

TUFLOW FV User Manual

Flexible Mesh Modelling

2014 (Build 2014)

www.TUFLOW.com
www.TUFLOW.com/fvforum
fwwiki.TUFLOW.com
support@tuflow.com

[What is TUFLOW FV?](#)
[Installing TUFLOW FV](#)

[Table of Contents](#)

[List of Figures](#)

[List of Tables](#)

[.fvc File Commands](#)

How to Use This Manual

This manual is designed primarily for digital usage. Section, table and figure references are *hyperlinked* (click on the Section, Table or Figure number in the text to move to the relevant page) and a number of web-based links are also provided.

Similarly, text file commands are hyperlinked and accessed through relevant lists. There are also command hyperlinks in the text (normally blue and underlined). Command text can be copied and pasted into the text files to ensure correct spelling.

Some useful keys to navigate backwards and forwards are *Alt Left / Right arrow* to go backwards / forwards to the last locations. *Ctrl Home* returns to the front page, which contains useful hyperlinks. Also, *Ctrl End* provides quick access to the end pages, which contain all the hyperlinks to the text file commands.

Any constructive suggestions are very welcome (support@tuflow.com).

About This Manual

This document is a User Manual for the TUFLOWFV.exe hydrodynamic computational engine. This engine is driven through a Console (DOS) Window and relies on third party software to provide the interface to the user and the engine. This typically includes a text editor (e.g. UltraEdit, Notepad++), a mesh generator (e.g. Aquaev SMS) and spreadsheet software (e.g. Microsoft Excel). TUFLOW FV output can be viewed using Aquaev SMS. Many pre and post-processes task can be significantly enhanced through a GIS platform with 3D surface mapping (e.g. MapInfo with Vertical Mapper) and/or advanced numerical analysis software package (e.g. MATLAB). Please also refer to the user documentation or help for the third party software you have chosen to use in addition to this manual.

Setting up a TUFLOW FV model generally requires building a flexible mesh. The quality of the mesh can have a significant influence on model performance. Recognising this, the manual provides guidance for developing a flexible mesh and an example of creating a flexible mesh using our preferred mesh generator, SMS.

Contents

TABLE OF CONTENTS

	How to Use This Manual	i
	About This Manual	i
1	INTRODUCTION	1
1.1	Non-Linear Shallow Water Equations (NLSWE)	3
1.2	Scalar Conservation Equations	4
1.3	Flexible Mesh Modelling	5
1.4	Multi-core processing	6
2	OVERVIEW	7
2.1	Software Structure	7
2.1.1	TUFLOW FV Licensing	8
2.2	Data Input and Model Output	9
2.2.1	Suggested Folder Structure	9
2.2.2	File Types and Naming Conventions	10
3	THE MODELLING PROCESS	13
3.1	Problem Definition	13
3.2	Model Limits (Space and Time)	14
3.3	Base Data Preparation	15
3.3.1	Bathymetry and Topography	15
3.4	Mesh Construction	17
3.5	Boundaries	19
3.5.1	Open Boundaries	19
3.5.2	Bed Resistance	19
3.5.3	Water Surface Boundary	20
3.5.4	Wetting and Drying	20
3.5.5	Initial Conditions	20
3.6	Model Parameterisation	21
3.6.1	Turbulent Mixing	21
3.6.1.1	<i>Eddy Viscosity</i>	21
3.6.1.2	<i>Scalar Diffusivity</i>	21
3.6.2	First or Second Order	22

Contents	iii
3.6.3 2D or 3D	22
3.6.3.1 <i>Baroclinic Calculations</i>	23
3.6.4 Heat Exchange	23
3.7 Test Model Performance	25
3.8 Calibration / Validation / Sensitivity Testing	26
4 DATA INPUT	29
4.1 Simulation Control Files (.fvc)	29
4.2 Geometry Inputs	30
4.2.1 Mesh Generation	30
4.2.1.1 <i>Mesh File Format (.2dm)</i>	32
4.2.2 Cell Centred Computations	35
4.2.2.1 <i>Elevation Update Options</i>	35
4.3 Boundary Conditions	37
4.4 Structures	41
4.4.1 Bridges	43
4.4.1.1 <i>Form Loss Method</i>	43
4.4.1.2 <i>HQH Specification</i>	44
4.4.2 Culverts	46
4.4.3 Weirs	51
4.4.3.1 <i>Control Structure Options</i>	51
4.4.3.2 <i>Auto Weir</i>	52
4.4.4 Logic Controls	53
5 MODEL OUTPUT	56
5.1 2D Model Output	56
5.1.1 2D Points Output	56
5.1.2 2D SMS Data File Output	57
5.2 3D Model Output Vertically Averaged	59
5.2.1 3D Vertical Averaging Options	59
5.2.1.1 <i>3D Depth-All</i>	60
5.2.1.2 <i>3D Depth-Range</i>	61
5.2.1.3 <i>3D Height-Range</i>	62
5.2.1.4 <i>3D Elevation-Range</i>	63
5.2.1.5 <i>3D Sigma-Range</i>	64
5.2.1.6 <i>3D Layer-Range-Top</i>	65
5.2.1.7 <i>3D Layer-Range-Bot</i>	66

5.3	netCDF 2D and 3D Model Output	67
5.4	Statistical Output	71
5.5	Profile Output	71
5.6	Check Files	71
5.7	Output Types and Parameters	72
6	INSTALLING AND RUNNING TUFLOW FV	1
6.1	Installing TUFLOW FV	1
6.1.1	Codemeter Features	3
6.1.2	Requesting a Licence Change	3
6.2	Running TUFLOW FV	5
6.2.1	Right Mouse Button in Microsoft Explorer	5
6.2.2	From a Text Editor	7
6.2.3	Using a Batch File	7
6.2.4	From the SMS Interface	8
6.2.5	Change Priority, Pause, Restart or Cancel a Simulation	8
7	COMMAND FILE (FVC) REFERENCE	10
7.1	Control File Layout	10
7.2	Command Line Syntax	12
	SIMULATION CONFIGURATION COMMANDS	A-2
	TIME COMMANDS	A-1
	MODEL PARAMETER COMMANDS	A-1
	TURBULENCE PARAMETER COMMANDS	A-1
	GEOMETRY COMMANDS	A-1
	MATERIAL PROPERTIES COMMANDS	A-1
	INITIAL CONDITION COMMANDS	A-1
	BOUNDARY CONDITION COMMANDS	A-1
	STRUCTURE COMMANDS	A-1
	OUTPUT COMMANDS	A-1
	AD SIMULATION CONFIGURATION COMMANDS	B-2
	AD MODEL PARAMETER COMMANDS	B-1
	AD TURBULENCE PARAMETER COMMANDS	B-1
	AD MATERIAL PROPERTIES COMMANDS	B-4
	AD TRACER COMMANDS	B-1
	AD INITIAL CONDITION COMMANDS	B-1

List of Figures

Figure 1-1	Example decrease in runtime using TUFLOW FV multi-thread processing	6
Figure 2-1	Example TUFLOW FV Sub-Folder Structure	10
Figure 3-1	Digital Elevation Model of Port Curtis, Queensland, Australia	16
Figure 3-2	TUFLOW FV Mesh of Port Curtis Estuary, Queensland, Australia	17
Figure 3-3	3D Model Vertical Discretisation Options	23
Figure 3-4	Example Tidal Water Level Timeseries Calibration Plot	27
Figure 3-5	Example Current Speed and Current Direction Timeseries Calibration Plots for a Tidal Estuary	27
Figure 3-6	Example Flow Timeseries Calibration Plots for a Tidal Estuary	28
Figure 4-1	Example TUFLOW FV Control File Syntax	30
Figure 4-2	Mesh Development Example	31
Figure 4-3	Example 2dm File	32
Figure 4-4	Example Node Definition	33
Figure 4-5	Example Quadrilateral Element Definition	33
Figure 4-6	Example Triangular Element Definition	34
Figure 4-7	Example Nodestring Definition	34
Figure 4-8	Cell Elevation Polyline Example	36
Figure 4-9	Cell Elevation Polygon Example	37
Figure 4-10	Boundary Condition Block Examples	38
Figure 4-11	Example Structure Definition	43
Figure 4-12	2D Bridge Example (Form Loss Coefficient)	44
Figure 4-13	hQh Structure Example	45
Figure 4-14	hQh Calculation Design Options	46
Figure 4-15	1D Outlet Control Culvert Flow Regimes	48
Figure 4-16	1D Inlet Control Culvert Flow Regimes	49
Figure 4-17	1D Culvert Example	49
Figure 4-18	Weir Flow	51
Figure 4-19	Fixed Elevation Weir Example	51
Figure 4-20	Variable Elevation Weir Example	52
Figure 4-21	Auto Weir Example	53
Figure 4-22	Logic Control Examples	55
Figure 5-1	Points Output Commands and Output Points File Contents Example	57
Figure 5-2	TUFLOW FV 2D Points Output File Example	57
Figure 5-3	SMS Mapped Output Commands Example	57

Figure 5-4	TUFLOW FV Mapped Current Velocity Output in the SMS Generic Mesh Module Environment	58
Figure 5-5	3D Depth-All Output Example Commands and Conceptual Illustration	60
Figure 5-6	3D Depth-Range Output Example Commands and Conceptual Illustration	61
Figure 5-7	3D Height-Range Output Example Commands and Conceptual Illustration	62
Figure 5-8	3D Elevation-Range Output Example Commands and Conceptual Illustration	63
Figure 5-9	3D Sigma-Range Output Example Commands and Conceptual Illustration	64
Figure 5-10	3D Layer-Range-Top Output Example Commands and Conceptual Illustration	65
Figure 5-11	3D Layer-Range-Bot Output Example Commands and Conceptual Illustration	66
Figure 5-12	TUFLOW FV Sheet Plot with Zoom Example: Velocity Magnitude Top 50% Water Column (top); Velocity Magnitude Bottom 50% Water Column (bottom)	68
Figure 5-13	TUFLOW FV Salinity Vertical Distribution: Model Mesh and Curtin Polyline (top); Salinity Curtin Plotted with Polyline Chainage; Salinity Curtain Plotted with Polyline Coordinates (bottom)	69
Figure 5-14	TUFLOW FV Velocity Vertical Distribution: River Bend Flood Flow and Cross-Section Locations (top); Total Velocity Magnitude (contours) with Radial Flow Vectors Cross-Sections	70
Figure 5-15	Statistical Output Example Commands	71
Figure 5-16	Profile Output Example Commands	71
Figure 5-17	Example Output Block Commands	76
Figure 7-1	Example TUFLOW FV Simulation Control File	11

List of Tables

Table 2-1	Recommended TUFLOW FV Sub-Folder Structure Description	9
Table 2-2	TUFLOW FV File Formats	11
Table 4-1	Minimum Model Input Requirements	29
Table 4-2	Cell Elevation File Examples	35
Table 4-3	Boundary Condition Types (Basic)	38
Table 4-4	Boundary Condition Types (Advanced)	39
Table 4-5	1D Culvert Flow Regimes	47
Table 4-6	Culvert File Inputs	50
Table 5-1	Output Types	73
Table 5-2	Output Parameters (Basic)	74
Table 5-3	Output Parameters (Advanced)	74
Table 6-1	Codemeter Dongle Status	3
Table 7-1	Recommended TUFLOW FV Simulation Control File Sections	10

1 Introduction

TUFLOW FV is a numerical hydrodynamic model for the two-dimensional (2D) and three-dimensional (3D) Non-Linear Shallow Water Equations (NLSWE). The model is suitable for solving a wide range of hydrodynamic systems ranging in scale from the open channels and floodplains, through estuaries to coasts and oceans.

The Finite-Volume (FV) numerical scheme employed by TUFLOW FV is capable of solving the NLSWE on both structured rectilinear grids and unstructured meshes comprised of triangular and quadrilateral elements. The flexible mesh allows for seamless boundary fitting along complex coastlines or open channels as well as accurately and efficiently representing complex bathymetries with a minimum number of computational elements. The flexible mesh capability is particularly efficient at resolving a range of scales in a single model without requiring multiple domain nesting. The governing equations are updated using an appropriate timestep that obeys the Courant-Freidrich-Levy (CFL) constraints imposed by the flow characteristics. Further details regarding the numerical scheme employed by TUFLOW FV are provided in the TUFLOW FV Science Manual.

Unstructured mesh geometries can be created using a suitable mesh generation tool. BMT staff generally use the SMS package (<http://www.aquaveo.com/sms>) for building meshes as well as undertaking a range of model pre-processing and post-processing tasks. Both Cartesian and Spherical mesh geometries can be used as the basis for TUFLOW FV simulations.

Three-dimensional simulations can be performed within TUFLOW FV using either sigma-coordinate or a hybrid z-coordinate vertical mesh. Three-dimensional simulations can optionally use a mode-splitting approach to efficiently solve the external (free-surface) mode in 2D at a timestep constrained by the surface wave speed while the internal 3D mode is updated less frequently. TUFLOW FV provides various options to vertically average 3D output and thereby simplify post-processing tasks.

Advection-Diffusion (AD) of multiple water-borne constituents can be solved within TUFLOW FV, either coupled with a hydrodynamic simulation, or alternatively in transport mode using a pre-calculated transport file. Simple constituent decay and settling can be accommodated in the AD solutions, or alternatively more complex sediment transport algorithms can be applied through the sediment transport module.

Baroclinic pressure-gradient terms can be optionally activated to allow the hydrodynamic solution to respond to temperature, salinity and sediment induced density gradients. Atmospheric heat exchange can also be calculated given standard meteorological parameter inputs by an integrated module.

TUFLOW FV has a variety of options for simulating horizontal turbulent mixing, including the Smagorinsky scheme. Simple parametric models for vertical mixing are incorporated within TUFLOW FV and for more complicated turbulence model algorithms an interface for linking with various external turbulence models has been implemented.

Both cohesive and non-cohesive sediment transport routines can be accessed through in-built TUFLOW FV modules which handle both bed and suspended load mechanisms. Dynamic morphology updating can be optionally activated.

TUFLOW FV provides a multitude of options for specifying model boundary conditions, including:

- Various open boundary conditions
- Point source inflows
- Moving point source inflows
- Spatially and temporally varied forcing e.g. windfields, short-wave forcing

Model output files are primarily map output in SMS format (2D or vertically averaged 3D), map output in netCDF format (2D, vertically averaged 3D or full 3D) and time-series output in comma-delimited format (2D or vertically averaged 3D). The netCDF output files can be viewed using any numerical analysis package with a netCDF library interface, including MATLAB, R, GNU Octave or Python NumPy. The TUFLOW FV netCDF output file structure is described in Appendix C.

1.1 Non-Linear Shallow Water Equations (NLSWE)

TUFLOW FV solves the NLSWE, including viscous flux terms and various source terms on a flexible mesh comprised of triangular and quadrilateral elements.

The NLSWE is a system of equations describing the conservation of fluid mass/volume and momentum in an incompressible fluid, under the hydrostatic pressure and Boussinesq assumptions. The standard form of the NLSWE, which relates the time-derivative of the conserved variables to flux-gradient and source terms, is given below.

$$\frac{\partial \mathbf{U}}{\partial t} + \nabla \cdot \mathbf{F}(\mathbf{U}) = \mathbf{S}(\mathbf{U}) \quad (1)$$

The finite-volume schemes are derived from the conservative integral form of the NLSWE, which are obtained by integrating the standard conservation equation over a control volume, Ω .

$$\int_{\Omega} \frac{\partial \mathbf{U}}{\partial t} d\Omega + \int_{\Omega} \nabla \cdot \mathbf{F}(\mathbf{U}) d\Omega = \int_{\Omega} \mathbf{S}(\mathbf{U}) d\Omega \quad (2)$$

Gauss' theorem is used to convert the flux-gradient volume integral into a boundary-integral:

$$\frac{\partial}{\partial t} \int_{\Omega} \mathbf{U} d\Omega + \oint_{\partial\Omega} (\mathbf{F} \cdot \mathbf{n}) ds = \int_{\Omega} \mathbf{S}(\mathbf{U}) d\Omega \quad (3)$$

where $\int_{\Omega} d\Omega$ represent volume integrals and $\oint_{\partial\Omega} ds$ represents a boundary integral and \mathbf{n} is the boundary unit-normal vector.

The NLSWE conserved variables are volume (depth), x-momentum and y-momentum:

$$\mathbf{U} = \begin{bmatrix} h \\ hu \\ hv \end{bmatrix} \quad (4)$$

where h is depth, u is x -velocity and v is y -velocity.

The x , y and z components of the inviscid flux (\mathbf{F}^I) and viscous flux (\mathbf{F}^V) terms in the NLSWE are given below.

$$\begin{aligned} \mathbf{F}_x^I &= \begin{bmatrix} hu \\ hu^2 + \frac{1}{2}gh^2 \\ huv \end{bmatrix}, \quad \mathbf{F}_x^V \approx \begin{bmatrix} 0 \\ -hK_v \frac{\partial u}{\partial x} \\ -hK_v \frac{\partial v}{\partial x} \end{bmatrix} \\ \mathbf{F}_y^I &= \begin{bmatrix} hv \\ huv \\ hv^2 + \frac{1}{2}gh^2 \end{bmatrix}, \quad \mathbf{F}_y^V \approx \begin{bmatrix} 0 \\ -hK_v \frac{\partial u}{\partial y} \\ -hK_v \frac{\partial v}{\partial y} \end{bmatrix} \\ \mathbf{F}_z^I &= \begin{bmatrix} hw \\ hwu \\ hwv \end{bmatrix}, \quad \mathbf{F}_z^V \approx \begin{bmatrix} 0 \\ -\nu_t \frac{\partial u}{\partial z} \\ -\nu_t \frac{\partial v}{\partial z} \end{bmatrix} \end{aligned} \quad (5)$$

where K_v and ν_t are the horizontal and vertical eddy-viscosity terms.

Some of the various source terms to the NLSWE are provided below:

$$\mathbf{S} = \begin{bmatrix} 0 \\ gh \frac{\partial z_b}{\partial x} + fvh - \frac{h}{\rho_0} \frac{\partial p_a}{\partial x} - \frac{hg}{\rho_0} \int_z^\eta \frac{\partial \rho}{\partial x} dz - \frac{1}{\rho_0} \left(\frac{\partial s_{xx}}{\partial x} + \frac{\partial s_{xy}}{\partial y} \right) + \frac{\tau_{sx}}{\rho_0} - \frac{\tau_{bx}}{\rho_0} \\ gh \frac{\partial z_b}{\partial y} - fuh - \frac{h}{\rho_0} \frac{\partial p_a}{\partial y} - \frac{hg}{\rho_0} \int_z^\eta \frac{\partial \rho}{\partial y} dz - \frac{1}{\rho_0} \left(\frac{\partial s_{yx}}{\partial x} + \frac{\partial s_{yy}}{\partial y} \right) + \frac{\tau_{sy}}{\rho_0} - \frac{\tau_{by}}{\rho_0} \end{bmatrix} \quad (6)$$

where,

- $\frac{\partial z_b}{\partial x}, \frac{\partial z_b}{\partial y}$ are the x- and y-components of bed slope;
- f is the coriolis coefficient;
- ρ is the local fluid density, ρ_0 is the reference density and p_a is the mean sea level pressure;
- s_{ij} is the short-wave radiation stress tensor; and
- τ_s and τ_b are respectively the surface and bottom shear stress terms (where applicable).

Other source terms not included above include inflow/outflow to/from the water column.

1.2 Scalar Conservation Equations

Analogous conservation equations are solved for the transport of scalar constituents in the water column.

$$U = [hC] \quad (7)$$

where C is the constituent concentration. The flux components of the scalar conservation equation are:

$$\begin{aligned} F_x^I &= [huC], \quad F_x^V \approx \left[-h \left(D_{xx} \frac{\partial C}{\partial x} + D_{xy} \frac{\partial C}{\partial y} \right) \right] \\ F_y^I &= [hvC], \quad F_y^V \approx \left[-h \left(D_{yx} \frac{\partial C}{\partial x} + D_{yy} \frac{\partial C}{\partial y} \right) \right] \\ F_z^I &= [hwC], \quad F_z^V \approx \left[-hv'_t \frac{\partial C}{\partial z} \right] \end{aligned} \quad (8)$$

The source components may include scalar decay and settling:

$$S = [-K_d hC - w_s C] \quad (9)$$

where K_d is a scalar decay-rate coefficient and w_s is a scalar settling velocity.

1.3 Flexible Mesh Modelling

TUFLOW FV is capable of solving the NLSWE on unstructured geometries and is commonly referred to as a ***flexible mesh model***. Compared to structured rectilinear grids (i.e. fixed grids) the design of the flexible mesh tends to have a greater influence on model performance. Therefore, more time and effort should be spent preparing the model mesh geometry. Over the life cycle of a modelling project, a well assembled mesh will save time (both the modellers and the computers).

The flexible mesh consists of a network of irregular triangular and quadrilateral elements. This has inherent advantages, including:

- Mesh resolution can be adjusted according to the needs of the study (i.e. fine resolution in the area of interest, coarser resolution in the regional extents). Therefore, a range of spatial scales can be modelled without resorting to nesting.
- Mesh alignment can neatly fit bathymetric contours and boundary extents, optimising mesh resolution. This is particularly relevant in regions with complex bathymetric features.
- Specific features, such as infrastructure or other man-made developments, can be included in the model mesh more accurately.

To exploit these advantages, the mesh needs to be designed carefully and appropriately for the specific model application. There are a number of mesh generators available to construct a model mesh; however BMT uses the SMS package, provided by Aquaveo (see www.aquaveo.com/sms). Mesh construction is discussed further in Section 3.4.

1.4 Multi-core processing

TUFLOW FV is parallelised for multi-processor machines using the OpenMP implementation of shared memory parallelism. This means that a TUFLOW FV model simulation will run faster if there is more than one processor (or thread) on a single computer. The increase in computational speed (or decrease in runtime) is not quite linear with the number of threads, as demonstrated in Figure 1-1.

Unless the user decides otherwise, TUFLOW FV will run using the maximum number of threads available to it, only limited by the software licence or computer hardware.

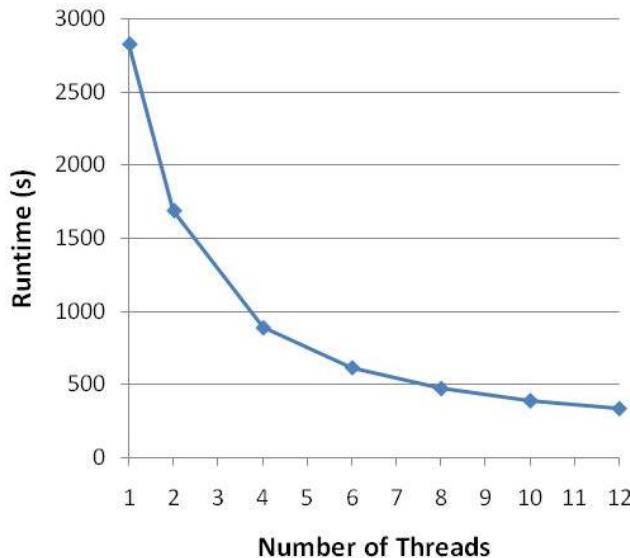


Figure 1-1 Example decrease in runtime using TUFLOW FV multi-thread processing

2 Overview

2.1 Software Structure

TUFLOW FV is the computational engine for carrying out 2D or 3D hydrodynamic calculations. TUFLOW FV does not have its own Graphical User Interface and utilises other third-party software for the creation, manipulation and viewing of data. As a minimum, a TUFLOW FV user requires access to:

- A text editor to create and edit TUFLOW FV control files (UltraEdit and Notepad++ are popular, although Notepad does suffice).
- SMS (Surface Modelling System - www.aquaveo.com/sms) for mesh generation, viewing 2D or vertically averaged 3D output (.dat format) and creating animations. The TUFLOW FV wiki contains some useful information to assist using SMS: http://fvwiki.tuflow.com/index.php?title=SMS_Tips. Other mesh generation tools could be used to create a TUFLOW FV mesh but have not been tested by BMT.
- Spreadsheet software such as Microsoft Excel for preparing boundary condition files and viewing/plotting model output.

Advanced TUFLOW FV users are likely to utilise other software packages, including:

- 3D surface modelling software (e.g. Vertical Mapper) for the creation and interrogation of a Digital Elevation Model (DEM).
- Geographic Information Systems (GIS) such as MapInfo or ArcGIS that provide powerful environments for developing model components and presenting model output.
- MATLAB is used by BMT for preparing input data, viewing 2D and 3D netCDF output data and creating animations. Other numerical analysis packages with a netCDF library interface such as R, GNU Octave or Python NumPy would be equally useful for these purposes but have not been extensively tested by BMT.

The TUFLOW FV executable (TUFLOWFV.exe) is a command console program. A model is started by calling the executable with the simulation control file (.fvc) as the first and only argument. If no argument is specified the command line will request the user input one.

Should a complete GUI that allows the user to create, manage and view models and model output within the one interface be desired, an interface for TUFLOW FV within SMS has been developed. At present the interface does not allow access to all the features of TUFLOW FV, however, will be expanded in the future.

Some common ways to call the TUFLOW FV executable and start a simulation are described in Section 6.2.

2.1.1 TUFLOW FV Licensing

Performing TUFLOW FV simulations will require the presence of a suitably licensed hardware lock. TUFLOW FV supports both local license and network license versions of the WIBU codemeter system dongles. A TUFLOW FV dongle will have one or more engine licenses and typically twice as many thread licenses as engines. For instance, a 4 license hardware lock would permit 4 simultaneous simulations utilising 2 threads each, or it would permit 1 simulation utilising 8 threads.

In addition to the basic TUFLOW FV engine license, various optional modules can be licensed via the WIBU Codemeter dongles. The number of module licenses can be less than or equal to the number of engine licenses available on a dongle.

Network dongles are also available, which then licences TUFLOW FV simulations across an office network.

A step-by-step guide to installing a WIBU Codemeter is provided in Section 6.1. The TUFLOW FV Wiki lists the steps required to request a new or upgrade an existing TUFLOW FV licence:

http://fwwiki.tuflow.com/index.php?title=Requesting_a_Licence

2.2 Data Input and Model Output

2.2.1 Suggested Folder Structure

Table 2-1 presents the recommended set of sub-folders to be set up for a TUFLOW FV model. Any folder structure may be used; however, it is strongly recommended that a system similar to that below be adopted.

For large modelling jobs with many scenarios and simulations, a more complex folder structure may be warranted, but should be based on that below. Other sub-folders can of course be added by the modeller. For example, a “matlab” sub-folder to store project related pre and post-processing scripts may be desired.

Table 2-1 Recommended TUFLOW FV Sub-Folder Structure Description

Sub-Folder	Description
Locate folders below on the system network under a folder named “tuflow_fv” in the project folder (e.g. J:\Project12345\tuflowfv)	
These folders should be backed up regularly	
bc	Boundary conditions, often with additional sub-folders for specific boundary condition types (e.g. tide, flow, meteorology, etc.)
exe	Optional sub-folder, placing the tuflowfv.exe (and associated dlls) within the TUFLOW FV folder structure may be desired. Alternatively, the tuflowfv.exe is located elsewhere on the network or local computer.
geo	Model geometry, often with additional sub-folders or links to locations where mesh development data is located (e.g. mesh generation files, DEMs, aerial photos, nautical charts, etc.).
input	TUFLOW FV simulation control files. Batch files are also stored here when performing multiple simulations in a series.
input\log	Location for automatically generated simulation log and model performance files.
output	The directory where specified model output is written. Often placed on a local drive rather than a network drive.
results	Post-processed model output, including model calibration/validation and design simulation results.

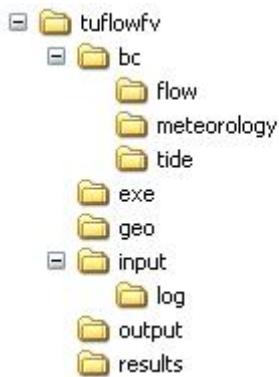


Figure 2-1 Example TUFLOW FV Sub-Folder Structure

2.2.2 File Types and Naming Conventions

Files are generally classified as:

- Control Files
- Data Input Files
- Model Output Files
- Check Files

Control files are used for directing inputs to the simulation and setting parameters. The style of input is very simple, free form commands, similar to writing down a series of instructions. This offers the most flexible and efficient system for experienced modellers. It is also easy for inexperienced users to learn.

Data input files are primarily comma-delimited files prepared using spreadsheet software. Simulations that require spatially and temporally varied forcing (e.g. windfields or short-wave forcing) typically rely on netCDF format data input. Some common examples of netCDF input file structures are provided in Appendix D. In some instances, the model initial condition may be defined by map output from a previous simulation, referred to as a TUFLOW FV restart file.

Data output files are primarily map output in SMS format (2D or vertically averaged 3D), map output in netCDF format (2D, vertically averaged 3D or full 3D) and time-series output in comma-delimited format. The TUFLOW FV netCDF output file structure is summarised in Appendix C.

In addition to the above, a range of check files are produced in text and comma-delimited formats to carry out quality control and model efficiency checks.

The most common file types and their extensions are listed in Table 2-2.

Table 2-2 TUFLOW FV File Formats

File	Extension	Description	Format
Control Files (see Section 4.1)			
TUFLOW FV Simulation Control File	.fvc	Controls the data input and output for a 2D or 3D simulation. Mandatory for 2D and 3D.	Text
Data Input (see Sections 4.2, 4.3 and 4.4)			
Mesh Geometry File	.2dm	A file containing the 2D geometry of the model mesh and elevations. It also contains information on the material types used to define areas with a specified bed roughness and the location of open boundaries. The structure of the .2dm file follows the SMS Generic Mesh Module structure. Mandatory for 2D and 3D.	Text
3D Model Vertical Mesh File	.csv	A file containing the z-coordinates of the vertical mesh. Mandatory for 3D using z-coordinate or z-sigma coordinate discretisation.	Text
Comma Delimited Files	.csv	These files are used for temporally varying boundary condition input, such as a tidally varying water level or inflow condition. They can be opened and saved using a text editor or spreadsheet software such as Microsoft Excel.	Text
netCDF File	.nc	These files are typically used to store data inputs that vary spatially and temporally. These inputs are often derived from outputs from other models and may include windfields, atmospheric conditions, short-wave forcing or ocean current forcing.	netCDF
Restart File	.rst	These files are generated by TUFLOW FV and contain the spatially varying conserved variables at an instant in time. Restart files are optionally used to define the initial condition of a TUFLOW FV simulation.	Binary
Model Output (see Section 5)			
Comma Delimited Files	.csv	These files are used for time-series data output (2D and vertically averaged 3D). They are typically opened and viewed using numerical analysis software (e.g. spreadsheet software such as Microsoft Excel).	Text
SMS Data File	.dat	SMS generic formatted simulation output file. TUFLOW FV map output can be written in the SMS .dat format (2D and vertically averaged 3D).	Binary

File	Extension	Description	Format
netCDF File	.nc	netCDF formatted simulation output file. TUFLOW FV map output can be written in the netCDF .nc format (2D, vertically averaged 3D and full 3D).	netCDF
Restart File	.rst	Spatially varying conserved variables at an instant in time for restarting TUFLOW FV simulations.	Binary
Check Files (see Section 5.6)			
Log File	.log	A file containing information about the model inputs and a log of the simulation. Automatically generated.	Text
Geometry File	.nc	A netCDF file containing the 2D or 3D model mesh geometry information. Automatically generated.	netCDF
Simulation Timestep Files	.csv	The minimum and mean timestep required for calculation of the free surface (external) gravity wave terms in each model cell is contained in the file ***_ext_cfl_dt.csv The minimum and mean timestep required for calculation of the advective (internal) terms in each model cell is contained in the file ***_int_cfl_dt.csv These files are generated automatically and can be used to identify the model cell(s) constraining the simulation timestep.	Text
Mass Output File	.csv	Optionally specified timeseries output used to check the volume of fluid within the model domain.	Text
Flux Output File	.csv	Optionally specified timeseries output used to check the rates of fluid entering/exiting the model boundaries or crossing specified nodestrings within the model domain.	Text

3 The Modelling Process

3.1 Problem Definition

Define the problem(s) that the numerical modelling exercise will seek to solve and explain.

Defining a modelling exercise often starts with a preferred, highly rigorous and scientifically thorough approach that strives to replicate the physical system as accurately as possible. This is followed by a series of compromises and simplifications due to practical constraints. The final problem definition strikes a balance, providing a fit-for-purpose outcome. Key considerations include:

- **What is the model expected to deliver?**
 - The purpose of the modelling exercise should be clearly defined.
- **What are the key physical processes?**
 - A clear understanding of what processes need to be investigated will inform the type of model, what parameters and modules will be used, the extents and degree of accuracies required and, importantly, whether modelling is required at all!
 - An understanding of scale is important in this regard:
 - time scales (hours, months, years, decades, etc.)
 - spatial scales (global, regional, local, sub-grid, etc.)
- **What data is available?**
 - Successful application of a specific modelling approach can only be achieved if suitable data is available.
- **What are the time, economic and logistic constraints?**
 - Sophisticated and rigorous modelling studies can take up significant time and resources. Timing, economic and/or logistical constraints can limit the modelling exercise.
 - Computer power is a common constraint that can limit the temporal and spatial extent, resolution and accuracy of a modelling exercise.

3.2 Model Limits (Space and Time)

Define a model domain that best fits the key physical processes to be represented and achieves the required spatial and temporal scales within the constraints of available computational power.

The computational effort required to run a model simulation is a function of:

- The spatial extent of the model domain (i.e. the area to be modelled). This is typically guided by:
 - the spatial extent of the problem to be solved
 - the availability and locality of data with which to define boundary conditions
 - the spatial extent of the key physical processes to be represented
- The specified start and end time of the simulation which is typically guided by the temporal extent of the key physical processes to be represented. Examples include:
 - a flood assessment requires simulation of individual flood events of hours duration
 - a coastal or estuarine assessment, where tidal forces dominate, may require the simulation of several tidal cycles over weeks, months or years
 - a morphological assessment may require simulation periods of years or decades
- The model mesh geometry and the number of active, wet elements (or cells) in the model domain. For coastal, estuarine and flood assessments the number of wet elements may vary with time.
- The timestep, which varies throughout a simulation and is selected by taking into account physical and numerical convergence and stability considerations. The appropriate timestep is calculated by TUFLOW FV such that CFL constraints imposed by the flow characteristics are obeyed.
- The complexity of the processes being simulated. A simulation that includes scalar transport calculations will require additional computational effort compared to a hydrodynamics only simulation.

3.3 Base Data Preparation

Consolidate and prepare base data. This typically includes bathymetry / topography and also boundary condition information.

The base data required to develop a TUFLOW FV model will typically comprise of:

- Spatial datasets that define elevations (bathymetric and topographic) throughout the model domain.
- Timeseries datasets that define the open boundary conditions, such as a temporally varying water level (tidal) or inflow condition.

This information is normally easy to prepare, especially with pre-processing tools such as spreadsheets, SMS, GIS/3D surface modelling software and other numerical analysis packages (e.g. MATLAB). Quality checking of input data is a crucial component of any modelling exercise (yes, the often quoted “garbage in, garbage out” phrase cannot be left out of any modelling manual).

3.3.1 Bathymetry and Topography

A good description of bathymetry (elevations below the water surface in open channels, rivers, seas or oceans) is crucial for all hydrodynamic modelling exercises. For overland flow assessments or for locations with a significant intertidal area (such as an estuary), a description of the topography (elevations above the water surface) is also required.

Bathymetric data is typically obtained via hydrographic surveys and/or nautical charts. These sources of data are generally restricted to areas of ship movements and recreational boating. In some instances a hydrographic survey specific to the project may be available. In the absence of reliable hydrographic survey or nautical chart information, bathymetry estimated from satellite data may be available.

For flooding or coastal inundation a description of the land topography is also required. This information is typically obtained via satellite radar or plane-mounted Laser Detection and Ranging (LIDAR or LADS) instruments.

In most modelling exercises an early step will be to develop a Digital Elevation Model (DEM) of the study area using the available sources of bathymetry/topography data and GIS/3D surface mapping software. A DEM is a regular structured grid of elevation values. An example DEM constructed using MapInfo and Vertical Mapper software from a combination of hydrographic survey, LIDAR and digitised nautical chart data sources is shown in Figure 3-1.

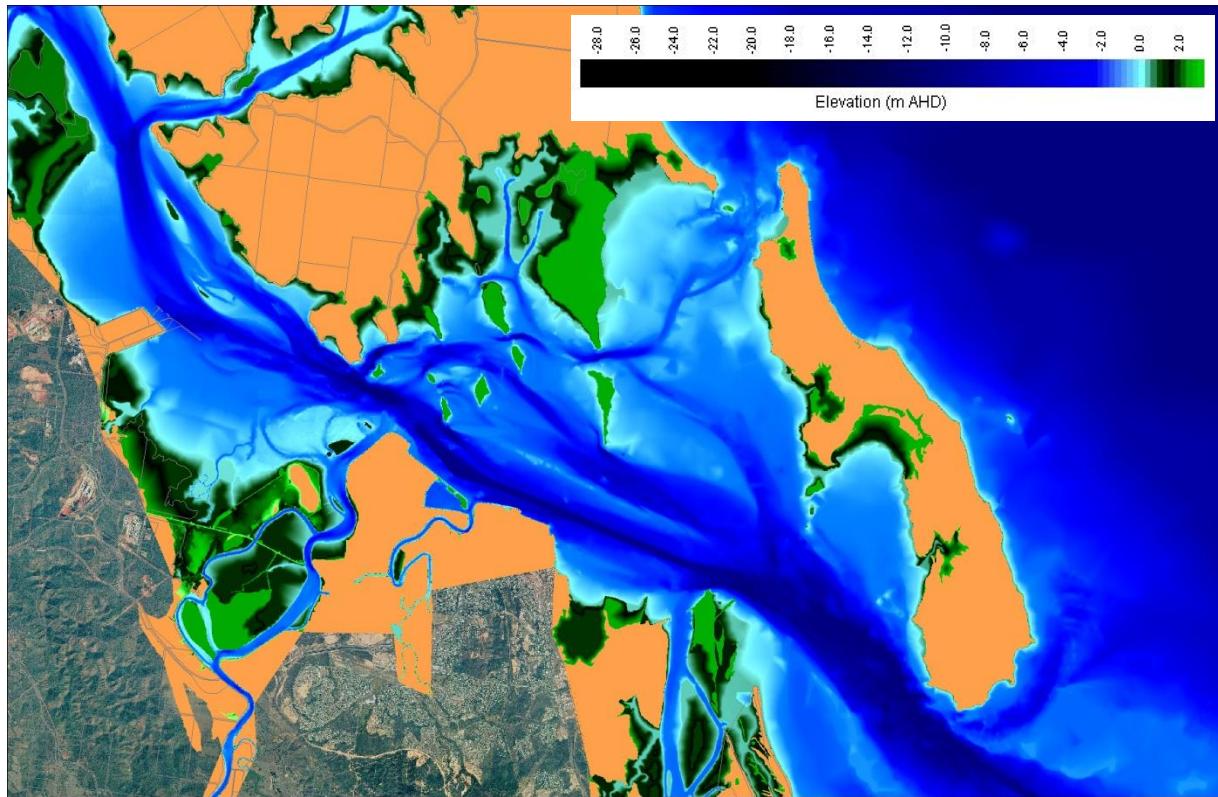


Figure 3-1 Digital Elevation Model of Port Curtis, Queensland, Australia

Not all TUFLOW FV modelling exercises will require a DEM to be developed. Pre-processing tools such as Aquaveo SMS allow elevation values from scattered datasets to be interpolated to a model mesh. Information about the SMS Scatter Module can be found on the Aquaveo XMS Wiki:

http://www.xmswiki.com/xms/SMS:Scatter_Module

3.4 Mesh Construction

Using mesh generation software, create a model mesh. Design a mesh that takes full advantage of the flexible mesh approach and also avoids pitfalls and disadvantages (Section 4.2.1 provides a summary of some mesh generation tips).

