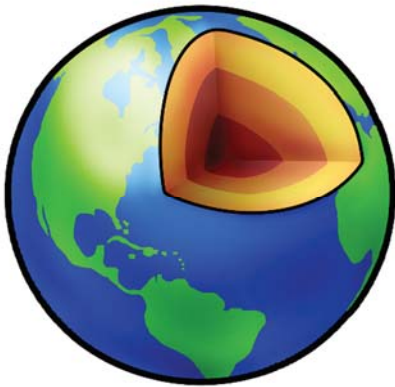


Earth Volumetric Studio Help Version 2021.4.2



Earth
Volumetric
Studio™

Table of Contents

Table of Contents	2
Earth Volumetric Studio Help System.....	10
Presentation Applications	11
How to Use EVS Presentations.....	11
Prerequisites to Using EVSPs	11
Opening an EVSP in Earth Volumetric Studio.....	11
EVS Installation and Licensing	23
Basic Training	23
Video Tutorials at www.ctech.com	23
The EVS Home Tab.....	24
Python Scripting	25
Python Functions & Operators	27
Animation.....	29
The Earth Volumetric Studio Environment	34
Visual Programming	35
The Earth Volumetric Studio Environment	35
Visual Programming	36
Load an Application.....	37
Transformations with the Mouse	38
Transformations with the Azimuth and Inclination Controls	39
Basic Statistics	42
Change Plume Level.....	44
Migration from Studio-32 to Studio-64.....	45
Exit EVS.....	56
EVS Project Structure for Maximum Portability	56
Instance Modules.....	57
Connect the modules	58
Home Tab Basic Options	60
Run the Application.....	63
Exporting from Excel to C Tech File Formats.....	67
Creating AIDV Files - Analyte Data Measured over Intervals.....	67
Creating APDV Files - Analyte Data Measured at Points	71
Creating GEO Files - Stratigraphic Horizons from Vertical Borings	74
Creating PGF Files - Lithology	77
Data Requirements Overview	79
Creating a Simple Application	81
Viewing PGF Files	83
Viewing GEO Files.....	88
Viewing GMF Files.....	91
Viewing APDV Files	97
Viewing AIDV Files.....	108

Packaging Data Into Your Applications	109
How to Package a Single File	109
Packaging All Files in an Application	110
How to Read a Packaged File in a New Module	111
Modules Requiring Special Packaging Treatment	113
read_vector_gis.....	113
read_cad	115
overlay_aerial	116
Geostatistics Overview	116
Standard Deviation	117
Confidence	118
Uncertainty.....	121
Min-Max Estimate	122
Visualization Fundamentals.....	125
■ Data Content Requirements	125
■ Direct Data Visualization.....	125
■ Gridding and Dimensionality.....	125
■ Rectilinear Grids.....	125
■ Convex Hull	125
■ Triangular Networks.....	125
■ Estimation Methods	125
■ Surfaces.....	125
■ Color	125
■ Model Subsetting.....	125
Data Content Requirements	125
Direct Data Visualization.....	128
Gridding and Dimensionality.....	132
Rectilinear Grids	133
Convex Hull	137
Triangular Networks.....	141
Estimation Methods	148
Surfaces.....	149
Color	151
Model Subsetting.....	152
Studio Projects Reference Applications	154
EVS Data Input & Output Formats	156
Input	156
Output.....	157
Handling Non-Detects	158
Consistent Coordinate Systems	159
Projecting File Coordinates.....	159
Discussion of File Coordinate Projection	159

Analytical Data.....	163
APDV: Analyte Point Data File Format.....	164
Discussion of analyte (e.g. chemistry) or Property Files	164
APDV: Analyte Point Data File Format.....	166
Discussion of analyte (e.g. chemistry) or Property Files.....	166
Three Dimensional Analyte Point Data File Example.....	168
Discussion of analyte (e.g. chemistry) Files for Fence Sections	171
AIDV: Analyte Interval Data File Format	171
AIDV: Analyte Interval Data File Format.....	174
AIDV File Examples	176
Analyte Time Files Format	177
Discussion of Analyte Time Files	177
Time Domain Analyte Data	179
Time Domain AIDV Example File.....	180
Analyte Time Files Format.....	182
Discussion of Analyte Time Files	182
Pre Geology File: Lithology	185
Pre Geology File: Lithology.....	187
PGF File Examples	190
PGF File Example with Depth.....	194
LPDV Lithology Point Data Value File Format	194
LSDV Lithology Screen Data Value File Format.....	196
GEO: Borehole Geology Stratigraphy.....	199
GEO: Borehole Geology Stratigraphy	202
Geologic File Example: Sedimentary Layers & Lenses.....	206
Geologic File Example: Outcrop of Dipping Strata	207
Geology Files for Production of a Fence Diagram	210
Geology Multi-File.....	211
Geology Multi-File.....	215
ctech_example.gmf.....	219
.PT File Format	220
Overview of Module Libraries	222
Overview of Module Libraries	222
Module Status Indicators	223
Module Input and Output Ports.....	225
krig_3d_geology.....	226
krig_3d	228
krig_2d	230
analytical_realization	232
indicator_realization.....	233
stratigraphic_realization	234
external_kriging	236

make_geo_hierarchy.....	236
3d_geology_map.....	237
geology_to_structured	237
layer_from_surface.....	238
geologic_surfaces	238
geologic_surface	239
indicator_geology	239
edit_horizons	244
horizon_ranking	245
material_mapping.....	246
combine_geology	246
subset_layers.....	247
make_single_layer.....	248
displace_block.....	248
post_samples.....	248
explode_and_scale	251
plume_shell	252
intersection_shell	252
change_minmax.....	255
contour_data	255
volume_renderer	256
adjust_opacity	256
illuminated_lines	256
texture_wave.....	258
slope_and_aspect.....	259
select_data.....	259
read_wavefront_obj.....	259
volumetrics.....	260
cell_volumetrics	264
area_integrate	264
file_statistics.....	264
statistics	265
well_decommission	266
legend	267
axes	268
north	269
add_logo	270
Titles	270
place_text	271
interactive_labels	271
format_string.....	272
external_faces	276

external_edges	276
thin_fence	277
slice	278
isolines	278
cut	279
plume	280
intersection	280
union	281
plume_cell.....	282
footprint.....	283
slope_aspect_splitter	284
crop_and_downsize	285
select_cells.....	286
orthoslice	286
edges	286
bounds.....	286
area_cut	286
surf_cut	287
surf cut example images.....	288
shape_cut	288
buffer	288
tunnel_cut.....	289
overburden.....	289
mask_geology.....	290
node_computation	290
combine_components	294
interp_data.....	295
thickness.....	295
data_translate.....	296
load_evs_field.....	296
EVS Field File Formats and Examples	300
read_vtk	313
read_cad.....	313
read_geometry	314
read_vector_gis	315
raster_to_geology	315
buildings	315
Sample Buildings File	316
read_lines	317
EVS Line File Example	318
Discussion of EVS Line Files.....	318
strike_and_dip	320

Strike and Dip File Example	321
load_glyph	322
symbols	323
save_evs_field	325
write_coordinates	327
write_cad	327
write_vector_gis.....	328
geology_to_raster	328
write_lines.....	329
geology_to_vistas.....	329
export_scene	329
streamlines.....	329
streamline_surface	330
drill_path	330
modpath	331
combine_vect	331
magnitude.....	332
gradient.....	332
capture_zone	333
seepage_velocity	333
regional_averages	335
modflow_converter	335
modflow_converter	336
Modpath DWR/DWZ File Example	336
draw_lines.....	338
polyline_spline	339
triangulate_polygons.....	339
tri_tool.....	339
tubes.....	340
volumetric_tunnel.....	341
cross_section_tubes.....	341
extrude.....	343
drive_glyphs.....	343
place_glyph	343
glyph	343
create_fault_surface	344
create_grid	344
surfmap	344
transform_field	345
transform_group	345
project_field	345
overlay_aerial	346

texture_walls	347
texture_geology	348
georeferenced_output	348
read_image	348
fly_through.....	349
image_transition	350
display_image.....	350
texture_sphere.....	351
texture_cylinder	351
texture_cross_section	351
load_eft	351
read_tcf	351
TCF File Format and Example	351
read_multi_tcf	353
TCF File Format and Example	354
time_value	355
TVF File Format	355
time_geology	361
time_loop.....	361
group_object	361
2d_overlay_group.....	361
trigger_script.....	361
merge_fields.....	362
float_math.....	362
scat_to_tin	363
scat_to_unif.....	363
material_to_cellsets.....	364
loop.....	364
modify_data_3d	364
cell_computation	365
cell_to_node	366
node_to_cell	367
shrink_cells	367
cell_centers	367
interp_cell_data	368
viewer	368
Output Images	369
write_vrml	370
Guidelines for 3D PDF Creation	371
Guidelines for 3D Printing.....	379
Recording (Capturing) 4DIM Files.....	381
merge_fences	383

geologic_surfmap	383
time_field.....	384
video_safe_area	384
advector.....	385
modpath_advvector.....	386
create_spheroid	386
advect_surface.....	386
fence_geology.....	387
file_output.....	388
adaptive_indicator_krig	388
krig_fence	390
fence_geology_map	390
application_notes	391
texture_colors.....	391
C Tech GMS Project File Converter	392
Tools	395
4DIM Playback	396
C Tech GMS Project File Converter	396
VM_to_EVS	400
Images to Animation	403
Georeference Image	406
Index	409

Earth Volumetric Studio Help System

C Tech's Earth Volumetric Studio is the world's leading three-dimensional volumetric Earth Science software system developed to address the needs of all Earth science disciplines. Studio is the culmination of C Tech's 30+ years of 3D modeling development, building upon the developments of legacy software EVS-Pro, MVS and EnterVol. Studio's customizable toolkit is targeted at geologists, environmental engineers, geochemists, geophysicists, mining engineers, civil engineers and oceanic scientists. Whether your project is a corner gas station with leaking underground fuel tanks, a geophysics survey of a large earthen dam combining 3D resistivity and magnetics data, or modeling of salt domes and solution mined caverns for the U.S. Strategic Petroleum Reserves, C Tech's Earth Volumetric Studio has the speed and functionality to address your most challenging tasks. Our software is used by organizations worldwide to analyze all types of analyte and geophysical data in any environment (e.g. soil, groundwater, surface water, air, etc.).

For more information visit ctech.com

- **[Presentation Applications \(.evsp files\)](#)**
- Installing C Tech Software
- **[EVS Data Input & Output File Formats](#)**
- **Workbooks**
 - [Basic Training Workbooks](#)
 - [Workbook 1: Earth Volumetric Studio Basics](#)
 - [Workbook 2: 2D Estimation of Analytical Data](#)
 - [Workbook 3: Exporting from Excel to C Tech File Formats](#)
 - [Workbook 4: Understanding 3D Data](#)
 - [Workbook 5: Packaging Data Into Your Applications](#)
 - [Workbook 6: Geostatistics Overview](#)
 - [Visualization Fundamentals](#)
 - Additional Training Videos at www.ctech.com/training/videos
- **Miscellaneous**
 - [EVS Software License](#)

Presentation Applications

The issues related to EVS Presentation Applications fall into two primary categories:

1. [How to Use Presentation Applications](#) :
 - This topic is geared towards the end user of EVS Presentation Applications.
 - However, creators should familiarize themselves with what is covered in this topic to optimize the usability of their applications.
2. [Creating Presentation Applications](#) :
 - This topic is geared towards the creator of EVS Presentation Applications, which must be a licensed EVS user running either a Floating or Enterprise License.

How to Use EVS Presentations

C Tech's EVS Presentations (EVSP) provide a single file deliverable which allows our customers to provide versions of their Earth Volumetric Studio (EVS) applications to their clients, who can then modify properties interactively.

