Modflow User Tools (MUT) Version 1.22 User's Guide

August 2024

Rob McLaren, Young-jin Park, Sorab Panday

Contents

Chapter 1

Introduction

This document describes a new MODFLOW-USG[1] development environment which has the following features:

- We refer to it as Modflow User Tools, or MUT for short.
- MUT is designed to work with a modified version of MODFLOW-USG, where a new surface water flow package, called SWF, has been added. Like the Connected Linear Network (CLN) package, the SWF package represents a new domain type that is fully-coupled to the 3D groundwater flow (GWF) domain. There can also be cell-to-cell flows between the SWF and CLN domains. The SWF domain uses the diffusion-wave approach so simulate 2D surface-water flow. We will refer to this new version of MODFLOW-USG as MODFLOW-USG^{Swf} in this manual.
- We currently develop and run it on a MICROSOFT WINDOWS 10-based computing platform, writing software using the INTEL FORTRAN compiler running inside the MICROSOFT VISUAL STUDIO interactive development environment, which includes software version control tools through GITHUB.
- A text-based approach is used for the MUT interface, in which we first develop an input file of instructions that define our MODFLOW-USG^{Swf} project, then run MUT to read it and write a complete MODFLOW-USG^{Swf} data set. MUT also writes output files for TECPLOT, a third-party visualization software package, which provides a 3D graphical visualization tool to review the model numerical mesh and material properties in the data set. In future, MUT could be extended to support other third-party visualization packages, for example the open source program Paraview.
- MUT can post-process a MODFLOW-USG^{Swf} simulation to provide a TECPLOT visualization of temporal model results, including hydraulic heads, saturations, water depths and flow budget data. *If applied to output files which were produced by an earlier version of Modflow, results may be mixed. It is not our intent here to support all existing Modflow packages, many of which have been superceded.*

This document is subivided into the following sections:

Chapter 2 Installation and Setup: How to install MUT, MODFLOW-USG^{Swf} and TECPLOT and define MICROSOFT WINDOWS environment variables.

[1] https://www.gsienv.com/software/modflow-usg/modflow-usg/

Chapter 3 Model Build: How to build a Mut input file, run Mut to produce a Modflow-Usg^{Swf}-compatible data set and Tecplot-compatible output files, and run Tecplot to review the results of the model build.

Chapter 4 Model Execution and Post-Processing How to run Modflow-Usg^{Swf} run Mut to convert the output to Tecplot-compatible output files, and run and run Tecplot to visualize the results of the model run.

Chapter 5 Examples Examples of Modflow-Usg^{Swf} models built using Mut for both the verification and illustration of Mut and Modflow-Usg^{Swf} useage.

Chapter 6 Tutorial In the tutorial, we

Chapter 2

Software Installation and Useage

The first step in the software installation process is to obtain the Mut examples, executables and database files from https://github.com/Grdbldr/MUT_Examples.git as shown in Figure 2.1. You do this by:

- Clicking on the green 'Code' button.
- Choosing 'Download ZIP' from the drop-down menu.

Note that the ZIP file also includes the executable for Modflow-Usg^{Swf}.

Before you run Mut for the first time, you need to define a windows environment variable called `USERBIN`, as shown in Figure 2.2. You do this by:

- Typing the string 'en' in the windows taskbar search field.
- Opening the 'Edit the system environment variables' dialogue.
- Clicking on the 'Environment variables...' button at the bottom of the dialogue.
- Adding the variable `USERBIN`.
- Adding the path to `USERBIN` to the PATH variable.

Here we have set `USERBIN` to be equal to `c:\bin` but you are free to choose a different drive and folder. You will need to copy the files in `_MUT_USERBIN` folder (see panel 4 in Figure 2.1) to the `USERBIN` directory.

You should now be able to run Mut from the command prompt. To test this, start a new command prompt, then type `mut`, you should see the Mut header, shown in Figure 2.3. Type `CTRL C` to stop the program.

You can also run Modflow-Usg^{Swf} by typing `usgs_1`. You should see the Modflow-Usg^{Swf} header, shown in Figure 2.4. Type `CTRL C` to stop the program.

If this is not the case, check the definitions of the `USERBIN` and PATH variables. If they are correct, you may need to re-boot your computer and try again.

A licensed version of Tecplot can be obtained from https://tecplot.com/products/tecplot-360/. They have a free 30-day trial option for those who want to assess the software before purchase. They also offer educational discounts.

