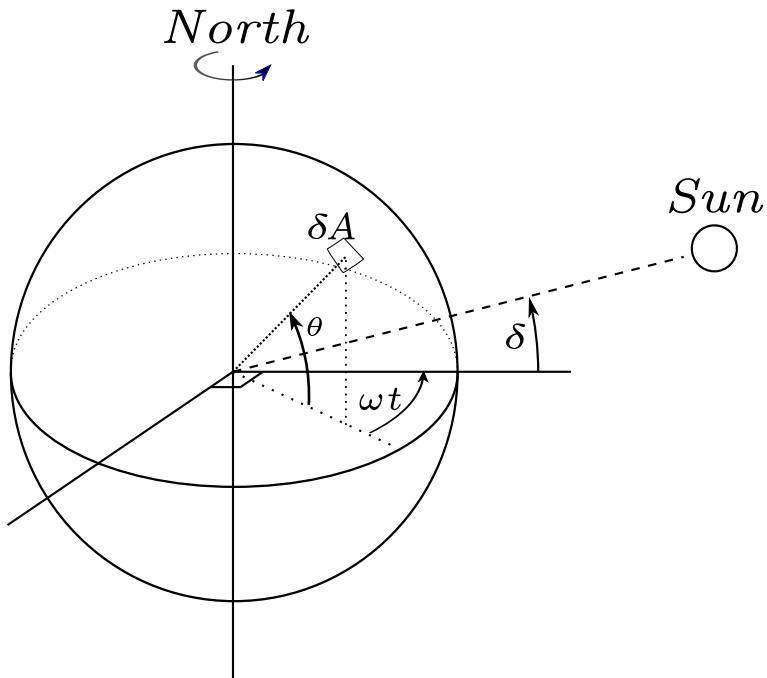


Figure 11.2: Co-ordinate system position Sun
 δ : declination; θ : latitude; ωt : angular speed

a cloud cover fraction F_c , the fraction of the sky covered by clouds. The correction factor for cloud cover is an empirical formula. The absorption of solar radiation is calculated (Gill, 1982) as the product of the net downward flux of short wave-radiation in cloudless conditions and factors correcting for reflection and cloud cover:

$$Q_{sn} = Q_s - Q_{sr} = (1 - \alpha) Q_{sc}(1.0 - 0.4F_c - 0.38F_c^2), \quad (11.12)$$

with:

Q_{sn}	net heat radiation (flux) from the Sun
Q_s	solar radiation (short wave radiation) in $[J/m^2 s]$
Q_{sr}	reflected solar radiation in $[J/m^2 s]$
Q_{sc}	radiation (flux) for clear sky condition
α	albedo (reflection) coefficient ($=0.06$)
F_c	fraction of sky covered by clouds (user-defined input)

11.3 Atmospheric radiation (long wave radiation)

Atmospheric radiation is primarily due to emission of absorbed solar radiation by water vapour, carbon dioxide and ozone in the atmosphere. The emission spectrum of the atmosphere is highly irregular. The amount of atmospheric radiation that reaches the earth is determined by applying the Stefan-Boltzmann's law that includes the emissivity coefficient of the atmosphere ε . Taking into account the effect of reflection by the surface and reflection and absorption by clouds, the relation for the net atmospheric radiation Q_{an} reads (Octavio *et al.*, 1977):

$$Q_{an} = (1 - r) \varepsilon \sigma \bar{T}_a^4 g(F_c), \quad (11.13)$$

where \bar{T}_a is the air temperature (in K) and the reflection coefficient $r = 0.03$. The emissivity factor of the atmosphere ε may depend both on vapour pressure and air temperature. The emissivity of the atmosphere varies between 0.7 for clear sky and low temperature and 1.0. The presence of clouds increases the atmospheric radiation. This is expressed in the cloud function $g(F_c)$.

with T_a the air temperature (in °C). The cloud function $g(F_c)$ in Equation (11.13) is given by:

$$g(F_c) = 1.0 + 0.17F_c^2 \cdot 2 - 9 \quad (11.14)$$

The linearisation of Equation (11.13) is carried out around $T_a = 15$ °C.

Remark:

- ◊ The atmospheric radiation is part of the total long-wave radiation flux, the so-called effective back radiation, see section 11.5.



11.4 Back radiation (long wave radiation)

Water radiates as a near black body, so the heat radiated back by the water can be described by Stefan-Boltzmann's law of radiation, corrected by an emissivity factor $\varepsilon = 0.985$ of water (Sweers, 1976; Octavio et al., 1977) and the reflection coefficient for the air-water interface $r = 0.03$:

$$Q_{br} = (1 - r) \varepsilon \sigma \bar{T}_s^4, \quad (11.15)$$

with \bar{T}_s the (absolute) water surface temperature in K.

11.5 Effective back radiation

The total net long wave radiation flux is computed. This is called the effective back radiation:

$$Q_{eb} = Q_{br} - Q_{an}. \quad (11.16)$$

The atmospheric radiation depends on the vapour pressure e_a , see section 11.6, the air temperature T_a and the cloud cover F_c . The back radiation depends on the surface temperature T_s .

The effective back radiation Q_{eb} is computed following:

$$Q_{eb} = \varepsilon \sigma \bar{T}_s^4 (0.39 - 0.05 \sqrt{e_a}) (1.0 - 0.6 F_c^2), \quad (11.17)$$

with the actual vapour pressure e_a given by Equation (11.22).

11.6 Evaporative heat flux

Evaporation is an exchange process that takes place at the interface between water and air and depends on the conditions both in the water near the surface and the air above it. The evaporation depends on meteorological factors (wind-driven convection) and vapour pressures.

Forced convection of latent heat

The latent heat flux due to forced convection for the ocean heat flux model reads:

$$Q_{ev,forced} = L_V \rho_a f(U_{10}) \{q_s(T_s) - q_a(T_a)\}, \quad (11.18)$$

with q_s and q_a the specific humidity of respectively saturated air and remote air (10 m above water level):

$$q_s(T_s) = \frac{0.62 e_s}{P_{atm} - 0.38 e_s}, \quad (11.19)$$

$$q_a(T_a) = \frac{0.62 e_a}{P_{atm} - 0.38 e_a}. \quad (11.20)$$

The saturated and remote vapour pressures e_s and e_a are given by:

$$e_s = 10^{\frac{0.7859+0.03477T_s}{1.0+0.00412T_s}}, \quad (11.21)$$

$$e_a = r_{hum} 10^{\frac{0.7859+0.03477T_a}{1.0+0.00412T_a}}. \quad (11.22)$$

With L_v the latent heat of vaporisation in J/kg water:

$$L_v = 2.5 \cdot 10^6 - 2.3 \cdot 10^3 T_s. \quad (11.23)$$

The wind function in [Equation \(11.18\)](#) is defined as:

$$f(U_{10}) = c_e U_{10}, \quad (11.24)$$

Without the influence of free convection, the Dalton number c_e in the Composite heat flux model was calibrated for the North Sea to be $c_e = 0.0015$. This value should be close to the C_d coefficient that is used in the computation of wind stresses. The exchange coefficients of latent heat and momentum transfer are closely related. Specifying a negative Dalton number in the mdu file forces the use of the specified C_d coefficient, thus taking into account the specified dependency between windspeed and the C_d coefficient.

Here r_{hum} is the relative humidity in [-].



Remarks:

- ◊ The relative humidity r_{hum} is specified in the input files in percentages.
- ◊ When the computed E is negative, it is replaced by zero, assuming that it is caused by modelling misfit and not by the actual physical process of water condensation out of the air into the water. The same applies to the part associated with free convection.

For the excess temperature model, the wind speed function $f(U_{10})$ following [Sweers \(1976\)](#) is used:

$$f(U_{10}) = (3.5 + 2.0U_{10}) \left(\frac{5.0 \times 10^6}{S_{area}} \right)^{0.05}, \quad (11.25)$$

where S_{area} is the exposed water surface in m^2 , defined in the input and fixed for the whole simulation. The coefficients calibrated by Sweers were based on the wind speed at 3 meter above the free surface; the coefficients in [Equation \(11.25\)](#) are based on the wind speed 10 meter above the water level.

Free convection of latent heat

Loss of heat due to evaporation occurs not only by forced convection, wind driven, but also by free convection. Free convection is driven by buoyant forces due to density differences (by temperature and/or water vapour content) creating unstable conditions in the atmospheric boundary layer. Evaporation due to free convection is important in circumstances where inverse temperature/density gradients are present and wind speeds are almost negligible so that the amount of forced convection is small. Neglecting free convection in this situation will lead to underestimating the heat loss. ([Ryan et al., 1974](#)) developed a correction to the wind function, accounting for free convection. The derivation of evaporation by just free convection is based on the analogy of heat and mass transfer.

The latent heat flux due to free convection reads:

$$Q_{ev,\text{free}} = k_s L_V \bar{\rho}_a (q_s - q_a), \quad (11.26)$$

with the average air density:

$$\bar{\rho}_a = \frac{\rho_{a0} + \rho_{a10}}{2}, \quad (11.27)$$

and with the heat transfer coefficient defined as:

$$k_s = \begin{cases} 0 & \text{if } \rho_{a10} - \rho_{a0} \leq 0 \\ c_{fr.\text{conv}} \left\{ \frac{g\alpha^2}{\nu_{air}\bar{\rho}_a} (\rho_{a10} - \rho_{a0}) \right\}^{1/3} & \text{if } \rho_{a10} - \rho_{a0} > 0 \end{cases} \quad (11.28)$$

where the coefficient of free convection $c_{fr.\text{conv}}$ was calibrated to be 0.14, see (Ryan *et al.*, 1974). The viscosity of air ν_{air} is assumed to have the constant value $16.0 \times 10^{-6} \text{ m}^2/\text{s}$. The molecular diffusivity of air $\alpha \text{ m}^2/\text{s}$ is defined as

$$\alpha = \frac{\nu_{air}}{\sigma}, \quad (11.29)$$

with $\sigma = 0.7$ (for air) the Prandtl number. In Equation (11.26), the saturated air density is given by:

$$\rho_{a0} = \frac{\frac{100P_{atm}-100e_s}{R_{dry}} + \frac{100e_s}{R_{vap}}}{T_s + 273.15}, \quad (11.30)$$

the remote air density (10 m above the water level):

$$\rho_{a10} = \frac{\frac{100P_{atm}-100e_a}{R_{dry}} + \frac{100e_a}{R_{vap}}}{T_{air} + 273.15}, \quad (11.31)$$

where R_{dry} is the gas constant for dry air: 287.05 J/(kg K) and R_{vap} is the gas constant for water vapour: 461.495 J/(kg K). The specific humidity of respectively saturated air and remote air (10 m above the water level), q_s and q_a are given by Equation (11.19) and Equation (11.20). The saturated and remote vapour pressure e_s and e_a are defined in Equation (11.21) and Equation (11.22).

The total heat flux due to evaporation then results from adding the forced convection of latent heat in Equation (11.18) and the free convection of latent heat in Equation (11.26):

$$Q_{ev} = Q_{ev,\text{forced}} + Q_{ev,\text{free}}. \quad (11.32)$$

11.7 Convective heat flux

In the Ocean heat flux model, the convective heat flux is split into two parts, just as the evaporative heat flux. The convective heat flux is divided into a contribution by forced convection and a contribution by free convection.

Forced convection of sensible heat

The sensible heat flux due to forced convection is computed by:

$$Q_{co,\text{forced}} = \rho_a c_p g (U_{10}) (T_s - T_a), \quad (11.33)$$

with c_p the specific heat of air. It is considered constant and taken to be 1 004.0 J/(kg K). The wind-speed function $g (U_{10})$ is defined following Gill (1982):

$$g (U_{10}) = c_H U_{10}, \quad (11.34)$$

with c_H the so-called Stanton number. Without the influence of free convection, the Stanton number was calibrated for the North Sea to be $c_H = 0.00145$.

Free convection of sensible heat

$$Q_{co,\text{free}} = k_s \bar{\rho}_a c_p (T_s - T_a), \quad (11.35)$$

with the heat transfer coefficient k_s given by [Equation \(11.28\)](#).

The total heat flux due to convection then results from adding the forced convection of sensible heat in [Equation \(11.33\)](#) and the free convection of sensible heat in [Equation \(11.35\)](#):

$$Q_{co} = Q_{co,\text{forced}} + Q_{co,\text{free}}. \quad (11.36)$$

12 Wind

Various external influences can exert a force on the flow field. One of these influences is the wind. The force exerted by the wind is coupled to the flow equations as a shear stress. The magnitude is determined by the following widely used quadratic expression:

$$|\tau_s| = \rho_a C_d U_{10}^2 \quad (12.1)$$

where:

- ρ_a the density of air.
 U_{10} the wind speed 10 meter above the free surface (time and space dependent).
 C_d the wind drag coefficient, dependent on U_{10} .

In order to specify the wind shear stress, a drag coefficient is required as well as the wind field in terms of velocity magnitude and wind direction. In this chapter, the backgrounds are provided of how wind fields should be imposed, in addition to [section 5.4.9.4](#). Relevant definitions are addressed in [section 12.1](#), whereas supported file formats are addressed in [section 12.2](#).

12.1 Definitions

When imposing wind conditions, two definitions are respected: a definition for the wind direction (see [section 12.1.1](#)) and a definition regarding the drag coefficient (see [section 12.1.2](#)).

12.1.1 Nautical convention

The wind direction is defined according to the nautical definition, i.e. relative to true North and positive measured clockwise. In [Figure 12.1](#) the wind direction is about +60 degrees, i.e. an East-North-East wind.

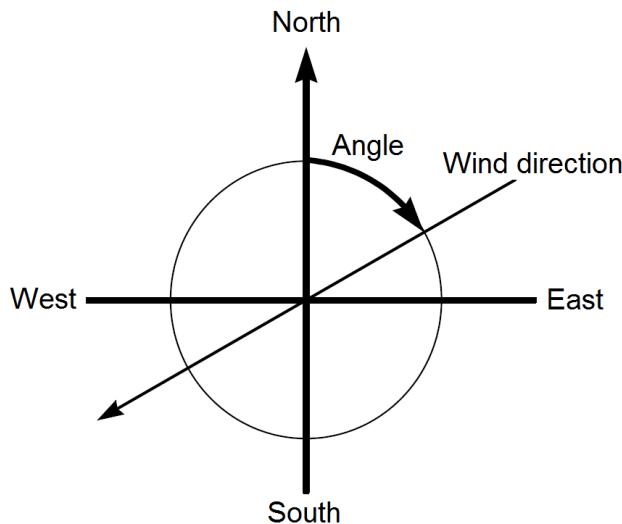


Figure 12.1: Nautical conventions for the wind.

12.1.2 Drag coefficient

The dependency of the drag coefficient on the wind speed should be specified by the user. The user can choose between the following concepts:

- ◊ a constant drag coefficient,
- ◊ a dependency according to [Smith and Banke \(1975\)](#),
- ◊ a dependency according to [Charnock \(1955\)](#),
- ◊ a dependency according to [Hwang \(2005a\)](#) and [Hwang \(2005b\)](#).

The specification of the type of wind drag formulation should be accomplished in the MDU-file. For this purpose, the keyword `ICdtyp` can be utilized. For this keyword `ICdtyp`, five options could be demanded for:

- ◊ `ICdtyp = 1` – constant drag coefficient,
- ◊ `ICdtyp = 2` – linearly varying drag coefficient (cf. [Smith and Banke \(1975\)](#)),
- ◊ `ICdtyp = 3` – piecewise linearly varying drag coefficient (cf. [Smith and Banke \(1975\)](#)),
- ◊ `ICdtyp = 4` – [Charnock \(1955\)](#) formulation (no breakpoints),
- ◊ `ICdtyp = 5` – [Hwang \(2005a\)](#) and [Hwang \(2005b\)](#) formulation (no breakpoints).

If a Smith & Banke type dependency is chosen for, the additional entries `Cdbreakpoints` and `Windspeedbreakpoints` come into play. In the following sections, the specification of either of these options are depicted.

Smith & Banke type formulation

When specifying a Smith & Banke type dependency, the definition as sketched in [Figure 12.2](#) should be kept in mind.

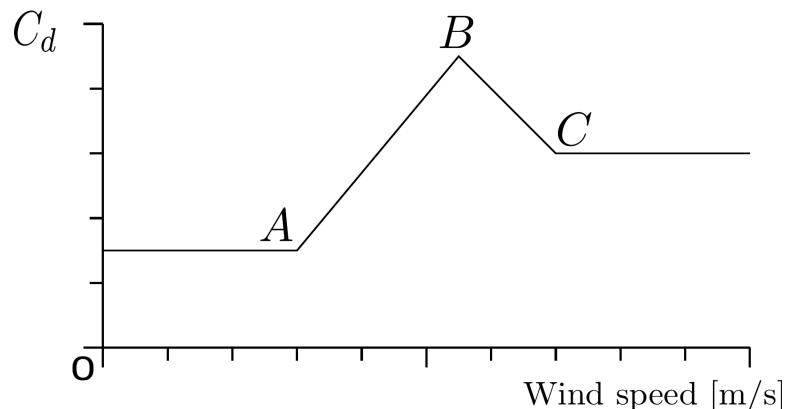


Figure 12.2: Prescription of the dependency of the wind drag coefficient C_d on the wind speed is achieved by means of at least 1 point, with a maximum of 3 points.

From this sketch, it can be seen that the wind drag is considered as dependent on the wind speed in a piecewise linear way. The options, that are facilitated in this respect, are:

- ◊ define *one* set of coordinates (breakpoint A), specifying a constant drag coefficient, valid for all wind speeds,
- ◊ define *two* sets of coordinates (breakpoints A and B), specifying a linearly varying dependency for one range of wind speeds,
- ◊ define *three* sets of coordinates (breakpoints A, B and C), specifying a piecewise linear dependency for two ranges of wind speeds.

Remark that for the latter two options, the drag coefficient is taken constant for wind speeds

lower/higher than the lowest/highest specified wind speed, with a drag coefficient equal to the drag coefficient associated with the lowest/highest specified lowest/highest wind speed. In case of three breakpoints, the expression reads:

$$C_d(U_{10}) = \begin{cases} C_d^A, & U_{10} \leq U_{10}^A, \\ C_d^A + (C_d^B - C_d^A) \frac{U_{10} - U_{10}^A}{U_{10}^B - U_{10}^A}, & U_{10}^A \leq U_{10} \leq U_{10}^B, \\ C_d^B + (C_d^C - C_d^B) \frac{U_{10} - U_{10}^B}{U_{10}^C - U_{10}^B}, & U_{10}^B \leq U_{10} \leq U_{10}^C, \\ C_d^C, & U_{10}^C \leq U_{10}, \end{cases} \quad (12.2)$$

By means of the entries `Cdbreakpoints` and `Windspeedbreakpoints`, the coordinates of the breakpoints (see Figure 12.2) can be specified. Typical values associated with the Smith and Banke (1975) formulation are $C_d = 6.3 \times 10^{-4}$ for $U = 0$ m/s and $C_d = 7.23 \times 10^{-3}$ for $U = 100$ m/s. In this case, the entries in the MDU-file should be specified as follows:

[wind]	
ICdtyp	= 2
Cdbreakpoints	= 0.00063 0.00723
Windspeedbreakpoints	= 0.00000 100.00000

Relative wind

In D-Flow FM, there is the possibility to compute wind shear stress based on the *relative* wind velocity, i.e. relative to the flow velocity. This becomes important when the flow is predominantly forced by the wind and the flow velocity is in the order of the wind velocity. In this way, one can avoid that the wind still forces the flow, despite a zero (or small) difference between flow and wind speed. In case of relative wind Equation (12.1) is changed to

$$|\boldsymbol{\tau}_s| = \rho_a C_d (U_{10} - U_{flow})^2 \quad (12.3)$$

with U_{flow} the flow velocity. This option can be switched on via keyword `Relativewind`.

Charnock formulation

The Charnock formulation (see Charnock (1955)) is based on the assumption of a fully developed turbulent boundary layer of the wind flow over the water surface. The associated wind speed profile follows a logarithmic shape. In the Charnock formulation, the wind speed is considered at 10 meters above the free water surface, hence yielding the following expression:

$$\frac{U_{10}}{u_*} = \frac{1}{\kappa} \ln \left(\frac{z_{10}}{z_0} \right) \quad (12.4)$$

with κ the Von Kármán constant, z_{10} the distance to the water surface (equal to 10 m), u_* the friction velocity and U_{10} the wind speed at 10 m above the water surface. The drag coefficient C_d is defined as:

$$C_d = \frac{u_*^2}{U_{10}^2}. \quad (12.5)$$

Charnock (1955) has proposed to represent the friction of the water surface as z_0 according to:

$$z_0 = \frac{b u_*^2}{g}, \quad (12.6)$$

with g the gravitation acceleration and b a specific constant. Charnock (1955) has proposed $b = 0.012$. The value of the constant b can be specified in the MDU-file by the user by means of one single value for `Cdbreakpoints`. Since the above relation yields an implicit relation for u_* , the system is solved for iteratively. The user should be aware of interpretation of the specified wind field as the wind field at 10 m above the water surface. See paragraph 12.2.4 for a space and time varying Charnock coefficient b .

Hwang formulation

The dynamic roughness could also be related to the steady state wave conditions of the flow field under consideration. The connection of the wave parameters with the drag coefficient as elaborated by Hwang (2005a) is available within D-Flow FM through `ICdtyp = 5`, given a wave field. The Hwang-formulation interprets the user defined wind speed as the wind speed at 10 m above the water surface.

The drag coefficient is computed as:

$$C_d = \left[\frac{1}{\kappa} \ln \left(\frac{k_p z_{10}}{k_p z_0} \right) \right]^{-2} \quad (12.7)$$

with $z_{10} = 10$ m, κ the Von Kármán constant. With wavelength scaling, $k_p z_0$ is the natural expression of the dimensionless roughness, where k_p is the wave number of the spectral peak, computed on the basis of the actual water depth and the provided peak period T_p as wave field. Further following Hwang (2005a),

$$k_p z_0 = \pi \exp \left(-\kappa C_{\lambda/2}^{-0.5} \right) \quad (12.8)$$

in which $C_{\lambda/2}$ is the drag coefficient at half the wavelength above surface. This parameter $C_{\lambda/2}$ is computed as:

$$C_{\lambda/2} = A_{10} \left(\frac{\omega_p U_{10}}{g} \right)^{a_{10}} \quad (12.9)$$

in which $A_{10} = 1.289 \times 10^{-3}$, $a_{10} = 0.815$, U_{10} the wind speed at 10 m above the water surface and ω_p the wave peak frequency ($\omega_p = 2\pi/T_p$). Thus, the drag coefficient C_d is defined.

12.2 File formats

The wind field should be provided by means of an ascii-type file. This file should contain the grid on which the wind field is defined as well as the wind velocity vector(s).

D-Flow FM currently supports four types of wind field prescriptions, i.e. four grid types on which the wind field can be given. This wind grid does not need to be the same as the computational grid. The grid options to provide the wind data on are:

- 1 the computational grid — in this case, no specific wind grid is provided. The provided wind field is considered to be uniform over the entire model area. The wind field can be time dependent.

- 2 an equidistant grid — in this case, a wind field can be prescribed that varies both in space and in time. A Cartesian arcinfo-type grid should be provided on which the wind field is defined.
- 3 a curvilinear grid — this case is conceptually similar to the previous type (the equidistant grid) in the sense that a wind field can be imposed that both varies in space and time. However, a separate file should be provided in which a curvilinear grid is defined (a classic <*.grd>-type file as known from Delft3D-FLOW) on which the wind field is defined.
- 4 a spiderweb grid — this type of wind specification is specially devoted to cyclone winds and is only available in combination with computational grids that are of spherical type. In this case, a cyclone wind field is given on a polar grid with the center ('eye') of the cyclone being the origin of the polar coordinate system. The location of this eye and the associated wind field usually varies in time.

Each of these filetypes can be assigned through the entry in the external forcings file (the <*.ext>-file) named FILETYPE. In this chapter, the various types of wind field specifications are highlighted subsequently. Each of the options is illustrated by means of an example.

12.2.1 Defined on the computational grid

In D-Flow FM, the specification of the wind on the computational grid is equivalent to the specification of a uniform wind, since no separate wind grid is provided to the model. The specification of a uniform wind field can be done in two ways:

- 1 componentwise: as velocity in the longitudinal *x*-direction [m/s] and in the latitudinal *y*-direction [m/s] — the associated FILETYPE in the external forcings file is depicted as uniform, which has FILETYPE=1.
- 2 by magnitude [m/s] and direction [degN] (see [Figure 12.1](#)) — the associated FILETYPE in the external forcings file is depicted as unimagdir, which has FILETYPE=2.

These two types are treated below separately.

12.2.1.1 Specification of uniform wind through velocity components

Since no particular wind grid is used, only timeseries for the *x*-component and the *y*-component of the wind need to be specified. The specification of these timeseries can be done separately (one single file for the *x*-component and one single file for the *y*-component) or jointly (one single file containing the *x*-component and the *y*-component of the wind).

Uniform wind should be provided as an <*.wnd>-file containing either 2 columns (in case of separate specification of the *x*-component and *y*-component of the wind) or 3 columns (in case of joint specification of the velocity components). In either case, the first column contains the time in minutes with respect to the overall reference time.

Example

As an example, a uniform wind field is applied to a certain model. The uniform wind is provided in a file named `windxdirydir.wnd`. The contents of this wind file are:

```
0.00000 10.00000 10.00000
60.00000 -10.00000 -10.00000
```

The first column denotes the time in minutes with respect to the reference date (specified in the mdu-file). The second column denotes the wind velocity in x -direction, whereas the third column denotes the wind velocity in y -direction; both wind components are provided in one single file.

The connection with the flow model itself is laid through the external forcings file. The actual specification of the wind is in this case:

```
QUANTITY =windxy
FILENAME =windxdirydir.wnd
FILETYPE =1
METHOD =1
OPERAND =O
```

Since the two components are given in one single file, the QUANTITY is set to `windxy`. If two separate files would have been provided, the QUANTITY would have been set to `windx` and `windy` over two separate datablocks in the external forcings file.

12.2.1.2 Specification of uniform wind through magnitude and direction

Instead of specifying the separate components of the wind field, the uniform wind vector can also be prescribed through its magnitude and direction (see [Figure 12.1](#)).

This kind of specification should be done by means of one single file, containing three columns, representing the time (in minutes with respect to the reference date), the velocity magnitude [m/s], not necessarily positive, and the direction (nautical convention).

Example

As an example, the previous uniform wind case is reformulated as a case with magnitude and direction of the wind field prescribed. The unimagdir wind is provided in a file named `<windinput.wnd>`. The contents of this file are:

```
0.00000 14.14213562373095 225.00000
60.00000 -14.14213562373095 225.00000
```

The first column denotes the time in minutes with respect to the reference date (specified in the mdu-file). The second column denotes the wind velocity magnitude, whereas the third

column denotes the wind direction. Note that there is a clear difference between the above case and a case in which the magnitude is kept positive (14.1421 m/s) and the direction varies (and hence *rotates!*) from 225 degN to 45 degN.

The connection with the flow model itself is laid through the external forcings file. The actual specification of the wind is in this case:

```
QUANTITY =windxy
FILENAME =windinput.wnd
FILETYPE =2
METHOD =1
OPERAND =0
```

12.2.2 Defined on an equidistant grid

The vector components of the velocity vectors can also be specified on a distinct grid, either of equidistant type or of curvilinear type. In both cases, the characteristics of the grid should be provided. In case of an equidistant grid, the grid is specified in arcinfo-style. That means, the constant grid sizes Δx and Δy should be specified such that a grid is spanned with respect to the location of the lower left corner of the grid (either the center of the lower left cel or the lower left corner of the lower left cell).

Example

As an example, a grid with $\Delta x = \Delta y = 100$ m is spanned, based on the center of the lower left cell, located at $x = y = 60$ m with respect to the origin. The input data for the x -component and the y -component should be specified separately, in two distinct files. The input of the x -component data should be given in an $<*.amu>$ -type file, such as $<\text{windxdir.amu}>$ as an example:

```
### START OF HEADER
### This file is created by Deltares
### Additional comments
FileVersion      = 1.03
filetype        = meteo_on_equidistant_grid
NODATA_value   = -9999.0
n_cols          = 5
n_rows          = 4
grid_unit       = m
x_llcenter     = 60
y_llcenter     = 60
dx              = 110
dy              = 110
n_quantity      = 1
quantity1       = x_wind
unit1           = m s-1
### END OF HEADER
TIME = 0 hours since 2006-01-01 00:00:00 +00:00
10 10 10 10 10
10 10 10 10 10
10 10 10 10 10
10 10 10 10 10
TIME = 1 hours since 2006-01-01 00:00:00 +00:00
-10 -10 -10 -10 -10
-10 -10 -10 -10 -10
```

```
-10 -10 -10 -10 -10  
-10 -10 -10 -10 -10
```

For the *y*-component data, a similar file (e.g. <windydir.amv>) should be provided. In addition, the pressure could be specified in a similar file (e.g. <pressure.amp>). Note that `x_llcorner` and `y_llcorner`, instead of `x_llcenter` and `y_llcenter`, are also supported.

Wind on an equidistant grid has been provided a filetype specification as `FILETYPE=4`. The connection with the flow model itself is laid through the external forcings file. The actual specification of the wind is in this case:

```
QUANTITY =windx
FILENAME =windxdirection.amu
FILETYPE =4
METHOD =2
OPERAND =O

QUANTITY =windy
FILENAME =windydir.amv
FILETYPE =4
METHOD =2
OPERAND =O

QUANTITY =atmosphericpressure
FILENAME =pressure.amp
FILETYPE =4
METHOD =2
OPERAND =O
```

12.2.3 Defined on a curvilinear grid

In analogy with the wind specification on an equidistant grid, the wind can be specified on a curvilinear grid. This curvilinear grid should be provided as a classic <*.grd>-file as known from Delft3D-FLOW. A difference with the equidistant grid wind is the necessity to compile all data blocks (i.e. pressure, *x*-component and *y*-component) in one single file. This file should have the extension <*.apwxwy>. The sequence of this datablock is: 1) pressure, 2) *x*-velocity component, 3) *y*-velocity component.

Example

As an example, a curvilinear grid named <meteo.grd> is present, providing the underlying coordinates of the wind data field. The input data, comprising the atmospheric pressure, the *x*-velocity component *and* the *y*-velocity component, are given in one single file (as is compulsory). The contents of the example <meteo.apwxwy>-file is:

```
### START OF HEADER
### This file is created by Deltares
### Additional comments
FileVersion      =    1.03
filetype        =    meteo_on_curvilinear_grid
NODATA_value   =    -9999.0
grid_file       =    meteo.grd
```

Note that `grid_llcenter`, instead of `grid_llcorner`, is also supported. On the contrary, `grid_column` is *not* supported instead of `grid_row`.

Wind on a curvilinear grid has been provided a filetype specification as FILETYPE=6. The connection with the flow model itself is laid through the external forcings file. The actual specification of the wind is in this case:

```
QUANTITY =airpressure_windx_windy  
FILENAME =meteo.apwxwy  
FILETYPE =6  
METHOD =3  
OPERAND =O
```

Notice that METHOD=3 is chosen for wind on a curvilinear grid, instead of METHOD=2 in case of wind on an equidistant grid.

12.2.4 Space and time varying Charnock coefficients

The value for the Charnock coefficient b in Eq. ((12.6)) can be a constant given in the MDU-file, but also be given as a space and time varying field.

For a space and time varying Charnock coefficient, the user should provide a NetCDF-file with meteorological forcing, including Charnock coefficients, and use the file specification FILETYPE=11.

The specification is in this case:

```
QUANTITY =airpressure_windx_windy_charnock
FILENAME =meteo.nc
FILETYPE =11
METHOD =3
OPERAND =O
```

12.2.5 Defined on a spiderweb grid

Cyclone winds can be imposed by means of a ‘spiderweb’-like polar grid. The origin typically coincides with the cyclone eye and can move in time. Spiderwebs can only be used in combination with a spherical computational grid. The origin of the spiderweb should be given as longitude (for x_{eye}) and latitude (for y_{eye}). The number of rows (discretisation in radial direction) and the number of columns (discretisation in angular direction) should be given, as well as the radius of the grid (in meters). The definition of the spiderweb grid is illustrated in Figure 12.3.

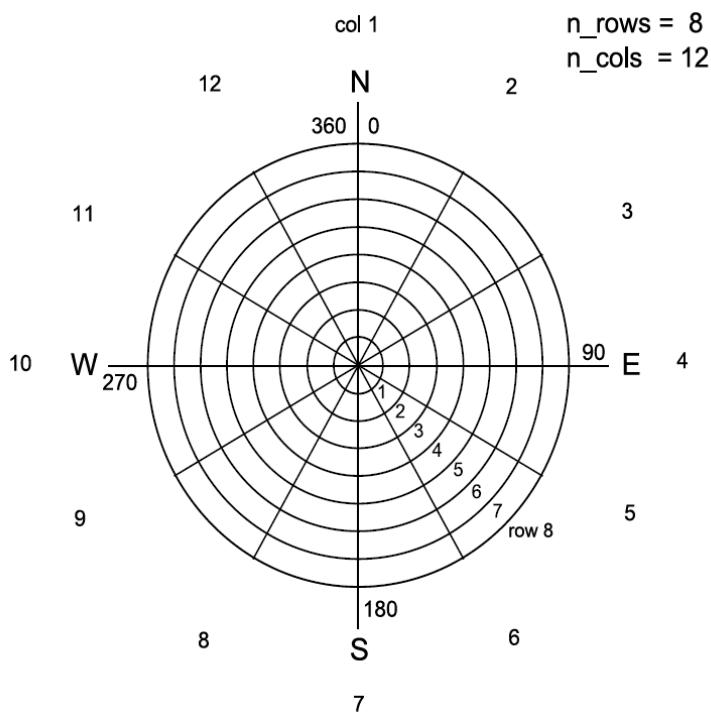


Figure 12.3: Grid definition of the spiderweb grid for cyclone winds.

The files containing the spiderweb data and metadata have the extension <*.spw>. They consist of a global header containing properties that are not varying in time, followed by blocks of data for subsequent time levels. Each of these data blocks is headed by a set of properties for the corresponding time level. A detailed description of the spiderweb file format can be found in [section C.12.3](#).

Through the specified unit for atmospheric pressure in the file, it can be specified whether the values should be interpreted as mbar (=hPa), instead of Pa, which is the default. Specifying the spiderweb merge fraction β in `spw_merge_frac` allows for linear fading of wind speed and pressure drop towards the outer rim. For a spiderweb with radius R , the weight assigned to the spiderweb wind and pressure at a radius r is given by $(R - r)/\beta R$ for r between $(1 - \beta)R$ and R . The weight equals unity within the inner circle and zero beyond the outer rim.

Example

As an example, a spiderweb grid named <`spwsimple.spw`> is present, providing the underlying coordinates of the wind data field. The input data, comprising the atmospheric pressure drops, the wind velocity magnitudes (in [m/s]) *and* the wind directions (in [degN]), are given in one single file (as is compulsory). The contents of the example <`spwsimple.spw`>-file is:

```
### Spiders web derived from TRACK file: gonu.trk
### This file is created by Deltares
### All text on a line behind the first # is parsed as commentary
### Additional comments
FileVersion      = 1.03
filetype        = meteo_on_spiderweb_grid
### Spiders web derived from TRACK file: gonu.trk
### This file is created by Deltares
### All text on a line behind the first # is parsed as commentary
### Additional comments
NODATA_value    = -1001
n_cols          = 4
n_rows          = 4
spw_radius     = 600000.0
spw_merge_frac = 0.75
spw_rad_unit   = m
### END OF HEADER
TIME            = 340000.00 minutes since 2005-01-01 00:00:00 +00:00
x_spw_eye       = 265.00
y_spw_eye       = 33.00
p_drop_spw_eye  = 7000.000
  5.000000  5.000000  5.000000  5.000000
10.000000 10.000000 10.000000 10.000000
15.000000 15.000000 15.000000 15.000000
20.000000 20.000000 20.000000 20.000000
  270.00      0.00    90.00   180.00
  270.00      0.00    90.00   180.00
  270.00      0.00    90.00   180.00
  270.00      0.00    90.00   180.00
  4000.00    4000.00  4000.00  4000.00
  3000.00    3000.00  3000.00  3000.00
  2000.00    2000.00  2000.00  2000.00
  1000.00    1000.00  1000.00  1000.00
TIME            = 380000.00 minutes since 2005-01-01 00:00:00 +00:00
x_spw_eye       = 275.00
y_spw_eye       = 18.00
p_drop_spw_eye  = 8000.000
  5.000000  5.000000  5.000000  5.000000
10.000000 10.000000 10.000000 10.000000
```

15.000000	15.000000	15.000000	15.000000
20.000000	20.000000	20.000000	20.000000
270.00	0.00	90.00	180.00
270.00	0.00	90.00	180.00
270.00	0.00	90.00	180.00
270.00	0.00	90.00	180.00
4000.00	4000.00	4000.00	4000.00
3000.00	3000.00	3000.00	3000.00
2000.00	2000.00	2000.00	2000.00
1000.00	1000.00	1000.00	1000.00

Wind on a spiderweb grid has been provided a filetype specification as FILETYPE=5. The connection with the flow model itself is laid through the external forcings file. The actual specification of the wind is in this case:

```
QUANTITY =airpressure_windx_windy
FILENAME =spwsimple.spw
FILETYPE =5
METHOD   =1
OPERAND  =O
```

Notice that METHOD=1 is chosen for wind on a spiderweb grid, instead of METHOD=2 in case of wind on an equidistant grid and METHOD=3 in case of wind on a curvilinear grid.

Alternatively, the spiderweb wind and pressure can be specified in the external forcings file as separate quantities referring to the same file (or specify one and omit the other):

```
QUANTITY =windxy
FILENAME =spwsimple.spw
FILETYPE =5
METHOD   =1
OPERAND  =O

QUANTITY =airpressure
FILENAME =spwsimple.spw
FILETYPE =5
METHOD   =1
OPERAND  =O
```

(NB. There is a slight difference between both specifications regarding their effect on the flow. The first approach requires an additional interpolation of wind on the velocity points, which in some cases of coarse computational grids and small scale wind could introduce smoothing.)

12.2.6 Combination of several wind specifications

The combination of the various wind specification types can *only* be achieved if the QUANTITY of the winds to be combined is the same, for instance QUANTITY=windx. The option OPERAND=+ can be used to add a wind field to an existing wind field.

Example

If the uniform wind is to be combined with a wind specified on an equidistant grid, then the wind field could be assigned in the external forcings file as follows:

```

QUANTITY =windx
FILENAME =windxdir.wnd
FILETYPE =1
METHOD =1
OPERAND =O

QUANTITY =windy
FILENAME =windydir.wnd
FILETYPE =1
METHOD =1
OPERAND =O

QUANTITY =windx
FILENAME =windxdir.amu
FILETYPE =4
METHOD =2
OPERAND =+

QUANTITY =windy
FILENAME =windydir.amv
FILETYPE =4
METHOD =2
OPERAND =+

```

The same is possible for e.g. a uniform wind and two spiderweb cyclones:

```

QUANTITY=windxy
FILENAME=uni.tim
FILETYPE=2
METHOD=1
OPERAND=O

QUANTITY =windxy
FILENAME =spwsimple.spw
FILETYPE =5
METHOD =1
OPERAND =+

QUANTITY =windxy
FILENAME =spwsimple2.spw
FILETYPE =5
METHOD =1
OPERAND =+

QUANTITY =atmosphericpressure
FILENAME =spwsimple.spw
FILETYPE =5
METHOD =1
OPERAND =+

QUANTITY =atmosphericpressure
FILENAME =spwsimple2.spw
FILETYPE =5

```

```
METHOD      =1
OPERAND    =+
```

In the above example, the wind is first prescribed as a spatially uniform time-varying field, onto which the spiderweb winds are added. Note that the uniform wind has the ‘O’ operand, which means that it overrides rather than adds. However, since the default wind is zero, one might just as well have used the ‘+’ operand. If the spiderweb merge fraction is specified in the spiderweb file, a gradual transition between the spiderweb wind and the background wind is applied using the linearly varying weight as described above. The same is done for atmospheric pressure, if specified except that pressure is added to, whereas wind is averaged with the background values. Without any input, the background atmospheric pressure is set to PavBnd in the section [wind] in the MDU-file, if present and zero otherwise.

12.3 Masking of points in the wind grid from interpolation ('land-sea mask')

A mask can be supplied by the user to prevent selected points in the wind grid from contributing to the wind interpolation on velocity points, e.g. to exclude land points. This feature was included to conform to SIMONA and therefore implemented in the same way.

For each individual grid point for which to interpolate from the wind grid:

- ◊ Masked wind points are excluded from the interpolation.
- ◊ The total of the weight factors for the remaining wind points is determined.
- ◊ If this total falls below 1E-03, the mask is ignored and the original bilinear weights are used.
- ◊ Otherwise, the weights for the remaining wind points are normalised again.

The effect of the mask, when applied as a land-sea mask, is that for velocity points close to shore the interpolated wind is no longer influenced by the wind over land (which would otherwise yield a zone of points with reduced wind near the shore).

Specification and format of the mask file

The name of the mask file, if any, is specified in the <.ext> file, labelled SOURCEMASK, directly following the FILENAME specification, e.g.:

```
QUANTITY      =windxy
FILENAME     =meteo.wxwy
SOURCEMASK   =meteo_mask.asc
FILETYPE     =6
METHOD       =3
OPERAND      =O
```

The mask file itself has the same layout as the wind file, though the number of required header fields is reduced, e.g.:

```
FileVersion      =      1.03
.
.
```

```
unit1      = Pa
### END OF HEADER
1       1       1       1       1
1       1       1       0       0
1       1       1       0       0
1       1       0       0       0
```

The lines in the header are ignored. The number of columns and rows in the matrix of ones and zeros should match those of a block (for a single variable and a single timestep) in the meteo files. Zeros signify the position of rejected points (and ones those of the accepted points) in the wind grid.

13 Hydraulic structures

13.1 Introduction

Obstacles in the flow may generate sudden transitions from flow contraction to flow expansion. The grid resolution is often low compared to the gradients of the water level, the velocity and the bathymetry. The hydrostatic pressure assumption may locally be invalid. Examples of these obstacles in civil engineering are: gates, barriers, dams, groynes and weirs. The obstacles generate energy losses and may change the direction of the flow.

The forces due to obstacles in the flow which are not resolved (sub-grid) on the horizontal grid, should be parameterised. The obstacles are denoted in D-Flow FM as hydraulic structures. In the numerical model the hydraulic structures should be located at velocity points of the grid. The direction of the forces should be specified at input. To model the force on the flow generated by a hydraulic structure, a quadratic energy or linear loss term is added to the momentum equation.

Note: Structure definitions can be made in Delta Shell, and will then be saved into a structures <*.ini>-file ([section C.11](#)).



13.2 Structures

The user can insert the hydraulic structures by the means of polygons on the grid. By selecting the required structure and drawing a polygon on the computational grid, the location of structure can be defined (see [Figure 13.1](#)). The supported structures in D-Flow FM are

- ◊ Fixed Weirs
- ◊ (Adjustable) weirs
- ◊ Gates
- ◊ Pumps
- ◊ Thin dams

In practice, the word 'barrier' is often used for structures. However, in D-Flow FM such structures are modelled as a gate or a weir in combination with a so-called Control Group (D-Real Time Control model).

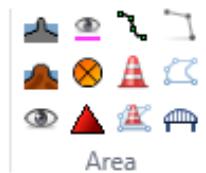


Figure 13.1: Selection of structures (and other items) in the toolbar.

13.2.1 Fixed weirs

In D-Flow FM, a fixed weir is a fixed non-regulated construction generating energy losses due to constriction of the flow. They are commonly used to model sudden changes in depth (roads, summer dikes) and groynes in numerical simulations of rivers. Such structures are applied to keep the river in its bed for navigation purposes. Others are built, for instance, to protect an area behind a tidal weir from salt intrusion. Upstream of a weir the flow is accelerated due to contraction and downstream the flow is decelerated due to expansion. The expansion introduces an important energy loss due to turbulence. The energy loss is dependent on the shape of the weir and its crest level.

Weirs are located in D-Flow FM on the velocity points of the computational grids. For a description of the input of weirs, we refer you to section 5.4.2.9.

In D-Flow FM, two different approaches are applied to simulate the energy losses by fixed weirs. First of all, a numerical approach has been implemented. Then, a special discretization of the advective terms around the fixed weir is applied. For a detailed description we refer to [D-Flow FM TRM \(2015, section 6.7.1–6.7.4\)](#). Next to the numerical approach, there is an empirical approach. Based on measurements in flume laboratories empirical formulae can be derived in order to match the measurements as accurately as possible. In this empirical approach, the energy loss due to a weir is described by the loss of energy height ($[m]$). The energy loss in the direction perpendicular to the weir is denoted as ΔE . This energy loss is added as an opposing force in the momentum equation by adding a term $-g\Delta E/\Delta x$ to the right hand side of the momentum equation, resulting in a jump in the water levels by ΔE at the location of the weir. For the computation of the energy loss ΔE two options are available in D-Flow FM, namely the so-called 'Tabellenboek' and 'Villemonte' approaches. The two corresponding empirical formulas have been taken from the Simona software suite. For a detailed description of both formulas we refer to [D-Flow FM TRM \(2015, section 6.7.5–6.7.6\)](#). In this empirical approach the discretization of the advective terms does not change, unlike the numerical approach in D-Flow FM for modelling fixed weirs.

13.2.2 (adjustable) Weirs

Unlike the fixed weir (with fixed crest level), an adjustable weir has geometric parameters that can be adjusted in time. The controllable weirs, which are called weirs or gates in D-Flow FM, are weirs which can be controlled with a predefined time series or get controlled based on the water level or other conditions. Two types of weirs are possible in D-Flow FM,

- 1 a so-called simple weir and
- 2 a general structure.

Simple weir

In case of a simple weir with constant parameters, only a crest level, a crest width and a contraction coefficient can be specified, which is shown in [Figure 13.2](#). The crest level can also be defined as time series. The actual flow is still calculated using the general structure formulation, but with all other geometric parameters automatically set to match the weir's geometry.

Name: weir01

Structure type Simple weir ▾

Weir

Crest level	0	m AD	<input type="checkbox"/>
Crest width		m	

Time Series

Coefficients

Contraction coefficient 1

Figure 13.2: Input for simple weir

The structure parameters for a (adjustable) weir with time varying crest level can be defined via the <structures.ini> file:

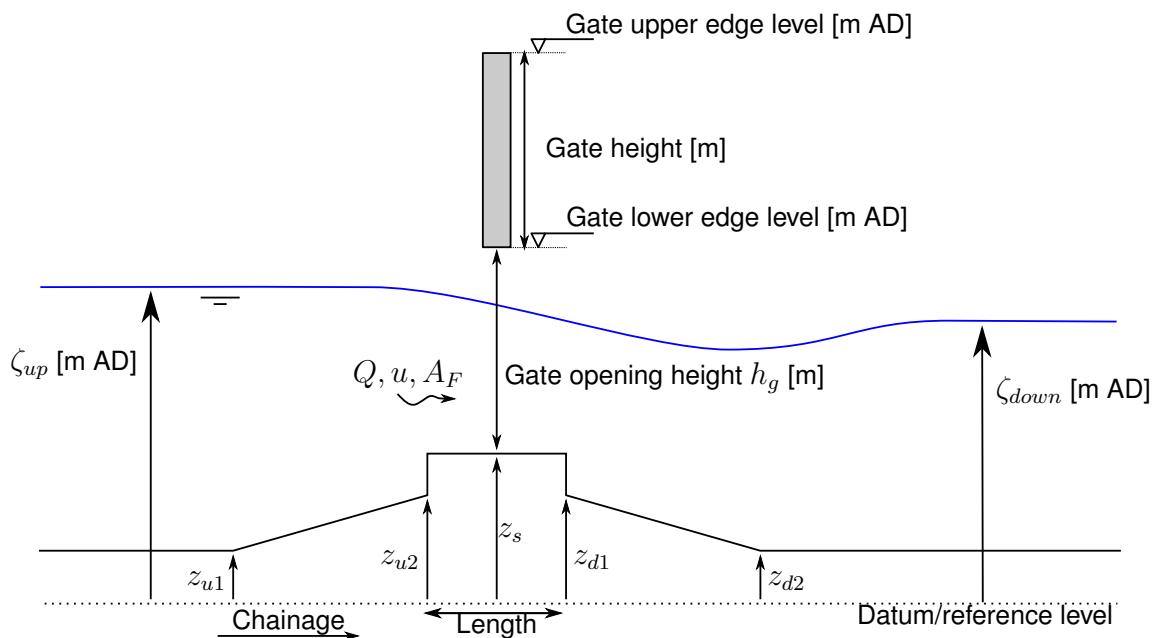
```

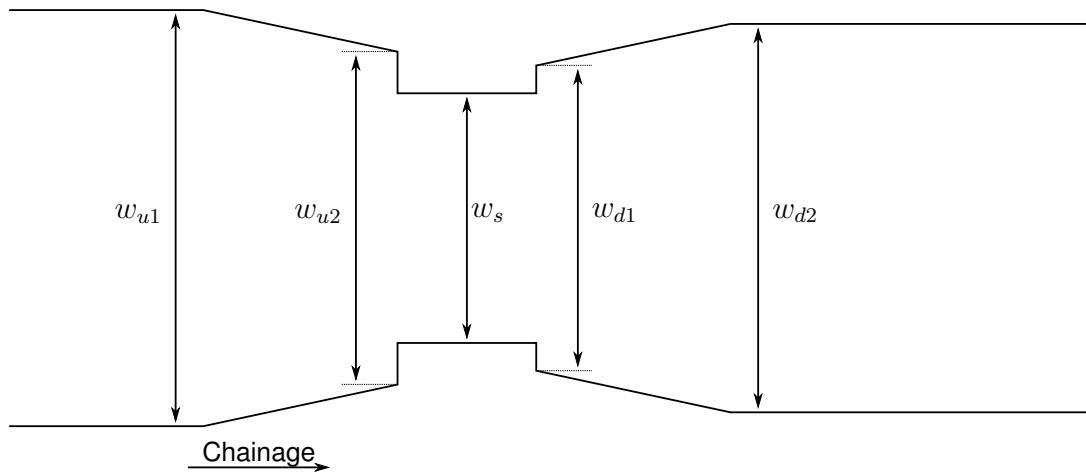
type      = weir
id       = weir01
polylinefile = weir01.pli
crest_level = weir01_crest_level.tim
crest_width =
lat_cont_coeff = 1

```

General structure

The general structure gives more freedom in defining the dimensions and the geometry of the hydraulic structure. The geometrical shape is given in [Figure 13.3](#) and [Figure 13.4](#). The discharge through a general structure is computed on basis of upstream and downstream energy levels.

**Figure 13.3:** General structure, side view

**Figure 13.4:** General structure, top view

In the GUI of D-Flow FM the coefficients of a general structure can be set as shown in Figure 13.5. Flow across the general structure can be of the following types:

- ◊ free gate flow,
- ◊ drowned gate flow,
- ◊ free weir flow, and
- ◊ drowned weir flow.

The choice depends on the dimensions of the structure and the flow conditions. Whether or not the gate is in the flow or above the flow yields either submerged or free flow. Furthermore, the flow can be either subcritical or critical. Both for incoming flow ("Flow") which represents flood and for outgoing flow ("Reverse") which represents ebb, contraction coefficients can be specified. This can be seen as tuning parameters for the user.


Warning:

- ◊ Order of edit fields is in the GUI (Figure 13.5) is different as shown in figures 13.3 and 13.4. The figures 13.3 and 13.4 show the situation as interpreted by the computational core (D-Flow FM)

The five geometric parameters as shown in Figure 13.3 can be specified, see Figure 13.5

z_s	Crest level,
z_{u1}	Upstream 1,
z_{u2}	Upstream 2,
z_{d1}	Downstream 1, and
z_{d2}	Downstream 2

Similarly for the five values for the width of the general structure, see Figure 13.4.

w_s	Width at crest level,
w_{u1}	Width upstream 1,
w_{u2}	Width upstream 2,
w_{d1}	Width downstream 1, and
w_{d2}	Width downstream 2

The crest level can also be given as time series.

Name: structure01

Structure type: General structure

Sill

	Time Series	Upstream 2	Upstream 1	Downstream 1	Downstream 2
Crest level	0 m AD	<input checked="" type="checkbox"/>	0	0	0
Crest width	m				

Gate

	Time Series	
Lower edge level	0 m AD	<input checked="" type="checkbox"/>
Door opening height	0 m	
Door height	0 m	
Horizontal opening width	0 m	<input checked="" type="checkbox"/>
Horizontal opening direction	Symmetric	

Coefficients

	Positive	Negative
Free gate flow	1	1
Drowned gate flow	1	1
Free weir flow	1	1
Drowned weir flow	1	1
Contraction coefficient	1	1

Extra Resistance

Extra Resistance	0
------------------	---

Figure 13.5: Input for a general structure.

How to determine the crest width?

A (2D) hydraulic structure is given as a single geometrical object by the user, but it may cross multiple 2D grid cells (or: flow links).

To compute the crest width the following algorithm is applied:

- ◊ If the crest width is not specified in the input, then the sum is taken of the width of all flow links that are intersected by this polyline. Therefore, it is advised to align the polyline of a structure with the grid. For example, if a polyline has an angle of 45 degrees with the grid, then the crest length is a factor of $\sqrt{2}$ larger than needed. Note that—contrary to these hydraulic structures—for fixed weirs a correct length of the polyline is computed, because all flow links of fixed weirs are scaled with the correct factor; see Section 6.7.7 in the Technical Reference Manual D-Flow FM TRM (2015).
- ◊ If the crest width is specified and this value is larger than the sum of the length of the flow links that are intersected, then the minimum of these two values is taken. So, a crest width cannot be larger than the sum of the length of the flow links that are intersected.

How to determine the corresponding flow links of a gate opening?

