TUFLOW FV 2014-01 Release Notes

We are pleased to announce the 2014-01 release. Notes on the changes, new features and enhancements to TUFLOW FV are provided within this document.

Should you require clarification or more detail on any of the points below, please email support@tuflow.com.

Table of Contents

TUFLOW

2014-01 Release Overview

The 2014-01 release includes some major new features and range of enhancements. They include:

- A new linux build.
- More commands for defining model geometry, including cell elevation specification and specification of nodestrings independently from the 2dm mesh file.
- A number of new and advanced boundary conditions.
- A new “structures module” which encompasses a range of The term ‘Structure’ encompasses a range of hydraulic structures (bridges, culverts and weirs) and also topography/bathymetry control commands.
- A number of new and advanced output types, including flexible depth-averaging of 3D results into 2D sheets and tracking of maximum and minimum parameter values.
- A new water quality module.
- A matlab toolbox for handling TUFLOW FV output and in particular for visualising 3D model output.
- A greatly expanded set of support resources, including a forum page, wiki page, updated user manual and downloadable tutorial models.
- Support for Aquaveo demo licensing (no hardware lock required).
- Tutorial models can be run with no licensing.

New Linux Build

1 A linux build of the TUFLOW FV executable is now available from the download page. This is packaged as a rpm for Redhat and Centos 6 x86_64.

The installation process should be:

1. Download and install codemeter service

   http://www.wibu.com/downloads-user-software.html

   >> rpm -ivh CodeMeter64-5.10.1224-501.x86_64.rpm

2. Install tuflowfv, e.g.

   >> rpm -ivh --nodeps --prefix=<installdir> tuflowfv-dev-1.x86_64.rpm

3. Set PATH and LD_LIBRARY_PATH environment variables, e.g.

   >> PATH=<installdir>/bin:$PATH

   >> LD_LIBRARY_PATH=<installdir>/lib:$LD_LIBRARY_PATH

4. Run tuflowfv

Model Geometry Definition

2 A number of new methods for specifying the **cell-centred model bathymetry**, have been added. These commands allow the user to overwrite the bathymetry specified in the mesh definition file (geometry 2D == <filepath>).

(a) **Cell elevation file == <filepath>**. The cell elevation file is a comma-separated-value file that specifies cell elevation updates based on specified cell IDs or on specified coordinates. Note that only cells that are directly specified (by ID or directly containing a point specified by coordinate) will be updated. There is no interpolation of elevations between the specified points.

(b) **Cell elevation polyline file == <filepath>, <polyline_ID>**. The cell elevation polyline file is a comma-separated-value file that specifies cell elevation updates along a polyline (defined by 2 or more X, Y, Z coordinates). These are interpolated to all cells that are intersected by the polyline. Multiple polylines may be stored in a single file, and a particular polyline can be specified with a unique polyline_ID.

(c) **Cell elevation polygon file == <filepath>, <polygon_ID>, <Z>**. The cell elevation polygon file is a comma-separated-value file that can define a region of a model mesh that will be updated with the specified Z value. Multiple polygons may be stored in a single file, and a particular polygon can be specified with a unique polygon_ID.

(d) **Global Bed elevation limits == <minimum, maximum>**. This command allows the user to specify global limits to the cell-centred model bathymetry values. These limits will potentially override any cell elevations specified in the mesh definition file (geometry 2D == <filepath>).

(e) **Bed elevation limits == <minimum, maximum>**. Within a material block, this command allows the user to specify limits to the cell-centred bathymetry values for this particular material type/s.

3 Nodestrings can now be specified within the tuflow fv control file in addition to the mesh definition file (geometry 2D == <filepath>) using the command: **Nodestring polyline file == <filepath>, <polyline_ID>, <boundary>**. This allows the user to manage the input of nodestrings (particularly those related to hydraulic structure definitions) independently of the mesh definition. The nodestring polyline file is a comma-separated-value file that specifies the nodestring path (X, Y coordinates) and optionally the nodestring elevation. Multiple polylines may be stored in a single file, and a particular polyline can be specified with a unique polyline_ID. The optional boundary switch specifies whether the nodestring should adhere to a mesh boundary.