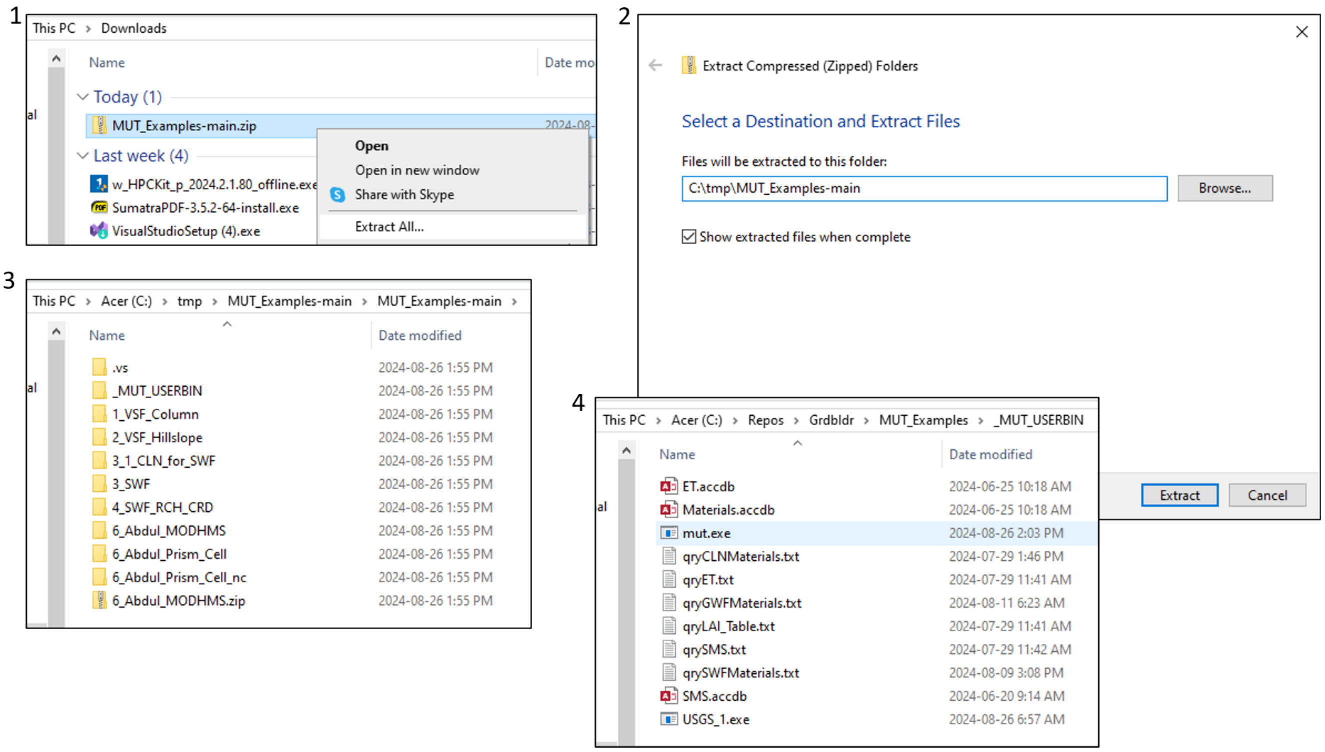


Figure 2.1: Downloading the MUT_Examples files

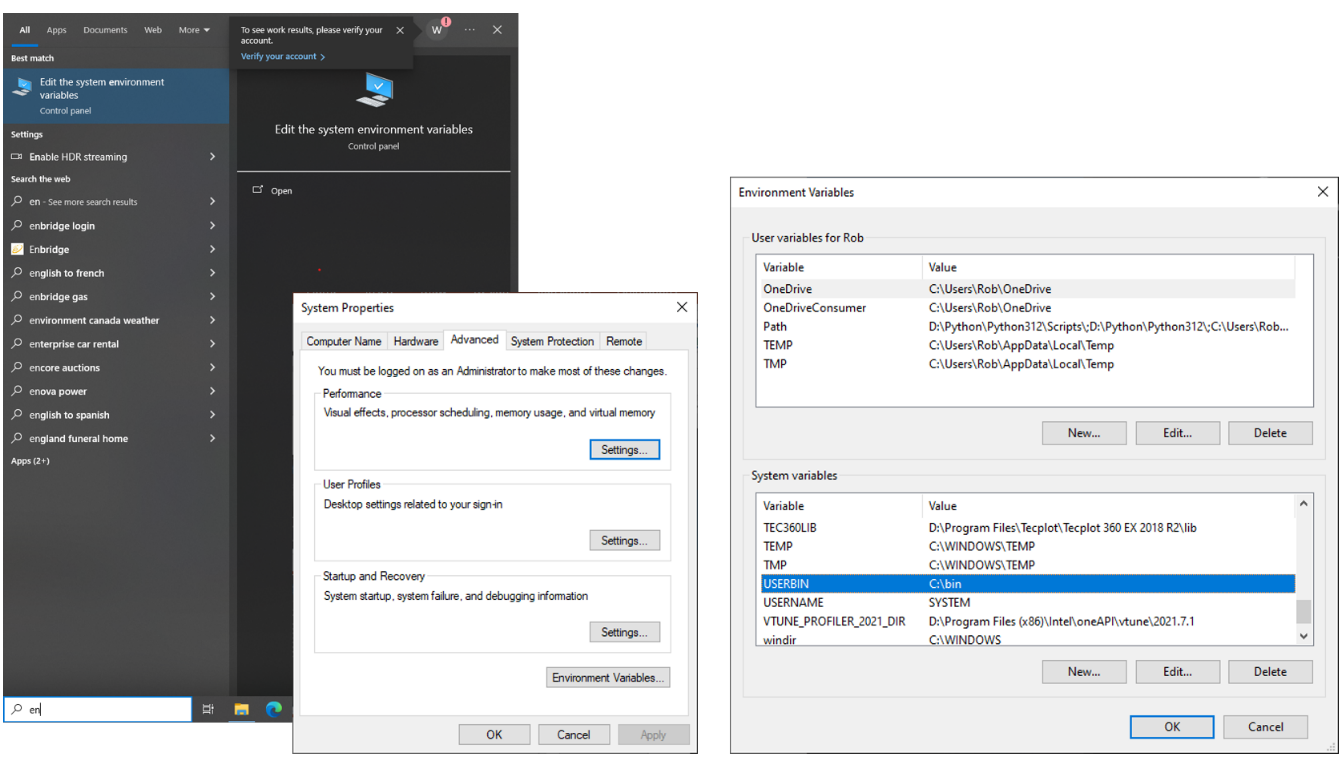


Figure 2.2: Defining a new environment variable

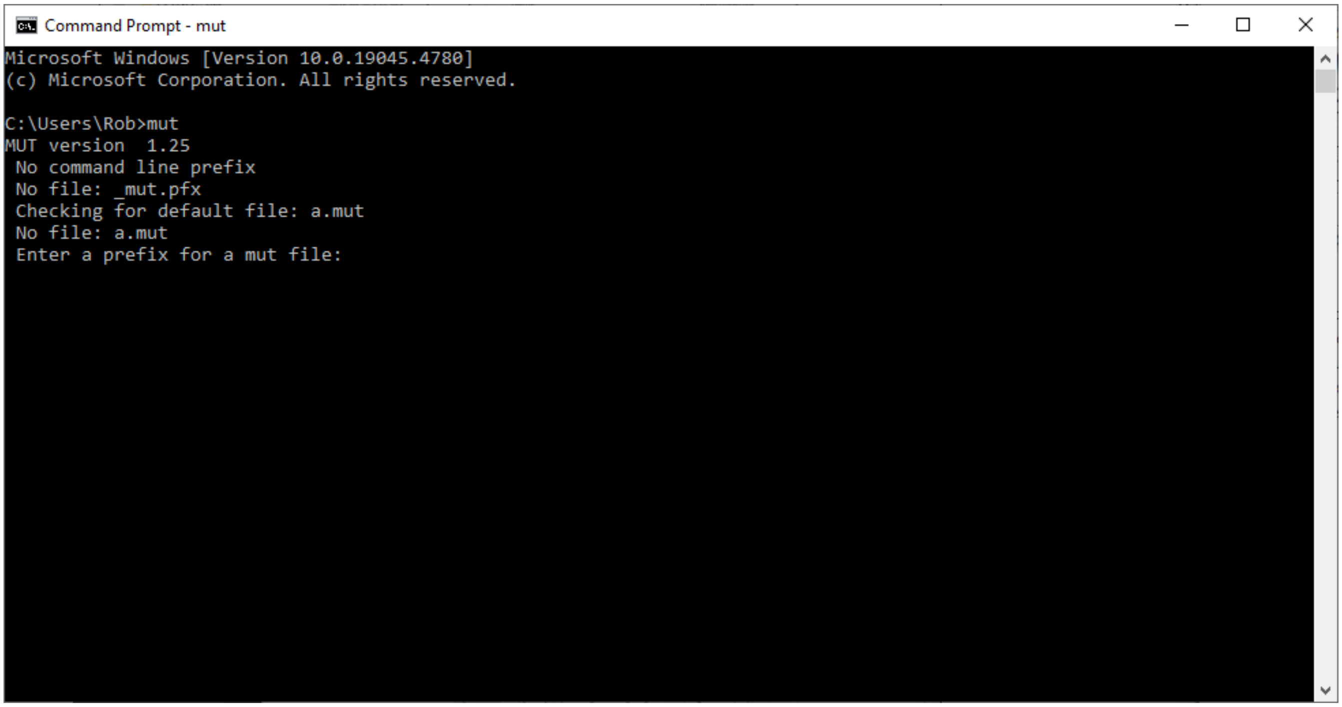


Figure 2.3: MUT Header

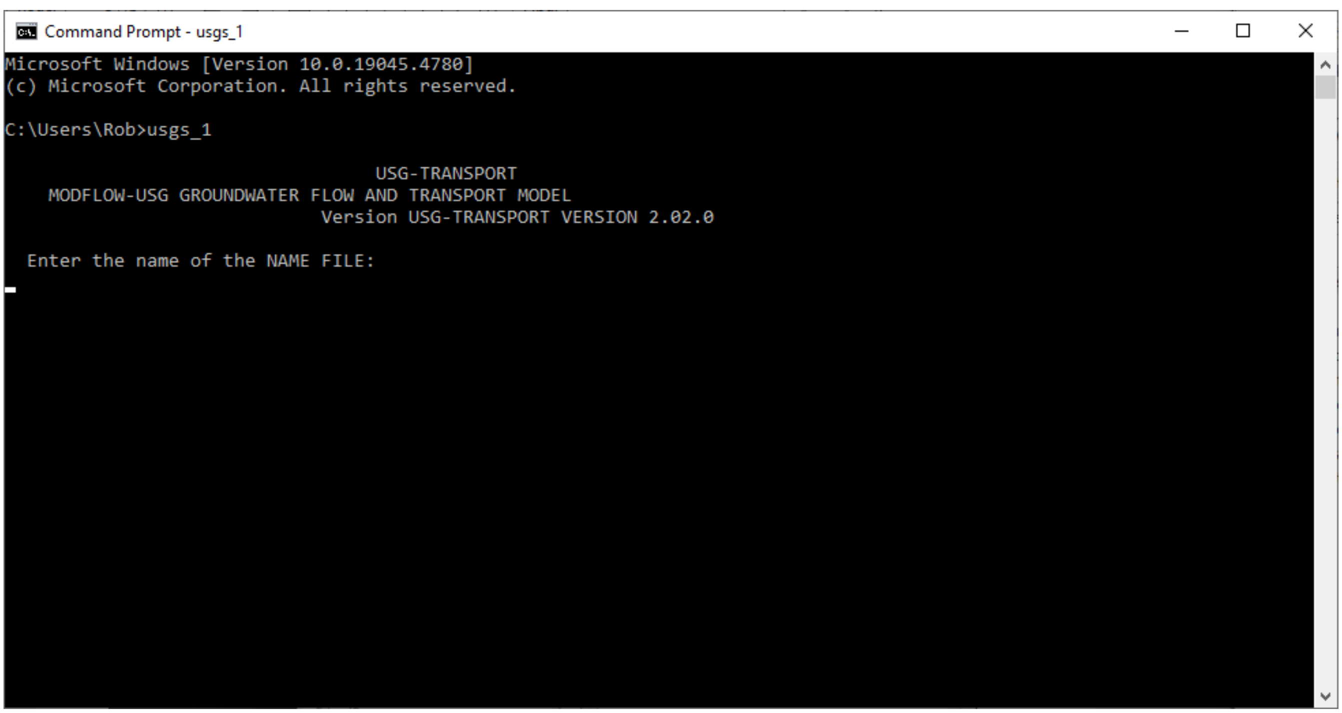


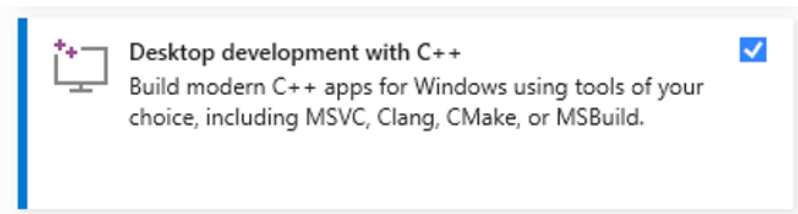
Figure 2.4: MODFLOW-USG^{Swf} header

Those of you who are just interested in running the Mut and Modflow-Usg^{Swf} programs have completed the required software installation tasks and can proceed to Chapter 3, **Model Build**.