The GateOpeningWidth is a compulsory parameter for gates and general structures. A single gate door with no horizontal movement should use GateOpeningWidth=0. A structure that has one or two gate doors that *do* have horizontal opening and closing movement should specify the width of the horizontal opening between the doors. The polyline of a gate is snapped to one or more flow links. Each flow link under the influence of this structure is modelled as a general structure with the correct geometrical parameters. Depending on the gate door positions, some flow links may have only flow across the weir crest in the opening, whereas others may have gate flow. Starting from the left and right end points of the polyline, snapped flow links are taken for as long as their sum is less than the closed width (which equals CrestWidth minus GateOpeningWidth). The remaining links in the middle will have weir flow without gate doors. At both ends of the gate opening flow links may be partially opened so that the GateOpeningWidth exactly matches the sum of the length of the open flow links.

13.2.3 Gates

Constructions that partially block the horizontal flow can be modelled as so-called "gates". Its horizontal and vertical position can be specified. Upstream of the gate the flow is accelerated due to contraction and downstream the flow is decelerated due to expansion. A gate may include two types of openings, namely, in horizontal and in vertical directions. In two-dimensional simulations, the vertical effect is parameterized by a quadratic energy loss term.

The horizontal effect is mimicked by setting the velocities of the computational faces (at position of the gate) to zero. This generates structure of the horizontal flow around the gate which is more realistic. There is no transport of salt or sediment through the blocked computational faces of a gate. The width of a gate is an user defined parameter. If the user does not define it, the width is assumed to be zero, so it has no influence on the water volume.

In D-Flow FM the gates can be imposed by polygons, and can be edited in a similar way as the other structures. For more details on gates in Delta Shell, we refer you to section 5.4.2.10. In Figure 13.6 the geometric parameters of a gate are shown.

Name:	gate01
Structure type	Simple gate
Sill	
Crest level	0
Crest width	m
Gate	
Lower edge level	0
Door opening height	0
Door height	0
Horizontal opening width	0
Horizontal opening direction	Symmetric

Figure 13.6: Input for a gate

The structure of the input file for the gates is as follows:

```

type                      = gate
id                       = gate01
polylinefile              = gate01.pli
sill_level                = 7
lower_edge_level          = 15
opening_width              = gate01_opening_width.tim
door_height                = 5
horizontal_opening_direction = symmetric
sill_width                 =

```

13.2.4 Pumps

Pumps are another type of structures in D-Flow FM. Unlike the other structures, a pump can force the flow only on one direction. However, pumps can be defined by polygons, like all other structures in D-Flow FM.

The pump includes specific capacity, and pumps the water by its capacity, as long as the water level is sufficient. In the case, the water level is lower than a required value, pump will not pump any flow, despite of their capacity. The structure of the input file for the pumps is as follows:

```

type                      = pump
id                       = pump01
polylinefile              = pump01.pli
capacity                 = pump01_capacity.tim

```

13.2.5 Thin dams

Thin dams are similar to fixed weirs. The only difference between the thin dams and fixed weirs are in their crest levels. Thin dams, in principle, include infinitely high crest levels and hence, they do not allow water flux. Similar to the other structures, the thin dams can be selected from the toolbar and drawn by a polygon. D-Flow FM adjusts the polygon to the nearest velocity points. The input data for a thin dam is identical to those for fixed-weir, except for the crest level.

14 Bedforms and vegetation

The terrain and vegetation exert shear stresses on the passing flow. The magnitude of the shear stress of the bed is often characterised by means of roughness coefficient of type Chézy, Manning or White-Colebrook. Within the main stream flow the shear stresses are largely determined by the local conditions of the alluvial bed (bed composition and bedform characteristics). In other areas, such the floodplains of rivers and in the intertidal areas of estuaries, the flow resistance is determined by a combination of vegetation and an alluvial bedforms or even a non-alluvial bed. To accurately represent such conditions in the numerical model, D-Flow FM has been extended with a vegetation model. Another related feature known from Delft3D-FLOW is the bedform roughness predictors; these are not available in D-Flow FM yet. These types of flow resistance may be resolved in a 2D numerical model using the trachytope approach (see [section 14.2](#)).

14.1 Bedform heights

The dune height and [Van Rijn \(2007\)](#) bedform roughness predictors, known from Delft3D-FLOW, are not available yet in D-Flow FM. They will be in an upcoming release.

14.2 Trachytopes

This functionality allows you to specify the bed roughness and flow resistance on a sub-grid level by defining and using various land use or roughness/resistance classes, further referred to as trachytopes after the Greek word *τραχύτης* for roughness. The input parameters and files to use the trachytopes functionality are described in [section C.6](#).

At every time step (or less frequent as requested by the user) the trachytopes are converted into a representative bed roughness C , k or n and optional linear flow resistance coefficient λ per velocity point with index j .

$$M = -\frac{1}{2}\lambda_j u_j |\mathbf{u}_j| \quad (14.1)$$

To save computational time the user may choose to update the computed bed roughness and resistance coefficients less frequently than every time step. See [section C.6](#) for a description of the keywords and input files associated with this feature.

The following two sections describe the various classes of trachytopes distinguished and the way in which they are combined, respectively.

14.2.1 Trachytope classes

Three base classes of trachytopes are distinguished: area classes, line classes and point classes. The area classes (type range 51–200) basically cover the whole area, therefore, they are generally the dominant roughness factor. The line classes (type range 201–250) may be used to represent hedges and similar flow resistance elements; it will add anisotropy to the roughness field. The point class (type range 251–300) represents a set of point flow resistance elements. The next six sections provide an overview of the various trachytope formulae implemented.

Special classes (1–50)

In addition to the three base classes two special trachytope classes have been defined: a flood protected area and a composite trachytope class. The first class represents a sub-grid area

that is protected from flooding and thus does not contribute to the bed roughness; however, the effect on the flow resistance should be taken into account. The second class can be used to make derived trachytype classes that are a combination of two other trachytypes: an area fraction α of trachytype type T_1 and an area fraction β (often equal to $1 - \alpha$) of trachytype type T_2 .

FormNr	Name	Formula
Special classes (1–50)		
1	flood protected area	area fraction shows up as f_b in Eqs. (14.53) and (14.56)
2	composite trachytype	fraction α of type T_1 and fraction β (generally $\beta = 1 - \alpha$) of type T_2

Area trachytype classes (51–200)

The class of area trachytypes is subdivided into three types: simple (51–100), alluvial (101–150) and vegetation (151–200). Four simple area trachytypes have been implemented representing the four standard roughness types of flow module.

FormNr	Name	Formula
51	White-Colebrook value	k
52	Chézy value	C
53	Manning value	$C = \sqrt[6]{h}/n$
54	z_0 value	$k = 30z_0$

Six alluvial trachytypes have been implemented.

FormNr	Name	Formula
101	simplified Van Rijn power relation	Equation (14.2)
102	Van Rijn (1984)	Equation (14.3)
103	Struiksma	Equations (14.4) to (14.12)
104	bedforms quadratic	Equations (14.13) to (14.16)
105	bedforms linear	Equation (14.17)
106		Equation (14.18)

The first alluvial roughness formula is a simplified version of the [Van Rijn \(1984\)](#) alluvial roughness predictor

$$k = Ah^{0.7} \left[1 - e^{-Bh^{-0.3}} \right] \quad (14.2)$$

it is obtained from [Equation \(14.4\)](#) by noting that $h_b \propto h^{0.7}$ and $L_b \propto h$ and ignoring the grain related roughness. The parameters A and B can be calibrated by the user. The second formula implemented is a straightforward general power law

$$C = Ah^B \quad (14.3)$$

where A and B are calibration coefficients. The Van Rijn (1984) alluvial roughness predictor reads

$$k = k_{90} + 1.1h_b (1 - e^{-25h_b/L_b}) \quad (14.4)$$

where the bedform height h_b and length L_b are given by

$$h_b = 0.11h \left(\frac{D_{50}}{h} \right)^{0.3} (1 - e^{-T/2}) (25 - T) \quad (14.5)$$

$$L_b = 7.3h \quad (14.6)$$

where h is the local water depth and the transport stage parameter T is given by

$$T = \frac{u'^2 - u_{*,cr}^2}{u_{*,cr}^2} \quad (14.7)$$

where u' is the bed shear velocity given by

$$u'^2 = gu^2/C_{g,90}^2 \quad (14.8)$$

where

$$C_{g,90} = 18^{10} \log(12h/k_{90}) \text{ and } k_{90} = 3D_{90} \quad (14.9)$$

and $u_{*,cr}$ is the critical bed shear velocity according Shields given by

$$u_{*,cr}^2 = g\Delta D_{50}\theta_c \quad (14.10)$$

given

$$\theta_c = \begin{cases} 0.24/D_* & \text{if } D_* \leq 4 \\ 0.14D_*^{-0.64} & \text{if } 4 < D_* \leq 10 \\ 0.04D_*^{-0.10} & \text{if } 10 < D_* \leq 20 \\ 0.013D_*^{0.29} & \text{if } 20 < D_* \leq 150 \\ 0.055 & \text{if } 150 < D_* \end{cases} \quad (14.11)$$

where

$$D_* = D_{50} \left(\frac{g\Delta}{\nu^2} \right)^{1/3} \quad (14.12)$$

This predictor does not contain any calibration coefficients but requires D_{50} and D_{90} data from the morphology module. It does not include the advective and relaxation behaviour that is available by explicitly simulating the dune height as described in section 14.1 combined with trachytope number 106.

The second alluvial roughness predictor proposed by (Struiksma, pers. comm.) allows for a lot of adjustments, it reads

$$\frac{1}{C^2} = (1 - \xi) \frac{1}{C_{90}^2} + \xi \frac{1}{C_{min}^2} \quad (14.13)$$

where

$$C_{90} = A_1^{10} \log(A_2 h / D_{90}) \quad (14.14)$$

and

$$\xi = \frac{\max(0, \theta_g - \theta_c)}{\theta_m - \theta_c} \frac{\theta_m^2 - \theta_c \theta_g}{(\theta_m - \theta_c) \theta_g} \quad (14.15)$$

which varies from 0 at $\theta_g \leq \theta_c$ to 1 at $\theta_g = \theta_m$ where

$$\theta_g = \frac{u^2}{C_{90}^2 \Delta D_{50}} \quad (14.16)$$

and $A_1, A_2, \theta_c, \theta_m, C_{\min}$ are coefficients that the user needs to specify. This formula requires also D_{50} and D_{90} data from the morphology module. The fifth formula is based on Van Rijn (2007) and reads

$$k = \min(\sqrt{k_{s,r}^2 + k_{s,mr}^2 + k_{s,d}^2}, \frac{h}{2}) \quad (14.17)$$

It uses the roughness heights of ripples k_r , mega-ripples k_{mr} and dunes k_d . These formulae depend on sediment properties D_{50} and D_{90} data which may be either specified as part of the roughness type or obtained from the morphology module. The sixth formula is similar, but uses a linear addition

$$k = \min(k_{s,r} + k_{s,mr} + k_{s,d}, \frac{h}{2}) \quad (14.18)$$

Four vegetation based area trachytopes have been implemented. Two formulae (referred to as ‘Barneveld’) are based on the work by Klopstra *et al.* (1996, 1997) and two on the work by Baptist (2005).

FormNr	Name	Formula
151	Barneveld 1	Eqs. (14.19) – (14.28), $C_D = 1.65$
152	Barneveld 2	Eqs. (14.19) – (14.25), (14.29) – (14.31)
153	Baptist 1	Eqs. (14.32) and (14.33)
154	Baptist 2	Eqs. (14.34), (14.36) and (14.37)

The formula by Klopstra *et al.* (1997) reads

$$C = \frac{1}{h^{3/2}} \left\{ \begin{array}{l} \frac{2}{\sqrt{2A}} \left(\sqrt{C_3 e^{h_v \sqrt{2A}} + u_{v0}^2} - \sqrt{C_3 + u_{v0}^2} \right) + \\ \frac{u_{v0}}{\sqrt{2A}} \ln \left(\frac{(\sqrt{C_3 e^{h_v \sqrt{2A}} + u_{v0}^2} - u_{v0})(\sqrt{C_3 + u_{v0}^2} + u_{v0})}{(\sqrt{C_3 e^{h_v \sqrt{2A}} + u_{v0}^2} + u_{v0})(\sqrt{C_3 + u_{v0}^2} - u_{v0})} \right) + \\ \frac{\sqrt{g(h - (h_v - a))}}{\kappa} \left((h - (h_v - a)) \ln \left(\frac{h - (h_v - a)}{z_0} \right) - a \ln \left(\frac{a}{z_0} \right) - (h - h_v) \right) \end{array} \right\} \quad (14.19)$$

where

$$A = \frac{n C_D}{2\alpha} \quad (14.20)$$

$$C_3 = \frac{2g(h - h_v)}{\alpha \sqrt{2A} (e^{h_v \sqrt{2A}} + e^{-h_v \sqrt{2A}})} \quad (14.21)$$

$$a = \frac{1 + \sqrt{1 + \frac{4E_1^2 \kappa^2 (h - h_v)}{g}}}{\frac{2E_1^2 \kappa^2}{g}} \quad (14.22)$$

and

$$z_0 = ae^{-F} \quad (14.23)$$

where

$$E_1 = \frac{\sqrt{2A} C_3 e^{h_v \sqrt{2A}}}{2 \sqrt{C_3 e^{h_v \sqrt{2A}} + u_{v0}^2}} \quad (14.24)$$

and

$$F = \frac{\kappa \sqrt{C_3 e^{h_v \sqrt{2A}} + u_{v0}^2}}{\sqrt{g(h - (h_v - a))}} \quad (14.25)$$

Here, h is the water depth, h_v is the vegetation height, and $n = mD$ where m is the number of stems per square metre and D is the stem diameter. For the first implementation the parameter α in Equation (14.21) is given by

$$\alpha = \max(0.001, 0.01 \sqrt{hh_v}) \quad (14.26)$$

and the velocity within the vegetation is approximated by $u_{v0} \sqrt{i}$ where

$$u_{v0}^2 = \frac{2g}{C_D n} \quad (14.27)$$

and i is the water level gradient. For emerged vegetation the first implementation reads

$$\frac{1}{C^2} = \frac{C_D nh}{2g} \quad (14.28)$$

The second implementation of Klopstra *et al.* (1996) is based on a modification by Van Velzen *et al.* (2003); it is identical except for the following modifications to Eqs. (14.26) – (14.28). The main difference between the two implementations is the inclusion of the roughness C_b of the bed itself (without vegetation). The parameter α in Equation (14.21) is now given by

$$\alpha = 0.0227 h_v^{0.7} \quad (14.29)$$

and the velocity within the vegetation is approximated by $u_{v0} \sqrt{i}$ where

$$u_{v0}^2 = \frac{h_v}{\frac{C_D h_v n}{2g} + \frac{1}{C_b^2}} \quad (14.30)$$

and i is the water level gradient. For emerged vegetation the second implementation reads

$$\frac{1}{C^2} = \frac{C_D nh}{2g} + \frac{1}{C_b^2} \quad (14.31)$$

For large values of C_b the latter two equations simplify to the corresponding equations of the first implementation. The first implementation requires vegetation height h_v and density n as input parameters (the drag coefficient C_D is equal to 1.65); for second implementation you'll also need to specify the drag coefficient C_D and the alluvial bed roughness k_b (C_b in Equation (14.31) is computed as $18^{10} \log(12h/k_b)$).

The first implementation of the roughness predictor by Baptist (Baptist, 2005) reads for the case of submerged vegetation

$$C = \frac{1}{\sqrt{\frac{1}{C_b^2} + \frac{C_D nh_v}{2g}}} + \frac{\sqrt{g}}{\kappa} \ln\left(\frac{h}{h_v}\right) \quad (14.32)$$

where n is the vegetation density ($n = mD$ where m is the number of stems per square metre and D is the stem diameter). The second term goes to zero at the transition from submerged to emerged vegetation. At that transition the formula changes into the formula for non-submerged vegetation which reads

$$C = \frac{1}{\sqrt{\frac{1}{C_b^2} + \frac{C_D nh}{2g}}} \quad (14.33)$$

which is identical to the non-submerged case of the second implementation of the work by Klopstra *et al.* (1996) (see Equation (14.31)).

The drawback of the three vegetation based formulations above is that they parameterize the flow resistance by means of the bed roughness. Consequently, the presence of vegetation will lead to a higher bed roughness and thus to a higher bed shear stress and larger sediment transport rates in case of morphological computations. Therefore, we have included a $-\frac{\lambda}{2}u^2$ term in the momentum equation where λ represents the flow resistance of the vegetation. For the case of non-submerged vegetation $h < h_v$ the flow resistance and bed roughness are strictly separated

$$C = C_b \quad \text{and} \quad \lambda = C_D n \quad (14.34)$$

In the case of submerged vegetation $h > h_v$ the two terms can't be split in an equally clean manner. However, we can split the terms such that the bed shear stress computed using the depth averaged velocity u and the net bed roughness C equals the bed shear stress computed using the velocity u_v within the vegetation layer and the real bed roughness C_b .

$$\frac{u^2}{C^2} = \frac{u_v^2}{C_b^2} \quad (14.35)$$

With this additional requirement we can rewrite Equation (14.32) as

$$C = C_b + \frac{\sqrt{g}}{\kappa} \ln\left(\frac{h}{h_v}\right) \sqrt{1 + \frac{C_D nh_v C_b^2}{2g}} \quad (14.36)$$

and

$$\lambda = C_D n \frac{h_v}{h} \frac{C_b^2}{C^2} \quad (14.37)$$

which simplify to Equation (14.34) for $h = h_v$. Both formulae by Baptist require vegetation height h_v , density n , drag coefficient C_D and alluvial bed roughness C_b as input parameters.

Linear trachytope classes (201–250)

Two formulae have been implemented for linear trachytopes such as hedges or bridge piers.

FormNr	Name	Formula
201	hedges 1	Eqs. (14.38) to (14.40)
202	hedges 2	Eqs. (14.41) to (14.43)

The first implementation reads

$$\frac{1}{C^2} = \frac{h}{2g} \frac{L_{hedge}}{W_{cell} L_{cell}} \frac{1 - \mu^2}{\mu^2} \quad (14.38)$$

where L_{hedge} is the projected length of the hedge, W_{cell} and L_{cell} are the width and length of the grid cell. The ratio L_{hedge}/W_{cell} may be interpreted as the number of hedges that the flow encounters per unit width. The second ratio is thus the inverse of the average distance between these hedges within the grid cell. The last term may be loosely referred to as the drag of the hedge, which is determined by the hedge pass factor μ given by

$$\mu = 1 + 0.175n \left(\frac{h}{h_v} - 2 \right) \quad (14.39)$$

if the hedge extends above the water level ($h_v > h$) and is given by

$$\mu = 1 - 0.175n \left(\frac{h}{h_v} \right) \quad (14.40)$$

if the hedge is fully submerged ($h > h_v$) where n is a dimensionless hedge density. The second implementation reads

$$\frac{1}{C^2} = \frac{C_D n L_{hedge} h}{2g L_{cell} W_{cell}} \quad (14.41)$$

or equivalently

$$C = \sqrt{\frac{2gL_{cell}W_{cell}}{hL_{hedge}}} \left(\sqrt{\frac{1}{C_D n}} \right) \quad (14.42)$$

for non-submerged conditions and

$$C = \sqrt{\frac{2gL_{cell}W_{cell}}{hL_{hedge}}} \left(\frac{h_v}{h} \sqrt{\frac{1}{C_D n}} + m_0 \sqrt{\frac{\left(\frac{h-h_v}{h}\right)^2}{1 - \left(\frac{h-h_v}{h}\right)^2}} \right) \quad (14.43)$$

for submerged conditions. We recognize the same ratio $L_{cell}W_{cell}/L_{hedge}$ that represents the average distance between hedges. Equation (14.41) can be directly compared to similar equations for area trachytopes (Equation (14.28)), point trachytopes (Equation (14.44)). Note that the formula for computing the loss coefficient for a bridge explicitly includes the reduction in the flow area and the resulting increase in the effective flow velocity, whereas the above mentioned trachytope formulae don't.

Point trachytype classes: various (251–300)

One formula for point trachytypes has been implemented. It may be used to represent groups of individual trees or on a smaller scale plants.

FormNr	Name	Formula
251	trees	Eqn. (14.44)

The implemented formula reads

$$C = \sqrt{\frac{2g}{C_D n \min(h_v, h)}} \quad (14.44)$$

where $n = mD$ with m the number of trees per unit area and D the characteristic tree diameter, h_v is the vegetation height and h is the local water depth. The formula is identical to [Equation \(14.33\)](#) except for the fact that the point trachytype formula has no bed roughness and area associated with it. The generalization of [Equation \(14.44\)](#) to the submerged case ($h > h_v$) lacks the extra term in [Equation \(14.32\)](#).

14.2.2 Averaging and accumulation of trachytypes

Point and linear roughnesses are accumulated by summing the inverse of the squared Chézy values C_i .

$$\frac{1}{C_{pnt}^2} = \sum_i \frac{1}{C_{pnt,i}^2} \quad (14.45)$$

$$\frac{1}{C_{lin}^2} = \sum_i \frac{1}{C_{lin,i}^2} \quad (14.46)$$

The area roughnesses are accumulated weighted by the surface area fraction f_i . These roughnesses are accumulated as White-Colebrook roughness values and as Chézy values; for the latter values both the linear sum (“parallel”) and the sum of inverse of squared values (“serial”) are computed. Roughness values are converted into each other as needed based on the local water depth.

$$k_{area} = \sum_i f_i k_i \quad (14.47)$$

$$\frac{1}{C_{area,s}^2} = \sum_i f_i \frac{1}{C_i^2} \quad (14.48)$$

$$C_{area,p} = \sum_i f_i C_i \quad (14.49)$$

For the fraction of the grid cell area for which no roughness class is specified the default roughness is used.

The flow resistance coefficients are also accumulated proportionally to the surface area fraction f_i associated with the trachytype considered. For the fraction of the grid cell area for which no flow resistance is specified, obviously none is used.

$$\lambda = \sum_i f_i \lambda_i \quad (14.50)$$

The final effective bed roughness of the grid cell may be computed by either one of the following two methods.

Method 1

The total mean roughness is computed by summing the White-Colebrook values for the areas and line and point resistance features.

$$k_m = k_{area} + k_{lin} + k_{pnt} \quad (14.51)$$

where $k_{lin} = 12h10^{-C_{lin}/18}$ and $k_{pnt} = 12h10^{-C_{pnt}/18}$. The effect of the water free area fraction f_b is taken into account by means of the following empirical relation in which $C_m = 18^{10}\log(12h/k_m)$ is the mean Chézy value corresponding to the total mean White-Colebrook roughness value obtained from [Equation \(14.51\)](#).

$$f_b = \max(\min(0.843, f_b), 0.014) \quad (14.52)$$

$$C_{total} = C_m \left(1.12 - 0.25f_b - 0.99\sqrt{f_b} \right) \quad (14.53)$$

The resulting C_{total} value is used in the computation. This method together with trachytope classes 1, 51, 101, 151 and 201 corresponds to the NIKURADSE option of the WAQUA/TRI-WAQ flow solver.

Method 2

The total roughness is computed by first averaging over the serial and parallel averages of the Chézy values according

$$C_{area} = \alpha_s C_{area,s} + (1 - \alpha_s) C_{area,p} \quad (14.54)$$

where $\alpha_s = 0.6$ by default. Subsequently the effect of the water free area fraction f_b is taken into account by means of the following empirical relation (identical to [Equation \(14.53\)](#) of method 1).

$$f_b = \max(\min(0.843, f_b), 0.014) \quad (14.55)$$

$$C_{area,corr} = C_{area} \left(1.12 - 0.25f_b - 0.99\sqrt{f_b} \right) \quad (14.56)$$

Finally the Chézy value representing the total bed roughness is computed by accumulating the inverses of the squared Chézy values.

$$\frac{1}{C_{total}^2} = \frac{1}{C_{area,corr}^2} + \frac{1}{C_{lin}^2} + \frac{1}{C_{pnt}^2} \quad (14.57)$$

The resulting C_{total} value is used in the computation. This method together with trachytope classes 1, 51, 52, 53, 101, 152, 202 and 251 corresponds to the ROUGHCOMBINATION option of the WAQUA/TRIWAQ flow solver.

14.3 (Rigid) three-dimensional vegetation model

The (rigid) 3D Vegetation model ([Winterwerp and Uittenbogaard \(1997\)](#)), as known from Delft3D-FLOW, is not available yet in D-Flow FM.

15 Calibration factor

The current chapter explains the effect of the calibration factor. The calibration factor is multiplier of the roughness. The calibration factor may be constant, discharge- or water-level-dependent. By assigning different areas to different calibration classes, each region can be independently calibrated. Calibration roughness definitions can also be combined by assigning multiple field definitions and a weighting for each gridcell edge (net link). This approach allows thus both abrupt and gradual transitions and the modeller can control how the calibration factor is going to be used. The approach is similar to the roughness and link definitions used in the trachytopes module for alluvial- and bedform-roughness (see [chapter 14](#)).

The calibration factor is defined by means of two files (see [section C.8](#)):

- ◊ Calibration factor definition file (CLD-file). This file defines the calibration factor (e.g. constant, water-level- or discharge dependent).
- ◊ Calibration factor area file (CLL-file). This file associates calibration factor definitions with the edges of the model grid and include a relative weighting.

The resulting weighted calibration factor is multiplied by the roughness type as expressed by the `UnifFRICTCoef` keyword in the [physics] section in the .mdu file (see [Appendix A](#)). This implies that the effect of the calibration will be different depending on the definition of `UnifFRICTCoef`. For example, if the `UnifFRICTCoef=0` (i.e. Chézy) a local calibration factor of 0.5 will imply an increase of the bed shear stress, whereas if `UnifFRICTCoef=2` (i.e. White-Colebrook) a reduction of the bed shear stress may be expected. A background calibration factor equal to one is applied if the sum of the weights at a single link is lower than one.

The calibration factor approach cannot be combined with multiple roughness types as specified through the external forcing file. This will lead to an error. Such a spatial variation in the roughness can be achieved by defining these areas through the trachytopes module.

It is good to be aware of the following known differences with the trachytopes module:

- ◊ The weighting of the calibration factors is done for all entries in the calibration area definition file. Averaging for the trachytopes area file is only done for the last sequence of trachytopes area definitions at one and the same location.
- ◊ When a calibration factor definition is imposed at a location outside of the grid, and this calibration factor definition depends on a water level station or cross-section which is also outside of the domain, the model crashes, however the trachytopes module allows this.

16 Coupling with D-Waves (SWAN)

This chapter is on the *coupling* of hydrodynamics and waves. Full documentation on D-Waves is available in its own User Manual; this chapter is limited to the details of running coupled flow-wave models, and the physical interaction processes between the two of them.

16.1 Getting started

The Delta Shell framework implements the concept of an *Integrated model* in order to couple different models, such as: hydrodynamics coupled with the controlling of structures, waves, morphology and/or water quality.

Two types of coupling are distinguished: *offline* and *online* coupling. In case of an *Integrated model* with *offline* coupling, the entire hydrodynamic simulation is done first, i.e., *separately* from the second simulation. The file-based hydrodynamic output serves as input for the second simulation. As such, the hydrodynamic flow drives the controlling of structures or the simulation of waves or water quality. In this case there is no feedback on the hydrodynamic simulation. For many applications, this is good practice.

An *online* coupling, on the other hand, exchanges data *every time* after computing a specified time interval. This tight coupling allows for direct feedback of the various processes on one another. This is crucial for controlling structures.

Note: *Offline* is also referred to as *sequential* coupling and *online* as *parallel* coupling.



With respect to waves, *offline* and *online* results into:

- 1 *offline*: First, a separate D-Waves computation is executed, resulting in a communication file (com-file) containing wave data. Then a D-Flow FM computation is executed, using the wave data from the com-file.
- 2 *online*: D-Flow FM computations are alternated with D-Waves calculations. D-Flow FM writes flow data to the com-file, D-Waves uses this flow data for the wave calculation and writes wave data to the com-file. D-Flow FM uses the updated wave data.

Both modes are started by executing DIMR with a <dimr_config.xml> file as argument. This file prescribes the mode and when the D-Flow FM computation should be paused to perform a D-Waves calculation. From within the working directory, the following run-scripts in the installation directory can be called:

- ◊ On Windows:
 <win64\scripts\run_dimr.bat>
- ◊ On Linux:
 <lnx64/scripts/run_dimr.sh>
- ◊ On Windows, a computation with D-Flow FM in parallel using MPI:
 <win64\scripts\run_dimr_parallel.bat>
- ◊ On Linux, submitting a job on the Deltares cluster (sequential and parallel):
 <lnx64/scripts/submit_dimr.sh>
 This script can be used for clusters outside Deltares too, but system specific modifications will be needed.

See the DIMR documentation for more information.

16.1.1 Input D-Flow FM

Optionally add the following lines to the .mdu file:

```
[numerics]
Epshu      = 0.05 # Input for threshold water depth for wet and dry cells

[waves]
Wavemodelnr = 3    # Wave model nr, 0=no, 1=fetch/depth limited hurdlestive,
                    2=youngverhagen, 3 = D-Waves, 4=wave group forcing
Rouwav     = FR84
Gammax     = 0.5   # Maximum wave height/water depth ratio

[output]
EulerVelocities = 1 # 0 (default): GLM, 1: Euler.
Only relevant when using waves
```

Description:

Wavemodelnr	Key switch to enable wave modelling. Use “3” for wave data from D-Waves (online or offline) and passing hydrodynamic data to D-Waves (online only). A file will be generated automatically named <”runid”_com.nc> to exchange data.
Rouwav	Necessary to include bed shear-stress enhancement by waves. See also Delft3D-FLOW manual.
EulerVelocities	Optional flag to write Eulerian velocities to the D-Flow FM map file. Currently, only Eulerian values will be written for the cell centre velocity x-component and y-component (parameters ucx and ucy). Check that the “long_name” contains the word “Eulerian”.
Epshu	Optionally overwrite the default value of “1.0e-4”. Depending on your model, the default value of Epshu in combination with modelling waves may lead to huge local velocities near (almost) dry points. This will result in very small time steps. Increasing Epshu might be a reasonable workaround.
Gammax	Optionally overwrite the default value of “1.0”. Depending on your model, the default value of Gammax may lead to huge local velocities in shallow water. Decreasing Gammax might be a reasonable workaround.

16.1.2 Input D-Waves

Optionally add the following lines to the .mdw file:

```
[Output]
MapWriteNetCDF      = true
COMFile             = ../fm/dflowfmoutput/fff_com.nc
NetCDFSinglePrecision = false
locationFile        = ../fm/loo_obs.xyn
writeTable          = true
```

Description:

MapWriteNetCDF	Default value “false”, resulting in no output written in NetCDF format. The coupling with D-Flow FM is only implemented for NetCDF format, so this flag must be set to “true” when being coupled with D-Flow FM.
COMFile	Necessary reference to the file being used to communicate data to and from D-Flow FM. The name must exactly match with the name of the com-file being generated by D-Flow FM.
NetCDFSinglePrecision	Optional flag to write data in single precision instead of double precision. Default value is “false”. Might be useful to decrease the size of the com-file or to compare with a Delft3D-FLOW computation.
locationFile	Optional reference to observation points in D-Flow FM. When in combination with the “writeTable” flag, D-Waves will produce a history file in NetCDF format for these observation points.
writeTable	Optional flag to force SWAN to produce an output file in table format for each set of locations specified in a locationFile. Default value is “false”.

16.1.3 Input dimr

Both D-Flow FM and D-Waves are used as dynamic libraries (DLL’s on Windows, so’s on Linux). DIMR is a small executable steering both dynamic libraries. Its input file, usually called “dimr_config.xml”, looks like this:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<dimrConfig xmlns="http://schemas.deltares.nl/dimrConfig"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://schemas.deltares.nl/dimrConfig
    http://content.oss.deltares.nl/schemas/d_hydro-1.00.xsd">
<documentation>
    <fileVersion>1.00</fileVersion>
    <createdBy>Deltares, Coupling team</createdBy>
    <creationDate>2015-05-20T07:56:32+01</creationDate>
</documentation>

<control>
    <parallel>
        <startGroup>
            <time>0.0 60.0 99999999.0</time>
            <coupler name="flow2rtc"/>
            <start name="myNameRTC"/>
            <coupler name="rtc2flow"/>
        </startGroup>
        <startGroup>
            <time>0.0 3600.0 99999999.0</time>
            <start name="myNameWave"/>
        </startGroup>
        <start name="myNameDFlowFM"/>
    </parallel>
</control>

<component name="myNameDFlowFM">
    <library>dflowfm</library>
    <process>0 1 2</process>
    <mpiCommunicator>DFM_COMM_DFMWORLD</mpiCommunicator>
    <workingDir>fm</workingDir>
    <inputFile>weirtimeseries.mdu</inputFile>
</component>
<component name="myNameWave">
    <library>wave</library>
    <process>0</process>
    <workingDir>wave</workingDir>
    <!-- component specific -->
```

```
<inputFile>weir.mdw</inputFile>
</component>
<component name="myNameRTC">
    <library>RTCTools_BMI</library>
    <process>0</process>
    <workingDir>rtc</workingDir>
    <!-- component specific -->
    <inputFile>.</inputFile>
</component>

<coupler name="flow2rtc">
    <sourceComponent>myNameDFlowFM</sourceComponent>
    <targetComponent>myNameRTC</targetComponent>
    <item>
        <sourceName>observations/Upstream/water_level</sourceName>
        <targetName>input_ObservationPoint01_water_level</targetName>
    </item>
</coupler>
<coupler name="rtc2flow">
    <sourceComponent>myNameRTC</sourceComponent>
    <targetComponent>myNameDFlowFM</targetComponent>
    <item>
        <sourceName>output_weir_crest_level</sourceName>
        <targetName>weirs/weir01/crest_level</targetName>
    </item>
</coupler>
</dimrConfig>
```

Description:

- <**control**> Specifies the workflow of the `deltahydro` executable. It indicates which components are started in which order. If the data transfer is to be arranged by the main program `dimr`, then a coupler should be included. The main <**control**> block is a sequential block; this means that each component is initialized, time stepped, and finalized before the next component starts. For each component/coupler listed inside the <**control**> block there will be a corresponding component/ coupler specification block defined below.
- <**parallel**> Within a <**parallel**> tag the components are started concurrently (if the mpi process ids listed per component don't overlap) or executed synchronously in sequence (first all initialize, then time stepping, and to conclude all finalization calls). The order of the components is retained.
- <**start**> A <**parallel**> block contains exactly one <**start**> component, defining the start and end time of the simulation. This is the component inside the <**parallel**> block with the smallest time step and can be denoted as the "master-component". All other components must be defined with a <**startGroup**> and can be denoted as a "slave-component".
- <**startGroup**> A <**startGroup**> should be used if a component (possibly including couplers) should only be executed at a subset of simulation time steps.
- <**time**> Start-, step- and stop-time (in seconds) at which this slave-component should be executed. The times are relative to the times of the master-component. Thus a start-time of 0.0 always refers to the start time of the master-component and a stop-time of "infinity" always refers to the end time of the master-component.
- <**component name="myComponentName"**> Component specification block. "myComponentName" is free to be defined by the user. It must match exactly with the reference in the <**control**> block above. The name of a component must be unique.
- <**library**> Reference to the component to be executed. Currently "flowfm", "wave" and "RTCTools_BMI" are supported. The name must match exactly the name of the related dll/so (excluding prefixes (e.g. "lib") and suffixes (e.g. ".dll" or ".so")).

The library should be located in the search path, or it may include an absolute or relative path. Multiple `<component>` blocks may refer to the same component to be executed.

`<process>` Optional list of the ids of the mpi processes that should be used to run the component. If not specified, then the component will run only in process “0” (i.e. non-parallel). The processes may be specified as a space separated list with series compressed using colons e.g. “16:31” represents processes “16 17 18” up to “31”.

`<mpiCommunicator>` D-Flow FM specific flag. Mandatory only when this D-Flow FM component should run a parallel (partitioned) model. Note that this is *unrelated* to the `<parallel>` tag as introduced above.

`<workingDir>` Specification of the working directory of this `<component>`, relative to the location of the “dimr_config.xml” file. The workingDir is the base/root directory of all input and output files for the component. All other files will be located RELATIVE TO this folder. If not specified, then workingDir will be equal to the folder of the configuration file.

`<inputFile>` Specification of the input file of this `<component>`, relative to `<workingDir>`:
D-Flow FM: mdu-file, D-Waves: mdw-file, D-RTC: .

`<coupler name="myCouplerName">` Coupler specification block. “myCouplerName” is free to be defined by the user. It must match exactly with the reference in the `<control>` block above. The name of a coupler must be unique.

`<sourceComponent>` Identifies which component provides the data.

`<targetComponent>` Identifies which component needs to receive the data. The coupler runs on the superset of the processes configured for the source and target components.

`<item>` For each quantity to be exchanged, the name in the source component and the name in the target component are specified. To support recursive components, a directory syntax is used with forward slashes. If a name includes a forward slash, then it needs to be escaped using a backward slash, like \; if a name includes a backward slash, then it needs to be escaped with a backward slash, like \\.

`<sourceName>` Identifies which parameter will be sent at the source component side.

`<targetName>` Identifies which parameter will be received at the target component side.

16.1.4 Online process order

When D-Flow FM runs online in alternation with D-Waves and D-RTC, process steps can be identified, where the order as prescribed in “dimr_config.xml” is respected as much as possible. The process order of the example “dimr_config.xml” file in the section above is:

1 D-Flow FM::Init

D-Flow FM is initialized. Grid- and Flow-data is written to the NetCDF com-file.

Even though D-Flow FM is not the first component in the example config-file, it must be the first component to be initialized, because the initialization results may be needed when other components are initialized. This is implemented as an exceptional rule: The master-component is always the first component to be initialized.

2 D-RTC::Init

D-RTC is initialized using (only) D-RTC input. Data from other components is not initialized yet; default values are used.

3 D-Waves::Init

D-Waves is initialized. Grid- and Flow-data created by D-Flow FM::Init is read from the com-file. Grid conversion factors are generated. The factors for the conversion from the (structured!) wave grid to the (unstructured) flow grid are calculated inside D-Waves. The factors for the conversion from the (unstructured) flow grid to the (structured) wave

grid are calculated by "ESMF_RegridWeightGen" via the execution of script <ESMF_RegridWeightGen_in_Delft3D-WAVE.bat> (Windows) or <ESMF_RegridWeightGen_in_Delft3D-WAVE.sh> (Linux).

4 D-RTC::Step

A D-RTC computation is performed at time "0.0": Data is communicated (by DIMR) from D-Flow FM to D-RTC, the actual D-RTC computation is executed, data is communicated (by DIMR) from D-RTC to D-Flow FM.

5 D-Waves::Step

A D-Waves calculation is performed at time "0.0": Flow data is read (by D-Waves) from the com-file and converted to the wave grid(s). SWAN input files are written, a SWAN calculation is started by executing script <swan.bat> (Windows) or <swan.sh> (Linux), SWAN output files are read, wave-data is converted to the flow grid and written to the com-file (by D-Waves).

6 D-Flow FM::Step

A D-Flow FM calculation is performed: The com-file is checked for wave data to be used/updated. The simulation time will proceed from "0.0" to "60.0" (next time that another component should be executed). Flow-data is written to the com-file (by D-Flow FM).

7 D-RTC::Step, D-Flow FM::Step

A D-RTC computation is performed after every "60.0" seconds. Then a D-Flow FM computation is performed and the simulation time will proceed another "60.0" seconds. This continues until the simulation time has progressed for "3600.0" seconds.

8 D-RTC::Step, D-Waves::Step, D-Flow FM::Step

A D-RTC computation is performed, then a D-Waves computation, then a D-Flow FM computation. The simulation time will proceed another "60.0" seconds and the next D-RTC computation can start.

9 D-RTC::Finish

D-RTC is finished at the end of the simulation.

10 D-Waves::Finish

D-Waves is finished at the end of the simulation.

11 D-Flow FM::Finish

D-Flow FM is finished at the end of the simulation.

16.1.5 Related files

Below is an overview of the related files in the default directory structure. In this example the runid is set to foo.

```
<testcaseroot>
  dimr_config.xml Configuration file, input for DIMR
  <fm>           Work directory for D-Flow FM
    foo.mdu       D-Flow FM input file
    <dflowfmoutput> D-Flow FM output directory
    foo_com.nc   Communication file, written/read by both D-Flow FM and D-Waves
  <rtc>           Work directory for D-RTC
    settings.json D-RTC input file (must have exactly this name)
    <xml_dir>     D-RTC directory containing xml input/output files and csv output file
    <xsd_dir>     D-RTC directory containing xsd input files
  <wave>          Work directory for D-Waves
    foo.mdw       D-Waves input file
    TMP_ESMF*_source.nc Temporary input file for ESMF_Regridder, created by D-Waves
    TMP_ESMF*_destination.nc Temporary input file for ESMF_Regridder, created
```

by D-Waves
 TMP_ESMF*_weights.nc Resulting weights file, created by ESMF_Regridder, read
 by D-Waves
 PET0.RegridWeightGen.Log ESMF_Regridder log file
 swn-diag.foo SWAN log file

16.2 Forcing by radiation stress gradients

The momentum equation in x -direction, averaged over the wave motion and expressed in GLM co-ordinates is given by (GLM: Generalized Lagrangian Mean):

$$\frac{\partial \bar{u}_j^L}{\partial t} + \bar{u}_i^L \frac{\partial \bar{u}_j^L}{\partial x_i} + \dots + g \frac{\partial \bar{\zeta}}{\partial x_j} - \frac{1}{\rho} \frac{\partial \bar{\tau}_{ij}^L}{\partial x_i} = F_j^L, \quad (16.1)$$

where for i and j the summation rule applies, $i, j = \{1, 2, 3\}$. As shown by Groeneweg (1999) the right-hand side of Equation (16.1) contains a term related to a Stokes correction of the shear stresses. In the current implementation this term is neglected.

The wave-induced force, i.e. the right-hand side of Equation (16.1), can be expressed in the wave parameters of the wave model that is being applied. For linear current refraction the expression can be derived analytically. To account for wave dissipation due to for instance bottom friction, wave breaking and whitecapping and wave growth due to wind one can rely on mild slope formulations with dissipation terms.

As shown by Dingemans *et al.* (1987), using the gradients of the radiation stresses in numerical models can result in spurious currents. Dingemans *et al.* (1987) showed that the divergence free part of the radiation stress is not capable of driving currents and can therefore be neglected if one is primarily interested in wave-driven currents. The remaining part of the radiation stress gradients is closely related to the wave energy dissipation, i.e. the right-hand side of Equation (16.1) can be written as:

$$F_i = \frac{Dk_i}{\omega}, \quad (16.2)$$

where D is the total energy dissipation due to waves, k_i is the wave number in i -direction and ω is the wave frequency; see Dingemans (1997) for many details and discussions on this subject.

2D implementation

For a depth averaged model the momentum equations in x - and y -direction, leaving out most of the terms, can be written as:

$$\frac{\partial U}{\partial t} + \dots + \frac{gU\sqrt{U^2 + V^2}}{C_{2D}^2 h} + \dots = \dots + F_x, \quad (16.3)$$

$$\frac{\partial V}{\partial t} + \dots + \frac{gV\sqrt{U^2 + V^2}}{C_{2D}^2 h} + \dots = \dots + F_y, \quad (16.4)$$

where F_x and F_y are the depth averaged wave-induced forcings and given by the gradients of the radiation stress tensor S , or following Dingemans *et al.* (1987) approximated by wave energy dissipation:

$$F_x = -\frac{\partial S_{xx}}{\partial x} - \frac{\partial S_{yx}}{\partial y} = D \frac{k_x}{\omega}, \quad (16.5)$$

$$F_y = -\frac{\partial S_{xy}}{\partial x} - \frac{\partial S_{yy}}{\partial y} = D \frac{k_y}{\omega}. \quad (16.6)$$

The dissipation rate D (a negative quantity) is computed by the wave model and read from the communication file. In SWAN, the dissipation rate may be computed from the bottom friction (orbital motion), depth-induced breaking and whitecapping.

You can choose to apply the radiation stress or the dissipation rate to determine the wave-induced forces.

3D implementation

There is no coupling available yet for 3D D-Flow FM models and D-Waves. The effect of the divergence free part of the radiation stresses is (the remaining part of the radiation stress gradients, after the wave dissipation has been subtracted) is added to the momentum equations in D-Flow FM (effect spread over the water column). This provides most adequate results, e.g. in situations with undertow.

16.3 Stokes drift and mass flux

In surface waves, fluid particles describe an orbital motion. The net horizontal displacement for a fluid particle is not zero. This wave induced drift velocity, the so-called Stokes-drift, is always in the direction of wave propagation. A particle at the top of the orbit beneath a wave crest moves slightly faster in the forward direction than it does in the backward direction beneath a wave trough. The mean drift velocity is a second order quantity in the wave height. The drift leads to additional fluxes in the wave averaged mass continuity equation.

The wave-induced mass fluxes M_x^S and M_y^S are found by integration of the components of the Stokes drift u^S and v^S over the wave-averaged total water depth:

$$M_x^S = \int_0^{\bar{h}} \rho_o u^S dz = \frac{E}{\omega} k_x \quad (16.7)$$

$$M_y^S = \int_0^{\bar{h}} \rho_0 v^S dz = \frac{E}{\omega} k_y \quad (16.8)$$

with E the wave energy defined as:

$$E = \frac{1}{8} \rho_0 g H_{rms}^2. \quad (16.9)$$

The mass fluxes M_x^S and M_y^S are computed by an interface program and are written to the communication file.



Remarks:

- ◊ The velocities written to the communication file for use in D-Waves, and D-Water Quality are based on the *total flux* velocities.
- ◊ The Eulerian velocities, which may be used in comparisons with measurements at a fixed location, are written to the hydrodynamic map and history files.

2D implementation

The depth-averaged Stokes drift is given by:

$$U^S = \frac{M_x^S}{\rho_0 h}, \quad (16.10)$$

$$V^S = \frac{M_y^S}{\rho_0 h}. \quad (16.11)$$

3D implementation

There is no coupling available yet for 3D D-Flow FM models and D-Waves.

16.4 Streaming

Streaming is a 3D feature. There is still no coupling available for 3D D-Flow FM models and D-Waves.

16.5 Enhancement of the bed shear-stress by waves

The boundary layers at the bed associated with the waves and the current interact non-linearly. This has the effect of enhancing both the mean and oscillatory bed shear-stresses. In addition the current profile is modified, because the extra turbulence generated close to the bed by the waves appears to the current as being equivalent to an enhanced bottom roughness. The bed shear-stress due to the combination of waves and current is enhanced beyond the value which would result from a linear addition of the bed shear-stress due to waves, τ_w , and the bed shear-stress due to current τ_c . For sediment transport modelling it is important to predict the maximum bed shear-stress, τ_{\max} , while the current velocity and the turbulent diffusion are determined by the combined wave-current bed shear-stress τ_m .

Various, often very complex, methods exist to describe the bottom boundary layer under combined current and wave action and the resulting virtual roughness. [Soulsby et al. \(1993a\)](#) developed a parameterisation of these methods allowing a simple implementation and comparison of various wave-current interaction models: [Fredsøe \(1984\)](#); [Myrhaug and Slaattelid \(1990\)](#); [Grant and Madsen \(1979\)](#); [Huynh-Thanh and Temperville \(1991\)](#); [Davies et al. \(1988\)](#); [Bijker \(1967\)](#); [Christoffersen and Jonsson \(1985\)](#); [O' Connor and Yoo \(1988\)](#); [Van Rijn et al. \(2004\)](#). All these methods have all been implemented in D-Flow FM and can be applied in 2D and 3D modelling. However, as there are minor, but specific differences in determining certain quantities, such as determining the shear-stress at the bottom, we prefer to discuss the 2D and 3D implementation separately.

2D implementation

Following [Soulsby et al. \(1993b\)](#), Figure 16.1 gives a schematic overview of the bed shear-stresses for wave current interaction.

[Soulsby et al. \(1993b\)](#) fitted one standard formula to all of the models, each model having its own fitting coefficients. The parameterisation of Soulsby for the time-mean bed shear-stress is of the form:

$$|\tau_m| = Y (|\tau_c| + |\tau_w|), \quad (16.12)$$

with

$$Y = X \{1 + bX^p(1 - X)^q\}, \quad (16.13)$$

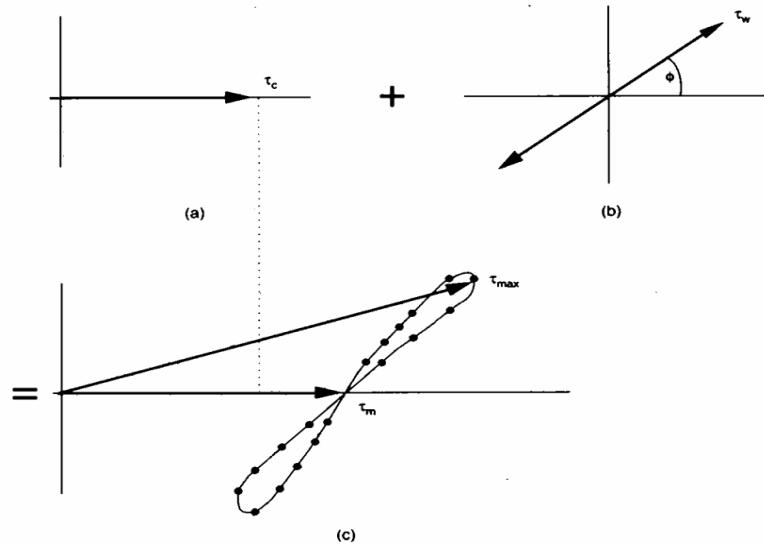


Figure 16.1: Schematic view of non-linear interaction of wave and current bed shear-stresses (from Soulsby et al. (1993b, Figure 16, p. 89))

and for the maximum bed shear-stress:

$$|\tau_{\max}| = Z (|\tau_c| + |\tau_w|), \quad (16.14)$$

with

$$Z = 1 + aX^m(1 - X)^n. \quad (16.15)$$

and:

$$X = \frac{|\tau_c|}{|\tau_c| + |\tau_w|}, \quad (16.16)$$

The value of the parameters a, b, p, q, m and n depends on the friction model which is parameterised, and:

- $|\tau_c|$ magnitude of the bed stress due to current alone
- $|\tau_w|$ magnitude of the bed stress for waves alone
- $|\tau_m|$ magnitude of the mean bed stress for combined waves and current
- $|\tau_{\max}|$ magnitude of the maximum bed stress for combined waves and current.



Remark:

- ◊ The stresses τ_m and τ_{\max} are assumed to have the same direction as τ_c .

Following Soulsby et al. (1993b) the expressions for the parameters $\chi (= a, b, p, q, m, n)$ and $\mathcal{J} (= I, J$; also depending on the friction model) have the form:

$$\chi = \left(\chi_1 + \chi_2 |\cos \phi|^{\mathcal{J}} \right) + \left(\chi_3 + \chi_4 |\cos \phi|^{\mathcal{J}} \right)^{10} \log \left(\frac{f_w}{C_{2D}} \right), \quad (16.17)$$

in which:

- C_{2D} drag coefficient due to current
- f_w wave friction factor
- ϕ the angle between the current direction and the direction of wave propagation.

Table 16.1: Fitting coefficients for wave/current boundary layer model

Model ¹	a_1	a_2	a_3	a_4	m_1	m_2	m_3	m_4	n_1	n_2	n_3	n_4	I
FR84	-0.06	1.70	-0.29	0.29	0.67	-0.29	0.09	0.42	0.75	-0.27	0.11	-0.02	0.80
MS90	-0.01	1.84	-0.58	-0.22	0.63	-0.09	0.23	-0.02	0.82	-0.30	0.19	-0.21	0.67
HT91	-0.07	1.87	-0.34	-0.12	0.72	-0.33	0.08	0.34	0.78	-0.23	0.12	-0.12	0.82
GM79	0.11	1.95	-0.49	-0.28	0.65	-0.22	0.15	0.06	0.71	-0.19	0.17	-0.15	0.67
DS88	0.05	1.62	-0.38	0.25	1.05	-0.75	-0.08	0.59	0.66	-0.25	0.19	-0.03	0.82
BK67	0.00	2.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00	0.50	0.00	0.00	1.00
CJ85	-0.01	1.58	-0.52	0.09	0.65	-0.17	0.18	0.05	0.47	-0.03	0.59	-0.50	0.64
OY88	-0.45	2.24	0.16	-0.09	0.71	0.27	-0.15	0.03	1.19	-0.66	-0.13	0.12	0.77
	b_1	b_2	b_3	b_4	p_1	p_2	p_3	p_4	q_1	q_2	q_3	q_4	J
FR84	0.29	0.55	-0.10	-0.14	-0.77	0.10	0.27	0.14	0.91	0.25	0.50	0.45	3.00
MS90	0.65	0.29	-0.30	-0.21	-0.60	0.10	0.27	-0.06	1.19	-0.68	0.22	-0.21	0.50
HT91	0.27	0.51	-0.10	-0.24	-0.75	0.13	0.12	0.02	0.89	0.40	0.50	-0.28	2.70
GM79	0.73	0.40	-0.23	-0.24	-0.68	0.13	0.24	-0.07	1.04	-0.56	0.34	-0.27	0.50
DS88	0.22	0.73	-0.05	-0.35	-0.86	0.26	0.34	-0.07	-0.89	2.33	2.60	-2.50	2.70
BK67	0.32	0.55	0.00	0.00	-0.63	0.05	0.00	0.00	1.14	0.18	0.00	0.00	3.00
CJ85	0.47	0.29	-0.09	-0.12	-0.70	0.13	0.28	-0.04	1.65	-1.19	-0.42	0.49	0.60
OY88	-0.06	0.26	0.08	-0.03	-1.00	0.31	0.25	-0.26	0.38	1.19	0.25	-0.66	1.50
VR04	$Y = 0.0$ and $Z = 1.0$												

¹ FR84=Fredsoe (1984), MS90=Myrhaug and Slaattelid (1990),
 HT91=Huynh-Thanh and Temperville (1991), GM79=Grant and Madsen (1979), DS88=Davies *et al.* (1988),
 BK67=Bijker (1967), CJ85=Christoffersen and Jonsson (1985), OY88=O' Connor and Yoo (1988),
 VR04=Van Rijn *et al.* (2004)

As the radiation stress is always in the wave direction, we can derive ϕ from:

$$|\cos \phi| = \frac{|UF_x + VF_y|}{|\mathbf{U}| |\mathbf{F}|}. \quad (16.18)$$

Values of the parameters a, b, p, q and J in Equation (16.17) have been optimised by Soulsby *et al.* (1993b), see Table 16.1 and Figure 16.2.

The bed shear-stress due to flow alone may be computed using various types of formulations like Chézy, Manning or White-Colebrook. The bed shear-stress due to current alone can be written in the form:

$$\tau_c = \frac{g \rho_0 \mathbf{U} |\mathbf{U}|}{C_{2D}^2}. \quad (16.19)$$

The magnitude of the wave-averaged bed shear-stress due to waves alone is related to the wave orbital velocity near the bottom \mathbf{u}_{orb} and the friction coefficient f_w :

$$|\tau_w| = \frac{1}{2} \rho_0 f_w u_{orb}^2. \quad (16.20)$$

The orbital velocity is computed from the linear wave theory and is given by:

$$u_{orb} = \frac{1}{4} \sqrt{\pi} \frac{H_{rms} \omega}{\sinh(kH)}, \quad (16.21)$$

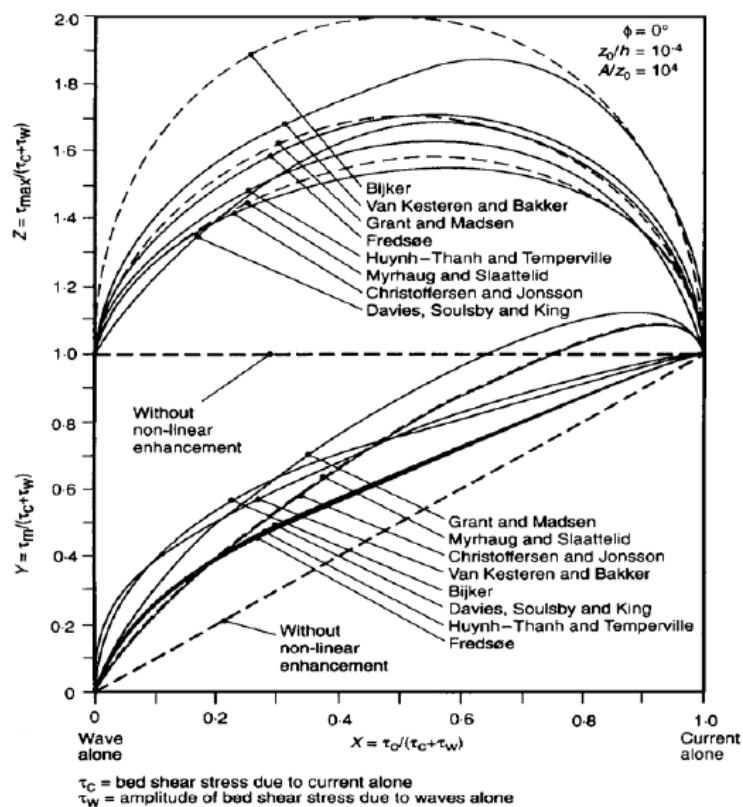


Figure 16.2: Inter-comparison of eight models for prediction of mean and maximum bed shear-stress due to waves and currents (from Soulsby et al. (1993b, Figure 17, p. 90))

where the root-mean-square wave height H_{rms} and the wave period $T(= 2\pi/\omega)$ are read from the communication file. The variation of the wave friction factor with relative orbital excursion at the bed under purely oscillatory flow is given by [Swart \(1974\)](#)

$$f_w = \begin{cases} 0.00251 \exp \left[5.21 \left(\frac{A}{k_s} \right)^{-0.19} \right], & \frac{A}{k_s} > \frac{\pi}{2}, \\ 0.3, & \frac{A}{k_s} \leq \frac{\pi}{2}, \end{cases} \quad (16.22)$$

with:

$$A = \frac{u_{orb}}{\omega}, \quad (16.23)$$

k_s is the Nikuradse roughness length-scale and ω is the wave angular frequency.

As the bed is in rest and the equations are formulated in GLM co-ordinates, we must correct the bed shear-stress used in the momentum equations for the Stokes drift. The total or effective bed shear stress is given by:

$$\tau_b = \frac{|\boldsymbol{\tau}_m|}{|\boldsymbol{U}|} (\boldsymbol{U} - \boldsymbol{U}^S), \quad (16.24)$$

where the components of the depth-averaged Stokes drift \boldsymbol{U}^S are given by Eqs. (16.10) and (16.11).

3D implementation

There is no coupling available yet for 3D D-Flow FM models and D-Waves.

17 Coupling with D-RTC (FBC-Tools)

This chapter is on the *coupling* of hydrodynamics and real-time control—more specific: feedback control—of hydraulic structures.

17.1 Introduction

The Delta Shell framework implements the concept of an *Integrated model* in order to couple different models, such as: hydrodynamics coupled with the controlling of structures, waves, morphology and/or water quality.

Two types of coupling are distinguished: *offline* and *online* coupling. In case of an *Integrated model* with *offline* coupling, the entire hydrodynamic simulation is done first, i.e., *separately* from the second simulation. The file-based hydrodynamic output serves as input for the second simulation. As such, the hydrodynamic flow drives the controlling of structures or the simulation of waves or water quality. In this case there is no feedback on the hydrodynamic simulation. For many applications, this is good practice.

An *online* coupling, on the other hand, exchanges data *every time* after computing a specified time interval. This tight coupling allows for direct feedback of the various processes on one another. This is crucial for controlling structures.

Note: *Offline* is also referred to as *sequential* coupling and *online* as *parallel* coupling.



A coupled flow-rtc model can be run either as an Integrated Model from within Delta Shell, or from the commandline using the dimr program (Deltares Integrated Model Runner).

In case of a flow-rtc coupling

- ◊ the flow model will exchange for example *Water levels*
- ◊ subsequently, the rtc model will prescribe for example the *Crest level* of a controlled structure, based on the exchanged *water levels*
- ◊ in case of an *online* coupling this *Crest level* will influence the hydrodynamic simulation.

17.2 Getting started

17.2.1 User interface: the first steps

First, the user must add a *real-time control* model to the *flow-fm* model. The **Project** window will look like this.

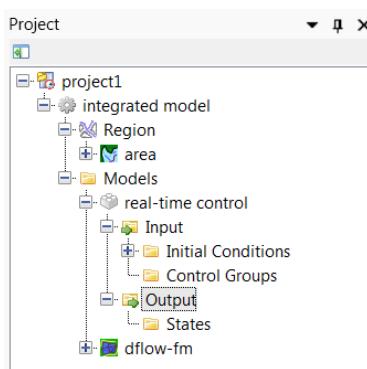


Figure 17.1: An Integrated model in the **Project** window

Secondly, the user will model the control flow for an hydraulic structure. This may look like this.

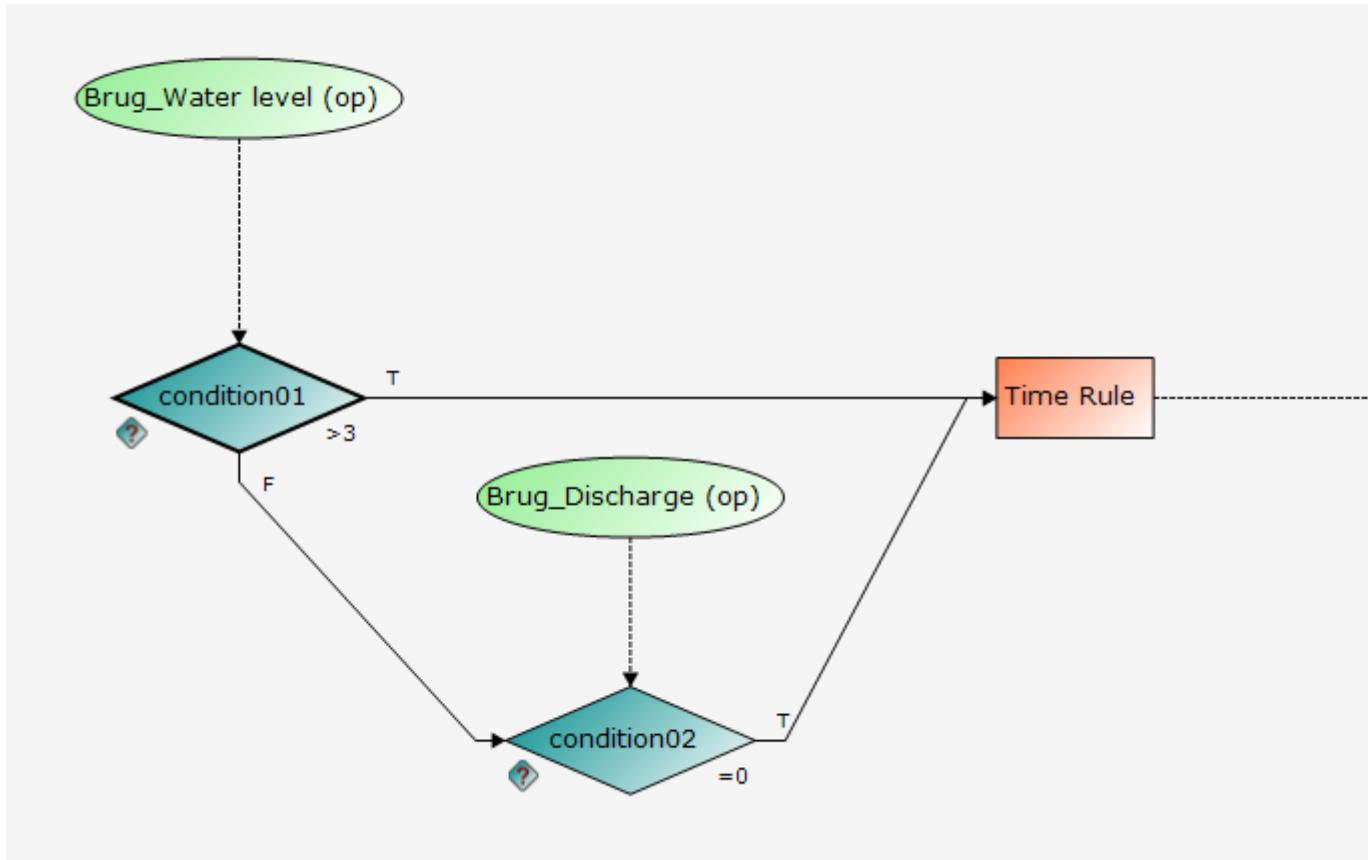


Figure 17.2: Example of a Control flow in D-RTC

Full documentation on D-RTC is available in its own User Manual. This chapter is limited to the details of running coupled flow-rtc models.

17.2.2 Input D-Flow FM

The hydraulic structures that are normally driven by scalar values or time series in <*.tim> or <*.bc> files, will now be fed by D-RTC. Replace the timeseries file name by the `realtime` keyword in the <structures.ini> file (section C.11):