4 Additional echo geometry outputs have been added, when processing and initialising the model geometry, model outputs and model structure definitions. These additional outputs are written to one or more comma-separated-values files written to the runtime log directory. The netcdf echo geometry file may still be written, and all echo geometry files are enabled by default. They may be suppressed with the following commands: **echo geometry == 0** (turns off all echo geometry files); **echo geometry netcdf == 0** (turns off netcdf formatted file); **echo geometry csv == 0** (turns off csv formatted file/s).

Boundary Conditions

5 A summary of the currently available boundary conditions is provided below:

(a) **BC == QN, <NS_ID>, <friction slope>.** "Normal flow" open boundary condition at boundary nodestring (NS_ID). A friction slope is also required.

(b) **BC == WL, <NS_ID>, <filepath>**. Water level timeseries open boundary condition at boundary nodestring (NS_ID). The water level timeseries is specified in a comma-separated-value file.

(c) **BC == Q, <NS_ID>, <filepath>**. Flow timeseries open boundary condition at boundary nodestring (NS_ID). The flow timeseries is specified in a comma-separated-value file.

(d) **BC == WLS, <NS_ID>, <filepath>**. Sloping water level timeseries open boundary condition at boundary nodestring (NS_ID). The water level timeseries is specified in a comma-separated-value file.

(e) **BC == OBC, <NS_ID>, <filepath>**. General Open Boundary Condition (OBC) at boundary nodestring (NS_ID). Water level, velocity and scalar concentration timeseries are specified in a comma-separated-value file. The basic OBC boundary condition does not vary spatially. If vertically or horizontally varying boundary conditions are required then either OBC_prof (profile) or OBC_curt (curtain) boundary conditions can be specified.

(f) **BC == SCALAR, <NS_ID>, <filepath>**. Scalar concentration open boundary condition at boundary nodestring (NS_ID). The scalar concentration timeseries is specified in a comma-separated-value file.

(g) **BC == OBC_PROF, <NS_ID>, <filepath>**. General OBC profile (allowing for vertical variation) at boundary nodestring (NS_ID). Water level timeseries as well as velocity and scalar concentration profile timeseries are specified in a netcdf format file (refer user manual for details).

(h) **BC == SCALAR_PROF, <NS_ID>, <filepath>**. General OBC profile at boundary nodestring (NS_ID). Scalar concentration profile timeseries are specified in a netcdf format file (refer user manual for details).

(i) **BC == OBC_CURT, <NS_ID>, <filepath>**. General OBC curtain (allowing for horizontal as well as vertical variation) at boundary nodestring (NS_ID). Water level timeseries as well as velocity and scalar concentration curtain timeseries are specified in a netcdf format file (refer user manual for details).

(j) **BC == OBC_GRID, <GRID_ID>, <filepath>**. General OBC curtain interpolating from 3D gridded hydrodynamic model dataset. Water level timeseries as well as velocity and scalar concentration timeseries are specified in a 3D gridded netcdf format file (refer user manual for details). These are automatically interpolated onto the boundary nodestrings specified in a **BC nodestrings == <ns_id1>,<ns_id2>,…** command. If this boundary condition is specified then an **Initial Condtion OGCM** command can be used as a model initial condition.

(k) Atmospheric data boundary conditions can now be specified as either globally applied timeseries or as spatially and temporally varying gridded datasets (refer user manual for details):

(i) **BC == W10, <filepath>**

(ii) **BC == PRECIP, <filepath>**

(iii) **BC == AIR_TEMP, <filepath>**

(iv) **BC == REL_HUM, <filepath>**

(v) **BC == SW_RAD, <filepath>**

(vi) **BC == LW_RAD, <filepath>**

(vii) **BC == W10, <filepath>**

(viii) BC == PRECIP_grid, <grid_id>, filepath

(ix) BC == AIR_TEMP_grid, <grid_id>, filepath

(x) BC == REL_HUM_grid, <grid_id>, filepath

(xi) BC == SW_RAD_grid, <grid_id>, filepath

(xii) BC == LW_RAD_grid, <grid_id>, filepath

(l) **BC == QG, <filepath>**. A globally applied source term boundary condition. Flow per unit area and associated scalar concentration timeseries are specified in a comma-separated-value file.