For example, an EVS Presentation can allow your clients to:

- Choose their own plume levels
- Change Z-Scale and/or Explode distance
- Move slices or cuts through the model
- Draw their own paths for (thin_fence) cross-sections

Therefore, we cannot anticipate the content of these application nor which parameters an EVS user may include in their EVS Presentation Applications. However, there are many fundamental features in EVS Presentation Applications that will be common to all applications, and that is what this topic will address.

Prerequisites to Using EVSPs

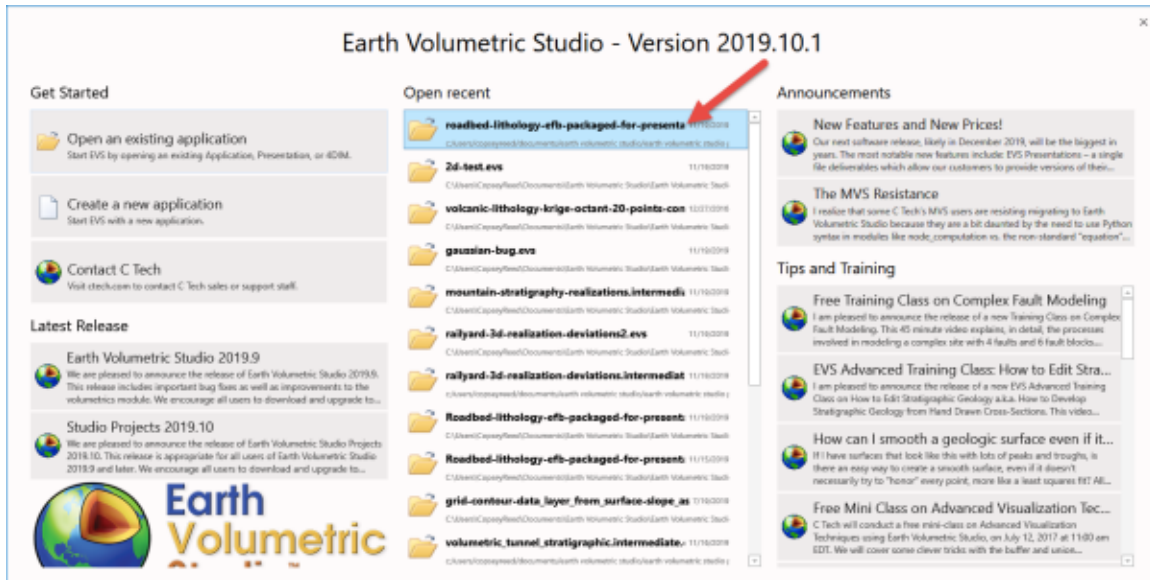
- Verify your computer hardware meets these requirements
- Install EVS: Download the [latest version here](#)
- Installation [instructions are here](#): Note that any license version can open EVSPs.
 - You do not need to install the *Presentation and Demo* version unless you do not have a license.

Opening an EVSP in Earth Volumetric Studio

Opening an EVSP in Earth Volumetric Studio is easy. You should expect EVSP files to be large. The size depends on many factors which only the application creator can change, but it is not unusual for these files to be 10-50 Mb or larger. Remember that this file contains a full 3D volumetric model, with all of its inherent data. Your ability to slice, cut, create plumes, etc., is the result of the 3D volumetric nature of these models.

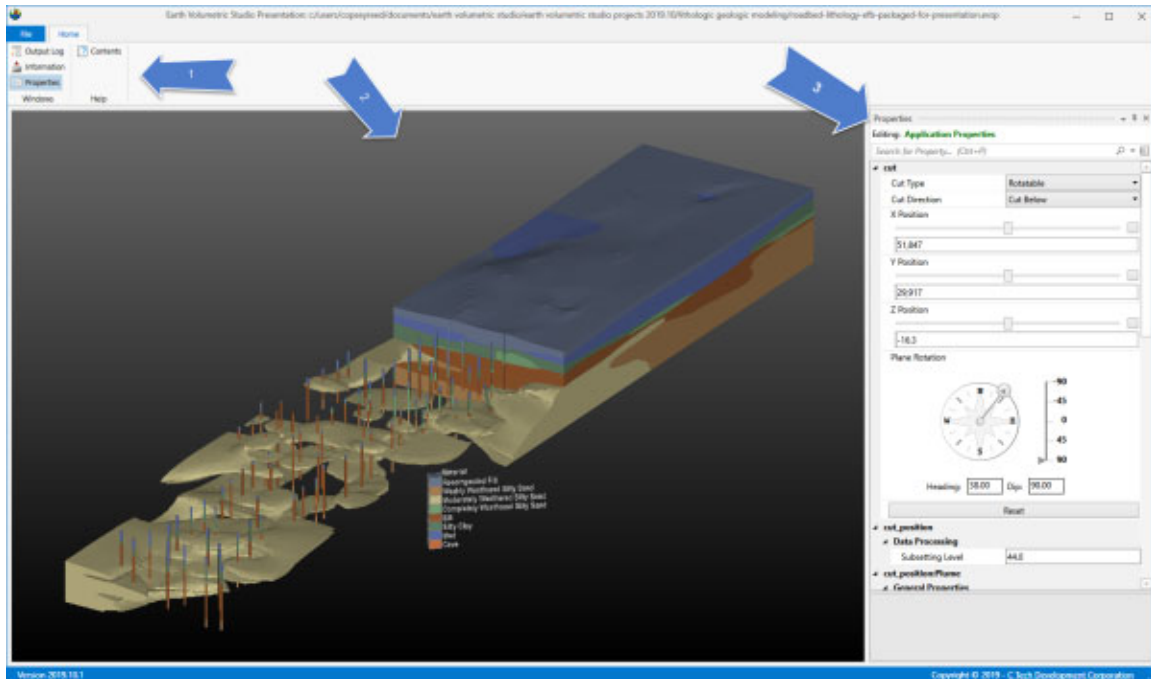
Once you have EVS installed, there are two (and a half) different ways to open an EVSP.

1. When you install EVS, it creates settings in Windows Explorer that allow you to just double-(left)-click on an .EVSP file and it will start EVS and open the file.
2. When you start EVS, the initial window that appears allows you to select any recent file that you've opened. If the file you want is one of the last several that you've already opened, this method works fine. (see first figure below)
 - You can also select the upper left option "Open an existing application" in the first figure below, to access several options for browsing to your file.



When your file opens, what you should normally see is the EVS Presentation Application which has three major components:

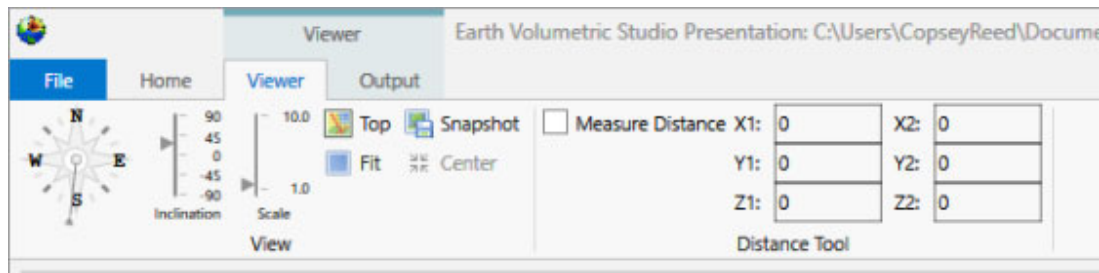
1. The Home Tab: Contains controls to turn on/off various Windows, as well as access to this Help.
2. The Viewer: This is where you'll see and interact with the model.
3. Application Properties: This is where all of the parameters are that you can adjust to affect the model.



The majority of an end-user's attention should (will) be focused on the Viewer and Application Properties. So we'll focus on those first.

Viewer

- To understand how to interact with the viewer, see this topic on [Mouse Interactions](#). You will be able to rotate, pan and zoom on the 3D objects in the viewer.
- When you first click in the Viewer, the Viewer tab will appear and they will remain there permanently.
 - The Viewer tab contains lots of useful features
 - The [Azimuth and Inclination](#) dial and sliders allow you to set any specific view.
 - The *Top* button will bring the model to a Top View (Azimuth South, Inclination 90°)
 - The *Fit* button will cause the model to re-FIT in the Viewer window.
 - The *Center* button is used in conjunction with probing objects in the Viewer.
 - Using CTRL-left-Mouse to probe any object in the view will bring up the Information window with the coordinates and data at that location.
 - Once you have a picked (probed) location, the *Center* button makes that location the center for rotations and zooming.
 - *Snapshot* allows you to capture the Viewer (at its current resolution) and save that image in common image file formats.
Note: the *View Scale* parameter if added to Application Properties, will affect the output resolution.



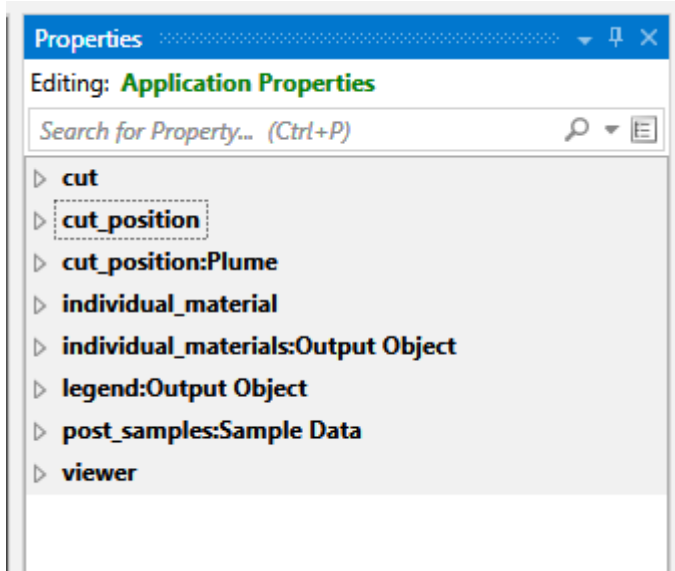
- The Distance Tool region of the Viewer tab contains the Measure Distance options
 - To use this, first turn on the *Measure Distance* check-box.
 - Now, probe two points (CTRL-left-Mouse). As you probe it will populate the coordinates in this tab.
 - When the second point is done, the Information window will appear with detailed information about the points you've probed and their distances (X, Y, Z and total)
- The Output sub-tab is inactive for Presentation Applications.

NOTE: The content in the Viewer can change dramatically depending on the application and parameters made available by the EVSP creator.