Those who want to view and possibly modify and re-compile the source code for Mut and Modflow-Usg^{Swf} should proceed with the following instructions for setting up your Microsoft Windows programming environment.

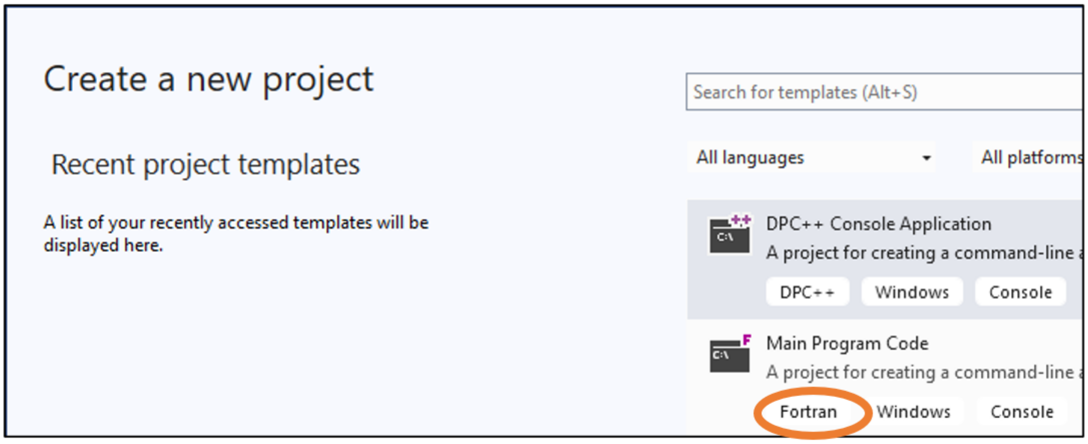
As was stated earlier, we use and recommend Microsoft Visual Studio and Intel Fortran. You should install Microsoft Visual Studio before Intel Fortran, which will then be automatically integrated into Microsoft Visual Studio.

A free version of the latest Microsoft Visual Studio (currently 2022) can be obtained from `https://visualstudio.microsoft.com/vs/community/`. Once you are on the site just click the `Download` button. This will download a file (e.g. `VisualStudioSetup.exe`) which can be run to install Microsoft Visual Studio. If you already have a version of Microsoft Visual Studio, you can choose to keep your old version and add the latest version. When you come to the installation options 'Workloads' page, be sure to check the option for `Desktop development with C++`, shown here:

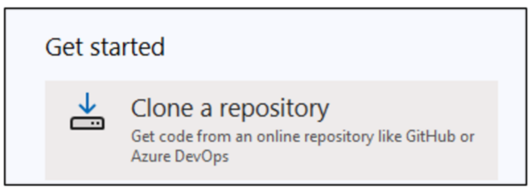


A free version of the latest Intel Fortran compiler can be obtained from `https://www.intel.com/content/www/us/en/developer/tools/oneapi/hpc-toolkit.html`. Once you are on the site just click the `Get It Now` button to download the Intel® HPC Toolkit, which includes Intel Fortran. Choose the `Windows` option then the `Offline Installer` option. Now you can either fill in the required information and start the download or choose to `Continue as guest(download starts immediately)`. This will download a file (e.g. `w_HPCKit_p_2024.2.1.80_offline.exe`) which can be run to install Intel Fortran.

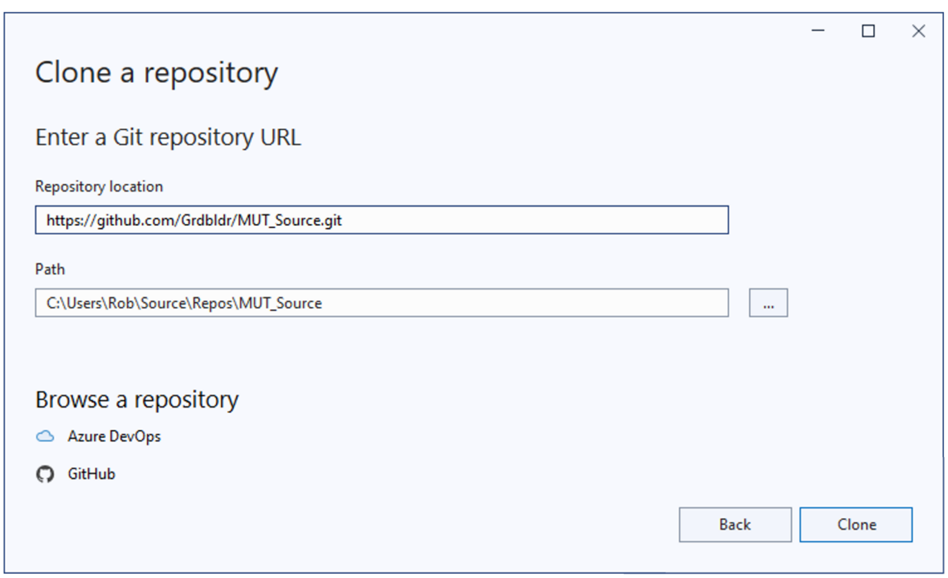
You can check the installion of Microsoft Visual Studio and Intel Fortran by starting Microsoft Visual Studio and choosing `Create a new project`. The window that appears should have links for creating Fortran projects, as shown here:



The MUT source files can be obtained from a GITHUB repository at https://github.com/Grdbldr/MUT_Source.git. Since GITHUB has been integrated into MICROSOFT VISUAL STUDIO we will use it to download the MUT repository. When you start MICROSOFT VISUAL STUDIO choose the following option:



This opens the dialogue box shown below, where you can define the repository location on GITHUB and the path to the local repository. You can copy the link from the PDF file by right-clicking on it and choosing `Copy Link Adress`.



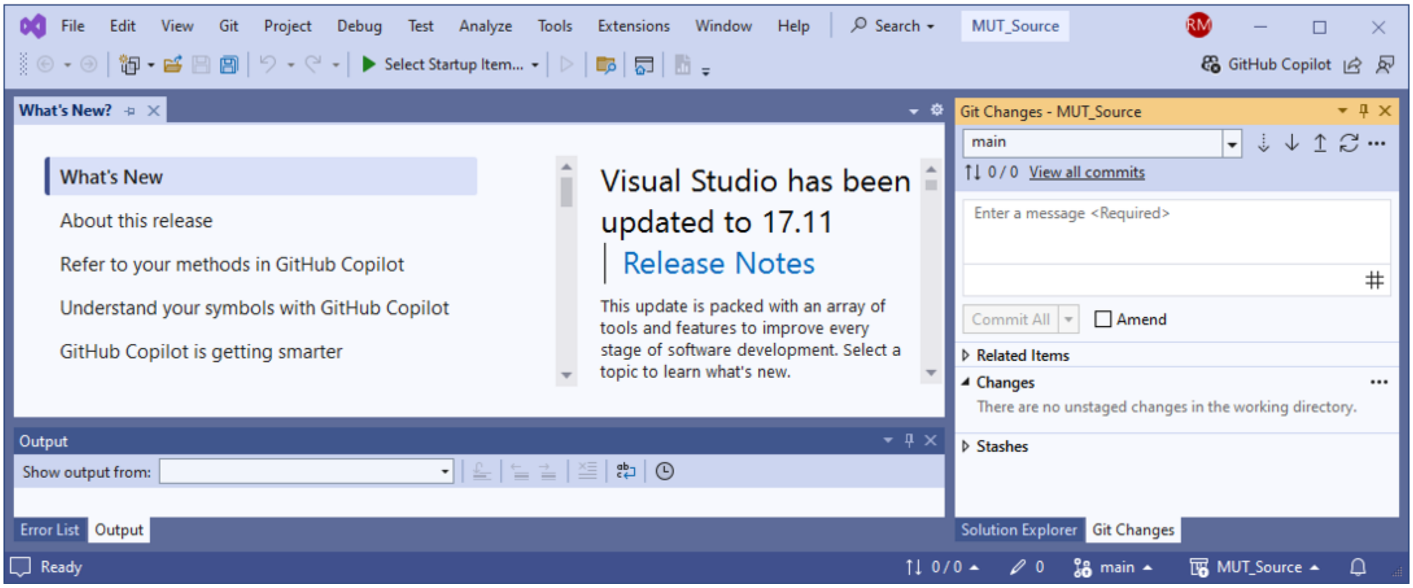


Figure 2.5: Microsoft Visual Studio after cloning `Grdbldr/MUT_Source`

Now choose the `GitHub` option under `Browse a Repository` and you will see this dialogue shown below, Choose `Grdbldr/MUT_Source` from the list of repositories then click the `Clone` button.