(m) **BC == CP, <filepath>**. Specified scalar concentration profile boundary condition (refer user manual for details).

(n) The following fixed point source boundary conditions are available:

(i) BC == QC, X-coord, Y-coord, <filepath>. Specify inflow rate and associated scalar concentration timeseries in a comma-separated-value file. Scalars include salinity, temperature, suspended sediments, water quality consituents and passive tracers.

(ii) BC == FC, X-coord, Y-coord, <filepath>. Specify scalar concentration mass flux timeseries in a comma-separated-value file. Scalars include salinity, temperature, suspended sediments, water quality consituents and passive tracers.

(iii) BC == FB, X-coord, Y-coord, <filepath>. Specify sediment mass flux timeseries to be input directly into the bed. This capability requires the ST module to be enabled.

(iv) BC == FORCE, X-coord, Y-coord, <filepath>. Specify force timeseries to be applied to the water column. This is equivalent to a momentum flux source term.

(o) The following moving point source boundary conditions are available:

(i) BC == QCM, <filepath>. Specify point-source location, inflow rate and associated scalar concentration timeseries in a comma-separated-value file. Scalars include salinity, temperature, suspended sediments, water quality consituents and passive tracers.

(ii) BC == FCM, <filepath>. Specify point-source location, scalar concentration mass flux timeseries in a comma-separated-value file. Scalars include salinity, temperature, suspended sediments, water quality consituents and passive tracers.

(iii) BC == FBM, <filepath>. Specify point-source location and sediment mass flux timeseries to be input directly into the bed. This capability requires the ST module to be enabled.

(iv) BC == FORCEM, <filepath>. Specify point-source location and force timeseries to be applied to the water column. This is equivalent to a momentum flux source term.

(p) **BC == CYC_HOLLAND, <filepath>**. Parametric Tropical Cyclone boundary condition (refer manual for details).

(q) **BC == WAVE, <GRID_ID>, <filepath>**. Gridded wave boundary condition (refer manual for details).

(r) **BC == TRANSPORT, <filepath>**. Transport file boundary condition, specifies that the model will use a “transport file” output from an earlier hydrodynamic simulation to undertake scalar transport modelling (passive tracers, sediment transport or water quality constituents). This approach can be

more computationally efficient than undertaking scalar transport modelling as part of a hydrodynamic simulation.

6 Certain boundary conditions also allow for optional specification of a "**sub-type**" in order to control certain details of how these are numerically implemented, as described below. In all cases **Sub-type == 1** is the default applied.

(a) OBC (Open Boundary Conditions) boundaries support the following sub-type specifications:

(i) **Sub-type == 1**. Specified water level. Boundary normal momentum flux is modified to avoid BC over-specification which can lead to boundary reflection of outgoing energy.

(ii) **Sub-type == 3**. Specified velocities. Water level is modified to avoid BC over-specification which can lead to boundary reflection of outgoing energy.

(iii) **Sub-type == 4**. Over-specified. Water level and velocities are applied exactly as specified in the input files. This can lead to boundary reflection of outgoing energy.

(iv) **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.

(b) Q boundary conditions support the following sub-type specifications:

(i) **Sub-type == 1**. Boundary inflow is uniformly distributed along nodestring. 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.

(ii) **Sub-type == 2**. Boundary inflow is uniformly distributed along nodestring. Boundary condition is specified as a reflective wall with a source distributed along the internal boundary cells. This sub-type is suitable for use in 3D model simulations.

(iii) **Sub-type == 3**. 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.

(iv) **Sub-type == 4**. 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.