```

[Structure]
type      = weir          # Type of structure
id       = weir01         # Name of the structure
polylinefile = weir01.pli # *.pli; Polyline geometry definition for 2D structure
crestLevel = realtime     # Crest level in [m]
  
```

Also, the MDU file may optionally contain an observation file and/or a cross section file, such that D-RTC's triggers can be set to respond to the computed values at these locations.

```

[output]
  
```

```

ObsFile      = obs.xyn
CrsFile      = river_crs.pli

```

17.2.3 Input D-RTC

The input of D-RTC consists of several `<*.xml>` files and a toplevel `<settings.json>`. We refer to the D-RTC User Manual ([Deltares, 2019](#)) for further details.

17.2.4 Input dimr

In [Section 16.1.3](#), details on a coupled flow-rtc-wave model are presented. This can serve as an example for a flow-rtc coupling, by leaving out all wave-related fragments.

17.2.5 Online process order

This is discussed in [Section 16.1.4](#) for a coupled flow-rtc-wave model. This can serve as an example for a flow-rtc coupling, by leaving out all wave-related fragments.

17.3 Technical background

The D-Flow FM kernel shared library offers an API that implements the Basic Model Interface (BMI). The BMI allows one to initialize, run, and finalize a model component via standardized interface calls; furthermore it allows one to get and set various quantities such that model components can dynamically react to the computed results. General details on the BMI can be found in [Peckham et al. \(2013\)](#). This section discusses some of the D-Flow FM specifics.

17.3.1 BMI interface to interact with the D-Flow FM model state

Many variables in D-Flow FM's model state can be inquired and changed via BMI calls. Setting a variable can be done via a call to `set_var(varname, userdata)`, which copies the data from the `userdata` array to the D-Flow FM memory associated with the `varname` variable. Alternatively, one may use a call to `get_var(varname, xptr)`, which returns a pointer to the requested variable, and directly assign new values to the D-Flow FM memory from outside the running simulation. The latter approach may be faster because less data needs to be copied.

17.3.2 DIMR and the BMI

The DIMR program takes care of the exchange of data between components, and each component will be addressed using BMI calls. The data to be exchanged is defined in the `<dimr.xml>` configuration file, under the `<dimrConfig><coupler><item>` elements. More details on DIMR can be found in [?](#).

The variable names specified there need to be supported by the addressed component. For the coupling to D-RTC typically scalar quantities need to be exchanged that are defined at specific locations (such as observation points or structures). Since BMI only identifies variables by a unique variable name without any location selection, D-Flow FM identifies both location and quantity by introducing *compound* variable names (not to be confused with a compound structure). A compound variable name takes the form:

```
varname = "groupname/locationid/fieldname"
```

For example the variable name "weirs/Lith01/crestLevel" will select the crest level at the weir named "Lith01".

The `groupname` part of the variable name identifies the location or object type. Currently known group names are listed below. Unless noted otherwise, please check [Section 17.3.3](#) for the list of quantities supported for those groups of locations.

- ◊ pumps
- ◊ weirs
- ◊ gates
- ◊ generalstructures
- ◊ sourcesinks
- ◊ observations (observation points, see [Section 17.3.4](#))
- ◊ crosssections (observation cross sections, see [Section 17.3.4](#))
- ◊ laterals (see [Section 17.3.5](#))

The `locationid` part of the variable name is the unique identifier of the intended object. More details on the specific identifier used can be found in the subsequent sections on the various object groups. The `fieldname` identifies the quantity at the selected location. Which quantities are supported depends on the `groupname`.

17.3.3 BMI access to hydraulic structure data in D-Flow FM

To access quantities defined at structures use a compound variable name as explained in [Section 17.3.2](#). The `locationid` part of the compound variable name is, for hydraulic structures, equal to the `id` key in the structure input file (??). Which quantities can be addressed via the BMI, depends on the structure type indicated by the `groupname`. [Table 17.1](#) lists the supported field names for each hydraulic structure group.

Table 17.1: Supported compound variable fields for hydraulic structures in D-Flow FM via BMI.

Group name	Field name	Access	Remark
pumps	capacity	read-write	Only for non-staged pumps. Prescribed capacity, not necessarily actual discharge.
weirs	crestLevel	read-write	
gates	CrestLevel	read-write	
	GateHeight	read-write	
	GateLowerEdgeLevel	read-write	
	GateOpeningWidth	read-write	
generalstructures	crestLevel	read-write	
	gateHeight	read-write	
	gateLowerEdgeLevel	read-write	
	gateOpeningWidth	read-write	
sourcesinks	discharge	read-write	Prescribed discharge, not necessarily actual discharge.
	change_in_salinity	read-write	Prescribed change, not necessarily actual change.
	change_in_temperature	read-write	Prescribed change, not necessarily actual change.

17.3.4 BMI access to observations in D-Flow FM

To access quantities defined at observation points and observation cross sections use a compound variable name as explained in [Section 17.3.2](#). The `locationid` for observation points should match the `name` specified in the relevant input file (See [Section F.2.2](#)). All observation point names must be unique when accessing the data via BMI. The same holds for the `locationid` for observation cross sections (See [Section F.2.4](#)).

[Table 17.2](#) lists the supported field names for each observation group. All supported variables/fields are read-only, evidently, as observations are only 'observing' the flow. Typically, these variables are used as input to hydraulic triggers in an RTC-coupling.

Table 17.2: Supported compound variable fields for observations in D-Flow FM via BMI.

Group name	Field name	Access	Remark
observations			Observation points
	water_level	read-only	
	water_depth	read-only	
	salinity	read-only	Only when salinity transport is on.
	temperature	read-only	Only when temperature transport is on.
	<tracername>	read-only	Only when that named tracer is in the model.
crosssections			Observation cross sections
	discharge	read-only	Space-integrated along cross section.
	velocity	read-only	Space-averaged along cross section.
	water_level	read-only	Space-averaged along cross section.
	water_depth	read-only	Space-averaged along cross section.

17.3.5 BMI access to forcings in D-Flow FM

To access quantities defined at local forcings (currently only lateral discharges) use a compound variable name as explained in [Section 17.3.2](#). The `locationid` for lateral discharges should match the `id` specified for them in the external forcings input (See [??](#)). [Table 17.3](#) lists the supported field names for each forcing group.

Table 17.3: Supported compound variable fields for forcings in D-Flow FM via BMI.

Group name	Field name	Access	Remark
laterals	water_discharge	read-write	Prescribed discharge, not necessarily actual discharge (when negative).

18 Coupling with D-Water Quality (Delwaq)

18.1 Introduction

D-Water Quality is a multi-dimensional water quality model framework developed by Deltares over the past decades. It solves the advection-diffusion-reaction equation on a predefined computational grid and for a wide range of model substances. D-Water Quality offers flexible configuration of the substances to be included, as well as the processes to be evaluated. D-Water Quality is not a hydrodynamic model, so information on flow fields is obtained from hydraulic models such as D-Flow 1D (SOBEK 3) and D-Flow FM.

Here, only the *coupling* is discussed. Full documentation on D-Water Quality is available.



18.2 Offline versus online coupling

parallel/sequential coupling, which is shared

The Delta Shell framework implements the concept of an *Integrated model* in order to couple different models, such as: hydrodynamics coupled with the controlling of structures, waves, morphology and/or water quality.

Two types of coupling are distinguished: *offline* and *online* coupling. In case of an *Integrated model* with *offline* coupling, the entire hydrodynamic simulation is done first, i.e., *separately* from the second simulation. The file-based hydrodynamic output serves as input for the second simulation. As such, the hydrodynamic flow drives the controlling of structures or the simulation of waves or water quality. In this case there is no feedback on the hydrodynamic simulation. For many applications, this is good practice.

An *online* coupling, on the other hand, exchanges data *every time* after computing a specified time interval. This tight coupling allows for direct feedback of the various processes on one another. This is crucial for controlling structures.

Note: *Offline* is also referred to as *sequential* coupling and *online* as *parallel* coupling.



The *online* coupling is scheduled for a future release.

18.3 Creating output for D-Water Quality

Creating output for D-Water Quality by D-Flow FM can be enabled in Delta Shell in the main window under the tab 'Output Parameters'. Change the "WAQ output interval" to a nonzero value. The MDU-equivalent is the keyword [output], WaqInterval, where three numbers can be put and they are in the order of "WAQinterval, WAQ output-start-time, WAQ output-end-time". Each number should be a whole number of seconds, and must be a multiple of the "User time step" (see tab "Time frame"). Moreover, if the start and end output time is not explicitly specified, they are automatically set to start and end time of the simulation, respectively.

D-Flow FM will create a special output folder named <DFM_DELWAQ_mdu_name>. In this folder a <.hyd>-file will be created that gives an overview of the coupling with references to the files containing the hydrodynamic exchange data.

Load the <.hyd>-file in the D-Water Quality GUI to prepare the input for a water quality calculation. For further information on how to run D-Water Quality please consult its user manual.

The coupling between D-Flow FM and D-Water Quality has currently been tested with good results for 2D and 3D (sigma-layer and z-layer) meshes.

The D-Water Quality GUI fully supports coupling with results from D-Flow FM. Postprocessing can be done in Delft3D-QUICKPLOT.

18.4 Current limitations

When using domain decomposition, D-Flow FM will create a special output folder per domain named <DFM_DELWAQ_mdu_name_nnnn>. Since D-Water Quality can accept results for a single domain only, you have to merge the results into one domain using a tool called ddcouplefm, which can be obtained by contacting support.

It can be useful to model the water quality processes on a coarser mesh than the hydrodynamic grid, and for this grid aggregation would be usefull. This can be done using the tools DIDO and agrhyd, which can be obtained by contacting support.

19 Morphology

Morphology is (starting with 2019.01) in a separate User Manual ([D-Morphology UM, 2019](#))

20 Water Quality Processes

Note: The implementation of water quality processes is currently a β -functionality of D-Flow FM.



20.1 Introduction

Beside using D-Flow FM hydrodynamics in a file base coupling for modeling water quality processes with D-Water Quality as described in [Chapter 18: Coupling with D-Water Quality \(Delwaq\)](#), it is also possible to use the processes in the D-Water Quality process library directly in D-Flow FM.

The file based coupling has several advantages. Hydrodynamic calculations can be time consuming, and one hydrodynamic data set can be used to run multiple water quality scenarios. However with the models becoming bigger and finer, the size of the coupling files is increasing. Also since D-Water Quality is only partly parallelised, cannot make use of MPI parallelism, and CPU frequencies aren't increasing any more for years, Delwaq runs have become very time consuming.

This was the reason that the possibility to use the processes of the D-Water Quality process library directly in D-Flow FM was developed. The coupling is implemented in such a way that only the processes from D-Water Quality are used, using the exact same code as used in D-Water Quality. For each substance that is define in the water quality processes setup, a tracer is defined. The advection-diffusion equations are solved by D-Flow FM as described in [Chapter 9: Transport of matter](#), which can make use of MPI parallelism. The processes them self do not need to make use of MPI parallelism, since the processes do not interact horizontally, but only in the vertical direction (e.g. sedimentation, light climate).

This approach seems to be useful for models with a high number of flow elements and/or layers, since this will avoid large hydrodynamic communications files. It also allows the use of MPI parallelism for the calculation of transport, and distribution of the water quality process calculations over the MPI nodes, which is possible without any MPI communication. The trade-off is of course that you have to recalculate the hydrodynamics for every run, which can be time consuming. In [Table 20.1](#) you can see an overview of some of the pro's and con's of both approaches.

Table 20.1: File base D-Water Quality versus D-Water Quality process in D-Flow FM.

Model type	Pro	Con
File based	<ul style="list-style-type: none">- run hydrodynamics once- simple models could be faster- aggregation is possible	<ul style="list-style-type: none">- big coupling files- complex models could be slower
Integrated	<ul style="list-style-type: none">- no coupling files- no coupling mistakes- use D-Flow FM's MPI capability	<ul style="list-style-type: none">- rerun hydrodynamics every scenario- no aggregation possible- can't reuse old water quality setups

Water quality modelling with D-Water Quality within D-Flow FM is very flexible. Substances, water quality processes, output variables, etc. are all free for you to choose from the library, whcih can also be extended by new processes.

Since processes in D-Flow FM are a new development, currently there is no user interface support (except the PLCT to create sub-files). The user has to manual edit the mdu-file and ext-file to set up the processes and their inputs.

The basic steps in water quality modelling within D-Flow FM are:

- 1 Set up a hydrodynamic simulation and make it suitable your water quality simulation.
- 2 Define the substances and water quality processes you want to include using the PLCT.
- 3 Reference to the sub-file you have created in the mdu-file, and optionally to an additional history output file.
- 4 Choose a sensible processes time step (DtProcesses) and mass balance output interval (DtMassBalance) in the mdu-file.
- 5 Define initial conditions, boundary conditions for your substances in the ext-file.
- 6 Define constant and spatial or temporal varying process parameters in the ext-file.
- 7 Define mass balance areas in the ext-file.
- 8 Run the simulation.
- 9 Check the output.

More information on this will follow in the next sections.

20.2 Definition of the water quality system

A water quality system is defined by its substances and the processes that influence these substances. These processes can range from simple decay to complex interactions between the substances.

The user can use all substances and processes that are available in D-Water Quality. A comprehensive description of the formulations and the input and output items of the processes can be found in Technical Reference Manual ([D-WAQ TRM, 2013](#)).

Processes can calculate actual fluxes between substances, but there are also 'informational' processes that e.g. calculate the light climate or the total bottom shear stress or the total resuspension flux. A setup (sub-file) can be created using the PLCT. The use of the Processes Library Configuration Tool (PLCT) is discussed in the User Manual for the [D-WAQ PLCT UM \(2015\)](#). For D-Water Quality, this sub-file is read by the user interface, and the settings are incorporated in the D-Water Quality input file. D-Flow FM reads the sub-file directly.

The substance file is in plain text format, which allows manipulation by the user. The substance file contains:

- ◊ a list of substances that are transported (active)
- ◊ a list of substances that are not transported (inactive)
- ◊ a list of constants and their values (see: [Section 20.3.1](#))
- ◊ a list of list of outputs (see: ref below)
- ◊ a list of the processes that will be active

To use a sub-file, it has to be mentioned in the mdu-file. Use the MDU keyword [processes], SubstanceFile.

Example of the reference to a sub-file in the mdu-file which will activate the processes in D-Flow FM:

```
[processes]
SubstanceFile = sed_3fraction.sub
```

20.2.1 Substances

There are two types of substances in D-Water Quality: active and inactive. Inactive substances usually reside only in the bottom water layer. They usually represent inorganic sediments or organic material at the bottom, or rooting vegetation.

Active substances are added to the constituents of D-Flow FM. You can define their initial conditions (or use the result from a previous run using a restart file), boundary conditions and sink-source concentrations in the same ways as for tracers in D-Flow FM.

The non-transported substances are not part of constituents in D-Flow FM. You can define their initial conditions using a special keyword in the ext-file, or use the result from a previous run using a restart file. You cannot define boundary conditions for them, since they are not transported over boundaries. Also, you do not need to add concentrations for them when you use sources and sinks, only for the transported substances.

An actual list of the constituents in the model and their order is written to the diagnostics file.

Format for a substance definition in the sub-file:

```
substance <substance name> <active/inactive>
    description      <description>
    concentration-unit <unit>
    waste-load-unit   <waste load unit>
end-substance
```

Excerpt of the substance definition in a sub-file:

```
substance 'IM1' active
    description      'inorganic matter (IM1)'
    concentration-unit '(gDM/m3)'
    waste-load-unit  '-'
end-substance
substance 'IM1S1' inactive
    description      'IM1 in layer S1'
    concentration-unit '(gDM/m2)'
    waste-load-unit  '-'
end-substance
```

20.2.2 Processes

The processes in the D-Water Quality processes library cover a large number of water quality problems. An overview can be seen in [Figure 20.1](#). The user can select which processes are relevant and switched on. D-Flow FM will evaluate if all required input is available to actually calculate the processes. A report of this will be available in the lsp-report file.

All processes from D-Water Quality can be used in D-Flow FM. Processes are evaluated at larger time steps than flow and transport using fractional stepping. A time step for the evaluation of processes must be set in the mdu-file. Each DtProcesses, the processes are evaluated, which leads to fluxes. The effect is directly applied to the concentration field, after which the transport continues without any further interaction between the substances until the next water quality time step.

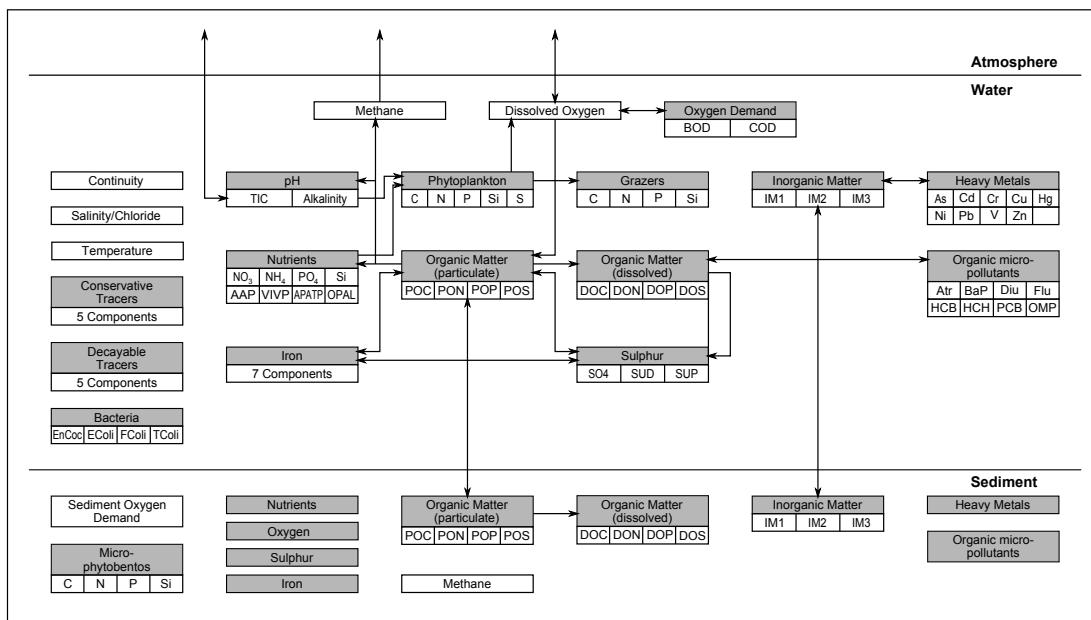


Figure 20.1: General overview of substances included in D-Water Quality. Substances are organised in functional groups indicated by a grey header, except for some substances that form a group of their own. Major links between substances are indicated by arrows; note that many links are omitted.

Limitation: DtProcesses must be a multiple of the dtuser, and can be specified in the processes section of the mdu-file using the MDU keyword [processes], DtProcesses. A list of the processes to be switched on is given in the sub-file.

Example of a processes time step definition in the mdu-file:

```
[processes]
DtProcesses      =      600.0
```

Format for a processes list in the sub-file:

```
active-processes
  name <name 1> <description 1>
  name <name 2> <description 2>
  :
  name <name n> <description n>
end-active-processes
```

Example of a processes list in the sub-file

```
active-processes
  name 'Compos' 'Composition'
  name 'Nitrif_NH4' 'Nitrification of ammonium'
  name 'DecFast' 'Mineralization fast decomp. detritus POC1'
  :
  name 'Daylength' 'Daylength calculation'
end-active-processes
```

20.2.3 Dry segments

When cells only contain a limited amount of water, or no water at all, some processes can behave peculiar, and cause negative concentrations or instabilities. To prevent this, processes receive a signal when segments are dry. Several processes consciously ignore the dry label and will always do calculations in dry cells because they deal with it properly, or e.g. are for processes in the sediment that continue in dry cells.

By default, segments are considered dry when the volume drops below $0.001m^3$ or the depth of the layer drops below $0.001m$. The same defaults are used in D-Water Quality. The user can adjust these settings by using the following keywords in the [processes] section of the mdu-file:

```
[processes]
VolumeDryThreshold = 1.0e-3
DepthDryThreshold = 1.0e-3
```

20.3 Definition of processes parameters

In D-Water Quality there are four types of parameters, constant in time and space, varying in time, varying in space, and varying in both time and space. Currently, only the first three types are supported. In [Table 20.2](#), you can see an overview of these types, with the D-Water Quality term for them in italics, and the way they can be specified in D-Flow FM.

Table 20.2: Various types of parameter inputs for water quality models in D-Flow FM.

	Constant in time	Varying in time
Constant in space	<i>constant</i> define a parameter in the sub-file (Section 20.3.1)	<i>function</i> define a waqfunction in the ext-file (Section 20.3.3)
Varying in space	<i>parameter</i> define a waqparameter in the ext-file (Section 20.3.2)	<i>segment function</i> not supported yet (Section 20.3.4)

20.3.1 Constant parameter input

While in D-Water Quality constants are part of 'block 7' in the input file, in D-Flow FM they are moved to the substance file, where (confusingly) they are called **parameter**. A constant value in the substance file can however be trumped by a spatial or temporal definition in the ext-file.

Format for a *constant* definition in the sub-file (in which they are called **parameter**):

```
parameter <parameter name>
    description      <description>
    unit            <unit>
    value           <value>
end-substance
```

An example of a *constant* definition in the sub-file (in which they are called **parameter**):

```
parameter 'V0SedIM1'
```

```
description      'sedimentation velocity IM1'
unit            '(m/d)'
value           7.2000E-00
end-parameter
parameter 'TaucRS1DM'
    description      'critical shear stress for resuspension DM layer S1'
    unit            '(N/m2)'
    value           0.2000E+00
end-parameter
```

20.3.2 Spatial parameter input

Spatially varying input for a process parameter is called a parameter in D-Water Quality. In D-Flow FM, these are defined in the ext-file using QUANTITY=waqparameter<name>. At the moment, data can only be specified in 2D, and is copied to all layers.

```
QUANTITY=waqparameterV0SedIM1
FILENAME=para1.pol
FILETYPE=10
METHOD=4
OPERAND=O
VALUE=3.6
```

```
QUANTITY=waqparameterV0SedIM1
FILENAME=para2.pol
FILETYPE=10
METHOD=4
OPERAND=O
VALUE=7.2
```

20.3.3 Temporal parameter input

Temporal varying input for a process parameter is called a function in D-Water Quality. In D-Flow FM, these are defined in the ext-file using QUANTITY=waqfunction<name>.

```
QUANTITY=waqfunctionV0SedIM1
FILENAME =V0SedIM1.tim
FILETYPE =1
METHOD   =1*
OPERAND  =O
```

```
QUANTITY=waqfunctionV0SedIM2
FILENAME =V0SedIM2.fun
FILETYPE =1
METHOD   =0*
OPERAND  =O
```

* file containing a time series for this parameter ** 0 = block, 1 = linear interpolation

20.3.4 Spatial and temporal parameter input

Parameter input data that is both spatial and temporal varying is called a segment function in D-Water Quality. In an integrated model, you can specify external data sources, but also connect to hydrodynamic and meteorological data that is already available within D-Flow FM in your model.

Defining external data sources for segment functions

In D-Flow FM, you can specify segment functions in the ext-file using QUANTITY=waqsegment function<name>. At the moment, only data specified on a curvilinear grid in a NetCDF is supported (filetype=11). We hope to support temporal varying sample files in the near future. Data can only be specified in 2D, and is copied to all layers.

```
QUANTITY=waqsegment functionIM1
FILENAME=f34_sediments.nc
VARNAME=IM1
FILETYPE=11
METHOD=3
OPERAND=O
```

Selecting internal data to make it available for the processes

Within D-Flow FM it is possible to connect several parameters to the water quality processes. If you mention the name of certain parameters in the sub-file, they will be replaced by data from D-Flow FM, when available. Table 20.3 shows which data can be connected by which parameters. Which data is actually connected is reported in the dia-file.

Table 20.3: Data form D-Flow FM that is available for water quality processes

D-Flow FM data	D-Water Quality name in sub-file
horizontal surface area	Surf (default)
bottom shear stress	Tau or TauFlow*
flow element center velocity	Velocity
salinity	Salinity
temperature	Temp
wind velocity magnitude	VWind
wind direction	WindDir
fetch length and fetch depth	Fetch and/or InitDepth**
solar radiation	RadSurf
rain (mm/day)	Rain (mm/h)

*The bottom shear stress can be connected to Tau or TauFlow. When you connect to TauFlow, you can use the CalTau to calculate a Tau for the water quality processes with an additional bottom shear stress. Please be aware that if the D-Flow FM switch jawaveSwartDelwaq has been set to anything other than zero or D-Waves is coupled, there is already wave bottom shear stress included in TauFlow. Adding extra wave bottom shear stress with CalTau would lead to a doubling of wave bottom shear stress.

**Fetch length and fetch depth are both connected when either Fetch and/or InitDepth is mentioned in the sub-file.

If do not want to use the available data from D-Flow FM but want to specify your own input, you must not mention the parameter in the sub-file. You can add spatial or temporal input with

possibly with just one uniform number if you actually wanted to provide a constant value.

20.4 Initial conditions, boundary conditions and sources and sinks

Most substances in the model will have certain initial conditions, boundary conditions and sources and sinks (the equivalent of D-Water Quality waste loads). An integrated D-Flow FM and D-Water Quality model makes uses of D-Flow FM tracers, therefore you have to make use of the D-Flow FM methods to prescribe them (See also: [Section 9.4](#)). The options will be briefly discussed here.

20.4.1 Initial condition

The initial conditions for transported water quality substances are handled in exactly the same manner as those for any other constituent, i.e. you can specify a horizontally spatially varying field in the usual way through the ext-file using QUANTITY=initialtracer<name>.

For non-transported water quality substances you have to use QUANTITY=initialwaqbot<name>.

You can also refer to a restart file in the mdu-file. Restart conditions are read from the restart file by substance name, not by order in the file.

```
QUANTITY=initialtracerIM1
FILENAME=inittracer.pol
FILETYPE=10
METHOD=4
OPERAND=O
VALUE=5.0
```

```
QUANTITY=initialwaqbotIM1S1
FILENAME=inittracer.pol
FILETYPE=10
METHOD=4
OPERAND=O
VALUE=100.0
```

20.4.2 Boundary conditions

All of the D-Flow FM options for constituents are also available for transported water quality substances. You can specify boundary conditions for all these substances at all open inflow boundaries in the ext-file using QUANTITY=tracerbnd<name>. When modelling in three dimensions you may choose to specify boundary concentrations that have a uniform, linear, or step distribution over the vertical. You may also choose to specify a “Thatcher-Harleman” return time to simulate the re-entry of material that flowed out of the model after the flow reverses direction. When no boundary concentration is prescribed the concentration is presumed to be zero.

```
QUANTITY=tracerbndIM1
FILENAME=leftIM1_sed.pli*
FILETYPE=9
METHOD=3
OPERAND=O
```

```
QUANTITY=tracerbndIM2
```

```
FILENAME=leftIM2_sed.pli*
FILETYPE=9
METHOD=3
OPERAND=O
```

* tim-file (time series) with the same name is read

20.4.3 Sources and sinks

Sources and sinks are the equivalent of waste loads in D-Water Quality. They may be provided using `QUANTITY=discharge_salinity_temperature_sorsin` in the ext-file as follows:

```
QUANTITY=discharge_salinity_temperature_sorsin
FILENAME      =WWTP.pliz*
FILETYPE      =9
METHOD        =3
OPERAND       =O
AREA          =0
```

* tim-file (time series) with the same name is read

The tim-file simultaneously prescribes sources and sinks of

- ◊ water volume itself (i.e. discharge),
- ◊ salinity (when switched on in the model), and
- ◊ temperature (when switched on in the model), and.
- ◊ any other constituents that are transported.

See [section 8.8](#) for more details.

The number of columns in the time file is equal to 2 (time and discharge) + 0/1/2 (depending on whether salinity and/or temperate are switched on) + <the number of constituents>. The diagnostic file contains a list of the actual order of the constituents. The non-transported substances are not part of the constituents, and do not need values in the tim-file. Please be aware that the order in which you specify boundary conditions, initial conditions in the ext-file and the substances in the sub-file might influence the order of the constituents. This can lead to another interpretation of the tim-file than you would have expected, because the columns in the tim file cannot be labeled with names.

Because D-Flow FM is a hydrodynamic model, the water discharge is always added to the model. Dry waste loads are not possible yet, but a workaround might be that you divided the discharges by a factor 1000 or 10^6 , and multiply the concentrations by the same factor.

20.5 Output options

20.5.1 Map and history output

All output mention in the sub-file is written to both the map-output and his-output of D-Flow FM. This might not always be desirable. To distinguish between map and history output, it is possible to move a selection of outputs from the sub-file to a separate file that must be mentioned in the mdu-file. The map-file will only contain outputs mention in the sub-file. The his-file will contain all outputs mention in both the sub-file and the AdditionalHistoryOutput-File. Use the MDU keyword [processes], AdditionalHistoryOutputFile.

Example of this setting in the mdu-file:

```
[processes]
AdditionalHistoryOutputFile = addhisout.echo
```

Format for an output definition in the sub-file or echo-file:

```
output <output name>
    description      <description>
end-output
```

Example of the output definition in the sub-file or echo-file:

```
output 'Tau'
    description  'total bottom shear stress'
end-output
output 'Surf'
    description  'Horizontal surface'
end-output
output 'TotalDepth'
    description  'Total depth of water column'
end-output
```

20.5.2 Mass balance areas

The user can define mass balance area using polygons using QUANTITY=waqmassbalancearea<name>. If there is any overlap in the polygons when using multiple mass balance areas, the last definition will overrule any previous definitions. If there are cells remaining that are not included in any mass balance area, they will be grouped in an extra mass balance area named *Remaining cells*. This will cut up the whole model area in non-overlapping mass balance areas.

D-Flow FM will produce mass balances for the defined areas, that gives an overview of the fluxes between the various mass balance area's, over the model boundaries, and the processes fluxes in each mass balance area. The mass balances will be written to text and binary files at a given interval, and at the end of the run. Use the MDU keyword [processes], DtMassBalance to set the interval.

Limitation: DtMassBalance must be a multiple of the dtuser

Example of this setting in the mdu-file:

```
[processes]
DtMassBalance          = 86400.0
```

Example of the mass balance area definition in the ext-file.

```
QUANTITY=waqmassbalanceareaBalArea1
FILENAME=barea1.pol
FILETYPE=10
```

```

METHOD=4
OPERAND=O
VALUE=1.0 *

QUANTITY=waqmassbalanceareaBalArea2
FILENAME=barea2.pol
FILETYPE=10
METHOD=4
OPERAND=O
VALUE=1.0 *

```

*Ignored

20.6 Running D-Flow FM with processes

When running D-Flow FM with processes, it needs to read a processes data base that contains input/output information of all the processes. The processes database is closely related to the code in the executable, and generally it is advised to use the process library that comes with the D-Flow FM executable. When using the BLOOM algae model, D-Flow FM also needs to read a BLOOM algae species file. Both must be given as command line arguments in the following way:

```

> dflowfm-cli <mdu-file> --autostartstop --processlibrary <proc_def>
--bloomspecies <bloom.spe>

--processlibrary PROCESSLIBRARYFILE
Specify the process library file to be used for water quality processes.

--bloomspecies BLOOMSPECIESFILE
Specify the BLOOM species definition file to be used for water quality processes.

```

20.7 Output

The water quality output can be found in the D-Flow FM map and history files according to the user specifications (see: Section 20.5).

The output for the mass balance areas is written into two text files named after the mdu-file, and appended with '_bal.txt' and '_baltot.txt'. They contain an overview mass balance areas, the active and inactive substances and the fluxes that affect the substances and their stoichiometry factor.

What follows for each mass balance area (and for each mass balance interval in the case of the _bal.txt-file), is a water and mass balance for each substance, followed by a water and mass balance for each substance for the whole area.

20.8 Known issues and limitations

The processes functionality has the following limitations: - although the sinks and source are working for tracers and thus water quality substances, it is a bit difficult to add concentrations to these sinks and sources. In the tim-file that accompanies the sinks and sources, besides the discharge the user has to specify the concentrations of constituents in unnamed columns. But

the order is determined by the order that constituents are defined in firstly the ext-file and then the sub-file (for substances that have no initial conditions or boundaries. Adding a substance means you'll immediately have to fix the tim-file. The dia-file now contains an actual list of the order of the constituents - there is a significant performance difference when substances are defined in a sub-file too (without any processes), compared to when they are only defined by boundary and initial conditions that should not be there.

And we would like to make the following improvements in the near future: - tim files are using time in minutes since reference date. We would like to see support for absolute dates too. - in D-Water Quality, the user could specify multiple concentrations for boundary conditions in the same table, and use 'USEFOR' statements for flexible boundary definitions. - we want to enable the statistical operations of D-Water Quality- the binary mass balance output should be written in a NetCDF format - we want to write (mass balances, and) average concentrations and other outputs for monitoring areas - user interface support in DeltaShell

21 Calibration and data assimilation

Note: Calibration of D-Flow FM with OpenDA is a β -functionality.



21.1 Introduction

A flow model in D-Flow FM will generally benefit from parameter *calibration* to closer match observation data. When the model runs in an operational system *data assimilation* can be used to into the running model. For the automatic calibration and data assimilation, the open source toolbox **OpenDA** is available.

OpenDA basically provides three types of building blocks: an optimisation algorithm that performs the calibration or data assimilation, communication routines for passing information between OpenDA and D-Flow FM, and methods for handling observation data. The communication between OpenDA and D-Flow FM is realised using a Black Box approach. A number of wrapper objects (so called dataObjects) are available for reading and writing D-Flow FM input and output files.

This chapter contains a description of how the OpenDA toolbox could be deployed to apply calibration and data assimilation. The OpenDA tools can run D-Flow FM models and analyze the model results. A more extensive design description of the D-Flow FM wrappers for OpenDA can be found in the [D-Flow FM TRM \(2015\)](#).

More information on OpenDA can be found on the website <http://www.opendata.org>.

General information on the installation of OpenDA to get started is provided in [section 21.2](#). [section 21.4](#) gives an overview of the black box wrapper for D-Flow FM. [section 21.4](#) describes the OpenDA configuration and the related D-Flow FM files. The generation of noise and how this noise is added to forcings and boundaries is given in [section 21.5](#). Examples case for calibration and data assimilation using the ensemble Kalman filter are described in [section 21.6](#).

21.2 Getting started with OpenDA

The required D-Flow FM wrapper is enclosed in the official OpenDA release since version 2.2.2. The following three elements are needed for a calibration or data assimilation run with D-Flow FM:

- 1 The D-Flow FM Command Line Interface (CLI) installation. All OpenDA algorithms start D-Flow FM with a shell script `start_dflowfm.sh` or a batch script `start_dflowfm.bat`. Both scripts assume that D-Flow FM executable is available on the search path (i.e. it should be executable from the command line). If this is not the case change the environment variable `PATH` to include the installation path of the D-Flow FM executable.
- 2 An OpenDA installation including the OpenDA core, the D-Flow FM specific wrapper code and examples.
- 3 A Java Runtime Environment (JRE) version 7 (or higher) is needed to run OpenDA (2.2.2). OpenDA can use one of the system installed JREs or alternatively an JRE can be installed directly in the OpenDA directory at the same level as the `<bin>` directory.

For Linux it is required to run `source settings_local.sh` to setup OpenDA. The OpenDA GUI can be started from the `<bin>` directory using `oda_run_gui.bat` (Windows) or `oda_run.sh` with the `-gui` command line option (Linux).

21.3 The OpenDA black box model wrapper for D-Flow FM

For a D-Flow FM model to function within the OpenDA toolbox we need to establish two things:

- 1 the control to propagate the model over time and
- 2 access to the model state, physical parameters, boundary conditions and external forcings.

In the black box approach the standard D-Flow FM command line executable is used to propagate the model over time. Access to the model state, physical parameters, boundary conditions and external forcings is obtained by reading from and writing to the D-Flow FM input and output files. Inside OpenDA all data is available in the form of exchange items which all have an unique identifier.

Calibration and data assimilation typically need multiple model evaluations with altered parameters, forcings, boundary conditions or the initial model state. In the black box approach this is achieved by creating multiple work directories containing altered model input files and starting the D-Flow FM executable in each work directory. D-Flow FM model results are than read from the work directories and compared to observation data.

For calibration this is an iterative process. Results from model evaluations $1, \dots, n$ are required to obtain a better estimate for parameter values, which are then evaluated in run $n + 1$.

In case of an ensemble Kalman filter (EnKF) run, the D-Flow FM computations (one run for each ensemble member) is stopped each time an observation is available, the input files for each ensemble member are modified according to the ensemble Kalman filter algorithm, after which the D-Flow FM run is restarted.

Next to the D-Flow FM model configuration OpenDA has its own configuration for selecting the algorithm, observations and interfacing with the D-Flow FM input an output files.

21.4 OpenDA configuration

21.4.1 Main configuration file and the directory structure

The OpenDA main configuration file has the `.oda` extension. All OpenDA configuration files use the xml format. It is advised to use a schema aware xml editor, when making changes to the OpenDA configuration. These editors provide direct access to the documentation that is stored in the schema and can validate the correctness of the xml files.

All other xml config file names and directories are configurable. However, there is a commonly used directory layout and naming convention. All the examples are configured using this convention. For example, the directory structure for the `simple_waal_kalman` example is given in [Table 21.1](#).

All the work directories are available in the `<stochModel>` directory, after performing a run with OpenDA they contain the D-Flow FM results.

It is a good practice to name the main configuration file corresponding to the algorithms executed by OpenDA. The following files are used in the provided examples:

- ◊ `<Simulation.oda>`: performs a regular D-Flow FM simulation run, only the executable is started by OpenDA. This algorithm is useful to check the configuration.
- ◊ `<Dud.oda>`: Dud (Doesn't Use Derivative) is one of the optimisation algorithms available for calibration purposes.

```

<simple_waal_kalman>
  <algorithm>..... calibration method or data-assimilation algorithm
    <enkfAlgorithm.xml> ..... algorithm specific configuration
    ...
  <stochModel> ..... model and its uncertainty description
    <bin>..... bat and .sh scripts for calling D-Flow FM
    <input_dflowfm> ..... D-Flow FM template configuration
    ...
    <work0/>..... work directory for propagated mean
    <work1/>..... work directory for first ensemble member
    ...
    <dflowfmModel.xml> ..... exchange items configuration
    <dflowfmStochModel.xml> ..... predictor, state and parameter
    <dflowfmWrapper.xml> ..... actions and data objects configuration
    ...
  <stochObserver> ..... observations and uncertainty
    <noosObservations.xml>..... observations and uncertainty
    <waterlevel_Obs01.noos> ..... raw observation file
    ...
  <Enkf.oda> ..... main configuration file
  ...

```

Table 21.1: Directory structure of the OpenDA Ensemble Kalman filtering configuration for the simple Waal D-Flow FM model.

- ◊ <SequentialSimulation.oda>: performs a D-Flow FM simulation run through which the executable is stopped and restarted by OpenDA at the moments for which observed data are available.
- ◊ <Enkf.oda>: performs an ensemble Kalman filtering.

The main configuration for an OpenDA application has three mandatory components, which make up an OpenDA application: `stochModelFactory`, `stochObserver` and `algorithm`. Each component is configured by specifying its `className` attribute, `workingDirectory` and `configFile/configString`. There are optional components to enable writing OpenDA results, to define OpenDA restart input and output files and to enable timings. The `resultwriter` is typically useful for writing stochastic properties that are only available to OpenDA and not in D-Flow FM.

The main configuration file `<Enkf.oda>` for the `simple_waal` example:

```

<?xml version="1.0" encoding="UTF-8"?>
<openDaApplication
  xmlns="http://www.opendata.org"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation
  ="http://www.opendata.org http://schemas.opendata.org/openDaApplication.xsd">
  <stochObserver className="org.opendata.observers.NoosTimeSeriesStochObserver">
    <workingDirectory>./stochObserver</workingDirectory>
    <configFile>noosObservations.xml</configFile>
  </stochObserver>
  <stochModelFactory className="org.opendata.blackbox.wrapper.BBStochModelFactory">
    <workingDirectory>./stochModel</workingDirectory>
    <configFile>dflowfmStochModel.xml</configFile>
  </stochModelFactory>
  <algorithm className="org.opendata.algorithms.kalmanFilter.EnKF">
    <workingDirectory>./algorithm</workingDirectory>
    <configString>EnkfAlgorithm.xml</configString>
  
```

```
</algorithm>
<resultWriter className="org.opendata.resultwriters.MatlabResultWriter">
  <workingDirectory>.</workingDirectory>
  <configFile>Enkf_results.m</configFile>
  <selection>
    <resultItem id="pred_f"/>
    <resultItem id="pred_f_0"/>
    <resultItem id="pred_f_1"/>
    <resultItem id="pred_f_std"/>
    <resultItem id="pred_f_central"/>
    <resultItem id="pred_a_linear"/>
    <resultItem id="analysis_time"/>
    <resultItem id="obs"/>
  </selection>
</resultWriter>
</openDaApplication>
```

Note that the directory layout in this section is created by setting the `workingDirectory` elements in `stochModelFactory`, `stochObserver` and `algorithm` parts of the configuration. Each of these components have their own configuration, which are described in the following sections.

21.4.2 The `algorithm` configuration

All provided methods for calibration and data assimilation are configurable through an xml file. The convention is to include the algorithm name in the file name e.g `<EnkfAlgorithm.xml>`. For data assimilation algorithms the configuration typically specifies the ensemble size and the option to use stochastic parameters, forcing and initialisation. For calibration algorithms the configuration typically contains definition of the cost function and tolerances and stopping criteria. For a list of algorithms and their configuration options see the general OpenDA documentation.

21.4.3 The `stochObserver` configuration

The access to observations is standardized in OpenDA using a `stochObserver` object. The configuration and the observation data are placed in the `<stochObserver>` directory. OpenDA contains a number of different `stochObserver` objects that can handle different types of data files. In this manual, we discuss the `NoosTimeSeriesStochObserver` and the more generic `IoObjectStochObserver`. For a more complete list of available `stochObservers` see the OpenDA web site.

21.4.3.1 `NoosTimeSeriesStochObserver`

The NOOS file format is used to store time series. The format is created for use by the members of the North West European Shelf Operational Oceanographic System <http://www.noos.cc/>. An example of (a part of) a NOOS file is:

```
#-----
#-----
# Location : station01
# Position : (0.0,0.0)
# Source : twin experiment DFlowFM
# Unit : waterlevel
# Analyse time: null
# Timezone : null
#-----
199101010000  1.0000
199101010100  0.8944
199101010200  0.6862
```

199101010300	0.5956
199101010400	0.3794
199101010500	0.1372
199101010600	-0.1300
199101010700	-0.3044
199101010800	-0.3963
199101010900	-0.3739
199101011000	-0.1930
...	

The file contains a header with meta data specifying among others the location name (**Location**) and the quantity (**Unit**). The data is written in two columns, the first gives the time of the observation using the 'YYYYMMDDhhmm' format and the second column gives the measured value.

The file <noosObserver.xml> defines a number of time series, each coupled to a NOOS file containing the measurements.