TUFLOW FV solves the NLSWE on regular structured grids or unstructured meshes. Most TUFLOW FV users take advantage of the flexible mesh capability, with the model mesh comprising of triangular and quadrilateral elements. The flexible mesh approach allows for seamless boundary fitting along complex coastlines or open channels as well as accurately and efficiently representing complex bathymetries with a minimum number of computational elements. The flexible mesh capability is particularly efficient at resolving a range of resolutions within a single model without requiring multiple domain nesting.

Figure 3-2 shows a TUFLOW FV mesh and DEM of Port Curtis (the DEM without the mesh is shown in Figure 3-1). This mesh was primarily developed to assess the impacts of a proposed shipping navigation channel expansion. Consequently, the mesh was constructed to neatly resolve the existing and proposed shipping channel geometry. Smaller mesh elements (higher mesh resolution) were necessary to resolve the complex tidal flows in the vicinity of the smaller islands and the harbour constriction. Larger mesh elements (lower mesh resolution) were used in regions located away from the areas of interest and/or where the flow varied more gradually, such as the shallow mud flats represented by the dark green areas in Figure 3-2.

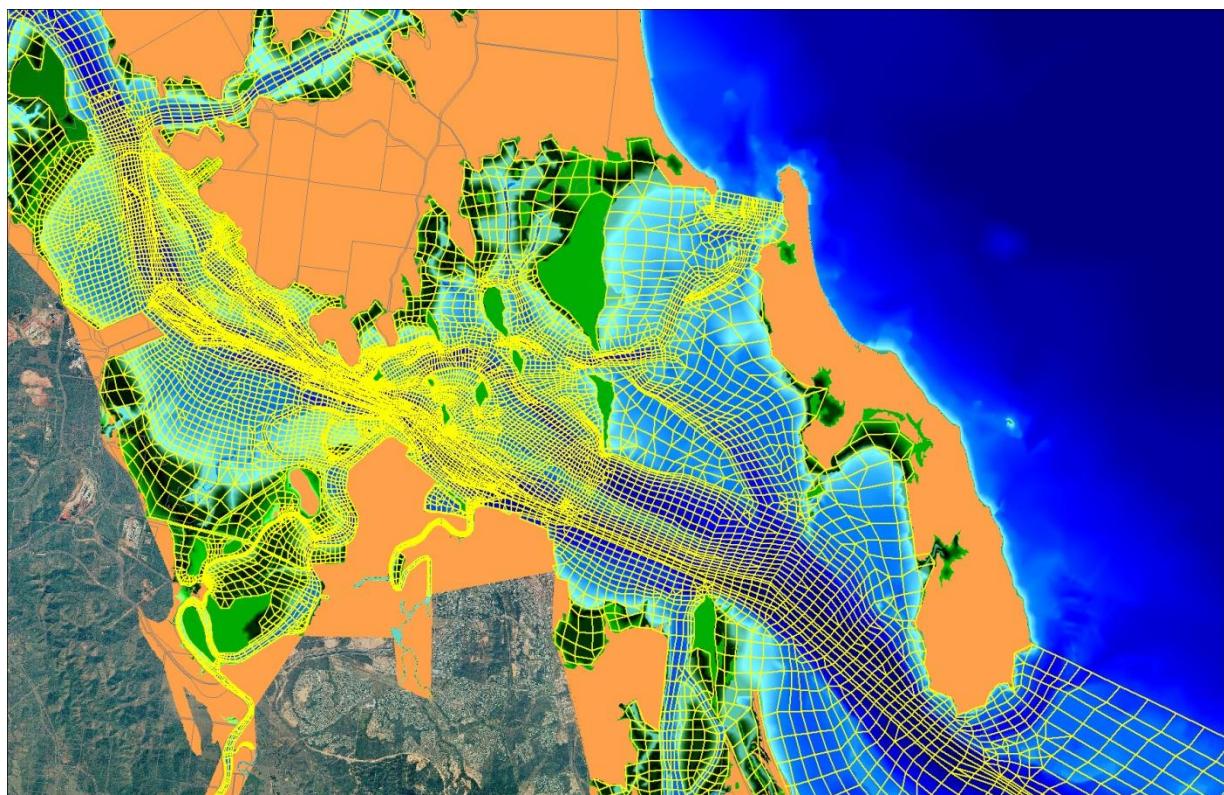


Figure 3-2 TUFLOW FV Mesh of Port Curtis Estuary, Queensland, Australia

Unstructured or flexible mesh geometries can be created using any suitable mesh generation tool. BMT staff generally use the Aquaveo SMS Generic Mesh Module for building meshes as well as undertaking a range of model pre-processing and post-processing tasks. Both Cartesian and Spherical mesh geometries can be used as the basis for TUFLOW FV simulations. Mesh building/editing tutorials are available from the following sources:

- Included with a SMS installation
- Via the Aquaveo SMS website: <http://www.aquaveo.com/software/sms-learning-tutorials>
- Via the Tutorial models on the TUFLOW FV Wiki: <http://fwwiki.tuflow.com>

In addition to these web-based resources, Section 4.2.1 outlines a series of mesh generation tips. This section is particularly useful for new flexible mesh modellers. Section 4.2.1.1 describes the contents and required format of a TUFLOW FV mesh geometry file which follows the SMS Generic Mesh Module format. Mesh geometry files generated using an alternative mesh generation tool need to follow the format described in Section 4.2.1.1. This may require manual manipulation of the mesh geometry file contents.

3.5 Boundaries

The effects at the boundaries of a TUFLOW FV model determine the resulting fluid motion and hydrodynamic prediction. Understanding what is happening at the edges of the model domain is therefore critical. There are different types of boundaries to be considered when developing a TUFLOW FV model:

- The open boundaries at the “wet” edges of the model domain
- The closed boundaries at the seabed, open channel bed and water surface
- The boundary at the coastline, river bank or other wet/dry interface
- The initial condition at the start of the simulation

3.5.1 Open Boundaries

Open boundaries to the TUFLOW FV model domain should be located where there is some knowledge of the behaviour at that location. For a given period, this information may come from a tide station or other instrument deployed to continuously measure the variation in water level, a gauging station that provides a river discharge measurement, or may be output from larger-scale model. Some coastal models require additional forcing from ocean circulation models (e.g. HYCOM, <http://hycom.org/>) to accurately resolve density-driven flows.

Descriptions of the various open boundary conditions, their commands and associated inputs are provided in Section 4.3. BMT can be contacted via TUFLOW support for further information on applying boundary conditions derived from ocean circulation models: support@tuflow.com.

3.5.2 Bed Resistance

For hydrodynamic simulations the bed boundary resistance is described using a bottom drag model. The default model is that attributed to Manning, in which case a Manning’s “n” coefficient should be specified. An alternative bottom drag model assumes a log-law velocity profile and requires specification of a surface roughness length-scale, “ks”. A single bed surface roughness can be set globally or the modeller can assign different roughness values to particular mesh cells within the model domain. The so-called “material type” definitions are stored in the TUFLOW FV mesh geometry file.

BMT typically uses the Aquaveo SMS Generic Mesh Module to create materials data definitions and assigning the material types that apply to each mesh cell. The material type for an area defined by a map polygon can be assigned using the 2D Mesh Polygon Properties dialog (prior to generating the mesh from the feature objects). Alternatively, the mesh types can be assigned on a cell-by-cell basis after the mesh has been generated.

3.5.3 Water Surface Boundary

Boundary conditions can be applied to the water surface. These typically include wind, ambient pressure and/or wave fields. In many locations, or for particular events (such as a storm), these forcing mechanisms can have a significant influence on local hydrodynamics or scalar transport and may extend through the water column. Wind, atmospheric and wave boundary conditions are typically defined by measurements and/or output from other models. These conditions may be applied globally (i.e. constant throughout the model domain) or allowed to vary spatially for a given timestep.

Simulations that require spatially and temporally varied forcing typically rely on input data arranged on regular structured grids and stored in netCDF format. BMT often utilises SWAN wave model output or hindcast meteorological data from global models (e.g. NCEP/NCAR <http://www.ncep.noaa.gov/>) as inputs to TUFLOW FV simulations.

Spatially and temporally varying data accessed from online sources or generated using other models can be very large datasets (up to gigabits in file size) and generally require some degree of processing prior to being used as input to a TUFLOW FV simulation. MATLAB is typically used by BMT for preparing input data; however, other numerical analysis software packages with GRIB (a binary format commonly used to store meteorological data) and netCDF libraries could also be used. Examples of common TUFLOW FV netCDF input files and the associated commands are provided in Appendix D.

3.5.4 Wetting and Drying

TUFLOW FV simulates the wetting and drying of areas within the model domain, such as that observed on a gently sloping beach over a tidal cycle or over land during a flood, storm surge or tsunami event. Dry/wet depths defined by the user will often depend on the scale of the simulation. For full-scale or “real world” simulations, dry/wet depths are typically in the order of centimetres. For some laboratory-scale simulations, for example a dam break or wave run-up experiment, the user defined wet/dry depths may be in the order of millimetres.

In terms of the TUFLOW FV computations, the drying value corresponds to a minimum depth below which the cell is dropped from computations (subject to the status of surrounding cells). The wet value corresponds to a minimum depth below which cell momentum is set to zero, in order to avoid unphysical velocities at very low depths.

3.5.5 Initial Conditions

All TUFLOW FV simulations start with an “initial condition”. Many hydrodynamic simulations will start with quiescent (still water) conditions and simply “warm-up” based on the boundary condition forcing. Under this scenario, the warm-up period should be long enough to allow any transients generated at the start of the simulation to propagate out of the model. Alternatively, the simulation initial condition can be defined manually by the modeller (and read from a .csv file).

In some situations the modeller may wish to set the initial condition using a TUFLOW FV restart file which contains the spatially varying conserved variables (at an instant in time) generated by a previous simulation.

3.6 Model Parameterisation

Define what processes and parameter values are to be assigned to the model, ensuring that their values are within scientifically justifiable ranges.

3.6.1 Turbulent Mixing

TUFLOW FV has a variety of options for simulating horizontal and vertical viscous fluxes. The horizontal eddy-viscosity can be specified directly or can be calculated using the Smagorinsky scheme. Simple parametric models for vertical mixing are also incorporated within the TUFLOW FV engine.

TUFLOW FV allows the horizontal scalar diffusivity to be specified as a constant value or be calculated from a Smagorinsky or Elder model. The vertical scalar diffusivity may also be directly specified or calculated using parametric formulations which vary depending on the scalar type.

A key step in the Finite-Volume numerical scheme is the calculation of numerical fluxes across cell boundaries. Advanced users may wish to access the TUFLOW FV Science Manual for further detail regarding these calculations and the numerical scheme employed by TUFLOW FV.

For more complicated turbulence model algorithms an interface for linking with various external turbulence models has been implemented (e.g. GOTM, <http://www.gotm.net/>). BMT can be contacted via TUFLOW support for further information on coupling TUFLOW FV with external turbulence models: support@tuflow.com.

The horizontal and vertical-mixing options are set in the TUFLOW FV Simulation Control File. These commands are described in Appendix A.

3.6.1.1 Eddy Viscosity

The horizontal-mixing eddy-viscosity can be defined as a constant value or can be calculated using the Smagorinsky model. The Smagorinsky model sets the diffusivity proportional to the local strain rate.

The vertical-mixing eddy-viscosity can be defined as a constant value or can be calculated using a parametric model. The parametric model is based on a parabolic eddy-viscosity profile and applies the Munk & Anderson limiters in the case of stable stratification.

Upper and lower bound values can be specified for the horizontal and vertical eddy-viscosities.

3.6.1.2 Scalar Diffusivity

The horizontal-mixing scalar diffusivity can be defined as a constant value or can be calculated using the Smagorinsky or Elder models. The Elder model calculates an isotropic diffusivity tensor with principal axes aligned with the flow direction and which scales on the local friction velocity. The Elder model allows the user to specify higher mixing in the longitudinal flow direction than transverse to the flow.

The vertical-mixing scalar diffusivity can be defined as a constant value or can be calculated using a parametric model. The parametric model is based on a parabolic eddy-viscosity profile and applies the Munk & Anderson limiters in the case of stable stratification.

Upper and lower bound values can be specified for the horizontal and vertical scalar diffusivities.

3.6.2 First or Second Order

Higher order spatial schemes will produce more accurate results in the vicinity of sharp gradients due to reduced numerical diffusion; however they will be more prone to developing instabilities and are more computationally expensive. The first-order schemes assume a piecewise constant value of the modelled variables in each cell, whereas the second-order schemes perform a linear reconstruction.

As a general rule of thumb, initial model development should be undertaken using low-order schemes, with higher-order spatial schemes tested during the latter stages of development. If a significant difference is observed between low-order and high-order results then the high-order solution is probably necessary, or alternatively further mesh refinement is required.

Second order spatial accuracy will typically be required in the vertical direction when trying to resolve sharp stratification.

3.6.3 2D or 3D

Most TUFLOW FV modelling exercises will commence in 2D. Once the 2D model has been optimised and output verified the modeller may then choose to perform a 3D simulation. Typically, a 3D simulation would be undertaken when the modeller believes the 2D depth-average assumption does not sufficiently describe the observed flow characteristics. In many cases, flow regimes are well mixed vertically (i.e. essentially 2D or “vertically uniform”) and a 3D simulation may not improve the predictive skill of the model or be required to meet the objectives of the assessment.

Three-dimensional simulations can be performed within TUFLOW FV using either sigma-coordinate, z-coordinate or hybrid z-sigma-coordinate vertical mesh. Three-dimensional simulations can optionally use a mode-splitting approach to efficiently solve the external (free-surface) mode in 2D at a timestep constrained by the surface wave speed while the internal 3D mode is updated less frequently.

Switching a TUFLOW FV model from 2D to 3D is relatively simple and requires only a few additional commands in TUFLOW FV Simulation Control File. These commands are described in Appendix A. Handling and viewing 3D map output is generally more challenging than 2D map output. BMT utilises numerical analysis software packages such as MATLAB for most 3D output processing tasks.

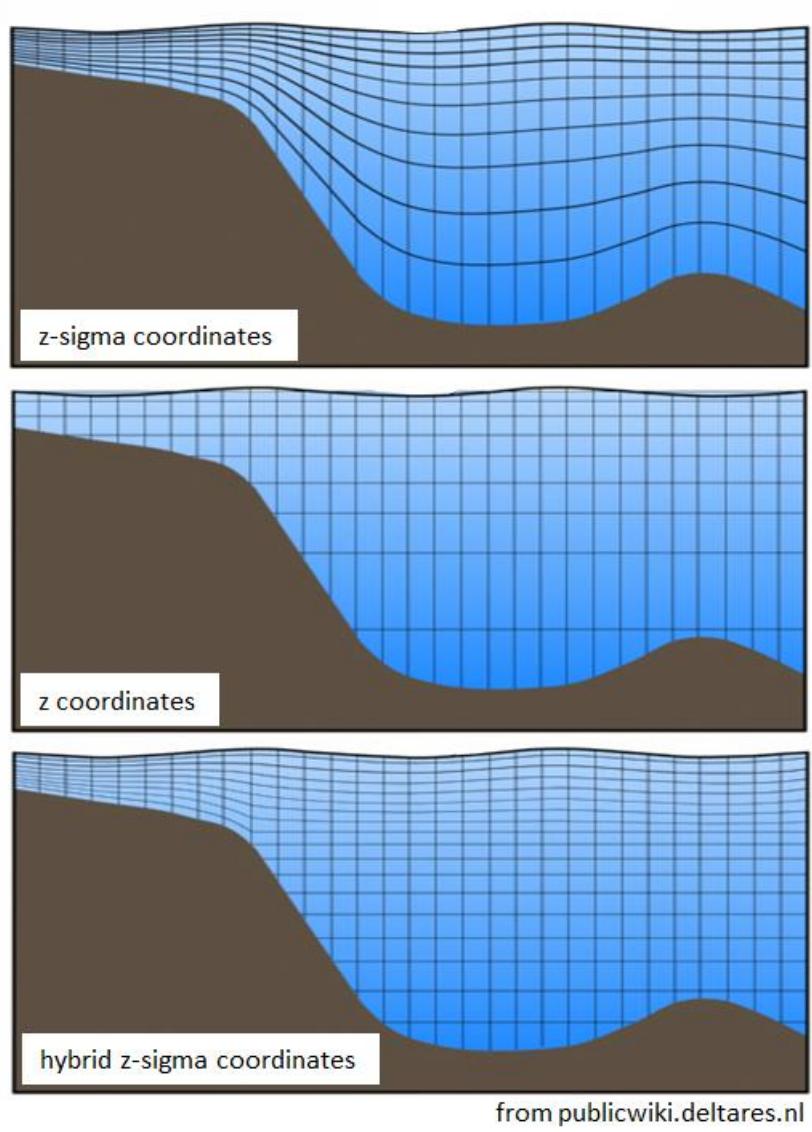


Figure 3-3 3D Model Vertical Discretisation Options

3.6.3.1 Baroclinic Calculations

For 3D simulations, the baroclinic pressure-gradient terms can be optionally activated to allow the hydrodynamic solution to respond to temperature, salinity and sediment induced density gradients. The influence of these terms may be significant in estuarine environments, for example, where fluvial and marine waters meet and stratification is known to occur.

TUFLOW FV employs the UNESCO equation of state for calculating the density of water in baroclinic simulations. Alternatively, the salinity tracer can be used as a direct proxy for density.

3.6.4 Heat Exchange

The transfer of mass, heat and momentum between the water column and the atmosphere can also be optionally calculated by TUFLOW FV for given standard meteorological parameter inputs. These inputs would typically come from recorded data or global atmospheric model output (e.g. NCEP/NCAR <http://www.ncep.noaa.gov/>) and may include surface:

- Air temperature
- Shortwave (visible light) and longwave (infrared light) radiation
- Relative humidity
- Cloud cover

Heat exchange module options are set in the TUFLOW FV Simulation Control File with the default settings assuming:

- The specific humidity as a function of vapour pressure calculated by the Magnus-Tetens formula;
- Incident short wave radiation estimated according to Jacquet (1983); and
- Incident long wave radiation albedo and water surface reflection calculated following TVA (1972). Long wave radiation emitted by the water surface is calculated assuming the Stefan-Boltzmann law.

The heat exchange module commands and required inputs are described in Appendix B. A full description of the heat exchange module is provided in the TUFLOW FV Science Manual.

3.7 Test Model Performance

Once the required input files have been prepared, model performance should be tested:

The range of model performance tests undertaken by the modeller will depend on the type of TUFLOW FV application; nevertheless there are a number of key checks to be completed prior to an initial simulation with a TUFLOW FV model:

- Once a mesh has been generated, the modeller needs to check for strange element shapes or sizes that may unnecessarily constrain the model timestep. The accidental creation of a very small model cell (often a thin triangle) is a common issue and these “bad” cells must be removed from the mesh in order to achieve maximum model efficiency.
 - The Aquaev XMS Wiki provides several mesh construction “rules” that will assist the user to create a clean mesh: http://www.xmswiki.com/xms/SMS:Mesh_Quality
- The mesh should accurately represents the bathymetry and topography, this can be checked by:
 - Specifying ‘ZB’ as a mapped output variable (either SMS Data File or netCDF format, see Section 5) and running the TUFLOW FV model for short period. The ZB results represent the model bathymetry and should be compared to the base bathymetry and topography dataset.
 - By default, TUFLOW FV automatically outputs a number model geometry files to the log directory (*_geo.nc and a series of .csv files). The .csv files provide a log of the model geometry and structure inputs and are an important check for flood applications. Using 3D surface modelling software, the user can create a TIN of the dataset and compare it against the base bathymetry/topography dataset and structure locations.

The TUFLOW FV Wiki lists a number of ways to review a simulation timestep and also various ways to increase the efficiency of a model:

http://fvwiki.tuflow.com/index.php?title=A_Model_Runs_Slow

3.8 Calibration / Validation / Sensitivity Testing

Calibrate the model to available data.

Verify the model to another set of independent data, preferably from a different location and/or a different time (with correspondingly different physical conditions).

Where knowledge or data is lacking, perform sensitivity tests on model parameters to quantify the uncertainty of model results.

Calibration is the process where the parameters of a model are adjusted, within reasonable bounds, so that results match measurements. Validation is the process where a calibrated model is compared to measurements from a different period with different physical conditions. In combination, calibration and verification prove that the model can replicate the physical processes and is a useful tool.

Choice of measurement periods for calibration depends upon the physical processes that need to be captured in the model. Typically, timeseries of response (for example river discharge, stage or tidal variations and current speed/direction) are more valuable for calibration purposes compared to instantaneous spot readings, however all relevant, reliable data should be absorbed into a calibration exercise.

As a minimum requirement for calibration and validation of a hydrodynamic tidal model, the following measurements are recommended:

- Calibration: A timeseries of water level, current speed and current direction at two or more locations, performed over a period that captures the tidal variation (e.g. over a spring-neap period)
- Validation: A timeseries of water level, current speed and current direction at two or more locations, over an independent period.

If seasonal variations are important, this exercise could be repeated at different times of year.

Example tidal calibration timeseries plots are shown in Figure 3-4 and Figure 3-5. The water level data was obtained from a permanent tide recording location. The current data was recorded using a fixed-location, bottom-mounted Acoustic Doppler Current Profiler (ADCP) instrument. TUFLOW FV points output has been obtained from these locations for direct comparison with the recorded data.

Flow (discharge) measurement across the entrance to an estuary or harbour is also a valuable model calibration datasets as shown in Figure 3-6. These measurements are typically obtained using a boat-mounted ADCP and performing continuous transects across the entrance or channel over a tide cycle. The model output corresponds to the flow through the location where the ADCP transect data was recorded. This type of output is obtained using the TUFLOW FV flux command.

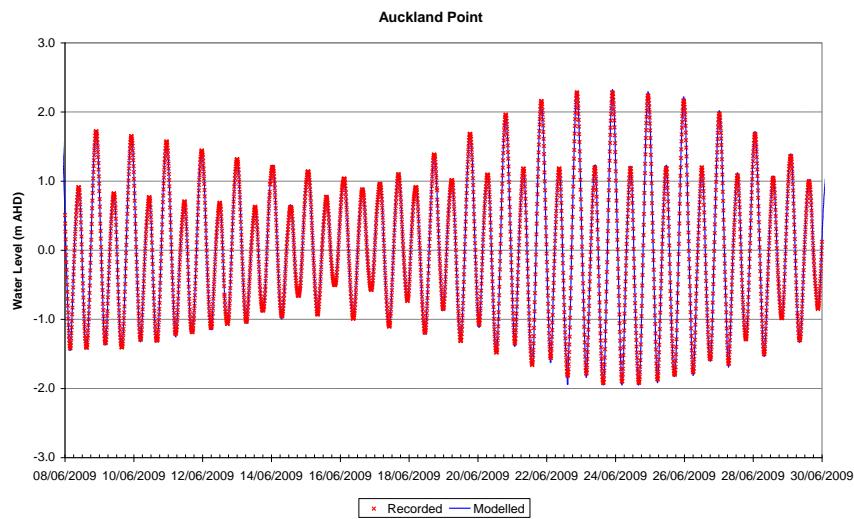


Figure 3-4 Example Tidal Water Level Timeseries Calibration Plot

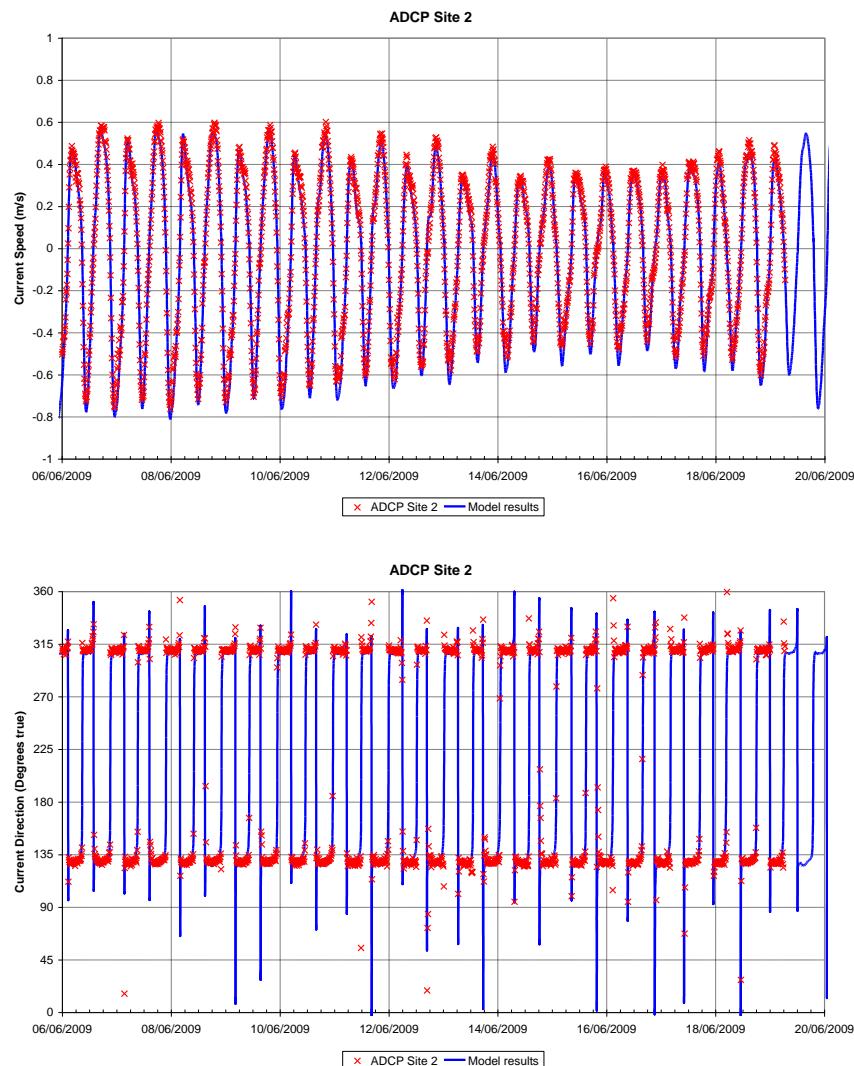


Figure 3-5 Example Current Speed and Current Direction Timeseries Calibration Plots for a Tidal Estuary

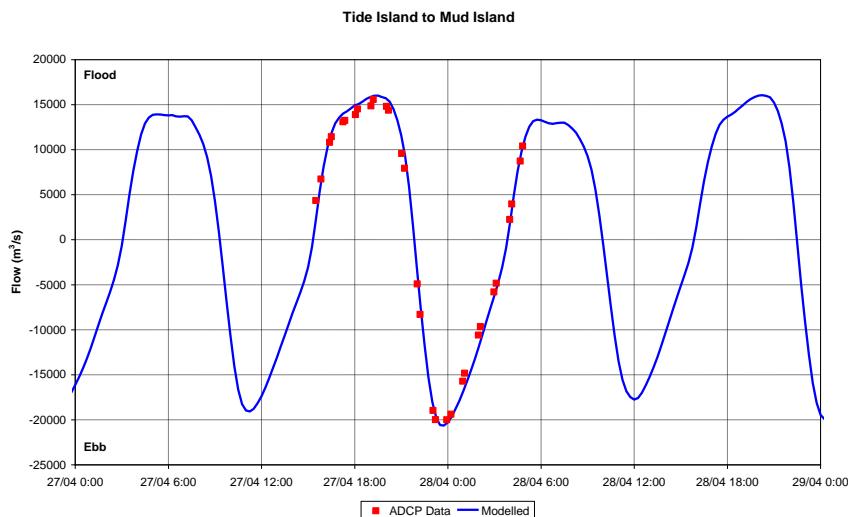


Figure 3-6 Example Flow Timeseries Calibration Plots for a Tidal Estuary

Overland flow calibration is less dependent upon instantaneous measurements performed at the time of the modelling study and more dependent upon historical records of floods. In these circumstances, all available information should be sought, quality checked and analysed, and used in the calibration exercise.

If a model cannot be calibrated due to a lack of data, don't despair; application of an uncalibrated model is not a complete waste of time. Be cautious with the model; interpret the results as indicators of specific trends and processes which, when combined with available data and experience, can provide worthwhile information.

4 Data Input

4.1 Simulation Control Files (.fvc)

The TUFLOW FV simulation control file (.fvc) is simple a command or keyword driven text file. The commands are entered free form based on the rules described below. Comments may be entered at any line or after a command. The typical simulation control file structure is shown in Figure 4-1 and the common data inputs are described in this Chapter. A number of simulation control file examples can be found on the TUFLOW FV Wiki: <http://fvwiki.tuflow.com/>.

Control File Rules

1. Only one command can occur on a single line.
2. A “==” following a command indicates the start of the parameter(s) for the command.
3. “#” or “!” represent comment syntax. All text after the comment command from that point onward will be ignored. This is useful for “commenting-out” unwanted commands, and for including modelling documentation within the control file.
4. Comments may be entered at any line or after a command.

The .fvc file sets simulation parameters and directs input from other data sources. It is the top of the tree, with all input files accessed via the .fvc file or files referred from the .fvc file.

A simple .fvc file example is shown below. Minimum input requirements for a basic 2D hydrodynamic simulation include:

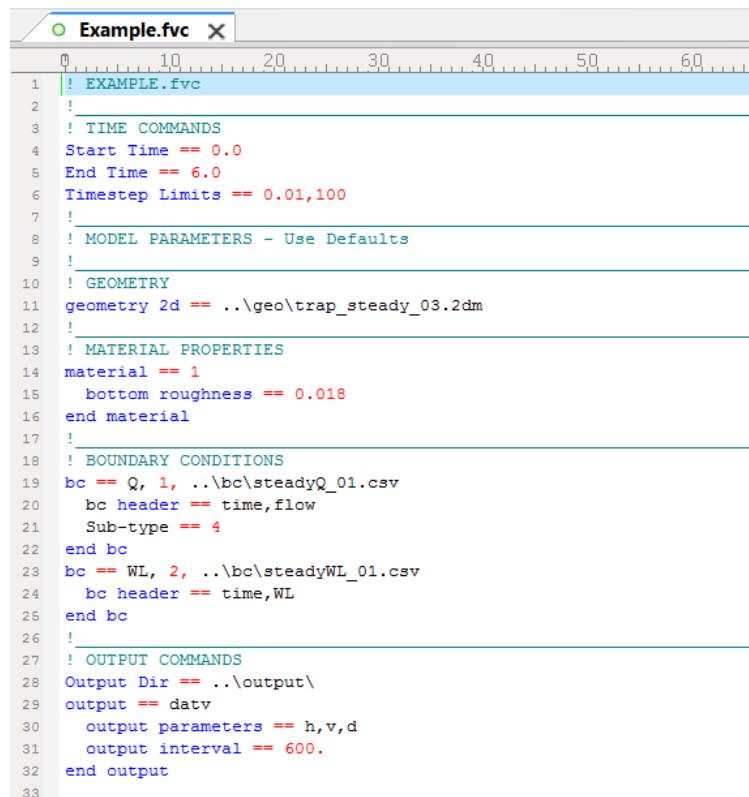
Table 4-1 Minimum Model Input Requirements

Input Category	Comment
Time Commands	Start time, end time and adaptive timestep limits
Geometry Commands	2dm mesh file: Including the schematisation of the model mesh, elevation data and spatial representation of different landuse (material roughness') within the model domain.
Material Commands	Material roughness values for each landuse defined within the 2dm mesh file
Boundary Conditions	Conditions at the open and closed boundaries and the initial condition
Output Commands	Variables, locations and formats to be output

A full list of the available TUFLOW FV commands is provided in Appendix A and Appendix B. In addition to the minimum input requirements listed above and in , advanced user commands enable full flexibility to:

- Specify alternate model parameters to the default settings

- Modify the model elevations
- Represent structures within the model domain (typically for flood or overland flow applications)
- Simulate the advection-diffusion of scalars (typically for sediment transport or water quality applications)



```

1 ! EXAMPLE.fvc
2 !
3 ! TIME COMMANDS
4 Start Time == 0.0
5 End Time == 6.0
6 Timestep Limits == 0.01,100
7 !
8 ! MODEL PARAMETERS - Use Defaults
9 !
10 ! GEOMETRY
11 geometry 2d == ..\geo\trap_steady_03.2dm
12 !
13 ! MATERIAL PROPERTIES
14 material == 1
15   bottom roughness == 0.018
16 end material
17 !
18 ! BOUNDARY CONDITIONS
19 bc == Q, 1, ..\bc\steadyQ_01.csv
20   bc header == time,flow
21   Sub-type == 4
22 end bc
23 bc == WL, 2, ..\bc\steadyWL_01.csv
24   bc header == time,WL
25 end bc
26 !
27 ! OUTPUT COMMANDS
28 Output Dir == ..\output\
29 output == datv
30   output parameters == h,v,d
31   output interval == 600.
32 end output
33

```

Figure 4-1 Example TUFLOW FV Control File Syntax

4.2 Geometry Inputs

Geometry inputs are primarily specified by the model mesh. The model mesh defines the model schematisation, the elevations and landuse specification (material roughness) which applies to each element.

4.2.1 Mesh Generation

The primary goal when designing a flexible mesh is to **describe the key bathymetric and hydrodynamic features using the least, largest element sizes possible**. This is why flexible meshes are used; to optimise computational efficiency whilst achieving desired modelling accuracy.

Creating a mesh is a combination of manual and automated steps. Maintaining a reasonable amount of manual intervention into the design of the mesh will ultimately produce a far more efficient mesh which will be more accurate and computationally efficient.

Using SMS, a TUFLOW FV mesh (2dm) is constructed using nodes, arcs and vertices. These “mesh controls” are generally positioned manually by the modeller using their preferred mesh generation tool. Important features of an area to be modelled may include islands, rivers and inlets, deep channels or man-made infrastructure. A good mesh is constructed using the mesh controls (nodes, vertices and arcs) to neatly resolve the important features within the model domain.

Figure 4-2 provides an example of the mesh controls and the resulting mesh for a section along a river bend. The left panel shows the mesh controls, namely:

- Nodes (red circles)
- Arcs (lines between two nodes)
- Vertices (small black squares along an arc)

The positions of the mesh controls have been defined by the modeller and in this case are located to resolve the river banks and the main channel. The vertices have been distributed evenly along each arc and control the number of mesh cells that can occur along the arc. The right panel shows the resulting mesh that is generated by the mesh software.

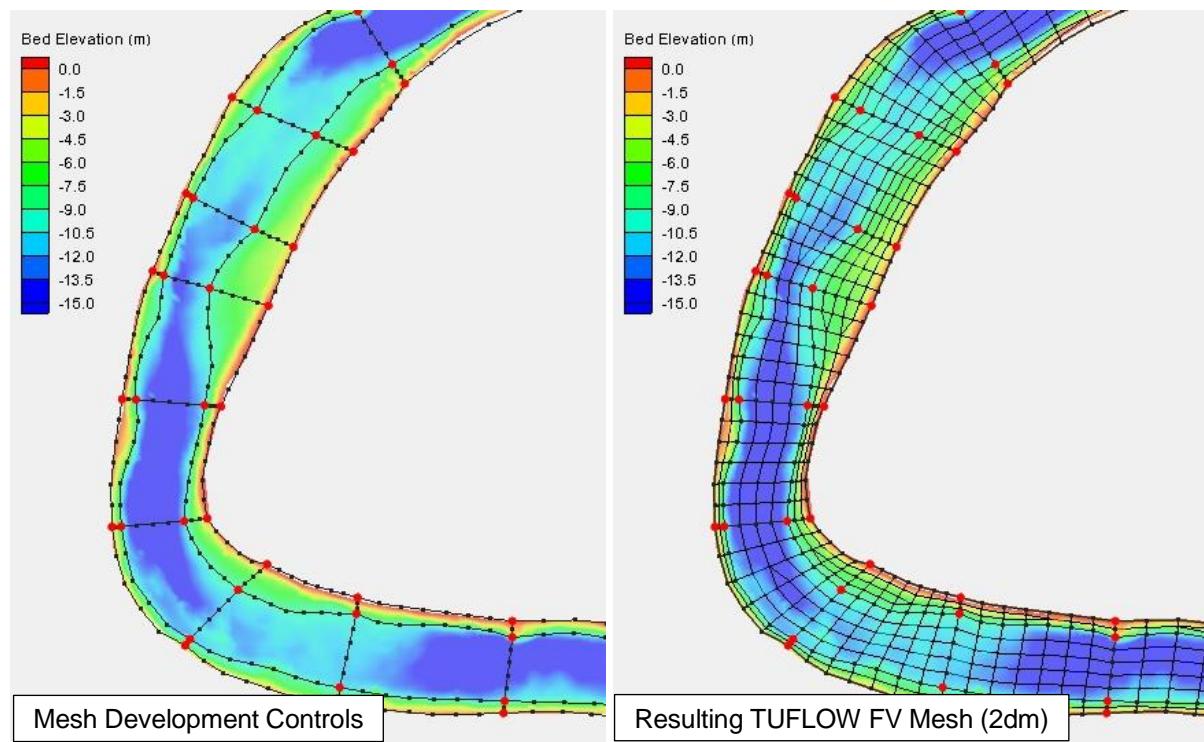


Figure 4-2 Mesh Development Example

The TUFLOW FV wiki includes a series of tutorial models which step through the process of developing a model. New users are advised to complete these tutorial models to learn how to create a model mesh and setup a TUFLOW FV model.

TUFLOW FV Tutorial Models

- Tutorial Module 1: http://fvwiki.tuflow.com/index.php?title=Tutorial_Module01
- Tutorial Module 2: http://fvwiki.tuflow.com/index.php?title=Tutorial_Module02

- Tutorial Module 3: http://fvwiki.tuflow.com/index.php?title=Tutorial_Module03
- Tutorial Module 4: http://fvwiki.tuflow.com/index.php?title=Tutorial_Module04

Additional Mesh Generation Tips: http://fvwiki.tuflow.com/index.php?title=Mesh_Generation_Tips

4.2.1.1 Mesh File Format (.2dm)

This section provides a reference to the various components of the mesh file and provides further insight into how TUFLOW FV uses it.