Application Properties

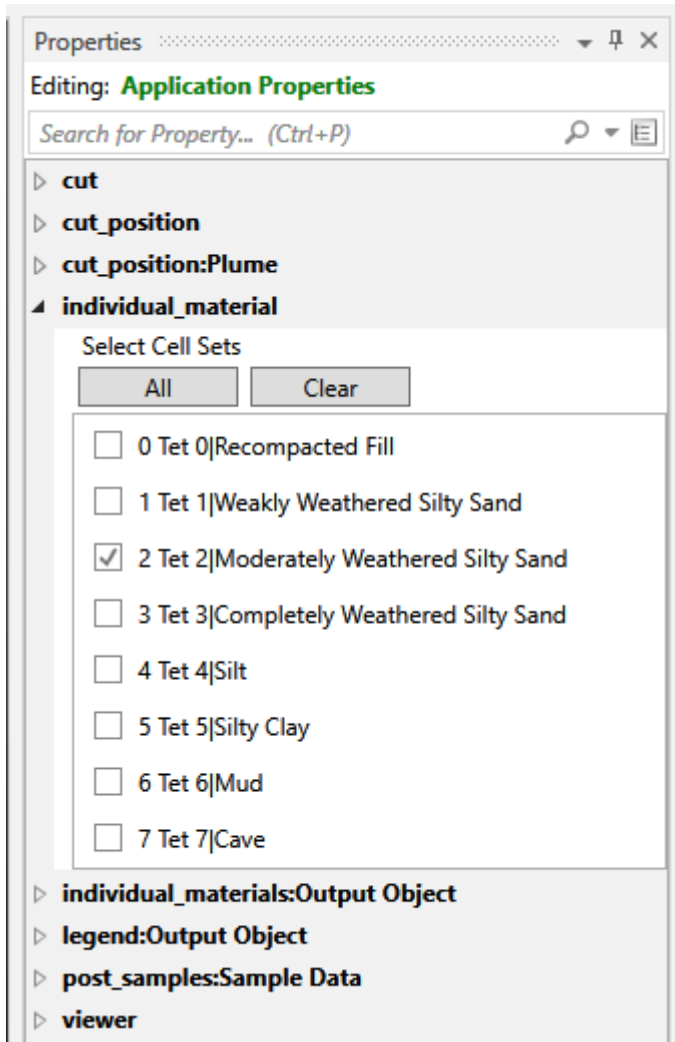
The parameters made available by the EVSP creator will be in Application Properties. This is the magic of EVSPs. The ability to interact with and change the objects in the Viewer are virtually limitless, which makes it impossible for us to address all of the parameters an end-user may encounter. Instead this help will only address how to find and interact with those parameters, and show one example of what is possible.

- Parameters reside in hierarchical groups based on the names of the modules in the application.
- The creator can change those module names to make their function more obvious to end users. Once a file is saved as an EVSP, it cannot be further edited in any way.
- In the EVSP above there are two modules (groups) that can be seen without scrolling: *cut* and *cut_position*.
- Every module has a triangular button before its name, that allows you to collapse or expand that group. Below you'll see all of the modules (groups) collapsed.



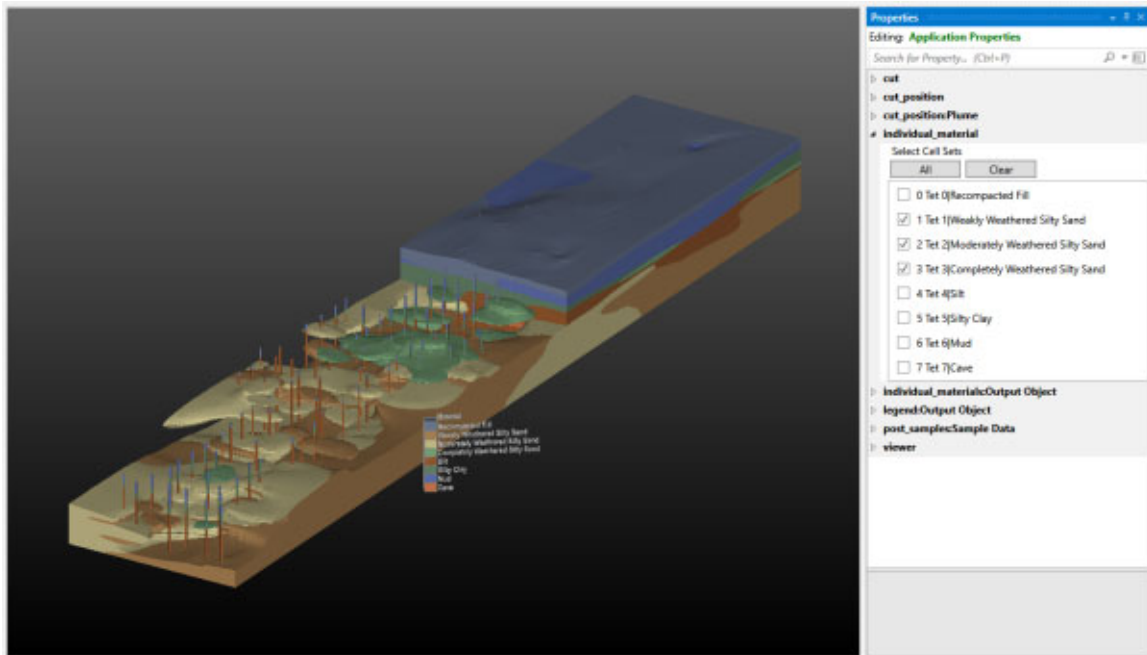
From this list of the collapsed groups we can anticipate that this application provides some level of control over cutting and cut position, the materials displayed, the legend, post_samples (the borings) and some Viewer properties.

If we expand the individual_material group, we see a set of check boxes: one for each geologic material in the model.

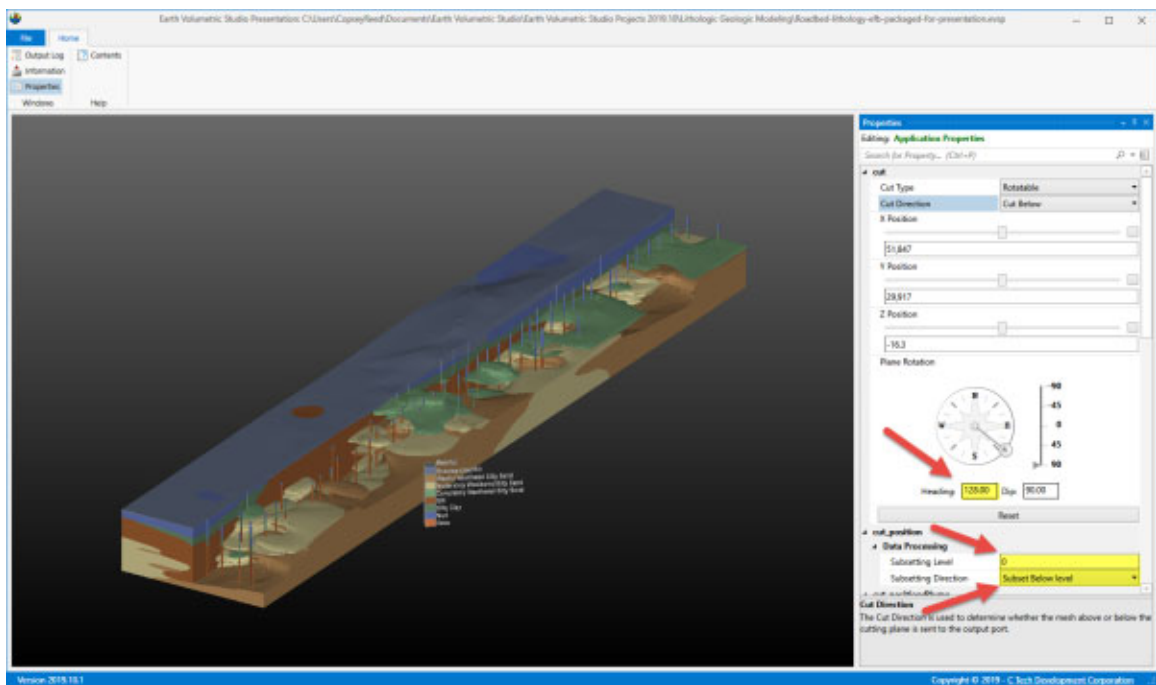


We can see that only "Moderately Weathered Silty Sand" is selected, and in the Viewer, the Southern portion of the model (below the cut) is displaying only this one (beige) material.

Selecting the materials listed above and below give us this:



If we now change 3 parameters as shown below, we get a dramatically different output:



When we start considering all of the permutations of the parameters in this application, the number of possible outputs is staggering:

- cut:Plane Rotation

- X, Y, Z Position of the center of rotation. Each coordinate can be set to .0001 precision, but let's just say we only allowed 100 steps in X, Y & Z = $100^3 = 1,000,000$ possible permutations
- Heading angle: Can be changed by 100th of a degree = 36,000 possibilities, but let's just say one degree = 360 possible permutations
- Dip angle: If only one degree increments = 180 possible permutations
- individual_material: Select Cell Sets : with 8 materials there are 255 possible permutations of on/off.
- Visibility ON/OFF for individual_materials, post_samples (borings) and the legend = $2*2*2 = 8$ possible permutations

Note: cut_position's Subsetting Level wasn't included because it provides a quicker way to move the cut, but not more output permutations.

Even being conservative regarding the precision with which we can adjust the above parameters, this EVSP has: $1,000,000 * 360 * 180 * 255 * 8 \dots$ OVER 132 Trillion possible outputs!

Creating EVS Presentations

The development of EVS Presentations (EVSP) from EVS Applications will nearly always require modification to your EVS application. The key steps are:

1. Save your application.
2. Replace Disallowed Modules with acceptable replacements, if applicable.
 - Note: Not all Disallowed Modules have replacements which can be included in EVS Presentations.
3. **Package** all data files referenced in any modules.
 - There cannot be any externally referenced data files.
 - Some modules cannot be packaged and are automatically replaced by the packaging process. These include:
 1. read_vector_gis
 2. read_cad
 3. read_wavefront_obj
 - Best if you don't do this step sooner.
4. Add all desired module properties to Application Properties, so they can be accessed once the application is saved as an EVS Presentation.
5. Backup your application as an EVS application to serve as an editable backup.
 - This is essential, should you wish to add additional module properties later.
6. Convert the application to an EVS Presentation (.evsp file).
 - This is not a reversible process, which is why you want the backup from step 5.

What is Packaged Data

- Applications with packaged data include the data with the .EVS file.

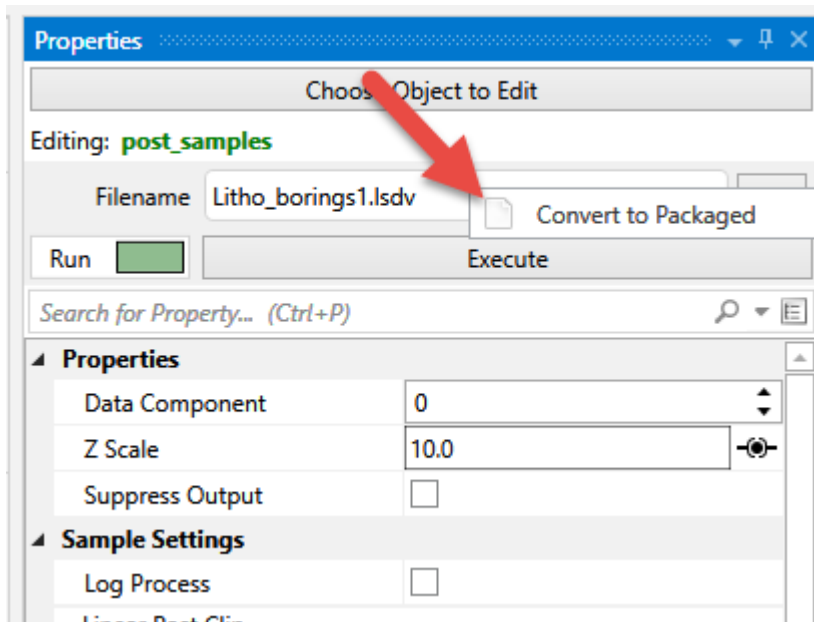
- Packaged data:
 - Is data files that have been *incorporated into your application*
 - Can be most EVS files types
 - Must be done for all modules, if you plan to create an EVS Presentation
- Packaging of data can be done on a module-by-module basis, or for your entire application in one step.
- Applications with packaged data include the data with the .EVS file.