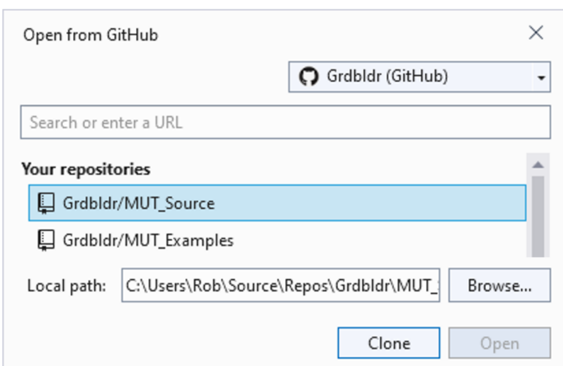


Figure 2.5 shows Microsoft Visual Studio after cloning `Grdbldr/MUT_Source`. Note the GitHub dialogue on the right side of the changes tab, with information along the bottom about the repository. Details about using GitHub in Microsoft Visual Studio are given in Tutorial 6.2.

The software has been developed and tested under:

- Windows 10
- Tecplot360 EX 2018 R2
- Microsoft Visual Studio Community 2022, Version 17.11.1
- Intel® Fortran Compiler 2024.1

Chapter 3

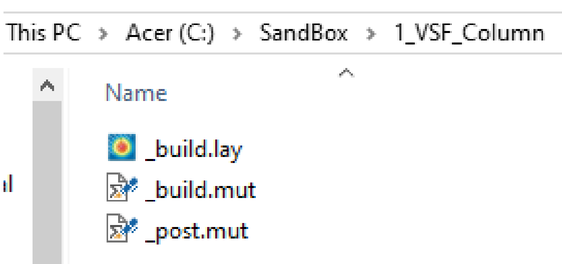
Model Build Workflow

This model build workflow is designed to create and fine-tune a MODFLOW-USG^{Swf} model, which represents a conceptual model we have of some hydrogeologic flow system, real or imaginary. MUT, provides the link between conceptual model and MODFLOW-USG^{Swf} model, and is intended to minimize the amount of time we spend building and testing it. [1]

The first step in the workflow is to define the conceptual model. To illustrate a typical model build, we'll adopt a simple conceptual example. [2] of groundwater flow though a 1D column of soil that is 100 m thick and which receives a constant rainfall of 0.4 m/yr, and where the groundwater table is fixed at the base of the column.

Next we need to develop a MUT input file, which is a plain ascii text file that you can edit with your preferred editor (e.g. Windows Notepad). [3] Each modflow input file name must have the extension `mut`, and a prefix of your choice. Examples of valid MUT input file names are `_build.mut` or `good.mut`. Most often, the easiest approach is to copy an existing input file and modify it as required. This helps reduce set-up time and avoid potential errors that are introduced when creating input files from scratch.

We recommend you copy the contents of the folder `MUT_Examples\1_VSF_Column` to a new location (e.g. `C:\SandBox`) and perform the actions yourself as we move through the workflow. If you do so, your working directory will look something like this:



In this example, there are two MUT input files, one for the model build called `_build.mut`, and one for post-processing called `_post.mut` (discussed later in chapter 4).

[1]As you read along, we urge you to you carry-out the workflow steps we describe here. Hands-on experience is probably the best way to learn how to use new software. You can modify our workflow to suit yourself, with the goal always being to save more of your valuable time and energy!

[2]This example is described in Section 5.1.1.

[3]Our personal favourite editor is WinEdt (https://www.winedt.com/snap.html).

MUT tries to obtain a prefix in one of these three ways:

1. **As a command line argument:** At the command prompt, MUT checks for the presence of a command line argument. For example, typing this:

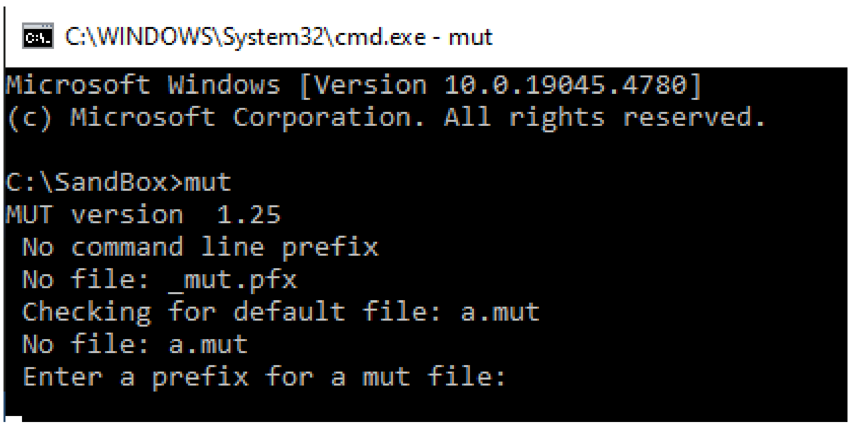
   ```
   mut  MyInput
   ```

   would cause MUT to process the input file `MyInput.mut`.

2. **Use a prefix file:** If there is no command line argument, MUT checks for the presence of the file `mut.pfx` in the folder. If present, MUT will read the prefix from it. For example, if the mut file was called `_build.mut` then the file `mut.pfx` would have the single line `_build`.

3. **Use the default input file name:** Finally, if there is no command line argument or `mut.pfx` file in the folder, MUT checks for the presence of the file `a.mut`. If present in the folder, MUT will use it.

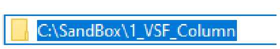
If none of these methods are successful, MUT will then prompt you for a prefix as shown here:



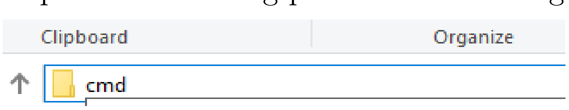
In our example, we will use option 1. First, start a command prompt in the folder `C:\1_VSF_Column\SandBox` as follows:

1. Navigate to the folder in File Explorer.

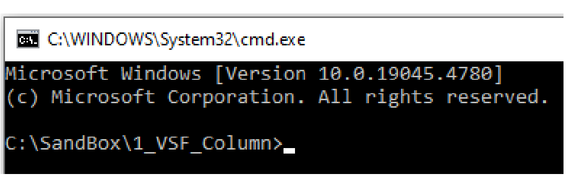
2. Click on the path in File Explorer:

3. Replace the existing path with the string 'cmd':

4. Press Enter/Return and you will see the following command prompt:

Now start MUT by typing:

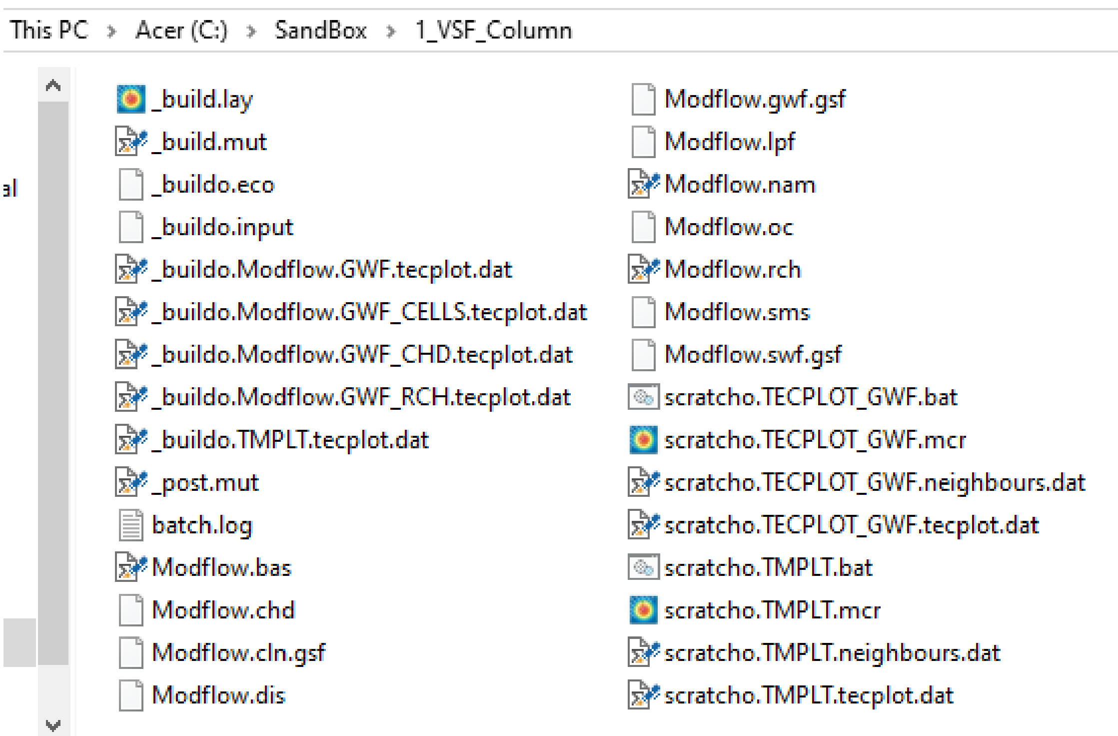
```
mut  _build
```

Note the following as MUT processes the input file:

- Output is written both to the screen and to the file `_buildo.eco` as execution progresses. The first thing MUT writes is the version number, which can be checked if something unexpected happens. If the run is successful the last line written will be `Normal exit`, otherwise an error message will be given.