Unstructured mesh geometries can be created using any suitable mesh generation tool. As a preference, BMT uses the SMS Generic Mesh Module (www.aquaveo.com/sms) for building meshes. As a result the TUFLOW FV mesh file format is the SMS mesh file format.

Setting up and running a TUFLOW FV model simulation does not necessarily require a detailed line by line inspection of the mesh file; SMS (or another mesh generator) provides a graphical interface to do this instead. Nevertheless, a modeller may find it necessary at times to interrogate the 2dm file in detail.

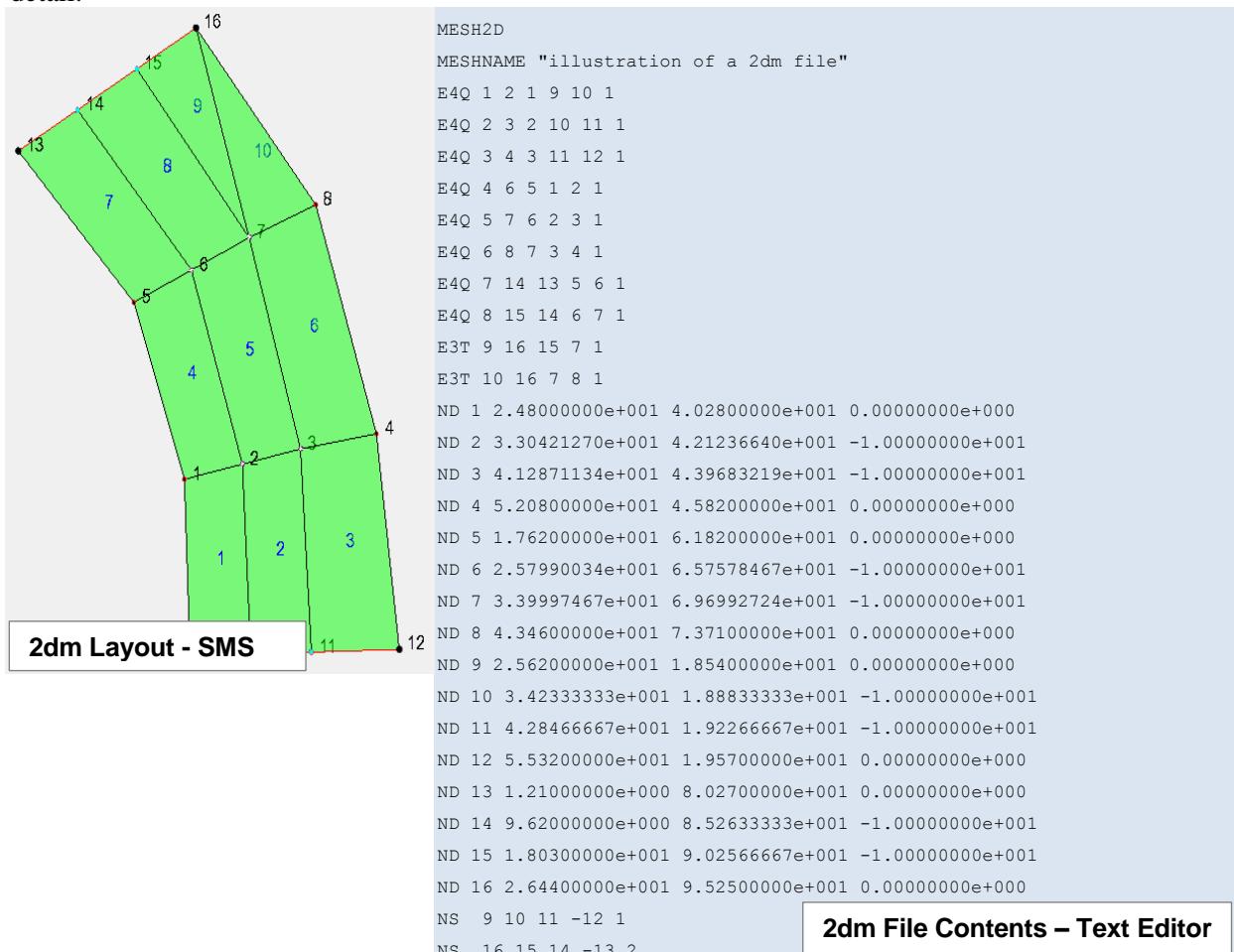


Figure 4-3 Example 2dm File

The 2dm file format is used to define the TUFLOW FV mesh. It is an ASCII format from the SMS Generic Mesh Module. The contents of the file relevant to TUFLOW FV simulations are:

Nodes: Lines that commence with a “ND” are nodes, or the points that define the edges of the elements. Each ND line describes the node ID and its x, y and z (i.e. bed level) coordinate. The screen shot in Figure 4-4 shows node 236 selected and its corresponding position and elevation displayed in the X,Y,Z dialog boxes.

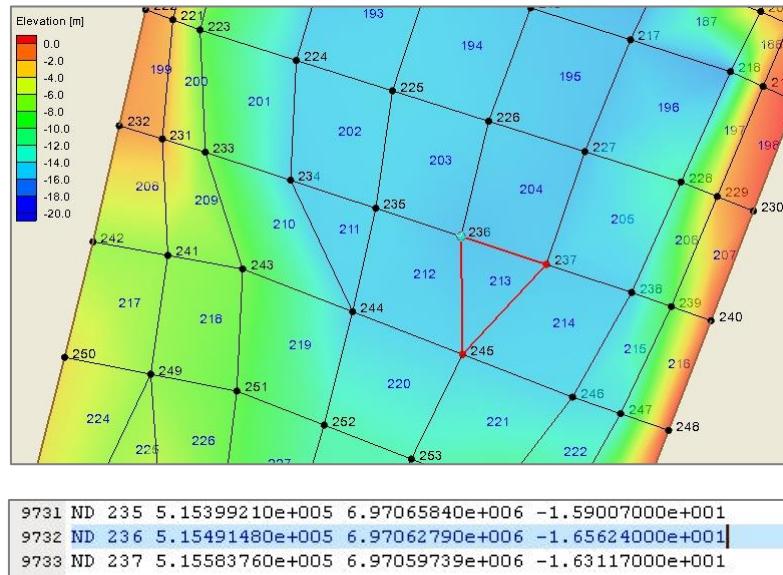


Figure 4-4 Example Node Definition

Elements: Lines that commence with an “E4Q” are quadrilateral (4 sided) elements. Each E4Q line describes the element ID, the four nodes that define its connectivity and spatial extent (in a counter-clockwise direction) and the material type.

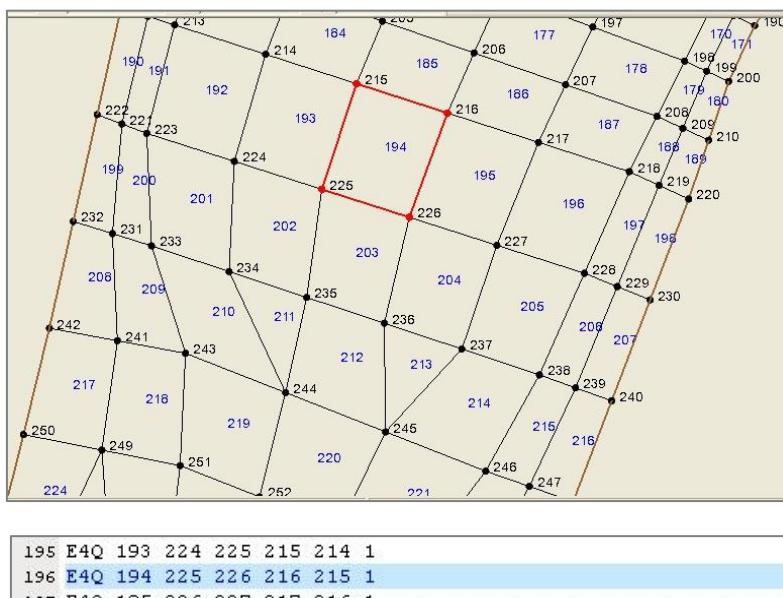


Figure 4-5 Example Quadrilateral Element Definition

Similar to E4Q, the “E3T” lines are triangular elements. Each E3T line describes the element ID, the three nodes that define its connectivity and spatial extent (in a counter-clockwise direction) and the material type.

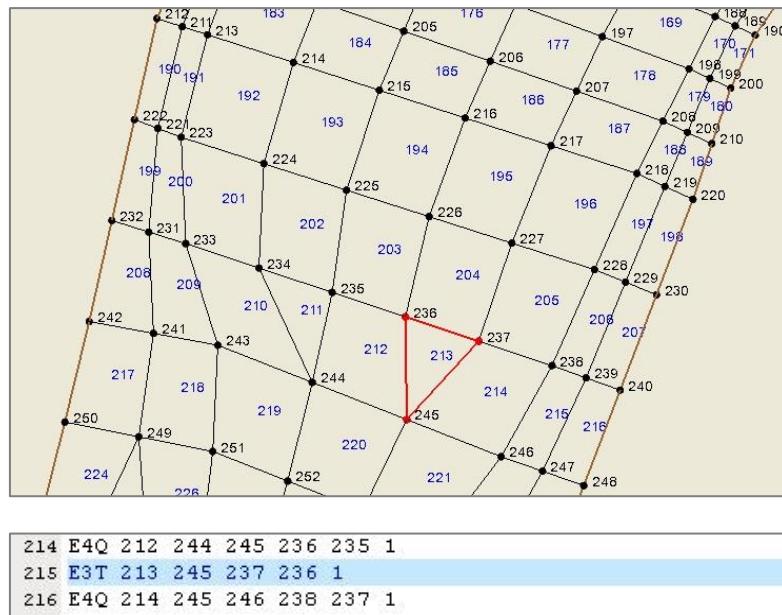


Figure 4-6 Example Triangular Element Definition

Nodestrings: Lines that commence with a “NS” are nodestrings, which are used to define boundary conditions. Each NS line defines the series of nodes that form the string, the last node number is assigned as negative. The number following the negative number is the nodestring ID.

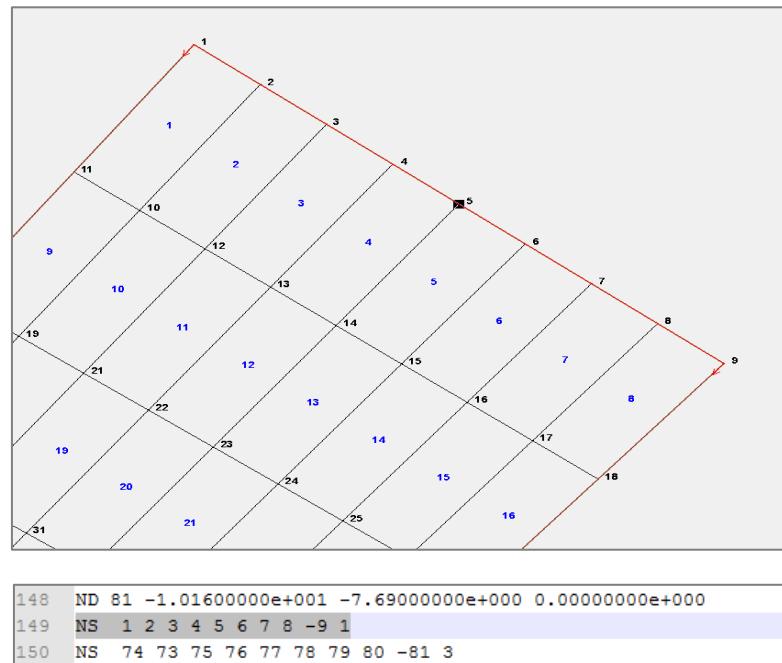


Figure 4-7 Example Nodestring Definition

4.2.2 Cell Centred Computations

The mesh cells (or elements) are the computational blocks of the finite volume approach used by TUFLOW FV. This means that TUFLOW FV uses a single bed elevation value assigned to each cell in its calculations, and then produces output that is applicable for each cell (cell velocities are derived from the values across each cell face).

At present, the SMS Data File Format only permits mesh file bed elevation input and simulation output to be stored on the cell nodes. When TUFLOW FV reads the mesh geometry file at the beginning of a model simulation the cell centred bed elevations are interpolated from the mesh nodes. Then, when writing SMS Data File Format output, TUFLOW FV interpolates cell centred results back onto the cell nodes.

In many instances, this interpolation of both input bed elevations and output results is not an issue. However, there may be instances where this is not preferred. The [Cell Elevation File](#) command allows users to specify exact cell centred elevation values (refer to Section 4.2.2.1). Options for viewing and processing TUFLOW FV cell centred results typically require access to numerical analysis software with a netCDF library. BMT typically uses MATLAB and can be contacted via TUFLOW support for further information post-processing TUFLOW FV cell centred results: support@tuflow.com.

4.2.2.1 Elevation Update Options

The 2dm file defines the base elevations within the model. These elevation values can be updated (adjusted or overwritten) using a variety of geometry commands. These commands allow the user to build a number of specific features into the model geometry in a systematic, structured manner, starting from the underlying geometry in the 2dm file and adding specific features (roads for example).

- [Cell Elevation File](#) and [Cell Elevation Points](#)

These commands can be used to update cell elevations by either referencing a cell ID or specifying an x,y coordinate which falls within a cell. These commands are useful when a user wishes to define an exact elevation value to a single cell, or multiple cells within a model (instead of applying interpolated values from the cell corners/nodes). Two example cell elevation files (.csv) are shown below.

Table 4-2 Cell Elevation File Examples

X	Y	Z
136.881	-11.0571	-58.7456
136.873	-11.0302	-58.7456
136.824	-11.1601	-55.6548
136.972	-10.7822	-58.9239
136.963	-10.7534	-60.1127
136.987	-10.7353	-59.9938
136.981	-10.7053	-59.9344
137.005	-10.6874	-59.9344

ID	Z
1	-58.7456
2	-58.7456
3	-55.6548
4	-58.9239
5	-60.1127
6	-59.9938
7	-59.9344
8	-59.9344

OR

More than one “cell elevation” command line can be defined with a simulation control file (fvc), and/or more than one point per cell can be entered. Depending upon input preference, each z value will overwrite the preceding z value entry, or an average of all points within each cell will be

assigned. If multiple cell elevation files are listed, where inputs from one entry fall with the cell of a preceding entry, the later (lower) entry supersedes the previous (higher).

The cell elevation file option does not interpolate between successive points. If using the cell elevation file for continuous linear features (such as a road or levee), ensure that the point resolution is sufficiently fine to accurately represent the elevations along the feature, or use the [Cell Elevation Polyline](#) command (discussed below).

Note: A list of all cell ID and corresponding X,Y co-ordinates can be obtained from the geometry log files.

- [Cell Elevation Polyline](#)

A cell elevation polyline can be used to define linear features which act as critical hydraulic controls within the study area. This is a useful feature for defining the crest elevation of levees or raised roads which traverse a floodplain, or alternatively the bed elevation of tributary creeks.

Input data is defined using a csv file containing X,Y, Z and ID, data. Points within the csv file define the vertices along the poly line. Intermediate cell elevations between vertices are interpolated.

Multiple polylines can be defined within a single csv file. The ID attribute is used to differentiate between the different polylines. A unique command line input for each polyline is required within the simulation control file (fvc), as shown below.

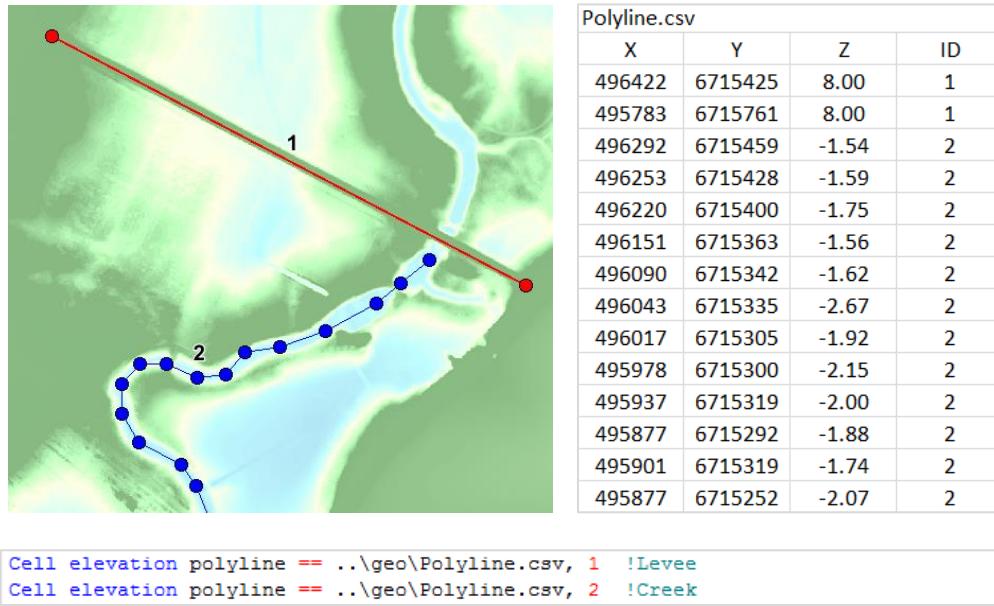


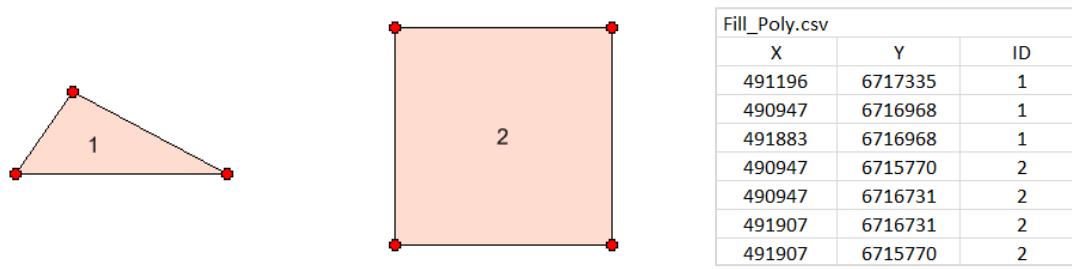
Figure 4-8 Cell Elevation Polyline Example

- [Cell Elevation Polygon](#)

The ‘Cell Elevation Polygon’ command applies a single elevation over a polygon. An example of its application is to assign a specific elevation to cells representing a reclamation or infill of a proposed development.

Input data is entered using a csv file containing X,Y, and optionally ID, data. Points within the csv file define the perimeter of the polygon. The definition of points needs to be consecutively listed and can be either clockwise or counter-clockwise.

As per the [Cell Elevation Polyline](#) example, multiple polygons can be defined within a single csv file. The ID attribute is used to differentiate between the different polygons. A unique command line input is required for each polygon within the simulation control file (fvc), as shown below.



```
Cell elevation polygon == ..\geo\Fill_Poly.csv, 6.0 , 1 !Triangle Fill Example: raised to 6mAHD
Cell elevation polygon == ..\geo\Fill_Poly.csv, 6.0 , 2 !Square Fill Example: raised to 6mAHD
```

Figure 4-9 Cell Elevation Polygon Example

4.3 Boundary Conditions

Boundary conditions are defined in the simulation control file within a [boundary condition](#) block, requiring a unique block definition for each input to the model (type and location). TUFLOW FV offers a wide variety of boundary condition types. Basic hydrodynamic boundary condition types are summarised in Table 4-3. Advance boundary conditions types, used during advection dispersion, 3D, sediment transport/morphology and water quality modelling are listed in Table 4-4. The tables list the boundary condition type ID, its description, method of application to the model, boundary condition data file format and the default column headers for the input file.

When defining a boundary condition block, the first line is used to specify the input type, location and reference file which contains the relevant boundary condition data. Where boundary conditions inputs vary from default values, these details are specified within the structure block.

Three methods are available for applying the model boundary condition inputs:

- 1) Cell: Applying the input boundary condition at a single cell location.
- 2) External Nodestring: Applied along a series of cell faces, defined by a nodestring.
 - a. Nodestrings can be defined within a model either when the model mesh is created (refer to Section 4.2.1) or using the [nodestring polyline](#) command.
- 3) Global: Applied to every computation cell within the model.
 - a. Common values can be specified across the entire model using csv file inputs; or
 - b. Gridded netCDF format data can be used to define temporally and spatially varying global inputs. If using gridded data, a [grid definition file](#) and [grid definition variables](#) are first required to define grid coordinates and variable names within the input file. These commands need to be specified prior to the boundary condition block.

When developing complex models with a large number of boundary conditions, [Include](#) files are recommended to manage/categorise data inputs. Some example boundary condition blocks are shown below. Worked examples are available via the TUFLOW FV tutorial models on the TUFLOW FV Wiki: <http://fvwiki.tuflow.com>

```
!
!BOUNDARY CONDITIONS
BC == QC, 6837,15228, ..\bc\Trib_Q.csv      !Flow boundary condition, point location (cell x,y), bc file
    BC header == Date,Q                         !Header information within the bc file
End BC                                         !End bc block

BC == WL, 1, ..\bc\TIDE.csv                    !WL boundary, location (nodestring), bc file (using default headers)
End BC                                         !End bc block

Nodestring polyline == ..\bc>Main_Dam.csv, 2   !Inflow nodestring location
BC == Q, 2, ..\bc>Main_River_Q.csv            !Flow boundary condition, location (nodestring), bc file
    BC header == TIME,NS2                      !Header information within the bc file
End BC                                         !End bc block

Grid definition file == ..\bc\WIND_Rec.nc     !Grid boundary condition file
Grid definition variables == long,lat          !Variable names
BC == W10_GRID, 1, ..\bc\WIND_Rec.nc          !Wind boundary condition, location (global grid), bc file
    BC header == time,u,v                     !Header information within the bc file
End BC                                         !End bc block
```

Figure 4-10 Boundary Condition Block Examples

Table 4-3 Boundary Condition Types (Basic)

BC Type	BC Description	BC Method	BC file format	Default Columns Header ¹
HQ	qH relationship	External nodestring	CSV	H, Q
Q	Nodestring flow	External nodestring	CSV	TIME, Q, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
QC	Cell inflow (m ³ /s) - uses internal concentration during outflow.	Cell	CSV	TIME, Q, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
QCA	Cell inflow (m ³ /s) - uses specified concentration during outflow.	Cell	CSV	TIME, Q, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
QG	Global Cell Inflow (m/s) - uses internal concentration during outflow.	Global	CSV	TIME, Q/A, [SAL], [TEMP], [SED_1,...], [TRACE_1,...]
QGA	Global Cell Inflow (m/s) - uses specified concentration during outflow.	Global	CSV	TIME, Q/A, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
QH	qH relationship (same	External	CSV	H, Q

¹ Note that the header names listed here are defaults; if a “bc header ==” line is not included in the fvc file then these column header titles are required. If however a “bc header ==” line is included in the fvc then the header descriptions then match the column header in the csv file.

BC Type	BC Description	BC Method	BC file format	Default Columns Header ¹
	as HQ)	nodestring		
QN	Normal flow condition (automatic qH BC)	External nodestring	N/A	No CSV file required; second entry on BC line is friction slope. For example, the following lines specify a QN boundary along nodestring 2: bc == QN, 2, 0.001 end bc
WL	Water level (m or ft, see units)	External nodestring	CSV	TIME, WL, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
WLS	Sloping Water Level (m or ft, see units) WL_A = level at start of nodestring. WL_B = level at end of nodestring.	External nodestring	CSV	Time, WL_A, WL_B, , [SAL_A, SAL_B], [TEMP_A, TEMP_B], [SED_1_A, SED_1_B,...], [SCAL_1_A, SCAL_1_B,...]

Table 4-4 Boundary Condition Types (Advanced)

BC Type	BC Description	BC Method	BC file format	Default Columns Header ²
AIR_TEMP	Temperature input (C°)	Global	CSV	TIME, AIR_TEMP
AIR_TEMP_GRID		Grid	NETCDF	TIME, AIR_TEMP
CLOUD	Cloud cover (fraction) - 0 = clear, 1 = full cloud cover.	Global	CSV	TIME, CLOUD
CLOUD_GRID		Grid	NETCDF	TIME, CLOUD
CP	Cell concentration profile	Cell	CSV	DEPTH, [SAL], [TEMP], [SED_1,...], [TRACE_1,...]
CYC_HOLLAND	Parametric cyclone wind and pressure field	Global	CSV	TIME, X, Y, PO, PA, RMAX, B, RHOA, KM, THETMAX, DELTAFM, WBGX, WBGY
FB	Sediment bed flux	Cell	CSV	TIME, FLUX_SED_1,...
FBM	Moving bed sediment flux	Cell	CSV	TIME, X, Y, FLUX_SED_1,...
FC	Cell scalar flux	Cell	CSV	TIME, [FLUX_SAL]. [FLUX_HEAT],

² Note that the header names listed here are defaults; if a “bc header ==” line is not included in the fvc file then these column header titles are required. If however a “bc header ==” line is included in the fvc then the header descriptions then match the column header in the csv file.

BC Type	BC Description	BC Method	BC file format	Default Columns Header ²
				[FLUX_SED_1,...], [FLUX_SCAL_1,...]
FCM	Moving scalar flux	Cell	CSV	TIME, X, Y, [FLUX_SAL]. [FLUX_HEAT], [FLUX_SED_1,...], [FLUX_SCAL_1,...]
LW_NET	Net longwave radiation (Wm^{-2})	Global	CSV	TIME, LW_NET
LW_NET_GRID		Grid	NETCDF	TIME, LW_NET
LW_RAD	Downward longwave radiation (Wm^{-2})	Global	CSV	TIME, LW_RAD
LW_RAD_GRID		Grid	NETCDF	TIME, LW_RAD
MSLP_Grid	Mean sea level pressure field (hPa)	Grid	NETCDF	TIME, MSLP
OBC	Fully specified boundary condition ³⁴	External nodestring	CSV	TIME, WL, U, V, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
OBC_PROF	Fully specified open boundary condition profile	External nodestring	CSV	TIME, WL, U, V, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
OBC_CURT	Fully specified open boundary condition curtain	External nodestring	CSV	TIME, WL, U, V, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
OBC_GRID	Fully specified boundary condition ⁵⁶	Grid	NETCDF	TIME, WL, U, V, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]
OP	Zero-gradient	External nodestring	N/A	Not Required
PRECIP	Precipitation (mday^{-1})	Global	CSV	TIME, PRECIP
PRECIP_GRID	Precipitation grid (m/day)	Grid	NETCDF	TIME, PRECIP
REL_HUM	Relative humidity (%)	Global	CSV	TIME, REL_HUM
REL_HUM_GRID		Grid	NETCDF	TIME, REL_HUM
RS	Reflective, free slip	External nodestring	N/A	N/A
RNS	Reflective, no Slip	External nodestring	N/A	N/A
SCALAR	Scalar concentration	External nodestring	CSV	TIME, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]{TRACE,...}[WQ,...]

³ Note: this boundary application can be used to specify a supercritical flow condition.

⁴ Note: u,v are cartesian velocity vectors (components in x and y directions).

⁵ Note: this boundary application can be used to specify a supercritical flow condition.

⁶ Note: u,v are cartesian velocity vectors (components in x and y directions).

BC Type	BC Description	BC Method	BC file format	Default Columns Header ²
	specification			
SCALAR_PROF	Scalar concentration profile	External nodestring	CSV	DEPTH, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]{TRACE,...}
SCALAR_CURT	Scalar concentration curtain	External nodestring	CSV	TIME, [SAL], [TEMP], [SED_1,...], [SCAL_1,...]{TRACE,...}
SURF_TEMP	Temperature input (C°)	Global	CSV	TIME, SURF_TEMP
SURF_TEMP_GRID	Temperature input (C°)	Grid	NETCDF	TIME, SURF_TEMP
SW_RAD	Downward shortwave radiation (Wm⁻²)	Global	CSV	TIME, LW_RAD
SW_RAD_GRID	Downward shortwave radiation (Wm⁻²)	Grid	NETCDF	TIME, LW_RAD
TRANSPORT	Transport file written from previous TUFLOW FV simulation	Conserved variables stored on TUFLOW FV mesh	NETCDF	TIME, U, F, [DIFFUSIVITY]
W10	Wind velocity at 10m (ms⁻¹) ⁷	Global	CSV	TIME, W10_X, W10_Y
W10_GRID	Wind velocity at 10m (ms⁻¹)	Grid	NETCDF	TIME, W10_X, W10_Y
WAVE	Inputs from SWAN (external spectral wave model)	Grid	NETCDF	TIME, HSIGN, TPS, DIR, UBOT, TMBOT, FORCE_X, FORCE_Y
WL_CURT	Water level curtain boundary condition	External nodestring	NETCDF	TIME, NS_1
ZG	Zero gradient boundary	External nodestring	N/A	N/A

4.4 Structures

TUFLOW FV includes a wide range of structure control options typically used in overland flow simulations. The term ‘Structure’ encompasses a range of hydraulic structures (bridges, culverts and weirs) and also topography/bathymetry control commands.

⁷ Note: x,y are cartesian velocity vectors (components in x and y directions).

Structure controls are defined using a [structure](#) block, requiring a unique block definition for every structure within the model (For example, two separate structure blocks will be required to define two bridges within a model).

The header line of the block is used to define the connection type and in most cases the location ([auto](#) [weir](#) being the exception). The structure location can be defined in the following ways:

Hydraulic Structures (Bridges, Weirs, Culverts)

- 1) Single Location (Nodestring): Defined using a single nodestring. Mass will be transported between neighbouring cells, regulated through the cell face by the flow controls defined by the structure. This option is commonly used to model bridges, weirs and in some cases culverts (depending on the model resolution).
- 2) Multiple Locations (Linked nodestrings): Defined by two separate nodestrings. Mass will be transported between the two locations in the model, regulated by the flow controls defined by the structure. This option is commonly used to represent culverts within a model.

Nodestrings can be defined within a model either when the model mesh is created (refer to Section4.2.1) or using the [Nodestring Polyline](#) command.

Other Controls (Bathymetry/Topography Updates)

- 1) Single Cell (Cell): Topography controls are applied to a single cell location.
- 2) Multiple Cells (Zone): Topography controls are applied to multiple cells which fall within a common region.

Details of the structure characteristic (type, design details) are specified within the [structure](#) block. When developing complex models with a large number of structures, [Include](#) files are recommended to manage the data inputs.

Some example structure blocks are shown below. The following sections provide a detailed description outlining the various structure options and the recommended way to define them within a model.

```

!
!STRUCTURE DEFINITIONS
Nodestring polyline == ..\geo\Struct_NString_002.csv, 5      !Weir location
Nodestring polyline == ..\geo\Struct_NString_002.csv, 6      !Bridge location
Nodestring polyline == ..\geo\Struct_NString_002.csv, 7      !Culvert inlet location
Nodestring polyline == ..\geo\Struct_NString_002.csv, 8      !Culvert outlet location
Nodestring polyline == ..\geo\Struct_NString_002.csv, 9      !Culvert inlet/outlet location

Structure == Nodestring, 5                                     !Structure location (NS 5)
  Flux function == Weir                                      !Weir flag
  Properties == 9.5,1.6                                       !Weir elevation(mAHD), weir coeffecient
End Structure

Structure == Nodestring, 6                                     !Structure location (NS 6)
  Energy loss function == coefficient                         !Loss type
  Form loss coefficient == 0.2                                !2D bridge - pier loss
End Structure

Structure == Linked nodestrings, 7,8                        !Structure location (NS 7,NS 8)
  Flux function == Culvert                                 !Culvert flag
  Culvert File == ..\geo\Culvert_dbase.csv, 1            !Culvert details
End Structure

Structure == Nodestrings, 9                                  !Structure location (NS 9)
  Flux function == Culvert                               !Culvert flag
  Culvert File == ..\geo\Culvert_dbase.csv, 2          !Culvert details
End Structure

Structure == autoweir
End Structure

```

Figure 4-11 Example Structure Definition

4.4.1 Bridges

Bridges can be modelled using two methods, by applying a form loss coefficient or alternatively specifying a water level afflux flow (hQh) relationship. These two approaches are discussed below.

4.4.1.1 Form Loss Method

Flow constriction commands allow the user to constrict the flow across a 2D cell side in a number of ways so as to model large hydraulic structures such as bridges and banks of box culverts. 2D cell sides can be modified in the following ways:

- Addition of form (energy) losses to the cell calculations via a [Form Loss Coefficient](#). The form loss coefficient applies a head loss across a cell face according to the equation:

$$\Delta h = FLC \frac{v^2}{2g}$$

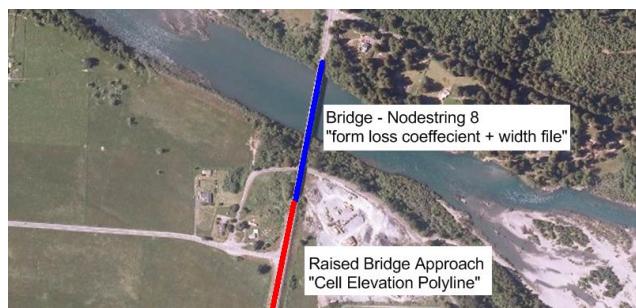
- Specification of a lid (obvert or soffit) on the cell and refinement of the waterway flow width. This is achieved using a [Width File](#).

An example structure is presented in Figure 4-12. When adapting structure loss coefficients from a 1D model or from coefficients that apply across the entire waterway, for example, from Hydraulics of Bridge Waterways (FHA 1973), the following should be noted:

- The 2D solution automatically predicts the majority of “macro” losses due to the expansion and contraction of water through a constriction, or round a bend, provided the resolution of the

grid is sufficiently fine. It is recommended that raised bridge approaches/abutments should be represented by the model topography (i.e. not included as a contribution to the form loss coefficient). A [Cell Elevation Polyline](#) may be useful for defining these topographic features. Where the waterway width varies slightly from the cumulative width of the cells across which the structure is being applied, a [Width File](#) can be used to refine the flow area and define the structure soffit within the model

- Where the 2D model is not of fine enough resolution to simulate the “micro” losses (eg. from bridge piers, vena contracta, losses in the vertical (3rd) dimension), additional form loss coefficients and/or modifications to the cells widths and flow height need to be added. This can be done by using a [Form Loss Coefficient](#).
- The additional or “micro” losses, which may be derived from information in publications, such as Hydraulics of Bridge Waterways, should be distributed evenly across the waterway (i.e. rather than being too specific about the representation of each individual cell).
- The head loss across key structures should be reviewed, and if necessary, benchmarked against other methods (eg. using HEC-RAS or Hydraulics of Bridge Waterways). Note that a well-designed 2D model will be more accurate than a 1D model, provided that any “micro” losses are incorporated. Ultimately, the best approach is to calibrate the structure through adjustment of the additional “micro” losses – but this, of course, requires good calibration data!



Width_Example.csv		
Z	WIDTH	Comment
-5	0	!Bed Invert
-4.9	100	!Bridge Opening
5	100	!Bridge Opening
5.1	0	!Bridge Deck and Railing
7.1	0	!Bridge Deck and Railing
7.2	100	
12	100	

```

Structure == Nodestring, 6           !Structure location (NS 6)
Energy loss function == coefficient !Loss type
Form loss coefficient == 0.2         !pier losses (Hydraulic of Bridged Waterways)
Width file == ..\geo\Width_Example.csv !Width file
End Structure

```

Figure 4-12 2D Bridge Example (Form Loss Coefficient)

4.4.1.2 hQh Specification

The hQh structure option ([Flux Function == Matrix](#) combined with [Flux File](#)) leaves the calculation of flow through a structure to the user. This makes the hQh structure flexible in its application (any structure, be it weir, culvert, pipe, etc. can be applied) but also means that the user needs to create the hQh relationship.

Structure flow is determined from an “hQh” relationship; flow (Q) across the nodestring is determined by the upstream water level (h_{us}) and downstream water level (h_{ds}), as specified in a user defined matrix of values (the hQh table). The following figure illustrates this.

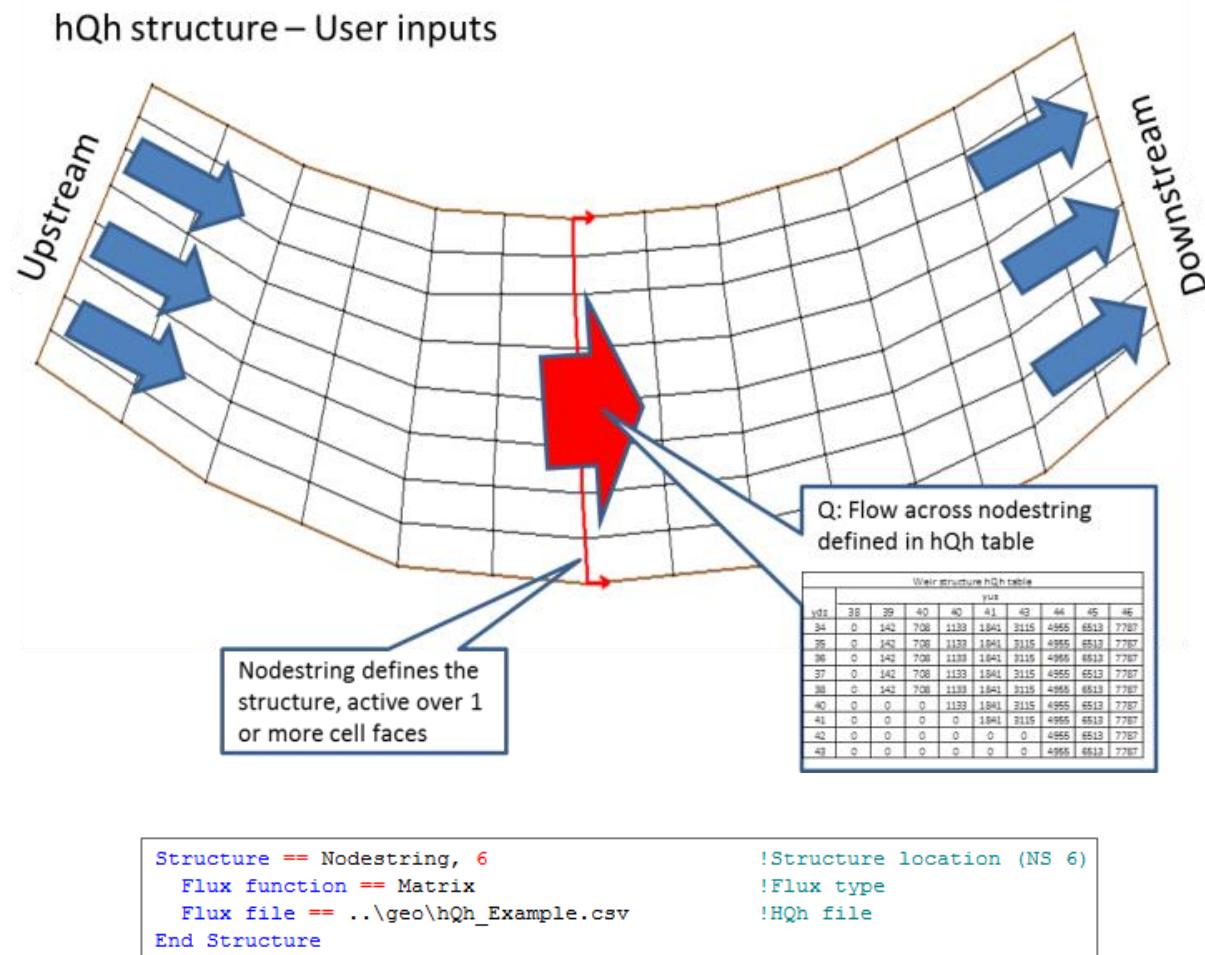


Figure 4-13 hQh Structure Example

The logic process for computing structure flow is as follows:

- 1 Flows in the hQh table are distributed across the nodestring according to the relative widths of each individual cell face (a cell face being the connecting line between two cells). Thus, each individual cell face has a unique hQh table with Q values factored from the original hQh table according to the cell face width.
- 2 During a model simulation step, at each cell face the upstream and downstream water levels are used to obtain Q from the hQh matrix.
- 3 A check is performed between the tabulated flow (Q_{hQh}) and that calculated using the Shallow Water Equation (Q_{SWE}). If the tabulated flow (Q_{hQh}) is less than the Shallow Water Equation (Q_{SWE}) equivalent, it is applied. Conversely, if the tabulated flow (Q_{hQh}) is greater than the Shallow Water Equation (Q_{SWE}) value, the Shallow Water Equation flow is used.
- 4 Momentum is actively transferred through the structure based on Q_{hQh} and upstream velocity.