How to Package your data:

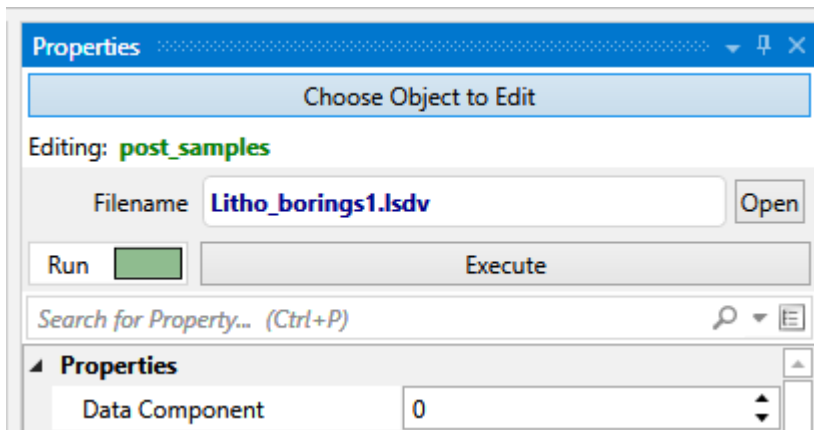
Module-by-Module method:

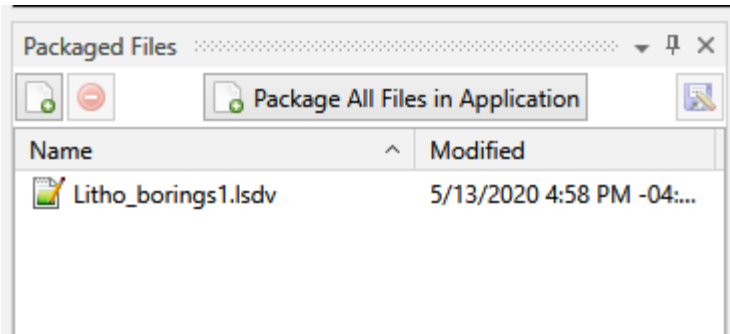
Right click on the filename in each module that read data (be sure you first remove all Disallowed Modules)

You will then see this message:



Select - "Convert to Packaged" and you should see the selected Filename change to **Bold Blue** text and also appear in the Packaged Files window.

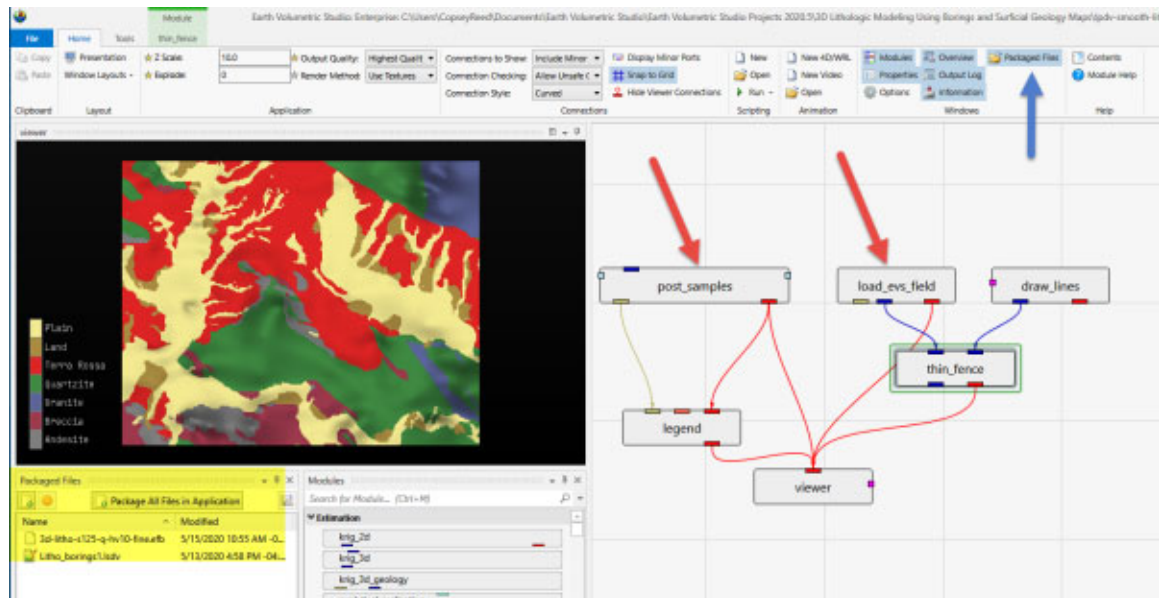




Packaging All Files in One Step:

Please note that in the image above, we also see the "Package All Files in Application" button.

This button will find all of the files in your application and will package each one. After pressing this button, for this application we can see that there are only two modules which read files and their files are both packaged. Also note that if you can't find the "Packaged Files" window, make sure it is on (selected) in the Windows section of the Home tab.



NOTE: EVS will package the files for modules that are Disallowed for EVS Presentations. Just because a module's files will package is not a guarantee that it is allowed in an EVS Presentation.

Disallowed Modules and Replacements:

The following table lists all disallowed modules and their replacements if there are any. Some modules, primarily interactive modules (e.g. modify_data_3d and make_geo_hierarchy) and export modules (e.g. write_cad and write_vector_gis).

Disallowed Module	Category	Replacement
external_kriging	Estimation	load_evs_field
krig_2d	Estimation	load_evs_field

krig_3d	Estimation	load_evs_field
scat_to_tri	Estimation	load_evs_field
scat_to_unif	Estimation	load_evs_field
modify_data_3d	Estimation/editing	none
combine_geology	Geologic modeling	load_evs_field
edit_horizons	Geologic modeling	load_evs_field
horizon_ranking	Geologic modeling	load_evs_field
indicator_geology	Geologic modeling	load_evs_field
krig_3d_geology	Geologic modeling	load_evs_field
make_geo_hierarchy	Geologic modeling	none
material_mapping	Geologic modeling	load_evs_field
drill_path	Geometry	none
analytical_realization	Geostatistics	load_evs_field
indicator_realization	Geostatistics	load_evs_field
stratigraphic_realization	Geostatistics	load_evs_field
well_decommission	Geostatistics	none
read_geometry	Import	load_evs_field
geology_to_raster	Export	none
geology_to_vistas	Export	none
georeferenced_output	Export	none
save_evs_field	Export	none
write_vector_gis	Export	none
write_cad	Export	none
write_coordinates	Export	none
write_lines	Export	none

cell_computation	Python (Enterprise License Only)	load_evs_field (for Floating Licens
node_computation	Python (Enterprise License Only)	load_evs_field (for Floating Licens
trigger_script	Python (Enterprise License Only)	none (for Floating Licens

Restricted Functions:

In addition to the disallowed modules, certain integrated functions are restricted such as:

- Writing VRML files (for 3D PDFs)
- Recording 4DIMs
- Animator
 - Creating 4DIMs
 - Creation of bitmap animations (e.g. .AVI files)
- Tools Tab: All functions
- Open Python Script
 - Enterprise License customers may package Python scripts using the trigger_script module

EVSP Outputs:

The EVSP itself is intended to be the primary output, therefore the ability to create any outputs while and end-user works with an EVSP is limited to bitmap images. End-users will not be able to write any 3D type outputs (4DIM, VRML, Shapefiles, CAD files, etc.).

EVS Installation and Licensing

Earth Volumetric Studio is available only as download from <https://client.ctech.com/>

There are four different licensing types:

1. Presentation and Demo
2. Fixed
3. Floating
4. Enterprise

Detailed instructions for installation and licensing of all license types are [available here](#).

Basic Training

- [Earth Volumetric Studio Basics](#)
- [2D Estimation of Analytical Data](#)
- [Exporting from Excel to C Tech File Formats](#)
- [Understanding 3D Data](#)
- [Packaging Data into Applications](#)
- [Geostatistics Overview](#)
- [Visualization Fundamentals](#)

Video Tutorials at www.ctech.com

The workbooks in this help cover only the most basic functionality. We offer two levels of training videos which can be accessed at [ctech.com](http://www.ctech.com) which provide more comprehensive training from a novice to an advanced user.

We offer two levels of training videos in addition to the limited workbooks which are built-into the software help system (and are included online). The training videos include:

- [Basic Free How-To Video Tutorials](#)
- [Comprehensive Video Training Classes](#)

As stated, the first category of training videos are free, whereas the second category are not. These classes range from \$350 to \$800 per person for classes that are 3 to 12 hours in duration. All of these classes are offered with a money-back guarantee. We ask that you use the knowledge you gained in the class for 30 days. At the end of that time if you feel that the class was not valuable to you and your company, we will refund the cost of the class. These provide students with far more than the mechanics of using Earth Volumetric Studio. The classes are taught by our most senior personnel with decades of experience with C Tech's software and experience in earth science consulting projects including litigation support. The courses focus as much on **why** we do things as *how* they are done. Our goal is to graduate modelers with a deeper understanding of critical issues to consider in their daily modeling tasks, whether they are doing a quick first look at a corner gas station or working on

litigation support for a Superfund site. New classes are announced on C Tech's Mailing List and the registration form to enroll in these classes is on the website.

The EVS Home Tab

The EVS Home Tab provides access to a huge amount of functionality in the software. Understanding the various sections is critically important to your successfully mastering the software.

Some of the sections are very simple to explain and will be addressed in this topic whereas others are deserving of their own topic.

CLIPBOARD: The Clipboard is the simplest and could be replaced by remembering CTRL+C (for copy) and CTRL+V (for paste). However, it is important that we remind you of the power that Copy & Paste provides in the Application window. Not only can you copy and paste one-or-more modules, but you can paste into a text editor to save modules as a text file for later use.

LAYOUT: The Layout section has only two parts/topics. *Presentation* engages "Presentation Mode", which will eventually allow application designers to create applications that can have all data packaged and encrypted so that the application (with limited functionality) will run without needing a License. *Windows Layouts* provides access to your saved Windows Layouts to allow you to quickly select any previously saved arrangement of windows. Note: Layouts are saved under File...Options.

APPLICATION: The Application section has 4 functions:

1. **Z Scale:** The master parameters for both Z Scale and Explode should be set only here or in the Application Properties window. All modules with these parameters that are "Linked" will inherit these values and will update when these values are changed.
2. **Explode**
3. **Origin: Reset or Edit. This allows you to view or change the Application Origin. This is an important concept in EVS that maintains precision and makes web viewing possible. Typically the first data file you read will be used to determine the origin by finding the centroid of that data.**
4. **Output Quality:** The master parameter for output quality is set here. This refers to the selection of a low or high quality EVS Field File in one or more *Load_EVS_Field* module in your application.

CONNECTIONS: The Connections section has 6 functions:

1. **Connections to Show:** Include Minor Ports is recommended
2. **Connection Checking:** The various levels are there to provide some protection against inappropriate connections which can cause application crashes. However, less restrictive levels (e.g. Allow Unsafe Connections) do allow users to make connections between modules before modules have run so that the appropriateness of the data types can be confirmed.

3. **Connection Style:** The two styles are Curved and Straight. Curved is recommended.
4. **Display Minor Ports:** is a toggle. It is blue when on.
5. **Snap to Grid:** is a toggle. It keeps your modules more aligned in your application .
6. **Hide Viewer Connections:** is a toggle. All red viewer connections are converted to a short red "tail" on the module being connected to the viewer only. It reduces clutter.

SCRIPTING: This section opens up the Python scripting functions.

ANIMATION: This section opens up the Animation functions.

WINDOWS: The Windows section has 7 functions all of which are toggles which control the visibility of a major EVS Window:

1. Modules
2. Properties
3. Options
4. Overview
5. Output Log
6. Information
7. Packaged Files

HELP: The Help section has only two functions: Contents exposes the main EVS Table of Contents, and Module Help opens up the help for the currently selected module.