```
<?xml version="1.0" encoding="UTF-8"?>
<noosObserver
    xmlns="http://www.opendata.org"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation
        ="http://www.opendata.org http://schemas.opendata.org/schemas/noosObservations.xsd">
<timeSeries status="use" standardDeviation="0.05">
    den_helder_waterlevel_astro.noos
</timeSeries>
<timeSeries status="use" standardDeviation="0.05">
    aberdeen_waterlevel_astro.noos
</timeSeries>
</noosObserver>
```

OpenDA creates an exchange item for each observation time series. The default exchange item id (identifier) is created using the location (**Location**) and the quantity (**Unit**). The standard deviation (measurement error) is specified with **standardDeviation** attribute.

21.4.3.2 IoObjectStochObserver

This observer uses a **dataObject** for handling the file IO. All exchange items that are provided by **dataObject** can be used by the observer. For instance the **NetcdfDataObject** can read and write to NetCDF files, and has an exchange item for each variable in the NetCDF file. The **lake_kalman** example uses this approach to use the *.his file from a D-Flow FM run as synthetic observations for a ensemble Kalman filter run. The <dflowfmStochObsConfig.xml> file reads:

```
<?xml version="1.0" encoding="UTF-8"?>
<ioObjectStochObserver
    xmlns="http://www.opendata.org"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation
        ="http://www.opendata.org http://schemas.opendata.org/openDaStochObserver.xsd">
<uncertaintyModule
    workingDirectory="."
    className="org.opendata.uncertainties.UncertaintyEngine">
    <arg>stochObsUncertainties.xml</arg>
</uncertaintyModule>
<ioObject
    workingDirectory="."
    className="org.opendata.exchange.dataobjects.NetcdfDataObject">
    <fileName>lake2d_his.nc</fileName>
</ioObject>
```

```
</ioObjectStochObserver>
```

In the configuration of UncertaintyEngine OpenDA object, a selection of these exchange items id's is made and a standard deviation is specified.

```
<?xml version="1.0" encoding="UTF-8"?>
<uncertainties
    xmlns="http://www.wldelft.nl"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation
        ="http://www.wldelft.nl http://schemas.opendata.org/uncertainties.xsd"
    version="1.0">
<uncertaintyType>ProbabilityDistributionFunction</uncertaintyType>
<probabilityDistributionFunction id="S1.waterlevel" isActive="true">
    <normal mean="0" stdv="0.05" stdvIsFactor="false"/>
</probabilityDistributionFunction>
<probabilityDistributionFunction id="S2.waterlevel" isActive="true">
    <normal mean="0" stdv="0.05" stdvIsFactor="false"/>
</probabilityDistributionFunction>
<probabilityDistributionFunction id="S3.waterlevel" isActive="true">
    <normal mean="0" stdv="0.05" stdvIsFactor="false"/>
</probabilityDistributionFunction>
<probabilityDistributionFunction id="S4.waterlevel" isActive="true">
    <normal mean="0" stdv="0.05" stdvIsFactor="false"/>
</probabilityDistributionFunction>
</uncertainties>
```

21.4.4 The stochModel configuration

The stochModel configuration usually consists of three <*.xml> files in the <stochModel> directory. These are called:

- ◊ <dflowfmWrapper.xml>: This file specifies the actions to perform in order to run a D-Flow FM simulation and list the D-Flow FM input and output files that can be used to let OpenDA interact with the model. For each file the dataObject is specified which is used for handling the specified file.
- ◊ <dflowfmModel.xml>: This file contains a list of the exchange items which are provided by the configured dataObject. The model time information is constructed by the time-InfoExchangeItems element. It also contains a number of alias values which can be used in three stochModel configuration files.
- ◊ <dflowfmStochModel.xml>: In this file defines the predictor, the selection of observations which are compared to the model results. For calibration this file also specifies which parameters can be changed. For data assimilation this file contains the definition of the model state and the noise (uncertainty) specification for the boundaries and forcings.

21.4.5 D-Flow FM files and the OpenDA dataObjects configuration

The dataObjects for reading and writing provide so-called exchange items that allow OpenDA to manipulate specific parts of the files of D-Flow FM. The D-Flow FM files that can be manipulated by OpenDA and the corresponding OpenDA class names are given in [Table 21.2](#).

Table 21.2: D-Flow FM files that can be manipulated and the corresponding OpenDA class names to be used in the `dflowfmWrapper.xml` file.

D-Flow FM filetype	OpenDA dataObject & exchange items
<*.mdu>	org.opendata.model_dflowfm.DFlowFMTIMEINFO IDs: start_time, end_time
<*.amu>	org.opendata.model_dflowfm.DFlowFMMeteoFile ID: x_wind
<*.amv>	org.opendata.model_dflowfm.DFlowFMMeteoFile IDs: y_wind
<*.amp>	org.opendata.model_dflowfm.DFlowFMMeteoFile ID: air_pressure
<*.tim>	org.opendata.model_dflowfm.DFlowFMTIMESERIESDataObject IDs: BOUNDARY_ID.#:QUANTITY
<*.xyz>	org.opendata.model_dflowfm.DFlowFMXYZFile IDs: FILENAME_#
<*_his.nc>	org.opendata.exchange.dataobjects.NetcdfDataObject IDs: STATION_ID.VARIABLE_NAME
<*_map.nc>	org.opendata.model_dflowfm.DFlowFMRestartFileWrapper IDs: VARIABLE_NAME
<*.cld>	org.opendata.model_dflowfm.DFlowFMCALIBRATIONFactorFile IDs: CalFactor-CALIBRATION_DEFINTION_NUMBER, CalFactor-CALIBRATION_DEFINTION_NUMBER-q{DISCHARGE}, CalFactor-CALIBRATION_DEFINTION_NUMBER-h{WATERLEVEL}
<*.ttd>	org.opendata.model_dflowfm.DFlowFMTrajectoryFile IDs: RoughNr_{ROUGHNESSNR}_FormulaNr{FORMULANR}_{FORMULAPARAMETER} RoughNr_{ROUGHNESSNR}_DISCHARGE{DISCHARGE}_FormulaNr{FORMULAPARAMETER} RoughNr_{ROUGHNESSNR}_WATERLEVEL{WATERLEVEL}_FormulaNr{FORMULAPARAMETER}

All the dataObjects and their configuration are described in the following sections.

21.4.5.1 Start and end time in the model definition file (.mdu)

The start and end time are set in <*.mdu>-file by OpenDA using the `start_time` and `end_time` exchange items. These are provided by the `DFlowFMTIMEINFO` data object.

D-Flow FM reference	Exchange Item Id	Remark
TStart	start_time	RefDate and Tunit needed for interpretation
TStop	end_time	RefDate and Tunit needed for interpretation

21.4.5.2 External forcings (.xyz)

All D-Flow FM external forcings are specified via the <*.ext> forcings file. Here a spatial forcing can be defined by using a .xyz-file (e.g. the bed friction coefficients). For instance the file <nikuradse.xyz> contains:

```
x1 y1 0.9994357525438934
x2 y2 0.9994357525438934
x3 y3 2.0021214673505600
x4 y4 2.0021214673505600
x5 y5 2.0021214673505600
x6 y6 2.0021214673505600
```

When performing calibration of a spatial field it is often required to group points in a select number of regions. The calibration then does not change the individual values but applies a factor to all values in the group. The best approach is to create a file with multipliers (e.g <friction_multiplier.xyz>) which are initially all equal to one.

```
x1 y1 1.0
x2 y2 1.0
x3 y3 1.0
x4 y4 1.0
x5 y5 1.0
x6 y6 1.0
```

The multiplication (or addition) with the values in `nikuradse.xyz` should be configured in the `*.ext` file.

Group from keywords in file

One option to construct groups is to use keywords directly in the <friction_multiplier.xyz> file:

```
x1 y1 1.0 #friction_3
x2 y2 1.0 #friction_3
x3 y3 1.0 #friction_1
x4 y4 1.0 #friction_1
x5 y5 1.0 #friction_4
x6 y6 1.0 #friction_4
```

In the OpenDA wrapper config the dataObject is than configured as:

```
<ioObject className="org.opendata.model_dflowfm.DFlowFMXyzFile">
<file>friction_multiplier.xyz</file>
<id>frictionCoefFile</id>
<arg>idsFromKeywordsInFile</arg>
</ioObject>
```

This will create exchange items with identifier `friction_3`, `friction_1` and `friction_4`.

Group from template file

An other options is to use a template file (<friction_multiplier_template.xyz>) with exactly the same (x,y) coordinates as in (<friction_multiplier.xyz>). The third column is used to define groups by using these values as group numbers:

```

x1 y1 3.0
x2 y2 3.0
x3 y3 4.0
x4 y4 4.0
x5 y5 1.0
x6 y6 1.0

```

In the OpenDA wrapper config the dataObject must be configured as:

```

<ioObject className="org.opendata.model_dflowfm.DFlowFMXyzFile">
  <file>friction_multiplier.xyz</file>
  <id>frictionCoefFile</id>
  <arg>idsFromTemplateFile=friction_multiplier_template.xyz</arg>
</ioObject>

```

This will create an exchange item for each group with identifier ‘FILE_BASENAME + _ + number from template file’, e.g. friction_multiplier_3, friction_multiplier_4 and friction_multiplier_1.

21.4.5.3 Boundary time series (.tim)

Boundary conditions are specified as a combination of a <*.pli> file and one or more .tim files. The DFlowFMTimeSeriesDataObject dataObject creates exchange items for all boundary conditions. It starts with reading the name of the external forcing file name from the <.mdu>-file (key ExtForceFile). The <*.ext>-file contains formatted blocks, one for each forcing. Forcings are defined along polylines, given in .pli-files. A <*.pli>-file is accompanied by a <*.cmp>- or a (number of) <*.tim>-file(s).

Noise can be added by means of an extra block in the .ext-file. As an example, noise is added to a boundary with a discharge imposed as:

```

QUANTITY =dischargebnd
FILENAME =sw_east_dis.pli
FILETYPE =9
METHOD =3
OPERAND =O

QUANTITY =dischargebnd
FILENAME =sw_east_dis_noise.pli
FILETYPE =9
METHOD =3
OPERAND =+

```

The discharge is set by the first block (operand=O), the information in the <*.pli>-files is identical and noise is added as a time series: the <*_noise.pli> file is always accompanied by a (number of) <*.tim> file(s). The location-information on the first line of the .pli-file combined with the quantity is used to construct the exchange item identifier: location.1.-dischargebnd. The numbering is used to discern between multiple <*.tim>-files possibly linked to a single <*.pli>-file.

21.4.5.4 Meteorological boundary conditions (<*.amu>, <*.amv>, <*.amp>)

OpenDA can read and write to the D-Flow FM <*.amu>, <*.amv> and <*.amp> files using the org.opendata.model_dflowfm.DFlowFMMeteoFile dataObject. These contain the x- and y components of the wind and the air pressure at the free surface on an equidistant

grid. In a typical data assimilation use noise fields are added to the wind. For the this purpose OpenDA can generate a spatial noise field on an equidistant grid (see [section 21.5](#)). D-Flow FM can combine fields defined in files on different to single field on the computational grid.

21.4.5.5 Result time series (<*_his.nc>)

The <*_his.nc>-file contains time series with D-Flow FM model results for a number of stations. The generic `org.opendataexchange.dataobjects.NetcdfDataObject` is used for handling this type of files. The `NetcdfDataObject` expects a NetCDF-file that contains dimensions `time` and `stations` plus a variable `station_id` of type `string` and dimension `stations`. For each variable in this NetCDF-file with dimensions `(time, stations)` an exchange item is created, that can be referred to as `station_id(i).variablename`. For instance:

```
dimensions:
time = UNLIMITED ; // (2882 currently)
stations = 3 ;
station_name_len = 40 ;
variables:
char station_id(stations, station_name_len) ;
(containing strings Obs1, Obs2, Obs3)
double waterlevel(time, stations) ;
(containing the computed values of the waterlevel)
```

results in three exchange items (a 1D vector in this case) with identifiers: `Obs1.waterlevel`, `Obs2.waterlevel` and `Obs3.waterlevel`.

21.4.5.6 Restart file (<*_map.nc>)

OpenDA provides the `org.opendamodel_dflowfm.DFlowFMRestartFileWrapper` for reading and writing the <*_map.nc>-file. This file contains all information needed to restart a D-Flow FM computation. Not all variables are relevant for manipulation by OpenDA: variable names that start with 'time', 'NetLink', 'BndLink', 'FlowLink', 'NetElem', 'FlowElem', 'NetNode', 'wgs84' and 'projected_coordinate_system' are ignored. Variables of other types than float or double are also ignored. For all other variables an exchange item is created, where the name of the variable in the NetCDF is used as the exchange item id.

Note: for Kalman filtering it is essential that the model starts from the <*_map.nc> file. It is not possible to specify an initial field by setting the `initialwaterlevel` or `initialsalinity` quantities in the <*.ext> file. If you want to set an initial field using these settings, make a custom D-Flow FM run where `TStop` is equal to `TStart` and use the created <*_map.nc> file for the starting point of the Kalman filtering.

21.4.5.7 Calibration factor definition file (<*.cld>)

The calibration factor definition file has the following layout. This file includes examples of the three different types of calibration definition factors and in comments the resulting names of the exchange items for OpenDA.

```
# [FileInfo]
#   FileType      = CalibrationFactorsDefinitionFile
```

```

# FileVersion = 1.0
# [CalibrationFactors]

# Non-Q-or-h-dependent
#
# calibration-class-nr    calibration-factor
#
34 0.9
#
# (will lead to exchange item CalFactor-34)

# Q-dependent calibration factors
#
# calibration-class-nr DISCHARGE "Cross-section name"
# calibration-class-nr Q1 ConstantValueAtQ1
# calibration-class-nr Q2 ConstantValueAtQ2
# calibration-class-nr Q3 ConstantValueAtQ3
# calibration-class-nr Q4 ConstantValueAtQ4
#
601 DISCHARGE "cross-section name"
601 0100 1.0
601 1000 1.0
601 5500 1.0

# (will lead to exchange items:
# CalFactor-601-q0100, CalFactor-601-q1000, CalFactor-601-q5500) etc.

# Waterlevel dependent calibration factor
# calibration-class-nr WATERLEVEL "Observation station name"
# calibration-class-nr H1 ConstantValueAtH1
# calibration-class-nr H2 ConstantValueAtH2
# calibration-class-nr H3 ConstantValueAtH3
# calibration-class-nr H4 ConstantValueAtH4
#
3 WATERLEVEL "water-level station name"
3 0.45 1.0
3 0.9 1.0
# (will lead to exchange items:
# CalFactor-3-h0.45 and CalFactor-3-h0.9)

```

OpenDA provides the `org.openda.model_dflowfm.DFlowFMCalibrationFactorFile` for reading and writing the `<*.cld>`-file. In the OpenDA wrapper config `<dflowfmWrapper.xml>` the `dataObject` can be configured as:

```

<iobj class="org.openda.model_dflowfm.DFlowFMCalibrationFactorFile">
  <file>FlowFM.cld</file>
  <id>calibFactorFileID</id>
</iobj>

```

In the `dflowfmModel.xml` a listing of the exchange items which are to be used can be found. To select all of the available exchange items from the calibration factor definition file, the following code can be used:

```

<exchangeItems>
  <vector id="allElementsFromIoObject" ioObjectID="calibFactorFileID"/>
</exchangeItems>

```

To select just a few of the exchange items, these can also be selected individually:

```

<exchangeItems>
  <vector id="CalFactor-601-q0100" ioObjectID="calibFactorFileID"
    elementID="CalFactor-601-q0100"/>

```

```

<vector id="CalFactor-601-q1000" ioObjectId="calibFactorFileID"
    elementId="CalFactor-601-q1000"/>
<vector id="CalFactor-601-h0.45" ioObjectId="calibFactorFileID"
    elementId="CalFactor-601-h0.45"/>
<vector id="CalFactor-601-q0.9" ioObjectId="calibFactorFileID"
    elementId="CalFactor-601-h0.9"/>
/>/exchangeItems>

```

21.4.5.8 Trachytopes roughness definition file (<*.ttd>)

The trachytopes definition file has the following layout. This file includes examples of the three different types of calibration definition factors and in comments the resulting names of the exchange items for OpenDA.

```

# [FileInformation]
#   FileType      = TrachytopesRoughnessDefinitionFile
#   FileVersion   = 1.0
# [RoughnessDefinitions]
#
# Constant roughness definitions
# roughness-definition-code formula-number formula-parameters
# Nikuradse 0.2
7      51      0.2
# (Leads to exchange-item RoughNr_7_FormulaNr_51_A)

# Linear trachytopes: wooden barrier
9      201     5.0      1.25
# (Leads to exchange-items RoughNr_9_FormulaNr_201_A
# and RoughNr_9_FormulaNr_201_B)

#      Uniform chezy value 25
10     52      25
# (Leads to exchange-item RoughNr_10_FormulaNr_52_A)

# Discharge dependent trachytopes
# roughness-definition-code DISCHARGE cross-section-name
# roughness-definition-code Q1 formula-number formula-parametersatQ1
# roughness-definition-code Q2 formula-number formula-parametersatQ2
11     DISCHARGE "m=1"
11     0.0      51      0.2
11     200.0    51      0.18
11     1000.0   51      0.1
%# (Leads to exchange-items RoughNr_11_DISCHARGE0.0_FormulaNr_51_A,
# RoughNr_11_DISCHARGE200.0_FormulaNr_51_A,
# RoughNr_11_DISCHARGE1000.0_FormulaNr_51_A)

# Water-level dependent trachytopes
# roughness-definition-code WATERLEVEL observation-station-name
# roughness-definition-code ZS1 formula-number formula-parametersatzS1
# roughness-definition-code ZS2 formula-number formula-parametersatzS2
11     WATERLEVEL "obs1"
16     -1.2    151     4.0      0.1
16     -1       151     4.0      0.3
16     2        151     5.0      0.1
# (Leads to exchange-items RoughNr_16_WATERLEVEL-1.2_FormulaNr_151_A,
# RoughNr_16_WATERLEVEL-1.2_FormulaNr_151_B,
# RoughNr_16_WATERLEVEL-1_FormulaNr_151_A,
# RoughNr_16_WATERLEVEL-1_FormulaNr_151_B,
# RoughNr_16_WATERLEVEL2_FormulaNr_151_A,
# RoughNr_16_WATERLEVEL2_FormulaNr_151_B)

```

OpenDA provides the `org.opendata.model_dflowfm.dflowfm.DFlowFMTrachytopeFile` for reading and writing the `<*.cld>`-file. In the OpenDA wrapper config `<dflowfmWrapper.xml>` the dataObject can be configured as:

```
<iobject className="org.opendata.model_dflowfm.dflowfm.DFlowFMTrachytopeFile">
<file>FlowFM.ttd</file>
<id>trachytopesFileID</id>
</iobject>
```

In the `dflowfmModel.xml` a listing of the exchange items which are to be used can be found. To select all of the available exchange items from the calibration factor definition file, the following code can be used:

```
<exchangeItems>
<vector id="allElementsFromIoObject" ioObjectId="trachytopesFileID"/>
</exchangeItems>
```

To select just a few of the exchange items, these can also be selected individually:

```
<exchangeItems>
<vector id="RoughNr_7_FormulaNr_51_A" ioObjectId="trachytopesFileID"
elementId="RoughNr_7_FormulaNr_51_A"/>
<vector id="RoughNr_11_DISCHARGE0.0_FormulaNr_51_A" ioObjectId="trachytopesFileID"
elementId="RoughNr_11_DISCHARGE0.0_FormulaNr_51_A"/>
<vector id="RoughNr_16_WATERLEVEL-1.2_FormulaNr_151_A" ioObjectId="trachytopesFileID"
elementId="RoughNr_16_WATERLEVEL-1.2_FormulaNr_151_A"/>
<vector id="RoughNr_16_WATERLEVEL-1.2_FormulaNr_151_B" ioObjectId="trachytopesFileID"
elementId="RoughNr_16_WATERLEVEL-1.2_FormulaNr_151_B"/>
</exchangeItems>
```

21.5 Generating noise

To add uncertainty to the external forcings and the boundary conditions, OpenDA has functionality for generating noise time series and spatial fields. The noise generation can be specified within the state definition in `<dflowfmStochModel.xml>`. For instance in the example `simple_waal_kalman`, a noise time series is created (ID `dischargenoise`), which is added to the inflow discharge (ID `eastboundary.1:dischargebnd`).

```
<?xml version="1.0" encoding="UTF-8"?>
<blackBoxStochModel
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation
    ="http://www.opendata.org http://schemas.opendata.org/blackBoxStochModelConfig.xsd"
  xmlns="http://www.opendata.org">
<modelConfig>
  <file>./dflowfmModel.xml</file>
</modelConfig>
<vectorSpecification>
  <state>
    <noiseModel
      id="boundaryNoiseModel"
      className="org.opendata.noiseModels.TimeSeriesNoiseModelFactory"
      workingDirectory=".">
      <configFile>BoundaryNoise.xml</configFile>
      <exchangeItems>
        <exchangeItem
          id="dischargenoise"
          operation="add"
          modelExchangeItemId="eastboundary.1:dischargebnd"/>
      </exchangeItems>
    </noiseModel>
    <vector id="s1"/>
    <vector id="unorm"/>
  </state>
</vectorSpecification>
```

```
</state>
</vectorSpecification>
</blackBoxStochModel>
```

The config file <BoundaryNoise.xml> contains the details of the noise:

```
<?xml version="1.0" encoding="UTF-8"?>
<mapsNoiseModelConfig>
<simulationTimespan timeFormat="dateTimeString">
    199208311200,199209010000,...,199212090000
</simulationTimespan>
<timeSeries
    id="dischargenoise"
    location="eastboundary"
    quantity="discharge"
    standardDeviation="0.2"
    timeCorrelationScale="6.0"
    timeCorrelationScaleUnit="hours"/>
</mapsNoiseModelConfig>
```

A noise time series is created (ID dischargenoise) with a correlation time of 6 hours and a standard deviation of 0.2. **Note:** currently OpenDA does not support the creation of D-Flow FM files, all files should exist in the <input_dflowfm>. In this case the boundary is defined in <sw_east_dis.pli>, but the noise is in a separate file <sw_east_dis_noise.pli>. Initially it contains zeros, but is filled with the noise values at run time. The time points in the file need to match the simulationTimespan in the OpenDA config. The total discharge at the boundary is constructed by D-Flow FM where the noise is added to original boundary values (see <simple_waal.ext>).

To generate a spatial correlation field the same approach can be used. In the lake_kalman example a noise field is added the wind. Instead of timeSeries a noiseItem is specified which contains the grid definition.

```
<?xml version="1.0" encoding="UTF-8"?>
<mapsNoiseModelConfig>
<simulationTimespan timeFormat="dateTimeString">
    200106240000,200106240100,...,200106270000
</simulationTimespan>
<noiseItem id="2DNoise" quantity="wind-x" unit="m/s" height="10.0"
    standardDeviation="20.0"
    timeCorrelationScale="12.0" timeCorrelationScaleUnit="hours"
    initialValue="0.0"
    horizontalCorrelationScale="10" horizontalCorrelationScaleUnit="km">
    <grid type="cartesian" coordinates="XY">
        <x>0,3000,...,63000</x>
        <y>0,3000,...,63000</y>
    </grid>
</noiseItem>
</mapsNoiseModelConfig>
```

The calculation of spatially correlated noise on a cartesian grid is quite fast as the *x* and *y*-direction are independent. The interpolation from an equidistant grid to the computational grid is performed within D-Flow FM. Again, note that an zero valued wind files should be present <input_dflowfm> folder, where the time stamps match with the ones given in simulationTimespan in the OpenDA configuration.

21.6 Examples of the application of OpenDA for D-Flow FM

In this section, some examples are elaborated for both the calibration of a model and the ensemble Kalman filtering (abbreviated as ‘EnKF’) of a model. All the examples can be found in the directory <examples/model_dflowfm_blackbox>.

21.6.1 Example 1: Calibration of the roughness parameter

The automatic calibration of a model needs two main choices from the user:

- 1 Which model parameters may be modified during the calibration process?
- 2 Which model results need to be compared to observations, to judge the model quality?

The remainder of this section will be in the form of a tutorial, to directly illustrate all steps in an example model. In this example you will use a small river model ‘simple_waal’ and use the bed roughness to calibrate this model for its three water level observation stations.

Step 1: Inspect the model

All the required model files can be found in the directory <simple_waal_calibration_roughness>. Consider the following steps:

- 1 Start D-Flow FM (standalone) in directory <input_dflowfm/>.
- 2 Select *Files* → *Load MDU-file*.
- 3 Load the model: select <simple_waal.mdu>.
- 4 You can run the model if you like (right mouse button).

The basic model is built to simulate a simple two-dimensional river with a spatially varying bed friction coefficient. It is driven by two boundary conditions: an upstream discharge inflow at the eastern boundary and a downstream water level at the western boundary. Inspect the model forcing in the following way:

- 1 Open the external forcings file <simple_waal.ext> in a text editor.
- 2 Notice how, in addition to the two boundaries, there are two blocks for the friction coefficient. The first one is a spatially varying roughness field in the <sw_nikuradse.xyz> file. The second refers to the <sw_frcfact_all.xyz> file that contains multipliers for the original friction coefficients. A third file <sw_frcfact_template.xyz> is present, which is used to define a number of subdomains.
- 3 In D-Flow FM, select *Files* → *Load sample file* and select <sw_frcfact_template.xyz>.
- 4 Notice how the loaded samples have three distinct values 1, 2 and 3, which act as identifiers: they approximately define the corner points of three subdomains of the entire river stretch. For each subdomain, a different roughness can be calibrated.

Step 2: Select the model parameters

Currently, the only calibratable parameters are the time-independent parameters in the external forcings file that use the .xyz sample file format. The most obvious parameter is the bed friction coefficient.

Step 3: Select the model results

The example directory <simple_waal> contains all the necessary configuration files for the so-called Black Box model wrapper for D-Flow FM to run a ‘twin experiment’. In a twin experiment a model setup with given solution (the synthetic observations) is perturbed after which OpenDA is applied to re-estimate the original settings. The effects of the parameter variations may be judged by comparing time series output in the <*_his.nc> history file to observed data in NOOS time series format. The model output and observation data are compared by calculating a cost function. Which cost function is used is configured in the <dudAlgorithm.xml> in the <algorithm> directory. See the OpenDA documentation for the available options for the cost function.

The D-Flow FM model simulates a 1D river flow. The input files for D-Flow FM are located in the directory <simple_waal/stochModel/input>. D-Flow FM allows the user to specify regions with a different bed friction coefficient (constant for each region) and is able to handle the interpolation of the coefficients between these regions. In the experiment we try to re-estimate the values of the bed friction coefficients of an earlier run. As observations, the waterlevel at three locations (stations) along the river is used. These results are written to the main output file (<*_his.nc>) as time series.

In the directory <simple_waal>, there are two main configuration files of OpenDA present:

- ◊ `Simulation.oda`: runs a single run of the model, this configuration is mainly used to test the black box configuration files.
- ◊ `Dud.oda` runs a calibration experiment with algorithm DUD (Doesn’t Use Derivative).

These files configure the main ingredients of an OpenDA run:

- 1 the `stochObserver` (`org.openda.observers.NoosTimeSeriesStochObserver`). Observations for this experiment were created by extracting the timeseries for all stations from the netcdf-file and convert them to NOOS format (script `nchis2noos.sh`) het script schrijft nog tijd sinds begin situatie.
- 2 the `stochObserver` (`org.openda.observers.NoosTimeSeriesStochObserver`). Observations for this experiment were created by extracting the time series for all stations from the netcdf-file and convert them to NOOS format (script `nchis2noos.sh`)
- 3 the `stochModelFactory` (`org.openda.blackbox.wrapper.BBStochModelFactory`). A black box model configuration consist of three configuration files that are described in more detail below.
- 4 the `algorithm` (`org.openda.algorithms.Dud`). Dud is a well known algorithm in calibration experiments, more information about it can be found on the OpenDA website or in the literature.

The configuration files for these 3 components are located in different sub directories to reflect the Object Oriented architecture of OpenDA. The fourth block in the configuration file specifies the result writer (`org.openda.resultwriters.MatlabResultWriter`). The resulting m-file may be loaded in Matlab to visualize results of the OpenDA run.

In this example, the data exchange between OpenDA and D-Flow FM is limited to the bed friction coefficients and the computed waterlevel at observation locations. The waterlevel at a observation location is expected to be written to NetCDF-file with the following features

- 1 dimension ‘time’ and ‘stations’ are defined
- 2 there exists a variable ‘station_id(stations)’ defined that contains strings with the station_id

For NetCDF-files that satisfy these two conditions OpenDA creates an ‘exchange item’ for each variable that has the dimensions (time, stations). The exchange item is referred to as ‘station id(nr)’.’name of variable’.

21.6.2 Example 2: EnKF with uncertainty in the tidal components

The geometry for this test case is the same as used in the Delft3D-FLOW model example for calibration that was presented in a Deltares webinar (recording, slides and all configuration files for this example are available at the OpenDA website).

Regularly, D-Flow FM uses 1 component file to specify all tidal component (one component at a line). In order to add different noise models to different components, you must split the component file and add one <*.tim>-files for noise for each component.

Again, all configuration files are available, but not much effort has been put into the exact configuration of the EnKF algorithm or the noise model specifications. The results of the SequentialSimulation show that this test case suffers much less from the inexact restart. The whole workflow is highlighted in more detail below.

A few remarks are made:

- ◊ the directories <bin> and <jre> should be on the same level,
- ◊ the computation is started through running the file oda_run_gui.bat,
- ◊ the bare D-Flow FM model is located in the directory <input_dflowfm>,
- ◊ the observations are in .noos-format, and are located in the directory <stochObserver>.

Step 2: Start the EnKF computation

The OpenDA run is launched through the core `oda_run_gui.bat` file. Once having opened this file, a user interface appears. Within the user interface, an <*.oda>-file can be opened from a certain case directory (in this case, we have `<estuary_kalman>`). One can choose `Enkf.oda`, `SequentialSimulation.oda` or `Simulation.oda`. In this case, we choose for `Enkf.oda`.

Step 3: Examine the applied noise

The basic necessary component of an EnKF computation comprises the noise applied to some variable. In this case, the noise is applied to the waterlevel boundary representing the tidal motion. Within the directory <input_dflowfm>, this noise is explicitly declared through a separate polyline and a separate data file for the boundary. The configuration of the noise is accomplished through two <*.xml>-files in the directory <stoch_model>.

Step 4: Run the EnKF computation

By means of the user interface, the EnKF computation can be launched. After having opened the file <EnKF.oda>, the *Run*-button can be pressed. The computation is being performed. Along the computation’s duration, multiple <work> directories are generated in the directory <stochModel>, i.e. <work0>, <work1>, <work2>, etc.

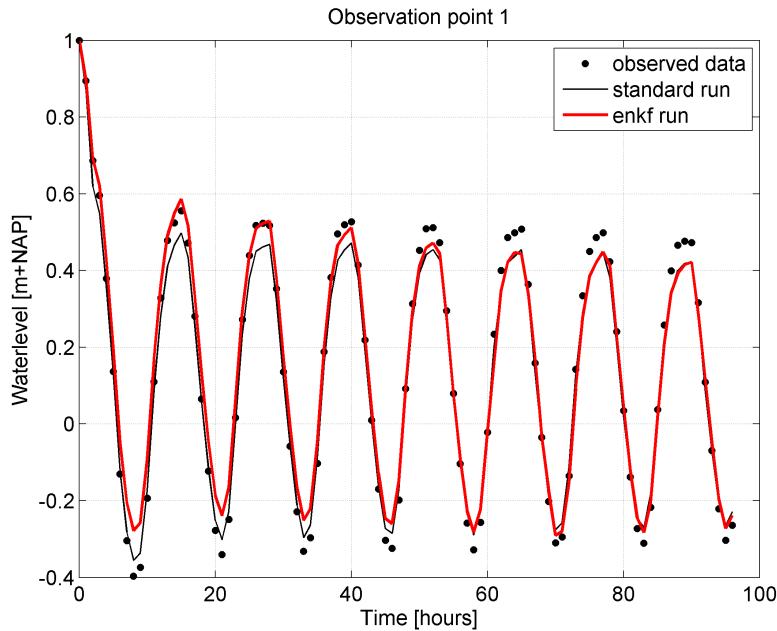


Figure 21.1: Visualisation of the EnKF computation results from OpenDA for a certain observation point. The dots represent the observed data, the black line represents the original computation with D-Flow FM (without Kalman filtering) and the red line represents the D-Flow FM computation with Kalman filtering.

Step 5: Evaluate the outcomes

After having run the computation, output files have been generated in Matlab-format. The relevant files are placed in the directory <estuary_kalman>. The data are stored in the Matlab file <Enkf_results.m>. Visualisation could be accomplished like shown in Figure 21.1.

21.6.3 Example 3: EnKF with uncertainty in the inflow velocity

The geometry in this example is the same as for the calibration example: a two-dimensional river model, the initial waterlevel is zero, the river bed is filled gradually due to the boundary conditions. At the inflow boundary, a constant discharge is prescribed (along the line <sw_east_dis_0001.pli>), whereas at the outflow boundary, a constant water level is prescribed (along the line <sw_west_wlev_0001.cmp>).

There are 3 observation locations along the river (Obs01, Obs02 and Obs03). The matlab script <plot_results_histfile.m> is available to plot the water level as a function of time for these 3 stations. Simulation time span is 100 days (Start: 199208310000, End: 199212090000). The noise contribution is found in file <sw_east_dis_noise.pli>. Initially, no noise is present, so the file contains zeroes in directory <input_dflowfm>.

21.6.4 Example 4: EnKF with uncertainty in the inflow condition for salt

The geometry in this example and the boundary conditions for waterlevel and velocity are exactly the same as in simple_waal_kalman. The transport of salt is added to the computation by a discharge boundary condition sw_east_dis_sal_001.pli and a noise component added to this boundary.

21.6.5 Example 5: EnKF with uncertainty on the wind direction

This example is also converted from a Delft3D-FLOW test case (d3d_lake_2d): a lake forced by an uniform wind field. This example is a twin experiment with spatially correlated 2D-noise added to the wind field. The noise is created on cartesian (equidistant grid), the noise realisations are written by OpenDA to the <lake2d_windx_noise.amu>. The 2D noise field is interpolated at computational grid by D-Flow FM and added to the uniform wind field. The <SequentialSimulationNoise.oda> can be used to perform a single D-Flow FM run where wind with noise is used as forcing. The resulting <lake2d_his.nc> file can be used as synthetic observations by the <stochObserver>.

21.6.6 Example 6: EnKF with the DCSM v5 model and uncertainty on the wind direction

This example is converted from the SIMONA DCSM v5 model with spatially correlated 2D-noise added to the wind field.

References

- Baptist, M. J., 2005. *Modelling floodplain biogeomorphology*. Ph.D. thesis, Delft University of Technology.
- Barenblatt, G. I., M. Bertsch, R. Dal Passo, V. M. Prostokishen and M. Ughi, 1993. "A mathematical model of turbulent heat and mass transfer in stably stratified shear flow." *Journal of Fluid Mechanics* 253: 341–358.
- Baumert, H. and G. Radach, 1992. "Hysteresis of Turbulent Kinetic Energy in Non-rotational Tidal Flows: A Model Study." *Journal of Geophysical Research* 97 (C3): 3669–3677.
- Beckers, J. M., H. Burchard, J. M. Campin, E. Deleersnijder and P. P. Mathieu, 1998. "Another reason why simple discretizations of rotated diffusion operators cause problems in ocean models: comments on "isoneutral diffusion in a z co-ordinate ocean model""." *American Meteorological Society* 28: 1552–1559.
- Bijker, E. W., 1967. *Some considerations about scales for coastal models with moveable bed*. Tech. Rep. 50, WL | Delft Hydraulics, Delft, The Netherlands.
- Blumberg, A. F. and G. L. Mellor, 1985. "Modelling vertical and horizontal diffusivities with the sigma co-ordinate system." *Monthly Weather Review* 113 (8): 1379.
- Burchard, H. and H. Baumert, 1995. "On the performance of a mixed-large model based on the k-epsilon turbulence closure." *Journal of Geophysical Research* 100 (C5): 8523–8540.
- Charnock, H., 1955. "Wind-stress on a water surface." *Q. J. Royal Meteorol. Soc.* 81: 639–640.
- Christoffersen, J. B. and I. G. Jonsson, 1985. "Bed friction and dissipation in a combined current and wave motion." *Ocean Engineering* 12 (5): 387–423.
- Colebrook, C., 1939. "Turbulent flow in pipes, with particular reference to the transition region between the smooth and rough pipe laws." *Journal of the Institution of Civil Engineers* 11 (4): 133–156.
- Colebrook, C. and C. White, 1937. "Experiments with Fluid Friction in Roughened Pipes." *Proceedings of the Royal Society of London, Series A, Mathematical and Physical Sciences* 161 (906): 367–381.
- D-Flow FM TRM, 2015. *D-Flow Flexible Mesh Technical Reference Manual*. Deltires, Delft, 1.1.124 ed.
- D-Morphology UM, 2019. *D-Morphology User Manual*. Deltires, 1.5 ed.
- D-WAQ PLCT UM, 2015. *D-Water Quality Processes Library Configuration Tool User Manual*. Deltires, 0.01 ed.
- D-WAQ TRM, 2013. *D-Water Quality Technical Reference Manual*. Deltires, 4.00 ed.
- Davies, A. G., R. L. Soulsby and H. L. King, 1988. "A numerical model of the combined wave and current bottom boundary layer." *Journal of Geophysical Research* 93 (C1): 491–508.
- Davies, A. M. and H. Gerritsen, 1994. "An intercomparison of three-dimensional tidal hydrodynamic models of the Irish Sea." *Tellus* 46A: 200–221.
- Deltires, 2019. *D-Real Time Control User Manual*. Deltires. D-Real Time Control (D-RTC) in Delta Shell.

- Dijkstra, Y. M., 2014. *Turbulence modelling in environmental flows. Improving the accuracy of the k-ε model by a mathematical transformation*. Master's thesis, Delft University of Technology. Faculty of Civil Engineering and Geosciences, Department of Environmental Fluid Mechanics.
- Dingemans, M. W., 1997. *Water Wave Propagation over Uneven Bottoms, Vol. 1 and 2*. Advanced Series on Ocean Engineering, Vol. 13. World Scientific, London.
- Dingemans, M. W., A. C. Radder and H. J. de Vriend, 1987. "Computation of the driving forces of wave-induced currents." *Coastal Engineering* 11: 539–563.
- Eckart, C., 1958. "Properties of water, Part II. The equation of state of water and sea water at low temperatures and pressures." *American Journal of Science* 256: 225–240.
- Fredsøe, J., 1984. "Turbulent boundary layer in wave-current interaction." *Journal of Hydraulic Engineering* 110: 1103–1120.
- Gill, A. E., 1982. *Atmosphere-Ocean dynamics*, vol. 30 of *International Geophysics Series*. Academic Press.
- Grant, W. D. and O. S. Madsen, 1979. "Combined wave and current interaction with a rough bottom." *Journal of Geophysical Research* 84 (C1): 1797–1808.
- Groeneweg, J., 1999. *Wave-current interactions in a generalized Lagrangian mean formulation*. Delft University of Technology, Delft, The Netherlands. Ph.D. thesis.
- Haney, R. L., 1991. "On the pressure gradient force over steep topography in sigma coordinate models." *Journal of Physical Oceanography* 21: 610–619.
- Hirsch, C., 1990. *Numerical computation of internal and external flows*. John Wiley & Sons, New York.
- Huang, W. and M. Spaulding, 1996. "Modelling horizontal diffusion with sigma coordinate system." *Journal of Hydraulic Engineering* 122 (6): 349–352.
- Huynh-Thanh, S. and A. Temperville, 1991. "A numerical model of the rough turbulent boundary layer in combined wave and current interaction." In R. L. Soulsby and R. Bettes, eds., *Sand transport in rivers, estuaries and the sea*, pages 93–100. Balkema Rotterdam.
- Hwang, P., 2005a. "Comparison of the ocean surface wind stress computed with different parameterization functions of the drag coefficient." *J. Oceanogr.* 61: 91–107.
- Hwang, P., 2005b. "Drag coefficient, dynamic roughness and reference wind speed." *J. Oceanogr.* 61: 399–413.
- Kalkwijk, J. P. T. and R. Booij, 1986. "Adaptation of secondary flow in nearly horizontal flow." *Journal of Hydraulic Research* 24 (1): 19–37.
- Karypis, G., 2013. *METIS - A Software Package for Partitioning Unstructured Graph, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices, Version 5.1.0*. Tech. rep., Department of Computer Science and Engineering, University of Minnesota.
- Klopstra, D., H. J. Barneveld and J. M. Van Noortwijk, 1996. *Analytisch model hydraulische ruwheid van overstroomde moerasvegetatie*. Tech. Rep. PR051, HKV consultants, Lelystad, The Netherlands. Commissioned by Rijkswaterstaat/RIZA, The Netherlands.
- Klopstra, D., H. J. Barneveld, J. M. Van Noortwijk and E. H. van Velzen, 1997. "Analytical model for hydraulic roughness of submerged vegetation." In *The 27th IAHR Congress, San Francisco, 1997; Proceedings of Theme A, Managing Water: Coping with Scarcity and Abundance*, pages 775–780. American Society of Civil Engineers (ASCE), New York.

- Kolmogorov, A. N., 1942. "Equations of turbulent motion in incompressible fluid." *Izv. Akad. Nauk. SSR, Seria fizicheska Vi No.1 2 (1-2)*: 56–58. English translation: 1968 Imperial College, Mech. Eng. Dept. Rept. ON/6.
- Lane, A., 1989. *The heat balance of the North Sea*. Tech. Rep. 8, Proudman Oceanographic Laboratory.
- Leendertse, J. J., 1990. "Turbulence modelling of surface water flow and transport: part IVa." *Journal of Hydraulic Engineering* 114 (4): 603–606.
- Love, A. E. H., 1927. *A Treatise on the Mathematical Theory of Elasticity*. Cambridge University Press, 4th ed.
- Manning, R., 1891. "On the flow of water in open channels and pipes." *Transactions of the Institution of Civil Engineers of Ireland* 20: 161–207.
- Millero, F. J. and A. Poisson, 1981. "International one-atmosphere equation of state of sea water." *Deep-Sea Research* 28A (6): 625–629.
- Myrhaug, D. and O. H. Slaattelid, 1990. "A rational approach to wave-current friction coefficients for rough, smooth and transitional turbulent flow." *Coastal Engineering* 14: 265–293.
- O'Connor, B. A. and D. Yoo, 1988. "Mean bed friction of combined wave-current flow." *Coastal Engineering* 12: 1–21.
- Octavio, K. A. H., G. H. Jirka and D. R. F. Harleman, 1977. *Vertical Heat Transport Mechanisms in Lakes and Reservoirs*. Tech. Rep. 22, Massachusetts Institute of Technology.
- Peckham, S. D., E. W. Hutton and B. Norris, 2013. "A component-based approach to integrated modeling in the geosciences: The design of CSDMS." *Computers & Geosciences* 53: 3–12.
- Phillips, N. A., 1957. "A co-ordinate system having some special advantages for numerical forecasting." *Journal of Meteorology* 14: 184–185.
- Postma, L., G. S. Stelling and J. Boon, 1999. "Three-dimensional water quality and hydrodynamic modelling in Hong Kong. Stratification and water quality." In *Proceedings of the 2nd International Symp. on Environmental Hydraulics, Hong Kong, December 1998*, pages 43–49. Balkema, Rotterdam.
- Prandtl, L., 1945. "Über ein neues Formelsystem für die ausgebildete Turbulenz." *Nachrichten von der Akademie der Wissenschaften in Göttingen. Mathematisch-Physikalische Klasse* pages 6–19.
- RGFGRID UM, 2016. *Delft3D-RGFGRID User Manual*. Deltares, 5.00 ed.
- Rijn, L. C. van, 1984. "Sediment transport, Part III: bed form and alluvial roughness." *Journal of Hydraulic Engineering* 110 (12): 1733–1754.
- Rijn, L. C. van, 2007. "Unified View of Sediment Transport by Currents and Waves. I: Initiation of Motion, Bed Roughness, and Bed-Load Transport." *Journal of Hydraulic Engineering* 133 (6): 649–667.
- Rijn, L. C. van, D. R. Walstra and M. v. Ormondt, 2004. *Description of TRANSPOR2004 and implementation in Delft3D-ONLINE*. Tech. Rep. Z3748.10, WL | Delft Hydraulics, Delft, The Netherlands.
- Rodi, W., 1984. "Turbulence models and their application in Hydraulics, State-of-the-art paper article sur l'état de connaissance." IAHR Paper presented by the IAHR-Section on Fundamentals of Division II: Experimental and Mathematical Fluid Dynamics, The Netherlands.

- Ryan, P. J., D. R. F. Harleman and K. D. Stolzenbach, 1974. "Surface Heat Loss From Cooling Ponds." *Water Resources Research* 10 (5): 930–938.
- Schrama, E., 2007. "Tides. Lecture Notes AE4-876."
- Slørdal, L. H., 1997. "The pressure gradient force in sigma-co-ordinate ocean models." *International Journal Numerical Methods In Fluids* 24: 987–1017.
- Smith, S. D. and E. G. Banke, 1975. "Variation of the sea surface drag coefficient with wind speed." *Quarterly Joournal of the Royal Meteorological Society* 101: 665–673.
- Soulsby, R. L., A. G. Davies, J. Fredsøe, D. A. Huntley, I. G. Jonnson, D. Myrhaug, R. R. Simons, A. Temperville and T. J. Zitman, 1993a. "Bed shear stresses due to combined waves and currents." In *Abstracts-in-depth of the Marine Science and Technology G8-M overall workshop, Grenoble.*, pages 2.1-1/2.1-4.
- Soulsby, R. L., L. Hamm, G. Klopman, D. Myrhaug, R. R. Simons and G. P. Thomas, 1993b. "Wave-current interaction within and outside the bottom boundary layer." *Coastal Engineering* 21: 41–69.
- Speziale, C. G., R. Abid and E. C. Anderson, 1992. "Critical evaluation of two-equation models for near-wall turbulence." *AIAA Journal* 30: 324–331.
- Stelling, G. S. and J. A. T. M. van Kester, 1994. "On the approximation of horizontal gradients in sigma co-ordinates for bathymetry with steep bottom slopes." *International Journal Numerical Methods In Fluids* 18: 915–955.
- Swart, 1974. *Offshore sediment transport and equilibrium beach profiles*. Ph.D. thesis, Delft University of Technology, Delft, The Netherlands. Delft Hydraulics Publ. 131.
- Sweers, H. E., 1976. "A nomogram to estimate the heat exchange coefficient at the air-water interface as a function of windspeed and temperature; a critical survey of some literature." *Journal of Hydrology* 30: –.
- Uittenbogaard, R. E., J. A. T. M. van Kester and G. S. Stelling, 1992. *Implementation of three turbulence models in 3D-TRISULA for rectangular grids*. Tech. Rep. Z81, WL | Delft Hydraulics, Delft, The Netherlands.
- UNESCO, 1981a. *Background papers and supporting data on the international equation of state 1980*. Tech. Rep. 38, UNESCO.
- UNESCO, 1981b. *The practical salinity scale 1978 and the international equation of state of seawater 1980*. Tech. Rep. 36, UNESCO. Tenth report of the Joint Panel on Oceanographic Tables and Standards (1981), (JPOTS), Sidney, B.C., Canada.
- Velzen, E. H. van, P. Jesse, P. Cornelissen and H. Coops, 2003. *Stromingsweerstand vegetatie in uiterwaarden*. Tech. Rep. 2003.029, Rijkswaterstaat/RIZA.
- Winterwerp, J. C. and R. E. Uittenbogaard, 1997. *Sediment transport and fluid mud flow*. Tech. Rep. Z2005, WL | Delft Hydraulics, Delft, The Netherlands.

A The master definition file

The mdu-file contains the key information of the flow model. Besides the names of the relevant user specified files, such as the grid file and the external forcings file, the values of various model parameters should be specified in the MDU-file. The most important model parameter settings are given in [Table A.1](#) as well as the associated default setting for these parameters.

The basename of the mdu-file, without .mdu, is also used as the model identification string, and is often denoted as *mdu_name* throughout this User Manual. It is equivalent to Delft3D-FLOW's *runid* concept.

Table A.1: Standard MDU-file with default settings.

Keyword	Default setting	Description
[model]		
Program	D-Flow FM	Program
Version	1.2.60 ← .64623M	Version number of computational kernel
MDUFormatVersion	1.08	File format version. Do not edit this.
GuiVersion	1.5.4.45545	Version number of GUI
AutoStart	0	Autostart simulation after loading MDU or not (0=no, 1=autostart, 2=autostartstop).
PathsRelativeToParent	0	Whether or not (1/0) to resolve file names (e.g. inside the *.ext file) relative to their direct parent, instead of to the toplevel MDU working dir.
[geometry]		
NetFile	*_net.nc. See Appendix B .	
BathymetryFile	*.xyz	
DryPointsFile	Dry points file *.xyz, third column dummy z values, or polygon file *.pol.	
StructureFile	File *.ini containing list of hydraulic structures. See section C.11	
WaterLevIniFile	Initial water levels sample file *.xyz	
LandBoundaryFile	Only for plotting	
ThinDamFile	*_thd.pli, Polyline(s) for tracing thin dams.	
FixedWeirFile	*_fxw.pliz, Polyline(s) x,y,z, z = fixed weir top levels (formerly fixed weir)	
VertplizFile	*_vlay.pliz), = pliz with x,y, Z, first Z =nr of layers, second Z = laytyp	
ProflocFile	*_proflocation.xyz) x,y,z, z = profile refnumber	
ProfdefFile	*_profdefinition.def) definition for all profile nrs	
ProfdefxyzFile	*_profdefinition.def) definition for all profile nrs	
ManholeFile	File containing manholes (e.g. *.dat)	
PartitionFile	*_part.pol, polyline(s) x,y	
Uniformwidth1D	2.	Uniform width for 1D profiles not specified bij profloc
WaterLevIni	0.	Initial water level
Bedlevuni	-5.	Uniform bed level, (only if bedlevtype>=3, used at missing z values in netfile
Bedslope	0.	bedslopeinclination, sets zk = bedlevuni + x*bedslope ans sets zbnzd = xbnzd*bedslope
BedlevType	3	1: at cell center (tiles xz,yz,bl,bob=max(bl)), 2: at face (tiles xu,yu,blu,bob=blu), 3: at face (using mean node values), 4: at face (using min node values), 5: at face (using max node values), 6: with bl based on node values
Blmeanbelow	-999.	if not -999d0, below this level the cell centre bedlevel is the mean of surrounding netnodes
Blminabove	-999.	if not -999d0, above this level the cell centre bedlevel is the min of surrounding netnodes
AngLat	0.	Angle of latitude S-N (deg), 0=no Coriolis
AngLon	0.	Angle of longitude E-W (deg), 0=Greenwich
Conveyance2D	-1	-1:R=HU,0:R=H, 1:R=A/P, 2:K=analytic-1D conv, 3:K=analytic-2D conv

(continued on next page)

Keyword	Default setting	Description
<i>(continued from previous page)</i>		
Nonlin2D	0	Non-linear 2D volumes, only i.c.m. ibedlevtype = 3 and Conveyance2D>=1
Sillheightmin	0.0	Weir treatment only if both sills larger than this value (m)
Makeorthocenters	0	1=yes, 0=no switch from circumcentres to orthocentres in geominit
Dcenterinside	1.	limit cell center; 1.0:in cell <-> 0.0:on c/g
Bamin	1.d-6	Minimum gridcell area, i.c.m. cutcells
OpenBoundaryTolerance	3.	Search tolerance factor between boundary polyline and grid cells. Unit: in cell size units (i.e., not metres).
Kmx	0	Max nr of vertical layers
Layertype	1	1= all sigma, 2 = all z, 3 = use VertplizFile
Numtopsig	0	Nr of sigmalayers in top of Zlayer model
SigmaGrowthFactor	1.	layer thickness growth factor from bed up
[numerics]		
CFLMax	0.7	Max. Courant nr.
AdvecType	33	Adv type, 0=no, 1= Wenneker, qu-udzt, 2=1, q(ui0-u), 3=Perot q(ui0-u), 4=Perot q(ui-u), 5=Perot q(ui-u) without itself
TimeStepType	2	0=only transport, 1=transport + velocity update, 2=full implicit step_reduce, 3=step_jacobi, 4=explicit
Limtyphu	0	Limiter type for waterdepth in continuity eq., 0=no, 1=minmod,2=vanLeer,3=Kooren,4=Monotone Central
Limtypmom	4	Limiter type for cell center advection velocity, 0=no, 1=minmod,2=vanLeer,3=Kooren,4=Monotone Central
Limtypsa	4	Limiter type for salinity transport, 0=no, 1=minmod,2=vanLeer,3=Kooren,4=Monotone Central
Icgssolver	4	Solver type , 1 = sobekGS_OMP, 2 = sobekGS_OMPthreadsafe, 3 = sobekGS, 4 = sobekGS + Saadilud, 5 = parallel/global Saad, 6 = parallel/Petsc, 7 = parallel/GS
Maxdegree	6	Maximum degree in Gauss elimination
FixedWeirScheme	9	6 = semi-subgrid scheme, 8 = Tabellenboek, 9 = Villemonde
FixedWeirContraction	1.	flow width = flow width*FixedWeirContraction
Izbndpos	0	Position of z boundary, 0=D3Dflow, 1=on net boundary, 2 = on specified polyline
Tlfsmo	0.	Fourier smoothing time on waterlevel boundaries (s)
Slopedrop2D	0.	Apply droplosses only if local bottom slope > Slope-drop2D, <=0 =no droplosses
Chkadvd	0.1	Check advection terms if depth < chkadvdp, => less setbacks
Teta0	0.55	Theta of time integration, 0.5 < Theta < 1d0
Qhrelax	0.01	Relaxation qhbnd ()
cstbnd	0	Delft-3D type velocity treatment near boundaries for small coastal models (1) or not (0)
Maxitverticalforestersal	0	0 : no vertical filter, > 0 = Max nr of iterations
Maxitverticalforesterterm	0	0 : no vertical filter for temp, > 0 = Max nr of iterations
Turbulencemode1	3	0=no, 1 = constant, 2 = algebraic, 3 = k-eps, 4 = k-tau
Turbulenceadvection	3	0=no, 3 = hor. expl., vert. impl.
AntiCreep	0	Flag for AntiCreep option (0=off, 1=on)
Maxwaterleveldiff	0.	upper bound (in m) on water level changes, <= 0: no bounds
Maxvelocitydiff	0.	upper bound (in m/s) on velocity changes, <= 0: no bounds
Epshu	0.0001	Input for threshold water depth for wet and dry cells
[physics]		
UnifFrictCoef	0.023	Uniform friction coefficient, 0=no friction
UnifFrictType	1	0=Chezy, 1=Manning, 2=White-Colebrook, 3=idem, WAQUA style

(continued on next page)

Keyword	Default setting	Description
<i>(continued from previous page)</i>		
UnifFrictCoef1D	0.023	Uniform friction coefficient in 1D links, 0=no friction
UnifFrictCoefLin	0.	Uniform linear friction coefficient for ocean models (m/s), 0=no
Umodlin	0	linear friction umod , only for ifrctyp=4,5,6
Vicouv	1.	Uniform horizontal eddy viscosity (m2/s)
Dicouv	1.	Uniform horizontal eddy diffusivity (m2/s)
Vicoww	5.d-5	Uniform vertical eddy viscosity (m2/s)
Dicoww	5.d-5	Uniform vertical eddy diffusivity (m2/s)
Vicwmnb	0.	Minimum visc in prod and buoyancy term (m2/s)
Smagorinsky	0.	Add Smagorinsky horizontal turbulence : vicu = vicu + ((Smagorinsky*dx)**2)*S, e.g. 0.2
Elder	0.	Add Elder contribution : vicu = vicu + Elder*kappa*ustar*H/6), e.g. 1.0
irov	0	0=free slip, 1 = partial slip using wall_ks
wall_ks	0.	Nikuradse roughness for side walls, wall_z0=wall_ks/30
Rhomean	1000.	Average water density (kg/m3)
Idensform	2	1=Eckart, 2=UNESCO
Ag	9.813	Gravitational acceleration
TidalForcing	0	Tidal forcing (0=no, 1=yes) (only for jsferic == 1)
Doodsonstart	55.565	TRIWAQ = 55.565D0 , D3D = 57.555D0
Doodsonstop	375.575	TRIWAQ = 375.575D0 , D3D = 275.555D0
Doodsoneps	0.0	TRIWAQ = 0.0 400 cmps , D3D = 0.03 60 cmps
Salinity	0	Include salinity, (0=no, 1=yes)
InitialSalinity	0.	Initial salinity concentration (ppt)
Sal0abovezlev	-999.	Salinity 0 above level (m)
DeltaSalinity	-999.	for testcases
BackgroundSalinity	30.	Background salinity concentration (ppt)
Temperature	0	Include temperature, (0=no, 1=only transport, 5=heat flux model (5) of D3D), 3=excess model of D3D
InitialTemperature	6.	Initial temperature (degC)
Secchidepth	2.	Water clarity parameter (m)
Stanton	-1	Coefficient for convective heat flux (), if negative, then Cd wind is used
Dalton	-1	Coefficient for evaporative heat flux (), if negative, then Cd wind is used
SecondaryFlow	0	Secondary flow (0=no, 1=yes)
BetaSpiral	0	Coefficient for secondary flow
<hr/>		
[wind]		
ICdtyp	2	(),1=const, 2=S&B 2 breakpoints, 3= S&B 3 breakpoints, 4=Charnock constant (), e.g. 0.00063 0.00723
Cdbreakpoints	6.3d-4 7.23d-3	
Windspeedbreakpoints	0. 100.	(m/s), e.g. 0.0 100.0
Rhoair	1.205	Air density (kg/m3)
PavBnd	0.	Average air Pressure on open boundaries, (N/m2), only applied if value > 0
PavIni	0.	Initial air Pressure, (N/m2), only applied if value > 0
<hr/>		
[time]		
RefDate	20010101	Reference date (yyyymmdd)
Tzone	0.	Data Sources in GMT are interrogated with time in minutes since refdat-Tzone*60
Tunit	S	Time units in MDU (D, H, M or S)
DtUser	300.	User timestep in seconds (interval for external forcing update & his/map output)
DtMax	30.	Max timestep in seconds
DtInit	1.	Initial timestep in seconds
TStart	0.	Start time w.r.t. RefDate (in TUnit)
TStop	8640000.	Stop time w.r.t. RefDate (in TUnit)
<hr/>		
[restart]		
<i>(continued on next page)</i>		

Keyword	Default setting	Description
<i>(continued from previous page)</i>		
RestartFile		Restart file, only from netcdf-file, hence: either *_rst.nc or *_map.nc
RestartDateTime		Restart time (YYYYMMDDHHMMSS), only relevant in case of restart from *_map.nc
<hr/>		
[externalforcing]		
ExtForceFile		Old format for external forcings file *.ext, link with tim/cmp-format boundary conditions specification
ExtForceFileNew		New format for external forcings file *.ext, link with bc-format boundary conditions specification. See section C.5.2.
<hr/>		
[trachytopes]		
TrtRou	N	Flag for trachytopes (Y=on, N=off)
TrtDef		
TrtL		
DtTrt	60.	Interval for updating of bottom roughness due to trachytopes in seconds
<hr/>		
[output]		
Wrishp_crs	0	Writing cross sections to shape file (0=no, 1=yes)
Wrishp_weir	0	Writing weirs to shape file (0=no, 1=yes)
Wrishp_gate	0	Writing gates to shape file (0=no, 1=yes)
Wrishp_fwx	0	Writing fixed weirs to shape file (0=no, 1=yes)
Wrishp_thd	0	Writing thin dams to shape file (0=no, 1=yes)
Wrishp_obs	0	Writing observation points to shape file (0=no, 1=yes)
Wrishp_emb	0	Writing embankments file (0=no, 1=yes)
Wrishp_dryarea	0	Writing dry areas to shape file (0=no, 1=yes)
Wrishp_enc	0	Writing enclosures to shape file (0=no, 1=yes)
Wrishp_src	0	Writing sources and sinks to shape file (0=no, 1=yes)
Wrishp_pump	0	Writing pumps to shape file (0=no, 1=yes)
OutputDir		Output directory of map-, his-, rst-, dat- and timings-files, default: DFM_OUTPUT_<modelname>. Set to . for no dir/current dir.
WAQOutputDir		Output directory of Water Quality files
FlowGeomFile		*_flowgeom.nc Flow geometry file in NetCDF format.
ObsFile		Space separated list of files, containing information about observation points. See section F.2.2.
CrsFile		Space separated list of files, containing information about observation cross sections. See section F.2.4.
HisFile		*_his.nc History file in NetCDF format.
HisInterval	120.	History output, given as 'interval' 'start period' 'end period' (s)
XLSInterval	0.	Interval (s) between XLS history
MapFile		*_map.nc Map file in NetCDF format.
MapInterval	1200.	Map file output, given as 'interval' 'start period' 'end period' (s)
RstInterval	86400.	Restart file output, given as 'interval' 'start period' 'end period' (s)
MapFormat	4	Map file format, 1: NetCDF, 2: Tecplot, 3: NetCFD and Tecplot, 4: NetCDF UGRID
Wrihis_balance	1	Write mass balance file, 1=yes, 0=no
Wrihis_structure_gen	1	Write general structures, 1=yes,0=no
Wrihis_structure_dam	1	Write dams, 1=yes,0=no
Wrihis_structure_pump	1	Write pumps, 1=yes,0=no
Wrihis_structure_gate	1	Write gates, 1=yes,0=no
Wrihis_structure_weir	1	Write weirs, 1=yes,0=no
Wrimap_waterlevel_s0	1	Write water levels at old time level, 1=yes,0=no
Wrimap_waterlevel_s1	1	Write water levels at new time level, 1=yes,0=no
Wrimap_velocity_component_u0	1	Write velocities at old time level, 1=yes,0=no
Wrimap_velocity_component_u1	1	Write velocities at new time level, 1=yes,0=no
Wrimap_velocity_vector	1	Write velocity vectors, 1=yes,0=no

(continued on next page)

Keyword	Default setting	Description
<i>(continued from previous page)</i>		
Wrimap_upward_velocity_\leftrightarrow component	0	Write upward velocity component, 1=yes,0=no
Wrimap_density_rho	1	Write density, 1=yes,0=no
Wrimap_horizontal_\leftrightarrow viscosity_viu	1	Write horizontal viscosity, 1=yes,0=no
Wrimap_horizontal_\leftrightarrow diffusivity_diu	1	Write horizontal diffusivity, 1=yes,0=no
Wrimap_flow_flux_q1	1	Write fluxes, 1=yes,0=no
Wrimap_spiral_flow	1	Write spiral flow, 1=yes,0=no
Wrimap_numlimdt	1	Write numlimdt, 1=yes,0=no
Wrimap_taucurrent	1	Write bottom friction, 1=yes,0=no
Wrimap_chezy	1	Write chezy values, 1=yes,0=no
Wrimap_turbulence	1	Write turbulence, 1=yes,0=no
Wrimap_wind	1	Write winds, 1=yes,0=no
Wrimap_heat_fluxes	0	Write heat fluxes, 1=yes,0=no
MapOutputTimeVector		File (.mpt) containing fixed map output times (s) w.r.t. RefDate
FullGridOutput	0	0:compact, 1:full time-varying grid data
EulerVelocities	0	Write Eulerian velocities, 1=yes,0=no
ClassMapFile		Name of class map file
WaterlevelClasses	0.	Series of values between which water level classes are computed
WaterdepthClasses	0.	Series of values between which water depth classes are computed
ClassMapInterval	0	Interval (in s) between class map file outputs
WaqInterval	0.	Interval (in s) between DELWAQ file outputs
StatsInterval	0.	Interval (in s) between simulation statistics output
Writebalancefile	0.	Write Balancefile, 1=yes, 0=no
TimingsInterval	0.	Timings output interval
Richardsononoutput	0.	Write Richardson number, 1=yes,0=no

B The net file for flexible meshes

The net file contains the full computational grid (mesh) information. It is a NetCDF file that can contain 2D grids, as well as 1D networks, and combinations of these. The NetCDF conventions used for this file are:

- ◊ CF-conventions (≥ 1.7) and UGRID-1.0 for the 2D grids. More information on: <http://ugrid-conventions.github.io/ugrid-conventions/>.

B.1 2D grids in NetCDF UGRID files

D-Flow FM reads net files that adhere to the 2D rules in the UGRID conventions. At the minimum this means that the file must contain the mesh node coordinates and for all grid cells ('faces') the corner node numbers in the face_node_connectivity table.

C Attribute files

C.1 Introduction

In the following sections we describe the attribute files used in the input file (MDU-file) of D-Flow FM. Most of these files contain the quantities that describe one specific item, such as the location of open boundaries, or time dependent data of fluxes discharged in the model area by discharge stations. Most of the attribute files can be generated by the Delta Shell GUI after defining an input scenario. Some files can almost only be generated by utility programs such as the unstructured grid generated by RGFGRID. Still, we describe both type of files as it might be useful to know how the input data is structured to be able to generate (large) files, such as astronomic boundary conditions, or time-series for wind speed and direction by client specific tools.

C.2 Polyline/polygon file

D-Flow FM uses the same format for polylines and (closed) polygons. When used as a polygon file, there is *not* the requirement that the first and last point should be identical. It is good modelling practice to name files containing polygons with the extension `.pol`, and files containing polylines with the extension `.pli`. When the polylines have a third column with *z*-coordinates, the extension `.pliz` is advised.

File contents	The coordinates of one or more polylines. Each polyline (piecewise linear) is written in a single block of data.
Filetype	ASCII
File format	Free formatted
Filename	<code><name.pol></code>
Generated	RGFGRID, QUICKIN, Delta Shell, etc

Record description:

Record	Record description
	Preceding description records, starting with an asterisk (*), and will be ignored.
1	A non blank character string, starting in column one.
2	Two integers N_r, N_c representing the numbers of rows and number of columns for this block of data.
	Two reals representing the x, y or λ, ϕ -coordinate, followed by remaining data values at that location (if $N_c > 2$).

Example:

```
*  
* Polyline L007  
*  
L007  
6  2  
132400.0    549045.0  
132345.0    549030.0  
132165.0    549285.0  
131940.0    549550.0  
131820.0    549670.0
```

```
131585.0      549520.0
*
* Polyline L008
*
L008
4  2
131595.0      549685.0
131750.0      549865.0
131595.0      550025.0
131415.0      550175.0
*
* Polyline L009
*
L009
6  2
131595.0      549655.0
148975.0      564595.0
150000.0      564935.0
152105.0      565500.0
153150.0      566375.0
154565.0      567735.0
```

C.3 Sample file

Sample file are used for several of D-Flow FM's input files.

File contents	The location and value of samples.
Filetype	ASCII
File format	Free formatted
Filename	<name.xyz>
Generated	Manually or Offline with QUICKIN or Delta Shell and data from digitised charts or GIS-database.

Record description:

Filetype	Record description
Free formatted	Location and sample value per row Two reals representing the x, y or λ, ϕ -coordinate and one real representing the sample value

Example:

Sample file with 12 sample values with their location (free formatted file).

```
213813.2      603732.1      -4.053000
214686.0      607226.1      -4.522000
214891.7      610751.2      -5.000000
210330.8      601424.1      -2.169000
211798.0      604444.8      -2.499000
212460.0      607475.7      -2.760000
212436.9      610362.5      -2.865000
185535.4      606607.9      1.360000
186353.0      603789.4      1.122000
187959.2      601197.6      0.9050000
190193.0      599101.5      0.7050000
208578.7      602513.7      -0.7990000
```

C.4 Time series file (ASCII)

Time series files are used for several of D-Flow FM's input files. There is no header, except for optional comment lines. There should be two or more columns: the first column contains time in minutes since the model's reference date, all remaining columns contain the data values. Each line contains a single time with its (space-uniform) values.

C.5 The external forcings file

Two definition files for the external forcings are supported, each with their own format.

C.5.1 Old style external forcings

The name of the file is specified in the MDU-file as "ExtForceFile". External forcings are specified in blocks key-value pairs, e.g.

```
* comments
* comments

QUANTITY =waterlevelbnd
FILENAME =tfl_01.pli
FILETYPE =9
METHOD =3
OPERAND =0

QUANTITY = ...
FILENAME = ...
...
```

The format is not case-sensitive, but key-value pairs are expected in the *particular order*

The majority of the keywords speak for themselves or are optional. The operand needs some explanation. This keyword determines the operation to be performed with a new connected forcing for a quantity already set by another forcing from a different source: '0' to replace the existing value with the new one and '+' to add the new to the existing value. Though generally applicable to all external forcings, it is most commonly used when working with spatial fields (such as wind and atmospheric pressure) originating from different types of sources. For instance a spatially uniform wind field could be set as a timeseries, but overwritten locally by a values interpolated from regional atmospheric model output, upon which a cyclone is superimposed from a spiderweb-type file. This means that the EXT-file likely has three entries for quantity wind in the aforementioned order: a uniform timeseries with an operand '0', a NetCDF file with operand '0' and a spiderweb file with operand '+'. The NetCDF file might not cover the entire flow grid. In the uncovered regions, the value from the uniform wind is left unaltered. The same is done in the case of the spiderweb file with operand '0'. Regions for which data is missing in spatial fields are treated as non-existent in the file (currently no interpolation is by default active to fill gaps, therefore it is highly recommendable to complete these files in a preprocessing step). In NetCDF files, missing data are recognized as values equal to the fill value designated in the _FillValue-attribute. In files of the Curvi-type (ascii) it is the NODATA-value in the header.

A comprehensive list of keywords is given in the table below.

quantity	character	Name of the quantity, see the list below
filename	character	File associated with this forcing
sourcemask*	character	File containing a mask
filetype	integer	Indication of the filetype: 1. Time series (C.4) 2. Time series magnitude and direction 3. Spatially varying weather 4. ArcInfo 5. Spiderweb data (cyclones) (C.12.3) 6. Curvilinear data (C.12.2) 7. Samples (C.3) 8. Triangulation magnitude and direction 9. Polyline (<*.pli>-file, C.2) 11. NetCDF grid data (e.g. meteo fields) 14. NetCDF wave data
method	integer	Method of interpolation: 1. Pass through (no interpolation) 2. Interpolate time and space 3. Interpolate time and space, save weights 4. Interpolate space 5. Interpolate time 7. Interpolate/Extrapolate time
operand	character	Overwriting or superimposing values already set for this quantity: 'O' Values are overwritten. '+' New value is superimposed.
value*	float	custom coefficients for transformation
factor*	float	
ifrctyp*	float	
averagingtype*	float	
relativesearchcellsize*	float	
extrapoltol*	float	
area*	float	Area for source/sink



Note: Keywords marked with * are optional

C.5.2 New style external forcing (boundary conditions only)

The name of this file is specified using the keyword `ExtForceFileNew` in the MDU-file (see [Appendix A](#)). External forcings are specified in the ini-file format. This file may contain one or more [Boundary] blocks, each followed by a set of keywords that specify the location and forcing for that forcing. See the following section for details.

C.5.2.1 Boundary definitions

Table C.1: Boundary definitions in new style external forcing file.

Keyword	Type	Default	Description
<i>(Part of new-style external forcings file, see Table C.1.)</i>			
<i>(continued on next page)</i>			

Keyword	Type	Default	Description
<i>(continued from previous page)</i>			
[Boundary] quantity	String		<i>Repeat as needed</i> Name of the quantity, see the list in Section C.5.3 (boundaries only)
locationFile	String		Name of boundary polyline <*.pli>.
forcingFile	String		Name of file containing the forcing for this boundary. The file may either be a <bc> file (see section E.2.3) or a NetCDF-file (see section E.2.4).
returnTime (return_time earlier versions)	Double in	0	(Optional) Thatcher-Harleman (section 9.4.4) return time. The default value 0 means that Thatcher-Harleman is disabled.

Example:

```
* comments
[Boundary]
quantity      = waterlevelbnd
locationFile  = tfl_01.pli
forcingFile   = tfl_01.bc

[Boundary]
quantity      = ...
...
```

C.5.3 Accepted quantity names

The list of accepted quantity names is subdivided into six categories:

- ◊ Boundary conditions
- ◊ Meteorological fields
- ◊ Structure parameters
- ◊ Initial fields
- ◊ Spatial physical properties
- ◊ Miscellaneous

Table C.2: List of accepted external forcing quantity names.

Quantity	pg.	Description
Boundary conditions (old and new ext):		
waterlevelbnd	137	Water level
neumannbnd	140	Water level gradient
riemannbnd	140	Riemann invariant
outflowbnd		
velocitybnd	139	Velocity
dischargebnd	138	Discharge
riemann_velocitybnd		Riemann invariant velocity
salinitybnd	171	Salinity
temperaturebnd	171	Temperature
sedimentbnd	see ¹	Suspended sediment

¹D-Morphology UM (2019)

Quantity	pg.	Description
uxuyadvectionvelocitybnd normalvelocitybnd, tangentialvelocitybnd qhbnd tracerbnd< <i>tracername</i> >	145 142 171	Normal and tangential velocity Discharge-water level dependency User-defined tracer
Meteorological fields: (old ext only) windx, windy, windxy airpressure_windx_windy atmosphericpressure rainfall humidity_airtemperature... ..._cloudiness humidity_airtemperature... ..._cloudiness_solarradiation discharge_salinity... ..._temperature_sorsin	201 201 201 201 170	Wind components, wind vector Atmospheric pressure and wind components Atmospheric pressure Precipitation Discharge, salinity and heat sources
Structure parameters: (old ext only) pump damlevel gatelowerededgelevel generalstructure	217	Use preferred structure INI file instead (See Section C.11). Pump capacity
Initial fields: (old ext only) initialwaterlevel initialsalinity initialsalinitytop initialtemperature initialverticaltemperatureprofile initialverticalsalinityprofile initialtracer< <i>tracername</i> >	327 327 327 327 327 327 327	
Spatial physical properties: (old ext only) frictioncoefficient horizontaleddyviscositycoefficient horizontaleddydiffusivitycoefficient advectiontype ibotlevtype	328	
Miscellaneous: shiptxy movingstationxy	340	Moving observation point for output (time,x,y)

C.6 Trachytopes

The trachytype functionality allows for the usage of different types of roughness formulations at different locations within the computational domain. Multiple formulation may be active in the same grid cell. Several keywords in the MDU file influence the functioning. All keywords below should be placed underneath the [trachytypes] section in the MDU file.

Keyword	Value	Description	Default
TrtRou	Y or N	Trachytope option activated	N
TrtDef	<name.ttd>	Definition file trachytopes	
TrtL	<name.arl>	Area Roughness on Link file trachytopes	
DtTrt	pos. real	Time step in seconds for updating roughness and resistance coefficients based on trachytopes. Must be a multiple of DtUser.	1 DtUser
TrtMnH	pos. real	Minimum water depth in roughness computation	.2 EpsHu
TrtMth	1 or 2	Area averaging method	1
TrtAsr	real ∈ [0, 1]	Serial factor in averaging of area roughnesses	0.6
TrtMxR	integer	Maximum recursion depth for mixed trachytope definitions	8

C.6.1 Area Roughness on Links (ARL-file)

File contents

The Area Roughness on Links file (ARL-file) is the input file for the spatial distribution of the alluvial and vegetation roughness which are handled by the trachytopes module.

Filetype

ASCII

File format

Space separated file format

Filename

<name.arl>

Generated

At present: Conversion possible from structured Delft3D-FLOW input files with matlab conversion tool from Open Earth Tools (<http://www.openearth.info/>), see example below ([section C.6.1.2](#)).

The ARL file has the following properties:

- ◊ A single line comment, starts with a hash tag “#” or an asterisk “*”.
- ◊ Each line has the format
“xu yu zu TrachytopesNr Fraction”,
 - where xu, yu, zu is the coordinate of the midpoint of the netlink,
 - TrachytopesNr is an integer corresponding to the number as described in the TrachyTopes Definition File (cf. [section C.6.2](#)),
 - and Fraction is a Fraction between 0.0 and 1.0
- ◊ A line with a midpoint-netlink-coordinate which is the same as the preceding line (comments not considered) allows multiple types of trachytopes roughness definitions, provided the sum of the Fraction keywords does not add to a value greater than 1.0.
- ◊ If the sum is less than 1.0 the background roughness prescribed in the [physics] chapter in the MDU file is prescribed for the remaining Fraction, cf. [Appendix A](#). Only a single roughness type is allowed in combination with the Trachytopes module.
- ◊ If a midpoint-netlink-coordinate is repeated in the .arl file with the midpoint-netlink-coordinate xu, yu, zu in the preceding line being different, all previous instances of the specific xu, yu, zu are ignored.

C.6.1.1 Example

An example of the <arl> file is given below based on the <ttd> file given in section C.6.2. In this case the netlink with u point located at $x_u = 10.542$ and $y_u = 11.6$, which is covered for 30 %, 30 % 20 % and 20 % for TrachytypeNr = 1, 2, 4 and 3 respectively.

(continued)

```
...
10.542    11.6    0     1     0.3
10.542    11.6    0     2     0.3
10.542    11.6    0     4     0.2
10.542    11.6    0     3     0.2
...
```

(continued)

C.6.1.2 Conversion from Delft3D 4 input files

Open Earth Tools has different matlab tools available for the conversion of Delft3D-FLOW models to D-Flow FM models (e.g. [dflowfmConverter.m](#) and [d3d2dflowfm.m](#)).

An extra Matlab conversion script has been added to convert Delft3D-FLOW's <*.aru> and <*.arv> files to <*.arl> files for D-Flow FM:

[d3d2dflowfm_friction_trachytypes.m](#).

It can be called as follows:

```
oetsettings
d3d2dflowfm_friction_trachytypes(filgrd,filaru,filarv,filnet,filarl)
```

where the variables are defined as follows:

filgrd	Delft3D-FLOW grid filename (*.grd)
filaru	Delft3D-FLOW area definition file in u-direction (*.aru)
filarv	Delft3D-FLOW area definition file in v-direction (*.arv)
filnet	name of the dflowfm network file (*_net.nc)
filarl	name of the dflowfm trachytype file (*.arl))

C.6.2 Trachytype Definition file (TTD-file)

The trachytype definition file contains lines of the following format defining the different types of trachytopes. The types which are supported are general, discharge dependent and water-level dependent formats.

C.6.2.1 General format

The general format is formatted as follows:

```
TrachytypeNr      FormulaNr      ...Parameters...
```

where ...Parameters... indicates a space separated list of formula specific parameters; the parameters required and their order are specified in section C.6.2.5. The user must specify for each trachytype (combination of formula number and parameters) a unique positive trachytype number. This trachytype number is used in the area files (see section C.6.1) to indicate the roughness types on a net link.

C.6.2.2 Example

An example of such a file where the first three codes 1–3 with a Nikuradse roughness height are defined, and the fourth (code 4) is Chézy roughness height.

```
1      51      0.1
2      51      0.2
3      51      0.3
4      52      35
```

C.6.2.3 Discharge dependent format

The discharge dependent format is formatted as follows:

```
TrachytopeNr    DISCHARGE    "Cross-section name"
TrachytopeNr    Q1          FormulaNr   ...Parameters...
TrachytopeNr    Q2          FormulaNr   ...Parameters...
TrachytopeNr    Q3          FormulaNr   ...Parameters...
TrachytopeNr    Q4          FormulaNr   ...Parameters...
...
...
```

which again expects a user defined TrachytopeNr. The first row contains the keyword “DISCHARGE” and subsequently a cross-section name occurring in CrsFile in the mdu-file.

The cross-section name can be enclosed by quotation marks, for instance when the cross-section name is made up of more than one word. The subsequent rows all have the same TrachytopeNr as the first row and furthermore every FormulaNr should be the same. The list of discharges Q1, Q2, Q3, Q4 should be monotonically increasing.

The ...Parameters... for each formula type are specified in [section C.6.2.5](#).

C.6.2.4 Water level dependent format

The water-level dependent format is similar to the discharge dependent format and is formatted as follows:

```
TrachytopeNr    WATERLEVEL   "Observation-station name"
TrachytopeNr    ZS1         FormulaNr   ...Parameters...
TrachytopeNr    ZS2         FormulaNr   ...Parameters...
TrachytopeNr    ZS3         FormulaNr   ...Parameters...
TrachytopeNr    ZS4         FormulaNr   ...Parameters...
...
...
```

which again expects a user defined TrachytopeNr. The first row contains the keyword “WATERLEVEL” and subsequently an observation-station name occurring in ObsFile in the mdu-file. The observation-station name can be enclosed by quotation marks, for instance when the observation-station name is made up of more than one word. The subsequent rows all have the same TrachytopeNr as the first row and furthermore every FormulaNr should be the same. The list of waterlevels ZS1, ZS2, ZS3, ZS4 should be monotonically increasing.

The ...Parameters... for each formula type are specified in [section C.6.2.5](#).

C.6.2.5 Supported roughness formulations

The roughness formulations specified in the table below are supported by D-Flow FM. These formulations are specified in the .ttd by `FormulaNr` and `...Parameters...` in the order in which they appear in the table below. The related formulae are presented in the Technical Reference Manual

FormulaNr	Description	Parameters
Special classes (1–50)		
1	flood protected area, reduces the effective area of the grid cell. It has no influence on the continuity equation (i.e. it does not decrease the surface area of the grid cell).	–
2	composite trachytype: fraction α of type T_1 and fraction β (generally $\beta = 1 - \alpha$) of type T_2	T_1, T_2, α, β
Area trachytype classes: simple type (51–100)		
51	constant White-Colebrook/Nikuradse value	k [m]
52	constant Chézy value	C [$m^{1/2}/s$]
53	constant Manning value	n [$s/m^{1/3}$]
54	constant z_0 value	z_0 [m]
61	constant White-Colebrook/Nikuradse values for ebb and flood	k_{ebb} [m], k_{flood} [m]
62	constant Chézy values for ebb and flood	C_{ebb} [$m^{1/2}/s$], C_{flood} [$m^{1/2}/s$]
63	constant Manning values for ebb and flood	n_{ebb} [$s/m^{1/3}$], n_{flood} [$s/m^{1/3}$]
64	constant z_0 values for ebb and flood	$z_{0,ebb}$ [m], $z_{0,flood}$ [m]
Area trachytype classes: alluvial type (101–150)		
101	simplified Van Rijn	A [$m^{0.3}$], B [$m^{0.3}$]
102	power relation	A [$m^{1/2}/s$], B [-]
103 ²	Van Rijn predictor	-
104 ²	Struiksma predictor	A_1 [$m^{1/2}/s$], A_2 [-], θ_c [-], θ_m [-], C_{min} [$m^{1/2}/s$]
105 ²	bedforms quadratic	-
106 ²	bedforms linear	-
Area trachytype classes: vegetation type (151–200)		
151	Barneveld 1	h_v [m], n [1/m]
152	Barneveld 2	h_v [m], n [1/m], C_D [-], k_b [m]
153	Baptist 1	h_v [m], n [1/m], C_D [-], C_b [$m^{1/2}/s$]
154	Baptist 2	h_v [m], n [1/m], C_D [-], C_b [$m^{1/2}/s$]
Linear trachytype classes: various (201–250)		
201	hedges 1	h_v [m], n [1/m]
202	hedges 2	h_v [m], n [1/m]
Linear trachytype classes: various (251–300)		
251	trees	h_v [-], C_D [-]

² The alluvial roughness predictors 103, 104, 105 and 106 are not yet supported, because the coupling with the morphology module is not yet available.

C.7 Fixed weirs

A fixed weir has many quantities. On a polyline several quantities have to be specified. In the table below an overview is given of all these quantities.

Keyword	Description	Default value
X-coordinate	X-coordinate of polyline point	-
y-coordinate	Y-coordinate of polyline point	-
Crest level	Absolute crest level	- [mAD]
Ground height left	Difference between crest level and toe level at left side	0.0 [m]
Ground height right	Difference between crest level and toe level at right side	0.0 [m]
Crest width	Width of weir in perpendicular direction	3.0 [m]
Slope left	Slope of weir at left side	4.0 [-]
Slope right	Slope of weir at right side	4.0 [-]
Roughness code	Roughness code for vegetation on weir	0.0 [-]
Weir type	(v)illemonde or (t)abellenboek	v [-]

in which 'mAD' stands for 'meter Above Datum', which denotes a level relative to the vertical reference system. The last column with the weir type (Villemonde or Tabellenboek) is optional. In this way, individual polyline can be given another weir type. For example, if FixedweirType=9 (Villemonde) has been specified in the MDU-file, then individual polylines can be set to Tabellenboek by adding a "t" to all points in the polyline; see the example below.

The default value for the ground height levels (left and right) is 0.0 m. However, in case of the Tabellenboek a minimum value for the ground heights of 0.1 m is applied, because the empirical Tabellenboek relation doesn't allow values smaller than 0.1 m. Note that in WAQUA also a minimum value of 0.1 m is applied in case of the Tabellenboek. For fixed weirs also a shape file can be generated, see [section 5.4.12](#). In case of the Tabellenboek the ground heights in the shape file can be slightly different from the value in the input file, because a minimum value of 0.1 m is applied.

C.7.0.1 Example

An example of a polyline with fixed weir input is given below. It consists of a polyline with two points

```
(continued)
...
weir
 2      10
 399.999420    99.997688    2.0    0.2    0.8    3.0    4.0    4.0    0    t
 399.999420   100.999542    2.1    0.3    0.9    3.0    4.0    4.0    0    t

...
(continued)
```

In general, fixed weir polylines do not coincide with the computational grid. The polylines are snapped to flow links, which is illustrated in [Figure 5.16](#). This is computed by the computational kernel of D-Flow FM.

In practice, it is possible that a certain flow link contains multiple snapped fixed weirs. This is for example the case when polylines with fixed weirs cross each other. Then, the fixed weir with the highest crest level is taken, because the crest level is used in the drying-flooding algorithm. In this way, overtopping of a weir the water level only occurs if the water level is above the weir with the highest crest level.

C.8 Calibration Factors

The calibration factor functionality is a multiplier for the roughness at different locations within the computational domain. Multiple formulation may be active in the same grid cell. Several keywords in the MDU file influence the functioning. All keywords below should be placed under the [calibration] section in the MDU file.

Keyword	Value	Description	Default
UseCalibration	1 or 0	Calibration factor option activated	N
DefinitionFile	<name.cld>	Calibration factor definition file	
AreaFile	<name.cll>	Calibration factor area file	

C.8.1 Calibration factor definition file (CLD-file)

The calibration class definition file contains lines of the following format defining the different calibration classes. It supports constant values and values that depend on the discharge at a named cross section or the water level at a named location.

C.8.1.1 Header of the CLD-file

The file starts with the following header

```
# [FileInformation]
#   FileType      = CalibrationFactorsDefinitionFile
#   FileVersion   = 1.0
# [CalibrationFactors]
```

C.8.1.2 Constant values

The format for constant values is as follows:

```
CalibrationClassNr  ConstantValue
```

where ConstantValue indicates the calibration value to be used (use 1.0 to use the uncalibrated roughness). The user must specify for each calibration class (line in this file) a unique calibration class number CalibrationClassNr. This calibration class number is used in the calibration area <.cll> file to indicate the area of influence of this class.

C.8.1.3 Discharge dependent format

The discharge dependent line is formatted as follows:

```
CalibrationClassNr DISCHARGE "Cross-section name"
CalibrationClassNr Q1 ConstantValueAtQ1
CalibrationClassNr Q2 ConstantValueAtQ2
CalibrationClassNr Q3 ConstantValueAtQ3
```

```
CalibrationClassNr Q4 ConstantValueAtQ4
...

```

which again expects a user defined calibration class number `CalibrationClassNr`. The first line contains the keyword `DISCHARGE` and subsequently a cross-section name occurring in the `CrsFile` in the `mdu`-file. The cross-section name can be enclosed by quotation marks, for instance when the name contains spaces. The subsequent lines all start with the same calibration class number as the first line. The list of discharges `Q1`, `Q2`, `Q3`, and `Q4` should be monotonically increasing. For each discharge value a calibration value should be specified. Between the discharges specified, the calibration values are linearly interpolated. Below the minimum and above the maximum discharge the first and last values are used respectively.

C.8.1.4 Water level dependent format

The water-level dependent format is similar to the discharge dependent format and is formatted as follows:

```
CalibrationClassNr WATERLEVEL "Observation-station name"
CalibrationClassNr ZS1 ConstantValueAtZS1
CalibrationClassNr ZS2 ConstantValueAtZS2
CalibrationClassNr ZS3 ConstantValueAtZS3
CalibrationClassNr ZS4 ConstantValueAtZS4
...

```

which again expects a user defined calibration class number `CalibrationClassNr`. The first line contains the keyword `WATERLEVEL` and subsequently an observation-station name occurring in `ObsFile` in the `mdu`-file. The observation-station name can be enclosed by quotation marks, for instance when the name contains spaces. The subsequent lines all start with the same calibration class number as the first line. The list of water levels `ZS1`, `ZS2`, `ZS3`, and `ZS4` should be monotonically increasing. For each water level value a calibration value should be specified. Between the water levels specified, the calibration values are linearly interpolated. Below the minimum and above the maximum water level the first and last values are used respectively.

C.8.1.5 Example

The example file given below defines three calibration classes. In the area associated with the first class, the roughness values are used without any adjustment. In the area associated with the second class, the roughness value is decreased by 20%. In the area associated with the third and final class, the increase in the roughness value varies between 0% and 30% depending on the discharge at a specific location.

```
# [FileInformation]
#   FileType      = CalibrationFactorsDefinitionFile
#   FileVersion   = 1.0
# [CalibrationFactors]

# areas without any calibration adjustment
1 1.0

# areas with 20% decreased roughness value
2 0.8

# area with calibration depending on discharge at crsX
10 DISCHARGE 'crsX'
10 500 1.0
10 1000 1.1
10 3000 1.3
```

C.8.2 Calibration Class Area on Links (CLL-file)

C.8.2.1 Header of the CLL-file

The file starts with the following header

```
# [FileInformation]
#   FileType      = CalibrationAreasFile
#   FileVersion   = 1.0
# [CalibrationAreas]
```

C.8.2.2 Body of the CLL-file

The body of the CLL-file consists of comment lines (any line starting with # or *) or data lines of the following format:

```
xu yu zu CalibrationClassNr AreaFraction
```

where xu, yu, zu is the coordinate of the midpoint of the netlink, CalibrationClassNr is an integer corresponding to the number as described in the Calibration Class Definition File (cf. [section C.8.1](#)), and AreaFraction is an area fraction between 0.0 and 1.0.

All lines that list the same xu, yu, zu coordinates are combined to a weighted average of multiple calibration classes, provided the sum of the fractions does not add to a value greater than 1.0.

If the sum of the areas specified for a certain netlink is less than 1.0, then the a constant calibration factor of 1.0 is assumed for the remaining area (i.e. no calibration effect). The result is that if for a specific netlink no line is specified at all, then the roughness of this netlink will be unaffected by the calibration process.

C.8.2.3 Example

The example of the <CLL>-file is given below based on the <CLD>-file given in [section C.8.1](#). In this case the roughness at netlink with midpoint located at xu = 10.542 and yu = 11.6 is completely affected by calibration class 1, the netlink with midpoint at xu = 11.120 and yu = 12.1 is affected for 60% by calibration class 1 and 40% by calibration class 10, and the final location listed uses only the value resulting from calibration class 10.

```
...
10.542 11.6 0 1 1.0
11.120 12.1 0 1 0.6
11.120 12.1 0 10 0.4
12.635 12.6 0 10 1.0
...
```

C.9 Sources and sinks

Sources and sinks ([section 8.8](#)) are defined as part of the <*.ext> file, as follows:

```
QUANTITY=discharge_salinity_temperature_sorsin
FILENAME=chan1_westeast.pli  # A file 'chan1_westeast.tim', with same basename
                           # as the polyline should be present
FILETYPE=9
METHOD =1
```

```
OPERAND =0
AREA     =1.5
```

The `<*.pli>` file is a polyline file ([section C.2](#)) with two or three columns (for 2D or 3D models, respectively). Along with the `<*.pli>`-file, there has to be a time series file ([section C.4](#)) with the same basename as the `<*.pli>`-file, and extension `<.tim>`. The columns in the `<*.tim>` are as follows: time in minutes, discharge Q in [m^3/s], and *all* constituents in the model. The order of constituents is: salinity in [ppt], temperature T in [$^\circ\text{C}$], sediment concentrations, spiral flow intensity and tracers. For example, if we only have temperature and two tracers we get 5 columns: time, discharge, temperature, tracer 1, tracer 2. For more information, see [section 8.8](#).

C.10 Dry points and areas

Dry points can either be defined by a basic sample file (see [section C.3](#)), or a polygon file (see [section C.2](#)).

Special attention must be given to the optional third column of a polygon file: when the first point has a z -value of -1 , the polygon mask is inverted, i.e., all points outside the polygon are set as dry points.

More details can be found in [section 5.4.2.7](#).

C.11 Structure INI file

The structure INI file is the preferred format for hydraulic structure input, and supersedes some of the old structure quantity names in [Table C.2](#) under "Structure parameters".

```
[structure]
type          = weir           # Type of structure
id            = weir01         # Name of the structure
polylinefile  = weir01.pli    # *.pli; Polyline location for structure
crest_level   = weir01_crest.tim # Crest level in [m]

[structure]
type          = gate           # Type of structure
id            = gate01         # Name of the structure
polylinefile  = gate01.pli    # *.pli; Polyline location for structure
sill_level    = gate01_crest.tim # Sill (crest) level in [m]
sill_width    = 39.5           # (Optional) sill width in [m], between
                               # the fixed side walls.
door_height   = 8.54           # Gate door height in [m], used for
                               # detecting flow across door.
lower_edge_level = gate01_ledge.tim # Position of gate's lower edge in [m],
                                     # used for flow underneath door.
opening_width  = gate01_opening.tim # (Optional) opening width between two
                                     # sideways closing gate doors.
horizontal_opening_direction = from_left/from_right/symmetric # (Optional) Opening
                                                               # direction of the gate door(s).

[structure]
type          = pump           # Type of structure
id            = pump01         # Name of the structure
polylinefile  = pump01.pli    # *.pli; Polyline location for structure
capacity      = pump01_cap.tim # Pumping capacity in [m3/s]
```

C.12 Space varying wind and pressure

In many cases the space varying wind data is provided by a meteorological station. This data is often defined on a different grid than the computational grid used in D-Flow FM. Translating these files into files defined on the grid of the computational engine is often a lengthy process and can result in huge files. This feature facilitates the reading of the meteorological data on its own grid and interpolates the data internally to the grid of D-Flow FM. D-Flow FM can handle three types of meteorological input:

- 1 Time- and space-varying wind on an equistant grid
- 2 Time- and space-varying wind on a curvilinear grid
- 3 Time- and space-varying wind on a Spiderweb grid
- 4 Time- and space-varying wind on a NetCDF file

C.12.1 Meteo on equidistant grids

File contents	Time-series of a space varying wind and atmospheric pressure defined on an equidistant (Cartesian or spherical) grid.
File format Generated	Free formatted or unformatted, keyword based. Some offline program.



Remark:

- ◊ The keywords are case insensitive.

Header description for the wind velocity files:

Keywords	Value	Description
FileVersion	1.03	version of file format
Filetype	meteo_on_equidistant_grid	meteo input on equidistant grid
NODATA_value	free	value used for input that is to be neglected
n_cols	free	number of columns used for wind datafield
n_rows	free	number of rows used for wind datafield
grid_unit	m or degree	unit of distances on the grid in both <i>x</i> - and <i>y</i> -direction
x_llcorner	free	<i>x</i> -coordinate of lower left corner of lower left grid cell (in units specified in <i>grid_unit</i>)
y_llcorner	free	<i>y</i> -coordinate of lower left corner of lower left grid cell (in units specified in <i>grid_unit</i>)
x_llcenter	free	<i>x</i> -coordinate of centre of lower left grid cell (in units specified in <i>grid_unit</i>)
y_llcenter	free	<i>y</i> -coordinate of centre of lower left grid cell (in units specified in <i>grid_unit</i>)
dx	free	gridsize in <i>x</i> -direction in units specified in <i>grid_unit</i>
dy	free	gridsize in <i>y</i> -direction in units specified in <i>grid_unit</i>

Keywords	Value	Description
n_quantity	1	number of quantities specified in the file
quantity1	x_wind or y_wind	the velocity component given in unit unit1
unit1	m s-1	unit of quantity1: metre/second

The user must specify the location of the equidistant grid on which the meteorological data is specified. If one has the location of the lower left corner of the lower left grid cell, one can specify the starting point of the grid using keywords `x_llcorner` and `y_llcorner`. If one has the location of the cell centre of the lower left grid cell, one should use the keywords `x_llcenter` and `y_llcenter`. Using the first option, the first data value is placed at $(x_{llcorner} + \frac{1}{2}dx, y_{llcorner} + \frac{1}{2}dy)$, which is the cell centre of cell (1,1). Using the latter option, the first data value is placed at $(x_{llcenter}, y_{llcenter})$, which is again the cell centre of cell (1,1), i.e. the data values are always placed at the cell centres of the meteorological grid. Note that the lower left grid cell is defined to be the grid cell with index (1,1). When using the option of meteorological data on a separate curvilinear grid, the origin and orientation of the data set can be chosen freely with respect to the grid on which it is specified, see [section C.12.2](#) for details.

Time definition and data block description for the wind velocity files

Keywords	Value	Description
Time	<i>fixed format described below</i>	time definition string

The time definition string has a fixed format, used to completely determine the time at which a dataset is valid. The time definition string has the following format:

TIME *minutes/hours since YYYY-MM-DD HH:MM:SS TIME ZONE*, e.g.

360 minutes since 2008-07-28 10:55:00 +01:00

The format of the string is completely fixed. No extra spaces or tabs can be added between the different parts of the definition. The time definition is followed by the datablock of input values corresponding to the specified time. The data block contains values for the wind velocity in *x*- or *y*-direction for *n_cols* by *n_rows* points, starting at the top left point. The time definition and the data block are repeated for each time instance of the time-series.

The atmospheric pressure file

The header for the atmospheric pressure is similar to that of the wind velocity files, except for the following differences.

Keywords	Value	Description
quantity1	air_pressure	air pressure
unit1	Pa or mbar	unit of quantity1: Pascal or millibar

The specification of the time definition and the data block is fully conform the wind velocity files.

File version and conversion

The current description holds for `FileVersion` 1.03. The table below shows the latest modifications in the file format (and version number).

FileVersion	Modifications
1.03	Use of keyword <code>Value_pos</code> to indicate the position of the lower left corner of the grid replaced by use of the combination of keywords: <code>x_llcorner</code> and <code>y_llcorner</code> or <code>x_llcenter</code> and <code>y_llcenter</code>
1.02	No changes for this meteo input type, but for the meteo type <code>meteo_on_spiderweb_grid</code>
1.01	Changed keyword <code>MeteoType</code> to <code>FileType</code> Changed fixed value of input type (Keyword <code>Filetype</code>) from <code>ArclInfo</code> to <code>meteo_on_equidistant_grid</code>

**Restrictions:**

- ◊ The contents of the file will not be checked on its domain.
- ◊ Keywords are followed by an equal sign '=' and the value of the keyword.
- ◊ When a keyword has value `free`, the value of this keyword is free to choose by the user. When only one value is given for a keyword, this keyword has a fixed value and when 2 or more options are shown, the user can choose between those values.
- ◊ Times must be specified exactly according to the time definition. See the examples shown in this section.
- ◊ The atmospheric pressure file must use the same grid definition and time frame as the files for the wind velocity components.
- ◊ The unit of the meteo grid must be the same as the computational grid, i.e. both with `grid_unit` = [m] or both with `grid_unit` = [degree].
- ◊ Input items in a data block are separated by one or more blanks.
- ◊ The wind components are specified at the cell centres (water level points) of the numerical grid.
- ◊ The wind components are specified in the west-east (`x_wind`) and south-north directions (`y_wind`).

**Remarks:**

- ◊ The time definition in the meteo files contains the number of minutes or hours since a reference date and time in a certain time zone. The reference time and time zone may differ from those of the simulation. During a simulation the computational engine will search in the meteo file for the current simulation time and interpolate between neighbouring times if necessary. Possible differences in time zone will be accounted for by shifting the meteo input data. The reference times within the time definition string may vary in a meteo file, i.e. it is possible to attach new input with a different reference time, behind the last data block. Consecutive times must always be increasing in the input file.
- ◊ Comments can be added after pound signs (#). These are not read.

Example of a file containing wind in x-direction (west-east)

The data blocks in this example are the result of the following FORTRAN statements:

```
do j = nrows,1,-1
  write(out,*) (xwind(i,j),i=1,ncols)
enddo
```

The `x-wind` velocity file for a 3 (`n_cols`) by 4 (`n_rows`) grid has the following layout:

```
FileVersion      = 1.03
filetype        = meteo_on_equidistant_grid
NODATA_value   = -999.000
```

```

n_cols      =      3
n_rows      =      4
grid_unit   =    degree
x_llcenter  = -12.000
y_llcenter  =    48.000
dx          =   0.12500
dy          = 0.083333333
n_quantity  =      1
quantity1   = x_wind
unit1       = m s-1
TIME =      0.0 hours since 2008-01-15 04:35:00 +00:00
2 3.0 3.6
3 4.5 2
2.2 1 2.3
1.2 0.7 -0.4
TIME =      6.0 hours since 2008-01-15 04:35:00 +00:00
-1.1 -2.3 -3.6
-3.2 0.8 1.1
2.2 -1 -1.6
1.2 -0.7 -0.4

```

This results in an *x*-component of wind velocity given - in [m/s] - on a spherical, 3 by 4, equidistant grid, with grid sizes given by *dx* and *dy* (in degrees) and where the centre point of the lower left cell of the grid lies in (longitude, latitude) (-12.0, 48.0) on the globe. Data is given at two times: 0 and 6 hours since January 15th, 2008, 4:35 AM, in UTC+0.

Remark:



- ◊ The layout of the data blocks is from north to south (whereas most of the other files in Delft3D-FLOW, such as the curvilinear grid file, are ordered from south to north).
- Usually the signal corresponds to Mean Sea Level. One actually wants to prescribe an input signal corresponding to the local pressure prescribed by the space varying meteo input. To this end, it is possible to specify an average pressure - which should correspond to your input signal on the open boundaries - which is then used to determine local pressure gradients that need to be applied along the open boundaries to obtain an input signal that is consistent with the local atmospheric pressure. Using this keyword one can specify an average pressure that is used on all open boundaries, independent of the type of wind input. The pressure must be specified in N/m². An example:

C.12.2 Curvilinear data

File contents	Time-series of a space varying wind and atmospheric pressure defined on a curvilinear (Cartesian or spherical) grid.	
File format	Free formatted or unformatted, keyword based.	
Generated	Some offline program.	

Remark:



- ◊ The keywords are case insensitive.

Header description for the wind velocity files:

Keywords	Value	Description
FileVersion	1.03	version of file format
Filetype	meteo_on_curvilinear_grid	meteo input on curvilinear grid
NODATA_value	free	value used for input that is to be neglected
grid_file	free.grd	name of the curvilinear grid file on which the data is specified
first_data_value	grid_llcorner or	see example below

Keywords	Value	Description
	grid_ulcorner or grid_lrcorner or grid_urcorner	
data_row	grid_row or grid_column	see example below
n_quantity	1	number of quantities specified in the file
quantity1	x_wind or y_wind	the velocity component given in unit unit1
unit1	m s-1	unit of quantity1: metres/second

Time definition and data block description for the wind velocity files

For a description of the time definition and data block see [section C.12.1](#).

The atmospheric pressure file

For a description of the atmospheric file see [section C.12.1](#).

File version and conversion

The current description holds for `FileVersion` 1.03. The table below shows the latest modifications in the file format (and version number).

FileVersion	Modifications
1.03	Fixed bug in interpolation of data from meteo grid to computational grid: Conversion script mirrored data set erroneously. This was treated correctly by meteo module. Fixed both the conversion script and the meteo module together: Required modification in meteo input file: Change <code>first_data_value = grid_llcorner</code> into <code>grid_ulcorner</code> or vice versa or Change <code>first_data_value = grid_lrcorner</code> into <code>grid_urcorner</code> or vice versa
1.02	No changes for this meteo input type, but for the meteo type <code>meteo_on_spiderweb_grid</code>
1.01	Changed keyword <code>MeteoType</code> to <code>FileType</code> Changed keyword <code>Curvi_grid_file</code> to <code>Grid_file</code> Changed fixed value of input type (Keyword <code>Filetype</code>) from <code>Curvi</code> to <code>meteo_on_curvilinear_grid</code>



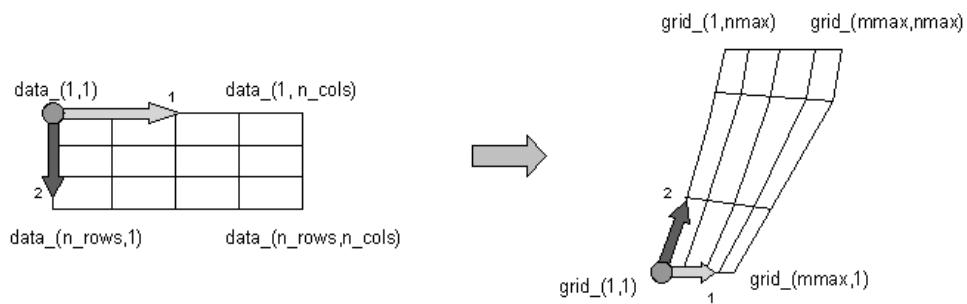
Restrictions:

- ◊ The restrictions for space varying wind and pressure on a separate curvilinear grid are the same as for space varying wind and pressure on an equidistant grid, described in [section C.12.1](#). A difference is that the data values on the curvilinear grid are not specified in the cell centres, but in the grid points (cell corners).
- ◊ The unit of the meteo grid must be the same as the computational grid, i.e. both with `grid_unit = [m]` or both with `grid_unit = [degree]`.



Remark:

- ◊ The remarks for space varying wind and pressure on a separate curvilinear grid are the same as for space varying wind and pressure on an equidistant grid, described in [section C.12.1](#).



1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20



16	17	18	19	20
11	12	13	14	15
6	7	8	9	10
1	2	3	4	5

Figure C.1: Illustration of the data to grid conversion for meteo input on a separate curvilinear grid

Example:

A file for input of *x*-velocity (in west-east direction) on a 4 (*n_rows*) by 5 (*n_cols*) curvilinear grid, where the meteorological data is mirrored vertically with respect to the grid, has the following layout:

```

FileVersion      = 1.03
filetype        = meteo_on_curvilinear_grid
NODATA_value   = -999.000
grid_file       = curviwind.grd
first_data_value = grid_llcorner
data_row        = grid_row
n_quantity     = 1
quantity1      = x_wind
unit1          = m s-1
TIME = 0.0 minutes since 1993-06-28 14:50:00 -02:00
 1 2 3 4 5
 6 7 8 9 10
11 12 13 14 15
16 17 18 19 20
TIME = 600.0 minutes since 1993-06-28 14:50:00 -02:00
 1 2 3 4 5
 6 7 8 9 10
11 12 13 14 15
16 17 18 19 20

```

This results in an *x*-component of velocity given - in [m/s] - on the curvilinear grid specified in file <curviwind.grd>. The data set will be mirrored such that the first value of the data (upper left corner, in the example '1') corresponds to the lower left corner of the grid (point (1,1)) and a row of data corresponds to a row on the grid, see [Figure C.1](#). Data is given at two times: 0 and 600 minutes since June 28th, 1993, 14:50 PM, in UTC-2.

C.12.3 Spiderweb data**Remarks:**

- ◊ The spiderweb file format used in D-Flow FM forms a subset of the the format described below. The extensive format is accepted by D-Flow FM, but not all keywords are required and/or meaningful.
- ◊ The keywords `grid_unit` and `air_pressure_reference` are ignored.
- ◊ The keyword `n_quantity` is ignored; the number of quantities is always 3.
- ◊ The keywords `quantity1`, `quantity1` and `quantity1` are ignored, the order of the variables in the file is assumed to be wind speed, wind direction and pressure drop.
- ◊ The keywords `unit1` and `unit2` are ignored.
- ◊ The keyword `unit3` is optional; if omitted or different from '`mbar`', Pa is silently assumed

Cyclone winds are governed by a circular motion, combined with a cyclone track. This type of wind is generally very difficult to implement on a curvilinear grid. This feature facilitates the reading of the so-called Spiderweb files and interpolates the wind and pressure data internally to the computational grid. A special feature of the space varying wind and pressure on the Spiderweb grid is that it can be combined with one of the other meteorological input options described in this manual, i.e. to either uniform wind and pressure, or to one of the space varying wind and pressure options, see [section C.12](#).

File contents	Time-series of a space varying wind and atmospheric pressure defined on a Spiderweb grid. This grid may be specified in Cartesian or spherical coordinates.
File format	Free formatted or unformatted, keyword based.
Generated	Some offline program.

**Remarks:**

- ◊ The keywords are case insensitive.
- ◊ Space varying wind and pressure on a Spiderweb grid is added to other wind input and the wind fields are interpolated and combined in and around the cyclone.

Header description of the Spiderweb wind and pressure file:

Keywords	Value	Description
<code>FileVersion</code>	1.03	version of file format
<code>Filetype</code>	<code>meteo_on_spiderweb_grid</code>	meteo input on Spiderweb grid
<code>NODATA_value</code>	<code>free</code>	value used for input that is to be neglected
<code>n_cols</code>	<code>free</code>	number of gridpoints in angular direction
<code>n_rows</code>	<code>free</code>	number of gridpoints in radial direction
<code>grid_unit</code>	<code>m</code> or <code>degree</code>	unit of the Spiderweb grid
<code>spw_radius</code>	<code>free</code>	radius of the spiderweb given in units given by <code>spw_rad_unit</code>
<code>spw_rad_unit</code>	<code>m</code>	unit of the Spiderweb radius
<code>spw_merge_frac</code>	[0.0,1.0]	fraction of the Spiderweb radius where merging starts of the background wind with the Spiderweb wind. Default is 0.5
<code>air_pressure_reference</code>	<code>air_pressure_default_from_computational_engine</code>	Both keyword and value are too long to fit on one line.

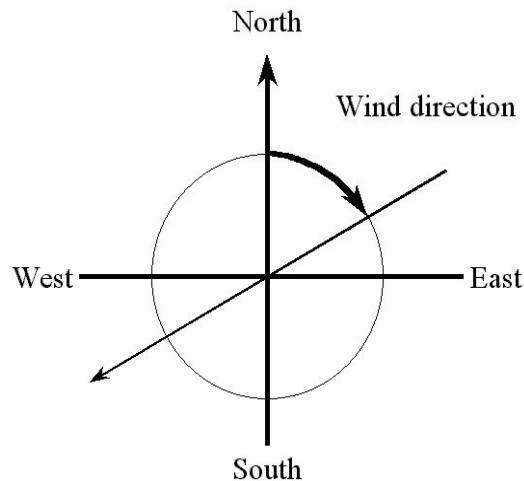


Figure C.2: Wind definition according to Nautical convention

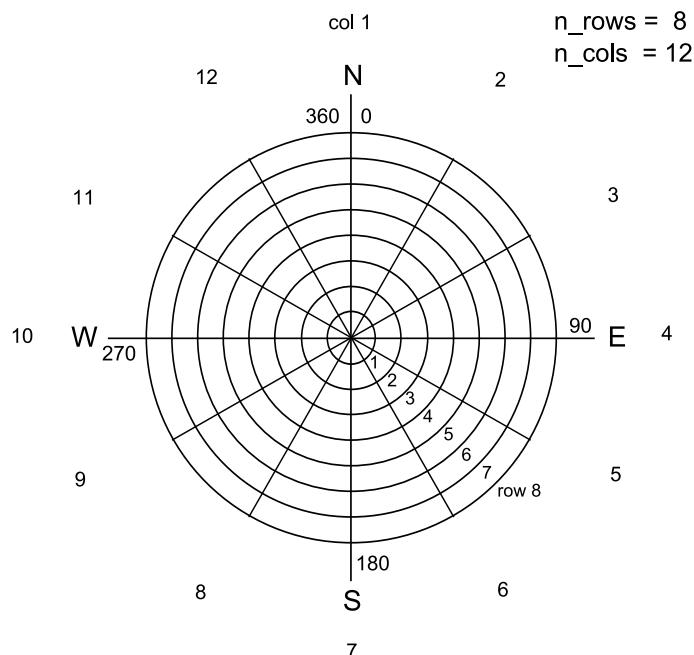
Keywords	Value	Description
	<i>or free</i>	Reference value related to p_drop is the default air pressure of the computational engine <i>or</i> the value specified. If missing, p_drop is extracted from the actual atmospheric pressure.
n_quantity	3	number of quantities specified in the file
quantity1	wind_speed	wind speed given in unit unit1
quantity2	wind_from_direction	direction where the wind is coming from given in unit unit2
quantity3	p_drop	drop in atmospheric pressure given in unit unit3
unit1	m s-1	unit of quantity1: metres/second
unit2	degree	unit of quantity2: degrees
unit3	Pa or mbar	unit of quantity3: Pascal or millibar

Time definition and data block description

For a description of the time definition see section C.12.1.

Cyclone track information:

For each time in the time series of space varying wind and pressure on a Spiderweb grid, the position of the cyclone eye (and thus also the spiderweb grid) must be given, as well as the drop of atmospheric pressure in the cyclone eye.

**Figure C.3:** Spiderweb grid definition**File version and conversion**

The current description holds for **FileVersion 1.03**. The table below shows the latest modifications in the file format (and version number).

FileVersion	Modifications
1.03	No changes for this meteo input type
1.02	Changed the use of keyword <code>n_rows</code> and <code>n_cols</code> . The radius of the cyclone is divided in <code>n_rows</code> rings of width: spw_radius/n_rows [m] and the circle is divided in <code>n_cols</code> parts of $2\pi/n_cols$ [rad].
1.01	Changed keyword <code>MeteoType</code> to <code>FileType</code> Changed fixed value of input type (Keyword <code>Filetype</code>) from <code>Spiderweb</code> to <code>meteo_on_spiderweb_grid</code>

**Restriction:**

- ◊ The restrictions for space varying wind and pressure on a Spiderweb grid are the same as for space varying wind and pressure on an equidistant grid, described in [section C.12.1](#).

**Remarks:**

- ◊ The remarks for space varying wind and pressure on a separate curvilinear grid are the same as for space varying wind and pressure on an equidistant grid, described in [section C.12.1](#).
- ◊ The Spiderweb grid is circular and the definitions of the number of rows `n_rows` and the number of columns `n_cols` is therefore different than for the other meteo input formats. For the Spiderweb grid, the number of rows determines the grid size in radial direction. The number of columns defines the grid size in angular direction. See [Figure C.3](#).
- ◊ The wind is specified according to the nautical convention, i.e. wind from the true North has direction zero and the wind turns clockwise with an increasing angle. See [Figure C.2](#).

Example:

A file for input of space varying wind and pressure on a 5x3 Spiderweb grid, has the following layout:

```

FileVersion          = 1.03
filetype            = meteo_on_spiderweb_grid
NODATA_value       = -999.000
n_cols              = 3
n_rows              = 5
grid_unit           = degree
spw_radius          = 600000.0
spw_rad_unit        = m
air_pressure_reference = air_pressure_default_from_computational_engine
n_quantity          = 3
quantity1           = wind_speed
quantity2           = wind_from_direction
quantity3           = p_drop
unit1               = m s-1
unit2               = degree
unit3               = Pa
TIME                = 0.0 hours since 1997-07-14 03:00:00 -06:00
x_spw_eye           = 115.1
y_spw_eye           = 18.9
p_drop_spw_eye      = 5300.0
1.38999             1.38261    1.38315
1.28251             1.34931    1.22571
1.27215             1.31214    1.32451
1.38999             1.86592    2.87732
1.43899             1.24912    2.21519
60.0000              180.0000   270.0000
28.7500              20.0000   31.2500
42.5000              53.7500   65.0000
49.3400              60.2400   81.5200
51.4100              62.0000   43.1200
5301.280             5294.490   5156.240
5043.460             5112.040   5264.020
5140.020             5202.520   5411.210
5294.730             5285.760   5235.250
5242.530             5156.190   5124.240
TIME                = 1.0 hours since 1997-07-14 03:00:00 -06:00
x_spw_eye           = 114.8
y_spw_eye           = 18.8
p_drop_spw_eye      = 5250.0
1.35763             1.35763    1.35763
1.35763             1.87273    2.24784
1.92214             2.47836    2.17266
1.87662             2.72116    2.82375
1.26585             2.24146    2.38722
159.0000             346.5200   290.6400
342.3200             282.1400   20.2400
10.7500              25.5300   36.4500
61.8400              81.6200   45.5100
49.5250              56.7500   75.1300
5314.520             5104.490   5287.240
5124.240             5285.760   5252.420
5152.460             5247.040   5222.020
5242.020             5223.520   5475.210
5244.270             5211.210   4998.110

```

This results in the following set of meteo data. Velocities given in [m/s] and pressure drops in [Pa] on a Spiderweb grid which is given in spherical coordinates (grid_unit = degree). The cyclone and spiderweb grid have a radius of 600 km. The grid is 5x3, which means the radius is divided in five parts of 120 km and the 360 degrees are divided in 3 parts of 120 degrees each. Wind speeds, wind directions and pressure drops are given at two times: 0 and 1.0 hour since July 14th, 1997, 03:00 AM, in UTC-6. Between these two times the cyclone eye moves from (longitude, latitude) (115.1, 18.9) to

(114.8, 18.8) on the globe and the pressure drop in the cyclone eye decreases from 5300.0 [Pa] to 5250.0 [Pa].

C.12.4 Meteo on a NetCDF file

This item will be added in near future



Remarks:

- ◊
- ◊

C.13 Fourier analysis

File contents

All parameters required to execute an online Fourier analysis for specified quantities, a specified period and for specified frequencies (Data Group *Output - Storage*).

Filetype

ASCII

File format

Fix formatted for text variables, free formatted for real and integer values.

Filename

<name.fou>

Generated

Manually offline

Record description:

Record	Record description																		
each record	<p>Variable on which the Fourier analysis is to be performed (2 characters):</p> <table> <tr><td>wl</td><td>water levels</td></tr> <tr><td>ws</td><td>wind magnitude</td></tr> <tr><td>ux</td><td>velocity in x-direction</td></tr> <tr><td>uy</td><td>velocity in y-direction</td></tr> <tr><td>uc</td><td>velocity magnitude</td></tr> <tr><td>ct</td><td>temperature</td></tr> <tr><td>cs</td><td>salinity</td></tr> <tr><td>cn</td><td>n-th constituent in the MDF-file</td></tr> <tr><td>bs</td><td>bed stress</td></tr> </table> <p>Analysis start time in Tunit [seconds, minutes, hours] after the Reference Date, Analysis stop time in Tunit [seconds, minutes, hours] after the Reference Date, Number of cycles within the analysis time frame, Nodal amplification factor, Astronomical argument, Layer number for the Fourier analysis, Flag for the computation of elliptic parameters: y/n (default no) if number of cycles > 0, min/max if number of cycles = 0.</p>	wl	water levels	ws	wind magnitude	ux	velocity in x-direction	uy	velocity in y-direction	uc	velocity magnitude	ct	temperature	cs	salinity	cn	n-th constituent in the MDF-file	bs	bed stress
wl	water levels																		
ws	wind magnitude																		
ux	velocity in x-direction																		
uy	velocity in y-direction																		
uc	velocity magnitude																		
ct	temperature																		
cs	salinity																		
cn	n-th constituent in the MDF-file																		
bs	bed stress																		



Remarks:

- ◊ If the number of cycles is set equal to 0, the mean level of the variable over the interval specified by the start and stop time is determined.
- ◊ If, in addition, the flag for the computation of elliptic parameters is set to "max" or "min" the maximal or minimal value is determined of the selected variable as it occurred during the simulation.
- ◊ Computed Fourier amplitudes slightly differ from the amplitude of the corresponding tidal component. When comparisons with co-tidal maps have to be made, this factor can be determined using the subsystem ASCON of Delft3D-TIDE, the tidal analysis package of Deltares.
- ◊ Computed Fourier phases are related to the reference date of the FLOW computation. For comparisons with co-tidal maps a phase shift equal to the astronomical argument has to be applied.
- ◊ The layer number is not relevant for water levels. In 3D models, the Fourier analysis is done on the whole 3D field, ignoring the layer number.
- ◊ For computational cells which are dry during all Fourier cycles, no values are associated, and

hence they contain missing value.

Restrictions:



- ◊ Times specified must be a multiple of the computational time step.
- ◊ Times specified must be a valid time, i.e. must fit in the simulation time frame of the FLOW computation.
- ◊ Items in a record must be separated by one or more blanks.
- ◊ The variable's name must start in position one.

Example:

A Fourier analysis is requested for:

- ◊ Water level: mean value and the first two harmonics.
- ◊ Velocity in x-direction: first harmonic, in the top layer.
- ◊ Velocity in y-direction: first harmonic, in the top layer.
- ◊ Velocity magnitude: first harmonic, in the top layer.
- ◊ Temperature: first harmonic, in the third layer.
- ◊ Salinity: mean value of the third layer.
- ◊ Four constituents: mean value for a slightly shifted time period in the top layer for 3 constituents; and the maximum for the fourth constituent.

wl	720.	1440.	2	1.000	0.000	
wl	720.	1440.	1	1.000	0.000	
wl	720	1440.	0	1.000	0.000	
ux	720.	1440.	1	1.000	0.000	1
uy	720.	1440.	1	1.000	0.000	1
uc	720.	1440.	1	1.000	0.000	1
ct	720.	1440.	1	1.000	0.000	3
cs	720.	1440.	0	1.000	0.000	3
c1	710.	1430.	0	1.000	0.000	1
c2	710.	1430.	0	1.000	0.000	1
c3	710.	1430.	0	1.000	0.000	1
c4	710.	1430.	0	1.000	0.000	1 max

Remarks:



- ◊ In this example the start and stop times are in minutes because the Tunit of this model is in minutes. Noted that it is also possible to specify these times in seconds or hours.
- ◊ A layer value of 0 in a 3D model is not allowed; i.e. this will NOT result in a depth-averaged value.

D Initial conditions and spatially varying input

D.1 Introduction

Spatially varying input is input data that varies in space, not in time, such as initial conditions (water levels, etc.), or spatially varying model parameters (friction coefficients, etc.). D-Flow FM uses input data in *model-independent coordinates*, that is, the input files should contain their own spatial coordinates, and do not correspond with the D-Flow FM grid cell numbering. As a result, the model grid can always be changed, without the need to change all other spatial input. The spatial input will be interpolated onto the active model grid. Spatial input has to be specified in the ext-file ([section C.5](#)). All supported quantities are listed in [Table C.2](#) under “Initial fields” and “Spatial physical properties”.

Initial conditions can also be set using a restart-file, which assigns the exact values in the file to the current model grid. No interpolation is performed, and the restart file should correspond exactly with the current model.

D.2 Supported quantities

D.2.1 Accepted quantity names in initial field files

D.2.2 Water levels

Water levels can come from both all file types listed below. The restart file sets the water level both at the old and new time step. Non-restart files should be specified using QUANTITY=initialwaterlevel.

D.2.3 Initial velocities

Initial velocities should come from restart files, such that they exactly correspond with the model's waterdepths. They can also come from non-restart files using QUANTITY=initialvelocityx/y, but this is not advised.

D.2.4 Salinity

Salinity concentrations from non-restart files are possible as spatially varying, depth-averaged values, or (for 3D models) as linearly interpolated values between a top layer concentration and a bottom layer concentration. The latter is achieved by specifying two QUANTITY blocks: initialsalinity and initialsalinitytop. For 3D models, alternatively, spatially constant, but depth-varying salinities can be set using the QUANTITY=initialverticalsalinityprofile, and a vertical profile file (see [section D.3.3](#)).

D.2.5 Temperature

Temperature values from non-restart files are possible as depth-averaged values, using QUANTITY=initialtemperature.

For 3D models, alternatively, spatially constant, but depth-varying temperatures can be set using the QUANTITY=initialverticaltemperatureprofile, and a vertical profile file (see [section D.3.3](#)).

D.2.6 Tracers

Initial tracer concentrations from non-restart files are similar to temperature values, now specified using QUANTITY=initialtracer<tracername>. The set of tracer name(s) is not listed in the MDU-file explicitly, they are inferred from all supplied initial tracer definitions *and* the tracer boundary conditions (see also [section E.1.6](#)). The <tracername> postfix can uniquely associate an initialtracer quantity with a tracerbnd quantity.

D.2.7 Sediment

Initial sediment concentrations for the Delft3D-SED-based sediment transport module (see section 9.2 for an explanation) is via the <*.sed> file. See the Delft3D-SED User Manual for further details.

D.2.8 Physical coefficients

A second group of spatially varying input are the physical and numerical coefficients listed below. They allow changing model parameters in space or in certain sub-regions of the domain, and as such these values override the uniform values from the MDU file. The input data can come from the in-polygon file type (section D.3.1 or the sample file type (section D.3.2).

quantity name	MDU uniform equivalent	description
frictioncoefficient	UnifFrictCoef	Friction coefficient (use IFRCTYP and UnifFrictType, resp. to set the friction type).
horizontaleddy... ...viscositycoefficient	Vicouv	Horizontal eddy viscosity coefficient (m^2/s).
horizontaleddy... ...diffusivitycoefficient	Dicouv	Horizontal eddy diffusivity coefficient (m^2/s).
advectiontype	AdvecType	Type of advection scheme.
ibotlevtype	BedlevType	Type of bed-level handling.

D.3 Supported file formats

All file formats below, except for restart files, are in *model-independent coordinates* and will be cropped and/or interpolated onto the model grid.

D.3.1 Inside-polygon option

A spatially constant value in a restricted part of the domain can be set by specifying a polygon file (section C.2) in the ext-file as follows:

```
QUANTITY=frictioncoefficient
FILENAME=winterbed.pol
FILETYPE=10
METHOD =4
VALUE   =55
IFRCTYP =1  # Optional, override uniform [physics] UnifFrictType=..
```

The interpolation option METHOD=4 simply specifies the no-interpolation is to be performed, only inside-polygon cropping. For QUANTITY=frictioncoefficient the default friction type may be overridden with the keyword IFRCTYP.

D.3.2 Sample file

Spatial data from sample files (<*.xyz>, see [section C.3](#)) is interpolated by triangle interpolation or sample averaging. The following options control the type of interpolation:

interpolation method	options	description
METHOD=5		Triangle interpolation
	EXTRAPOLTOL=20	Allow extrapolation from convex hull of samples up to ... meters (default: 0).
METHOD=6		Sample averaging inside control volumes:
	AVERAGINGTYPE=1	Normal (unweighted) averaging
	AVERAGINGTYPE=2	Nearest neighbour
	AVERAGINGTYPE=3	Maximal value
	AVERAGINGTYPE=4	Minimal value
	AVERAGINGTYPE=5	Inverse weighted distance averaging
	AVERAGINGTYPE=6	Minimal absolute value
	RELATIVESEARCHCELLSIZE=...	Control size of search volume: 1=actual cell, 1.5=50 % larger in all directions.

D.3.3 Vertical profile file

Initial values that are constant in 2D space, but vary in depth can be specified using vertical profile files, which are basic polyline files ([section C.2](#)). For example for a linearly varying salinity from 30 to 20 ppt from -10 to 0 m :

```
L1
2 2
-10 30
0 20
```

The METHOD keyword must be specified, but is further ignored: linear interpolation is always used.

D.3.4 Map file

Initial field data from a map file may be used as non-restart input. Values are read in from the map files as if they were samples, and further treated and interpolated as such. All interpolation options from [section D.3.2](#) can be used here. This option may be used as a non-exact restart state when the model grid has been changed (e.g., locally refined or cropped to a subdomain).

D.3.5 Restart file

Restart files are discussed in [section F.3.4](#). They should be included in your model in the mdu-file as follows:

```
[restart]
RestartFile      = mdu_name_yyyyymmdd_HHMMSS_rst.nc
RestartDateTime  = # Restart time (YYYYMMDDHHMMSS), only relevant
                  # in case of restart from *_map.nc
```

Concerning parallel runs, each subdomain can generate its own restart/map file. To restart a parallel run, one can use any of the following approaches:

- ◊ On each subdomain use its own restart/map file, provided that the partition does not change. (This means that in each partition MDU file specify its own restart/map file.)
- ◊ Merge the subdomain restart/map files to one global file (see [6.2.5.2](#)). This file enables to restart a parallel run with a different or the same partition. (This means that specify the merged file for all the partition MDU files.) One can also run a sequential simulation with this merged restart/map file.

E Boundary conditions specification

Boundary conditions are part of the external forcing of a model and as such declared in the external-forcings file ([section C.5](#)).

E.1 Supported boundary types

E.1.1 Astronomic boundary conditions

Boundary values can be specified for any location in terms of astronomical components in attribute files with extension <cmp> (the BC-format, discussed furtheron, is supported as an alternative). Tidal motion are described as a series of simple harmonic constituent motions, each with its own characteristic period:

$$H(t) = A_0 + \sum_{i=1}^k A_i F(c_i) \cos \left(\frac{2\pi}{T(c_i)} t + (V_0 + u)(c_i) - G_i \right) \quad (\text{E.1})$$

in which:

c_i	i -th component, specified by label (name)
A_i	amplitude of the i -th component
G_i	phase of the i -th component
$T(c_i)$	period
$H(t)$	boundary value at time t
A_0	constant offset value
k	number of relevant components
$F(c_i)$	nodal amplitude factor
$(V_0 + u)(c_i)$	astronomical argument

The component (by label) c_i , amplitude A_i and phase G_i for each component i are required in the <cmp>-file. In addition to the primary constituents, compound and higher harmonic constituents may have to be used. This is the case in shallow water areas for example, where advection, large amplitude to depth ratio, and bottom friction give rise to non-linear interactions.

F , V_0 and u are also *time-dependent*. For a given period, their values are easily calculated or obtained from various tidal year books. V_0 is a frequency dependent phase correction factor relating the local time frame of the observations to an internationally agreed celestial time frame. F_i and u_i are slowly varying amplitude and phase corrections. The variations depend on the frequency, often with a cyclic period of 18.6 years. By default, the nodal amplitude factors and astronomical arguments are re-calculated every 6 hours.

E.1.2 Astronomic correction factors

For individual astronomic components, correcting factor α_i for amplitude and offset δ_i may be specified by the user so that the effective amplitude for that component becomes $\alpha_i A_i$ and the effective phase becomes $G_i + \delta_i$.

E.1.3 Harmonic flow boundary conditions

Harmonic boundary conditions are specified in a manner similar to their astronomical counterpart, except that the periods (first column of a .cmp file) are declared explicitly, rather than using component names. Equation ((E.1)) applies with $T(c_i) = T_i$ and furthermore $F = 1$ and $(V_0 + u) = 0$.

E.1.4 QH-relation flow boundary conditions

In this type of boundary condition, a waterlevel is prescribed as a function of the integrated discharge through a cross-section defined by the polyline of the boundary.

The relation between waterlevel and discharge is specified in a lookup-table.

Within this table linear interpolation is applied.

Note that only one waterlevel is set for the entire QH-boundary, and that its behaviour is specified by a single QH-table for the entire boundary.

E.1.5 Time-series flow boundary conditions

Time-series in the general time-series file format ([section C.4](#)) can be used to specify values on boundaries.

Boundary values are provided at discrete points in time, which are then used for interpolation at times in between.

Extrapolation beyond the range of specified times is not supported, Time-series are also supported at multiple vertical levels for quantities that are defined on layers (tracers, salinity, temperature, velocity). These need to be specified in the BC-format ([section E.2.3](#)).

E.1.6 Time-series transport boundary conditions

The boundary conditions of tracers, salinity and temperature are specified similar to any other quantity already mentioned. The quantity name should be in the form `tracerbnd<name>`, where name is the user-defined tracer name.

E.1.7 Time-series for the heat model parameters

In order to conduct a heat flux simulation (if in the MDU-file "Temperature" under [physics] was set to 5), then four time-varying meteorological fields become relevant:

- ◊ humidity
- ◊ airtemperature
- ◊ cloudiness
- ◊ solar radiation

These are read as a three- or four-column vector (the first three variables of this list, either with or without solar radiation). The corresponding quantity names in the <ext>-file are:

- ◊ `humidity_airtemperature_cloudiness`
- ◊ `humidity_airtemperature_cloudiness_solarradiation`

As for filetype, currently only the curvilinear format ([C.12.2](#)) and the multicolumn time series ([C.4](#)) are supported.

E.2 Boundary signal file formats

E.2.1 The <cmp> format

```
* comments
*
c[0]      amplitude[0]    phase[0]
c[1]      amplitude[1]    phase[1]
...
```

where `c` represents a period in minutes or the name of a valid astronomic component. Amplitudes are assumed to be in the unit of the quantity.

E.2.2 The <qh> format

```
* comments
* ...
discharge[0]      waterlevel[0]
discharge[1]      waterlevel[1]
...
```

E.2.3 The <bc> format

The external forcings file may point to this type of file via the `forcingFile` keyword (see section C.5.2). The type of forcing is not restricted to boundary conditions, but may also contain meteo time series (global or stations). The <bc> format allows you to combine multiple forcing signals in a single (ASCII) file. Multiple **Boundary** or **Meteo** blocks in the external forcings file may then point to the same file. The file consists of a **General** block followed by one or more **Forcing** blocks. Each forcing block consists of a column-wise table (the data) and a header specifying how this table should be interpreted.

Table E.1: Description of <bc> format.

Keyword	Type	Default	Description
[General]			
fileVersion	String	1.01	File version. Do not edit this.
fileType	String	boundConds	File type. Do not edit this.
[Forcing]			<i>Repeat as needed</i>
name	String		Location name. Possible values are: <ul style="list-style-type: none"> ◊ global: Spatially uniform values, e.g., for meteo forcing. ◊ <polyline point>: Boundary polyline (point) reference. ◊ <stationId>: Station id, e.g., for meteo stations.
function	String		More details for each of these options can be found in Section E.2.3.1. Function type. The supported options are timeSeries, harmonic, astronomic, harmonic-Correction, Astronomic-Correction, t3D, QHTable, and constant. A description of these options follows after this table.
offset	Double	0	If function equals timeSeries, t3D or constant all values in the table are increased by the offset (after multiplication by factor).
factor	Double	1	If function equals timeSeries, t3D or constant all values in the table are multiplied with the factor. If function equals harmonic or astronomic only the amplitude column is multiplied by the given factor.
Vertical Position Specification	Double[]		The vertical position is specified as a space- or comma-separated list of floats, the interpretation of which varies according to the vertical position type. The order of the positions in the list becomes relevant when referring to them from the quantity blocks. However, no specific ordering of these positions (ascending or descending) is assumed.

(continued on next page)

Keyword	Type	Default	Description
<i>(continued from previous page)</i>			
Vertical Interpolation	String		Possible values are: <ul style="list-style-type: none"> ◊ linear: linear interpolation between vertical positions. ◊ log: logarithmic interpolation between vertical positions (e.g. vertical velocity profiles). ◊ block: equal to the value at the directly lower specified vertical position.
Vertical Position Type	String		Possible values are: <ul style="list-style-type: none"> ◊ percBed: Percentage wrt. water depth from the bed upward (sigma layers). ◊ ZBed: Absolute distance from the bed upward (z-layers).
Time Interpolation	String		Possible values are: <ul style="list-style-type: none"> ◊ linear: linear interpolation between times. ◊ Block-From: equal to that at the start of the time interval (latest specified time value). ◊ Block-To: equal to that at the end of the time interval (upcoming specified time value).

The next three keywords repeated for every column in this data block

quantity	String	Name of quantity
unit	String	Unit of quantity
Vertical Position	Integer	This is a (one-based) index into the vertical-position-specification, assigning a vertical position to the quantity (<code>t3D-blocks</code> only)

The data block

...

This block holds the actual data in columns. The interpretation of the columns is determined by the order of the aforementioned quantity keywords, i.e. the n -th quantity-block refers to the n -th column. The columns should contain valid floating point numbers (scientific notation is allowed). The only exception is the `Astronomic Component` column, which should contain strings representing the component names.

E.2.3.1 Name in a bc block

The `name` keyword in the above Table E.1 depends on the type of forcing.

E.2.3.1.1 Boundary forcings

Boundary conditions can be prescribed by one or more **Forcing** blocks, each of which is uniquely tied to an individual boundary support point. For a boundary location defined as a polyline, the support points are the ones listed in the `<.pli>`-file. In that case the name in the `bc`-file should read `pliname_<point number>`. The point number should be formatted as a 4-digit zero-padded number. Blocks with the function `QHTable` are an exception to this rule, because in `qh`-boundaries no data is required for individual support points. For this specific type of boundary condition, the name in the `<.bc>`-file should read `pliname`.

E.2.3.2 Function in a bc block

The function keyword specifies how the **Forcing** block should be interpreted. The options are:

timeSeries	Value(s) as a function of time. One of the quantities must be called Time (see remarks below for unit).
harmonic	Period, amplitude, phase. One of the quantities must be called Harmonic Component (-). The other quantities should come in pairs: <ul style="list-style-type: none">◊ <i>quantity</i> amplitude (the unit depends on the quantity)◊ <i>quantity</i> phase (degrees)
astronomic	Component-name, amplitude, phase. One of the quantities must be called Astronomic Component (-). The other quantities should come in pairs: <ul style="list-style-type: none">◊ <i>quantity</i> amplitude (the unit depends on the quantity)◊ <i>quantity</i> phase (degrees)
Harmonic-Correction	Period, amplitude-factor, phase-offset. One of the quantities must be called Harmonic Component (-). The other quantities should come in pairs: <ul style="list-style-type: none">◊ <i>quantity</i> amplitude (the unit depends on the quantity)◊ <i>quantity</i> phase (degrees)
Astronomic-Correction	Component-name, amplitude-factor, phase-offset. One of the quantities must be called Astronomic Component (-). The other quantities should come in pairs: <ul style="list-style-type: none">◊ <i>quantity</i> amplitude (the unit depends on the quantity)◊ <i>quantity</i> phase (degrees)
t3D	Values on multiple vertical positions versus time. One of the quantities must be called Time (see remarks below for unit).
QHTable	Discharge as a function of water level. This data block should contain two quantities: <ul style="list-style-type: none">◊ <i>location</i> WaterLevel (m AD)◊ <i>location</i> Discharge (m³/s)
constant	Constant value

Remarks:

- ◊ Although typical files will use UpperCamelCase keywords for chapters and lowerCamelCase for keywords and constants, all strings (including user-defined location and quantity names) will be verified by case insensitive comparison.
- ◊ Valid time units are string indicating the time unit (days, hours, minutes, or seconds) since a reference date (YYYY-MM-DD) and optional time (HH-MM-SS) and time zone (+/-HH).
- ◊ All quantities in a harmonic or astronomic block are forced using the same components. All quantities in a time series block are forced using the same time points.



Example:

```
[General]
fileVersion      = 1.01
fileType         = boundConds

[Forcing]
name             = left01_0001
function          = Harmonic
quantity          = Harmonic Component
```

```

unit          = -
quantity      = waterlevelbnd amplitude
unit          = m
quantity      = waterlevelbnd phase
unit          = rad/deg/minutes
0.0 4.114 0.0

[Forcing]
name          = left01_0001
function      = Harmonic
quantity      = Harmonic Component
unit          = -
quantity      = normalvelocitybnd amplitude
unit          = m/s
quantity      = normalvelocitybnd phase
unit          = rad/deg/minutes
0.0 1.215 0.0

[Forcing]
name          = right01_0001
function      = TimeSeries
timeInterpolation = Linear
quantity      = time
unit          = minutes since 2001-01-01
quantity      = waterlevelbnd
unit          = m
0.000000    2.50
1440.000000  2.50

```

E.2.4 The NetCDF-format for point-location time-series

The external forcings file may point to this type of file (for boundary condition support points via the `forcingFile` keyword (see section C.5.2)).

NetCDF files that provide forcing data on point locations should meet the following structure:

- ◊ The location labels have to be stored in a character variable with attribute `cf_role = 'timeseries_id'`. This variable should have two dimensions, the primary being the series dimension and the secondary the string length dimension. The labels should always be unique.
- ◊ The variable containing the timeseries data is identified by its attribute `standard_name = <quantity-name>` and has two dimensions, the series dimension and the time dimension. The latter can be the unlimited dimension.

In analogy with the ASCII BC-format (Section E.2.3), location label and quantity name uniquely identify the timeseries.

Example header of a NetCDF file with point data:

```

netcdf timeseries {
dimensions:
    strlen = 7 ;
    node = 3 ;
    time = UNLIMITED ; // (18 currently)
variables:
    double time(time) ;
        time:units = "hours since 2000-01-01 00:00:00" ;
    char location(node, strlen) ;
        location:cf_role = "timeseries_id" ;
    double x(node) ;
        x:axis = "x" ;
        x:units = "km" ;
        x:long_name = "x coordinate of projection" ;

```

```
    x:standard_name = "projection_x_coordinate" ;
double y(node) ;
    y:axis = "y" ;
    y:units = "km" ;
    y:long_name = "y coordinate of projection" ;
    y:standard_name = "projection_y_coordinate" ;
double rainfall(time, node) ;
    rainfall:long_name = "Rainfall" ;
    rainfall:standard_name = "precipitation" ;
    rainfall:units = "mm" ;
    rainfall:_FillValue = "-999.9f" ;
data:
location =
"Station One",
"Station Two",
"Station Three" ;
}
```


F Output files

D-Flow FM can produce several types of output into several different files. This chapter summarizes the types of output and how to configure them in them model definition. We distinguish five types of output here:

- 1 The diagnostics file, a log file with live details of a single model run.
- 2 NetCDF output files for history, map and restart data.
- 3 Tecplot.
- 4 Timings file with performance statistics.
- 5 Shapefiles.

F.1 Diagnostics file

The diagnostics (dia) file is the log file of a single model simulation run. It is an important file to check for warning or error messages, when suspecting a faulty model run. The filename is automatically chosen as `<mdu_name.dia>`. For parallel model runs, each process writes its own file `<mdu_name_000X.dia>`. The dia file has the following global structure:

```
Various INFO/DEBUG messages:  
** INFO : Opened file : unstruc.hlp  
[.]  
The version of D-Flow FM used for this calculation:  
* Deltares, D-Flow FM Version 1.1.145.41271M, Aug 11 2015, 18:05:31  
Date and time of model run start:  
File creation date: 18:27:44, 06-09-2015  
[.]  
  
Check for possible model initialization errors:  
** WARNING: readMDUfile: [numerics] cflwavefrac=0.1 was in file, but not used.  
[.]  
  
Upon successful initialization, a full printout of the effective model settings is given:  
** INFO : ** Model initialization was successful **  
** INFO : * Active Model definition:  
# Generated on 18:27:45, 06-09-2015  
# Deltares, D-Flow FM Version 1.1.145.41271M, Aug 11 2015, 18:05:31  
[model]  
[.]  
  
Next, the time loop is started:  
** INFO : **  
** INFO : Writing initial output to file(s)...  
** DEBUG : Opened NetCDF file 'DFM_OUTPUT_035hammen/035hammen_his.nc' as #3.  
** DEBUG : Opened NetCDF file 'DFM_OUTPUT_035hammen/035hammen_map.nc' as #4.  
** INFO : Done writing initial output to file(s).  
[.]  
  
Finishing summary of model run:  
** INFO : simulation period (s) : 6400.0000000000  
** INFO : nr of timesteps ( ) : 771.0000000000  
** INFO : average timestep (s) : 8.3009079118  
[.]  
Some basic run time statistics  
** INFO : time steps (s) : 76.2840000003  
** INFO : Computation started at: 18:27:45, 06-09-2015  
** INFO : Computation finished at: 18:29:04, 06-09-2015  
** INFO :  
Which parallelization options have been active in this run:  
** INFO : MPI : no.  
** INFO : OpenMP : yes.
```

```
#threads max : 8
```

F.2 Demanding output

The configuration of what output should be produced during a model run is via the MDU file, and several attribute files for history files.

F.2.1 The MDU-file

The MDU-file has an aptly named [output]-section, with a range of key-value-pair options. See Table A.1 on page 294 for all available output options.

F.2.2 Observation points

The observation point file defines the locations for time series output of flow "point" variables such as water level and water depth. Observation points in 2D (or 3D) parts of the model domain are used to monitor quantities at grid cell centres. The keyword ObsFile should be used to point to the observation point file; optionally a space separated list of files can be specified (see Appendix A).

The following general remarks hold for observation point input:



Remarks:

- ◊ The observation points are stationary by default.
- ◊ Observation points may have non-unique names (also together with moving observation points), but the results can only be distinguished by the column number in the output file, and this practice is not advised.
- ◊ In 2D/3D, the x,y -locations are snapped to the grid cell in which they lie. When an observation point lies on a grid cell edge, snapping results are unpredictable.
- ◊ Time series are reported for the cell-centered solution data, that is: *no* interpolation to the exact x,y -location takes place.

F.2.2.1 The observation point file with extension <*.xyn>

The observation point file with extension <*.xyn> is basically like a sample file (section C.3), but now with station id strings in the third column:

```
-2.0625000e+00 5.71583333e+01 ABDN
-2.7375000e+00 5.15083334e+01 AVMH
-2.1125000e+00 4.9175000e+01 'St Helier Jersey'
```



Remarks:

- ◊ The station id/name may contain spaces, in which case it should be surrounded by single quotes, 'as follows'.
- ◊ Each observation point is first checked whether it lies inside a 2D grid cell. When this is not the case, in a second attempt it is snapped to the nearest 1D grid point.

F.2.3 Moving observation points

Moving observation points are specified via the <*.ext>-file:

```
QUANTITY=movingstationtxy
FILENAME=movingstation1.tim
FILETYPE=1
```

```
METHOD=1
OPERAND=0
```

The <*.tim> file (one for each moving observation point) is a standard time series file ([section C.4](#)) with three columns, containing the time in minutes since the reference data, the *x*- and *y*-position of the moving station at that time.

Remarks:



- ◊ Stationary and moving observation points form one large set of observation points in the output file.
- ◊ No space or time interpolation is done: the reported values are instantaneous values for the grid cell in which the moving observation point lies at that each output time.

F.2.4 Observation cross sections

Observation cross sections (or cross sections for short) define the location for time series output of flow "flux" variables such as discharge and velocity. Other typical output quantities are instantaneous as well as cumulative, space-integrated discharges and constituent transport. The keyword `CrsFile` should be used to point to the observation cross section file; optionally a space separated list of files can be specified (see [Appendix A](#)).

The location of an observation cross section can be specified using polylines using <*_crs.pli> files. The following general remarks hold for cross section input:

Remarks:



- ◊ Observation cross sections may have non-unique names, but the results can only be distinguished by the column number in the output file, and this practice is not advised.
- ◊ A cross section polyline is snapped to all grid cell edges (i.e., velocity points), whose flow link is intersected by the polyline.
- ◊ The flow data on velocity points is integrated along the polyline and across all vertical layers, resulting in a single time series per cross section and flow quantity.

F.2.4.1 Observation cross section file with extension <*_crs.pli>

The format of the <*_crs.pli> file is identical to that of a general polyline/polygon file ([section C.2](#)).

```
CrossSec1
4 2
    132345.0    549030.0
    132165.0    549285.0
    131940.0    549550.0
    131820.0    549670.0
CrossSec2
3 2
    131750.0    549865.0
    131595.0    550025.0
    131415.0    550175.0
```

Remarks:



- ◊ The first header line of each polyline block contains the name of each cross section.
- ◊ The cross section id/name may contain spaces, no quotes or other delimiters must be used.

F.3 NetCDF output files

All flow data output produced by D-Flow FM is written in the NetCDF format. This cross-platform storage format is widely used in the world. The contents of a file can be documented using variable

attributes and standardized meta data. For this purpose the D-Flow FM output files aim to satisfy the CF-conventions.

F.3.1 Timeseries as NetCDF his-file

The history file <*mdu_name_his.nc*> contains the dimensions and variable definitions in its header, followed by the actual data. For parallel runs, all data is gathered by domain #0 into <*mdu_name_0000_his.nc*> (section 6.2.5).

F.3.1.1 Dimensions

The following dimensions are defined in a his file header:

```
netcdf weirfree_his {
dimensions:
    time = UNLIMITED ; // (64 currently)
    name_len = 64 ;
    stations = 18 ;
    cross_section = 4 ;
    cross_section_name_len = 64 ;
    cross_section_pts = 3 ;
    npumps = 2 ;
    // and possibly nweirgens, nweirs, ngategens, ngates, nsources/sinks
```

F.3.1.2 Location variables

The original location and IDs for stations, cross sections and sources/sinks are also included in the his file:

```
variables:
double station_x_coordinate(stations) ;
station_x_coordinate:units = "m" ;
station_x_coordinate:standard_name = "projection_x_coordinate" ;
station_x_coordinate:long_name = "x-coordinate" ;
double station_y_coordinate(stations) ;
station_y_coordinate:units = "m" ;
station_y_coordinate:standard_name = "projection_y_coordinate" ;
station_y_coordinate:long_name = "y-coordinate" ;
char station_name(stations, name_len) ;
station_name:cf_role = "timeseries_id" ;
station_name:long_name = "Observation station name" ;

double cross_section_x_coordinate(cross_section, cross_section_pts) ;
cross_section_x_coordinate:units = "m" ;
cross_section_x_coordinate:standard_name = "projection_x_coordinate" ;
cross_section_x_coordinate:long_name = "x-coordinate" ;
double cross_section_y_coordinate(cross_section, cross_section_pts) ;
cross_section_y_coordinate:units = "m" ;
cross_section_y_coordinate:standard_name = "projection_y_coordinate" ;
cross_section_y_coordinate:long_name = "y-coordinate" ;
char cross_section_name(cross_section, cross_section_name_len) ;

double source_sink_x_coordinate(source_sink, source_sink_pts) ;
source_sink_x_coordinate:units = "m" ;
source_sink_x_coordinate:standard_name = "projection_x_coordinate" ;
source_sink_x_coordinate:long_name = "x-coordinate" ;
double source_sink_y_coordinate(source_sink, source_sink_pts) ;
source_sink_y_coordinate:units = "m" ;
```

```

source_sink_y_coordinate:standard_name = "projection_y_coordinate" ;
source_sink_y_coordinate:long_name = "y-coordinate" ;
char source_sink_name(source_sink, source_sink_name_len) ;

```

F.3.1.3 Variables on stations

Variables on observations station are typically defiend as follows:

```

double waterlevel(time, stations) ;
waterlevel:standard_name = "sea_surface_level" ;
waterlevel:long_name = "Water level" ;
waterlevel:units = "m" ;
waterlevel:coordinates = "station_x_coordinate station_y_coordinate station_name" ;
waterlevel:_FillValue = -999.
;
```

Other quantities that can be written are:

- ◊ Water depth
- ◊ Flow velocity (*x* and *y*)
- ◊ Salinity, temperature and other transported quantities, such as tracers

F.3.1.4 Variables on cross sections

Variables on cross sections are typically defined as follows:

```

double cross_section_discharge(time, cross_section) ;
cross_section_discharge:units = "m^3/s" ;
cross_section_discharge:coordinates = "cross_section_name" ;

```

Other quantities that can be written are:

- ◊ Cross section flow area at current time;
- ◊ Cross section average velocity at current time;

F.3.1.5 Mass balance output

The history file may also contain model-global, time-integrated mass balance output, for example:

```

double WaterBalance_total_volume(time) ;
WaterBalance_total_volume:units = "m3" ;

```

Other quantities that can be written are:

- ◊ Net storage since Tstart;
- ◊ Volume error since Tstart;
- ◊ Cumulative inflow through boundaries since Tstart;
- ◊ Cumulative outflow through boundaries since Tstart;
- ◊ Net inflow through boundaries since Tstart;
- ◊ Cumulative inflow via SOBEK-D-Flow FM 1D–2D boundaries since Tstart;

- ◊ Cumulative outflow via SOBEK-D-Flow FM 1D–2D boundaries since Tstart;
- ◊ Net inflow via SOBEK-D-Flow FM 1D–2D boundaries since Tstart;
- ◊ Total precipitation volume since Tstart;
- ◊ Net inflow via source–sink elements since Tstart;

F.3.1.6 Variables on sources/sinks

Variables on sources/sinks are typically defined as follows:

```
double source_sink_prescribed_discharge(time, source_sink) ;  
source_sink_prescribed_discharge:units = "m^3/s" ;  
source_sink_prescribed_discharge:coordinates = "source_sink_name" ;
```

Other quantities that can be written are:

- ◊ source sink prescribed salinity increment;
- ◊ source sink prescribed temperature increment;
- ◊ source sink current discharge;
- ◊ source sink cumulative volume;
- ◊ source sink discharge average.

These variables are automatically written to his-file if any source/sink exists. One can switch it off by setting in MDU-file that [output] Wrihis_sourcesink = 0.

F.3.1.7 Hydraulic structure output

A history file can also contain automatic output for hydraulic structures, without the need to add an explicit cross section at the same location. An example variable is:

```
double weirgen_discharge(time, weirgens) ;  
weirgen_discharge:long_name = "Discharge through weir" ;
```

Other quantities that can be written are:

- ◊ weir crest level at current time;
- ◊ Total discharge through gate at current time;
- ◊ gate sill level at current time;
- ◊ gate sill width at current time;
- ◊ gate door lower edge level at current time;
- ◊ gate door opening width at current time;
- ◊ Total pump discharge at current time;
- ◊ Max. pump capacity at current time;

F.3.2 Spatial fields as NetCDF map-file

The map file <mdu_name_map.nc> contains data on the entire model grid in 1D, 2D and 3D. Three formats are currently supported, selected by the MapFormat keyword in the MDU:

```
[output]  
MapFormat = 1 # Map file format, 1: NetCDF, 2: Tecplot,  
# 3: NetCFD and Tecplot, 4: NetCDF-UGRID
```

D-Flow FM is using the more standardized UGRID¹ format. This is the default option. Delta Shell and Delft3D-QUICKPLOT also support this format. In this section only the conventional NetCDF format (1) is further discussed.

For parallel runs, each process writes a separate file for each partition as `<mdu_name_000X_map.nc>`.

The map file contains the dimensions and variable definitions in its header, followed by the actual data.

Dimensions

The following dimensions are defined in a map file header:

```
netcdf weirfree_map {
dimensions:
    nNetNode = 310 ;
    nNetLink = 486 ;
    nNetLinkPts = 2 ;
    nBndLink = 252 ;
    nNetElem = 180 ;
    nNetElemMaxNode = 4 ;
    nNetLinkContourPts = 4 ;
    nFlowElem = 180 ;           // Nr.
of grid cells
    nFlowElemMaxNode = 4 ;
    nFlowElemContourPts = 4 ;
    nFlowLink = 246 ;          // Nr.
of flow links (open cell edges)
    nFlowLinkPts = 2 ;
    time = UNLIMITED ; // (64 currently)
```

For plotting solution data, the grid cells (flow nodes) are important, and the flow links (open cell edges). All Net* dimensions relate the flow grid tot the original unstructured network. More explanation of these topological naming conventions can be found in [section 8.3.1](#).

Location variables

The location and shape of grid cells is stored in several x,y variables:

```
double FlowElem_xcc(nFlowElem) ;
    FlowElem_xcc:units = "m" ;
    FlowElem_xcc:standard_name = "projection_x_coordinate" ;
    FlowElem_xcc:long_name = "Flow element circumcenter x" ;
    FlowElem_xcc:bounds = "FlowElemContour_x" ;
double FlowElem_ycc(nFlowElem) ;
    FlowElem_ycc:units = "m" ;
    FlowElem_ycc:standard_name = "projection_y_coordinate" ;
    FlowElem_ycc:long_name = "Flow element circumcenter y" ;
    FlowElem_ycc:bounds = "FlowElemContour_y" ;
// [...]
double FlowElemContour_x(nFlowElem, nFlowElemContourPts) ;
    FlowElemContour_x:units = "m" ;
    FlowElemContour_x:standard_name = "projection_x_coordinate" ;
    FlowElemContour_x:long_name = "List of x-points forming flow element" ;
    FlowElemContour_x:_FillValue = -999 .
;
// [...]
```

¹<https://github.com/ugrid-conventions/ugrid-conventions>

```

double FlowElem_bl(nFlowElem) ;
FlowElem_bl:units = "m" ;
FlowElem_bl:positive = "up" ;
FlowElem_bl:standard_name = "sea_floor_depth" ;
FlowElem_bl:long_name = "Bottom level at flow element's circumcenter." ;

```

Variables on grid cells/pressure points

Variables on grid cells (represented by the pressure point/cell circumcenter) are typically defined as follows:

```

double s1(time, nFlowElem) ;
s1:coordinates = "FlowElem_xcc FlowElem_ycc" ;
s1:standard_name = "sea_surface_level" ;
s1:long_name = "waterlevel" ;
s1:units = "m" ;
s1:grid_mapping = "projected_coordinate_system" ;

```

Other quantities that can be written are:

- ◊ Water level at beginning of time step;
- ◊ Water depth;
- ◊ Numlimdt: number of times each cell was limiting the time step;
- ◊ Bed shear stress;
- ◊ Cell-centered velocity components;
- ◊ Effective Chézy roughness;
- ◊ Salinity;
- ◊ Temperature;
- ◊ Tracers and other constituents;
- ◊ Heat flux quantities, air temperature, relative humidity, cloudiness, and more detailed fluxes ([chapter 11](#));
- ◊ Streamline curvature and spiral flow intensity ([section 8.6](#));
- ◊ Suspended sediment concentrations and erodible layer thicknesses ([D-Morphology UM \(2019\)](#));
- ◊ Cell-centered wind speed components and atmospheric pressure;

Variables on flow links/velocity points

Variables on grid cell edges (represented by the velocity point) are typically defined as follows:

```

double q1(time, nFlowLink) ;
q1:coordinates = "FlowLink_xu FlowLink_yu" ;
q1:long_name = "Flow flux" ;
q1:units = "m3/s" ;

```

Other quantities that can be written are:

- ◊ Normal velocities at the old and new time level;
- ◊ Horizontal viscosity and diffusivity;
- ◊ Edge-centered wind speed components;
- ◊ Effective roughness values from trachytopes/vegetation ([section 14.2](#))

F.3.3 Class map files as NetCDF clm-file

The class map option gives output in a NetCDF file, where the size of the output file is significantly less than the size of a map-file. This is achieved by storing the field of interest (currently only water levels and water depths) not as a double (8 bytes), but as a byte that represents the class it is in. As the class will not change every time, compressing is used to get even smaller files than the factor 8 based on the ratio double/byte.

This format is very suitable for making animations of dryfall and flooding.

The class map file is defined in the mdu-file by:

```
[output]
...
ClassMapFile      = weirfree_clm.nc      # Filename for class map output
WaterlevelClasses = -3.0 2.0 5.0        # Water level classes (m)
WaterdepthClasses = 0.5 1.0 5.0 10.0     # Water depth classes (m)
ClassMapInterval  = 5.0                  # Class map output interval (s)
```

The class map file is always in UGRID format. When running in parallel, a class map file is generated for each process. Use mapmerge to collect them in a single file.

F.3.4 Restart files as NetCDF rst-file

Restart files <mdu_name_yyyymmdd_HHMMSS_rst.nc> are almost identical to map files, except that they contain all data for just a single time. The grid and flow geometry information is not present, except for the flow cell/link coordinates. Moreover, restart files resulted from a parallel run contains parallelization information. For this reason, restart files are less suitable for plotting, but efficient for restarting a computation.

F.4 Shapefiles

Shapefiles² are widely used for visualization in geographic information system (GIS) software. A shapefile stores geometric locations and corresponding attributes information for the spatial features. D-Flow FM can generate shapefiles by setting keyword in the MDU file. Table F.1 shows what features can be written to a shapefile by D-Flow FM, and the corresponding MDU settings. (By default these keywords are zero, which disables writing the shapefile.) Moreover, this function is also valid in parallel simulation.

Table F.1: Features and MDU settings for generating shapefiles

Features for shapefiles	MDU setting
Cross sections	[output] Wrishp_crs = 1
Weirs	[output] Wrishp_weir= 1
Gates	[output] Wrishp_gate= 1
Fixed weirs	[output] Wrishp_fxw = 1
Thin dams	[output] Wrishp_thd = 1
Embankments	[output] Wrishp_ebm = 1
Observation stations	[output] Wrishp_obs = 1
Source-sinks	[output] Wrishp_src = 1

²<https://www.esri.com/library/whitepapers/pdfs/shapefile.pdf>

F.5 Post processing on his-file

There two options for post processing on the his file. Both options are for finding the maximum value for each history station. They are options of the program dfmoutput.

F.5.1 Max25 function

The max25 function finds the maximum after applying a running mean over 25 points (in time).

Basic usage for max25 is:

```
dfmoutput max25 --infile [hisfile.nc]
```

There are three optional arguments:

```
--outfile [name of the outfile] , default max25.out  
--varname [variable to find maximum for] , default waterlevel  
--filterlength [list of numbers of points in time] , default 13,25
```

F.5.2 Maximum based on a generic filter

The second option is to find the maximum value after applying a 4th order monotone filter on the time serie for each station. This filter has three parameters: the filter length and two coefficients. These parameters can be given by the user, see below.

Basic usage for maximum based on a generic filter is:

```
dfmoutput genfilter --infile [hisfile.nc]
```

There are five optional arguments:

```
--outfile [name of the outfile] , default max.out  
--varname [variable to find maximum for] , default waterlevel  
--intval [filter period] , default 6  
--coefimpl [filter coefficient implicit part] , default 0.3  
--coefexpl [filter coefficient explicit part] , default 0.3
```

G Model generation

G.1 Introduction

Parts of the model building process can also be automated, to support model generation. This comes in addition to the full modelling process as described in [chapter 5](#).

G.2 Grid generation

Automated grid generation is limited to a few basic types, described hereafter. More specific grid generation should be done by the modeller in RGFGRID. Automated grid generation is currently done from the commandline using the `dflowfm-cli.exe` program (`dflowfm` on Linux).

G.2.1 Uniform Cartesian grid

A uniform Cartesian (i.e., rectangular) grid can be generated by specifying the bounding box and desired grid resolution:

```
> dflowfm-cli.exe --gridgen:x0=xLL:y0=yLL:dx=Δx:dy=Δy:ncols=M:nrows=N
```

This produces a file `<out_net.nc>`. When the coordinate system for the grid should be spherical (WGS84), add the option `:spherical=1`.

G.2.2 Locally refined Cartesian grid

An existing grid can be locally refined based on an additional input file that defines the refinement locations. Here, we will now consider a uniform Cartesian (i.e., rectangular) that needs to be refined based on a raster file (e.g. an ArcInfo file). The grid refinement is currently based on 'Courant' refinement, but this can be applied to different refinement metrics as well. The command for refinement is:

```
> dflowfm-cli.exe --refine:hmin=Δxmin:dtmax=Δtmax:connect=1:outsidecell=1  
unif_net.nc refineC.asc
```

The input file `<unif_net.nc>` is a file that has been produced by the `--gridgen` command (for this Cartesian case, see [section G.2.1](#)), for example with a grid resolution Δx_{max} . The input file `<refineC.asc>` should contain values 1 for locations that need the finest size Δx_{min} , 4 for one step coarser, 16 for two steps coarser. In general: 2^{2n} , with n the number of coarsening steps. The value of Δt_{max} is a fictitious value here, and should be set to $\Delta t_{\text{max}} = \Delta x_{\text{min}} / \sqrt{9.81}$. When entering only a few decimal digits for Δt_{max} , make sure to round it up.

This again produces a file `<out_net.nc>`.

H Spatial editor

H.1 Introduction

The spatial editor is a generic feature of the Delta Shell framework for editing spatial data (e.g. bed level, roughness, viscosity, initial conditions, sediment availability). The spatial editor supports both point clouds and coverages (e.g. data on a grid or network). Therefore, you can use the spatial editor both to edit spatial data in general and to prepare model input. This Chapter describes the general features of the spatial editor (section H.2), (spatial) quantity selection (section H.3), geometry operations (section H.4), spatial operations (section H.5) and the operation stack (section H.6). The examples given below are typically focusing on use of the spatial editor in the D-Flow FM plugin, but are in principal applicable to any Delta Shell plugin.

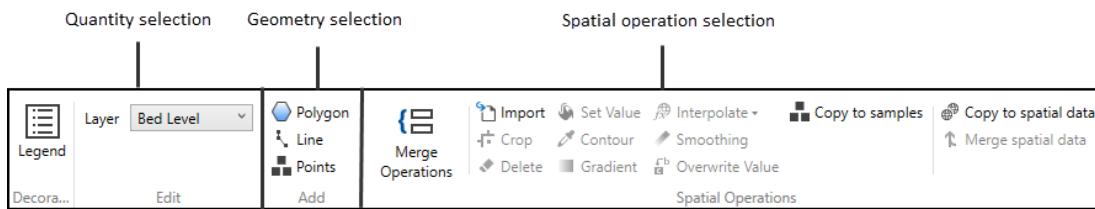


Figure H.1: Overview of spatial editor functionality in Map ribbon

H.2 General

H.2.1 Overview of spatial editor

The spatial editor functionality can be accessed from the “Spatial Operations” ribbon (Figure H.1). Typically, you first select which data set or quantity (either a point cloud or a coverage) to work on (e.g. bed level, roughness, viscosity, initial conditions, sediment availability), then a geometry (e.g. point, line, polygon) and finally which spatial operation to perform (e.g. crop, delete, set value, contour, interpolate, smoothing). Typical workflows are as follows:

Working on a point cloud dataset:

- 1 Import the dataset as point cloud (section H.2.2)
- 2 Activate/select the dataset (quantity) in the spatial editor (section H.3)
- 3 Select a geometry to perform an operation on (section H.4)
- 4 Select the spatial operation for this geometry (section H.5)
- 5 Repeat steps 3 and 4 until you are satisfied with the data
- 6 Export the dataset (section H.6.7)

Working on a coverage (e.g. model input):

- 1 Activate/select the spatial quantity to work on in the spatial editor (section H.3)
- 2 Optional: import a dataset as point cloud (section H.5.1)
- 3 Select a geometry to perform an operation on (section H.4)
- 4 Select the spatial operation for this geometry (section H.5)
- 5 Repeat steps 3 and 4 until you are satisfied with the data
- 6 Interpolate the point cloud to the grid or network (section H.5.10)
- 7 Optional: export the dataset (section H.6.7)

Upon performing a spatial operation, the ‘Operations’ panel will open (see Figure H.53) with the operations stack. This stack keeps track of the workflow of spatial operations that you performed. This helps you to make transparent how you arrived at your ‘final’ dataset without having to save all the intermediate datasets (steps) separately. Moreover, the stack is reproducible and easily editable without having to start all over again. When working on a coverage (e.g. the second workflow), point clouds can be used to construct the coverage. In this case the coverage (for example ‘Bed level’) is the ‘trunk’ of the workflow and the point clouds (appearing as sets in the stack) are ‘branches’ or subsets of this trunk

(see [Figure H.53](#)). By selecting the set or coverage in the ‘Operations’ panel you determine on which dataset you are working. The interpolate operation ([section H.5.10](#)) allows you to bring data from the point cloud (branch) to the coverage (trunk). For more information on the stack you are referred to section [section H.6](#).

H.2.2 Import/export dataset

To import a (point cloud) dataset use the context menu on “project” in the “Project Tree”, select “import” from the context menu and select the option “Points from XYZ-file” ([Figure H.2](#)). There is yet another method to import the point cloud. Click on “Import” icon under the Home ribbon to obtain a drop-down menu with the list of importers. Select the option “Points from XYZ-file” under Spatial data section ([Figure H.3](#)) to launch the import wizard.

After the import, the point cloud will be added to the project tree. To activate the point cloud in the spatial editor, either double click the dataset in the project tree ([Figure H.4](#)) or select it from the drop down box in the spatial editor ribbon ([Figure H.5](#)).



Note: Exporter still to be implemented

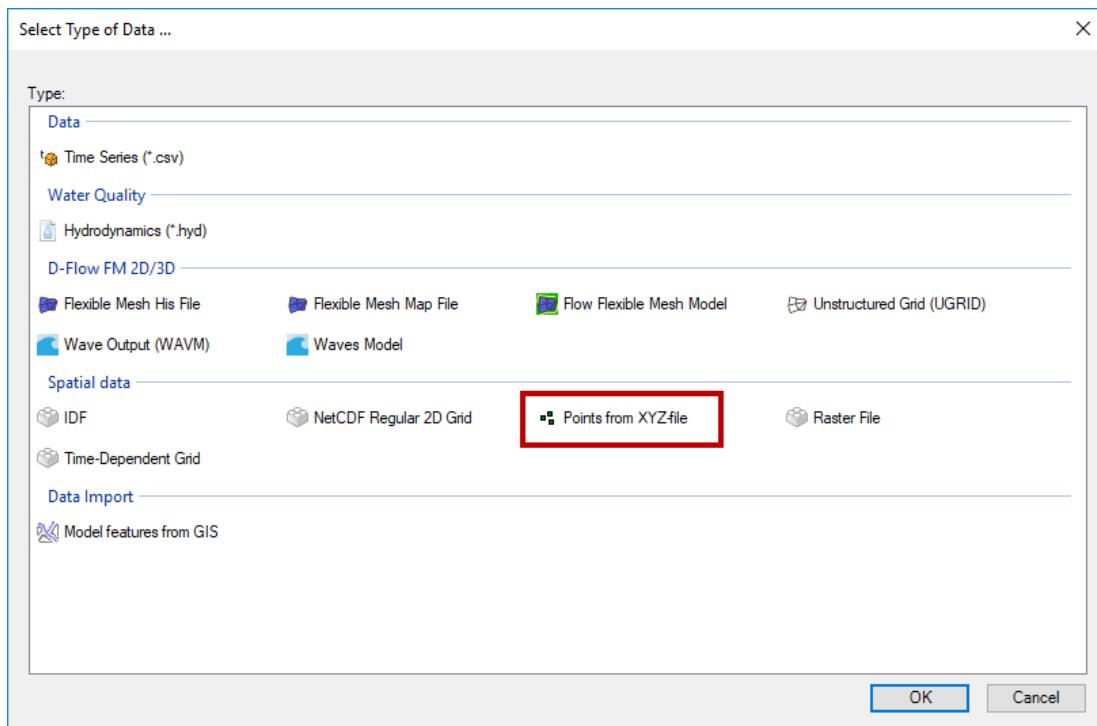


Figure H.2: Importing a point cloud into the project using the context menu on “project” in the project tree

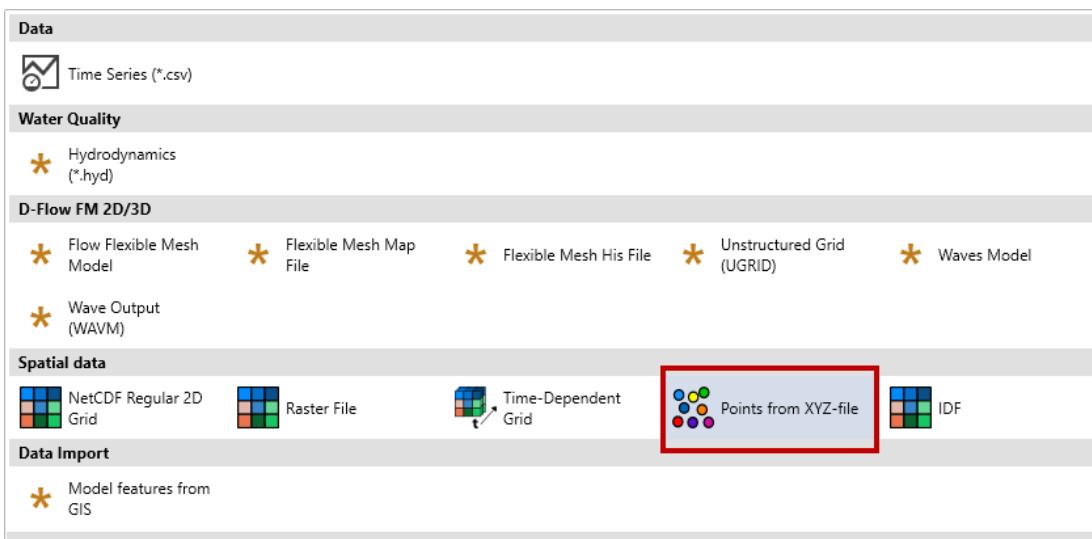


Figure H.3: Importing a point cloud into the project using the "Import" drop-down menu in the Spatial Operations ribbon



Figure H.4: Activate the imported point cloud in the spatial editor by double clicking it in the project tree

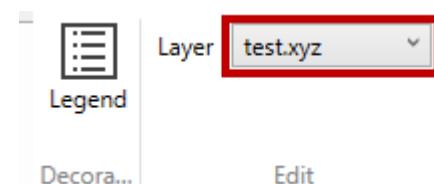


Figure H.5: Activate the imported point cloud in the spatial editor by selecting it from the dropdown box in the Map ribbon

H.2.3 Activate (spatial) model quantity

Similar to activating an imported dataset in the spatial editor, you can also activate a (spatial) model quantity (e.g. bed level, initial conditions, roughness, viscosity) in the spatial editor by double clicking the quantity in the project tree or selecting it from the dropdown box in the "Spatial Operations" ribbon.

H.2.4 Colorscale

In the spatial editor the colorscale for a spatial quantity can be made visible in the map window by clicking on the "Legend" button in the "Spatial Operations" ribbon (Figure H.6). Then, in the Map window a colorscale is activated (Figure H.7). You can (de-)activate the color scale by clicking on the "Legend" button again. By default the colorscale is ranging from the minimum to the maximum value of the active dataset.

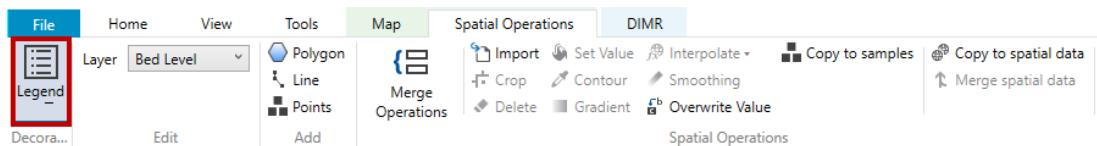


Figure H.6: Legend button in Map ribbon

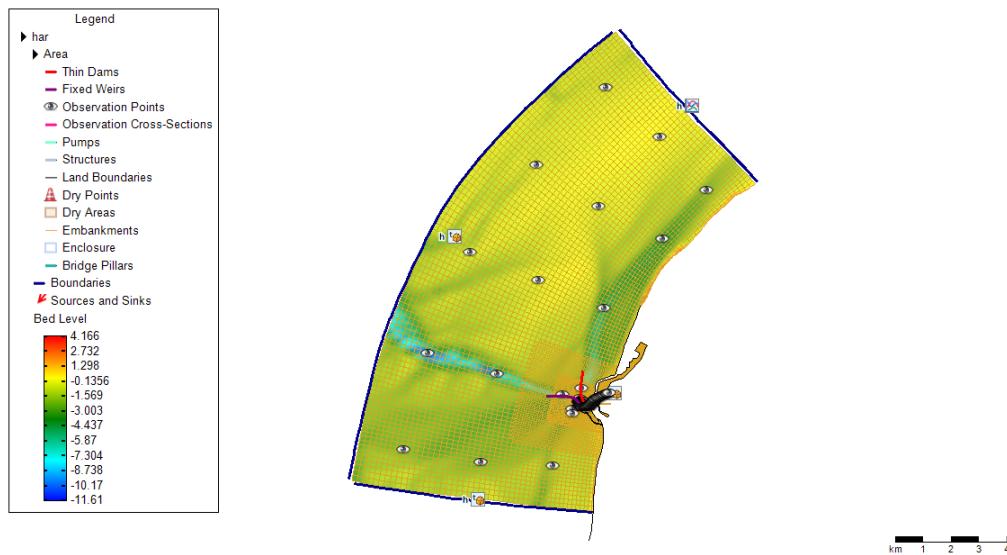


Figure H.7: Activate the colorscale by using the Legend in the map ribbon and the colorscale will become visible in the map window.

You can adjust the colormap and classes of the colorscale using the context menu on the spatial quantity in the "Map tree" and selecting "Legend Properties" (Figure H.8 left panel). A layer properties editor will open in which you can set the colormap to your own preferences (Figure H.8 right panel).

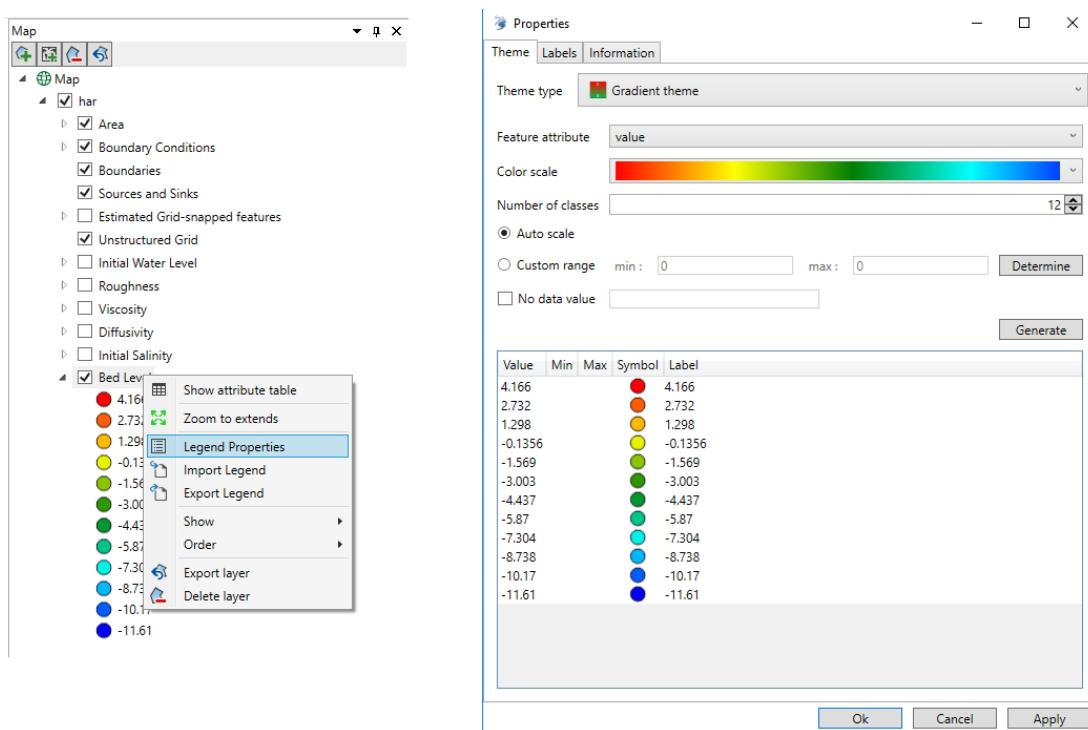


Figure H.8: Edit the colorscale properties using the context menu on the active layer in the Map Tree

H.2.5 Render mode

By default point clouds are rendered as (colored) points and coverages as shades (e.g. 'FillSmooth'). The render mode can be edited using the properties of the active layer [Figure H.9](#). Delta Shell offers the following render modes:

- ◊ Points
- ◊ Lines (only for coverages)
- ◊ Shades or 'FillSmooth' (only for coverages)
- ◊ Colored numbers
- ◊ Mono colored numbers

An example of a coverage rendered as colored numbers is given in [Figure H.10](#).

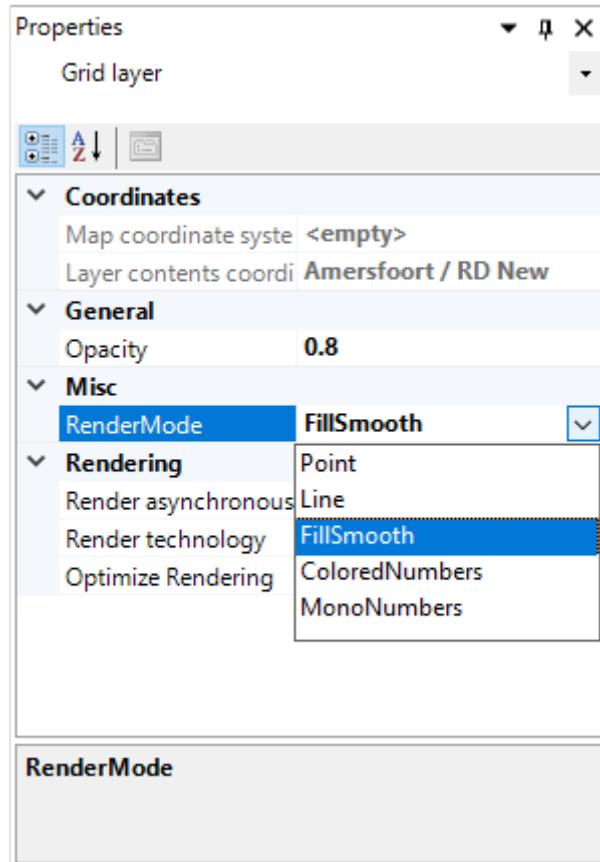


Figure H.9: Select the rendermode for the active layer in the property grid.

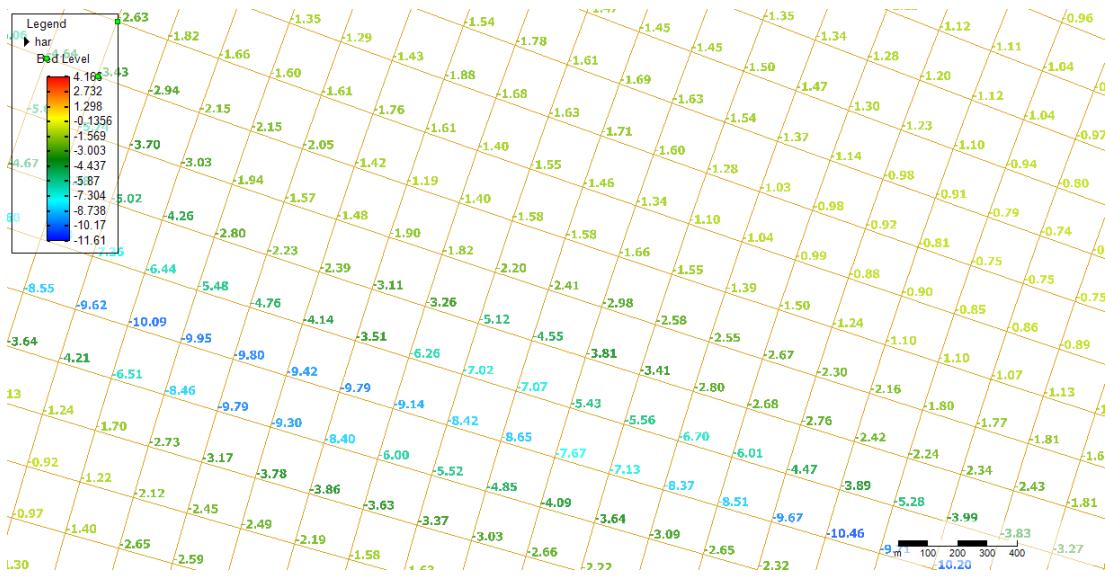


Figure H.10: Example of a coverage rendered as colored numbers.

H.2.6 Context menu

In addition to the selection of spatial operation from the 'Spatial Operations' ribbon (see section H.5), you can also select spatial operations using the context menu (e.g. context menu). After drawing a geometry and clicking the context menu all spatial operation available for the geometry will pop-up (see Figure H.11). The spatial operation will become active upon selecting it from the context menu.

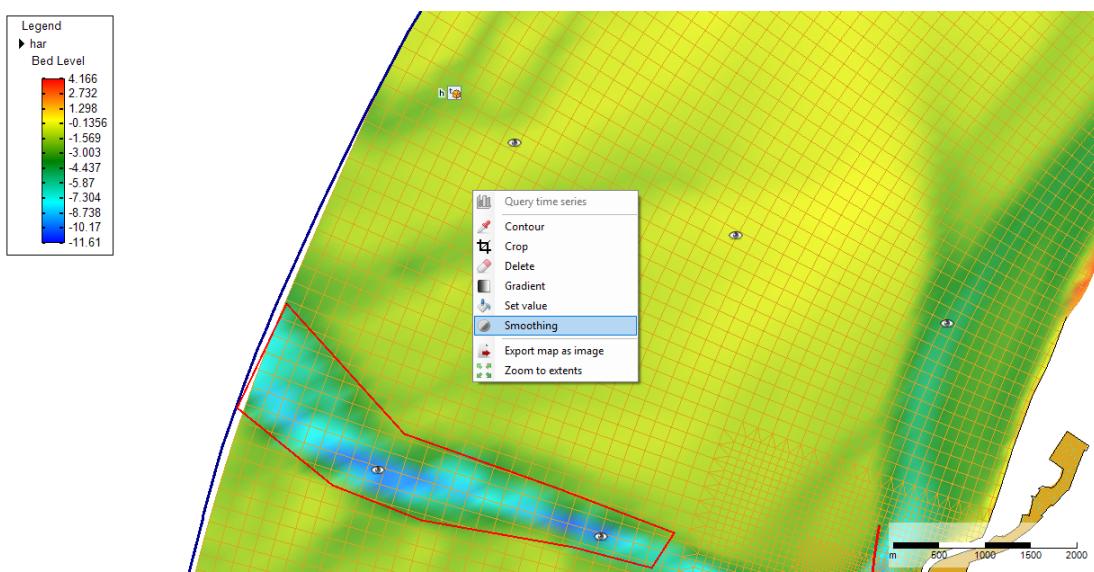


Figure H.11: Selecting a smoothing operation for a polygon geometry from the context menu (using context menu)

H.3 Quantity selection

A spatial quantity can be activated/selected either by double clicking it in the project tree (Figure H.12) or by selecting it from the dropdown box in the “Spatial Operations” ribbon (Figure H.13). Upon selecting the spatial quantity it will be shown as a point cloud (for a dataset) or coverage (for model input) on the central map. Typically, you start from a point cloud (either obtained from import or by generating samples yourself) which will eventually be interpolated to a grid or network (e.g. coverage). The spatial editor will keep track of both the changes made to the point cloud(s) and coverage of the selected spatial quantity. The information will be saved in the Delta Shell project and available the next time you open the project. **Note:** The operations are not yet saved in a human-readable/editable file

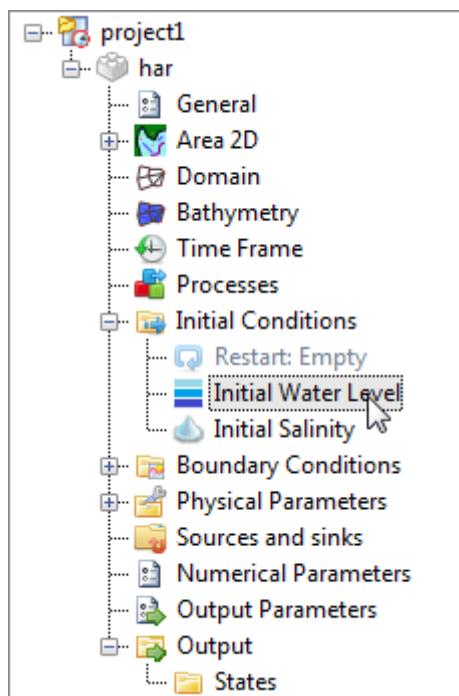


Figure H.12: Activating a spatial quantity by double clicking it in the project tree (in this example ‘Initial Water Level’)

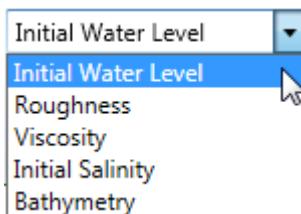


Figure H.13: Activating a spatial quantity by selecting it from the dropdown box in the ‘Spatial Operations’ ribbon

H.4 Geometry operations

The spatial editor supports three types of geometry operations: (1) polygons, (2) lines and (3) points (see also [Figure H.14](#)). The following sub-sections subsequently describe how these geometries can be selected and edited. If you do not select any of these three geometry operations, the spatial operation automatically applies to all the data.



Note: Please note that the drawn geometries are not yet persistent, implying that the geometries once drawn cannot be edited yet. Upon pressing the “Esc” button while in editing mode all drawn geometries will disappear.

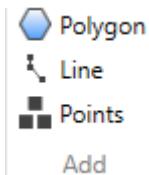


Figure H.14: Overview of the available geometry operations in the ‘Spatial Operations’ ribbon

H.4.1 Polygons

Upon selecting “Polygon” from the “Map” ribbon you can draw one or multiple polygons ([Figure H.15](#)). Each polygon point is defined by a single click with the LMB. The polygon is closed by double clicking the LMB. After drawing the (first) polygon, the available spatial operations for polygons are enabled in the “Spatial Operations” ribbon. The following spatial operations are available for polygons:

- ◊ Crop ([section H.5.2](#))
- ◊ Delete ([section H.5.3](#))
- ◊ Set Value ([section H.5.4](#))
- ◊ Contour ([section H.5.5](#))
- ◊ Gradient ([section H.5.9](#))
- ◊ Smoothing ([section H.5.11](#))
- ◊ Interpolate - only in case samples and a grid/network are available ([section H.5.10](#))
- ◊ Copy to samples ([section H.5.6](#))
- ◊ Copy to spatial data ([section H.5.7](#)) - only for grid coverages

The selected spatial operation applies to all the drawn polygons.

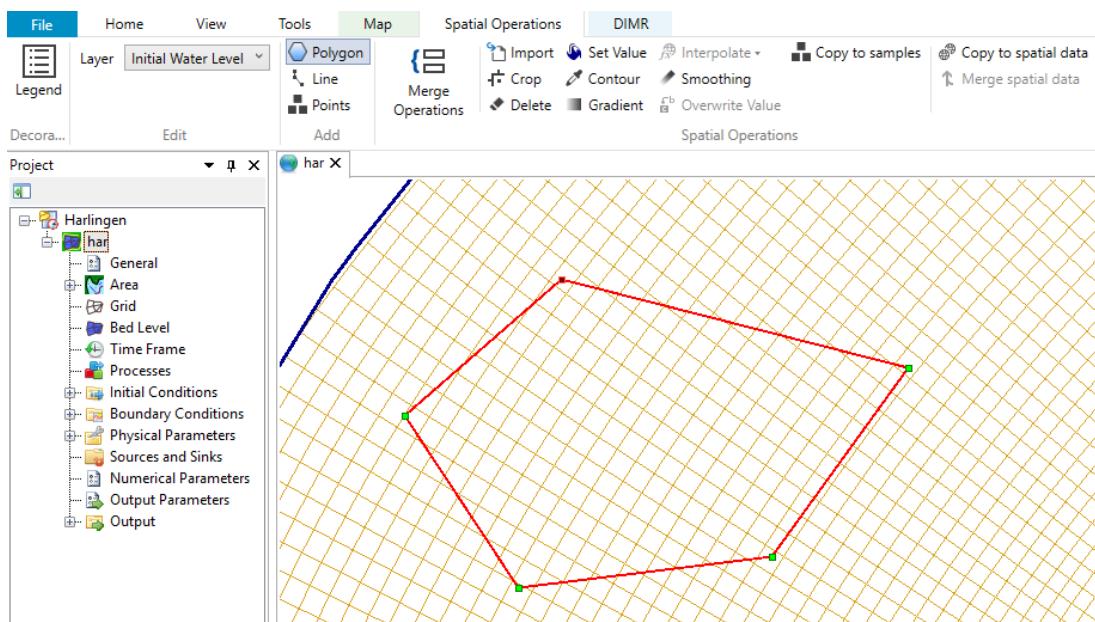


Figure H.15: Activating the polygon operation and drawing polygons in the central map.

H.4.2 Lines

Upon selecting “Line” from the “Map” ribbon you can draw one or multiple lines (Figure H.16). Each line point is defined by a single click with the LMB. The line is completed by double clicking the LMB. After drawing the (first) line, the available spatial operations for lines are enabled in the “Map” ribbon. The following spatial operations are available for lines:

- ◊ Contour (section H.5.5)
- ◊ Copy to samples (section H.5.6)
- ◊ Copy to spatial data (section H.5.7) - only for grid coverages

The selected spatial operation applies to all the drawn lines.

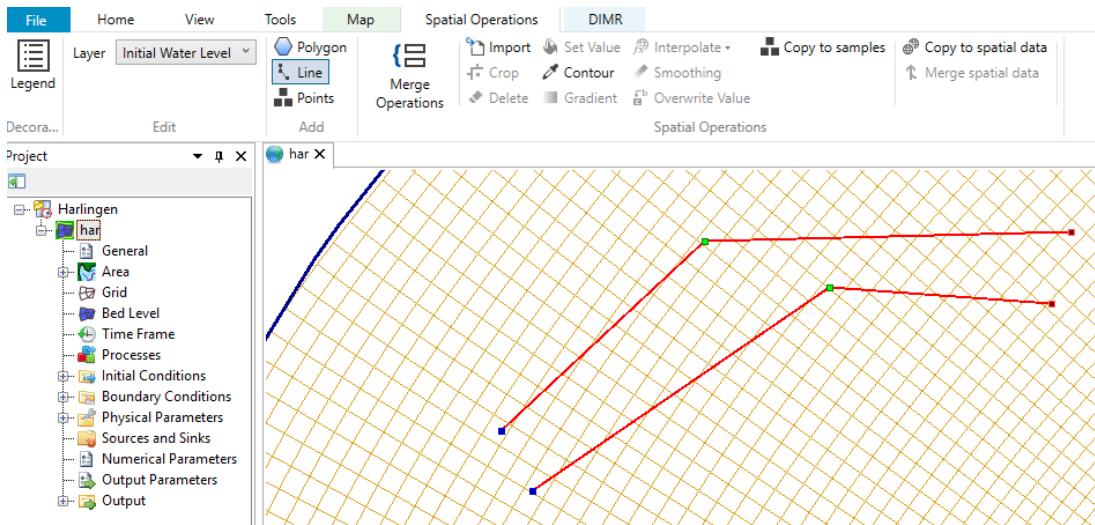


Figure H.16: Activating the line operation and drawing lines in the central map.

H.4.3 Points

Upon selecting “Add points” from the “Map” ribbon you can draw one or multiple points and assign a uniform value to them (Figure H.17). Each point is defined by a single click with the LMB. The group of points is completed by double clicking the LMB. Upon double clicking a popup appears in which you can assign a value to the points.

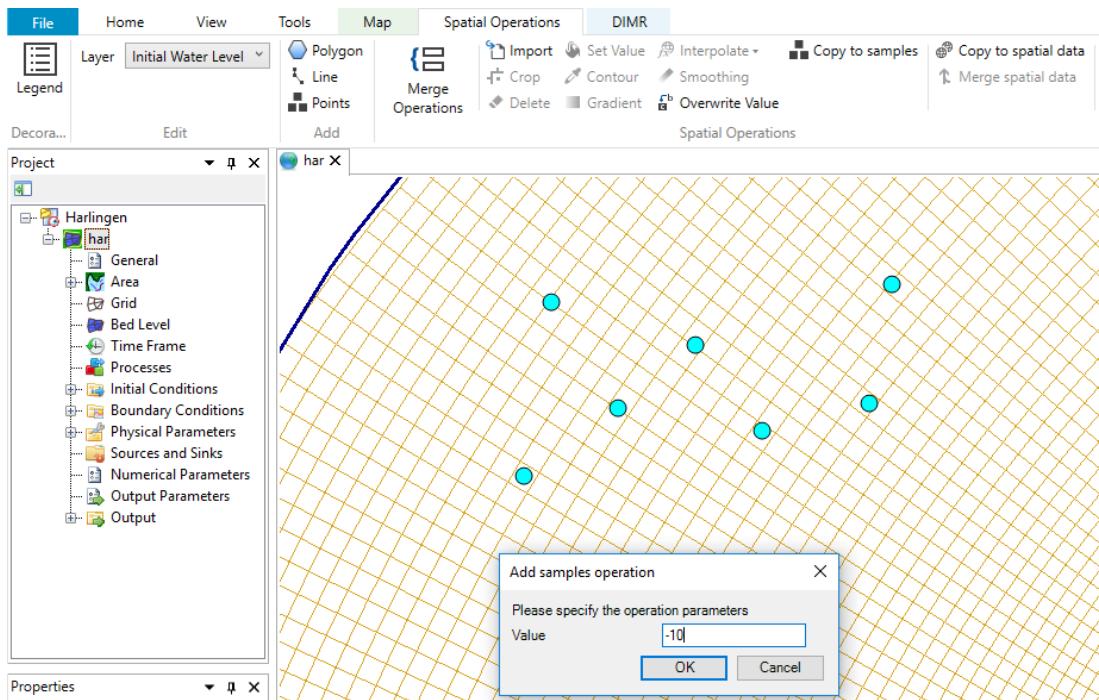


Figure H.17: Activating the ‘Add points’ operation, drawing them in the central map and assigning a value to them.

H.5 Spatial operations

The spatial editor supports the following spatial operations (see also Figure H.18):

- ◊ Import (section H.5.1) - only for point clouds
- ◊ Crop (section H.5.2)
- ◊ Delete (section H.5.3)
- ◊ Set Value (section H.5.4)
- ◊ Contour (section H.5.5) - only for point clouds
- ◊ Gradient (section H.5.9)
- ◊ Interpolate (section H.5.10) - only for point clouds
- ◊ Smoothing (section H.5.11)
- ◊ Change single value (section H.5.12) - only for grid coverages
- ◊ Merge spatial data (section H.5.8) - only for grid coverages
- ◊ Copy to samples (section H.5.6) - only for grid coverages
- ◊ Copy to spatial data (section H.5.7) - only for grid coverages

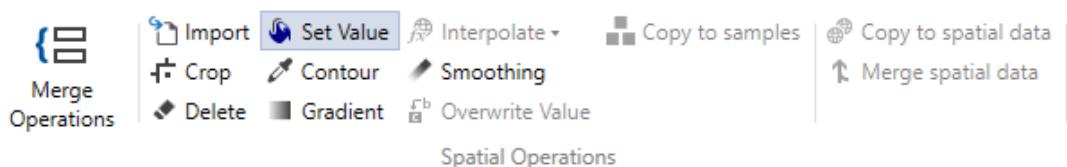


Figure H.18: Overview of the available spatial operations in the ‘Spatial Operations’ ribbon

The sections below provide a description of each operation.

H.5.1 Import point cloud

With the import operation you can import a point cloud for the selected spatial quantity. For this operation no geometry is required. The import operation is activated from the 'Spatial Operations' ribbon (Figure H.19). Upon importing a point cloud you are asked whether a coordinate transformation should be applied to the imported dataset (Figure H.20). By default it will be assumed that the imported data is in the same coordinate system as the model. If not, you can indicate from which to which coordinate system the data should be transformed. After import the point cloud is added to the operations stack (Figure H.21). The difference between this importer and importing a point cloud on the project level in the project tree (section H.2.2) is that for this importer the point cloud is directly assigned to the selected spatial quantity (e.g. model input) instead of being treated as a separate dataset.

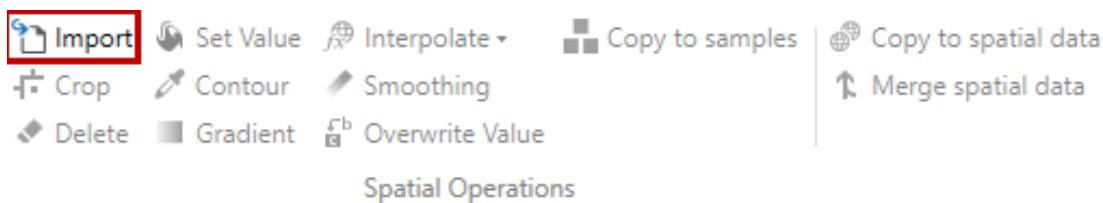


Figure H.19: Importing a point cloud using the 'Import' operation from the 'Spatial Operations' ribbon

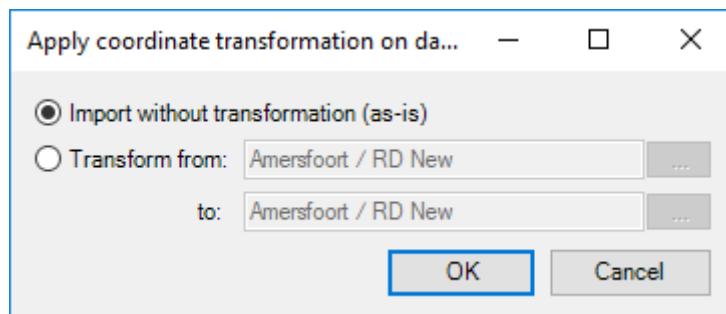


Figure H.20: Option to perform a coordinate transformation on the imported point cloud

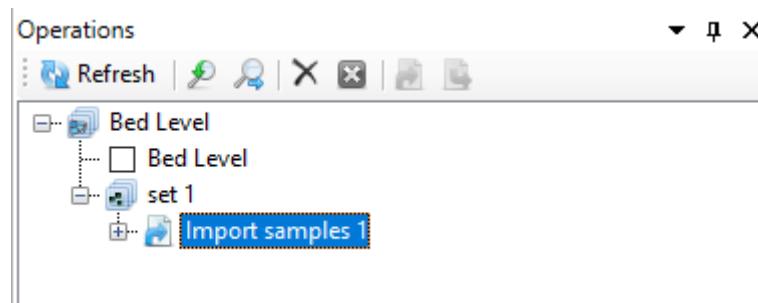


Figure H.21: Appearance of import point cloud operation in the operations stack

H.5.2 Crop

The crop operation crops a point cloud or coverage (depending on which one is active). The crop operation is activated from the 'Spatial Operations' ribbon and only available for polygon geometries. You can control which part of the data should be deleted by using polygons. If you provide a polygon outside the point cloud or coverage, all data will be deleted. For an example see Figure H.22.

After cropping (part of) the point cloud or coverage the operation is added to the operations stack (Figure H.23).

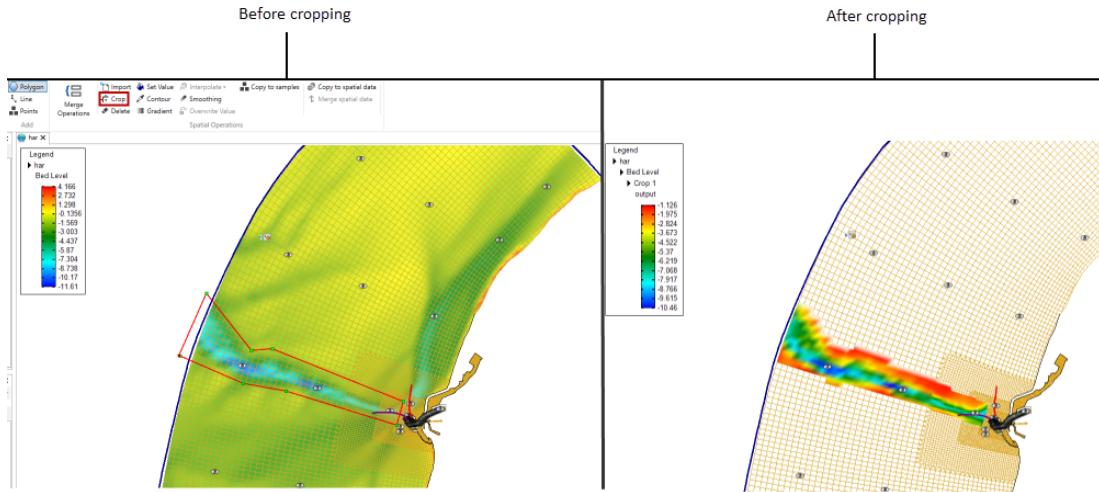


Figure H.22: Performing a crop operation on a point cloud with a polygon using 'Crop' from the 'Spatial Operations' ribbon

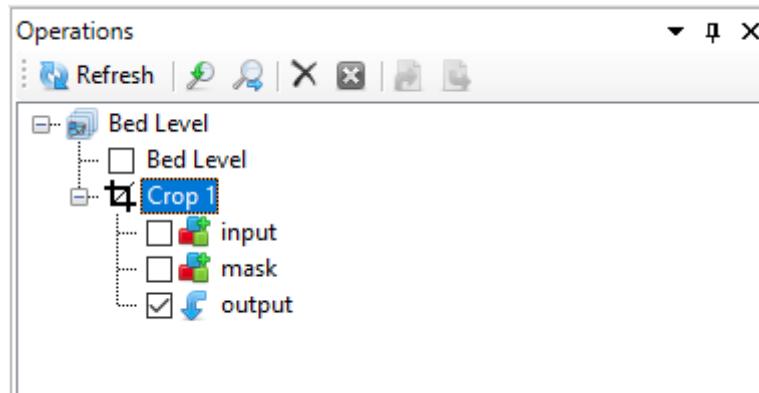


Figure H.23: Appearance of crop operation in the operations stack

H.5.3 Delete

The delete operation deletes (part of) a point cloud or coverage (depending on which one is active). The delete operation is activated from the 'Spatial Operations' ribbon. You can control which part of the data should be deleted by using polygons. If no polygons are provided, the total dataset will be deleted. For an example see Figure H.24. After erasing (part of) the point cloud or coverage the operation is added to the operations stack (Figure H.25).

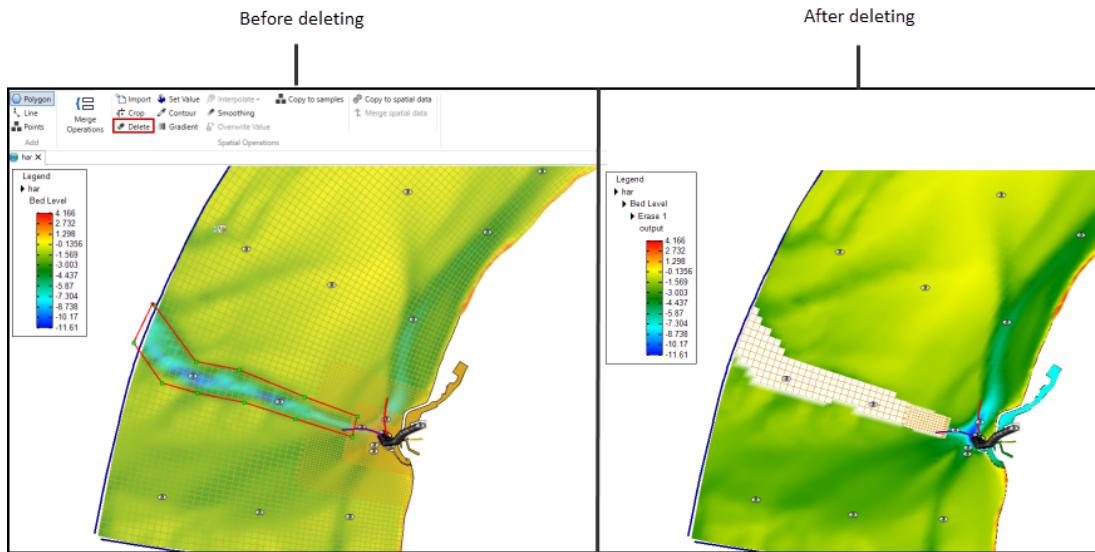


Figure H.24: Performing an delete operation on a point cloud with a polygon using 'Delete' from the 'Spatial Operations' ribbon

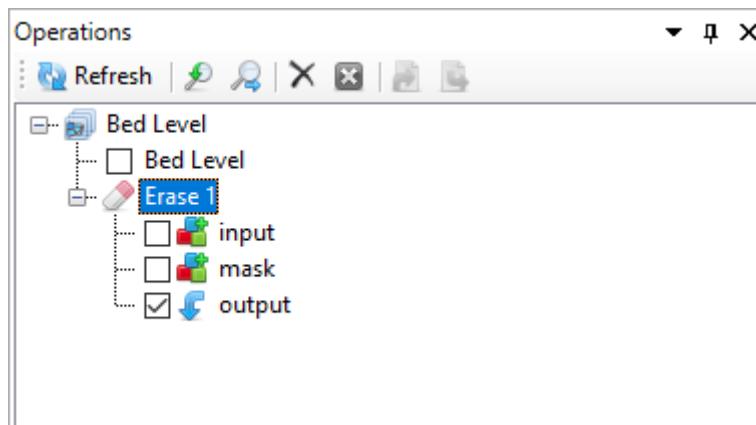


Figure H.25: Appearance of delete operation in the operations stack

H.5.4 Set value

The set value operation assigns a value to a point cloud or coverage (depending on which one is active). The set value operation is activated from the 'Spatial Operations' ribbon and only available for polygon geometries or for the total data set if no polygon is provided. By assigning a value, the user can choose from the following operations:

- ◊ **Overwrite** : overwrites all existing points within the polygon (excluding no data values) with the uniform value
- ◊ **Overwrite where missing (only for coverages)** : overwrites all missing values within the polygon with the uniform value
- ◊ **Add** : Adds the uniform value to all existing points within the polygon (excluding no data values)
- ◊ **Subtract** : Subtracts the uniform value from all existing points within the polygon (excluding no data values)
- ◊ **Multiply** : Multiplies all existing points within the polygon (excluding no data values) with the uniform value
- ◊ **Divide** : Divides all existing points within the polygon (excluding no data values) by the uniform value
- ◊ **Maximum** : Sets all existing points within the polygon (excluding no data values) to the maximum of its current value and the uniform value