A hQh relationship can be calculated either from first principles, or using other models. In particular, HEC-RAS is commonly used to establish flow conditions through structures. The calculated Q values from HEC-RAS simulations for a range of upstream and downstream water levels can provide a relatively straightforward means of creating a hQh matrix.

When deriving the hQh relationship, care must be taken to ensure that entry and exit losses are being applied appropriately. Depending upon the layout of the structure in the mesh design, the hQh relationship can represent all of the losses that occur in a structure (macro and micro) or only internal (micro) losses. This concept is described in Section 4.4.1.1 and illustrated below. The format requirements for the hQh matrix csv file are provided in [Flux File](#) command description.

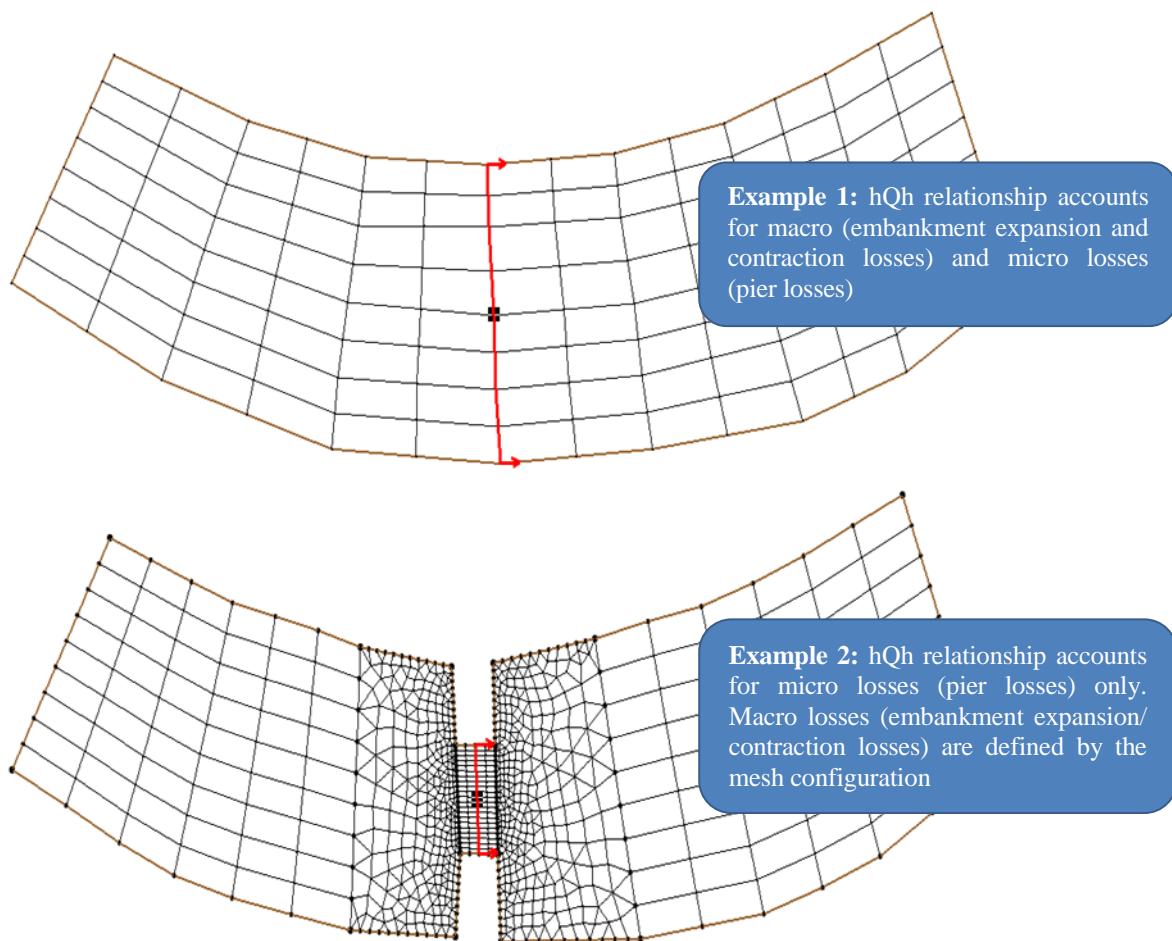


Figure 4-14 hQh Calculation Design Options

4.4.2 Culverts

Sub-grid sized culverts can be modelled as 1D structures. The standard structure equations which have been used by TUFLOW Classic the late 1990's have been include in TUFLOW FV⁸. The calculations

⁸ For benchmarking of culvert flow to the literature, see Huxley (2004):

<http://www.tuflow.com/Downloads/TUFLOW%20Validation%20and%20Testing,%20Huxley,%202004.pdf>

of culvert flow and losses are carried out using techniques from “Hydraulic Charts for the Selection of Highway Culverts” and “Capacity Charts for the Hydraulic Design of Highway Culverts”, together with additional information provided in Henderson 1966. The calculations have been compared and shown to be consistent with manufacturer's data provided by both “Rocla” and “Armco”. The equations allow for a range of different flow regimes, as summarised in Table 4-5Table 4-5 1D Culvert Flow Regimes, Figure 4-16 and Figure 4-15.

Table 4-5 1D Culvert Flow Regimes

Regime	Description
A	Unsubmerged entrance and exit. Critical flow at entrance. Upstream controlled with the flow control at the inlet.
B	Submerged entrance and unsubmerged exit. Orifice flow at entrance. Upstream controlled with the flow control at the inlet.
C	Unsubmerged entrance and exit. Critical flow at exit. Upstream controlled with the flow control at the culvert outlet.
D	Unsubmerged entrance and exit. Sub-critical flow at exit. Downstream controlled.
E	Submerged entrance and unsubmerged exit. Full pipe flow. Upstream controlled with the flow control at the culvert outlet.
F	Submerged entrance and exit. Full pipe flow. Downstream controlled.
G	No flow. Dry or flap-gate active.
H	Submerged entrance and unsubmerged exit. Adverse slope. Downstream controlled.
J	Unsubmerged entrance and exit. Adverse slope. Downstream controlled.
K	Unsubmerged entrance and submerged exit. Critical flow at entrance. Upstream controlled with the flow control at the inlet. Hydraulic jump along culvert.
L	Submerged entrance and exit. Orifice flow at entrance. Upstream controlled with the flow control at the inlet. Hydraulic jump along culvert.

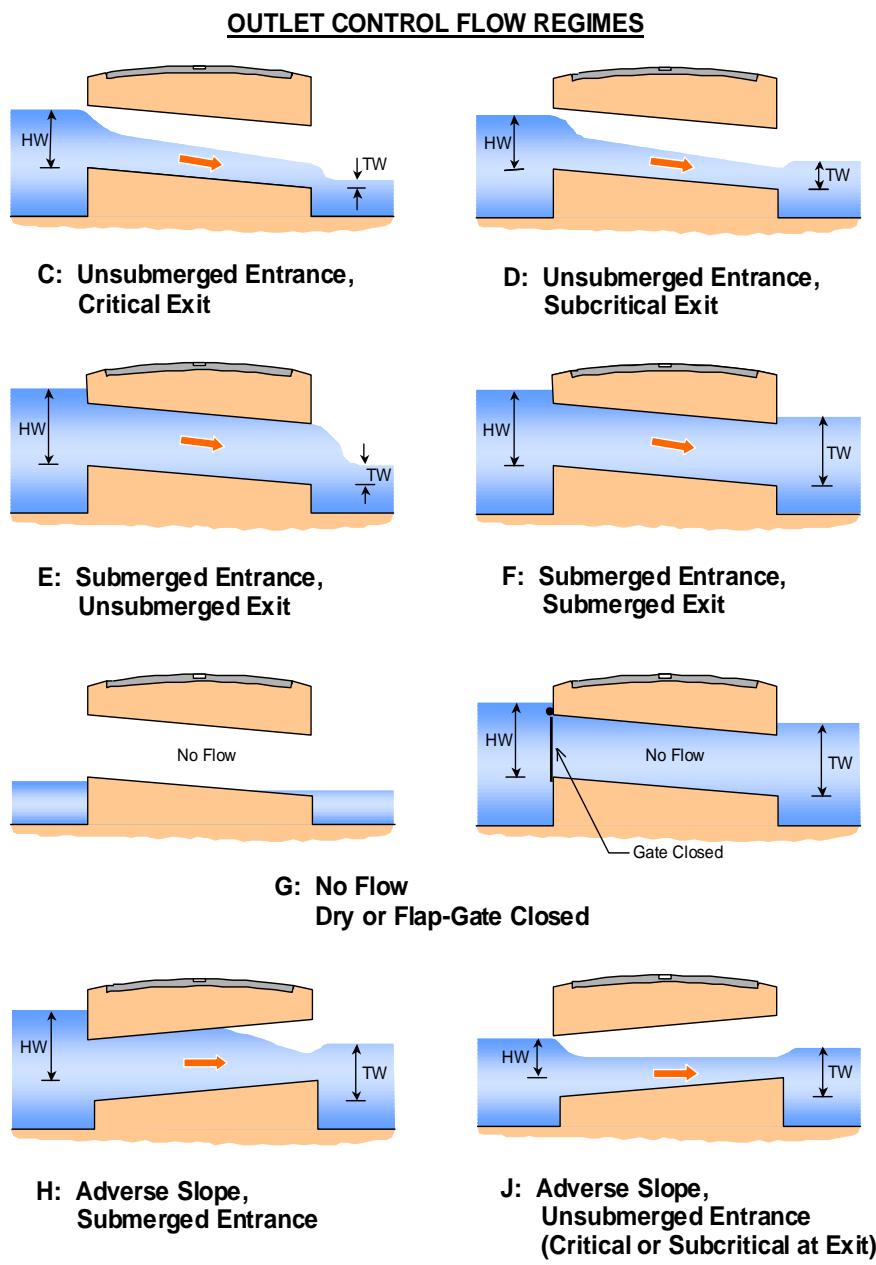


Figure 4-15 1D Outlet Control Culvert Flow Regimes

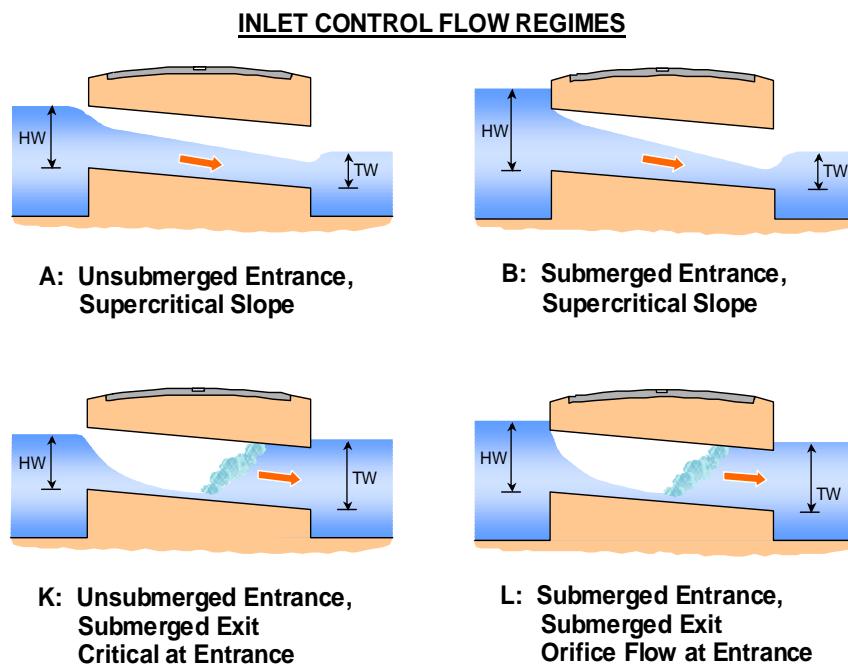


Figure 4-16 1D Inlet Control Culvert Flow Regimes

Culverts are defined using the [Flux function](#) command. In combination with the Flux function command, culvert structure information is defined within the model via a culvert database (see [Culvert file](#)). The culvert file contains a list of the culvert attributes, such as the culvert ID, culvert type, dimensions, length, upstream and downstream inverts, number of barrels, Manning's n, entrance and exit losses. A description of the required culvert file inputs is summaries in Table 4-6.

Multiple culverts can be listed within a culvert file (i.e. a unique culvert file for each culvert is not required). The culvert ID within the culvert file is used to associate a specific structure with the command line input. Example inputs are shown below in Figure 4-17.

```

Structure == Linked nodestrings, 7,8          !Structure location (NS 7,NS 8)
  Flux function == Culvert                   !Culvert flag
  Culvert File == ..\geo\Culvert_dbase.csv, 2 !Culvert details = ID 2 in culvert file
End Structure

Structure == Nodestrings, 9                  !Structure location (NS 9)
  Flux function == Culvert                 !Culvert flag
  Culvert File == ..\geo\Culvert_dbase.csv, 3 !Culvert details = ID 3 in culvert file
End Structure

```

Culvert_database.csv																
ID	Type	Ignore	Len_or_ANA	n_or_n_F	US_Invert	DS_Invert	Form_Loss	pBlockage	Width_or_Dia	Height_or_WF	Number_of_Height_Cont	Width_Cont	Entry_Loss	Exit_Loss		
1	2	0	15	0.015	2.1	2.1	0	0	0.9	1.4	1	0.6	1.0	0.5	1	
2	1	0	20	0.015	5.0	4.3	0	0	1.8	0.0	1	0.6	1.0	0.5	1	
3	1	0	15	0.015	3.0	2.9	0	0	0.6	0.0	1	0.6	1.0	0.5	1	
4	2	0	16	0.015	3.0	2.8	0	0	2.4	1.4	6	0.6	1.0	0.5	1	
5	2	0	12	0.015	1.1	1.0	0	0	2.4	1.4	3	0.6	1.0	0.5	1	
6	2	0	12	0.015	1.1	1.1	0	0	2.4	1.4	3	0.6	1.0	0.5	1	

Figure 4-17 1D Culvert Example

Table 4-6 Culvert File Inputs

Header Column	Description
ID	Culvert identifier. Links to “culvert file” command line.
Type	1 = circular, 2 = rectangular, 4 = gated circular (unidirectional), 5 = gated rectangular (unidirectional).
Ignore	If = 1 culvert is ignored (Default = 0).
UCS	Currently not used.
Len_or_ANA	Culvert length (m or ft – see units).
n_or_n_F	Friction (Mannings).
US_Invert	Upstream invert level (relative to model datum: m or ft – see units).
DS_Invert	Downstream invert level (relative to model datum: m or ft – see units).
Form_Loss	Form loss coefficient; an additional dynamic head loss coefficient applied when culvert flow is not critical at the inlet.
pBlockage	% blockage (for 10%, enter 10). For rectangular culverts, the culvert width is reduced by the % Blockage, while for circular culverts the pipe diameter is reduced by the square root of the % Blockage. (Default = 0).
Inlet_Type	Currently not used.
Conn_2D	Currently not used.
Conn_No	Currently not used.
Width_or_Dia	Width for rectangular culverts or diameter for circular culverts (m or ft – see units).
Height_or_WF	Height for rectangular culverts (m or ft – see units).
Number_of	Number of culvert barrels.
Height_Cont	Height contraction coefficient for orifice flow at the inlet. Recommended values, 0.6 for square edged entrances to 0.8 for rounded edges. Not used for unsubmerged inlet flow conditions or outlet controlled flow regimes. Not used for C channels.
Width_Cont	The width contraction coefficient for inlet-controlled flow. Usually 0.9 for sharp edges to 1.0 for rounded edges for R culverts. Normally set to 1.0 for C culverts. If value exceeds 1.0 or is less than or equal to zero, it is set to 1.0. Not used for outlet controlled flow regimes.
Entry_Loss	The entry loss coefficient for outlet controlled flow (recommended value of 0.5).
Exit_Loss	The exit loss coefficient for outlet controlled flow (recommended value of 1.0).

4.4.3 Weirs

Weir flow can be specified within the model in a variety of ways depending on the intended application. In some situations, using the SWE to calculate weir flow (just letting TUFLOW FV simulate weir flow without any direct specification of a weir structures) may be entirely acceptable. Where model resolution is insufficient (the structure width is represented by less than 5 cells), manual specification of the weir is recommended.

When manually specified, TUFLOW FV uses the standard weir equation (shown below), where the coefficient C and crest level P are user inputs. TUFLOW automatically calculates the weir width b based on the cell face across which the weir is being specified. The following sections summarise the available weir options.

$$Q = C b h_1^{2/3}$$

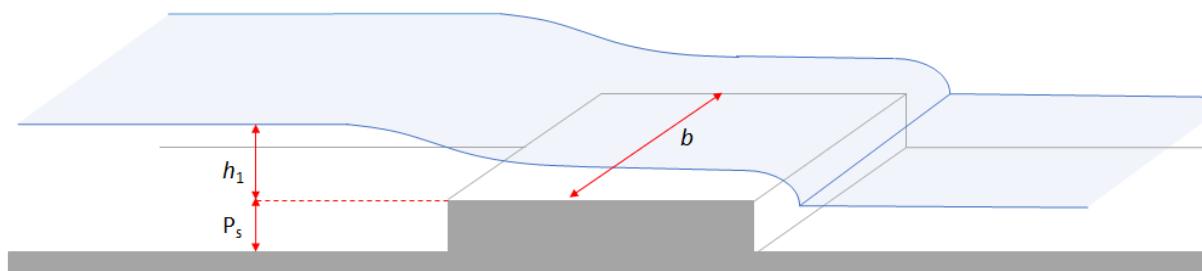


Figure 4-18 Weir Flow

4.4.3.1 Control Structure Options

A variety of control structure options are available (see [flux function](#)). These weir commands are applied at cell faces within the model at specified nodestring locations. For all of these options, using default values for C provides an exact solution to the broad crested weir equation. Two fixed and variable weir options are available, summarised below.

Fixed Weir Options: used to model flow control structures or levees.

- 1) Weir: A weir structure with a fixed crest level.
- 2) Weir_dz: A weir structure with a crest level (dz) above the existing cell elevation.

[Properties](#) input are required both of the fixed weir options. Example inputs are shown below in Figure 4-19.

```

Structure == Nodestring, 5           !Structure location (NS 5)
  Flux function == Weir             !Weir flag
  Properties == 9.5,1.6            !Weir elevation(mAHD), weir coeffecient
End Structure

Structure == Nodestring, 6           !Structure location (NS 6)
  Flux function == Weir_dz         !Weir flag
  Properties == 2.0,1.6            !Weir elevation above cell elevation(m), weir coeffecient
End Structure
  
```

Figure 4-19 Fixed Elevation Weir Example

Variable Weir Options: used to assess levee breach scenarios.

- 1) Weir_adjust: A weir structure with a time varying adjustable crest level relative to the model datum.
- 2) Weir_dz_adjust: A weir structure with a time varying adjustable crest level (dz) above the existing cell elevation.

Control types are used to specify how the weir elevation should be varied, either by time series from a trigger location/water level from somewhere within the model domain, or from the start of the model simulation. Examples of both input types are shown below in Figure 4-19.

<pre> Structure == Nodestring, 5 !Structure location (NS 5) Flux function == Weir_dz_adjust !Variable weir flag Properties == 9.5,1.6 !Elevation(m), Coeffecient Control == trigger !Control option Sample point == 125332.0,542533.0 !Sample location (x,y) Sample dt == 0.0166 !Smpale time interval (hr) Trigger value == 9.7 !Trigger value (mAHD) Control file == dz_adjustable_weir.csv !Time varying levee crest End Structure </pre>	<pre> Structure location (NS 5) !Variable weir flag !Elevation(m), Coeffecient !Control option !Sample location (x,y) !Smpale time interval (hr) !Trigger value (mAHD) !Time varying levee crest </pre>	<table border="1" style="margin-top: 20px;"> <thead> <tr> <th colspan="2"><i>dz_adjustable_weir.csv</i></th> </tr> <tr> <th>Time</th> <th>dz</th> </tr> </thead> <tbody> <tr> <td>0.0</td> <td>9.7</td> </tr> <tr> <td>2.0</td> <td>9.2</td> </tr> <tr> <td>4.0</td> <td>8.7</td> </tr> <tr> <td>6.0</td> <td>8.2</td> </tr> </tbody> </table>	<i>dz_adjustable_weir.csv</i>		Time	dz	0.0	9.7	2.0	9.2	4.0	8.7	6.0	8.2
<i>dz_adjustable_weir.csv</i>														
Time	dz													
0.0	9.7													
2.0	9.2													
4.0	8.7													
6.0	8.2													
<pre> Structure == Nodestring, 6 !Structure location (NS 6) Flux function == Weir_adjust !Variable weir flag Properties == 2.0,1.6 !Elevation(m), Coeffecient Control == time series !Control option Control file == adjustable_weir.csv !Time varying levee crest End Structure </pre>	<pre> Structure location (NS 6) !Variable weir flag !Elevation(m), Coeffecient !Control option !Time varying levee crest </pre>	<table border="1" style="margin-top: 20px;"> <thead> <tr> <th colspan="2"><i>adjustable_weir.csv</i></th> </tr> <tr> <th>Time</th> <th>weir_crest</th> </tr> </thead> <tbody> <tr> <td>20.0</td> <td>2.0</td> </tr> <tr> <td>22.0</td> <td>1.5</td> </tr> </tbody> </table>	<i>adjustable_weir.csv</i>		Time	weir_crest	20.0	2.0	22.0	1.5				
<i>adjustable_weir.csv</i>														
Time	weir_crest													
20.0	2.0													
22.0	1.5													

Figure 4-20 Variable Elevation Weir Example

4.4.3.2 Auto Weir

The auto weir function identifies all cell faces (not nodestrings) in the model domain that are elevated above the adjacent cells. These cell faces are then assigned a weir flow condition. This feature ensures that critical floodplain hydraulic controls such as roads and perched riverbanks regulate floodplain flows accurately. This command is recommended for catchment flood modelling. In combination with this command, the geometry log files can be used to identify all locations within the model where this function has been assigned.

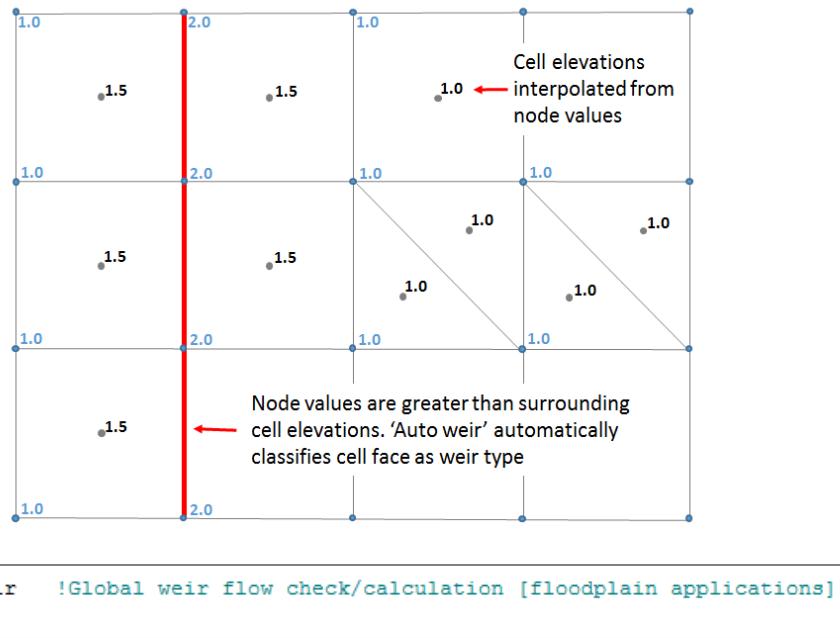


Figure 4-21 Auto Weir Example

4.4.4 Logic Controls

Logic controls adjust flow conditions through a structure according to a series of logical rules specified by the user. This is particularly useful for applications with adjustable structures, such as drop gates / sluices / pumps or for levee breach/failure assessments.

The commands required when using logic controls include

Control: Defines the type of control which is applied:

1. *Trigger*: The control will commence after the first exceedance of a specific trigger value. This command is typically applied during event scenario tests (such as levee breach scenarios)
2. *Timeseries*: The control will commence according to a defined time series. This command is used during assessments where regulation of flow at a structure is either regular or known (i.e. recorded historic flow release data).
3. *Sample_Rule*: The control will be tied to a sampling location (e.g. water level monitoring) within the model domain. This command is used when flow at a structure is self-regulated (automated) based on a recorded water level.
4. *Target_Rule*: The control will be tied to sampling within the model domain. The flow at the structure is determined based on the water level difference relative to a target level at the sampling location.
5. Fully Open: No logic controls are applied.

Control Parameter: Defines the parameter that will be controlled:

1. Bed Elevation Logic Controls:

- a. *Weir_Crest*: The weir crest level in absolute terms (i.e. from 0m datum).
 - b. *Zb*: The change in bed level in absolute terms (i.e. from 0m datum).
 - c. *Dzb*: The change in bed elevation relative to the existing bed level (difference in bed elevation).
2. Flow Logic Controls:

- a. *Fraction_Open*: Fraction of a structure which is open to flow.
- b. *Min_Flow*: Specifies the minimum flow through a structure. Some hydraulic controls include low flow ‘environment’ outlets which operate irrespective of the conditions at the ‘main’ structure. This command allows the user to represent these ‘environmental’ outlets with the model.

Sample Point Commands: Define the location, model update time and trigger values for the specified sample point.

Target File (if ‘Control == Target_Rule’): Defines the target value at the sample location.

Control File: Defines the change in parameter value.

The following page provides examples for three logic control scenarios.

Example 1: Levee Breach

Levee failure after water level exceeds 24.0mAHD. Period of failure occurs over 10 hours. Extent of breach is limited by the area defined by the polygon file, “breach_poly.csv”.

```
Structure == Cell           !Structure option,
Cell function == zb_adjust !Adjustable bed elevations
Polygon file == ..\geo\breach_poly.csv !Defining where breach occurs
Control == trigger          !Structure control type
Sample point == 622943,4334061 !Sample location (x,y)
Sample type == WL            !Sampling parameter
Sample dt == 0.05             !Check WL
Trigger value == 24.0          !When to initiate breach
Control file == ..\bcs\Breach_Val.csv !Levee breach timeseries
End Structure
```

Breach_Val.csv	
Time	Zb
0.0	24.0
2.0	23.0
4.0	22.0
6.0	21.0
8.0	20.0
10.0	19.0

Example 2: Floodgate Control - Timeseries

Historic event modelling. Floodgate operation timeseries specification.

```
Structure == Nodestring, 2 !Structure option, location (NS 2)
Flux function == matrix   !HQh matrix
Flux file == hQh.csv      !HQh matrix file
Control == timeseries     !Structure fraction open timeseries
Control parameter == fraction_open
Control update dt == 1.    !HR
Control file == ..\bcs\tseries.csv !csv: "TIME", "FRACTION_OPEN"
Max opening increment == 0.1 !Max change frac open (per update dt)
End Structure
```

tseries.csv	
Time	Fraction_Open
0.0	0.0
2.0	0.0
4.0	0.2
6.0	0.4
8.0	1.0
10.0	1.0

Example 3: Floodgate Control - Sample Control

Floodgate operation defined using sample control (i.e. the gate operation is based on water level monitoring at sample location).

```
Structure == Nodestring, 3           !Structure option, location (NS 3)
Flux function == matrix             !HQh matrix
Flux file == hQh.csv                !HQh matrix file
Control == sample_rule              !Structure control type
Control parameter == fraction_open
Control update dt == 1.              !HR
Control file == control_rule.csv   !csv: "SAMPLE_VALUE", "FRACTION_OPEN"
Sample type == WL                  !Sampling parameter (water level)
Sample point == 364352., 8139574.    !Sampling location (x,y)
Sample dt == 0.1                   !HR
Max opening increment == 0.1        !Max change frac open (per update dt)
End Structure
```

ctrl_rule.csv	
Sample_Value	Fraction_Open
10.0	0.0
10.2	0.1
10.4	0.3
10.6	0.5
10.8	0.8
11.0	1.0
12.0	0.8
13.0	0.5

Figure 4-22 Logic Control Examples

5 Model Output

TUFLOW FV offers various model output options. The output type, parameter(s) and time interval are specified using [output block](#) commands. Common model output types include:

- Point timeseries in text format (2D or vertically averaged 3D)
- Mapped output in SMS Data File (2D or vertically averaged 3D) or netCDF (2D, vertically averaged 3D or full 3D) formats
- Mapped statistical output in SMS Data File (2D or vertically averaged 3D) or netCDF (2D, vertically averaged 3D or full 3D) formats
- Vertical profile timeseries output at point locations in netCDF format (full 3D)

The types of output will typically depend on the TUFLOW FV hydrodynamic calculation mode (2D or 3D), the available model calibration/validation data, the objectives of the modelling exercise and the modeller's preferred method of communicating assessment results.

5.1 2D Model Output

TUFLOW FV 2D model output can be generated in the following formats:

- Timeseries at a point location (or multiple point locations) defined in an [output points file](#);
- Mapped output in SMS Data File format; or
- Mapped output in netCDF format (see Section 5.3).

The 2D output options are described below.

5.1.1 2D Points Output

The TUFLOW FV 2D ‘points’ option provides model parameter timeseries at point locations, each defined by a x,y coordinate specified within an [output points file](#). The output points file is a comma separated variable (.csv) file with headers X, Y and ID (optional) and contains the coordinates for the point, or list of points, of interest. This type of model output is often used to compare with timeseries data recorded at a fixed location as part of a model calibration/validation exercise. Example points [output block](#) commands and are provided in Figure 5-1.

```

output == points
  output points file == ..\geo\points\output_points.csv      !Output points file
  output parameters == H,V                                     !Output parameters
  output interval == 900.                                     !Output interval in seconds
end output

```

output_points.csv		
X	Y	ID
145.8104	-16.7804	ADCP 1
145.8040	-16.8089	ADCP 2
145.8162	-16.8561	ADCP 3

Figure 5-1 Points Output Commands and Output Points File Contents Example

The TUFLOW FV 2D points output file is comma separated variable format (*.POINTS.csv) that can be opened and viewed in a text editor or spreadsheet software such as Microsoft Excel. The file contains the timeseries of model parameters at the point locations defined in the [output points file](#). Following the example in Figure 5-1, a sample of the corresponding *.POINTS.csv output file is shown in Figure 5-2.

TIME	PT1_H [m]	PT1_VX [m s^-1]	PT1_VY [m s^-1]	PT2_H [m]	PT2_VX [m s^-1]	PT2_VY [m s^-1]	PT3_H [m]	PT3_VX [m s^-1]	PT3_VY [m s^-1]
01/03/2013 01:00	0.3825	0.0668	0.0508	0.3799	0.0705	0.0610	0.4117	0.0561	0.0792
01/03/2013 01:15	0.3990	0.0420	0.0224	0.3988	0.0335	0.0219	0.3717	0.0692	0.0981
01/03/2013 01:30	0.1201	0.1398	0.1204	0.1277	0.1405	0.1324	0.2078	0.0934	0.2013
01/03/2013 01:45	0.1230	0.1529	0.0981	0.1187	0.1543	0.1061	0.0721	0.2355	0.2531
01/03/2013 02:00	-0.1350	0.1030	0.0967	-0.1457	0.0896	0.1092	-0.1312	0.1002	0.3421
01/03/2013 02:15	-0.1840	0.1360	0.0968	-0.1719	0.1313	0.1096	-0.2174	0.2300	0.3492
01/03/2013 02:30	-0.3811	0.1638	0.1342	-0.3852	0.1548	0.1449	-0.3587	0.1630	0.3688

Figure 5-2 TUFLOW FV 2D Points Output File Example

5.1.2 2D SMS Data File Output

The TUFLOW FV ‘datv’ option provides output on the mesh nodes at the specified output time interval. The output format is the SMS Data File binary format (.dat) and can be viewed using the SMS Generic Mesh Module. A separate output file is created for each specified model parameter (e.g. *_H.dat, *_V.dat, *_ZB.dat). Example datv [output block](#) commands and are provided in Figure 5-3.

```

output == datv
  output parameters == H,V,ZB      !Output parameters
  output interval == 900.          !Output interval in seconds
end output

```

Figure 5-3 SMS Mapped Output Commands Example

To view the output, the model mesh (.2dm) must be first opened using the SMS Generic Mesh Module, followed by the 2D mapped output file(s). The screenshot in Figure 5-4 provides an example of TUFLOW FV ‘datv’ output opened in the SMS Generic Mesh Module environment.

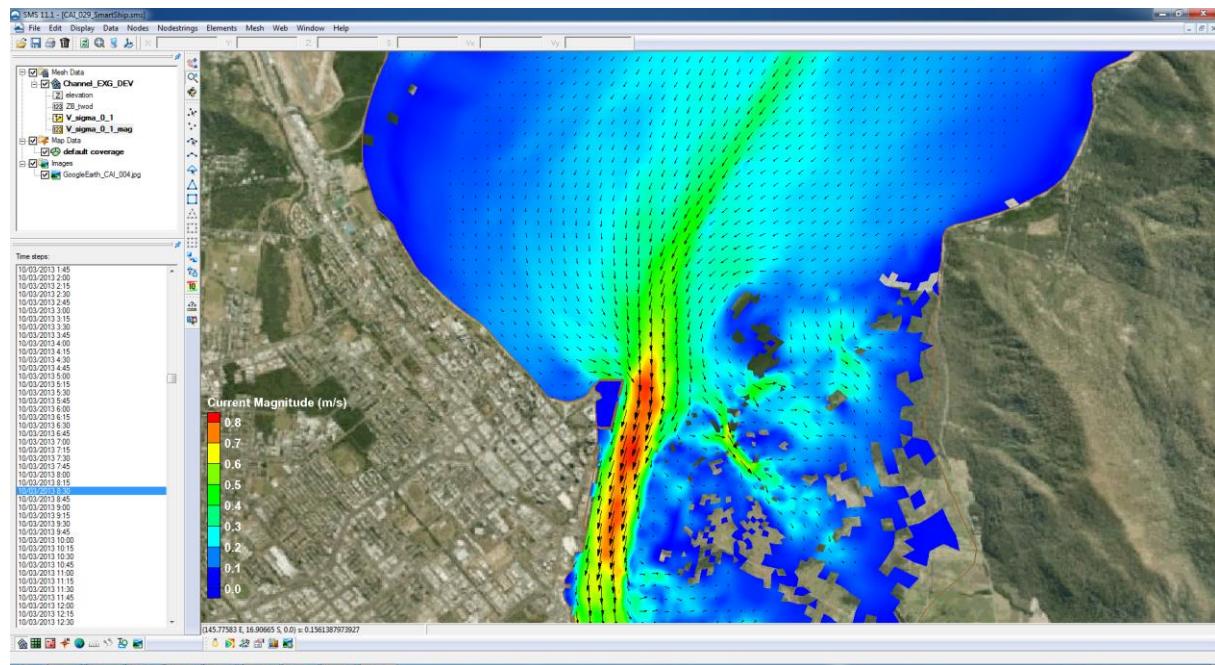


Figure 5-4 TUFLOW FV Mapped Current Velocity Output in the SMS Generic Mesh Module Environment

Various post-processing tasks can be undertaken using SMS, including data extraction and creating animations. The TUFLOW FV Wiki and Aquaveo SMS website provide post-processing examples and tips:

- TUFLOW FV Wiki: <http://fvwiki.tuflow.com>
- Aquaveo SMS website: <http://www.aquaveo.com/software/sms-learning-tutorials>

5.2 3D Model Output Vertically Averaged

TUFLOW FV 3D offers various vertically averaged output options intended to simplify 3D post-processing tasks and allow output based on 3D calculations to be viewed using the SMS Generic Mesh Module. TUFLOW FV 3D vertically averaged model output can be generated in the following formats:

- Timeseries at a point location (or multiple point locations) defined in an [output points file](#);
- Mapped output in SMS Data File format; or
- Mapped output in netCDF format (see Section 5.3).

The 3D vertical averaging options and commands are described below.