Python Scripting

The Scripting section of the Home tab has only three functions:

1. New: Creates a new Python script
2. Open: opens an existing Python script
3. Run: runs an existing Python script

When a new script is created or an existing script is opened, the Python window is exposed, where all Python controls are located:

At the top of the window there are a series of (icon) buttons which provide the following functions (listed from left to right):

1. Open
2. Save
3. Save-As
4. Cut
5. Copy
6. Paste
7. Run
8. Record (turns red when on)
9. Undo
10. Redo
11. Decrease Indentation (of selected lines)
12. Increase Indentation (of selected lines)
13. Comment-Out selected lines
14. Uncomment selected lines
15. Convert Tabs to Spaces (on selected lines)
16. Remove all trailing whitespace characters (on selected lines)
17. Find or Replace
18. GoTo a specific line

On the far right side are two separate buttons which each have multiple options:

1. The first is the Open Windows button which provides these functions:
 2. The second (far right) is the Options button which controls display of Whitespace.
- The methodology for using Python in EVS is covered in a [Premium Training Video](#).

Python Functions & Operators

Earth Volumetric Studio uses Python 3.3

A listing of Python Functions & Operators can be found at python.org. Below are links to relevant pages:

- [Functions](#)
- [Math Operators](#)
- [String Operators](#)
- [Date and Time Operators](#)
 - [Date & time syntax](#)

Please note: C Tech **does not** provide Python programming or syntax assistance as a part of Technical Support (included at no additional cost with current maintenance). Python scripting and functionality is provided as an advanced feature of Earth Volumetric Studio, but is not required to use the basic functionality.

Below are Earth Volumetric Studio specific functions which provide means to get and set parameters and to act upon the modules in the libraries and network.

evs.get_application_info():

Gets basic information about the current application.

Keyword Arguments: None

evs.get_module(module, category, property):

Get a value from a module within the application.

Keyword Arguments:

module: the name of the module (required)

category: the category of the property (required)

property: the name of the property to read (required)

evs.get_module_extended(module, category, property):

Get an extended value from a module within the application.

Keyword Arguments:

module: the name of the module (required)

category: the category of the property (required)

property: the name of the property to read (required)

evs.set_module(module, category, property, value):

Set a property value in a module within the application.

Keyword Arguments:

module: the name of the module (required)

category: the category of the property (required)

property: the name of the property to set (required)

value: the new value for the property (required)

evs.get_port(module, port, category, property):

Get a value from a port in a module within the application.

Keyword Arguments:

module: the name of the module (required)

port: the name of the port (required)

category: the category of the property (required)

property: the name of the property to read (required)

evs.get_port_extended(module, port, category, property):

Get an extended value from a port in a module within the application.

Keyword Arguments:

module: the name of the module (required)

port: the name of the port (required)

category: the category of the property (required)

property: the name of the property to read (required)

evs.set_port(module, port, category, property, value):

Set a property value in a port in a module within the application.

Keyword Arguments:

module: the name of the module (required)

port: the name of the port (required)

category: the category of the property (required)

property: the name of the property to set (required)

value: the new value for the property (required)

evs.connect(starting_module, starting_port, ending_module, ending_port):

Connect two modules in the application.

Keyword Arguments:

starting_module: the starting module (required)

starting_port: the port on the starting module (required)

ending_module: the ending module (required)

ending_port: the port on the ending module (required)

evs.disconnect(starting_module, starting_port, ending_module, ending_port):

Disconnect two modules in the application.

Keyword Arguments:

starting_module: the starting module (required)

starting_port: the port on the starting module (required)

ending_module: the ending module (required)

ending_port: the port on the ending module (required)

evs.delete_module(module):

Delete a module from the application.

Keyword Arguments:

module: the module to delete (required)

evs.instance_module(module, suggested_name, x, y):

Instances a module in the application.

Keyword Arguments:

module: the module to instance (required)

suggested_name: the suggested name for the module to instance (required)

x: the x coordinate (required)

y: the y coordinate (required)

Result - The name of the instanced module

evs.get_module_position(module):

Gets the position of a module.

Keyword Arguments:

module: the module (required)

Result - A tuple containing the (x,y) coordinate

evs.refresh():

Refreshes the viewer and processes all mouse and keyboard actions in the application. At each occurrence of this function, your scripts will *catch-up* to behave more like manual actions. In most cases this is the only way that you can see the consequences of the commands reflected in your viewer upon this function's execution.

This is a potentially unsafe operation under certain (hard to predict) circumstances.

If your script is malfunctioning with this command, try removing or commenting all occurrences.

We do not recommend using this command within Python scripts executed by the trigger_script module.

Keyword Arguments: None

evs.suspend():

Suspends the execution of the application until a resume is called.

evs.resume():

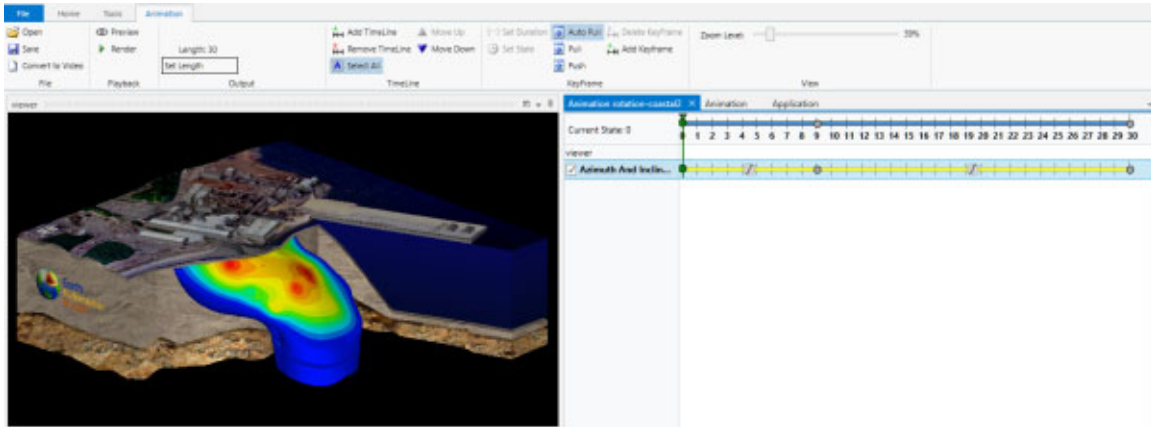
Resumes the execution of the application, causing any suspended operations to run

Animation

The Animation section of the Home tab has only three functions:

1. New 4D/WRL: Creates a new 4DIM or WRL (3D PDF) Animation
2. New Video: Creates a new bitmap animation
3. Open: opens an existing animation file. **btw: animations are saved as .evsani files.**

When a new Animation is created or an existing Animation is opened, the Animation tab is exposed:



FILE: The File section has three functions: Open and save .evsani files and the option to Convert 4DIM/PDF animation files to Video files.

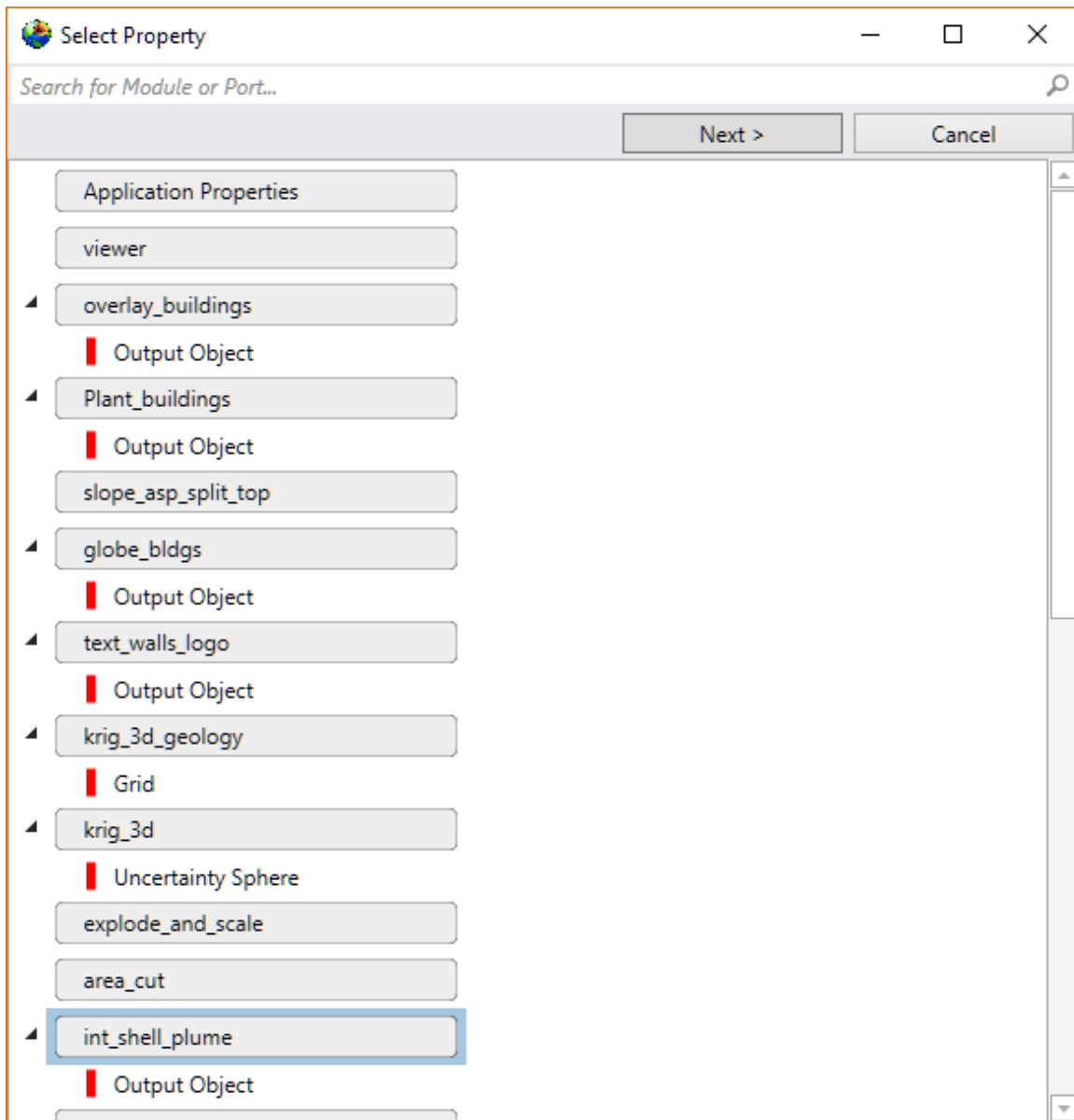
PLAYBACK: There are two options here. Preview allows you to play your animation without writing any output. Render creates the final target output.