- Comment lines are stripped from the input file and echoed to the screen and file.

- Several new output files are created:



- MUT output files have the prefix `_buildo`. Because we started the prefix with the underscore character, output files appear near the start of the list if sorted by name.

- There are several TECPLOT output files that are indicated by the suffix `.tecplot.dat`.

- Modflow files are written using the default prefix `Modflow`, (e.g. `Modflow.nam, Modflow.bas` etc.) The prefix can be customized if desired but there are advantages to keeping this 'generic' one, such as portability of post-processing scripts or TECPLOT layout files that follow this generic naming convention.

- Several scratch files (with prefix `scratcho`) are written which may be useful for debugging during, for example, code development. These can be ignored in most cases.

- Mut deletes previously generated output files and writes a fresh set each time it is run. This prevents confusion that can arise when out-of-date output files are present. [4]

We will now go through the contents of the example input file _build.mut. Open the file MUT_Examples\1_VSF_ in your preferred text editor.

You will see the first couple of lines are comments describing the problem:

```
! Examples\1_VSF_Column:
!   A modflow project of a 1D column generated from a simple 2d rectangular mesh
```

Comments begin with an exclamation point character '!' and are ignored by Mut. Mut initially strips the input file of all comments and creates a clean copy called *prefix*o.input, which is then processed by Mut.

In addition to comments, the input file contains Mut instructions. Some instructions require additional data, which can be numbers or alphanumeric strings. The next line in the input file is the instruction:

```
build modflow usg
```

This instruction starts what we refer to as a subtask. Each subtask has it's own set of instructions, which are read until an **end** card is encountered, which teminates the processing of the subtask. We suggest appending the subtask name to the **end** instruction, which makes debugging easier when subtasks are nested. So for example, we would end this subtask with the following instruction:

```
end build modflow usg
```

The steps required to build a Modflow-Usg^{Swf} GWF model with Mut are:

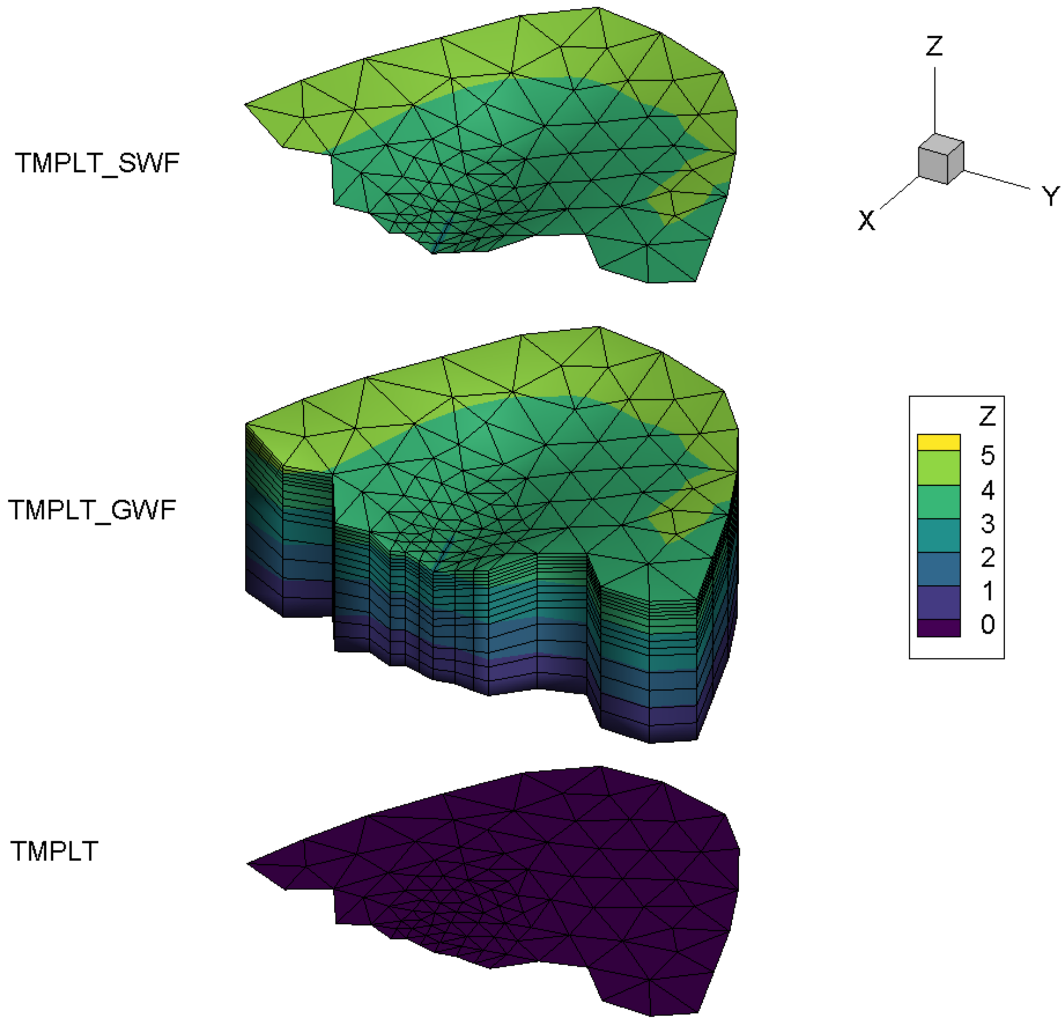
- Define a template mesh.
- Generate a 3D GWF domain from the template mesh.
- Assign material properties to the GWF domain.
- Assign the boundary and starting conditions.
- Assign the stress period and time-stepping parameters.
- Assign the solver parameters.
- Assign the output controls.

The following sections describe the rest of the input file contents.

[4]For example, if we define a recharge boundary condition, Mut will create the file *prefix*o.Modflow.SWF_RCH.Tecplot.dat which shows the locations and recharge values assigned to Modflow cells. If we then removed the recharge condition from the input file, but did not delete this output file, it may lead us to think that recharge condition is still being applied.

3.1 Template Mesh

A template mesh is a 2D finite-element mesh that is used, ultimately, for generating MODFLOW-USG^{Swf} finite-difference meshes composed of cells. As an intermediate step, a finite-element mesh is created, which is then used to define the MODFLOW finite-difference cells. Below is an example [5] showing an exploded view of a template mesh (bottom image) that was used as a basis for generating intermediate finite-element meshs for the GWF (middle image) and SWF (upper image) domains:



Note the following:

- The template mesh is assigned an elevation of zero, and only the xy coordinate data are used to define the other domains.
- The GWF domain has been assigned a base elevation of zero, and a variable top elevation.
- The SWF domain has been assigned the same elevation as the GWF domain i.e. they are coincident.

There are options for defining the template meshes:

[5] See TECPLOT layout file `MUT_Examples\6_Abdul_Prism_Cell\FIG Template Abdul.lay`.

generate uniform rectangles The file

2d mesh from gb The value is assigned to all nodes in the top sheet of the TECPLOT_GWF mesh.

2d quadtree mesh from groundwater vistas The file

2d mesh from gb' ' generate uniform rectangles'

In our example, the template mesh is defined by the following lines in the input file:

```
generate uniform rectangles
1.0, 1    !  Mesh length in X-direction and number of rectangular elements
1.0, 1    !  Mesh length in Y-direction and number of rectangular elements
```

The `generate uniform rectangles` instruction reads two lines of input data:

1. **xl, nbx** Domain length and number of rectangles in the x-direction
2. **yl, nby** Domain length and number of rectangles in the y-direction

This data is used to generate a grid for a rectangular domain made up of uniform rectangles, which is formed by subdividing the domain in the x-direction into **nbx** blocks, each of length **xl/nbx**. The domain is subdivided in a similar fashion in the y-direction, using the second line of input data.