5.2.1 3D Vertical Averaging Options

The following vertical averaging options are available when using TUFLOW FV 3D:

- depth-all – averaging over entire water column
- depth-range – averaging between specified minimum and maximum absolute depths measured downward from water surface
- height-range – averaging between specified minimum and maximum absolute heights measured upward from the bed
- elevation-range – averaging between specified minimum and maximum elevations relative to model vertical datum
- sigma-range – averaging between specified decimal fraction of the water column where 0 is the bed and 1 is the water surface
- layer-range-top – averaging between layers referenced from the water surface (i.e. surface layer is 1, positive downwards)
- layer-range-bot – averaging between layers referenced from the bed (i.e. bottom layer is 1, positive upwards)

The depth-all option simply averages the 3D output over the entire water column, giving 2D depth averaged output based on the 3D calculations. The other options allow the modeller to specify a range of the water column to vertically average over. Some example commands are provided below with conceptual cross-section illustrations used to assist the descriptions. Some key tips when using the vertical averaging output options include:

- The use of the [suffix](#) command to add a clear identifier to the output filename
- Multiple model [output parameters](#) may be specified in a single [output block](#)
- Separate [output block](#) commands must be used for each vertical averaging specification

5.2.1.1 3D Depth-All

The ‘depth-all’ command vertically averages over the entire water column. Example ‘depth-all’ output block commands and a conceptual illustration are provided in

```

output == points
  output points file == ..\geo\points\output_points.csv
  vertical averaging == depth-all
  suffix == depth_all
  output parameters == V
  output interval == 900
end output

output == datv
  vertical averaging == depth-all
  suffix == depth_all
  output parameters == V
  output interval == 900
end output

```

!Output type
 !Output points file
 !Vertical averaging type, entire water column
 !Output file identifier suffix
 !Output parameters
 !Output interval in seconds

!Output type (datv or netcdf)
 !Vertical averaging type, entire water column
 !Output file identifier suffix
 !Output parameters
 !Output interval in seconds

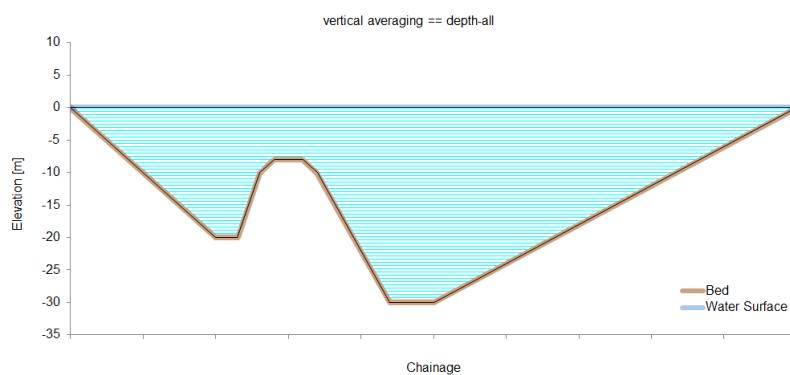


Figure 5-5 3D Depth-All Output Example Commands and Conceptual Illustration

5.2.1.2 3D Depth-Range

The ‘depth-range’ command vertically averages between specified minimum and maximum absolute depths measured downward from water surface. Example ‘depth-range’ output block commands and a conceptual illustration are provided in Figure 5-6.

```

output == points
  output points file == ..\geo\points\output_points.csv
  vertical averaging == depth-range, 6, 12
  suffix == depth_6_12
  output parameters == V
  output interval == 900
end output

output == datv
  vertical averaging == depth-range, 6, 12
  suffix == depth_6_12
  output parameters == V
  output interval == 900
end output

```

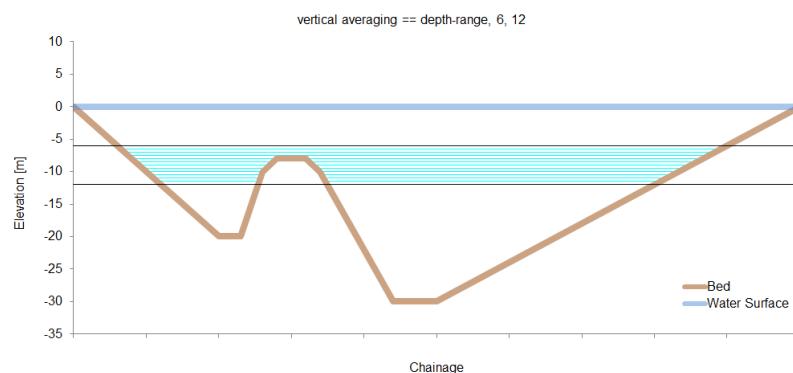


Figure 5-6 3D Depth-Range Output Example Commands and Conceptual Illustration

5.2.1.3 3D Height-Range

The ‘height-range’ command vertically averages between specified minimum and maximum absolute heights measured upward from the bed. Example ‘height-range’ output block commands and a conceptual illustration are provided in Figure 5-7.

```

output == points
  output points file == ..\geo\points\output_points.csv
  vertical averaging == height-range, 0, 6
  suffix == height_0_6
  output parameters == V
  output interval == 900
end output

output == datv
  vertical averaging == height-range, 0, 6
  suffix == height_0_6
  output parameters == V
  output interval == 900
end output

```

!Output type
!Output points file
!Vertical averaging type, absolute height range minimum and maximum
!Output file identifier suffix
!Output parameters
!Output interval in seconds

!Output type (datv or netcdf)
!Vertical averaging type, absolute height range minimum and maximum
!Output file identifier suffix
!Output parameters
!Output interval in seconds

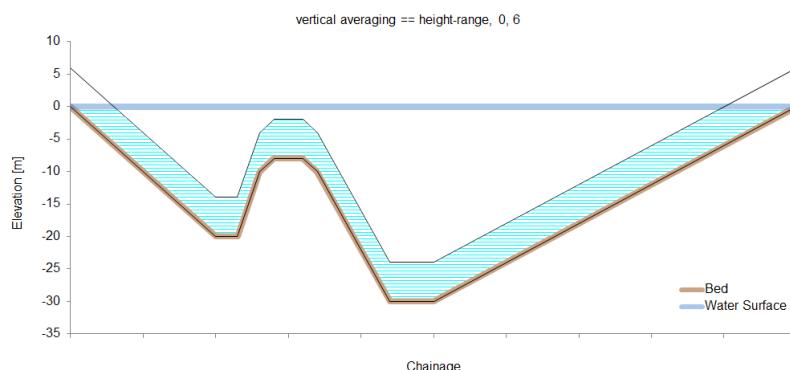


Figure 5-7 3D Height-Range Output Example Commands and Conceptual Illustration

5.2.1.4 3D Elevation-Range

The ‘elevation-range’ command vertically averages between specified minimum and maximum elevations relative to model vertical datum. Example ‘elevation-range’ output block commands and a conceptual illustration are provided in Figure 5-8.

```

output == points
  output points file == ..\geo\points\output_points.csv
  vertical averaging == elevation-range, -16, -10
  suffix == elevation_-16_-10
  output parameters == V
  output interval == 900
end output

output == datv
  vertical averaging == elevation-range, -16, -10
  suffix == elevation_-16_-10
  output parameters == V
  output interval == 900
end output

```

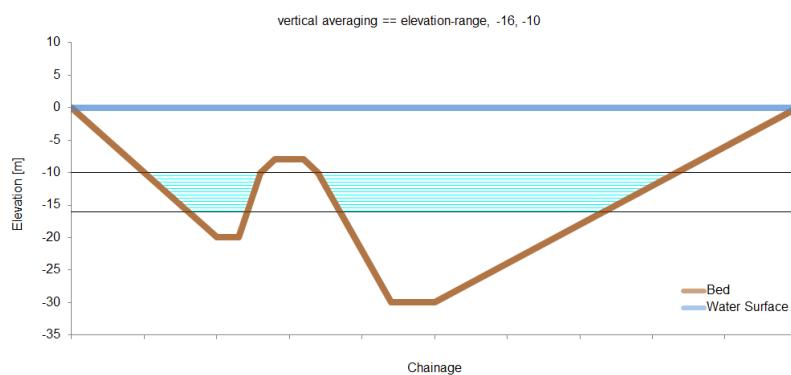


Figure 5-8 3D Elevation-Range Output Example Commands and Conceptual Illustration

5.2.1.5 3D Sigma-Range

The ‘sigma-range’ command vertically averages between specified decimal fractions of the water column where 0 is the bed and 1 is the water surface. Example ‘sigma-range’ output block commands and a conceptual illustration are provided in Figure 5-9.

```

output == points          !Output type
  output points file == ..\geo\points\output_points.csv !Output points file
  vertical averaging == sigma-range, 0.25, 0.75 !Vertical averaging type, sigma range minimum and maximum
  suffix == sigma_0.25_0.75 !Output file identifier suffix
  output parameters == V !Output parameters
  output interval == 900 !Output interval in seconds
end output

output == datv           !Output type (datv or netcdf)
  vertical averaging == sigma-range, 0.25, 0.75 !Vertical averaging type, sigma range minimum and maximum
  suffix == sigma_0.25_0.75 !Output file identifier suffix
  output parameters == V !Output parameters
  output interval == 900 !Output interval in seconds
end output

```

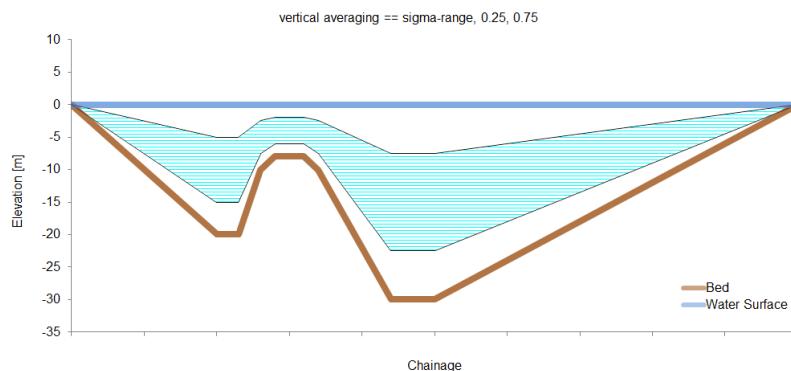


Figure 5-9 3D Sigma-Range Output Example Commands and Conceptual Illustration

5.2.1.6 3D Layer-Range-Top

The ‘layer-range-top’ command vertically averages between layers referenced from the water surface (i.e. surface layer is 1, positive downwards). Example ‘layer-range-top’ output block commands and a conceptual illustration are provided in Figure 5-10. Single layer output can be obtained by specifying an equal minimum and maximum.

```

output == points
  output points file == ..\geo\points\output_points.csv
  vertical averaging == layer-range-top, 2, 4
  suffix == layer_top_2_4
  output parameters == V
  output interval == 900
end output

output == datv
  vertical averaging == layer-range-top, 2, 4
  suffix == layer_top_2_4
  output parameters == V
  output interval == 900
end output
  
```

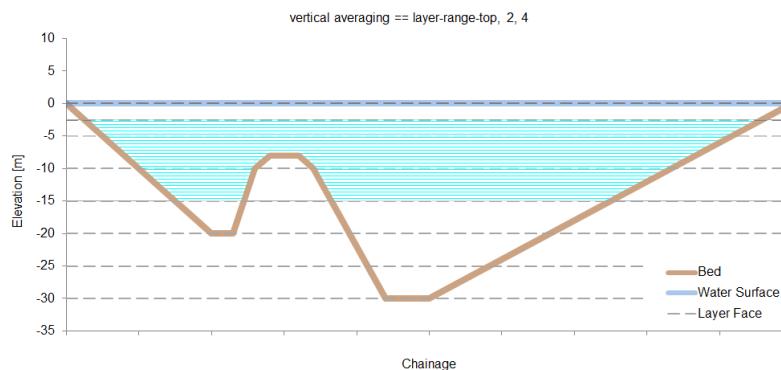


Figure 5-10 3D Layer-Range-Top Output Example Commands and Conceptual Illustration

5.2.1.7 3D Layer-Range-Bot

The ‘layer-range-bot’ command vertically averages between layers referenced from the water surface (i.e. bottom layer is 1, positive upwards). Example ‘layer-range-bot’ output block commands and a conceptual illustration are provided in Figure 5-11. Single layer output can be obtained by specifying an equal minimum and maximum.

```

output == points
  output points file == ..\geo\points\output_points.csv
  vertical averaging == layer-range-bot, 2, 4
  suffix == layer_bot_2_4
  output parameters == V
  output interval == 900
end output

output == datv
  vertical averaging == layer-range-bot, 2, 4
  suffix == layer_bot_2_4
  output parameters == V
  output interval == 900
end output

```

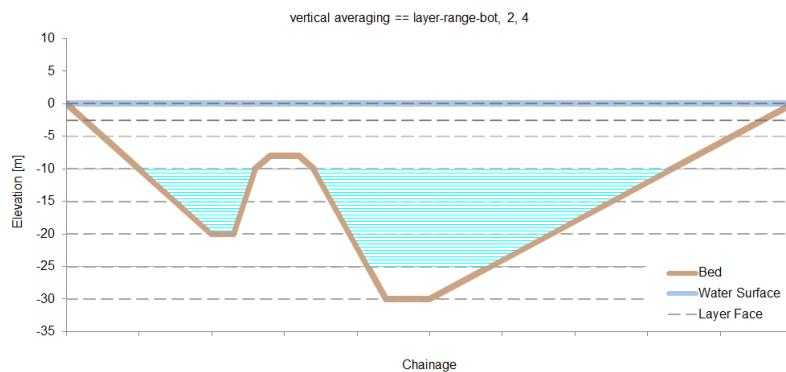


Figure 5-11 3D Layer-Range-Bot Output Example Commands and Conceptual Illustration

5.3 netCDF 2D and 3D Model Output

The netCDF (network Common Data Form) software was developed at the Unidata Program Centre in Boulder, Colorado. It is an interface for array-oriented data access and a library that provides implementation of the interface. The netCDF library also defines a machine-independent format for representing scientific data. Together, the interface, library and format support the creation, access and sharing of scientific data (Unidata, 2014).

TUFLOW FV netCDF output adopts the netCDF-4/HDF5 format. The output file is self-describing and contains information regarding the model geometry (2D or 3D) together with the mapped output at the specified time interval. The netCDF format offers the following advantages:

- Storage of the “cell-centred” output as calculated by TUFLOW FV (i.e. no interpolation to the cell nodes as required by the SMS Data File Format, refer Section 4.2.2);
- The files are machine-independent and can be viewed using any numerical analysis package with a netCDF library interface, including MATLAB, R, GNU Octave or Python NumPy;
- An ability to store full 3D output in a single compressed file format; and
- An ability to view the vertical distribution of modelled parameters.

MATLAB is typically used by BMT to view TUFLOW FV netCDF output, extract data and generate animations. Some commonly used post-processing functions include:

- Sheet plots of vertically averaged, cell-centred model output (example in Figure 5-12);
- Curtain plots (longitudinal or cross-sectional) showing the vertical distribution of model output (examples in Figure 5-13 and Figure 5-14); and
- Conversion of TUFLOW FV netCDF output to vertically averaged SMS Data File Format.

BMT is able to provide customised TUFLOW FV netCDF tools to existing MATLAB users. Alternatively, some TUFLOW FV netCDF MATLAB functions can be complied and used on Windows machines with the freely-available MATLAB Compiler Runtime (MCR) installed. BMT can be contacted via TUFLOW support for further information on post-processing TUFLOW FV netCDF output: support@tuflow.com.

The dimensions, variable definitions and attributes of a TUFLOW FV netCDF output file are provided Appendix C. This information is intended to assist advanced users wishing to develop functions and scripts to post-process and/or view TUFLOW FV netCDF output.

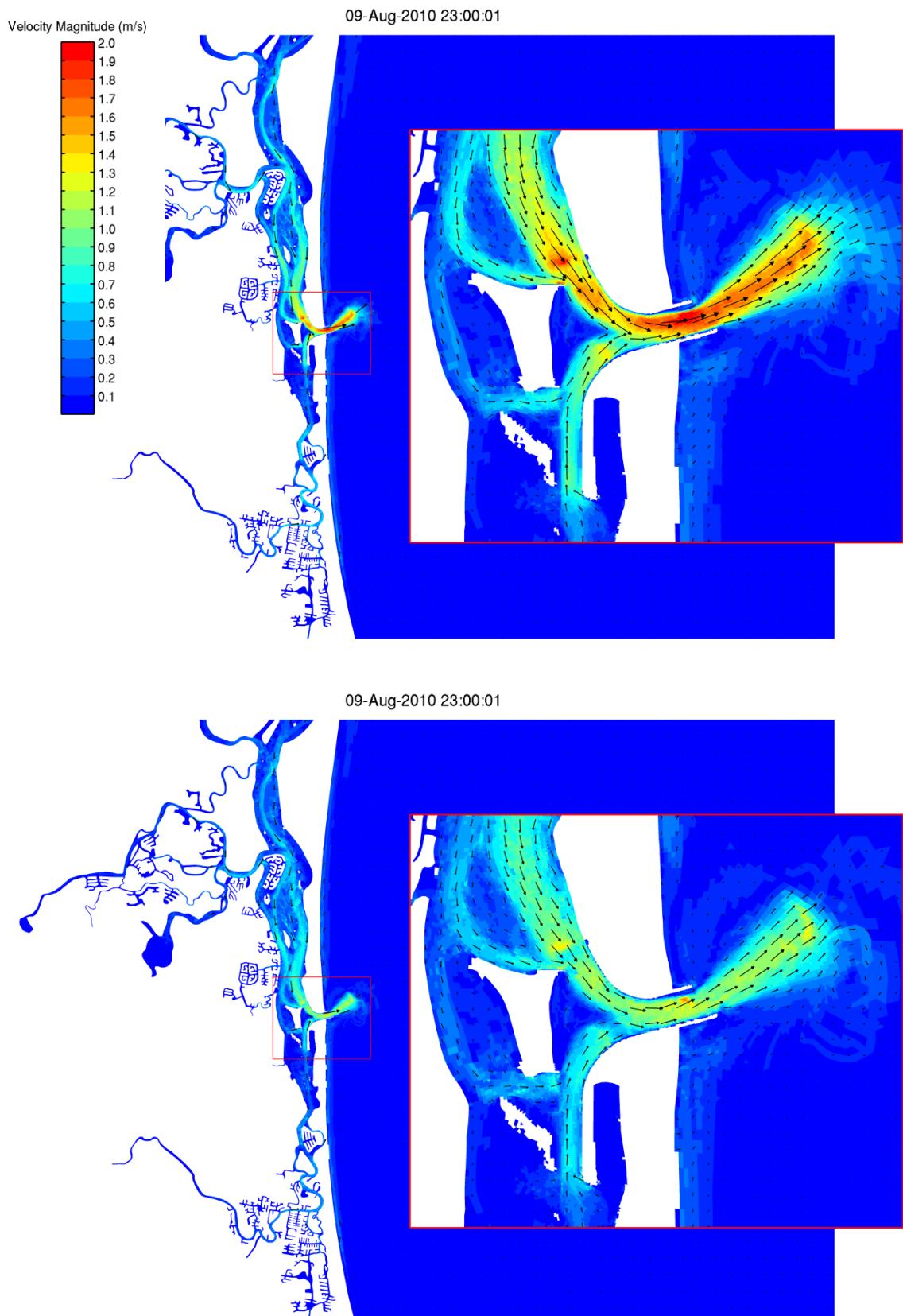


Figure 5-12 TUFLOW FV Sheet Plot with Zoom Example: Velocity Magnitude Top 50% Water Column (top); Velocity Magnitude Bottom 50% Water Column (bottom)

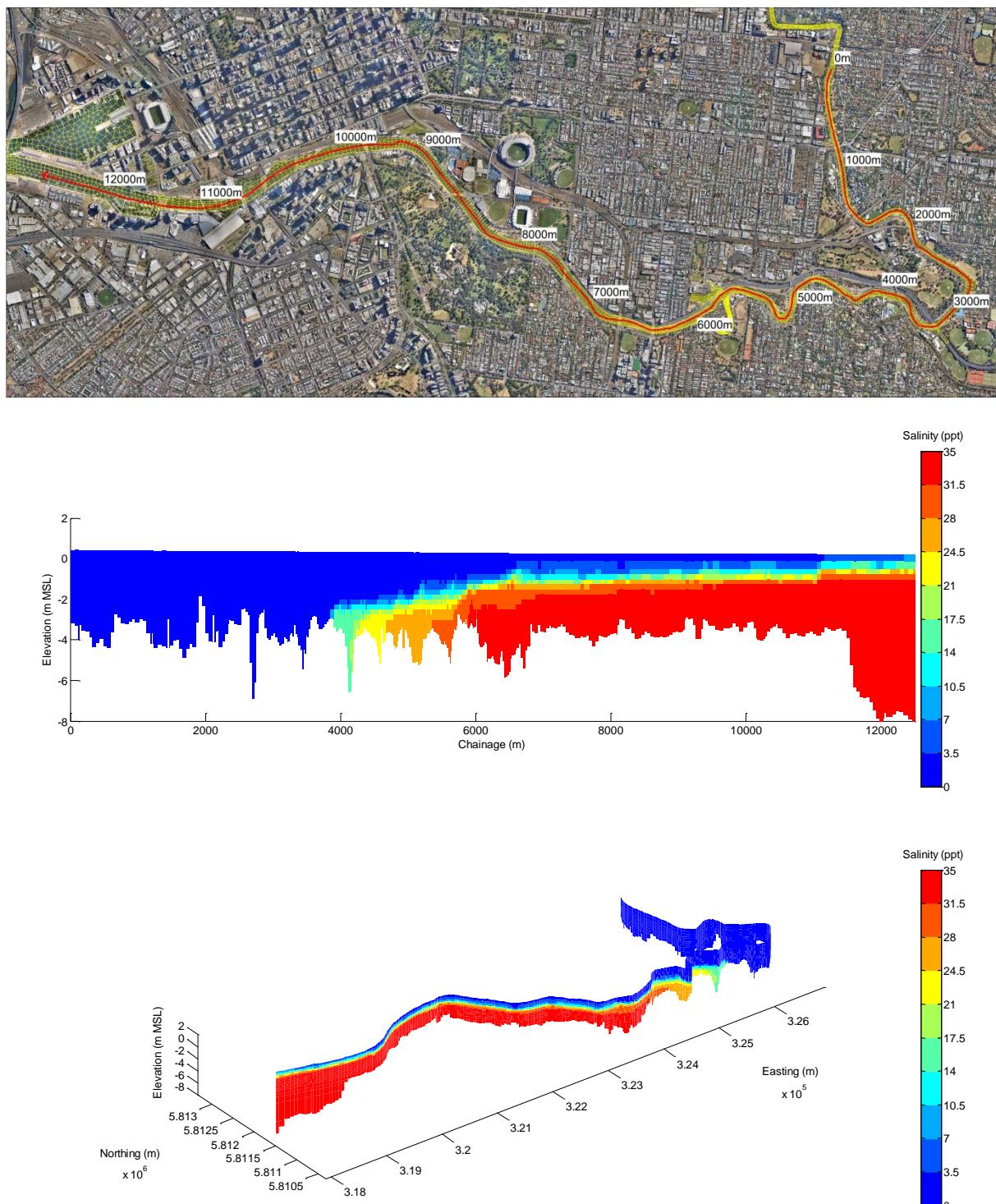


Figure 5-13 TUFLOW FV Salinity Vertical Distribution: Model Mesh and Curtin Polyline (top); Salinity Curtin Plotted with Polyline Chainage; Salinity Curtain Plotted with Polyline Coordinates (bottom)

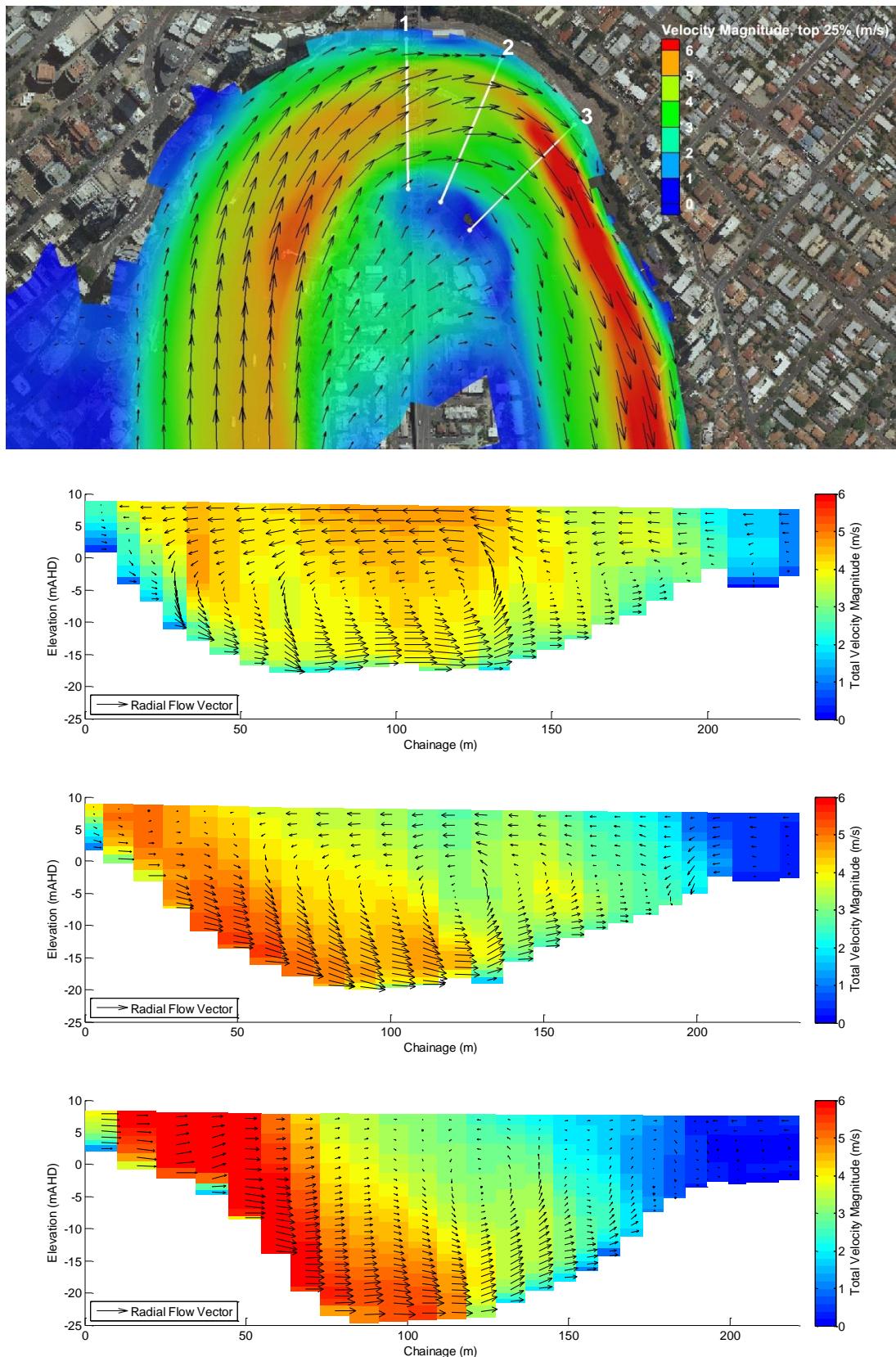


Figure 5-14 TUFLOW FV Velocity Vertical Distribution: River Bend Flood Flow and Cross-Section Locations (top); Total Velocity Magnitude (contours) with Radial Flow Vectors Cross-Sections

5.4 Statistical Output

The maximum and/or minimum values of the specified model parameters can be tracked during a TUFLOW FV simulation using the ‘output statistics’ command. Statistical output can be generated for the following output formats:

- Mapped output in SMS Data File format; or
- Mapped output in netCDF format (see Section 5.3).

Example use of the output statistics command within an [output block](#) is provided in Figure 5-15. The ‘output statistics dt’ must be specified in addition to the ‘output interval’. In the example below, SMS Data File format output files would be created at 900 second intervals with the maximum tracked at a 1 second interval.

```

output == datv          !Output type (datv or netcdf)
output parameters == H,V !Output parameters
output interval == 900  !Output interval in seconds
output statistics == max !Output statistics type, max or min
output statistics dt == 1 !Output statistics dt in seconds
end output

```

Figure 5-15 Statistical Output Example Commands

5.5 Profile Output

For 3D simulations, profile output at point locations may be requested and output to a netCDF file. Example use of the output profile command within an [output block](#) is provided in Figure 5-16.

```

output == profiles        !Output type
output points file == ..\geo\points\profile_points_000.csv !Output points file
output parameters == H, V !Output parameters
output interval == 900   !Output interval
end output

```

Figure 5-16 Profile Output Example Commands

5.6 Check Files

Check files are produced so that modellers and reviewers can readily check that the constructed model is as intended. Advanced models draw upon a wide variety of data sources. The check files represent the final data set after all data inputs, allowing the model construction to be viewed in its final form. The check files are typically in text format and include:

- The simulation .log file which is automatically generated at the beginning of (and continuously written to during) a TUFLOW FV simulation.

- The simulation timestep files which contain the minimum and mean timestep required for calculation of the external (free-surface) and internal (advective) terms within each model cell are automatically written at the end of a TUFLOW FV simulation. This information can be used to identify model cell(s) constraining the simulation timestep. A model ‘timestep review’ example is provided on the TUFLOW FV Wiki:

http://fvwiki.tuflow.com/index.php?title=A_Model_Runs_Slow

- The simulation ‘mass’ file output used to check the volume of fluid and, where applicable, other simulated quantities within the model domain. The command to output the mass file is simply:

```
output == mass
  output interval == 900           !Output interval in seconds
end output
```

- The simulation flux file used to check the rates of fluid and, where applicable, other simulated quantities entering/exiting the model boundaries or crossing specified nodestrings within the model domain. The command to output the flux file is simply:

```
output == flux
  output interval == 900           !Output interval in seconds
end output
```

5.7 Output Types and Parameters

The full range of output types and parameters are summarised in Table 5-1, Table 5-2 and Table 5-3. Example output block syntax is provided throughout this Chapter, in Figure 5-17 and in Appendix A. Additional examples are also available via the TUFLOW FV tutorial models on the TUFLOW FV Wiki: <http://fvwiki.tuflow.com>.

Table 5-1 Output Types

Output Format	Description	Parameters	Relevant Commands
Points	Timeseries results for specific point locations in a csv file format. This output requires an Output Points File . The points file specifies the output location x,y coordinates.	See Table 5-2 and Table 5-3	Output Points File Output Parameters Output Interval Output Statistics
Flux	Outputs flux across all model nodestrings.	N/A. This will output flow, salinity, temperature, sediment and scalar fluxes as required.	Output Interval Output Statistics
Mass	Outputs a comma separated variable file with mass output for the entire model domain. ⁹	N/A. This will output flow, salinity, temperature, sediment and scalar "mass" as required.	Output Interval
Dat	Sheet output at cell centroids in SMS .dat format. ¹⁰	See Table 5-2 and Table 5-3	Output Parameters Output Interval Output Statistics
Datv	Sheet output at cell vertices (nodes) in SMS .dat format. This is the required format to view results in SMS. ¹⁰	See Table 5-2 and Table 5-3	
Netcdf	Binary "self-describing" library of data output arrays and metadata at cell centroids in netcdf-4/HDF5 format.	See Table 5-2 and Table 5-3	
Netcdfv	Binary "self-describing" library of data output arrays and metadata at cell vertices (nodes) in netcdf-4/HDF5 format.	See Table 5-2 and Table 5-3	
Profile	Outputs the time history of the vertical profile at a point location	See Table 5-2 and Table 5-3	
Transport	A file containing the conserved variables stored on the TUFLOW FV mesh to be used in a subsequent advection-diffusion simulation.	NA	Output Interval

⁹ Note that this output format can be read into SMS as a scatter dataset and not as a data file attached to a 2dm geometry file (see "datv" below).

¹⁰ TUFLOW Utilities are available for post processing this result dataset type. The Utilities can be used to perform a range of post processing options and also data conversions so data can be imported into third party GIS packages (MapInfo, ArcGIS, QGIS): See the TUFLOW FV Wiki for more details http://fvwiki.tuflow.com/index.php?title=TUFLOW_FV_Utils

Table 5-2 Output Parameters (Basic)

Parameter	Description
D	Water depth (m)
FLOW	Flow (m ³ /s)
H	Water surface elevation (m)
Taub	Bed shear stress (N/m ²) (Hydrodynamic module)
Taus	Surface shear stress (N/m ²) (Hydrodynamic module)
V	Velocity vector (m/s)
Volume	Volume (m ³)
W	Vertical velocity (m/s)
ZB	Bed elevation (m)

Table 5-3 Output Parameters (Advanced)

Parameter	Description
Air_temp	Air temperature (degrees Celsius)
Bed_mass_total	Total bed mass (kg/m ²)
Bed_mass_Layer_#	Bed mass in layer # (kg/m ²)
Bedload	Bed load (g/m/s)
Bedload_TOTAL	Total Bed load (g/m/s)
Deposition_total	(g/m ² /s)
DZB	Bed elevation change (m)
Heat_content	(Degrees Celsius m ³)
LW_rad	Downward long wave radiation flux (W/m ²)
MSLP	Mean sea level pressure (hPa)
Netsedrate_total	(g/m ² /s)
Pickup_total	(g/m ² /s)
PRECIP	Precipitation rate (m/day)
Rel_hum	Relative humidity (%)
Rhow	Water density (kg/m3)
Sal	Salinity concentration (TBC)
Salt_flux	(psu m ³ /s)
Salt_mass	(psu m ³)
Sed_#	Suspended concentration of sediment fraction # (mg/L)
Sed_#_BED_MASS	Sediment bed mass of fraction # (kg)
Sed_#_FLUX	Suspended sediment flux of fraction # (10 ⁻³ kg/s)
Sed_#_MASS	Suspended sediment mass of fraction # (10 ⁻³ kg)
Sedload	Sediment load (g/m/s)
Sedload_TOTAL	Total Sediment load (g/m/s)

Parameter	Description
Susload	Suspended load (g/m/s)
Susload_TOTAL	Total Suspended load (g/m/s)
SW_rad	Downward short wave radiation flux (W/m ²)
Tauc	Current related effective bed shear stress component (N/m ²) (Sediment transport module)
Tauw	Wave related effective bed shear stress component (N/m ²) (Sediment transport module)
Taucw	Combined effective current/wave bed shear stress (N/m ²) (Sediment transport module)
Temp	Temperature (degrees Celsius)
Temp_flux	(degrees Celsius m ³ /s)
THICK	Total bed thickness (m)
Trace_#	Tracer concentration (units/m ³)
Trace_#_FLUX	Tracer flux (units m ³ /s)
Trace_#_MASS	Tracer mass (units)
TSS	Total suspended solids concentration (mg/L)
TURBZ	Output of vertical turbulence parameters, which includes: <ul style="list-style-type: none"> • TURBZ_TKE (m²/s²) • TURBZ_EPS (m²/s³) • TURBZ_L (m) • TURBZ_SPFSQ (/s²) • TURBZ_BVFSQ (/s²) • TURBZ_NUM (m²/s) • TURBZ_NUH (m²/s) • TURBZ_NUS (m²/s)
W10	10 m wind speed vector (m/s)
WQ_ALL	Output of water quality parameters
WQ_DIAG_ALL	Output of water quality parameters
Wvht	Wave height (m) - typically significant wave height
Wvper	Wave period (s) - typically peak wave period
Wvdir	Wave direction (degrees true coming from)
Wvstr	Wave stress vector (N/m ²)

```
!  
!OUTPUT COMMANDS  
  
output dir == ..\output  
  
output == points  
    output points file == ..\geo\Output_Pts.csv  
    output parameters == H,V,D  
    output interval == 900.  
end output  
  
output == flux  
    output interval == 900.  
end output  
  
output == mass  
    output interval == 1800.  
end output  
  
output == DATV  
    output parameters == H,V,D  
    output interval == 3600.  
end output  
  
output == netcdf  
    output parameters == H,V,D  
    output interval == 3600.  
end output
```

Figure 5-17 Example Output Block Commands

6 Installing and Running TUFLOW FV

6.1 Installing TUFLOW FV

As for all TUFLOW products, TUFLOW FV requires a hardware lock (or dongle) for licencing purposes. In addition to the dongle, TUFLOW FV requires dongle drivers be installed on your computer to run.

The model engine is TUFLOWFV.exe. It doesn't need to be installed, just placed in a folder on your computer (or on your network). There are also several dll files (dynamic link libraries) which are also placed in the same folder.

Software installation steps are broadly described below.

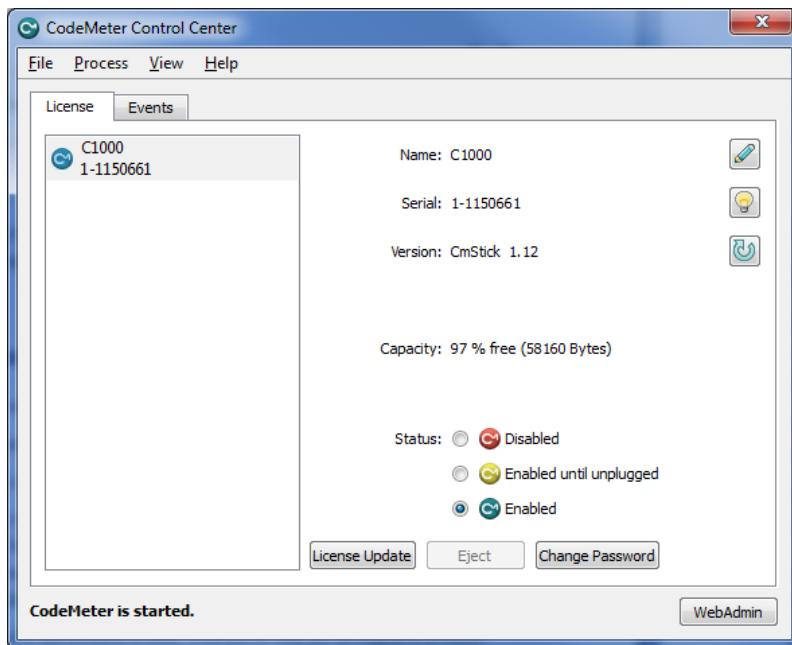
- 1 Contact sales@tuflow.com to request a TUFLOW FV licence file and compatible dongle.
Software pricing is available via the TUFLOW website: <http://www.tuflow.com/Prices.aspx>.
- 2 Download TUFLOW FV: <http://www.tuflow.com/FV%20Latest%20Release.aspx>
- 3 Download and install the dongle drivers: <http://www.tuflow.com/FV%20Latest%20Release.aspx>

Once installed, a Codemeter icon  should appear in the system tray. The icon changes colour and appearance depending on the number of dongles attached to the computer as discussed further below.