- ◇ **Minimum** : Sets all existing points within the polygon (excluding no data values) to the minimum of its current value and the uniform value

For an example see [Figure H.26](#). After performing a set value operation to (part of) the point cloud or coverage the operation is added to the operations stack ([Figure H.27](#)).

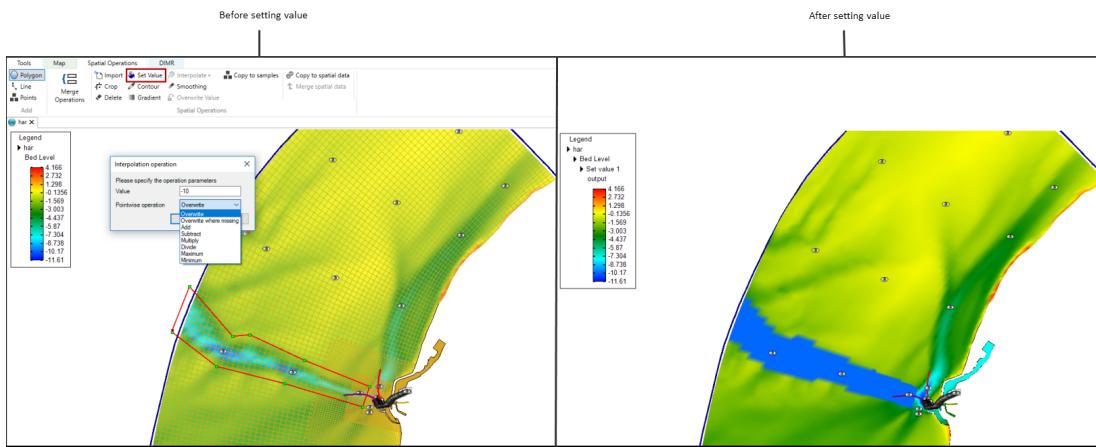


Figure H.26: Performing a set value operation (e.g. overwrite) on a point cloud with a polygon using 'Set Value' from the 'Spatial Operations' ribbon

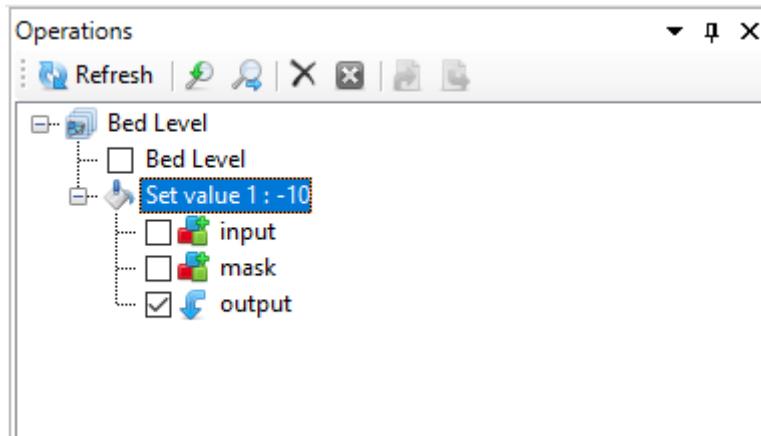


Figure H.27: Appearance of set value operation in the operations stack

H.5.5 Contour

The contour operation creates a point cloud with a uniform value along a line or polygon (depending on which one is active). The contour operation is activated from the 'Spatial Operations' ribbon. After drawing the lines or polygons you have to assign the uniform value (argument) and the sampling interval in m. This spatial operation can be useful to digitalize information from nautical charts for example. In this case you first have to import the nautical chart as a geotiff ([Figure H.28](#)), set the right map coordinate system ([Figure H.29](#)) and then use the contour operation [Figure H.30](#). Sometimes the samples are created behind the geotiff. Then you can use the context menu on the samples layer in the Map tree to bring the samples to the front ([Figure H.31](#)). After applying the contour operation it is added to the stack ([Figure H.32](#)).

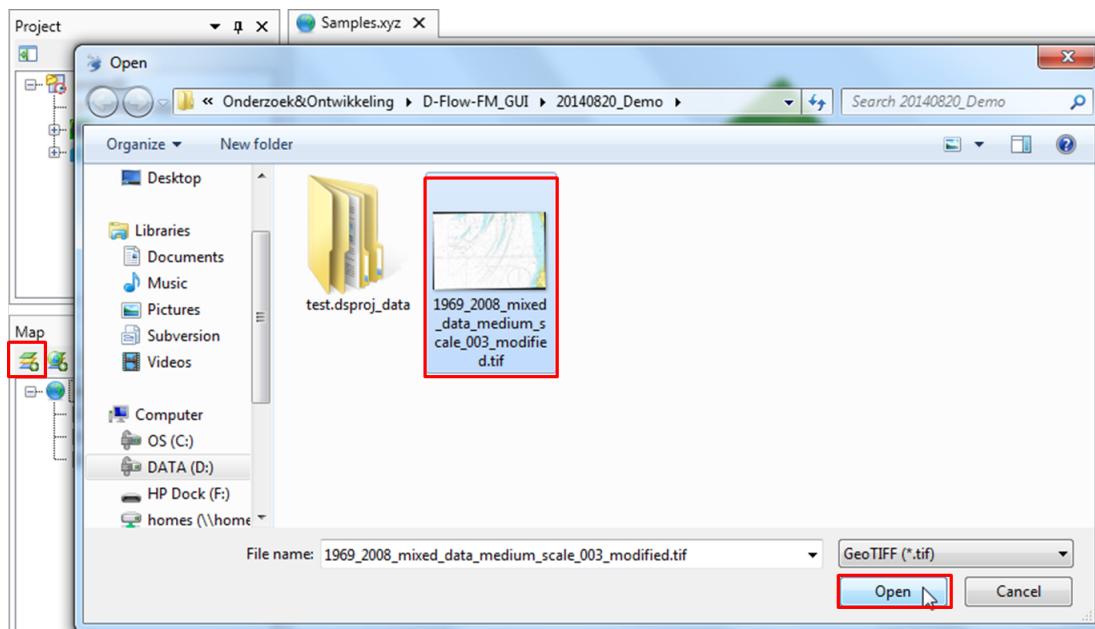


Figure H.28: Import a nautical chart as a georeferenced tiff file

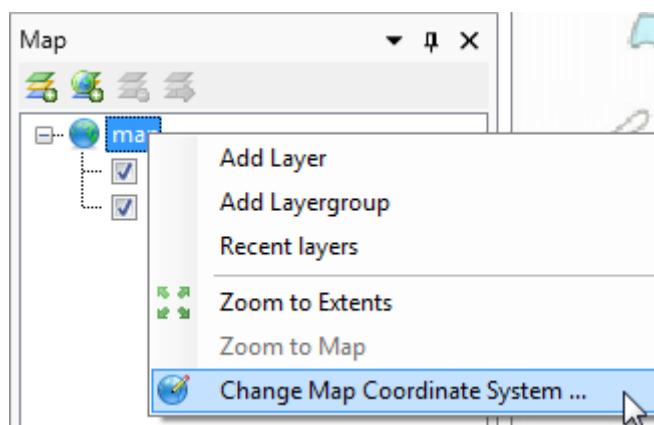


Figure H.29: Set the right map coordinate system for the geotiff

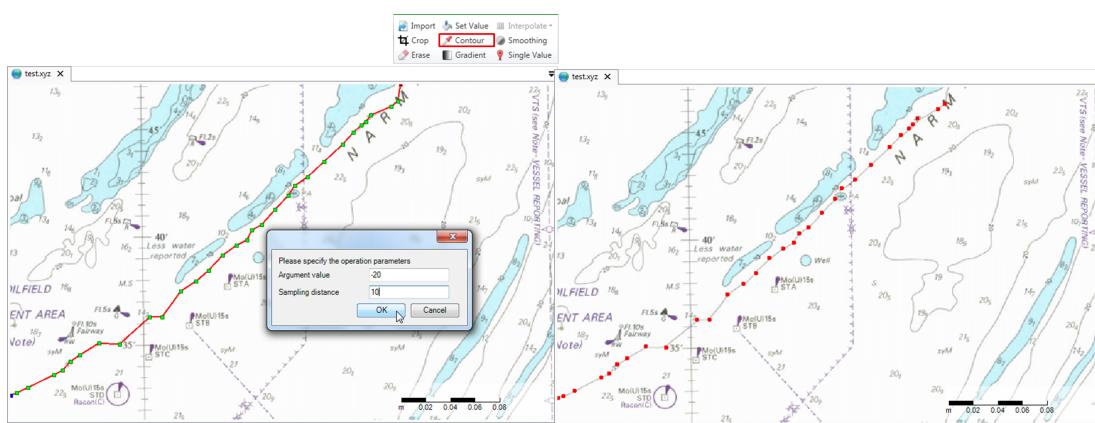


Figure H.30: Performing a contour operation on a nautical chart using lines to define the contours and 'Contour' from the 'Spatial Operations' ribbon

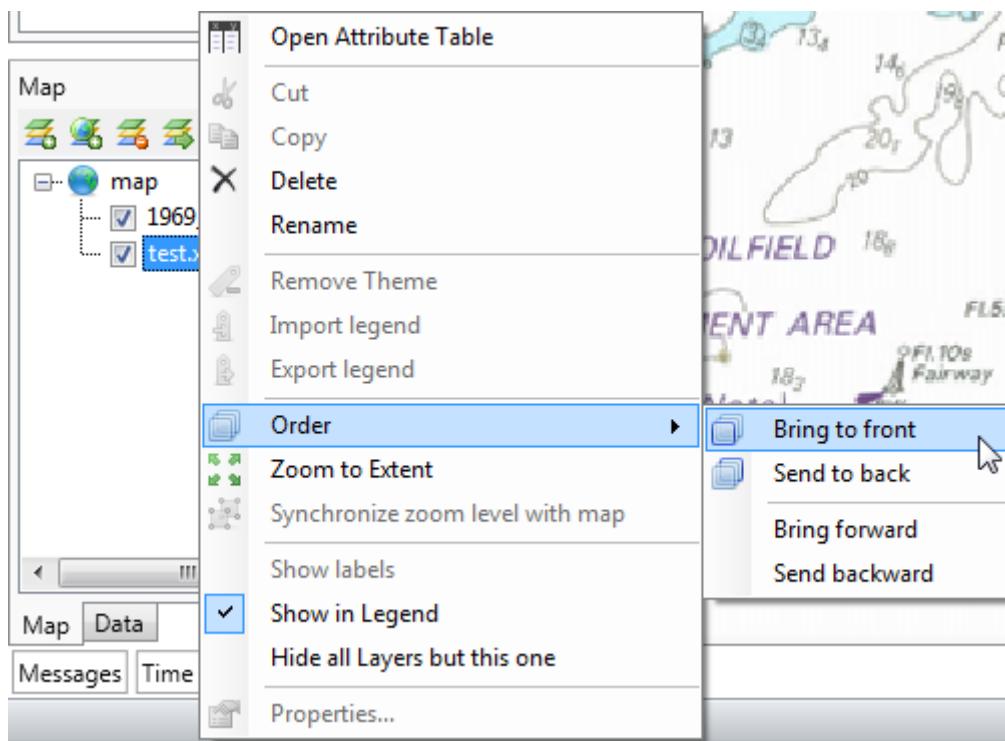


Figure H.31: Bring the sample set to the front if it appears behind the nautical chart

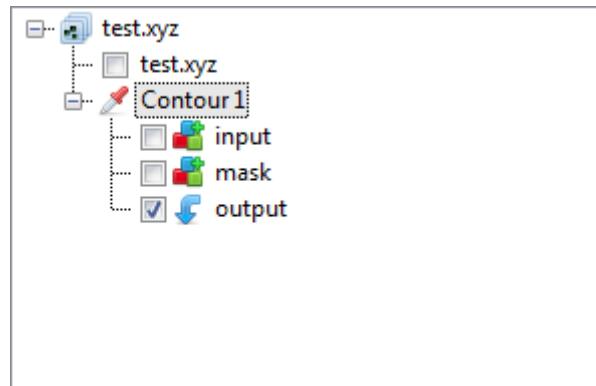


Figure H.32: Appearance of contour operation in the operations stack

H.5.6 Copy to samples

This operation converts the currently selected grid coverage to a sample set, which becomes that starting point of a new subset. If polygons have been drawn, the operation will confine the copy to the interiors. The operation will not convert missing values to samples. Note that the operation does keep a reference to the original copied grid coverage; if it is changed by a re-evaluation of the stack, the changes will affect the output point cloud.

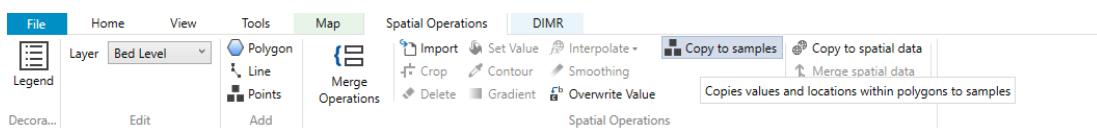


Figure H.33: Applying the copy to samples operation



Figure H.34: Copy to samples operation result

H.5.7 Copy to spatial data

This operation simply clones the grid coverage, starting a new subset with a snapshot of the currently selected operation output. Similarly to H.5.6, it keeps a reference to the input data and will perform the clone again upon re-evaluation.

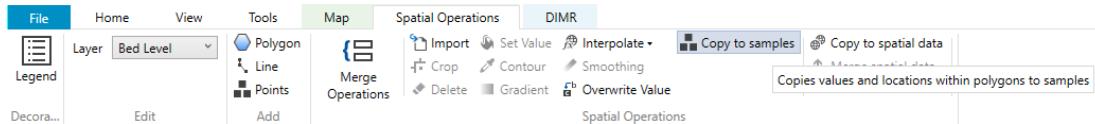


Figure H.35: Applying the copy spatial data operation

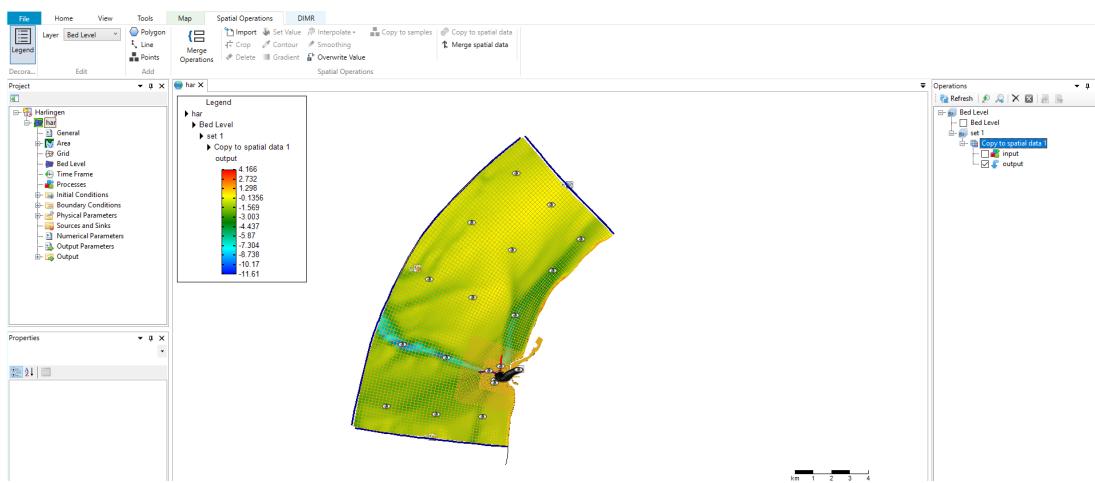


Figure H.36: Copy spatial data operation result

H.5.8 Merge spatial data

Whenever a subset contains a grid coverage as its editing data (after applying e.g. H.5.7) on the same grid as the main operation set, its result can be combined with the main set by applying this operation, similarly to the interpolation operation for sample sets, discussed below. Combining the grid coverages can be achieved with the usual point-wise methods.

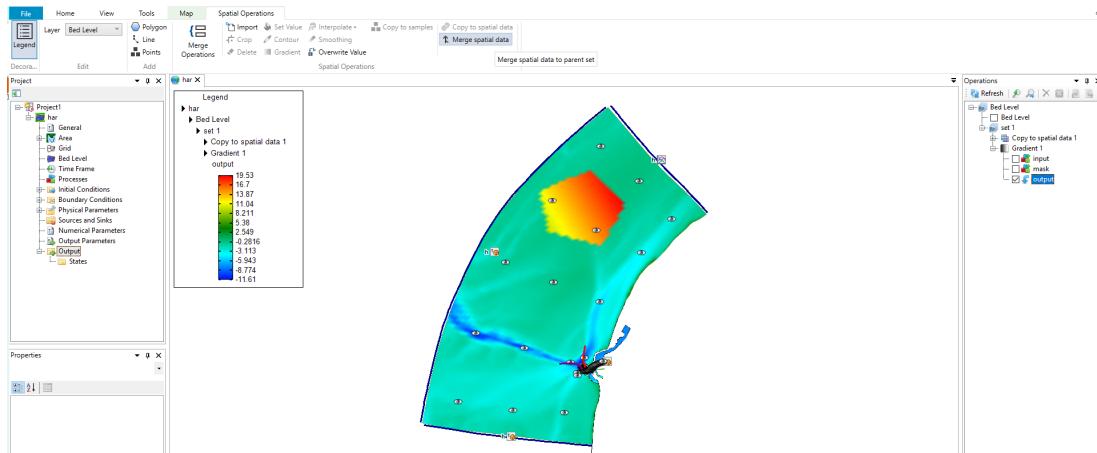


Figure H.37: Activating the merge spatial data tool from the ribbon

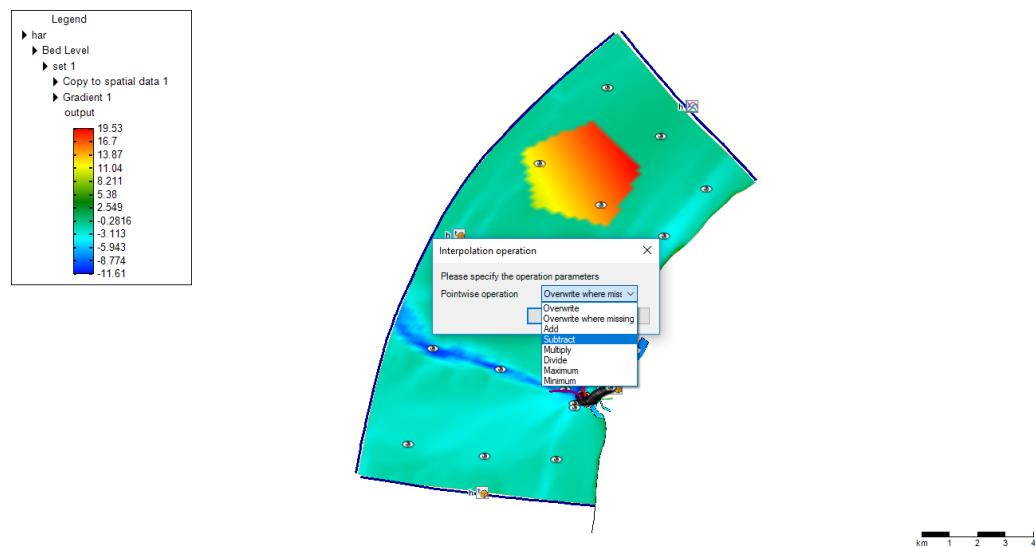


Figure H.38: The merge operation requests a pointwise combination method

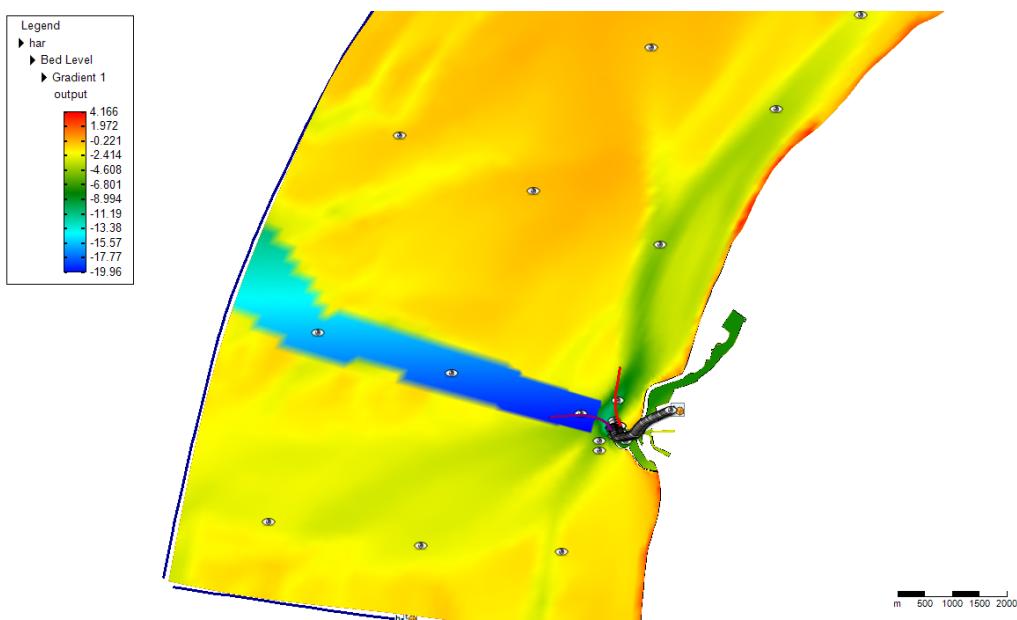


Figure H.39: Resulting grid coverage

H.5.9 Gradient

The gradient operation applies a gradient to a point cloud or grid coverage (depending on which one is active). The gradient operation is activated from the ‘Spatial Operations’ ribbon and only available for polygon geometries or for the total data set if no polygon is provided. You have to assign the initial (start) value, the final (end) value and the going to angle (according to the Cartesian convention with 0 degrees is East, 90 degrees is North, etc **Note:** This is not working properly yet). For an example see Figure H.40. After applying a gradient to (part of) the point cloud or coverage the operation is added to the operations stack (Figure H.41).

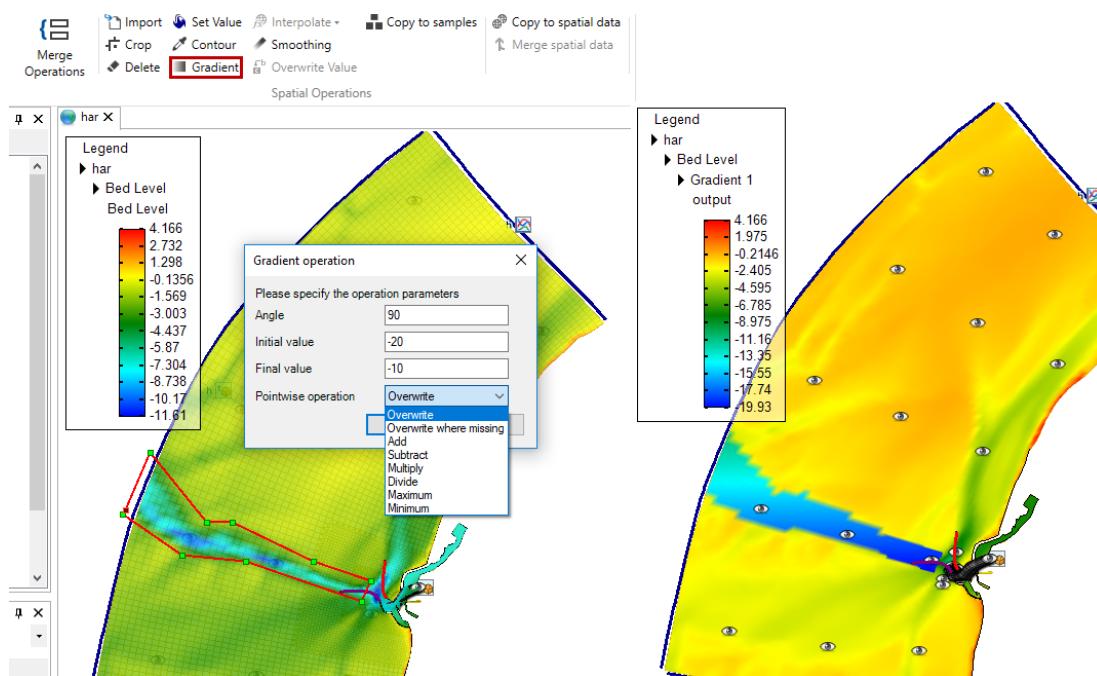


Figure H.40: Performing a gradient operation on a point cloud with a polygon using ‘Gradient’ from the ‘Spatial Operations’ ribbon

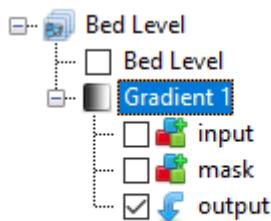


Figure H.41: Appearance of gradient operation in the operations stack

H.5.10 Interpolate

The interpolate operation is the way to get sample set(s) to a grid or network (e.g. coverage). In the operation stack this means that we are actively switching from working on a point cloud (helping to construct the coverage) to working on the selected coverage (or spatial quantity). The interpolation is performed on the data within a polygon or polygons (if provided) or all the data (if no polygons are provided). The interpolate operation can be performed on a single (selected) sample set or on multiple sample sets. Both methods are discussed below. The methods for interpolation are either

- ◊ Triangulation: performs a Delauney triangulation on the sample point set before projecting onto the grid.
- ◊ Averaging: combines sample points within a possible enlarged cell according to an algorithm of choice. The user can set the search cell expansion factor (rel. search cell size) and a threshold for the number of sample points within a cell (minimum sample points), see [Figure H.42](#).

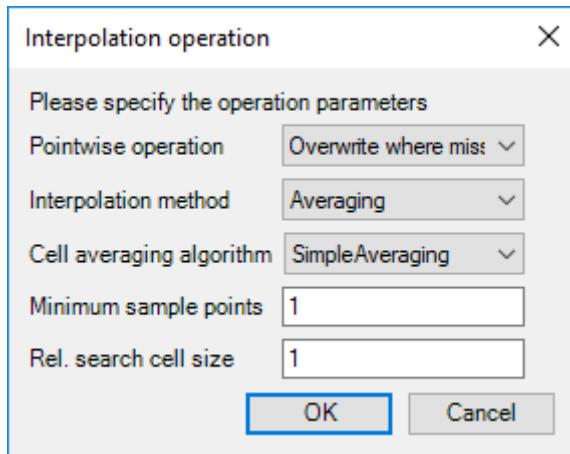


Figure H.42: Interpolation Operation options

Seven Cell averaging algorithms can be chosen by the user; see [Figure H.43](#). These algorithms are explained below:

- ◊ SimpleAveraging: bilinear interpolation is applied, which uses a distance-weighted average of the surrounding samples. The closer the sample point the larger the weighted value.
- ◊ ClosestPoint: the value of the closest sample inside the search area is taken.
- ◊ MaximumValue: the maximum value of the samples inside the search area is taken.
- ◊ MinimumValue: the minimum value of the samples inside the search area is taken.
- ◊ InverseWeightedDistance: Instead of a distance-weighted average (w) in case of the inverse of SimpleAveraging, the distance-weighted average ($1/w$) is taken. The closer the sample point the smaller the weighted value.
- ◊ MinAbs: the minimum of the absolute values of the samples inside the search area is taken.
- ◊ KdTree: This is an obsolete option, which will be removed in a future release.

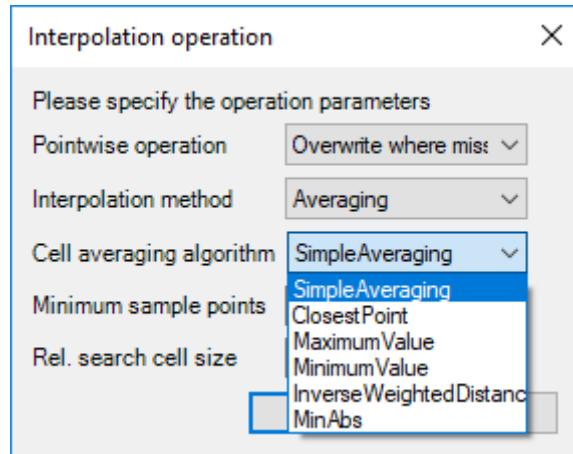


Figure H.43: Averaging options

By default, the interpolation will only overwrite missing values in the gridded data set. However, if the grid coverage already contains values, the user may choose to overwrite or combine the data by a pointwise arithmetic operation.

Interpolate single (selected) set

To perform interpolation on a single sample set, select the sample set (i.e. 'set1') in the operation stack and press 'Interpolate' in the 'Map' ribbon (Figure H.44). Since no polygon is provided in this example, all the samples will be interpolated to the grid. Use polygons if you would like to have more control over the interpolation. After the interpolation the operation is added to the operations stack (Figure H.45). Please note that after performing the interpolation the workflow in the stack is shifting from the sample set (i.e. 'set1' - which was a side step to construct the coverage) to the coverage (i.e. 'bed level').

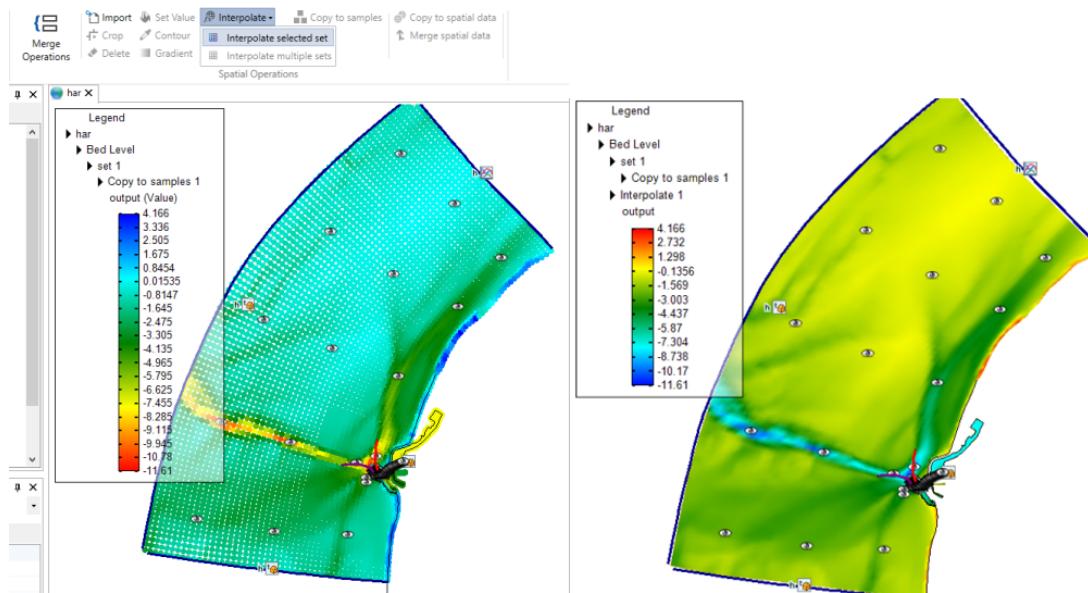


Figure H.44: Performing an interpolation operation on a single sample set (without using a polygon) using 'Interpolate' from the 'Spatial Operations' ribbon

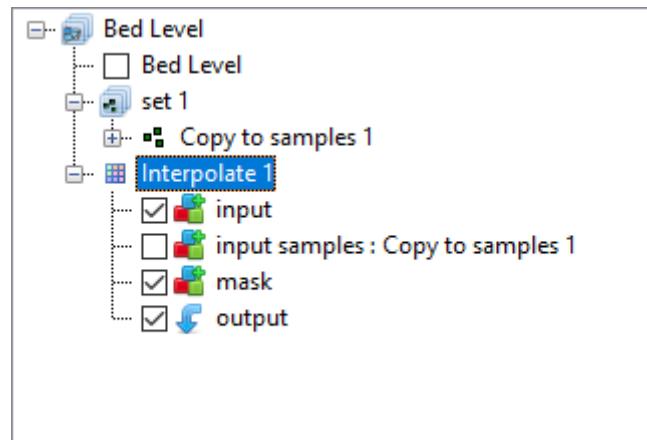


Figure H.45: Appearance of interpolation of 'set1' to the coverage 'bed level' in the operations stack

Interpolate multiple sets

To perform interpolation on multiple sample sets, select the active coverage (i.e. 'bed level') in the operation stack and press 'Interpolate' in the 'Map' ribbon (Figure H.46). In the popup you can select which sample sets to include in the interpolation (in this example both). Since no polygon is provided, all the samples (from the two sets) will be interpolated to the grid. Use polygons if you would like to have more control over the interpolation. After the interpolation the operation is added to the operations stack (Figure H.47). Again note that after performing the interpolation the workflow in the stack is shifting from the sample set (which was a side path to construct the coverage) to the coverage (i.e. bed level).



Note: Please note that interpolation of multiple sample sets can also be achieved by importing/combing different sample sets into the same set in the stack instead of using two separate sets. In this case you can just interpolate the single (selected) set.

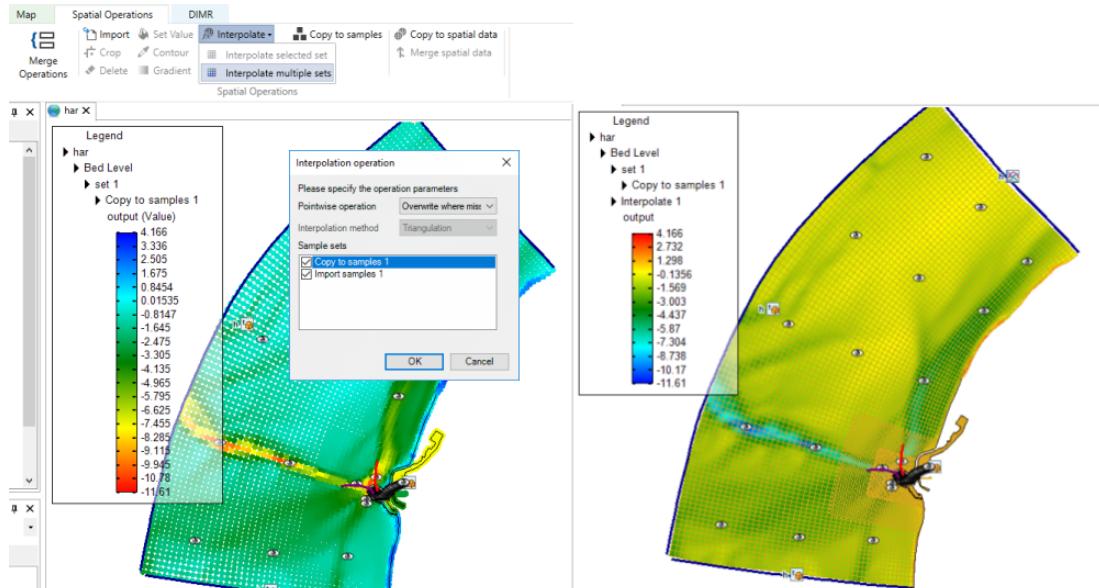


Figure H.46: Performing an interpolation operation on multiple sample sets (without using a polygon) using 'Interpolate' from the 'Spatial Operations' ribbon

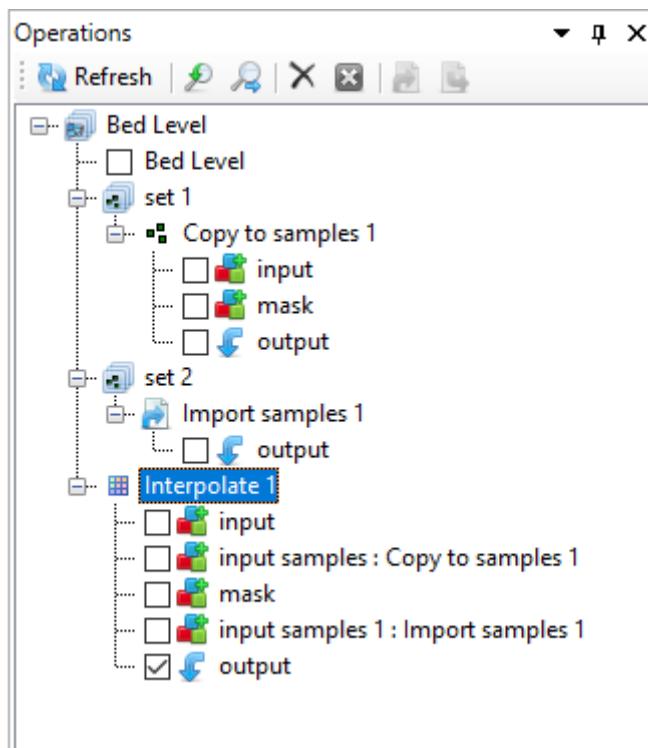


Figure H.47: Appearance of interpolation of 'set1' and 'set2' to the coverage 'bed level' in the operations stack

H.5.11 Smoothing

The smoothing operation smooths out (steep) gradients in a point cloud or coverage (depending on which one is active). The smoothing operation is activated from the 'Spatial Operations' ribbon and only available for polygon geometries or for the total data set if no polygon is provided. You have to assign the smoothing exponent and number of smoothing steps. The higher the exponent and the number of smoothing steps, the heavier the smoothing. For an example see [Figure H.48](#). After applying smoothing to (part of) the point cloud or coverage the operation is added to the operations stack ([Figure H.49](#)).

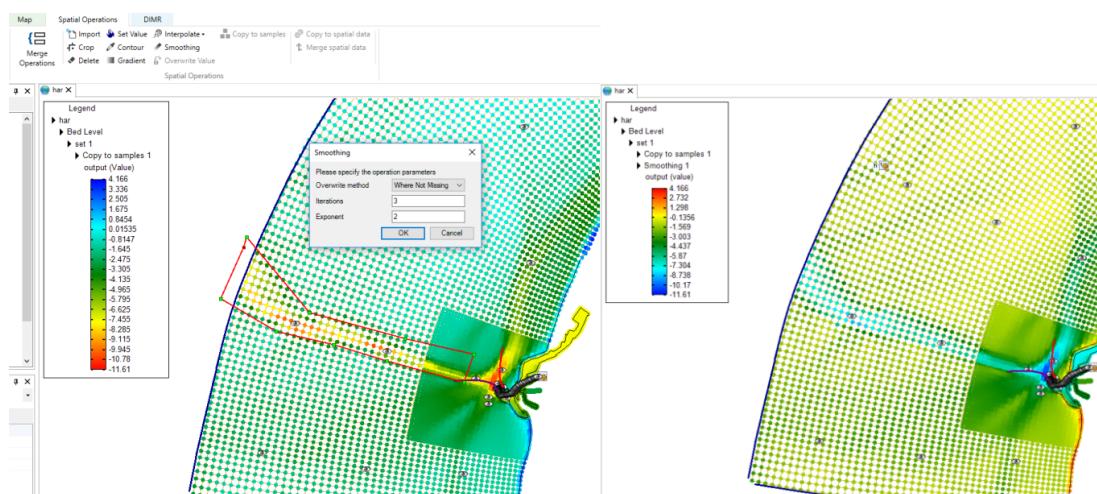


Figure H.48: Performing a smoothing operation on a point cloud with a polygon using 'Smoothing' from the 'Spatial Operations' ribbon

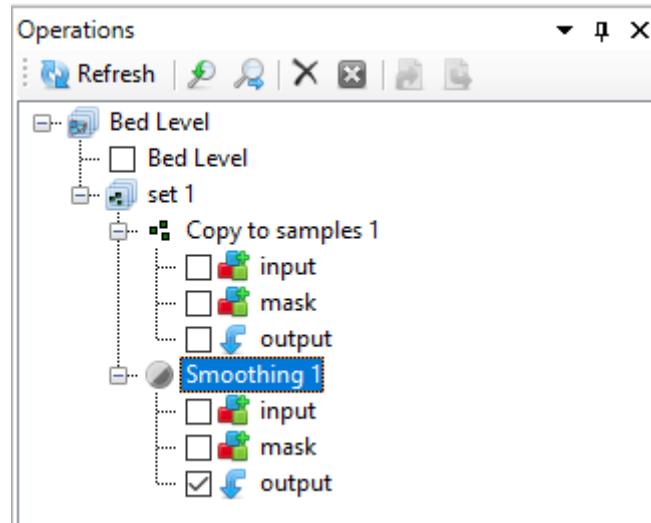


Figure H.49: Appearance of smoothing operation in the operations stack

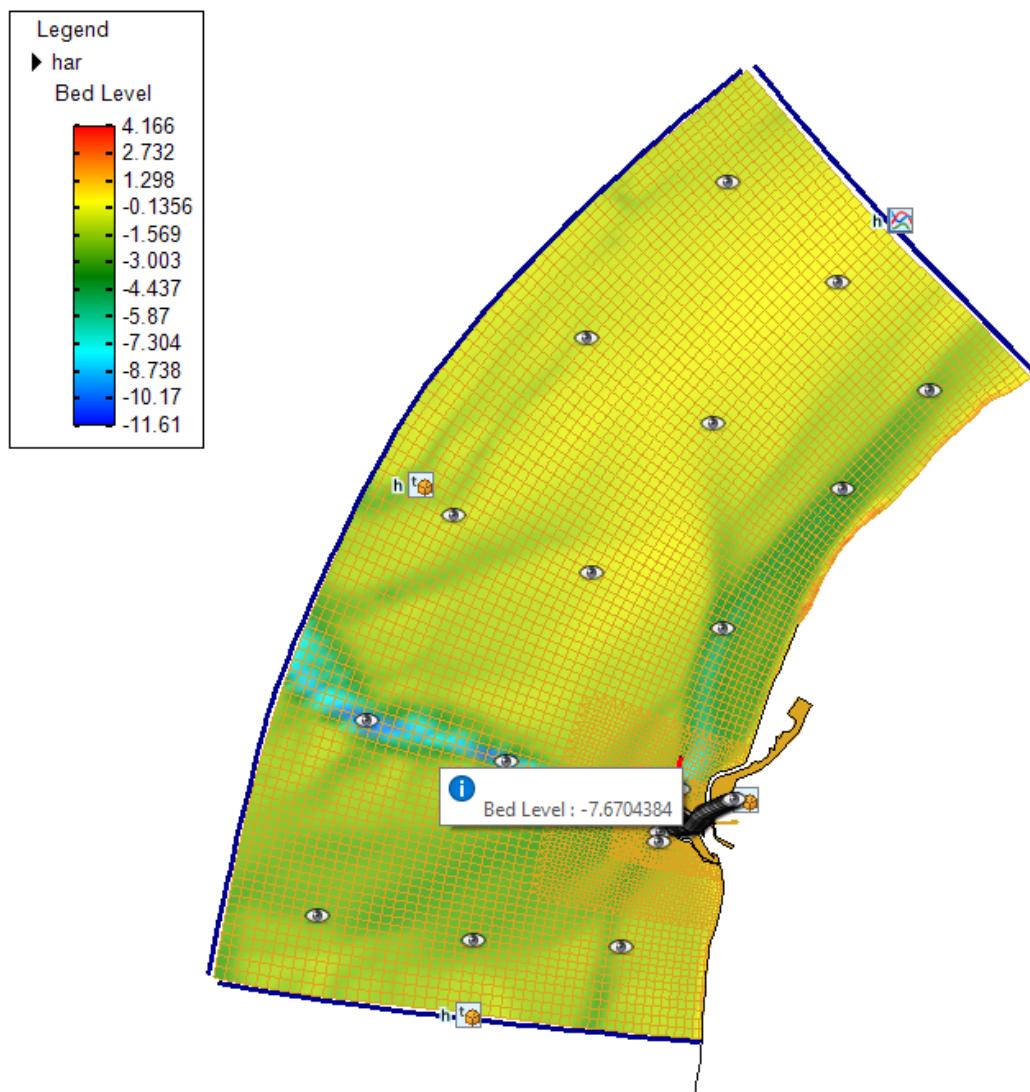


Figure H.50: The cursor for the overwrite operation showing the value of the closest coverage point

H.5.12 Overwrite (single) value

The ‘overwrite (single) value’ operation allows you to edit single values on the active coverage after the interpolation. The ‘overwrite (single) value’ operation is activated from the ‘Spatial Operations’ ribbon. There is no geometry required for this operation. Upon selecting the operation from the ribbon a cursor will become active showing the coverage value closest to the cursor in a tooltipstring (Figure H.50). Upon clicking LMB a popup appears in which you can overwrite the value of this coverage point (Figure H.51). After applying the overwrite operation it is added to the operations stack (Figure H.52).

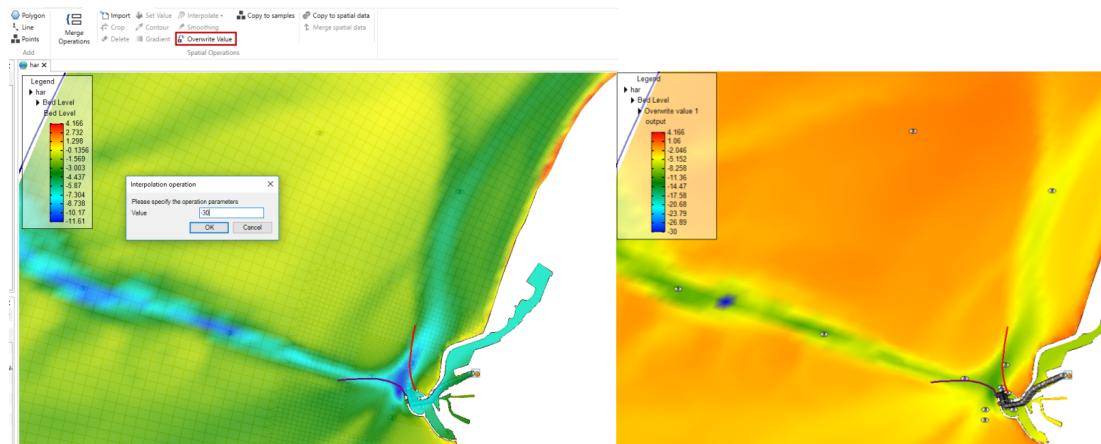


Figure H.51: Performing an overwrite operation on a coverage point using ‘Single Value’ from the ‘Spatial Operations’ ribbon

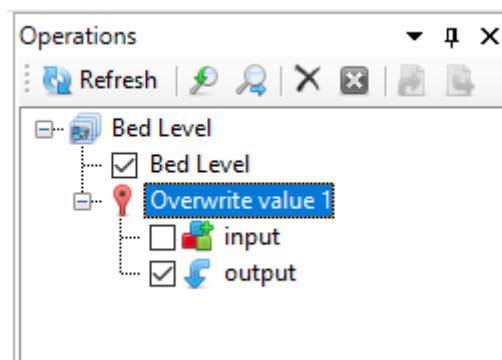


Figure H.52: Appearance of overwrite operation in the operations stack

H.6 Operation stack

The operation stack keeps track of the workflow of spatial operations that you performed. This helps you to make transparent how you arrived at your ‘final’ dataset without having to save all the intermediate datasets (steps) separately. Moreover, the stack is reproducible and easily editable without having to start all over again. This section describes the stack workflow (section H.6.1), how to edit operation properties (section H.6.2), how to enable/disable (section H.6.3), delete (section H.6.4), refresh(section H.6.5) operations, quick links (section H.6.6) and import/export functionality (section H.6.7).

Note: Currently, the stack is saved in the Delta Shell project upon saving the project. The next time you open the project, the stack will reappear. The stack is not (yet) saved in a human readable/editable file.



H.6.1 Stack workflow

Upon performing a spatial operation, the ‘Operations’ panel will open (see Figure H.53) with the operations stack (tree). The stack first shows on which point cloud or coverage you are working (in this

example ‘bed level’). Subsequently, all the operations on this dataset are listed. For each operation you can inspect what the input, mask (e.g. the geometry used for the operation) and output are for the operation (Figure H.54). By default the stack continues from the last operation that you performed. If you wish to work on a different dataset or operation within a dataset, you have to select that dataset or operation in the ‘Operations’ panel with the LMB.

When working on a coverage, point clouds (or sets) can be used to construct the coverage. In that case the stack jumps from the ‘trunk’ to a ‘branch’ and the subsequent operations are performed on the point cloud (see Figure H.53). By selecting the set or coverage in the ‘Operations’ panel you determine on which dataset you are working. The interpolate operation (section H.5.10) allows you to bring data from the point cloud (branch) to the coverage (trunk). See also Figure H.53.

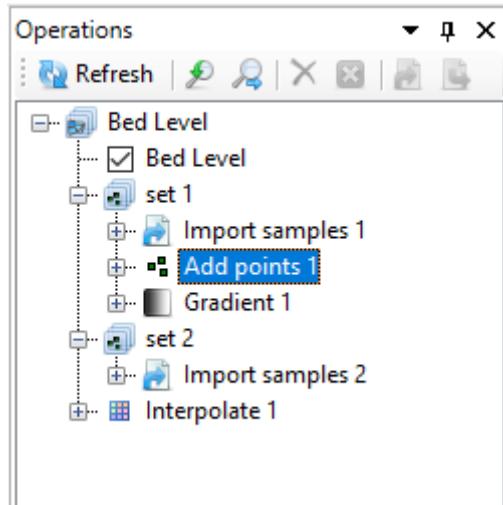


Figure H.53: The ‘Operations’ panel with the operations stack. In this example ‘bed level’ is the coverage (e.g. trunk) that is edited. The point clouds ‘set 1’ and ‘set 2’ (e.g. branches) are used to construct the ‘bed level’ coverage.

H.6.2 Edit operation properties



For each operation that you performed the properties (such as the value or ‘Pointwise operation’ of a ‘Set Value’ operation) can be edited using the ‘Properties’ window (Figure H.55). **Note:** Please note that the mask of an operation cannot (yet) be edited. By editing the operation properties the operation stack becomes ‘out of sync’ and has to be refreshed for the changes to become active (see section H.6.5).

H.6.3 Enable/disable operations

You can (temporarily) enable/disable operations by selecting the operation and pressing boxed cross icon in the stack menu (Figure H.56). Upon disabling an operation the operation will be made grey in the stack and the operation is not taken into account anymore upon evaluation of the overall result. The result of disabling an operation is not directly activated. This is indicated in the stack with the ‘out of sync’ exclamation mark (Figure H.56). You need to refresh the stack (see section H.6.5) for the changes to become active.

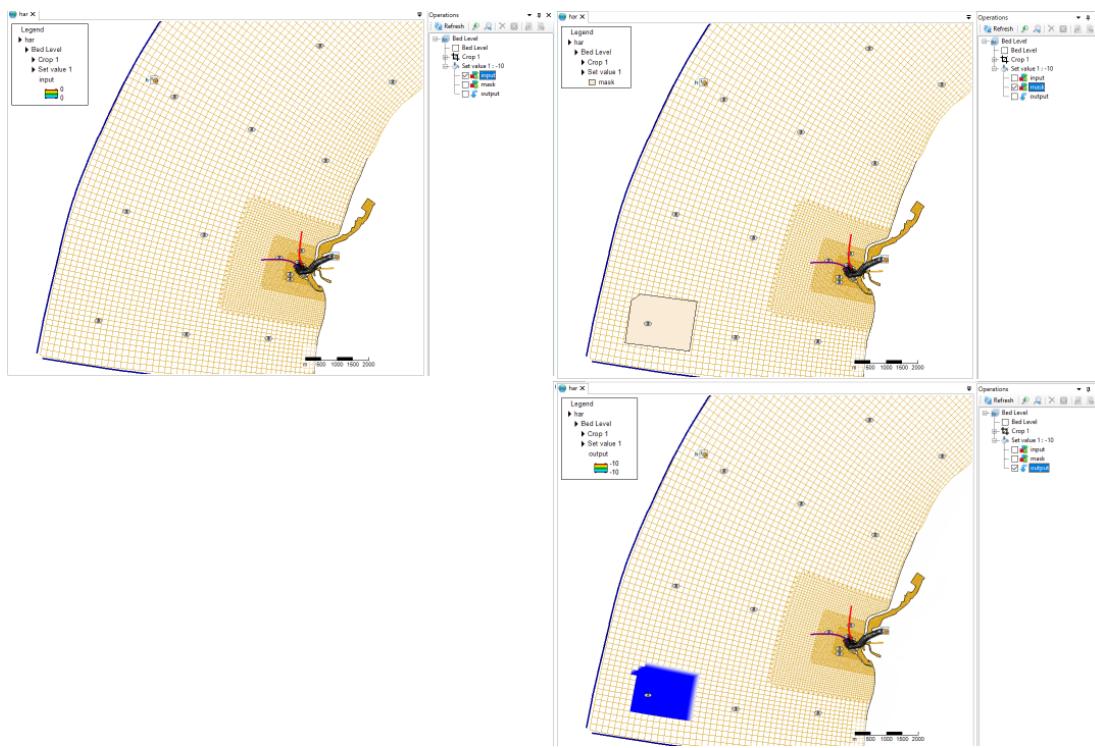


Figure H.54: Input for the operation (top panel), mask for the operation (middle panel) and output of the operation (bottom panel)

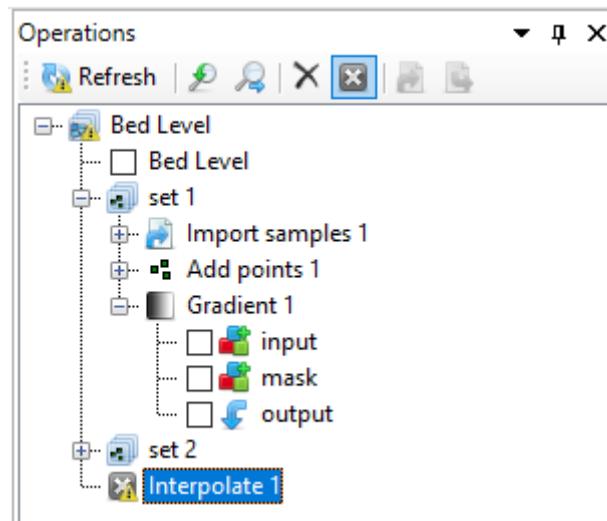


Figure H.56: Disabling an operation using the boxed cross icon in the stack menu. The operation will become grey. Note the exclamation marks marking the stack 'out of sync'.

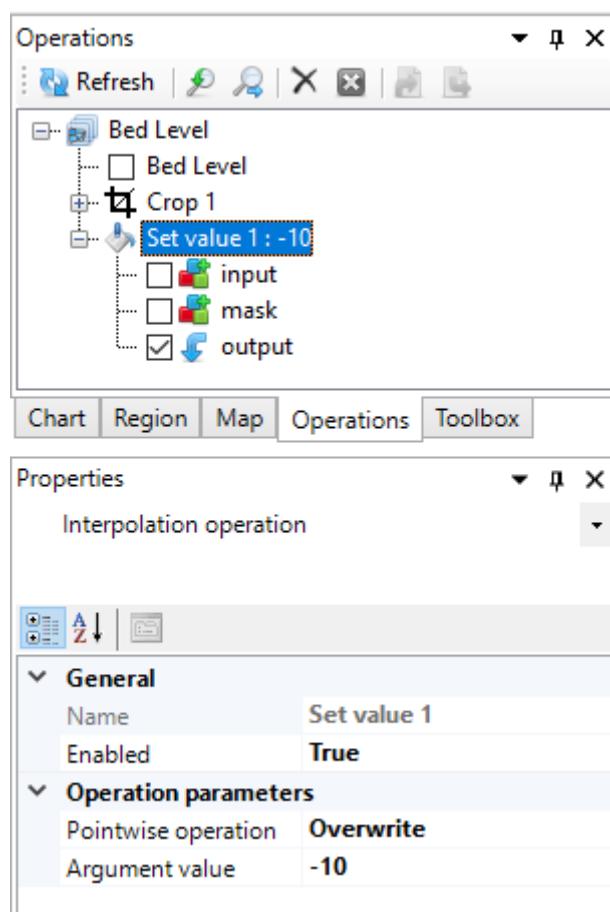


Figure H.55: Editing the value or 'Pointwise operation' of a 'Set Value' operation using the properties panel

H.6.4 Delete operations

To delete an operation permanently you have to select the operation and either press the cross icon (Figure H.57) or use the context menu and select delete (Figure H.58). The operation will be removed from the stack. The result of deleting an operation is not directly activated. This is indicated in the stack with the 'out of sync' exclamation mark. You need to refresh the stack (see section H.6.5) for the changes to become active.

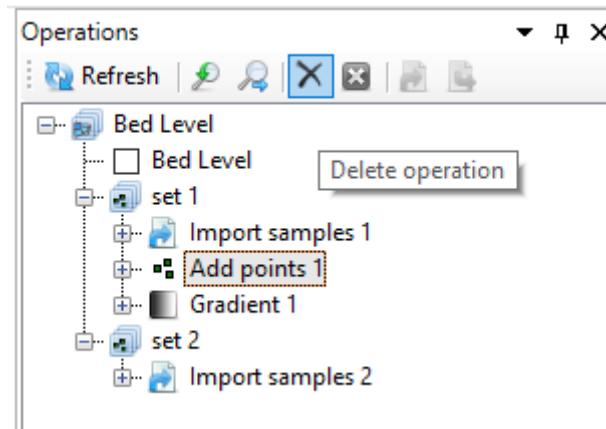


Figure H.57: Removing an operation from the stack using the cross icon in the stack menu

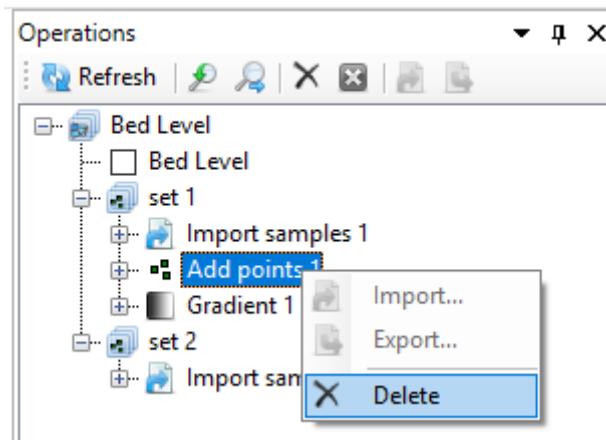


Figure H.58: Removing an operation from the stack using the context menu on the selected operation

H.6.5 Refresh stack

When the stack is marked ‘out of sync’ by exclamation marks, you can refresh the stack by pressing the ‘Refresh’ button for the changes to become active (Figure H.59). Upon refreshing the stack all the (enabled) operations in the stack will be (re-)evaluated. **Note:** Please note that refreshing the stack can take some time when large datasets are (re-)evaluated!

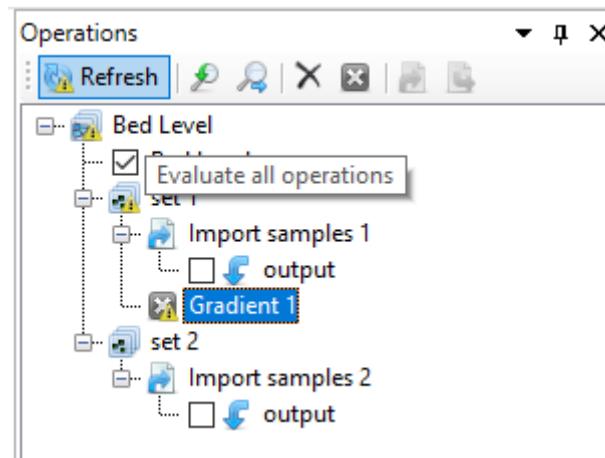


Figure H.59: Refresh the stack using the ‘Refresh’ button so that all operation are (re-)evaluated

H.6.6 Quick links

The stack menu contains two quick links to quickly show the original dataset (e.g. where you started from, Figure H.60) and the end result of the spatial operations (Figure H.61).

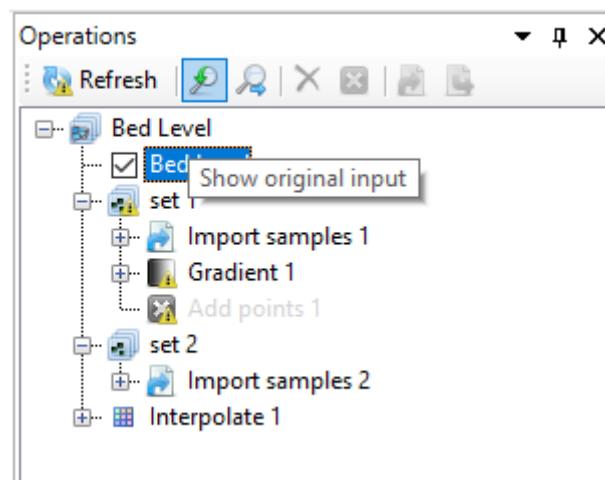


Figure H.60: Quick link to the original dataset before performing any spatial operations

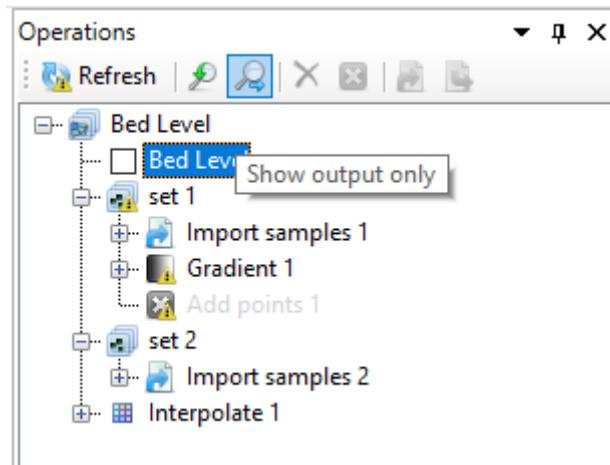


Figure H.61: Quick link to the output after performing all (enabled) operations

H.6.7 Import/export

Note: Importing and exporting data into or from the stack is still under construction



Index

weirs, 212





Deltarès systems

PO Box 177
2600 MH Delft
Boussinesqweg 1
2629 VH Delft
The Netherlands

+31 (0)88 335 81 88
sales@deltaressystems.nl
www.deltaressystems.nl