In our example, this results in a single rectangular element that is 1 by 1 m in size, as shown in Figure **??**(a).

3.2 Groundwater Flow(GWF) Domain

A MODFLOW-USG^{Swf} 3D groundwater flow (GWF) domain can be generated from the template using the instruction generate layered gwf domain, with the option of defining the MODFLOW cell control volumes using template mesh element (mesh-centred control volume) or node (node-centred control volume) locations. , as shown in Figure 3.1 for a triangular-element template mesh.

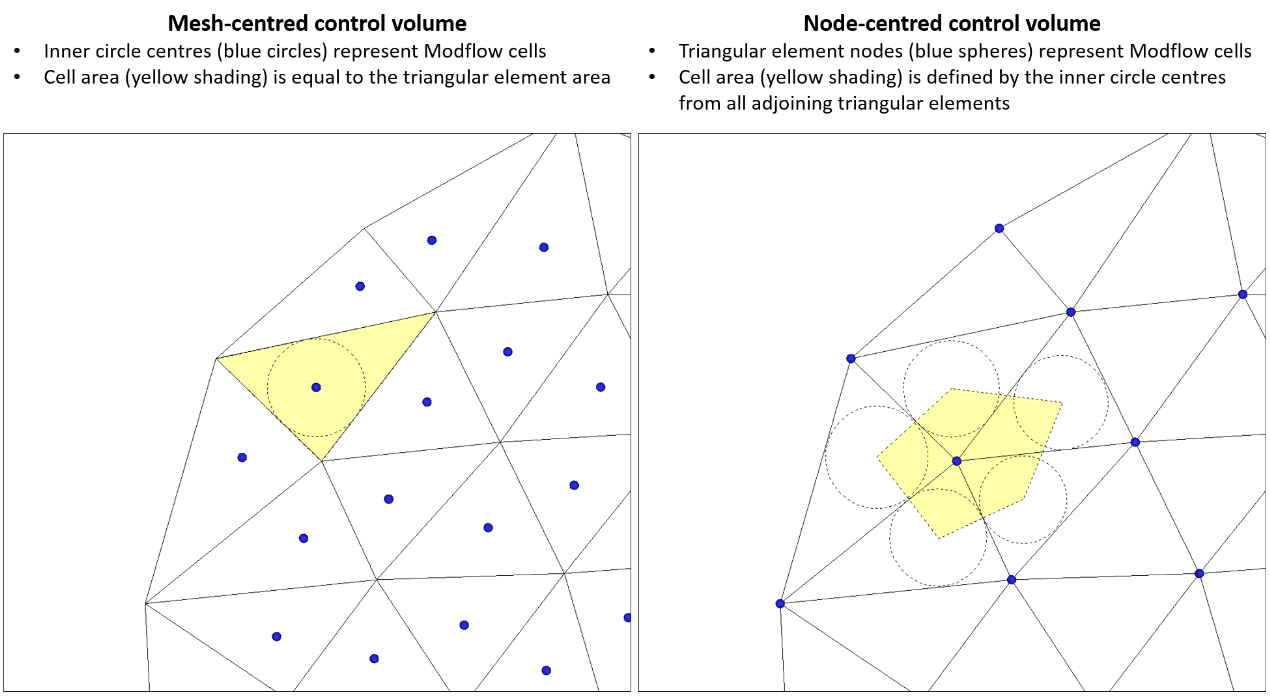


Figure 3.1: Modflow cell control volume types defined from a triangular-element mesh. On the left is a mesh-centred control volume, on the right a node-centred control volume.

Note the following: By default, the mesh-centred control volume approach is used. If you would like to use the node-centred control volume approach instead, the first entry in the `build modflow usg` subtask should be:

```
nodal control volumes
```

, as shown in this example:

```
generate layered gwf domain

    top elevation
        elevation constant
        100.0
    end top elevation

    new layer
        layer name
        Top layer

        uniform sublayering
        100

        elevation constant
        0.0
    end new layer

end generate layered gwf domain
```

The generate layered gwf domain instruction is a subtask so it has a corresponding end generate layered gwf domain instruction. The GWF domain is generated from the top layer down, and the first instruction, top elevation, defines the ground surface elevation at the layer of elements in the TECPLOT_GWF mesh. It must begin with a top elevation instruction and at least one new layer instruction. Not shown in this example is an optional additional instruction, zone by template.

There are options for defining the top elevation:

elevation constant The value is assigned to all nodes in the top sheet of the TECPLOT_GWF mesh.

elevation from gb file The file

elevation from ascii file The file

elevation from xz pairs The list

3.2.1 Constant Head(CHD)

```
Pre-requisites:
Activate one of GWF, SWF or CLN domains
Choose cells

Instructions:
gwf constant head

Inputs:
Head    L
```

All chosen cells will be assigned a constant head equal to the specified total head value.

3.2.2 Drains(DRN)

```
Pre-requisites:
Activate one of GWF, SWF or CLN domains
Choose cells

Instructions:
gwf drain

Inputs:
Drain conductance  L/T
```

All chosen cells will be assigned a drain elevation equal to the top elevation of the cell with the specified drain conductance.

3.2.3 Recharge(RCH)

```
Pre-requisites:
Activate one of GWF, SWF or CLN domains
Choose cells

Instructions:
    gwf recharge
    0.4     ! recharge
    3       ! recharge option

Inputs:
Recharge rate   L/T
Recharge Option  -
```

All chosen cells will be assigned a recharge.

3.2.4 Pumping

3.3 Connected Linear Networks(CLN) Domain

3.4 Surface Water Flow(SWF) Domain

In Mut a 2D template mesh is first defined and then a surface water flow (SWF) domain is generated from it using the instruction generate swf domain, as shown in this example:

```
generate swf domain
    top elevation
        elevation from gb file
        ./gb/grid.nprop.Surface elevation
    end
end
```

3.4.1 Critical Depth

3.4.2 Zero-Gradient Depth

Chapter 4

Model Execution and Post-Processing

Chapter 5

Examples

5.1 Verification

5.1.1 Unsaturated flow in a 1D Column

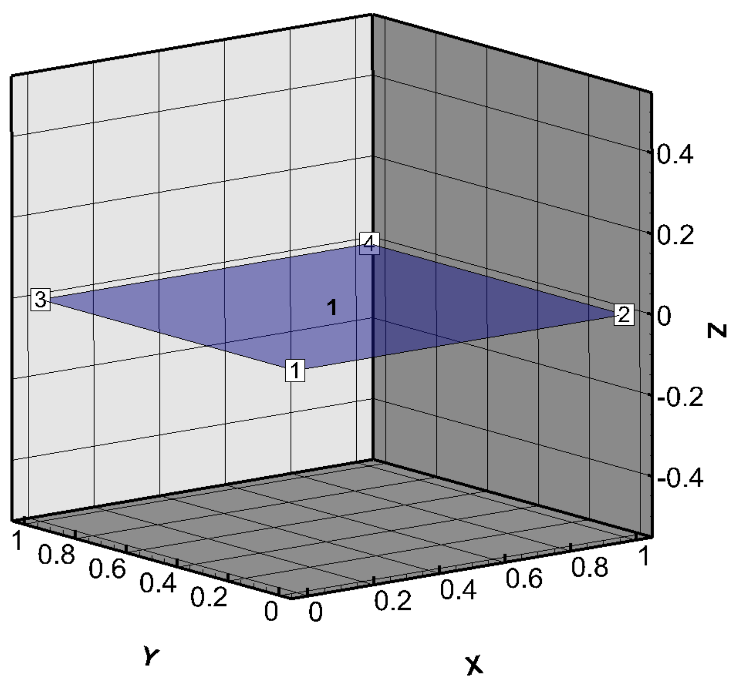
This example can be found in the folder `MUT_Examples\1_VSF_Column`

```
! Examples\1_VSF_Column:
!   A modflow project of a 1D column generated from a simple 2d rectangular mesh
build modflow usg
```

The template mesh is generated using the following instructions:

```
    generate uniform rectangles
    1.0, 1   !  Mesh length in X-direction and number of rectangular elements
    1.0, 1   !  Mesh length in Y-direction and number of rectangular elements
```

The 1D column example uses a single 1m X 1m rectangular element as the template mesh for building



the GWF domain.