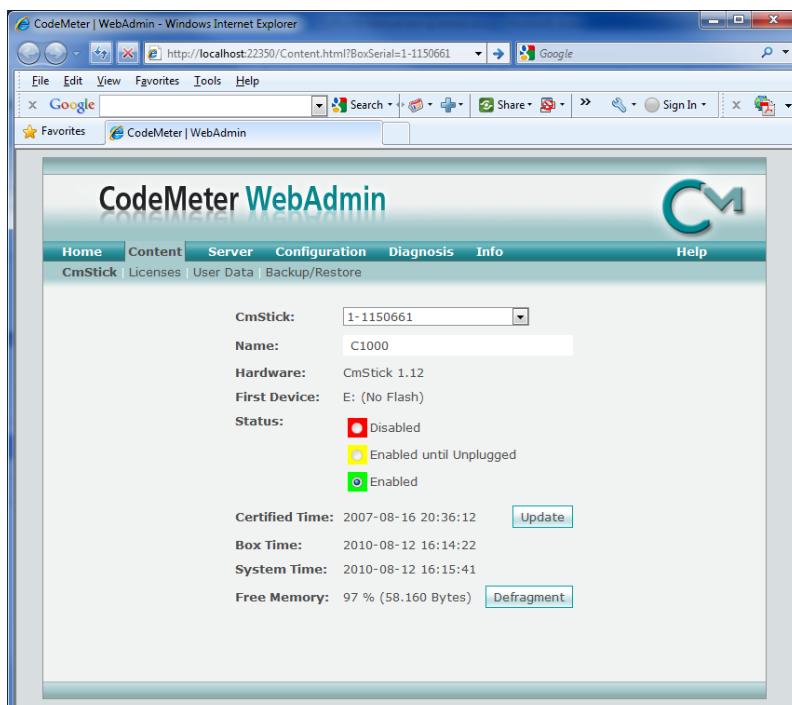
For **Local licences**, double click on the TUFLOWFV.EXE executable, when prompted for an input file press RETURN. The licence information should be presented. Your installation has been successful if this information is displayed in the DOS window. Contact support@tuflow.com if you have any problems

For **Network licences**, the Codemeter will need to be setup as a service using the following steps:

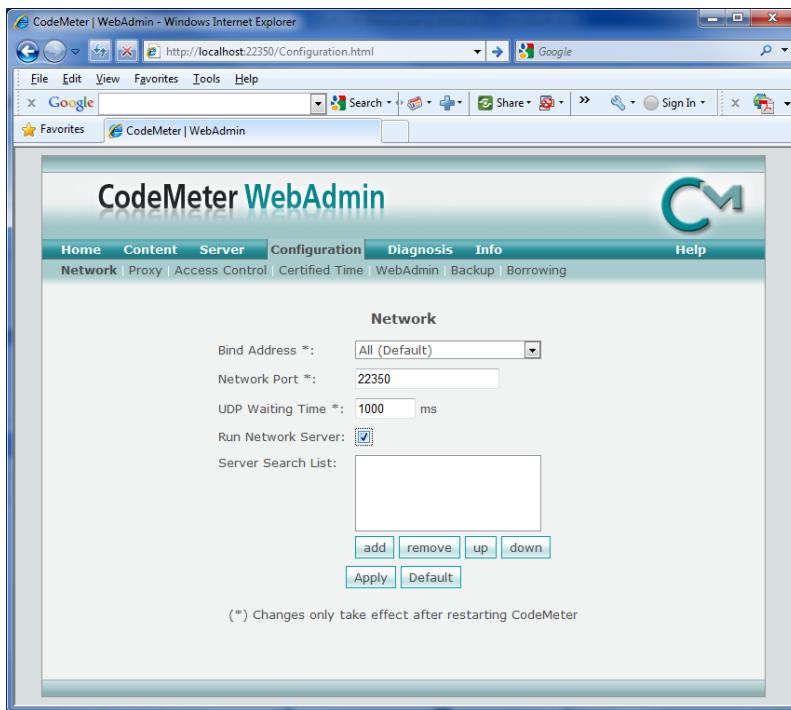
- 4 On the machine with the Network dongle attached, click on the Codemeter icon  in the system tray to bring up the Codemeter Control Center dialog below.



- 5 The dongle should appear under the License tab.
- 6 Click WebAdmin (WebAdmin may also directly be accessed by right clicking on the Codemeter icon in the system tray). This will open your preferred internet browser and appear something like the below.



- 7 Go to the Configuration tab (see below) and tick “Run Network Server”, then click Apply.



- 8 The network dongle should now be accessible by other computers. Note, the above process only needs to be run on the computer with the network dongle attached. All computers wishing to access the network licence need only install the dongle drivers:
<http://www.tuflow.com/FV%20Latest%20Release.aspx>

6.1.1 Codemeter Features

Supported Microsoft Windows platforms include Windows 7, Vista, XP, 2000 and Server 2000/2003/2008. Non-windows platforms supported by Codemeter dongles include Mac OS X, a range of Linux platforms, and Sun Solaris, and are being investigated for use by TUFLOW.

Once installed, a Codemeter icon  should appear in the system tray. The icon changes colour and appearance depending on the number of dongles attached to the computer.

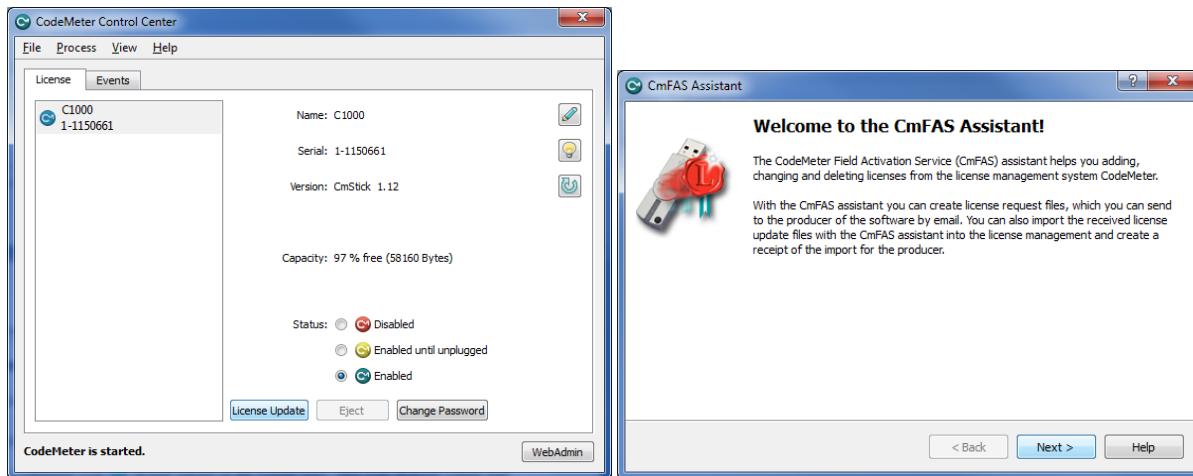
Table 6-1 Codemeter Dongle Status

	Green icon	An enabled Codemeter dongle is connected.
	Gray icon	No Codemeter dongles are attached.
	Yellow icon	A Codemeter dongle is connected, which is enabled until it is un-plugged.
	Red icon	A disabled Codemeter dongle is connected.
	Blue icons	Multiple Codemeter dongles are connected.

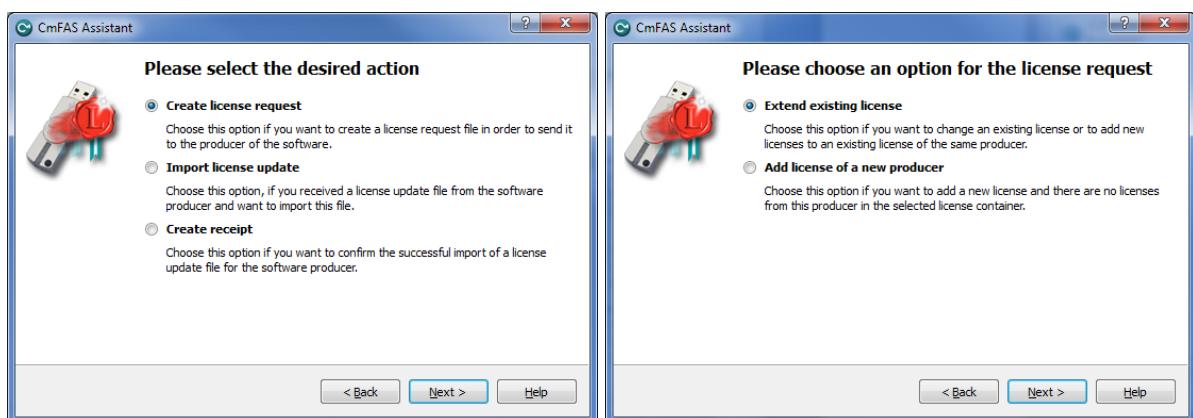
6.1.2 Requesting a Licence Change

To request a change to your WIBU licence please contact sales@tuflow.com. Once the change/upgrade has been agreed upon, follow the steps below and email the update .WibuCmRaC file.

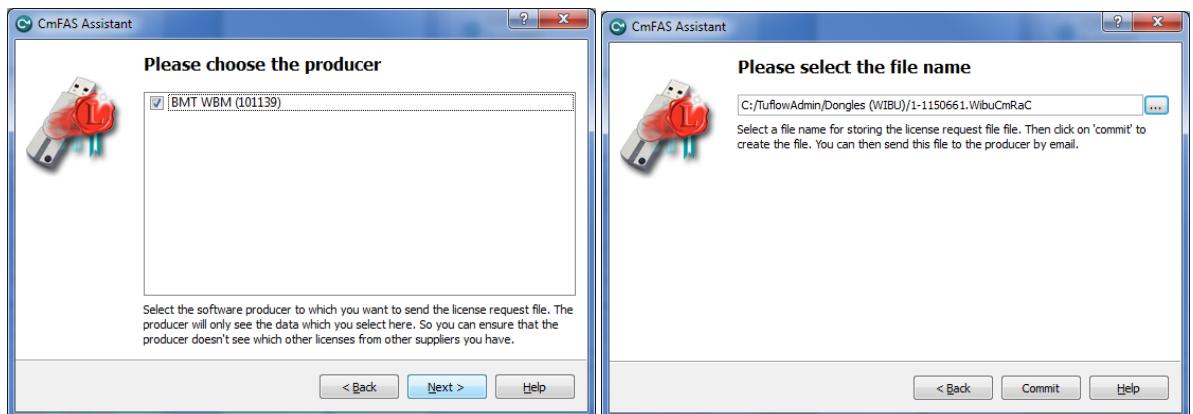
- 1 Click on the Codemeter icon in the system tray to bring up the dialog below (make sure the dongle is plugged in). Your dongle should appear in the dialog below.
- 2 Click on “License Update” to bring up the below.



- 3 Click Next >, then select “Create License Request” in the below.
- 4 Click Next > then select “Extend existing license” in the below.



- 5 Click Next > and make sure BMT WBM (101139) is selected in the next dialog.
- 6 Click Next > to bring up the below. Change if you wish to (and remember the location) of the filepath and click Commit.



- 7 Email the .WibuCmRaC file to sales@tuflow.com. We'll then email back another file for you that will update your dongle for a trial period.

6.2 Running TUFLOW FV

There are a wide variety of ways available to run TUFLOW FV, including:

- Double-click;
- From a Console (DOS) Window or "Run";
- Microsoft Explorer right mouse button shortcut;
- From a text editor (Ultra Edit or Notepad);
- Using a batchfile; or
- From the SMS interface.

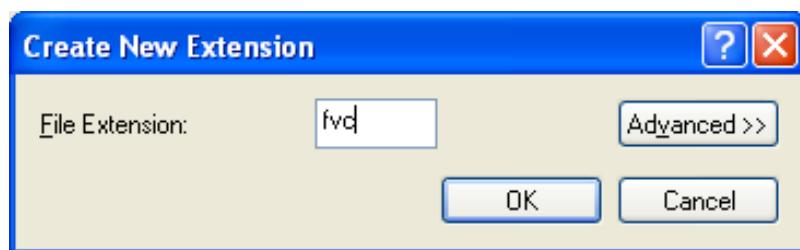
Keep in mind that for all the above approaches, the executable is a single file "tuflowfv.exe"; the operating system, console program or 3rd party program simply accesses this file with associated command line arguments.

The most commonly used options are discussed in the following sections. For more details, refer to the TUFLOW FV Wiki: http://fvwiki.tuflow.com/index.php?title=Running_TUFLOW_FV

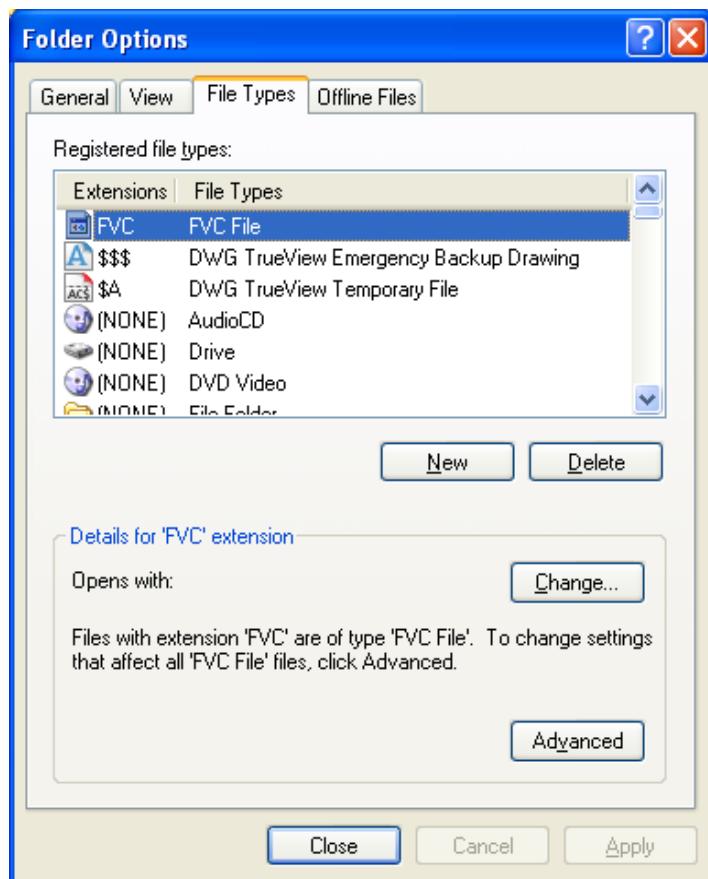
6.2.1 Right Mouse Button in Microsoft Explorer

To start a simulation in Microsoft Explorer by using the right mouse button, first follow the following steps to set up a file association:

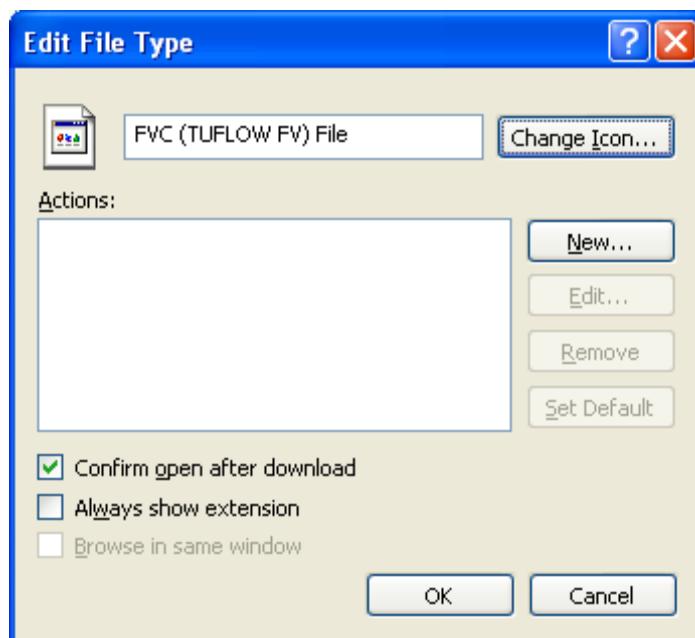
- 8 In Explorer, open the "View" (or "Tools"), "Folder Options..." menu and select the "File Types" tab. If . fvc files are not in the "Registered file types:" list box, choose "New Type...", otherwise select the .fvc file entry under "Registered file types:" as shown in Step 3 below.
- 9 If adding a new type, enter in a description (e.g. "TUFLOW FV Control File") and "fvc" as the associated extension (see below) and press OK.



- 10 The Folder Options dialog should appear something like the below.



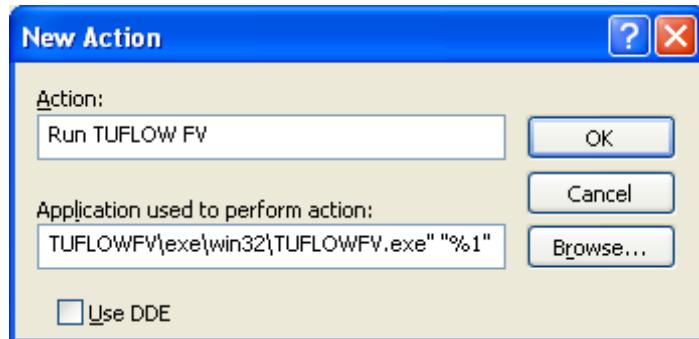
- 11 Click “Advanced” to bring up the dialog below (you can add a new icon and change the file type description here).



- 12 Choose “New...” and enter text to describe the “Action:” (e.g. “Run TUFLOW FV”) – this text appears on the pop-up menu when you click the right mouse button on an .fvc file in Explorer. Enter or use “Browse...” to specify the path to TUFLOWfv.exe; note the need for quotes if the

path has any spaces. After “TUFLOWfv.exe”, add a space then “%1” including the quotes, as shown below. Choose “OK”. The “Application used to perform action:” field should be something like:

"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" "%1"



- 13 The action should now appear in the list under “Actions:”. It is not recommended that a “Run TUFLOW FV” action be set as the default action as it is easy to accidentally start a simulation, which instantly overwrites any existing result files. You may wish to set up other associations at this point, for example, to access your preferred editor.
- 14 Choose “OK” or “Close”, then “Close” on the “Folder Options” menu.
- 15 Check the file association, by clicking the right mouse button on an .fvc file in Windows Explorer. The “Run TUFLOW FV” action should appear in the list.

Once the file association is complete, clicking the right mouse button on an .fvc file in Explorer, and selecting the “Run TUFLOW FV” action, starts a simulation. A Console Command Window opens and TUFLOW FV starts.

6.2.2 From a Text Editor

The benefits of running TUFLOW FV from a text editor is that it provides a common environment where the control files can be edited, simulations started and text file output be viewed. There is no need to close the .fvc file (or other control and output files) to run TUFLOW FV.

Setting up TUFLOW FV to run from UltraEdit and Notepad ++ is described in the TUFLOW FV wiki: http://fvwiki.tuflow.com/index.php?title=Running_TUFLOW_FV.

6.2.3 Using a Batch File

One or many simulations can all be specified within a batch file. The simplest format is to specify each simulation one after another. The following shows the contents of a 4 line batch file (which could be named “TUFLOW FV Simulations.bat”):

```
"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" run01.fvc
"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" run02.fvc
"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" run03.fvc
Pause
```

The .bat file is run or opened by double clicking on it in Explorer. This opens a Console Window and then executes each line of the .bat file. The pause at the end stops the Console window from closing automatically after completion of the last simulation.

Note that the full path and executable is within double quotes; this is needed when there is a space in the command.

TUFLOW FV is a multi-threaded program based on the OpenMP shared-memory model. It will automatically spawn multi-threaded simulations. If not explicitly specified the number of threads will equal either the limit of the computers capacity or the number of licences available. A user can control the number of threads used during a simulation using the batchfile command “OMP_NUM_THREADS”.

Example:

```
C:\>set OMP_NUM_THREADS=4
"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" run01.fvc
"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" run02.fvc
"C:\Program Files\TUFLOWFV\exe\win32\TUFLOWFV.exe" run03.fvc
Pause
```

6.2.4 From the SMS Interface

Should a complete GUI that allows the user to create, manage and view models and model output within the one interface be desired, an interface for TUFLOW FV within SMS has been developed. At present the interface does not allow access to all the features of TUFLOW FV, however, will be expanded in the future.

The TUFLOW FV wiki outlines the steps required to install the SMS interface:

http://fwwiki.tuflow.com/index.php?title=SMS_Tips#TUFLOW_FV_SMS_Interface

Tutorial Module 2 on the TUFLOW FV wiki steps through the process of developing and running a model using the interface:

http://fwwiki.tuflow.com/index.php?title=Tutorial_Module02

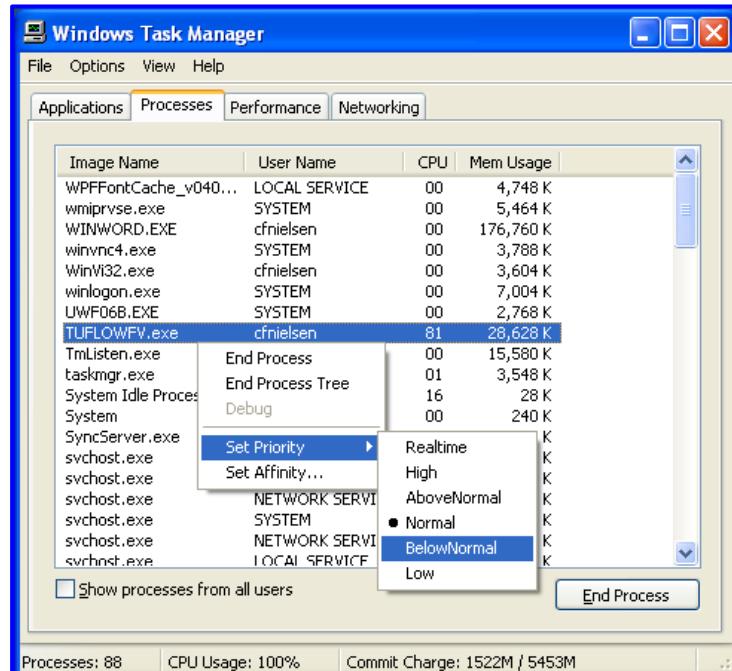
6.2.5 Change Priority, Pause, Restart or Cancel a Simulation

Windows NT/2000/XP/7 can assign a process a different priority level using the Task Manager. This is very useful for running simulations in the “background” without slowing down other computer work you need to do.

To change the priority level of simulation manually:

- Open Task Manager (see your System Administrator if you’re not sure how to do this)
- Click on the Processes Tab
- Find the TUFLOWFV.exe process you wish to change
- Right click, choose Set Priority, then the priority desired as shown in the image below

Note, don't choose High or Realtime as this will cause TUFLOW FV to take over your CPU and you may not be able to do much until the simulation is finished.



Simulation priority can also be specified when using a batchfile. Refer to the TUFLOW FV wiki for more details:http://fvwiki.tuflow.com/index.php?title=Running_TUFLOW_FV.

To pause a model simulation, highlight the console window and press “Ctrl-S”. This will temporarily halt the model simulation.

To continue again, press “Ctrl-Q”.

To cancel a simulation, “Ctrl-C”.

7 Command File (FVC) Reference

7.1 Control File Layout

A TUFLOW FV simulation control file (.fvc) is a text file describing how a simulation is to be performed. Recommended command categories are summarised in Table 7-1.

Table 7-1 Recommended TUFLOW FV Simulation Control File Sections

Section	Command Categories
Definition	General definitions for the simulation, what modules are included, locations of files, etc.
Logging and Pre-processing feedback	Echo of input data, spatial references, etc.
Time	Time Format and Reference Time
	Start / End Times
Geometry	Mesh file
	3D geometry definitions
Solution Scheme	Wetting / drying, CFL limits, etc.
Turbulence	
Physical Parameters	
Materials	Material properties (roughness, mixing parameters, etc.)
Initial Conditions	Initial model state (initial parameters, restart files, etc.)
Boundary Conditions	Global (winds, waves, rainfall, etc.)
	Nodestring (external boundaries, water levels, flows, etc.)
	Cell (source)
	Node (point source)
Output	Output directory
	Prescribe model output
Additional Modules	Depending upon your preference, these commands can be included in the above structure (for example, your Definition category may include specification to include the advanced modules) or as separate entries (for example, if you started with an HD model then added salinity as a subsequent step).
	Structures
	3D
	Salinity, temperature, density
	Wind, Atmospheric Pressure and Waves
	Heat exchange
	Sediments
	Water Quality

An example fvc file is shown in Figure 7-1. New users wishing to learn how to create a simulation control file are advised to complete the TUFLOW FV tutorial models which are available via the TUFLOW FV Wiki (<http://fvwiki.tuflow.com>).

```

1 ! EXAMPLE FV SIMULATION CONTROL FILE ( Example.fvc )
2 !
3 !DEFINITION COMMANDS
4 include salinity == 1.0
5 !
6 !LOGGING AND PREPROCESSING COMMANDS
7 !
8 !TIME COMMANDS
9 Start time == 0.0
10 End Time == 6.0
11 cfl == 1.0
12 Timestep Limits == 0.01,100
13 !
14 ! GEOMETRY COMMANDS
15 geometry 2d == ..\geo\trap_steady_03.2dm
16 !
17 ! MODEL PARAMETERS COMMANDS
18 stability limits == 10.0,100.0
19 Momentum mixing model == Smagorinsky
20 Scalar mixing model == constant
21 Global horizontal eddy viscosity == 0.2
22 !
23 ! MATERIAL PROPERTIES
24 material == 1
25   bottom roughness == 0.018
26 end material
27 !
28 !INITIAL CONDITIONS
29 initial water level == -3.5
30 Initial Salinity == 0
31 !
32 !BOUNDARY CONDITIONS
33 bc == Q, 1, ..\bc\steadyQ_01.csv
34   bc header == time,flow,Sal
35   Sub-type == 4
36 end bc
37 bc == WL, 2, ..\bc\steadyWL_01.csv
38   bc header == time,WL,Sal
39 end bc
40 !
41 !OUTPUT COMMANDS
42 Output Dir == ..\output\
43 output == datv
44   output parameters == h,v,d,Sal
45   output interval == 600.
46 end output

```

Figure 7-1 Example TUFLOW FV Simulation Control File

7.2 Command Line Syntax

Each command line entry is defined by a descriptor, followed by a “==”, followed by the specified value (or values) for the particular command line. The syntax in the tables that follow use a triangular bracket to specify a value that requires user specification.

- As an example, for the command line “***Include == <file name>***” the syntax inserted into the fvc would be (for example) “***Include == includefile.inc***”.

For command lines that have an option of several values, a “;” separator is specified in the syntax.

- For example, the command line “***Time format == <Hours;ISODate>***” requires a choice of two options, so that the command line will be either “***Time format == Hours***” or “***Time format == ISODate***”.

For command lines that have a series of values to specify, a “,” separator is specified in the syntax.

- For example, the command line “***Timestep Limits == <min timestep (s), max timestep (s)>***” requires two entries in the command line, such as “***Timestep Limits == 0.01, 10.0***”.

Some command lines specify an “on or off” switch for a particular parameter.

- In such cases a “1” means “on” (or TRUE) and a “0” means “off” (or FALSE).

When specifying file names in the fvc file it is recommended that relative file paths are specified. This will make the TUFLOW FV simulation files more portable (it’s easier to move an entire folder structure in this way). However, a full path name can also be inserted if preferred (a common example is when output files are written to a separate folder on another disk drive).

Strictly speaking, TUFLOW FV inputs are entered as integers (whole numbers), reals (float, or decimal numbers) and characters (text). The command line entries in the following tables adhere to this syntax, although real numbers can be inserted as integers.

- For example, the default “***CFL == 1.0***” can also be entered as “***CFL==1***”.

Finally, take note that ***not all command lines have to be included in an fvc file***. A simple model setup often requires only a small list of command lines, while the remaining model parameters are either unused or remain as the default value.

Appendix A – Simulation Control File Commands (.fvc)

SIMULATION CONFIGURATION COMMANDS

[Logdir](#)

[Spherical](#)

[Units](#)

[Momentum mixing model](#)

[Vertical mixing model](#)

[External turbulence model dir](#)

[Spatial order](#)

[Bottom drag model](#)

[Include](#)

[Include Coriolis](#)

[Include inviscid](#)

[Include viscous](#)

[Include wind](#)

[Include bed friction](#)

[Mode split](#)

[External model 2d](#)

Logdir == <filepath>

(Optional)

This command specifies the directory for TUFLOW FV simulation log file (.log) output. A log file is automatically generated for each simulation, the contents of which are the same as that displayed in the simulation window. The log filename has the same prefix as the simulation control file.

If not specified, the log file will be written to either the same location at the simulation control file or to the input\log sub-directory if this has been first created by the user (see suggested sub-folder structure in Section 2.2.1).

Other check files (*geo.nc, *cfl_dt.csv) and [restart](#) files (if specified) are also written to the specified log directory location.

Spherical == <0;1>

(Optional; Default == 0)

Flag to specify that the model is in spherical coordinates:

- 0 = Cartesian where geometry inputs and computational coordinates are in metres / feet.
- 1 = Spherical where geometry inputs and computational coordinates are in degrees.

Units == <metric; Imperial; US Customary>

(Optional; Default == metric)

Optional command to apply Imperial or US Customary units (if not specified the default is metric). All simulation inputs, model parameters and outputs will follow the specified units

Note that currently the units are valid only for 2D hydrodynamics; please contact support@tuflow.com if considering using customary units for additional modules.

Momentum mixing model == <None; Constant; Smagorinsky>

(Optional; Default == None)

Sets the horizontal eddy viscosity calculation method. See also [global horizontal eddy viscosity](#).

- None: horizontal momentum mixing is not represented.
- Constant: specify a constant eddy viscosity using the [global horizontal eddy viscosity](#) command.
- Smagorinsky: the horizontal eddy viscosity is calculated according to the Smagorinsky model - specify the Smagorinsky coefficient using the [global horizontal eddy viscosity](#) command.

Vertical mixing model == <Constant; Parametric; External>

(Optional; Default == Constant)

Sets the vertical momentum and scalar mixing model:

-
- Constant: a constant viscosity / diffusivity value is applied to the vertical mixing of both momentum and scalars.
 - Parametric: a zero-equation parametric turbulence model in which a parabolic eddy viscosity / diffusivity profile is calculated. Stratification is represented using the Munk & Anderson stability formulae.
 - External: any external turbulence model that has been built by the user to couple with TUFLOW FV through the fvwbm_external_turb.dll.

External turbulence model directory == <directory path>

(Optional)

Optional command to specify the directory for external turbulence model definition files if an external [vertical mixing model](#) is used. If not specified, external turbulence model files must be located in the same directory at the simulation control file.

Spatial Order == <1;2 (horizontal), 1;2 (vertical)>

(Mandatory; Default == 1,1)

Specifies the spatial order of accuracy of the solution schemes used in the simulation:

- 1 = first order scheme
- 2 = second order scheme

The first-order schemes assume a piecewise constant value of the modelled variables in each cell, whereas the second-order schemes perform a linear reconstruction.

Higher order spatial schemes will produce more accurate results in the vicinity of sharp gradients; however they will be more prone to developing instabilities and are more computationally expensive.

As a general rule of thumb, initial model development should be undertaken using low-order schemes, with higher-order spatial schemes tested during the latter stages of development. If a significant difference is observed between low-order and high-order results then the high-order solution is probably necessary, or alternatively further mesh refinement is required.

Second order spatial accuracy will typically be required in the vertical direction when trying to resolve sharp stratification.

See also the [horizontal gradient limiter](#) and [vertical gradient limiter](#) commands, which may be used to specify the Total Variation Diminishing (TVD) limiting schemes employed during the higher-order reconstructions.

Bottom drag model == <'Manning' ; 'ks'>

(Optional; Default == Manning)

This command can be used to specify the bottom drag model to be used in the simulation.

The default model is Manning, in which case a Manning's "n" coefficient(s) should be specified using the [global bottom roughness](#) or material [bottom roughness](#) command.

An alternative model assumes a log-law velocity profile and requires specification of a surface roughness length-scale, in which case "ks" values should be specified using the [global bottom roughness](#) or material [bottom roughness](#) command.

Include == <file path>

(Optional)

At any location in the simulation control file (.fvc) an 'include file' can be used. Commands contained in the include file will be read as if they are listed in the .fvc file.

Include Coriolis == <0;1>

(Optional; Default == 1)

Optional command used to switch off the Coriolis force source term from the momentum conservation equations:

- 0 = False (i.e. Coriolis forces source term is not included).
- 1 = True (i.e. Coriolis forces source term is included).

Include inviscid == <0;1>

(Optional; Default == 1)

Optional command used to switch off the inviscid flux terms in the momentum and mass transport equations:

- 0 = False (i.e. inviscid flux terms are not included).
- 1 = True (i.e. inviscid flux terms are included).

Include viscous == <0;1>

(Optional; Default == 1)

Optional command used to switch off the viscous flux terms in the momentum and mass transport equations:

- 0 = False (i.e. viscous flux terms are not included).
- 1 = True (i.e. viscous flux terms are included).

Include bed friction == <0;1>

(Optional; Default == 1)

Optional command used to switch off bed friction and thereby simulate an 'ideal' fluid:

- 0 = False (i.e. bed friction is not included).

-
- 1 = True (i.e. bed friction is included).

Include wind == <0;1>

(Optional; Default == 1)

An optional command used to remove the wind stress terms from the momentum and mass transport equations (only relevant if wind is a specified input using the [BC](#) command):

- 0 = False (i.e. wind stress terms are not included).
- 1 = True (i.e. wind stress terms are included).

Mode split == <0;1>

(Optional 3D; Default == 1)

For three-dimensional simulations, TUFLOW FV uses a mode-splitting approach to efficiently solve the external (free-surface) mode in 2D at a timestep constrained by the surface wave speed while the internal 3D mode is updated less frequently. This command is used to disable mode-splitting:

- 0 = False (i.e. mode-splitting is disabled).
- 1 = True (i.e. mode-splitting is enabled).

External mode 2d == <0;1>

(Optional 3D; Default == 1)

This command is used to disable solving the external (free-surface) mode in 2D:

- 0 = False (i.e. external mode is solved in 3D).
- 1 = True (i.e. external mode is solved in 2D).

TIME COMMANDS

[Time format](#)

[Reference time](#)

[Start time](#)

[End time](#)

[Timestep](#)

[CFL](#)

[CFL internal](#)

[CFL external](#)

[Timestep Limits](#)

[Display dt](#)

Time Format == <Hours;ISODate>

(Mandatory; Default == Hours)

Specifies the simulation time format:

- ‘Hours’ time in decimal hours (e.g. Start Time == 3.0)
- ‘ISODate’ requires a date specification in the form dd/mm/yyyy HH:MM:SS (or some truncation thereof, e.g. Start Time == 03/01/2009 03:00).

Subsequent simulation time commands and simulation inputs must be in the specified time format. Simulation outputs will be in the specified time format.

Reference Time == <Input/Output reference time>

(Optional; For Time Format == Hours, Default == 0;

For Time Format == ISODate, Default == 01/01/1990 00:00:00)

Optional command to set the simulation reference time.

Start Time == <simulation start time>

(Mandatory; No Default)

Specifies the start time for the simulation:

- For Time Format == Hours, units are in decimal hours.
- For Time Format == ISODate, inputs are in date form *dd/mm/yyyy HH:MM:SS* (or some truncation thereof).

End Time == <simulation end time>

(Mandatory; No Default)

Specifies the end time for the simulation:

- For Time Format == Hours, units are in decimal hours.
- For Time Format == ISODate, inputs are in date form *dd/mm/yyyy HH:MM:SS* (or some truncation thereof).

Timestep == <constant timestep (s)>

(Optional)

Specifies the value of a constant timestep that is to be used during the simulation.

If not entered then a variable timestep is applied according to the CFL stability criterion, see CFL and Timestep Limits)

CFL == <global maximum Courant-Friedrichs-Lowy number>

(Mandatory; Default == 1)

Sets the Courant–Friedrichs–Lewy (CFL) condition used for the calculation of both the internal (advective) and external (free-surface) terms.

The default value is 1, which is the theoretical stability limit. Sometimes models can be successfully ‘overclocked’ with $\text{CFL} > 1$.

CFL internal == <internal maximum Courant-Friedrichs-Lowy number>

(Optional; Default == 1)

Optional command to specify a Courant–Friedrichs–Lewy (CFL) condition for the calculation of internal (advective) terms only.

CFL external == <external maximum Courant-Friedrichs-Lowy number>

(Optional; Default == 1)

Optional command to specify a Courant–Friedrichs–Lewy (CFL) condition for the calculation of external (free-surface) terms only.

Timestep Limits == <min timestep (s) , max timestep (s)>

(Mandatory; No Default)

Specifies the maximum and minimum variable timestep allowed according to the CFL stability criterion. See also [CFL](#).

Display dt == <display timestep (s)>

(Optional; Default == 300)

Allows the user to specify the simulation time interval (in seconds) for displaying timestep information.

MODEL PARAMETER COMMANDS

[Stability limits](#)

[Cell dry/wet depths](#)

[Cell 3D depth](#)

[Horizontal gradient limiter](#)

[Horizontal AlphaR](#)

[Vertical gradient limiter](#)

[Vertical Alpha R](#)

[g](#)

[Latitude](#)

[Reference MSLP](#)

[Wind stress params](#)

Stability Limits == <maximum WL, maximum velocity>

(Optional)

Optional command to specify a maximum water level and maximum velocity which indicate an unstable model. The simulation will stop if these limits are exceeded.

Cell dry/wet depths == <cell dry depth (m), cell wet depth(m)>

(Optional; Default == 1.0e-6, 1.0e-2)

Sets the cell wetting and drying depths in metres:

- The drying value corresponds to a minimum depth below which the cell is dropped from computations (subject to the status of surrounding cells).
- The wet value corresponds to a minimum depth below which cell momentum is set to zero in order to avoid unphysical velocities at very low depths.

Cell 3D depth == <threshold depth (m)>

(Optional 3D)

An optional command to set the threshold water depth for 3D calculations. Areas where the depth is less than the threshold value essentially revert to a 2D calculation.

Horizontal gradient limiter == <LCD; MLG>

(Optional; Default == LCD)

Sets the Total Variation Diminishing (TVD) limiting scheme for 2nd order horizontal spatial integration scheme, the options are:

- LCD is the less compressive option and the least computationally intensive
- MLG is the most compressive option and the most computationally intensive

Horizontal AlphaR == <alphaH (depth), alphaV (velocity), alphaS (scalars)>

(Optional; Default == 1.0, 1.0, 1.0)

This command can be used to apply a reduction factor to high-order cell reconstruction gradients, which may be useful in stabilising a higher-order simulation.

Default is <1.0, 1.0, 1.0>, which corresponds to no gradient reduction, whereas <0.0, 0.0, 0.0> would revert to a first-order scheme.

Vertical gradient limiter == <MINMOD;MC;Superbee>

(Optional 3D; Default == MC)

Sets the Total Variation Diminishing (TVD) limiting scheme for 2nd order vertical spatial integration scheme.

The options are MINMOD, MC (Monotized Central) and Superbee (ranging from least compressive to most compressive).

Vertical AlphaR == <alphav (velocity), alphas (scalars)>

(Optional; Default == 1.0, 1.0)

This command can be used to apply a reduction factor to high-order cell reconstruction gradients, which may be useful in stabilising a higher-order simulation.

Default is <1.0, 1.0>, which corresponds to no gradient reduction, whereas <0.0, 0.0> would revert to a first-order scheme.

g == <gravitational acceleration (m/s²) or (ft/s²)>

(Optional)

Gravitational acceleration. If not specified, then the default value depends on the specified [units](#):

- 9.81 m/s²
- 32.174 ft/s²

Latitude == <latitude in degrees (-ve for Southern Hemisphere)>

(Optional; Default == 0.0)

Sets the latitude for Coriolis calculations when a Cartesian coordinate system is used.

Reference MSLP == <Mean Sea Level Pressure (hPa)>

(Optional; Default == 1013.25)

Optionally sets the reference mean sea level pressure value.