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

(i) **Sub-typ == 1.** When outflow is specified (Q<0) the scalar flux is determined by the interior model concentration (the BC file value will be ignored).

(ii) **Sub-typ == 2.** When outflow is specified (Q<0) the scalar flux is determined by the BC file specified value.

7 **BC Offset == <offset$_1$, offset$_2$…>**. This allows the value/s specified in the BC file to be offset by a constant value.

8 **BC Scale == <scale$_1$, scale$_2$…>**. This allows the value/s specified in the BC file to be multiplied by a factor.

9 **BC Default == <default$_1$, default$_2$…>**. This allows the user to specify default bc variable value/s that will be applied when a particular bc variable is not found in the specified bc file. If a bc default is not specified then any missing bc variables will be given a NaN ("Not-a_Number") value.

10 **BC Time Units == <seconds,minutes,hours,days>**. This allows the user to specify the units of time in a bc file where time is specified as a decimal unit from a reference time. This command is relevant to comma-separated-value timeseries inputs where the time variable is stored as a decimal number, but will be ignored where the time variable is specified as an "ISODATE". This command is also relevant to netcdf file timeseries inputs, where the time variable is stored as a "real" number.

11 **BC Reference Time == <ISODATE>**. This command allows the user to specify a bc specific reference time and should be used where the bc file (typically netcdf) has a different reference time to the model simulation. If not specified the BC reference time will be the same as for the model simulation (default is 01/01/1990 00:00). This command is only relevant for simulations using the ISODATE time specification.

12 **Vertical Distribution File == <filepath>**. This command allows the user to specify a vertical distribution for the stationary and moving point source term boundary conditions (the default is to apply the source term uniformly over the entire water column). This command is only relevant for 3D model simulations and should be used in conjunction with a Vertical Coordinate Type == <Type> command, where the type can be DEPTH (below surface), HEIGHT (above sea bed), ELEVATION or SIGMA (0. at surface and 1. at seabed). The vertical distribution file is a comma-separated-value format and should have one column with the heading equivalent to the specified vertical coordinate type (e.g. HEIGHT) and one column with the heading WEIGHT. The units of WEIGHT don't matter as the distribution gets normalised.

Hydraulic Structures

TUFLOW FV now 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.

Structure controls are defined using a structure block, requiring a unique block definition for every structure within the model. The syntax of a structure block is:

Structure == <type>
Structure behaviour specifications...
End Structure

The structure type defines the "geometry" of the structure representation. The following structure types are currently supported:

13 **Structure == nodestring**. Regulates flow across a nodestring.

14 **Structure == linked nodestrings.** Creates a flow linkage between two nodestrings and regulates the flow between them.

15 **Structure == zone**. Applies some control to a group of cells within a polygon region. For instance the bed elevation may be "controlled". The zone is defined using a **Polygon file == <filepath>** command. Refer to the user manual for polygon file format details.

16 **Structure == linked zones**. Creates a flow linkage between two zones and regulates the flow between them.

17 **Structure == autoweir**. Automatically inspects the model nodal elevation and applies weir flow relationships between cells where applicable (i.e. where the nodal elevations are higher than the adjacent cell elevations).

The following structure behaviour specifications are now supported:

18 Bed adjustment behaviour specifications, applicable to a zone structure type:

- **(a) Bed adjust == Zb_adjust.** Specify a controlled bed elevation. The control may be a timeseries or a trigger type (see below).
- **(b) Bed adjust == dZb_adjust.** Specify a controlled bed elevation change. Control types as per above.