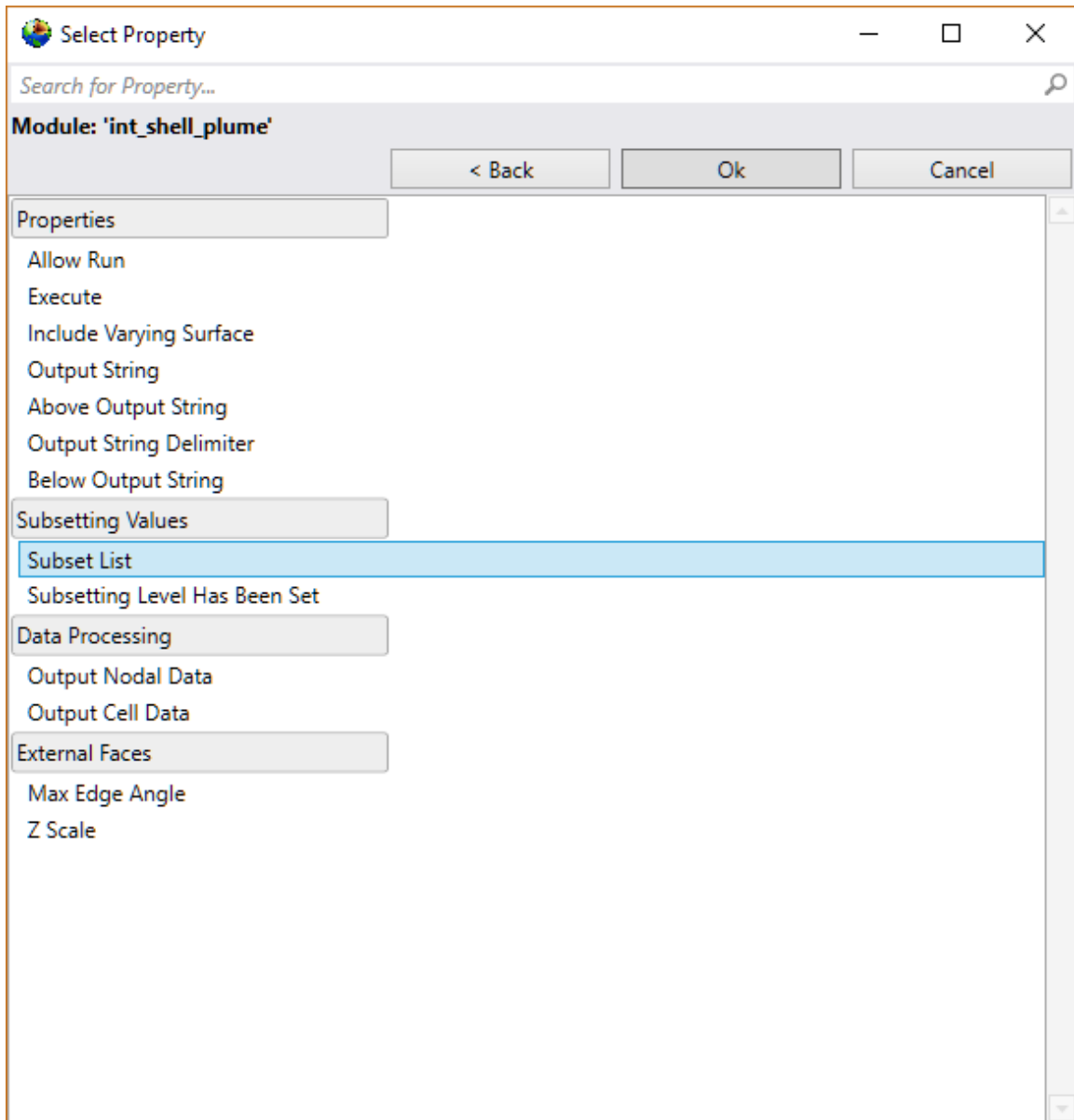
OUTPUT: Allows you to set (change) the length (duration) of the animation. There are two options when changing duration: Truncate or Scale Key Frames.

TIMELINE: There are 5 functions in the Timeline section: Timelines are the variables that are being animated. They can be as complex as the view in the viewer (Azimuth, Inclination, Scale, Roll), as simple as the visibility of a single object in your application, or a numeric value like a plume level or the position of a slice plane that can vary linearly or with a cosine variation which provides a slow start and stop.

1. **Add Timeline:** Button: Allows you to specify any parameter in your application.
2. **Remove Timeline:** Button: Allows you to any existing (selected) timeline.
3. **Select All:** Toggle: Select all timelines, so that they will all be affected by Keyframe actions.
4. **Move Up:** Button: Allows rearranging timelines.
5. **Move Down:** Button: Allows rearranging timelines.

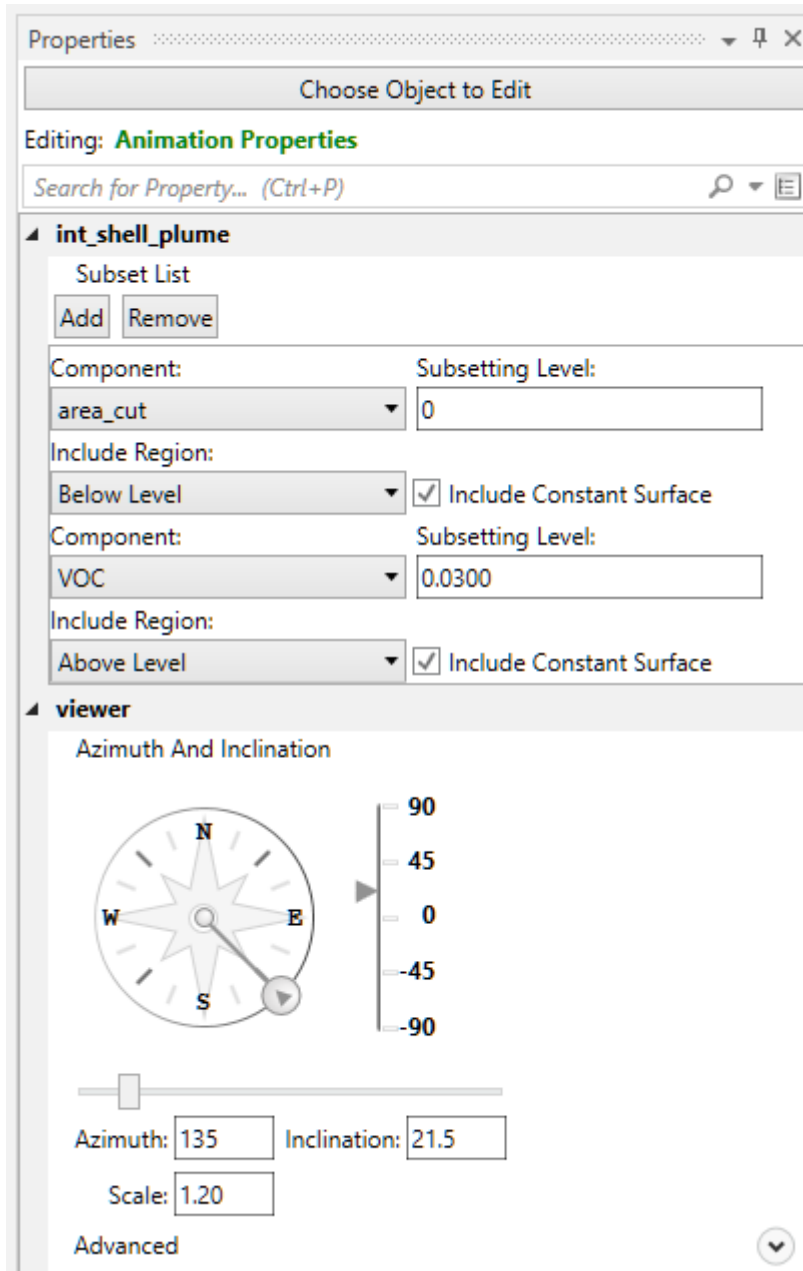
The Add Timeline Windows are shown below. The top image is where you select the module and below that are the functions/parameters in that module.





For this application we've selected `int_shell_plume` which is an `intersection_shell` module. By selecting "Subset List" we actually have the potential to animate a list of parameters in a single timeline.

In the Animation Properties window we can see all of the timeline parameters, and the Subset List has two items: `area_cut` and `VOC levels`.



KEYFRAME: There are 7 functions in the Keyframe section: Keyframes are the time markers that establish when actions occur along timelines.

When trying to understand concepts like Pull and Push, remember that everything is relative to the Animator. Push transfers parameters in the Animator UP TO the Application.

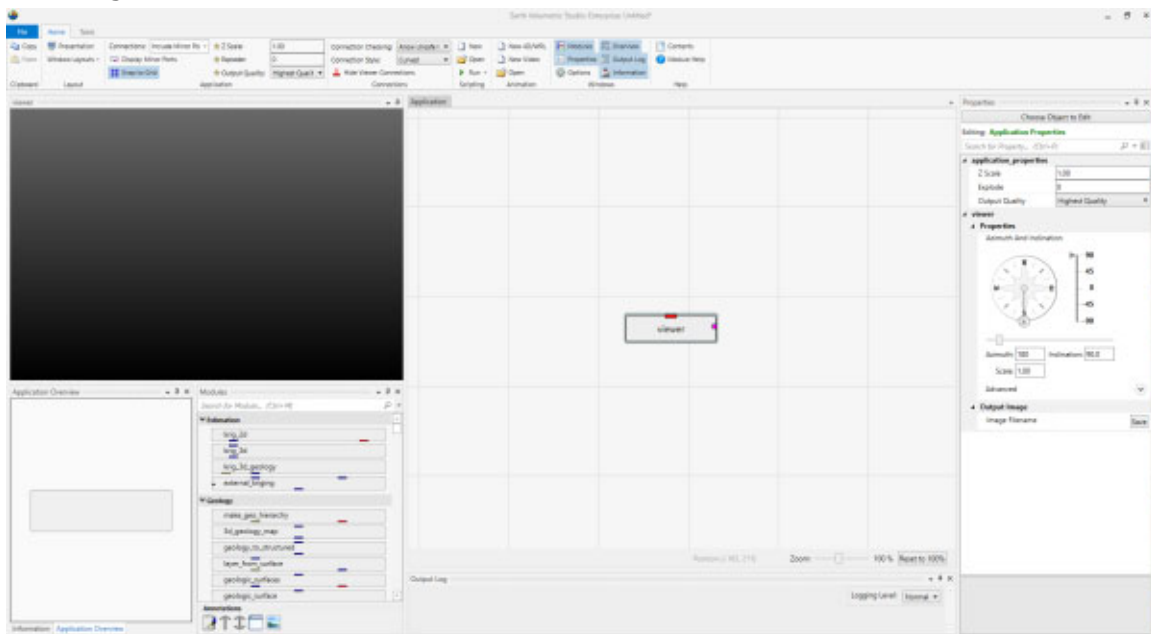
1. **Set Duration:** Button: This button is only active when a keyframe (circle) is selected. It allows you to insert or delete states (or time) in the middle of an animation. You can change the duration of the portion of the timeline that corresponds to the timeline up to the selected keyframe.

2. **Set State / Set Time:** Button: This button is only active when a keyframe (circle) is selected. It allows you to set the states (time) in the middle of an animation. You can change the duration of the portion of the timeline that corresponds to the timeline up to the selected keyframe.
3. **Auto Pull:** Toggle: When on, as new Keyframes are added, the values are automatically pulled from the application.
4. **Pull:** Button: When this button is pushed, and a Keyframe is selected, all active timelines will pull values from the application, thus changing the values at that keyframe.
5. **Push:** Button: When this button is pushed, and a Keyframe is selected, all active timelines will push their values into the application, thus changing the values in their corresponding modules (or the viewer) at that keyframe.
6. **Delete KeyFrame:** Button:
7. **Add KeyFrame:** Button:

VIEW: The Zoom Level changes how much of the timeline you can see.

The Earth Volumetric Studio Environment

If you have not already done so, start EVS at this time. To start EVS, double-click on the Earth Volumetric Studio icon located in the programs listing of your Windows Start Menu. If you have not changed the default settings, EVS will open to show the following subwindows.



Each window can be resized, moved or undocked from the main window. This makes it easy to optimally use multiple monitors.