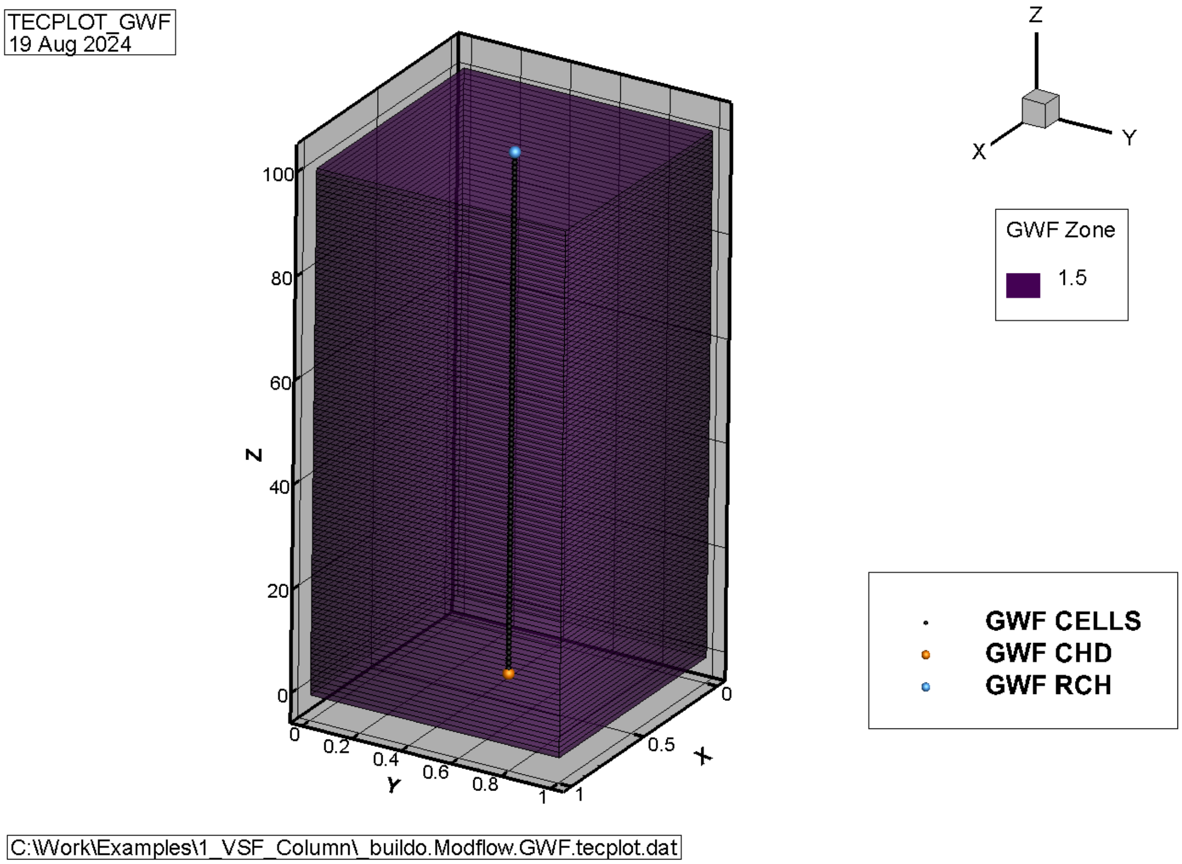


Figure 5.1: 1D column build

Figure 5.1 shows the The 1D column example uses a single 1 m by 1 m rectangular element as the template mesh for building the GWF domain.

5.1.2 Unsaturated flow in a 2D Hillslope: Drains vs Surfacewater Flow

5.1.3 Abdul's Experiment: 3D Unsaturated GroundwaterFlow

5.2 Illustration

Chapter 6

Tutorial

6.1 Mut Useage

In this tutorial, we will build a 3D fully-coupled GWF-SWF model, check the build using TECPLOT, run MODFLOW-USG^{Swf} to generate output, then examine the results using TECPLOT.

```
! This example builds a modflow project of the Abdul Field Experiment
! The SWF mesh and top of the GWF mesh are defined by a 2D Grid Builder triangular mesh
build modflow usg
```

Any input line beginning with is considered to be a comment and will be ignored by MUTḢere we begin the file with two comments describing the project.

The third line activates the MUT 'build modflow usg' environment, which accepts further instructions required to define the project. This environment can be split into roughly 4 sections:

Grid definition Instructions for defining the GWF, SWF and CLN numerical discretizations.

Modflow parameters Instructions for supplying Modflow parameter values (e.g. solver inputs for the SMS package, hydraulic properties for the LPF package etc.)

Stress periods, boundary conditions These instructions are repeated once for each desired stress period and include instructions about time stepping parameters and boundary conditions that are to be applied.

Output control Instructions defining a list of output times at which Modflow output files (e.g. heads, drawdowns, cell-by-cell flows etc.) are to be written.

The first group of instructions are used to build the Modflow unstructured mesh. MUT requires a 2D 'template'

```
! -----------------------------------Grid definition
2d mesh from gb
./gb/grid
```

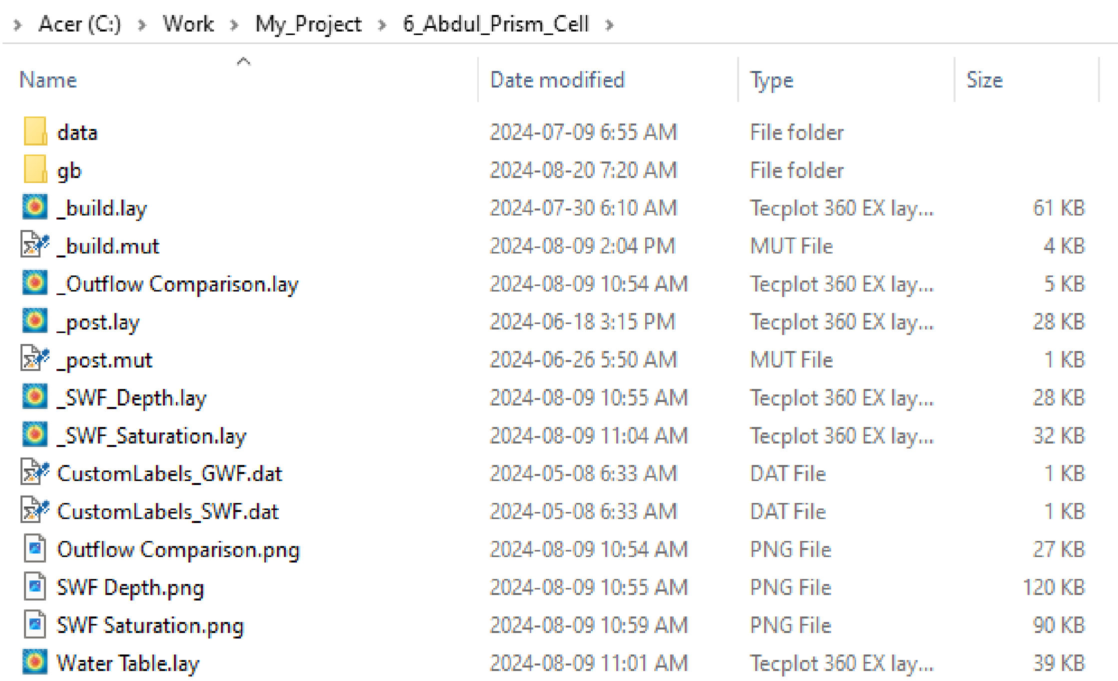


Figure 6.1: The contents of the 6_Abdul_Prism_Cell folder

Step 1: Copy an Existing Mut Project

Create a new folder called e.g. `My_Project` and copy the folder `MUT_Examples\6_Abdul_Prism_Cell` into it. Figure 6.1 shows the contents of our `6_Abdul_Prism_Cell` folder. Yours may look different depending on the root drive and folder location.

This example contains several files you might typically find in a Mut Modflow project, including Mut input files (extension .mut), Tecplot layout files (extension .lay), Tecplot input files (extension .dat). For now, our focus will be on the Mut input file `_build.mut`.

Step 2: Modify the Input File(s)

The input file `_build.mut` is set up to build a Modflow project. If you open the file in a text editor you will see that it consists of a sequence of comments (lines beginning with an exclamation mark `!`), Mut instructions and data (numbers or alphanumeric strings). Details of the input file contents are described in detail in Section **??**. For now, we will only make a minor change to the input file before moving on to the next step, which is to add a new comment line of your choice at the start of the file.