Wind Stress Parameters == <W_a (m/s) , C_a(-) , W_b (m/s) , C_b(-)>

(Optional)

Optionally specifies the parameter values in the following wind stress drag model:

$$Cd = Ca; [W_{10} < W_a]$$

$$Cd = Ca + (W_{10} - W_a) / (W_b - W_a) * (C_b - C_a); [W_a \leq W_{10} \leq W_b]$$

$$Cd = Cb; [W_{10} > W_b]$$

The default parameters are <0.0, 0.8e-03, 50.0, 4.05e-03> corresponding to the Wu parameterisation (with a 50 m/s upper limit).

TURBULENCE PARAMETER COMMANDS

[Kinematic viscosity](#)

[Global horizontal eddy viscosity limits](#)

[Global vertical eddy viscosity limits](#)

[Vertical mixing parameters](#)

[Turbulence update dt](#)

Kinematic Viscosity == <kinematic viscosity value (m²/s)>

(Optional; Default == 1.0e-6)

Optionally specifies the background water kinematic viscosity.

Global Horizontal Eddy Viscosity == <eddy viscosity (m²/s) ; coefficient/s (-)>

(Mandatory)

Globally sets a constant horizontal eddy viscosity (m²/s) or the horizontal eddy viscosity coefficient. This is dependent on the momentum mixing model set using the [momentum mixing model](#) command:

- Constant: specify a constant eddy viscosity; Default == 0
- Smagorinsky: specify the Smagorinsky coefficient; Default == 0.2

Global Horizontal Eddy Viscosity Limits == <min eddy viscosity (m²/s)>, <max eddy viscosity (m²/s)>

(Optional)

For use with Smagorinsky [momentum mixing model](#), globally sets the minimum and maximum horizontal eddy viscosity (m²/s) limits.

Not applicable if a Constant horizontal eddy viscosity is set using the [global horizontal eddy viscosity](#) command.

Vertical mixing parameters == <eddy viscosity (m²/s) ; coefficients (-)>

(Optional)

Globally sets a constant vertical eddy viscosity (m²/s) or the vertical eddy viscosity coefficients. This is dependent on the vertical mixing model set using the [vertical mixing model](#) command:

- Constant: specify a constant eddy viscosity; Default == 0
- Parametric: specify the parametric model coefficients; Default == 0.4, 0.4

Not applicable if coupling with an External vertical mixing model.

Vertical Eddy Viscosity Limits == <min eddy viscosity (m²/s) , <max eddy viscosity (m²/s)>

(Optional)

For use with Parametric or External [vertical mixing model](#), globally sets the minimum and maximum vertical eddy viscosity (m²/s) limits.

Not applicable if a Constant vertical eddy viscosity is set using the [vertical mixing parameters](#) command.

Turbulence update dt == <time (s)>

(Optional)

Optional command to specify the timestep for updating the vertical turbulence mixing eddy-viscosity and scalar-diffusivity terms. If not specified, this will occur at every timestep.

GEOMETRY COMMANDS

[Geometry 2d](#)

[Cell elevation file](#)

[Cell elevation polyline](#)

[Cell elevation polygon file](#)

[Nodestring polyline file](#)

[Global bed elevation limits](#)

[Vertical mesh type](#)

[Layer faces file](#)

[Surface sigma layers](#)

[Min bottom layer thickness](#)

[Echo geometry netcdf](#)

[Echo geometry csv](#)

[Structure logging](#)

Geometry 2d == <mesh file location and name (.2dm)>

(Mandatory)

Specifies the model 2D geometry input file. The input file must be in a format consistent with the SMS generic mesh module format (see Section 4.2.1.1).

When following the suggested TUFLOW FV folder structure (refer Section 2.2.1):

geometry 2d == ..\geo\ mesh_name.2dm

Cell elevation file == <cell elevation file (.csv), xytype, ztype>

(Optional)

This optional command can be used to set the bed elevations for some or all cells in the model domain, overriding the elevations defined by the .2dm file.

If xytype = cell_ID (Default):

- The .csv file must contain a header line:
 - ID, Z
 - with cell ids and corresponding elevations entered within the rows below the header line

If xytype = coordinate:

- The .csv file contains a first line header:
 - X,Y,Z
 - with the x-coordinate, y-coordinate and corresponding elevations entered within the rows below the header line

If ztype = overwrite (Default):

- the cell elevation will correspond to the last z value read (i.e. overwrite the elevations defined by the .2dm file).

If ztype = average:

- the elevation will be the average of the z values read for a given cell (i.e. the average elevation defined by the .2dm file and read from the cell elevation file).

Note that more than one cell elevation file can be listed in the simulation control file (.fvc). Refer to Section 4.2.2.1 for more details.

Cell elevation polyline file == <polyline file (.csv), ID>

(Optional)

This optional command will set bed elevations for cells that are intersected by a polyline. Cell Z values will be interpolated from the z values specified at vertices along the polyline, overriding the corresponding elevations defined by the .2dm and/or cell elevation file.

Polyline file:

-
- The .csv file contains a first line header:
 - X,Y,Z,ID
 - Input data is entered within the rows below the column header

ID:

- Only those vertices listed in the .csv file with an ID value matching the specified value in the command line will be read by the model.
 - note: If ID = 0 or is blank, all vertices will be read from the polyline file

Refer to Section 4.2.2.1 for more details.

Cell elevation polygon file == <polygon file (.csv), ZB, ID>
(Optional)

This optional command reads a polygon defined in a comma separated variable file. All cell centres that lie within the polygon are assigned an elevation ZB, overriding the corresponding elevations defined by the .2dm and/or cell elevation file.

Polygon file:

- The .csv file contains a first line header:
 - X,Y,ID (ID is optional)
 - Input data is entered within the rows below the column header.
 - Points within the csv file define the perimeter of the polygon. The definition of points needs to be consecutively listed and can be either clockwise or counter-clockwise.

ZB:

- The elevation that all cells within the polygon will be assigned.

ID:

- Only those vertices listed in the .csv file with an ID value matching the specified value in the command line will be read by the model.
 - note: If ID = 0 or is blank, all vertices will be read from the polygon file

Refer to Section 4.2.2.1 for more details.

Nodestring polyline file == <nodestring file (.csv), ID , Boundary>
(Optional)

Specifies a comma separated variable file that contains the vertex coordinates of a polyline defining a nodestring path.

The nodestring path is identified by (1) finding the nearest node to each vertex and (2) identifying internal nodes between them.

Two vertices per nodestring are required as minimum inputs. If additional vertices are defined between the start and end vertices, the definition needs to be consecutively listed

Polygon file:

- The .csv file contains a first line header:

- X,Y,Z,ID (Z, ID are optional)
- Input data is entered within the rows below the column header
- Z inputs will assign elevations to the vertices along the nodestring. This will not influence cell elevations (use the [cell elevation polyline](#) for this task), but can be used to assign elevations for [nodestring structures](#).

ID:

- If an ID column exists, only those vertices listed in the .csv file with an ID value matching the specified value in the command line will be read by the model.
 - note that if ID = 0 or is blank, all vertices will be read from the polyline file.

Boundary (optional input):

- If “boundary” is specified, the nodestring path is restricted to being along the boundary

Nodestring numbering is as follows:

- If ID = 0 or is blank, the nodestring ID is the next incremental number after the nodestring IDs in the 2dm file.
- If ID /= 0, the nodestring ID is assigned the ID value. Existing nodestrings will be overwritten if the ID is the same as an existing nodestring.

Global bed elevation limits == <zbmin, zbmax>

(Optional)

Provides option to apply limits to the bed elevations. Can also be applied to specific material types (see [Bed Elevation Limits](#)).

Vertical mesh type == <sigma;z>

(Mandatory 3D; Default == sigma)

Specifies the type of discretisation applied to the 3D layer structure, either:

- Sigma-coordinates
- Z-coordinates

The number of Sigma-coordinate layers is specified using the [sigma layers](#) command or the [layer faces](#) command. Layer face elevations corresponding to the Z-coordinate mesh type are defined using the [layer faces](#) command.

Layer face file == <file specifying layer interface levels (.csv)>

(Optional for Sigma-coordinate mesh; Mandatory for Z-coordinate mesh)

Specifies the location and name of the comma separated variable file containing the 3D layer face information, depending on the [Vertical Mesh Type](#):

- In the case of Sigma-coordinates, the layer faces are optionally specified as decimal fractions between 1 (water surface) and 0 (bed level) in a ‘SIGMA’ column. This command is used if a non-uniform vertical distribution of sigma layers is desired. Alternatively, a uniform vertical distribution of sigma layers is specified using the [sigma layers](#) command.
- In the case of Z-coordinates, fixed layer face elevations are specified in a ‘Z’ column.

In tidal environments the user may wish to adopt a hybrid z-sigma-coordinate mesh. In this instance, the Z-coordinate [Vertical Mesh Type](#) is set and “always wet” fixed layer face elevations are specified in a ‘Z’ column. The number of sigma layers between the maximum “always wet” elevation and the free-surface is then specified using the [sigma layers](#) command.

Surface Sigma layers == <Nsigma>

(Optional 3D)

Depending on the [Vertical Mesh Type](#), this command is optionally used to:

- In the case of Sigma-coordinates, specify the number sigma layers to be uniformly distributed over the entire water column. Alternatively, a non-uniform vertical distribution of sigma layers is specified using the [layer faces](#) command.
- In the case of Z-coordinates, specify the number sigma layers to be uniformly distributed between the maximum “always wet” fixed layer elevation and the free-surface. This creates a hybrid z-sigma-coordinate vertical mesh.

Min bottom layer thickness == <dzmin>

(Optional 3D)

Optionally specify the minimum thickness of the lowest layer (i.e. layer at the bed). This command is used to avoid a thin vertical layer (and associated small timestep) at the bed.

Echo geometry netcdf == <0;1>

(Optional, Default == 1)

Setting this to 0 stops the model from writing a *_geo.nc (netCDF format geometry) check file.

Contents of the *_geo.nc file include:

- Cell geometry including connectivity, coordinates, layers, areas, elevations, materials, bottom roughness, vertices and faces.
- Face geometry including connectivity, coordinates, elevations, lengths, vertices and CFL values.
- Node geometry including connectivity, coordinates, elevations and weightings of adjacent cells.

Echo geometry csv == <0;1>

(Optional, Default == 1)

Setting this to 0 stops the model from writing a series of comma separated variable files that include various relevant geometry and spatial features of the simulation, including:

- Mesh details
- Cell elevations and material types
- Cell elevations that have been updated externally to the .2dm file (for example by using the [cell elevation](#) command)
- Nodestring locations and their purposes (boundary, structure, etc.)
- Output locations

Structure logging == <0;1>

(Optional, Default == 0)

Setting this to 1 will write a structural log file (.slf) that contains the operational behaviour of included structures through time.

MATERIAL PROPERTIES COMMANDS

[Global bottom roughness](#)

[Material](#)

[Inactive](#)

[Bottom roughness](#)

[Surface roughness](#)

[Horizontal eddy viscosity](#)

[Horizontal eddy viscosity limits](#)

[Vertical eddy viscosity](#)

[Vertical eddy viscosity limits](#)

[Bed elevation limits](#)

[Spatial reconstruction](#)

Global bottom roughness == <bottom roughness>

(Optional)

Globally sets the bottom roughness value. The bottom roughness specification depends on the [bottom drag model](#), and may be a Manning's "n" coefficient (default) or an equivalent Nikuradse roughness, "ks" (m).

Material == <material id #>

...

...

...

End Material

(Mandatory)

This command indicates the beginning of a material block, specifying unique properties for cells with material id #. Material properties are listed in the following rows and the 'end material' command is used to indicate the end of the material block.

The following example material block specifies unique bottom roughness, horizontal eddy viscosity and horizontal scalar diffusivity values for all cells with material type 1 (thereby overriding the default or corresponding global turbulence parameters):

```
material == 1
    bottom roughness == 0.020
    horizontal eddy viscosity == 0.20
    horizontal scalar diffusivity == 60.0, 6.0
end material
```

Note that several material types can be grouped into a single material block:

```
material == 2,3,4
    bottom roughness == 0.1
end material
```

As a minimum, the roughness for each material type specified in the geometry file should be defined using material block commands (see also Section 4.2.1.1) or the [global bottom roughness](#) command. The commands that can be used to within a material block include:

- [Inactive](#)
- [Bottom roughness](#)
- [Surface roughness](#)
- [Horizontal eddy viscosity](#)

-
- [Horizontal scalar diffusivity](#)
 - [Horizontal eddy viscosity limits](#)
 - [Horizontal scalar diffusivity limits](#)
 - [Vertical eddy viscosity limits](#)
 - [Vertical scalar diffusivity limits](#)
 - [Bed elevation limits](#)
 - [Spatial reconstruction](#)
-

Inactive == <0;1>

(Optional, Default == 0)

[Material](#) block command used to exclude cells with material id# from the computational domain:

- 0 = False (i.e. cells included).
- 1 = True (i.e. cells excluded).

Example material block commands:

```
material == 1
inactive == 1
end material
```

Bottom roughness == <roughness value>

(Optional, Default == value set using [global bottom roughness](#) command)

[Material](#) block command used to set the bottom roughness value for cells with material id#. The bottom roughness specification depends on the [bottom drag model](#), and may be a Manning's "n" coefficient (default) or an equivalent Nikuradse roughness, "ks" (m).

Surface roughness == <roughness value>

(Optional, Default == 0)

[Material](#) block command used to set the surface roughness value, typically used to simulate ice cover.

Horizontal eddy viscosity == <eddy viscosity (m²/s) ; coefficient (-)>

(Optional, Default == value set using [global horizontal eddy viscosity](#) command)

[Material](#) block command to specify the horizontal eddy viscosity Constant value (m²/s) or Smagorinsky model coefficient for cells with material id# (thereby overriding the default or

corresponding globally parameters), depending on the turbulence model used. See [momentum mixing model](#) command to set momentum mixing turbulence model.

Horizontal eddy viscosity limits == <dv_limit1, dv_limit2>
(Optional)

[Material](#) block command for use with Smagorinsky [momentum mixing model](#) to set the minimum and maximum horizontal eddy viscosity (m^2/s) limits for cells with material id# (thereby overriding the default or corresponding globally set parameters).

Vertical eddy viscosity == <eddy viscosity (m^2/s); coefficient (-)>
(Optional, Default == value set using [global vertical eddy viscosity](#) command)

[Material](#) block command to specify the vertical eddy viscosity Constant value (m^2/s) or Parametric model coefficient for cells with material id# (thereby overriding the default or corresponding globally set parameters), depending on the [vertical mixing model](#) used. See [vertical mixing model](#) command to set the vertical mixing model.

Vertical eddy viscosity limits == <dv_limit1, dv_limit2>
(Optional)

[Material](#) block command for use with Parametric or External [vertical mixing model](#) to set the minimum and maximum vertical eddy viscosity (m^2/s) limits for cells with material id# (thereby overriding the default or corresponding globally set parameters).

Bed Elevation Limits == <zbmin, zbmax>
(Optional)

[Material](#) block command to specify limits to bed elevations for cells with material id#. Can also be set globally using the [global bed elevation limits](#) command.

Spatial reconstruction == <0;1>
(Optional, Default == 0)

[Material](#) block command used in high order simulations to optionally limit spatial reconstruction for cells with material id# (effectively reducing the spatial order of accuracy of the solution):

- 0 = False (i.e. no higher order reconstruction)
- 1 = True (i.e. higher order reconstruction)

INITIAL CONDITION COMMANDS

[Initial water level](#)

[Initial condition 2d](#)

[Initial condition 3d](#)

[Initial condition quiescent](#)

[Restart file](#)

[Use restart file time](#)

Initial Water Level == <water level (m)>

(Optional)

Command to globally set an initial, quiescent water level.

Initial Condition 2d == <initial condition file (.csv)>

(Optional)

Optional command to read the initial conditions from a comma separated variable file. The .csv file contains initial conditions for each cell of the mesh.

As a minimum, the following column headers are required in this file:

ID, WL, U, V

If salinity, temperature or other scalars are included in the simulation they should also be specified in the .csv file (e.g. Sal, Temp, Scal_1,...). An example of the command usage and corresponding .csv file is given below:

Initial condition 2d == ..\bc\initial_conditions_001.csv

and the contents of initial_conditions.csv:

ID, WL, U, V, Scal_1, Scal_2, Scal_3

1, 0.300, 0.000, 0.000, 1.000, 0.000, 0.000

Initial Condition quiescent

(Optional)

Sets a quiescent initial condition.

Restart file == <restart file name (.rst)>

(Optional)

Optional command to load the simulation initial conditions from a restart file (.rst) generated by a previous TUFLOW FV simulation.

Unless the use restart [file time](#) command is used the simulation start time will be set to the timestamp in the restart file. See also [write restart](#) command.

Use Restart File Time == <1;0>

(Optional, Default == 1)

This command resets the model start time to be equal to the value specified using [start time](#) when a restart file is used (see also [restart](#)). Without this command or when set to 1 (true), the start time is set equal to the restart file timestamp:

-
- 0 = False (i.e. use start time set with command [start time](#))
 - 1 = True (i.e. use time equal to the [restart](#) file timestamp)

BOUNDARY CONDITION COMMANDS

[Grid definition file](#)

[Grid definition variables](#)

[BC default update dt](#)

[BC](#)

[BC header](#)

[Sub-type](#)

[BC offset](#)

[BC scale](#)

[BC default](#)

[BC update dt](#)

[BC time units](#)

[BC reference time](#)

[Include MSLP](#)

[Vertical distribution file](#)

[Vertical coordinate type](#)

[BC nodestrings](#)

[WaveModel](#)

```
Grid definition file == <netcdf file defining grid coords (.nc)>
(Optional)
```

Specifies a netCDF location and filename that defines grid coordinates to be used in mapping input files to the model mesh. Used in conjunction with the [grid definition variables](#) command and prior to the [BC](#) block commands for gridded BC types.

```
Grid definition variables == <v1, v2>
(Optional)
```

Specifies the x,y coordinate variable names (typically Easting, Northing or Longitude, Latitude) contained in the netCDF file defined using the [grid definition file](#) command.

```
grid definition file == ..\bc\wind_10_grid.nc
grid definition variables == Easting, Northing
```

```
BC default update dt == <Update timestep>
(Optional)
```

A global command that allows the user to specify the update timestep for all boundary conditions. If not specified, the boundary condition is updated at every simulation timestep. See [BC update dt](#) for setting the update timestep for a specific boundary condition.

```
BC == <bc type, [id], [input file]>
```

```
...
...
...
```

```
End BC
```

```
<OR>
```

```
BC == <bc type, [xid], [yid], [input file]>
```

```
...
...
...
```

```
End BC
```

(Mandatory)

At least one boundary condition will be required for a TUFLOW FV simulation and often a number of different boundary condition types will be applied. Each boundary condition type is defined using a boundary condition (BC) block. The 'BC' and 'End BC' commands indicate the beginning and end of a boundary condition block.

See Table 4-3and Table 4-4 for lists of boundary types.

Boundary conditions can be applied:

- Spatially (typically metrological conditions and/or wave fields)
- Along a nodestring (external boundaries such as water levels or flows)
 - The [id] value is the nodestring identifier included in the mesh geometry file (see Section 4.2.1.1)
- As a point source (within a single cell such as an outfall discharge)
 - The [xid], [yid] values are the coordinates of the source location within the model domain.

The commands that can be used within a BC block are:

- [BC header](#)
- [Sub-type](#)
- [BC offset](#)
- [BC scale](#)
- [BC default](#)
- [BC update dt](#)
- [BC time units](#)
- [BC reference time](#)
- [Include MSLP](#)
- [Include wave stress](#)
- [Include stokes drift](#)
- [Layer](#)
- [Vertical distribution](#)
- [Vertical coordinate type](#)
- [BC nodestrings](#)

BC Header == <Header1,Header2,...>

(Optional)

[BC](#) block command that allows the user to specify the .csv input file column headers or netCDF file variable names (thereby overriding the defaults in described in Table 4-3and Table 4-4). This command should immediately follow a [BC](#) command.

For example, the following lines apply a cell inflow (QC boundary condition type) at the cell which lies at the x,y coordinate 1025.5, 950.5. It looks in the specified .csv file for columns Time, Tailwater_Flow and Turbidity:

```
BC == QC, 1025.5, 950.5, ..\bc\ tailwater_discharge.csv
  BC header == Time,Tailwater_Flow,Turbidity
End BC
```

Another example shows a water level boundary (WL) applied to nodestring 1, which looks in the specified .csv file for columns Time and Tide_1:

```
BC == WL, 1, ..\bc\ tidal_water_level.csv
  BC header == Time, WL_Loc1
End BC
```

A final example shows a gridded wind field (W10_Grid) applied to a domain previously defined using the [grid definition variables](#) command, which looks in the specified netCDF file for variables Time, Wind_X and Wind_Y:

```
BC == W10_Grid, 1, ..\bc\wind_10_grid.nc
  BC header == Time, Wind_X, Wind_Y
End BC
```

Sub-type == <sub-type number>

(Optional, Default == 1)

The sub-type [BC](#) block command is applicable for Q, WL and OBC boundary types and allows the user to control certain details of how these are numerically implemented.

For a Q type boundary condition:

- If sub-type == 1 (default), flow is:
 - Applied as a flux
 - Distributed across a nodestring by cell width
 - Note: While the net flow will match the input file specifications, using this sub-type with 3D simulations does not guarantee uniform inflow over the entire water column. In some cases one part of the water column can be flowing in while another is flowing out. It is therefore recommended to use sub-type 2 or 4 for 3D models.
- If sub-type == 2, flow is:
 - Applied as a source term
 - Distributed across a nodestring by cell width
 - Note: Boundary condition is specified as a reflective wall with a source distributed along the internal boundary cells.
- If sub-type == 3, flow is:
 - Applied as a flux
 - Distributed across a nodestring by cell width and depth ($W \cdot H^{1.5}$)

- Note: Boundary inflow is distributed according to depth along nodestring. This boundary condition treatment is otherwise the same as sub-type 1. This sub-type may not be suitable for use in 3D model simulations.
- If sub-type == 4, flow is:
 - Applied as a source term
 - Distributed across a nodestring by cell width and depth ($W \cdot H^{1.5}$)
 - Note: Boundary inflow is distributed according to depth along nodestring. This boundary condition treatment is otherwise the same as sub-type 2. This sub-type is suitable for use in 3D model simulations.

Note: for overland application with inflows over an initially dry bed, subtype = 4 is recommended.

For all OBC boundary condition (i.e. OBC, OBC_PROF, OBC_CURT, OBC_GRID):

- If sub-type == 1 (default), the boundary is a specified water level. Boundary normal momentum flux is modified to avoid BC over-specification which can lead to boundary reflection of outgoing energy.
- If sub-type == 2, the boundary is an incoming wave (TBC)
- If sub-type == 3, the boundary specified flow velocity. Water level is modified to avoid BC over-specification which can lead to boundary reflection of outgoing energy.
- If sub-type == 4, the boundary is over specified (both water level and velocity are specified). Water level and velocities are applied exactly as specified in the input files. This can lead to boundary reflection of outgoing energy.
- If sub-type == 5, specified water levels are treated as an increment to apply to the previously specified water level. This can for instance be used to add a tidal signal to a separately specified non-tidal OBC. This sub-type can also be used in conjunction with WL, WLS and WL_CURT BCs.

Note: for application with supercritical upstream boundaries, subtype = 4 is recommended.

QG and QC boundary conditions support the following sub-type specifications:

- If sub-type == 1. When outflow is specified ($Q < 0$) the scalar flux is determined by the interior model concentration (the BC file value will be ignored).
- If sub-type == 2. When outflow is specified ($Q < 0$) the scalar flux is determined by the BC file specified value.

BC offset == <Var1_Offset, [Var2_Offset], ...>

(Optional)

BC block command to apply an offset to boundary condition values.

BC scale == <Var1_Scale_Factor, [Var2_Scale_Factor], ...>

(Optional)

[BC](#) block command to apply a scale factor to boundary condition values.

BC default == <Var1_default, [Var2_default],...>

(Optional)

[BC](#) block command to specify a default boundary condition value if entry in the input file is empty.

BC update dt == <Update timestep>

(Optional)

[BC](#) block command that allows the user to specify the update timestep for a boundary condition. If not specified, the boundary condition is updated at every simulation timestep.

BC time units == <hours;...>

(Optional)

[BC](#) block command used to specify the unit of time for a boundary condition specified using a netCDF file. The options are:

- Days
- Hours
- Minutes
- Seconds

If not specified, the default is hours relative to the simulation [reference time](#).

BC reference time == <Hours;ISODate>

(Optional)

[BC](#) block command to set the boundary condition reference time. If not specified, the boundary condition is assumed to be consistent with the simulation [reference time](#).

Includes MSLP == <1;0>

(Optional, Default == 1)

[BC](#) block command that allows the user to specify whether a water level boundary condition (WL or WLS) includes an inverse barometer offset.

The default assumption (1) is that the boundary does already include an inverse barometer component.

If includes MSLP == 0 then an offset determined by the local MSLP difference from the [reference MSLP](#) is applied at the boundary.

Vertical coordinate type == <elevation;depth;sigma;height>

(Optional 3D)

[BC](#) block command to specify the BC vertical coordinate type for vertically distributed boundary conditions. The options are:

- Elevation
- Depth
- Sigma
- Height

This command is followed by speciation of a [vertical distribution](#) file that defines the vertical distribution.

If not specified, the boundary condition is distributed evenly over the water column.

Vertical distribution file == <file path>

(Optional 3D)

[BC](#) block command used in conjunction with [vertical coordinate type](#) to specify a .csv file that describes the boundary condition vertical distribution.

The .csv file should contain two columns:

- The first column is the vertical coordinate (e.g. DEPTH) type reference.
- The second column is the weighting (between 0 and 1) at the corresponding vertical reference. The units of WEIGHT are irrelevant as the distribution is normalised.

The first example .csv file corresponds to the [vertical coordinate type](#) “depth” and the boundary condition being applied to the top 2m of the water column:

```
DEPTH, WEIGHT
0.0, 1
2.0, 1
2.1, 0
9999.0, 0
```

The second example corresponds to the [vertical coordinate type](#) “height” and the boundary condition being applied to the bottom 1m of the water column.

```
HEIGHT, WEIGHT
0.0, 1
1.0, 1
1.11, 0
9999.0, 0
```

The final example corresponds to the [vertical coordinate type](#) “elevation” and the boundary condition being applied at -10 to -20 meters (below the model datum).

```
ELEVATION, WEIGHT
```

```
0.0, 0
-1.0, 0
-5.0, 0
-9.9, 0
-10.0, 1
-20.0, 1
```

BC nodestrings == <id1, ..., idn>

(Optional)

[BC](#) block command to apply the boundary condition to multiple nodestrings (only relevant to the OBC_GRID boundary type).

STRUCTURE COMMANDS

Structures	Width file
Name	
Flux function	
Flux type	
Culvert file	
Flux file	
Properties	
Control	
Control parameter	
Control file	
Control update dt	
Sample point	
Sample type	
Sample dt	
Max open increment	
Trigger value	
Target file	
Bed adjust	
Cell function	
Polygon file	
Destratification Unit	
Energy loss function	
Form loss coefficient	
Energy loss file	
Blockage file	

Structure == <Structype, ID>

...

...

...

End Structure

(Optional)

Marks the beginning of a structure block.

Structype can be:

- Nodestring
 - The structure is situated between one or more elements (i.e. – along the cell faces, defined by a nodestring)
 - The [id] value is the nodestring identifier from SMS that represents the structure in the model geometry.
- Linked nodestrings
 - The structure is situated between two nodestrings.
 - The first [ID1] nodestring is upstream and the second nodestrings [ID2] is downstream.
 - For this structure, flow is distributed across a nodestring by cell width and depth (WH^{1.5}).
- Cell
 - The structure is a single cell.
 - The [id] value the cell identifier.
- Zone
 - The structure is a series of cells, defined by a polygon.
 - No [id] value is required.
- Linked zones
 - The structure is situated between two zones.
 - No [id] value is required.
- Autoweir
 - This structure identifies all cell faces (not nodestrings) in the model domain that are elevated above the adjacent cells. These cell faces are then assigned a weir flow condition.

Refer to Section 4.3 for more details.

Name == <sname>

(Optional)

Name of structure

Flux function == <fluxtype>

(Optional)

If structype = nodestring or structype = linked nodestrings then the flux function type defines the flux (or flow).

Flux function type can be:

- Culvert:

-
- A culvert structure.
 - See [culvert file](#) and Section 4.4.2 for more details.
 - Porous:
 - A porous structure (Darcy flow conditions)
 - Timeseries:
 - A specified timeseries of flow (see [flux file](#)).
 - Matrix:
 - A hQh relationship defines the structure (contained in the flux file). For this structure, flow is distributed across a nodestring by cell width and depth (WH^{1.5}). (see [flux file](#) and Section 4.4.1.2).
 - Wall:
 - a solid wall (Q=0)
 - Weir:
 - A broad crested weir structure with a fixed crest level
 - Crest level is specified in the [properties](#) command.
 - See Section 4.4.3 for more details.
 - Weir_dz:
 - A broad crested weir structure with a crest level dz above existing bed levels
 - Weir crest levels are specified by:
 - A nodestring polyline with a “Z” column specification
 - If not using a nodestring polyline, weir crest is the highest of the 2dm file vertices and any additionally specified cell elevations.
 - An additional increment dz, in the [properties](#) command (can be dz = 0).
 - Weir_adjust:
 - A broad crested weir structure with an adjustable crest level
 - Crest level is specified in the [properties](#) command.
 - [Control types](#) are used to specify how the weir elevation should be varied, either by time series from a trigger location/water level from somewhere within the model domain, or from the start of the model simulation.
 - See Section 4.4.3 for more details.
 - Weir_dz_adjust:
 - A broad crested weir structure with an adjustable crest level dz above existing bed levels.
 - [Control types](#) are used to specify how the weir elevation should be varied, either by time series from a trigger location/water level from somewhere within the model domain, or from the start of the model simulation.
 - See Section 4.4.3 for more details.

Flux type == <fluxtype>

(Optional)

Same as [Flux function](#).

Culvert file == <culvertfile(.csv), ID>

(Optional)

Required if [Flux function](#) == Culvert

Reads in a comma separated variable file (csv) with properties for a list of culverts.

- Culvertfile is the file containing a list of culvert properties.
- ID identifies the specific culvert properties from the list of culverts in culvertfile.

The file contains a header line with column labels. Each subsequent line contains the property values listed in Table 4-6. Refer to Section 4.4.2 for more details.

Flux file == <hQhfile(.csv)>

(Optional)

Required if fluxtype = matrix.

The flux file is a comma separated variable file with the hQh flux matrix, defining discharge for a combination of upstream and downstream water levels.

It contains header lines (as many header lines as desired but with no more than 2 commas in each line), then a matrix as follows:

- First row is a list of upstream water levels
- First column is a list of downstream water levels
- Matrix is discharge values corresponding to the listed water levels (corresponding row for downstream, corresponding column for upstream).
- The first value on the first line is a scale factor, which is applied to the Q values in the matrix.

An example of a CSV file is shown below. Refer to Section 4.4.1.2 for more details.

```
Yds, Yus
1., 0., 2., 3., 5.
0., 0., 10., 100., 125.
1., 10., 30., 100., 125.
2., 20., 50., 100., 125.
4., 30., 10., 100., 125.
```

Properties == <p1, . . . , pn>

(Optional)

If fluxtype = “Weir” or “Weir_dz”, then

- P1 = weir crest level (for a weir) or level above existing bed levels
 - for weir there is no default (level required)
 - for weir_dz, default = 0.0
- P2 = weir coefficient
 - default = 1.6

If structype = “Autoweir”, then

- P1 = threshold elevation difference, where the autoweir is activated when the minimum of the face vertices elevations are P1 higher than the adjacent cell elevations
 - default = 0.1 m or ft
- P2 = weir coefficient
 - default = 1.6

If fluxtype type = “Porous” then

- P1 = Porous structure hydraulic conductivity
- P2 = Porous structure width

If celltype = “Bubbler” then

-
- P1 = elevation of bubbler
 - P2 = air flow rate of bubbler (m^3/s)
 - P3 = alpha
 - P4 = b1
 - P5 = Lr
 - P6 = gamma
-

Control == <controltype>

(Optional)

Specification of structure logic definition.

If the structure fluxtype= weir_adjust or weir_dz_adjust, or the structure celltype = zb_adjust or dzb_adjust, then options available are:

- Trigger
- Time series
- Sample_rule
- Target_rule
- Fully_open

Refer to Section 4.4.4 for more details.

Control parameter == <controlparam>

(Optional)

Specification of the parameter that will be controlled.

Options available are:

- Fraction_open
- Min_flow
- Weir_crest
- Zb
- Dzb

Refer to Section 4.4.4 for more details.

Control file == <cfile(.csv)>

Reads in a comma separated file (.csv) with structure controls.

The file contains a header line with specific column labels required for specific structure types:

If cell function == zb_adjust

- Column headers = “Time, zb”

If flux function == weir_adjust

- Column headers = “Time, weir_crest”

If cell function == dzb_adjust

If flux function == weir_dz_adjust

-
- Column headers = “Time, dzb”

If control == trigger

If control == time series

- Column headers = “Time, ControlParameter” (e.g. “Time, Fraction_open”)

If control == sample

- Column headers = “Sample_value, ControlParameter” (e.g. “Sample_value, Fraction_open”)

If control == target

- Column headers = “, Target_deficit, ControlParameter” (e.g. “Target_deficit, Fraction_open”)

The “Time” values are specified as:

- If control == trigger
 - Time (hr) from the moment that the structure adjustment commences.
- If control == time series
 - Time (hr) from the start of the model simulation.

Refer to Section 4.4.4 for more details.

Control update dt == <cdt (hours)>

(Optional)

The frequency of updating the control structure operation (hours).

Refer to Section 4.4.4 for more details.

Sample point == <spx, spy>

(Optional)

spx, spy defines the location that controls the variable z value structure (i.e. the “control” point)

Commands commonly used in conjunction with Sample point are Sample type and Sample dt.

Refer to Section 4.4.4 for more details.

Sample type == <st>

(Optional)

Defines the model parameter which is used for the sampling.

Options available are:

- WL: Water level is currently the only supported sample type.

Refer to Section 4.4.4 for more details.

Sample dt == <sdt (hours)>

(Optional)

The frequency of updating the variable structure (hours).

Refer to Section 4.4.4 for more details.

Max opening increment == <moi>

Maximum change in structure CFL

fraction open (See [control](#)) per update time step (see [Sample dt](#)). Refer to Section 4.4.4 for more details.

Trigger value == <tv>

The value of the specified model parameter at the [Sample point](#) spx, spy that, when exceeded, will trigger a change in structure elevations.

Note that currently the trigger value can only be an absolute water level. Refer to Section 4.4.4 for more details. Command is commonly used in conjunction with [Control file](#) and [Polygon file](#)

Target file == <tfile(.csv)>

Reads in a comma separated file (.csv) defining the target value for the [Sample Type](#)

- Column headers = "Time", "Target_Value"

Refer to Section 4.4.4 for more details.

Bed adjust == <celltype>

(Optional)

If [structype](#) = cell or zone then a bed adjust command can be used.

Bed adjust function type can be:

- ZB_adjust:
 - Adjustable bed elevations for a series of cells with a specified crest level
- DZB_adjust:
 - Adjustable bed elevations for a series of cells with a specified crest level dz above existing bed levels
- ZB:
 - Bed elevations for a series of cells
- DZB:
 - Bed elevations for a series of cells with a level dz above existing bed levels

A control specification is required to initiate bed adjustments (See [control](#)). Refer to Section 4.4.4 for more details.

Cell function == <celltype>

(Optional)

Same as [Bed adjust](#)

Polygon file == <polyfile(.csv)>

(Optional)

Reads in a comma separated variable file defining the perimeter vertices of a polygon.

The file contains a header line with column labels “x” and “y”, which define the coordinates of points describing the perimeter of the polygon. The definition of points needs to be consecutively listed and can be either clockwise or counter-clockwise. TUFLOW FV searches for cell centres that lie within the polygon.

Commonly used with [Bed adjust](#) or [Cell function](#). Refer to Section 4.4.4 for more details.

Destratification unit == <celltype>

(Optional)

If [structype](#) = cell or zone then a destratification unit can be specified.

Destratification unit function type can be:

- Bubbler:
 - A bubbler structure, parameters as specified in the [properties](#) command
- Pump:
 - TBC
- Jet:
 - TBC

Energy loss function == <energytyp>

(Optional)

If [structype](#) = nodestring or [structype](#) = linked nodestrings then an energy loss function can be specified.

Energy loss function type can be:

- Coefficient:
 - Requiring specification of a [form loss coefficient](#).
- Table:
 - Requiring a hQh relationship to define the structure (see [flux file](#)).

Refer to Section 4.4.1.1 for more details.

Form loss coefficient == <flc>

(Optional)

If [energy loss function](#) = Coefficient, form loss coefficient applied to structure. FLC applies a head loss across a cell face according to the equation:

$$\Delta h = FLC \cdot v^2 / 2g$$

Refer to Section 4.4.1.1 for more details.

Energy loss file == <energyfile(.csv)>

(Optional)

TBC

Blockage file == <blockfile(.csv)>

(Optional)

The blockage file is a comma separated variable file with a relationship of flow fraction and depth, commonly used during modelling of bridge structures (in conjunction with [Form loss coefficient](#)). See Section 4.4.1.1 for details.

The file contains the header line with column labels “Z” and “FRAC”.

- The Z column is a list of elevations (lowest to highest).
- The FRAC column is the fraction of flow (0 to 1) for the vertical section between the corresponding Z value and its previous value.

An example of a CSV file is given below:

Z, FRAC

5.0, 0.9

7.0, 0.9

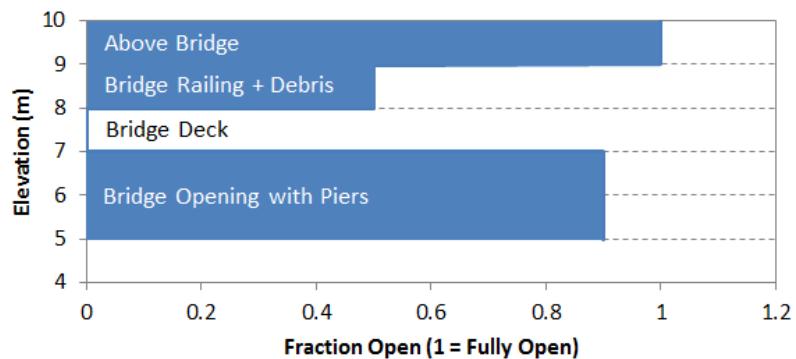
7.1, 0.0

7.9, 0.0

8.0, 0.5

8.9, 0.5

9.0, 1.0



Width file == <widthfile(.csv)>

(Optional)

The width file is a comma separated variable file with a relationship of flow width and depth which is commonly used during modelling of bridge structures (in conjunction with [Form loss coefficient](#)).. Refer to in Section 4.4.1.1 details.