Visual Programming

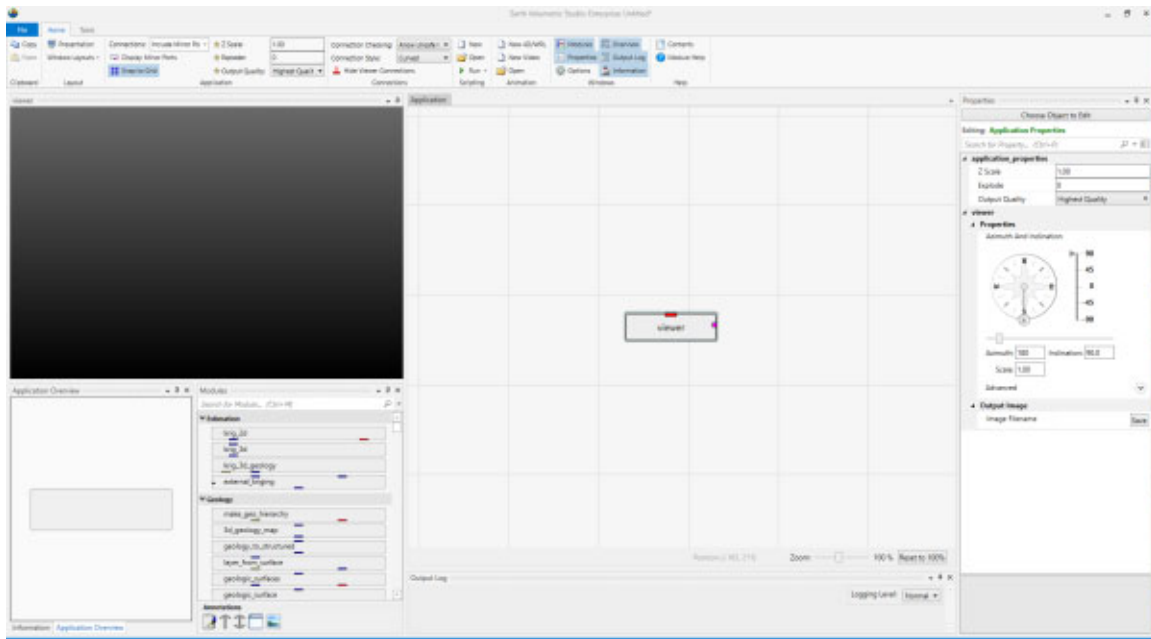
EVS uses a visual programming paradigm. The Earth Volumetric Studio application has several primary subwindows. Each can be moved, resized, undocked or stacked with tabs.

- The viewer is in the upper left. It will be a faded black (or whatever color you set as default) and empty until you have an application that has run.
- Below the viewer (left) is the Application Overview window. It automatically resizes to show you a thumbnail view of your application. It allows you to navigate, zoom and pan your application.
- To the right of the Application Overview window are the Modules which are listed in 19 sublibraries. The easiest way to find and use a module is to type the first 1-2 letters of the module name. The list will expand to show only those modules that include those letters. You can then instance (copy to your application) any module by selecting it and hitting ENTER, double clicking, or dragging it to the Application.
- The Application window is in the center. This is where you will add the modules and interconnect them to create a custom application to perform your required tasks
- Below it is the Output Log which provides useful information when modules run.
- The Properties window is where you can set the parameters for each module. By double clicking on any module (or connection) in your application, you can set its properties.

The *Application area* is the workspace where you use the modules to build a custom application. The Modules libraries are like a toolbox, in that there are many different tools that serve different purposes, but cannot be used until they are taken out of the toolbox. Similarly, the *Application window* is similar to a workbench: it is the place where the tools are used to create models. Most modules in the library have *input* and/or *output ports*. These are colored regions (ports) on the modules which represent the pipelines through which data flows to and from each module. A collection of modules that have their ports connected by pipelines comprise an EVS Application (.evs file).

The Earth Volumetric Studio Environment

If you have not already done so, start EVS at this time. To start EVS, double-click on the Earth Volumetric Studio icon located in the programs listing of your Windows Start Menu. If you have not changed the default settings, EVS will open to show the following subwindows.



Each window can be resized, moved or undocked from the main window. This makes it easy to optimally use multiple monitors.

Visual Programming

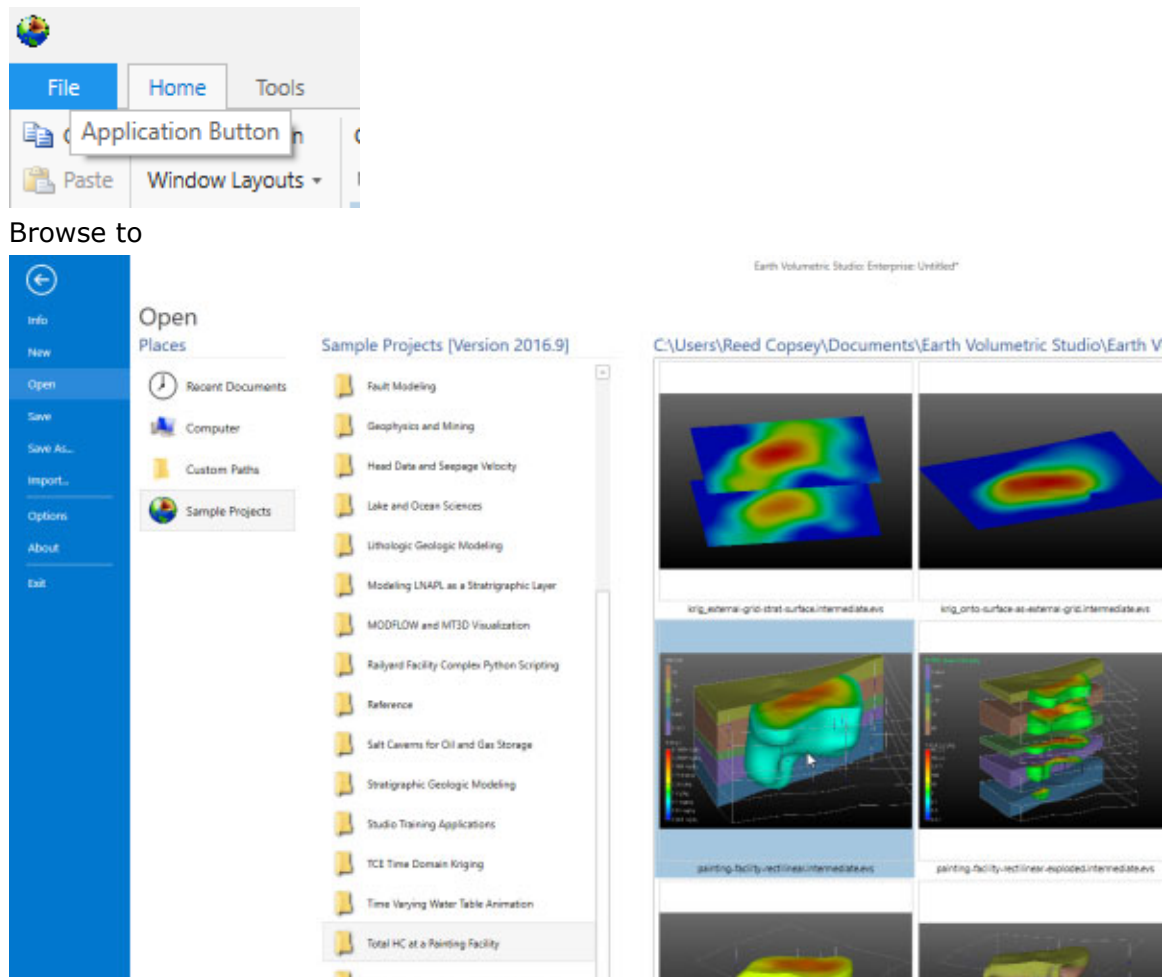
EVS uses a visual programming paradigm. The Earth Volumetric Studio application has several primary subwindows. Each can be moved, resized, undocked or stacked with tabs.

- The viewer is in the upper left. It will be a faded black (or whatever color you set as default) and empty until you have an application that has run.
- Below the viewer (left) is the Application Overview window. It automatically resizes to show you a thumbnail view of your application. It allows you to navigate, zoom and pan your application.
- To the right of the Application Overview window are the Modules which are listed in 19 sublibraries. The easiest way to find and use a module is to type the first 1-2 letters of the module name. The list will expand to show only those modules that include those letters. You can then instance (copy to your application) any module by selecting it and hitting ENTER, double clicking, or dragging it to the Application.
- The Application window is the in the center. This is where you will add the modules and interconnect them to create a custom application to perform your required tasks
- Below it is the Output Log which provides useful information when modules run.
- The Properties window is where you can set the parameters for each module. By double clicking on any module (or connection) in your application, you can set its properties.

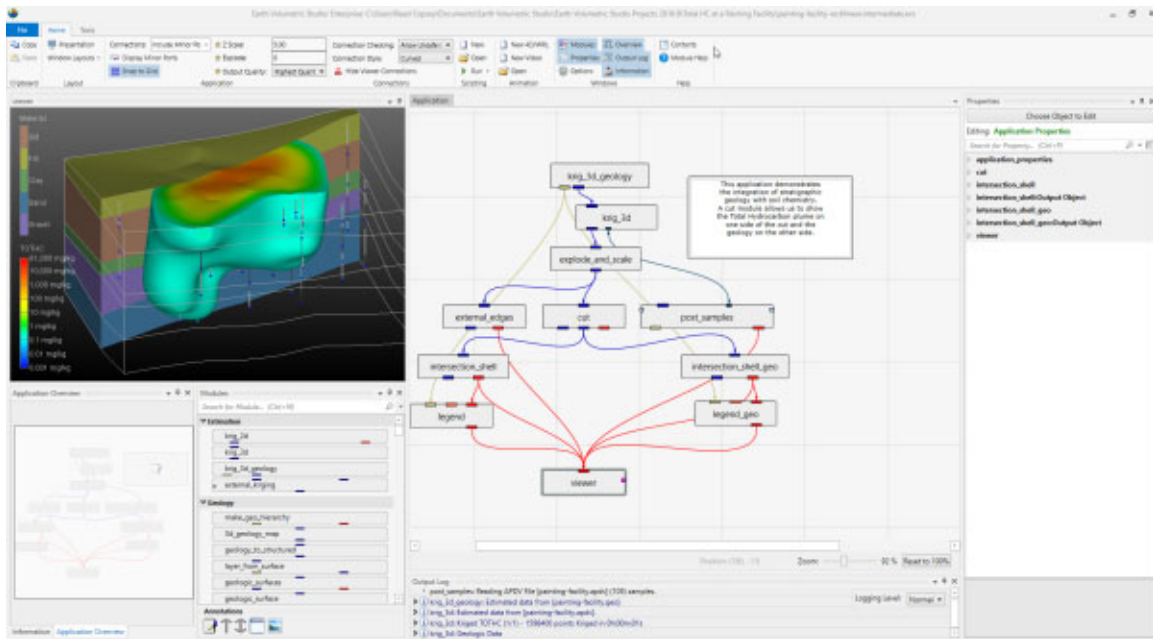
The *Application area* is the workspace where you use the modules to build a custom application. The Modules libraries are like a toolbox, in that there are many different tools that serve different purposes, but cannot be used until they are taken out of the toolbox. Similarly, the *Application window* is similar to a workbench: it is the place where the tools are used to create models. Most modules in the library have *input* and/or *output ports*. These are colored regions (ports) on the modules which represent the pipelines through which data flows to and from each module. A collection of modules that have their ports connected by pipelines comprise an EVS Application (.evs file).

Load an Application

Let's load an application to get an idea of how EVS works.



The application will run and in less than one minute you will see:



Transformations with the Mouse

Now that we have an application loaded, let's investigate the many ways we can interact with it.