19 Flux specifications, applicable to nodestring, linked nodestrings and linked zones structure types:

- **(a) Flux function== NLSWE**. Apply the non-linear shallow water equations to calculate the flux.
- **(b) Flux function == Timeseries**. Specify a flow timeseries in a csv file, using the command **Flux file == <filepath>**.
- **(c) Flux function == Matrix**. Specify a hQh relationship in a csv matrix file, using the **Flux file == <filepath>**. Command.
- **(d) Flux function == Weir**. Specify a fixed elevation weir flow relationship. Weir properties are specified using the following command, **Properties == <weir height, weir coefficient>**.
- **(e) Flux function == Weir_dz**. Specify a fixed height (above ground level) weir flow relationship. Weir properties are specified using the following command, **Properties == <weir height above ground level, weir coefficient>**.
- **(f) Flux function == Weir_adjust**. Specify a weir that adjusts according to control specifications (see timeseries or trigger control types below).
- **(g) Flux function == Weir_dz_adjust**. As above.
- **(h) Flux function == Wall**. Specify a solid wall (no flow). Currently only walls that cover the full water column height are supported.
- **(i) Flux function == Porous**. Specify flow according to Darcy's law through a porous media. Porous media properties are specified using the following command, **Properties == <hydraulic conductivity, porous media flow path length>**.
- **(j) Flux function == Culvert**. Specify culvert flow. Culvert properties are specified in a culvert file using the following command, **Culvert file == <filepath>**. Refer to user manual for culvert file details.

20 Blockage/width specifications, can be used to specify the effective flow width in conjunction with the flux functions described above. The effective flow width might vary with elevation as specified in either a blockage or width file.

(a) **Blockage file == <filepath>**. A blockage file specifies the fraction of the total flow width as a function of elevation.

(b) **Width file == <filepath>**. A width file specifies the structure flow width as a function of elevation.

21 Energy loss specifications, can be used to specify any additional unresolved energy (or head) losses (∆h) which occur as flow passes through the structure.

(a) **Energy loss function == coefficient**. The energy loss is calculated using ∆h =coeff.v^2/(2g) where v is the effective flow velocity through the structure.

(b) **Energy loss function == table**. The energy loss ∆h is specified in a csv file as a function of flow rate.

(c) **Energy loss file == <filepath>**. Specify the file defining the structure energy loss relationship.

22 Control specifications, provide the ability to further control the temporal behaviour of various structure types, either through an *a priori* timeseries or in response to conditions within the model (e.g. monitored water levels).

(a) **Control Parameter == <parameter>**, e.g. fraction open, weir crest, zb (bed elevation), dzb (bed elevation change), minimum flow. Specifies what particular structure property is being controlled (depends on the type of structure).

(b) **Control == Timeseries**. Structure operation is controlled with a timeseries defined in a csv file using the **Control file == <filepath>** command. An example would be where historic gate operation is known and can be specified directly (as a timeseries).

(c) **Control == Sample Rule**. Structure operation is governed by a "control rule" which defines the control parameter state as a function of a monitored value (e.g. water level). An example would be where gate operation is tied to water level monitoring.

(d) **Control == Target Rule**. Structure operation is governed by three components; a "target timeseries", a "control rule" and an internally monitored value. An example would be where gate operation is tied to water level monitoring and a target water level timeseries.

(e) **Control == Trigger**. Structure operation is triggered by a monitored parameter reaching a specified value. Once the trigger value is reached the structure is controlled with a timeseries defined in a csv file using the **Control file == <filepath>** command.

Further detailed information about specifying sampling (monitoring), target and trigger based controls is provided in the user manual.

More detailed descriptions and examples of structure block specifications are provided in the TUFLOW FV User Manual.

Output Types and Options

23 The following output types are now available within TUFLOW FV:

(a) **Output == datv**. Write requested output parameters as a 2D "sheet" to SMS dat files. A separate file is created for each individual parameter. The SMS dat file format only supports 2D output and therefore output vertical averaging is applied in the case of 3D simulations (see below).

(b) **Output == flux**. Write timeseries of nodestring fluxes (flow and scalar flux) to a csv file.

(c) **Output == points**. Write timeseries of requested output parameters to a csv file. Points are specified by coordinates in a csv file (Points file == <filepath>). Vertical averaging is applied in the case of 3D simulations (see below).

(d) **Output == mass**. Write timeseries of total volume and scalar mass integrated over the entire model domain.