The file contains the header line with column labels “Z” and “WIDTH”.

- The Z column is a list of elevations (lowest to highest).
- The WIDTH column is the width of flow (m or ft – depending on [units](#)) for the vertical section between the corresponding Z value and its previous value.

An example of a CSV file is given below:

Z, WIDTH

5.0, 9.0

7.0, 9.0

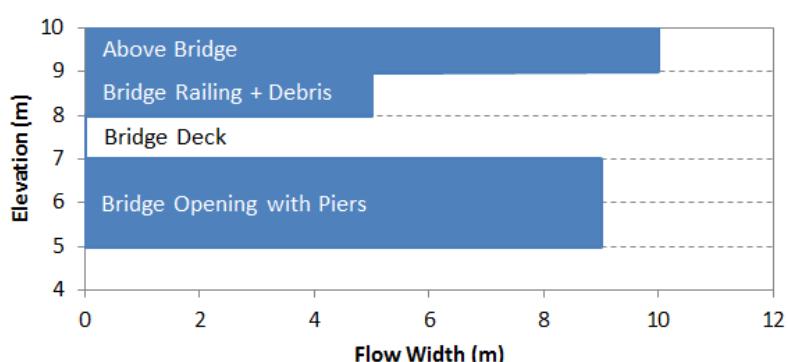
7.1, 0.0

7.9, 0.0

8.0, 5.0

8.9, 5.0

9.0, 10.0



OUTPUT COMMANDS

[Output dir](#)

[Write restart dt](#)

[Restart overwrite](#)

[Output](#)

[Output parameters](#)

[Output statistics](#)

[Vertical averaging](#)

[Output interval](#)

[Start output](#)

[Final output](#)

[Suffix](#)

[Output points file](#)

[Output compression](#)

Output dir == <filepath>

(Optional)

Command to specify the location where simulation output files are to be written. The first example below specifies the output directory assuming the TUFLOW FV sub-folder structure recommended in Section 2.2.1:

```
output dir == ..\output
```

Alternatively, the user may wish the output directory to be located on a local drive, for example:

```
output dir == D:\project12345\tuflowfv\output
```

Output is written to the same location at the simulation control file (.fvc) if this command is not used.

Write restart dt == <time (hours)>

(Optional)

Writes a restart file (.rst) to the [log directory](#) location at the time interval specified. The restart file is binary format and contains the spatially varying conserved variables at an instant in time.

A restart file is used to specify the initial condition for subsequent TUFLOW FV simulations using the [restart](#) command.

Restart overwrite == <0;1>

(Optional, Default == 1)

Option to overwrite the restart file at the time interval specified using the [write restart dt](#) command (default) or create a series of restart files for each timestep:

- 0 = False (i.e. the single restart file will be overwritten)
- 1 = True (i.e. the restart file will not be overwritten and series of restart files will be generated)

Output == <output format>

...

...

...

End output

(Mandatory)

Each output type is defined using an output block. The ‘Output’ and ‘End output’ commands indicate the beginning and end of an output block. The output block specifies the type of output and the output properties including the desired parameters and time definitions. Table 5-1 presents a summary of the the output types available, which include:

- dat
- datv
- flux

-
- mass
 - netcdf
 - netcdfv
 - points
 - profile
 - transport

The commands that can be used within an output block include:

- [Output parameters](#)
- [Output interval](#)
- [Start output](#)
- [Final output](#)
- [Suffix](#)
- [Output points file](#)
- [Output compression](#)

Example output block:

```
output == datv
    output parameters == h, v, d, zb
    output interval == 900
end output
```

Output parameters == <many>

(Mandatory)

[Output](#) block command used within the output block to specify the required output parameters. The available output parameters are summarised in Table 5-2 and Table 5-3 (note that some output parameters are dependent on the output type).

Output statistics == <type 1, type 2>

(Optional)

Output additional requested statistics. The following statistics are currently supported: MAX & MIN. This feature is available with datv and netcdf output types.

Vertical averaging == <type>

```
(Optional 3D, Default == depth-all)
```

Optional output block command to vertically average 3D results over a specified range. The vertical averaging types are:

- depth-all – averaging over entire water column
- depth-range – averaging between specified minimum and maximum absolute depths measured downward from water surface
- height-range – averaging between specified minimum and maximum absolute heights measured upward from the bed
- elevation-range – averaging between specified minimum and maximum elevations relative to model vertical datum
- sigma-range – averaging between specified percentage of the water column where 0 is the water surface and 1 is the bed
- layer-range-top – averaging between layers referenced from the water surface (i.e. surface layer is 1, positive downwards)
- layer-range-bot – averaging between layers referenced from the bed (i.e. bottom layer is 1, positive upwards)

With the exception of depth-all, the vertical averaging type command must be followed by the minimum and maximum limits. For example, the commands for sigma vertical averaging over the top 25% of the water column:

```
output == datv
vertical averaging == sigma-range, 0,0.25      !top 25% of water column
suffix == sigma_0_0.25
output parameters == V
output interval == 1800.
end output
```

Or averaging over the bottom 2m measured upward from the bed:

```
output == datv
vertical averaging == height-range, 0,2      !bottom 2m measured from the bed
suffix == height_0_2
output parameters == V
output interval == 1800.
end output
```

In these examples the [suffix](#) command is used to distinguish between output files. This is particularly important when a simulation control file specifies multiple vertically averaged outputs.

Output Interval == <timestep (s)>

(Optional)

Output block command used to specify the desired output interval in seconds. If this command is not specified output will be produced at each timestep. In many applications this will not be desired (possibly leading to extremely large output files) and an output interval at 10min (600s) or 30min (1800s) will be more appropriate.

Start Output == <time>

(Optional)

Output block command to specify the start time for an output request. The time format must be consistent with the simulation time format. If not specified, the output start time will be consistent with the simulation start time.

Final Output == <time>

(Optional)

Output block command to specify the final time for an output request. The time format must be consistent with the simulation time format. If not specified, the output final time will be consistent with the simulation end time.

Suffix == <suffix>

(Optional)

Output block command to add a suffix to the output filename.

Output points file == <filepath>

(Optional)

Mandatory command when points output type is required. This provides the location and name of a comma separated variable file with the coordinates of the required output points. The following column headers are required in .csv file:

- X, Y, ID (optional)
-

Output compression == <0 ;1>

(Optional, Default == 0)

Output block command to compress netCDF output files:

- 0 = False (i.e. no netCDF file compression)
 - 1 = True (i.e. netCDF file compression)
-

Appendix B – Advection-Dispersion (AD) Module Commands

AD SIMULATION CONFIGURATION COMMANDS

[Scalar mixing model](#)

[Include salinity](#)

[Include temperature](#)

[Equation of state](#)

[Include parallel transport](#)

[Water quality model](#)

[Water quality model dir](#)

[Disable water quality model](#)

[Include heat](#)

[Heat 1h model](#)

[Heat 1w model](#)

[Heat sw model](#)

[Ntracer](#)

Scalar mixing model == <None; Constant; Smagorinsky; Elder; Warmup>

(Optional; Default == None)

Sets the scalar mixing calculation method. See also [global horizontal scalar diffusivity](#).

- None: horizontal scalar mixing is not represented.
- Constant: specify a constant isotropic scalar diffusivity using the [global horizontal scalar diffusivity](#) command.
- Smagorinsky: the horizontal non-isotropic scalar diffusivity is calculated according to the Smagorinsky model - specify the Smagorinsky coefficient using the [global horizontal scalar diffusivity](#) command.
- Elder: the horizontal non-isotropic scalar diffusivity is calculated according to the Elder model - specify the longitudinal and transverse coefficients using the [global horizontal scalar diffusivity](#) command.
- Warm up: can be used for initialising scalar distribution (diffusivity is set to maximum within stability constraints)

Include salinity == <0;1, 0;1>

(Optional 3D; Default == 0,0)

Flag to specify salinity as a modelled parameter:

- 0 = False (i.e. salinity is not modelled).
- 1 = True (i.e. salinity is modelled).

The second flag specifies whether density is a function of the modelled salinity:

- 0 = False (i.e. density is not a function of the modelled salinity).
- 1 = True (i.e. density is a function of the modelled salinity).

Include temperature == <0;1, 0;1>

(Optional 3D; Default == 0,0)

Flag to specify temperature as a modelled parameter:

- 0 = False (i.e. temperature is not modelled).
- 1 = True (i.e. temperature is modelled).

The second flag specifies whether density is a function of the modelled temperature:

- 0 = False (i.e. density is not a function of the modelled temperature).
- 1 = True (i.e. density is a function of the modelled temperature).

Equation of state == <UNESCO; Direct>

(Optional; Default == UNESCO)

Sets the model for calculating the density of water in baroclinic simulations:

- UNESCO: use the UNESCO equation of state (Fofonoff and Miller, 1983).
- Direct: the salinity tracer is assumed to be a direct proxy for density.

Include parallel transport == <0;1>

(Optional; Default == 1, for Spherical coordinate system only)

Optional command used to switch off the parallel transport terms in the momentum flux equations:

- 0 = False (i.e. parallel transport terms are not included).
- 1 = True (i.e. parallel transport terms are included).

These terms ensure that advective tendencies follow great circle paths on the sphere. This will be significant for very large domains (ocean scale) or at high latitudes but may be neglected for smaller domains.

Water quality model == <external>

(Optional)

Optional command to link an external water quality model with TUFLOW FV.

External water quality model dir == <path>

(Optional)

Optional command to specify the directory for external [water quality model](#) definition files if an external water quality model is used. If not specified, external water quality model model files must be located in the same directory at the simulation control file.

Disable water quality model == <0;1>

(Optional; Default == 0)

Optional command used to disable an external water quality model:

- 0 = False (i.e. if specified, external water quality model calculations are enabled).
- 1 = True (i.e. if specified, external water quality model calculations are disabled).

Include heat == <0;1>

(Optional, Default == 0)

Optional command to include atmospheric heat calculations:

- 0 = False (i.e. atmospheric heat calculations not included)

-
- 1 = True (i.e. atmospheric heat calculations included)

Heat lh model == <LHmodel>

(Optional, Default == 1)

Latent heat transfer model:

- 1 = Vapour pressure is calculated by the Magnus-Tetens formula (TVA, 1972) – requires air temperature and relative humidity inputs
- 2 = Vapour pressure is calculated via modified equations based on Lowe (1977) and Reed (1977) – requires air temperature and cloud cover inputs

Heat lw model == <LWinput>

(Optional, Default == 2)

Long wave radiation heat transfer model:

- 1 = Net long wave radiation (accounting for both the incident long wave radiation and long wave radiation emitted by the water surface) – requires net downward long wave radiation input
- 2 = Incident long wave radiation with long wave radiation albedo and water surface reflection are calculated following TVA (1972). Long wave radiation emitted by the water surface is calculated assuming the Stefan-Boltmann law – requires incident downward long wave radiation input
- 3 = Incident long wave radiation is calculated assuming the Stefan-Boltmann law with a correction for cloud cover following TVA (1972) - requires incident downward long wave radiation and cloud cover inputs
- 4 = Incident long wave radiation with corrections for cloud cover and the long wave radiation emitted by the water surface due to the air/water temperature difference following Zillman (1972) - requires incident downward long wave radiation, cloud cover and air temperature inputs
- 5 = Based on air temperature and vapour pressure (Chapra, 2008) – requires air temperature and relative humidity inputs

Heat sw model == <SWinput>

(Optional, Default == 1)

Short wave radiation heat transfer model

- 1 = Incident short wave radiation estimated according to Jacquet (1983) – requires downward short wave radiation input
- 2 = Incident short wave radiation under clear sky estimated according to Zillman (1972) with cloud cover correction factor given by Reed (1977) – requires air temperature and cloud cover inputs

Ntracer == <number of tracers>

(Optional, Default == 0)

Command used to specify the number of tracers in an AD simulation. The properties of each tracer are defined using the [tracer](#) block commands.

AD MODEL PARAMETER COMMANDS

[Reference salinity](#)

[Reference temperature](#)

[Reference density](#)

[Density air](#)

[Heat CP](#)

[Heat CPA](#)

[Heat CLN](#)

[Heat CSN](#)

[Heat water emissivity](#)

[Heat NIR fraction](#)

[Heat UVA fraction](#)

[Heat UVB fraction](#)

[Heat PAR extinction](#)

[Heat NIR extinction](#)

[Heat UVA extinction](#)

[Heat UVB extinction](#)

[Heat SED absorption](#)

[Heat ref height](#)

[Heat albedo sw](#)

[Heat albedo lw](#)

[WQ update dt](#)

[Cell WQ depth](#)

[Heat relax dt](#)

[Atmospheric update dt](#)

Reference Salinity == <Salinity (PSU)>

(Optional; Default == 0.0)

Optionally sets the model reference salinity for simulations including baroclinic terms.

Reference Temperature == <Temperature (°C)>

(Optional; Default == 20.0)

Optionally sets the model reference temperature for simulations including baroclinic terms.

Reference Density == <Density (kg/m³)>

(Optional; Default == 1000.0)

Sets the reference density of water value used in calculation of the baroclinic pressure terms.

Density Air == <Air Density (kg/m³)>

(Optional; Default == 1.2)

Allows the user to specify the density of air used in atmospheric heat calculations.

Heat CP == <Water Specific Heat (J/kg/°C)>

(Optional, Default == 4181.3)

Specific heat capacity of water at 25°C.

Heat CPA == <Air Specific Heat (J/kg/°C)>

(Optional, Default == 1005.0)

Specific heat capacity of dry air at 25°C.

Heat CLN == <CLN>

(Optional, Default == 0.0013)

Bulk aerodynamic latent heat transfer coefficient under neutral conditions.

Heat CSN == <CSN>

(Optional, Default == 0.0013)

Bulk aerodynamic sensible heat transfer coefficient under neutral conditions.

Heat water emissivity == <EPS w>

(Optional, Default == 0.96)

Emissivity of the water surface

Heat NIR fraction == <NIR frac>

(Optional, Default == 0.43)

Fraction of near-infrared (NIR) in short wave radiation.

Heat UVA fraction == <UVA frac>

(Optional, Default == 0.048)

Fraction of ultraviolet A (UVA) in short wave radiation.

Heat UVB fraction == <UVB frac>

(Optional, Default == 0.002)

Fraction of ultraviolet B (UVB) in short wave radiation.

Heat PAR extinction == <PAR eta>

(Optional, Default == 0.25m⁻¹)

Extinction coefficient of photosynthetically active radiation (PAR) in short wave radiation.

Heat NIR extinction == <NIR eta>

(Optional, Default == 1.0m⁻¹)

Extinction coefficient of near-infrared (NIR) in short wave radiation.

Heat UVA extinction == <UVA eta>

(Optional, Default == 1.0m⁻¹)

Extinction coefficient of ultraviolet A (UVA) in short wave radiation.

Heat UVB extinction == <UVB eta>

(Optional, Default == 2.5m⁻¹)

Extinction coefficient of ultraviolet B (UVB) in short wave radiation.

Heat SED absorption == <sed abs>

(Optional, Default == 0.9)

Rate of light absorption by sediments.

Heat ref height == <meters>

(Optional, Default == 10.0)

Meteorological sensor height.

Heat albedo sw == <alb swo>

(Optional, Default == 0.08)

Mean short wave radiation albedo at the equator.

Heat albedo lw == <alb lw>

(Optional, Default == 0.03)

Mean long wave radiation albedo at the equator.

WQ update dt == <timestep (s)>

(Optional)

Specifies the timestep for performing water quality parameter updating, if not specified water quality parameter updating occurs every hydrodynamic model timestep.

Cell WQ depth == <depth (m)>

(Optional)

An optional command to set the threshold water depth for water quality calculations. Water quality calculations are not undertaken in areas where the depth is less than the threshold value.

Transport mode depth == <depth (m)>

(Optional)

An optional command to set the threshold water depth for transport mode AD calculations. AD calculations are not undertaken in areas where the depth is less than the threshold value.

Heat relax dt == <timestep (hour)>

(Optional, Default == 0)

Specifies the heat relaxation timestep in hours.

Atmospheric update dt == <timestep (s)>

(Optional)

Specifies the timestep for performing atmospheric parameter updating, if not specified atmospheric parameter updating occurs every hydrodynamic model timestep.

AD TURBULENCE PARAMETER COMMANDS

[Global horizontal scalar diffusivity](#)

[Global horizontal scalar diffusivity limits](#)

[Global vertical scalar diffusivity](#)

[Global vertical scalar diffusivity limits](#)

[Diffusivity limiter dt](#)

Global Horizontal Scalar Diffusivity == <diffusivity (m²/s) ; coefficient/s (-)>

(Optional)

Globally sets the horizontal diffusivity (m²/s) or diffusivity model coefficients. This is dependent on the scalar mixing model set using the [scalar mixing model](#) command:

- Constant: specify a constant isotropic scalar diffusivity; Default == 0.
- Smagorinsky: specify the Smagorinsky coefficient; Default == 0 (typical value is 0.2)
- Elder: specify longitudinal and transverse coefficients – calculates a non-isotropic diffusivity; typically only used for 2D simulations; Default == 0, 0 (typical values 100, 10)

See [scalar mixing model](#) command to set scalar mixing turbulence model.

Global Horizontal Scalar Diffusivity Limits == <min diffusivity (m²/s)>, <max diffusivity (m²/s)>

(Optional)

For use with Smagorinsky or Elder [scalar mixing model](#), globally sets the minimum and maximum horizontal scalar diffusivity (m²/s) limits.

Not applicable if a Constant isotropic scalar diffusivity is set using the [global horizontal scalar diffusivity](#) command.

See [scalar mixing model](#) command to set scalar mixing turbulence model.

Global Vertical Scalar Diffusivity == <diffusivity (m²/s) ; coefficient/s (-)>

(Optional)

Globally sets the vertical diffusivity (m²/s) or diffusivity model coefficients. This is dependent on the scalar mixing model set using the [scalar mixing model](#) command:

- Constant: specify a constant isotropic scalar diffusivity; Default == 0
- Smagorinsky: specify the Smagorinsky coefficient; Default == 0 (typical value is 0.2)
- Elder: specify longitudinal and transverse coefficients – calculates a non-isotropic diffusivity; typically only used for 2D simulations; Default == 0, 0 (typical values 100, 10)

See [scalar mixing model](#) command to set scalar mixing turbulence model.

Global Vertical Scalar Diffusivity Limits == <min diffusivity (m²/s)>, <max diffusivity (m²/s)>

(Optional)

For use with Smagorinsky or Elder [scalar mixing model](#), globally sets the minimum and maximum vertical scalar diffusivity (m²/s) limits.

Not applicable if a Constant isotropic scalar diffusivity is set using the [global horizontal scalar diffusivity](#) command.

See [scalar mixing model](#) command to set scalar mixing turbulence model.

AD MATERIAL PROPERTIES COMMANDS

[Horizontal scalar diffusivity](#)

[Horizontal scalar diffusivity limits](#)

[Vertical scalar diffusivity](#)

[Vertical scalar diffusivity limits](#)

```
Horizontal scalar diffusivity == <diffusivity (m2/s) ;
coefficient (-)>
```

(Optional, Default == value set using [global horizontal scalar diffusivity](#) command)

[Material](#) block command to specify the horizontal scalar diffusivity value (m²/s) or model coefficients for cells with material id# (thereby overriding the default or corresponding global turbulence parameters), depending on the scalar mixing model used. See [scalar mixing model](#) command to set momentum mixing turbulence model.

```
Horizontal scalar diffusivity limits == <ds_limit1, ds_limit2>
```

(Optional)

[Material](#) block command for use with Smagorinsky and Elder [scalar mixing model](#) to set the minimum and maximum horizontal scalar diffusivity (m²/s) limits for cells with material id# (thereby overriding the default or corresponding globally set parameters).

```
Vertical scalar diffusivity == <diffusivity (m2/s) ; coefficient (-)>
```

(Optional, Default == value set using [global vertical scalar diffusivity](#) command)

[Material](#) block command to specify the vertical scalar diffusivity value (m²/s) or model coefficients for cells with material id# (thereby overriding the default or corresponding global turbulence parameters), depending on the scalar mixing model used. See [scalar mixing model](#) command to set momentum mixing turbulence model.

```
Vertical scalar diffusivity limits == <ds_limit1, ds_limit2>
```

(Optional)

[Material](#) block command for use with Smagorinsky and Elder [scalar mixing model](#) to set the minimum and maximum vertical scalar diffusivity (m²/s) limits for cells with material id# (thereby overriding the default or corresponding globally set parameters).

AD TRACER COMMANDS

[Tracer](#)

[Settling velocity](#)

[Decay rate](#)

```
Tracer == <tracer id #>
```

```
...
```

```
end tracer
```

(Optional)

This command indicates the beginning of a tracer properties block, specifying the tracer id # that the properties should be applied to. Tracer properties are listed in the following rows and the ‘end tracer’ command is used to indicate the end of the tracer block.

Tracer properties include:

- [Settling Velocity](#)
- [Decay Rate](#)

Example Tracer Block:

```
tracer == 2
settling velocity == 1.0e-5
decay rate == 0.05
end tracer
```

```
Settling Velocity == <ws0 (m/s)>
```

(Optional)

[Tracer](#) block command to specify the scalar settling velocity in m/s. This results in a sink term flux, S :

$$S = -w_{s0}C$$

where C is the scalar concentration.

```
Decay Rate == <Kd (units/day)>
```

(Optional)

[Tracer](#) block command to specify the scalar decay rate in concentration units/day. This results in a sink term flux, S :

$$S = K_dCh$$

where C is the scalar concentration and h is the flow depth.

AD INITIAL CONDITION COMMANDS

[Initial salinity](#)

[Initial temperature](#)

[Initial scalar profile](#)

[Initial tracer concentration](#)

[Initial WQ concentration](#)

See also:

[Initial condition 2d](#)

[Initial condition 3d](#)

Initial Salinity == <salinity (psu)>

(Optional; Default == 0.0)

Globally sets the initial salinity for simulations including baroclinic terms.

Initial Temperature == <temperature (degrees Celsius)>

(Optional; Default == 0.0)

Globally sets the initial temperature for simulations including baroclinic terms.

Initial tracer concentration == <t_1, ..., t_Nwq>

(Optional)

Globally sets the initial tracer concentration fields.

Initial WQ concentration == <wq_1, ..., wq_Nwq>

(Optional)

Globally sets the initial water quality scalar concentration fields.

Initial Scalar Profile == <initial condition file (.csv)>

(Optional 3D)

Command used in conjunction with [initial condition 2d](#) to specify a .csv file that describes the initial scalar profile.

The .csv file should contain two columns:

- The first column is the depth reference.
- The second column is the concentration at the corresponding depth reference.

If salinity, temperature are included in the simulation they should also be specified in the .csv file (e.g. Sal, Temp, Scal_1,...). An example of the command usage and corresponding .csv file is given below:

Initial condition 2d == ..\bc\initial_scalar_profile_001.csv

and the contents of initial_conditions.csv:

```
DEPTH, SAL, TEMP, SCAL_1
0.0, 30, 20, 100
2.0, 32, 19, 50
2.1, 35, 17, 25
9999.0, 35, 17, 25
```

Appendix C – TUFLOW FV netCDF Output File Description

The dimensions, variable definitions and attributes of a TUFLOW FV netCDF 3D output file are provided below. This information is intended to assist advanced users wishing to develop functions and scripts to post-process and/or view TUFLOW FV output using a numerical analysis package with a netCDF library interface (such as MATLAB, R, GNU Octave or Python NumPy).

```

Source: C:\TUFLOWFV\output\TUFLOWFV_netcdf_3d_output.nc
Format: 64bit
Global Attributes:
    Origin      = 'Created by TUFLOWFV'
    Type        = 'Cell-centred TUFLOWFV output'
    spherical   = 'true'
Dimensions:
    NumCells2D      = 38839
    NumCells3D      = 386802
    NumVert2D       = 36790
    NumVert3D       = 378581
    MaxNumCellVert  = 4
    NumLayerFaces3D = 425641
    NumSedFrac     = 1
    Time           = 13441 (UNLIMITED)
Variables:
    RestTime
        Size:          13441x1
        Dimensions:   Time
        Datatype:      double
        Attributes:
            long_name = 'output time relative to 01/01/1990
00:00:00'
            units       = 'hours'
    cell_Nvert
        Size:          38839x1
        Dimensions:   NumCells2D
        Datatype:      int32
        Attributes:
            long_name = 'Cell number of vertices'
    cell_node
        Size:          4x38839
        Dimensions:   MaxNumCellVert, NumCells2D
        Datatype:      int32
        Attributes:
            long_name = 'Cell node connectivity'
    NL
        Size:          38839x1
        Dimensions:   NumCells2D
        Datatype:      int32
        Attributes:
            long_name = 'Number of layers in profile'
    idx2
        Size:          386802x1
        Dimensions:   NumCells3D
        Datatype:      int32
        Attributes:
            long_name = 'Index from 3D to 2D arrays'
    idx3
        Size:          38839x1
        Dimensions:   NumCells2D
        Datatype:      int32
        Attributes:
            long_name = 'Index from 2D to 3D arrays'
    cell_X
        Size:          38839x1

```

```
Dimensions: NumCells2D
Datatype:    single
Attributes:
    long_name = 'Cell Centroid X-Coordinate'
    units      = 'm'

cell_Y
Size:        38839x1
Dimensions:  NumCells2D
Datatype:    single
Attributes:
    long_name = 'Cell Centroid Y-Coordinate'
    units      = 'm'

cell_Zb
Size:        38839x1
Dimensions:  NumCells2D
Datatype:    single
Attributes:
    long_name = 'Cell Bed Elevation'
    units      = 'm'

cell_A
Size:        38839x1
Dimensions:  NumCells2D
Datatype:    single
Attributes:
    long_name = 'Cell Area'
    units      = 'm^2'

node_X
Size:        36790x1
Dimensions:  NumVert2D
Datatype:    single
Attributes:
    long_name = 'Node X-Coordinate'
    units      = 'm'

node_Y
Size:        36790x1
Dimensions:  NumVert2D
Datatype:    single
Attributes:
    long_name = 'Node Y-Coordinate'
    units      = 'm'

node_Zb
Size:        36790x1
Dimensions:  NumVert2D
Datatype:    single
Attributes:
    long_name = 'Node Bed Elevation'
    units      = 'm'

layerface_Z
Size:        425641x13441
Dimensions:  NumLayerFaces3D,Time
Datatype:    single
Attributes:
    long_name = 'Layer Face Z-Coordinates'
    units      = 'm'

stat
Size:        38839x13441
Dimensions:  NumCells2D,Time
Datatype:    int32
```

```
Attributes:
    long_name = 'Cell wet/dry status'
    units      = 'boolean'

H
Size:          38839x13441
Dimensions:   NumCells2D,Time
Datatype:     single
Attributes:
    long_name = 'water surface elevation'
    units      = 'm'

V_x
Size:          386802x13441
Dimensions:   NumCells3D,Time
Datatype:     single
Attributes:
    long_name = 'x_velocity'
    units      = 'm s^-1'

V_y
Size:          386802x13441
Dimensions:   NumCells3D,Time
Datatype:     single
Attributes:
    long_name = 'y_velocity'
    units      = 'm s^-1'

W
Size:          386802x13441
Dimensions:   NumCells3D,Time
Datatype:     single
Attributes:
    long_name = 'vertical velocity'
    units      = 'm s^-1'

SAL
Size:          386802x13441
Dimensions:   NumCells3D,Time
Datatype:     single
Attributes:
    long_name = 'salinity'
    units      = 'psu'

TEMP
Size:          386802x13441
Dimensions:   NumCells3D,Time
Datatype:     single
Attributes:
    long_name = 'temperature'
    units      = 'degrees celsius'
```

Appendix D – Common netCDF Input File Examples

The dimensions, variable definitions and attributes of common TUFLOW FV netCDF input files are provided below. This information is intended to assist users wishing to apply temporally and spatially varying boundary conditions. Two examples are provided:

1. SWAN wave model output
2. NCEP Reanalysis II atmospheric data

1. SWAN Wave Model netCDF Output Boundary Condition Example

Wave forcing is often important when simulating the advection-diffusion of sediments or water-borne constituents in estuarine or coastal environments. Mapped, time-varying SWAN wave model output in netCDF format can be used as a boundary condition for a subsequent TUFLOW FV advection-diffusion simulation. The dimensions, variable definitions and attributes of a SWAN netCDF output file are provided below and would typically contain the following variables:

- significant wave height (hs)
- surface peak period ‘smoothed’ (tps)
- surface mean direction (theta0)
- x-component of the wave induced force (xforce)
- y-component of the wave induced force (yforce)
- near bottom orbital velocity (ubot)
- near bottom period (tmbot)

The TUFLOW FV boundary condition block commands to read the SWAN netCDF output file and include the parameters in the advection-diffusion calculations are:

```
grid definition file == ..\bc\waves\SWAN_output.nc
grid variables == longitude, latitude
grid definition label == SWAN_wavesRegional

bc == wave, SWAN_wavesRegional, ..\bc\waves\SWAN_output.nc
bc header == time, hs, tps, theta0, ubot, tmbot, xforce, yforce
bc reference time == 01/01/1970 00:00
bc time units == seconds
bc update dt == 3600
end bc
```

The variables listed following the ‘bc header’ command should follow the order in the example provided above. Specification of the ‘bc reference time’ and the ‘bc time units’ is crucial since SWAN and TUFLOW FV have differing default time formats. Without this information TUFLOW FV would not apply the SWAN output at the intended time. The ‘bc update dt’ specifies the time interval in seconds between wave field updates, generally consistent with the temporal resolution of the SWAN output.

It is not a requirement for ubot, tmbot, xforce and yforce to be included in the SWAN netCDF output file. If not specified, these variables will be either approximated by TUFLOW FV or simply not included in the advection-diffusion calculation. As a minimum, the following variables should be specified:

```
bc header == time, hs, tps, theta0
```

In this example, TUFLOW FV would approximate the near bottom orbital velocity (ubot) using the available parameters and following linear wave theory and the surface peak period (tps) would be applied as the near bottom period (tmbot). If not specified, the wave induced forces (xforce, yforce) are not approximated or used by TUFLOW FV.

In situations where ubot and tmbot are present in the SWAN netCDF output file but the user wishes to use the TUFLOW FV approximations, the ‘dummy’ command should be used:

```
bc header == time, hs, tps, theta0, dummy, dummy, xforce, yforce
```

Recent versions of SWAN source code and many SWAN binary distributions for Windows support netCDF model output. More information may be found on the SWAN website:

<http://swanmodel.sourceforge.net/>

SWAN netCDF output file structure

Source:

C:\TUFLOWFV\bc\waves\SWAN_output.nc

64bit

Global Attributes:

Conventions = 'CF-1.5'
History = 'Created with agioncmd version 1.2'
Directional convention = 'cartesian'
project = '001'
run = '001'

Dimensions:

time = 18633 (UNLIMITED)
xc = 251
yc = 651

Variables:

time

Size: 18633x1

Dimensions: time

Datatype: int32

Attributes:

units = 'seconds since 1970-01-01'
calendar = 'gregorian'
standard_name = 'time'
long_name = 'time'

longitude

Size: 251x651

Dimensions: xc,yc

Datatype: single

Attributes:

units = 'degrees_east'
long_name = 'longitude'
standard_name = 'longitude'

latitude

Size: 251x651

Dimensions: xc,yc

Datatype: single

Attributes:

units = 'degrees_north'
long_name = 'latitude'
standard_name = 'latitude'

hs

Size: 251x651x18633

Dimensions: xc,yc,time

Datatype: int16

Attributes:

units = 'm'

standard_name =

'sea_surface_wave_significant_height'

long_name = 'hs'

coordinates = 'longitude latitude'

_FillValue = -3.28e+04

scale_factor = 0.000763

add_offset = 25

tps

Size: 251x651x18633

Dimensions: xc,yc,time

```

Datatype: int16
Attributes:
    units      = 's'
    long_name = 'tps'
    coordinates = 'longitude latitude'
    _FillValue = -3.28e+04
    scale_factor = 0.000763
    add_offset   = 25
theta0
Size: 251x651x18633
Dimensions: xc,yc,time
Datatype: int16
Attributes:
    units      = 'degrees'
    standard_name = 'sea_surface_wave_to_direction'
    long_name = 'theta0'
    coordinates = 'longitude latitude'
    _FillValue = -3.28e+04
    scale_factor = 0.00549
    add_offset   = 180
xforce
Size: 251x651x18633
Dimensions: xc,yc,time
Datatype: int16
Attributes:
    units      = 'N m-2'
    long_name = 'x-component of wave driven force
per unit surface area'
    coordinates = 'longitude latitude'
    _FillValue = -3.28e+04
    scale_factor = 0.0305
    add_offset   = 0
yforce
Size: 251x651x18633
Dimensions: xc,yc,time
Datatype: int16
Attributes:
    units      = 'N m-2'
    long_name = 'y-component of wave driven force
per unit surface area'
    coordinates = 'longitude latitude'
    _FillValue = -3.28e+04
    scale_factor = 0.0305
    add_offset   = 0
ubot
Size: 251x651x18633
Dimensions: xc,yc,time
Datatype: int16
Attributes:
    units      = 'm s-1'
    long_name = 'orbital velocity near bottom'
    coordinates = 'longitude latitude'
    _FillValue = -3.28e+04
    scale_factor = 0.000229
    add_offset   = 7.5
tmbot
Size: 251x651x18633
Dimensions: xc,yc,time

```

```
Datatype:    int16
Attributes:
    units          = 's'
    long_name     = 'Bottom wave period'
    coordinates   = 'longitude latitude'
    _FillValue    = -3.28e+04
    scale_factor  = 0.000763
    add_offset    = 25
```

2. NCEP-DOE Reanalysis 2 Atmospheric Data Boundary Condition Example

For TUFLOW FV 3D simulations, the baroclinic pressure-gradient terms can be optionally activated to allow the hydrodynamic solution to respond to temperature, salinity and sediment induced density gradients. In addition, atmospheric (surface) heat exchange calculations can also be included for given standard meteorological parameter inputs.

In the example below mapped, time-varying atmospheric data derived as part of the NCEP-DOE Reanalysis 2 project (website: <http://www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis2.html>) were downloaded and post-processed to create a single netCDF file containing the following surface variables:

- u-component of the 10-minute average wind velocity (u)
- v-component of the 10-minute average wind velocity (v)
- air temperature (temp)
- relative humidity (rhum)
- downward shortwave solar radiation (dswr)
- downward long-wave non-penetrative radiation (dlwr)
- precipitation (rain)

The TUFLOW FV boundary condition block commands to read the netCDF file containing the NCEP-DOE Reanalysis 2 data and include the parameters in the heat exchange calculations are:

```
grid definition file == ..\bc\atmospheric\NCEP_DOE_R2_data.nc
grid variables == lon, lat
grid definition label == NCEP

bc == W10_GRID, NCEP, ..\bc\atmospheric\NCEP_DOE_R2_data.nc
bc header == time,u,v
bc reference time == 01/01/1990 00:00
bc time units == hours
bc update dt == 14400
end bc

bc == AIR_TEMP_GRID, NCEP, ..\bc\atmospheric\NCEP_DOE_R2_data.nc
bc header == time,temp
bc reference time == 01/01/1990 00:00
bc time units == hours
bc update dt == 14400
end bc
```

```
bc == REL_HUM_GRID, NCEP, ..\bc\atmospheric\NCEP_DOE_R2_data.nc
  bc header == time,rhum
  bc reference time == 01/01/1990 00:00
  bc time units == hours
  bc update dt == 14400
end bc

bc == SW_RAD_GRID, NCEP, ..\bc\atmospheric\NCEP_DOE_R2_data.nc
  bc header == time,dswr
  bc reference time == 01/01/1990 00:00
  bc time units == hours
  bc update dt == 14400
end bc

bc == LW_RAD_GRID, NCEP, ..\bc\atmospheric\NCEP_DOE_R2_data.nc
  bc header == time,dlwr
  bc reference time == 01/01/1990 00:00
  bc time units == hours
  bc update dt == 14400
end bc

bc == PRECIP_GRID, NCEP, ..\bc\atmospheric\NCEP_DOE_R2_data.nc bc header == time,rain
  bc reference time == 01/01/1990 00:00
  bc time units == hours
  bc update dt == 14400
end bc
```

With the exception of precipitation¹¹, the variables provided in this example correspond to the inputs required by the TUFLOW FV heat exchange module in default mode. Additional inputs, such as cloud cover, may be required when activating non-default atmospheric module settings. A full description of the heat exchange calculation options is available in the TUFLOW FV Science Manual.

¹¹ Precipitation is an optional input (assumed freshwater) for baroclinic mode simulations.

2. Meteorological Grid netCDF Input File Example

Source: C:\TUFLOWFV\bc\atmospheric\NCEP_DOE_R2_data.nc

Format: classic

Global Attributes:

```
origin = 'NCEP Reanalysis'
```

Dimensions:

```
ni      = 11
nj      = 14
time   = 1464  (UNLIMITED)
```

Variables:

time

```
Size:          1464x1
Dimensions:    time
Datatype:      double
Attributes:
    units      = 'hours'
    longname   = 'time' in decimal hours since
01/01/1990 00:00'
    reference  = 'AEST'
```

lon

```
Size:          11x1
Dimensions:    ni
Datatype:      single
Attributes:
    units      = 'degrees'
    longname   = 'longitude'
    projection = 'LL_WGS84'
```

lat

```
Size:          14x1
Dimensions:    nj
Datatype:      single
Attributes:
    units      = 'degrees'
    longname   = 'latitude'
    projection = 'LL_WGS84'
```

u

```
Size:          11x14x1464
Dimensions:    ni,nj,time
Datatype:      single
Attributes:
    longname = 'u'
    units    = 'm s^-1'
```

v

```
Size:          11x14x1464
Dimensions:    ni,nj,time
Datatype:      single
Attributes:
    longname = 'v'
    units    = 'm s^-1'
```

temp

```
Size:          11x14x1464
Dimensions:    ni,nj,time
Datatype:      single
Attributes:
    longname = 'air temperature'
```

```
                units      = 'degC'
rhum
Size:      11x14x1464
Dimensions: ni,nj,time
Datatype:   single
Attributes:
            longname = 'relative humidity'
            units    = 'percent'

dswr
Size:      11x14x1464
Dimensions: ni,nj,time
Datatype:   single
Attributes:
            longname = 'downward shortwave radiation'
            units    = 'W m^-2'

dlwr
Size:      11x14x1464
Dimensions: ni,nj,time
Datatype:   single
Attributes:
            longname = 'downward longwave radiation'
            units    = 'W m^-2'

rain
Size:      11x14x1464
Dimensions: ni,nj,time
Datatype:   single
Attributes:
            longname = 'precipitation'
            units    = 'm d^-1'
```

Appendix E – External Turbulence Model Interface

Appendix F – External Water Quality Model Interface