Step 3: Execute Mut to Build the Project

Assuming you have followed the set-up instructions in Section **??**, you can execute Mut with the input file `_build.mut` by typing:

```
mut _build
```

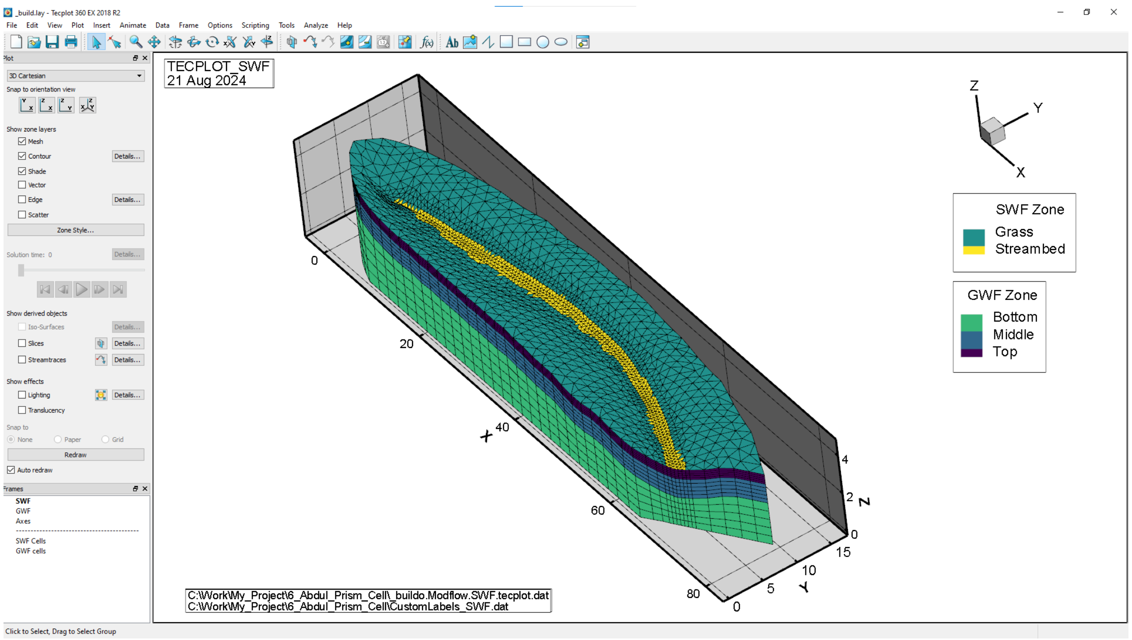
Step 4: Run Tecplot to Examine the Built Project

You can run Tecplot and load the Tecplot layout file `_build.lay` by typing:

```
tec360 _build.lay
```

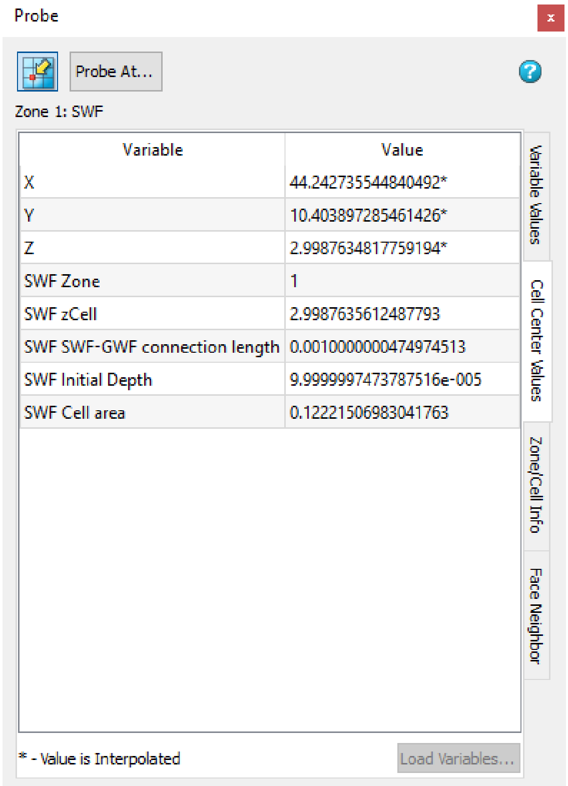
Note the following:

- A Tecplot window should open:



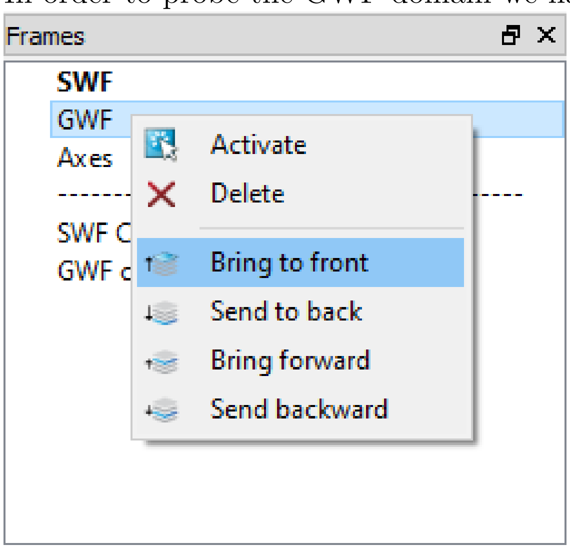
  This Tecplot layout file has been constructed with multiple frames (see lower left 'Frames' window) showing details about the SWF and GWF model domains. This default view shows the distribution of the various materials defined in the model, such as the SWF domain materials called 'Grass' and 'Streambed'. Detailed information about manipulating the data in Tecplot to produce the desired plots is discussed in Section **??**.

- Tecplot data can be probed using the probe tool  . Here we see the results of probing a location in the SWF domain:

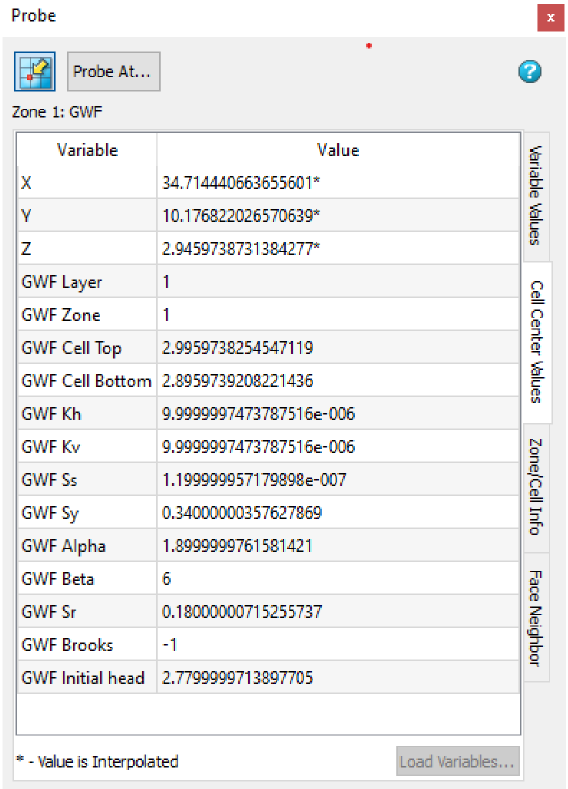


SWF results were returned because the SWF frame is at the front of the frame stack.

- In order to probe the GWF domain we have to move it to the front of the stack:



Here we see the results of probing a location in the GWF domain:



Step 5: Run Modflow to Generate Output

Step 6: Run Mut to Post-Process the Modflow Output

Step 7: Run Tecplot to Visualize the Modflow Output

6.1.1 Suggested Workflow

A well-designed workflow should minimize the introduction of human error into the modelling process and facilitate later review by senior modellers. Below we describe one possible approach that can be used as a starting point for implementing your own personal workflow. We will use the verification example `6_Abdul_Prism_Cell` to demonstrate our suggested workflow. The steps in the workflow are:

1. Copy an existing Mut project folder to a new working folder.
2. Modify the `_build.mut` file (and other input files if necessary) to reflect the new Modflow project.
3. Run Mut to build the new Modflow project, which produces Tecplot output files for the various Modflow domains (i.e. GWF, SWF and/or CLN) created during the build process.
4. Run Tecplot and examine the build output files. Repeat steps 2-3 until the new project is defined correctly.
5. Run Modflow to create the new project output files (e.g. time-varying hydraulic head, drawdown etc).

6. Run `_post.mut` to post-process the Modflow project, which produces Tecplot output files for the various Modflow domains (i.e. GWF, SWF and/or CLN) created during the Modflow simulation.

7. Run Tecplot and examine the Modflow output files.

6.1.2 Mut Input File Structure

Mut recognizes files which have the extension `.mut` as input files and reads and interprets them to produce both Modflow-Usg^{Swf} output files and Tecplot input files.

Comments begin with an exclamation point character `!` and are ignored by Mut. Mut initially strips the input file of all comments and creates a clean copy called *prefix*`o.input`, which is then processed by Mut. This means comments can be placed anywhere in the input file.

6.2 GitHub Useage in Microsoft Visual Studio

6.3 Tecplot Useage

Index