(e) **Output == netcdf**. Write requested parameters as a 3D output to a TUFLOW FV specific format netcdf file (see user manual for format details). These files can be visualised using various matlab routines written specifically for TUFLOW FV (see below).

(f) **Output == transport**. Write a transport file for use in subsequent "transport mode" simulations. This allows for more computationally efficient simulation of passive scalar transport.

(g) **Output == structflux**. Write timeseries of structure fluxes (flow and scalar flux) to a csv file.

(h) **Output == profiles**. Write requested parameters as profile output to a netcdf file.

24 **Output Parameters == <param1, param2,…>**. A much more extensive list of output parameters is now available, see the user manual for details.

25 **Suffix == <string>**. Allows for a user-specified suffix to be appended to the output filename. This is required when multiple outputs of a particular type are requested in order to differentiate the filenames.

26 **Output compression == <1/[0]>**. Enable in-built compression of netcdf output files.

27 **Output statistics == <type1, type2,…>**. Output additional requested statistics. The following statistics are currently supported: **MAX** & **MIN**. This feature is currently available only with datv and netcdf output types.

28 The following vertical averaging options can be applied to datv, netcdf and points output.

(a) **Vertical averaging == depth-all**. 2D sheet output based on vertically averaging over the entire water column.

(b) **Vertical averaging == depth-range, <min, max>**. Vertically averaging over a depth-range (from depth below surface = min to depth = max).

(c) **Vertical averaging == height-range, <min, max>**. Vertically averaging over a height-range (from height above bed = min to height = max).

(d) **Vertical averaging == elevation-range, <min, max>**. Vertically averaging over an elevation-range (from elevation = min to elevation = max).

(e) **Vertical averaging == sigma-range, <min, max>**. Vertically averaging over a sigma-coordinate range (from sigma = min to sigma = max). The sigma-coordinate is defined as the height above bed divided by the total depth.

(f) **Vertical averaging == layer-range-top, <min, max>**. Vertically average over the layers (layer from top = min to layer from top =max).

(g) **Vertical averaging == layer-range-bot, <min, max>**. Vertically average over the layers (layer from bed = min to layer from bed =max).

If unspecified the vertical averaging will default to "depth-all" for datv and points output files and will default to "off" for netcdf files (i.e. output will be 3D). Note that vertical averaging is a property of the entire output file (and not of individual parameters within an output file).

Water Quality Module (AED^2)

TUFLOW FV is now coupled with an advanced aquatic ecosystem model (AED^2 - http://aed.see.uwa.edu.au/research/models/AED).

If you are interested in becoming a beta-tester for the TUFLOW FV water quality module please express your interest via sales@tuflow.com.

Miscellaneous

29 There is now limited support for TUFLOW FV simulations using US Customary / Imperial Units as well as the default Metric units. The alternative units systems are supported for 2D hydrodynamic simulations only.

(a) **Units == Metric**. Default.

(b) **Units == US Customary**; **Units == Imperial**; **Units == English**.

30 TUFLOW FV can now be run using an Aquaveo Trial license, without requiring an enabled hardware lock (dongle).

The following syntax is required when running TUFLOW FV through an Aquaveo Trial license

>> tuflowfv –SMS control_file.fvc

Wiki Page and Tutorial Models

A TUFLOW FV wiki page is now online at http://fvwiki.tuflow.com. The wiki page provides links to several tutorial models. These tutorial models can now be run independently of any licensing.

Results Visualisation Using Matlab

A toolbox for visualising 2D and 3D TUFLOW FV results in Matlab is now available for download from the tuflow website. http://www.tuflow.com/FV%20Utilities.aspx

A wiki page with instructions on using the matlab toolbox is currently under construction and should shortly be available through http://fvwiki.tuflow.com

Forum

A forum for TUFLOW FV users is now online at http://tuflow.com/fvforum/. New and existing users are encouraged to join and contribute to the forum as a community support resource.

Manuals

An updated user manual is available for download from :
http://www.tuflow.com/FV%20Documentation.aspx