Rotate the model

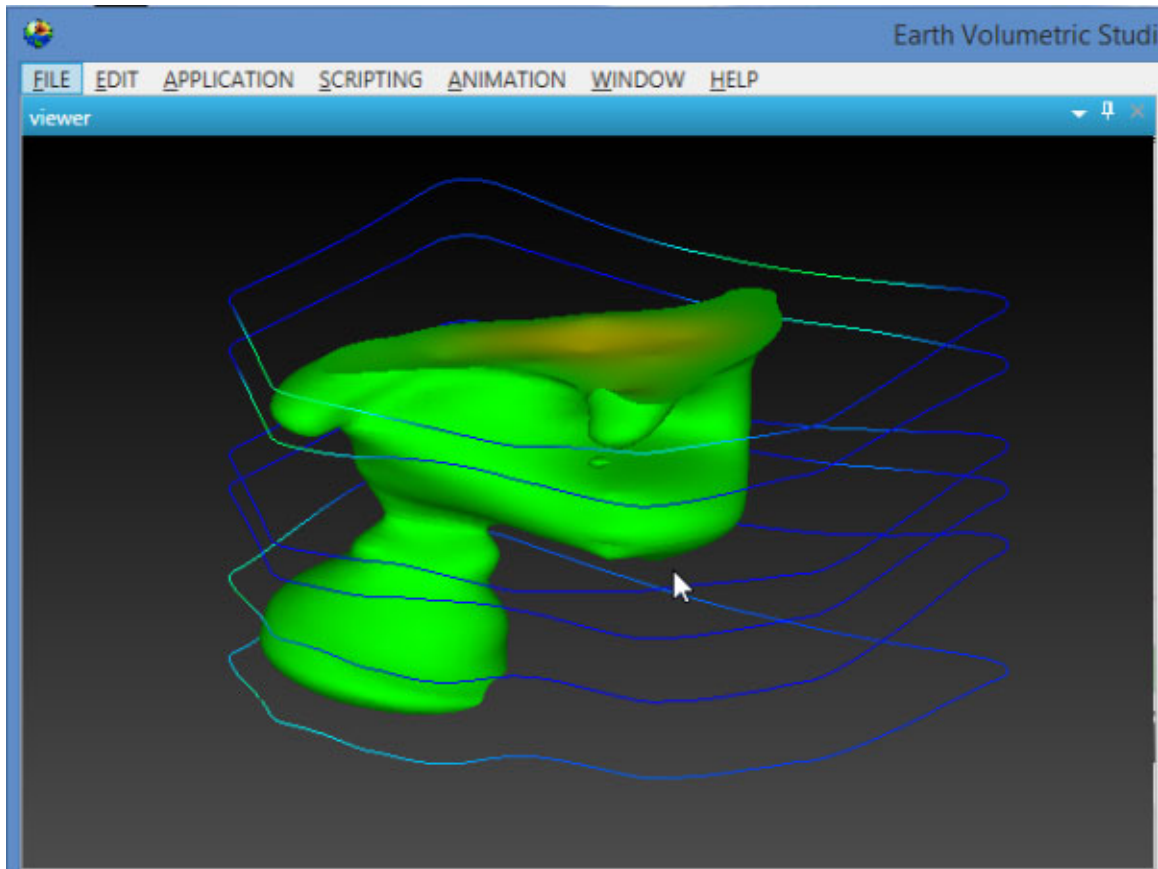
- Hold down the left mouse button and move the mouse pointer in various directions. The model rotates.
- Vertical motions rotate the model about a horizontal axis.
- Horizontal motions rotate the model about a vertical axis.
- Roll is suppressed so that mouse rotations always keep vertical objects (e.g. telephone poles) vertical.

Scale (zoom) the model

- The wheel on wheel mice also zooms in and out.
- Alternate method:
 - Hold down both the Shift key and the left mouse button (or the middle button alone).
 - Keeping the Shift key and mouse button held down, move the mouse pointer downward or to the left. As we do, the model scales down. Moving the mouse pointer upward or to the right scales up.

Move (Translate or Pan) the model

- Hold down the right mouse button and drag the object up, down, and around, then center the model.



Mouse-controlled operations

What to do

Translate	Drag the object with the right mouse button (RMB)
Rotate	Drag the object with the left mouse button. (LMB)
Scale	Use the wheel to zoom in and out or Hold down the Shift key and drag the object with the left mouse button. (Shift-LMB) or Use the middle mouse button or wheel as a button without Shift

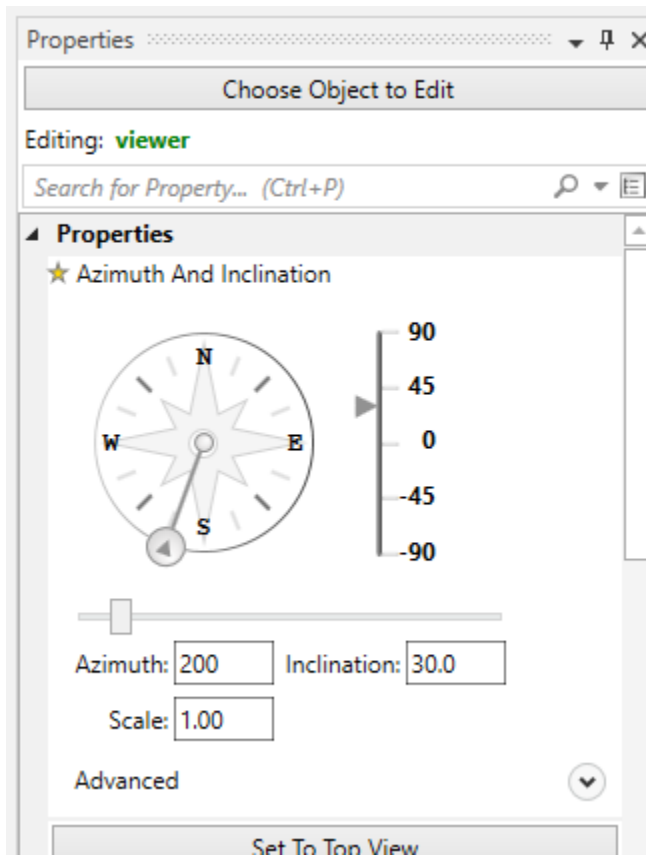
Transformations with the Azimuth and Inclination Controls

The viewer's *Properties* window gives us more precise ways to transform (scale, pan and rotate) an object: through the Azimuth and Inclination Controls.

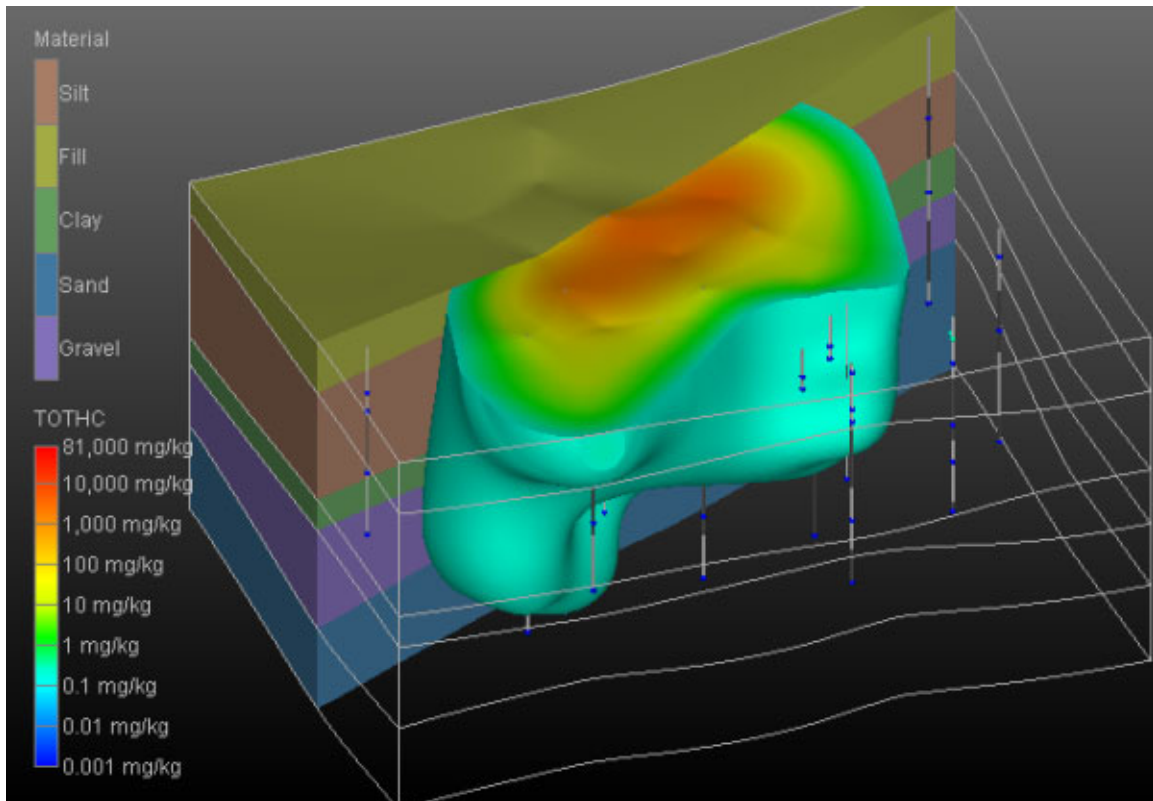
Double click on the viewer module to open the Properties window with view controls including sliders and an array of buttons. These controls allows you to instantly select a view from any azimuth and inclination. For a given (positive) inclination, selecting different azimuth buttons is equivalent to flying to different compass points on a circle at a constant elevation. The azimuth buttons are the direction from which

you view your objects. (i.e. 180 degrees views the objects **from** the south). An inclination of 90 degrees corresponds to a view from directly overhead, 0 degrees is a view from the horizontal plane (side view) and -90 degrees is a view from the bottom.

c. Use the Azimuth and Inclination Panel to obtain a specific view by setting the scale slider and inclination slider to desired settings and click once on the desired azimuth button. If you choose a scale of 1.0, an Inclination of 30 degrees and an azimuth of 200 degrees



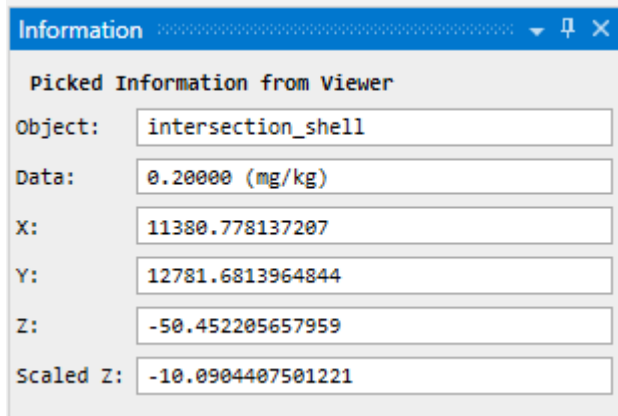
The viewer will show:



The **Advanced** options provide the ability to allow rotations about a user defined center, as opposed to the default center of the objects, which is chosen by EVS. Additionally you can apply a ROLL to the view which will make vertical objects (such as the Z axis) not appear vertical.

Below the Advanced options, there are three buttons

1. Set to Top View: Returns the model view to Azimuth 180, Inclination 90 and Scale of 1.0
2. Zoom To Fit: Returns the Scale to 1.0
3. Center On Picked: This button is normally inactive, but is activated by probing with CTRL-Left mouse on any object in the view. The default center of an object shown in our viewer is midway between the min-max of the x, y and z dimensions. This button then causes the view to recenter on the selected point. When you pick a point on an object, the following information is displayed in the *Information* window.



The **Perspective Mode** toggle switches to Perspective (vs. Orthographic) viewing. In perspective mode, parallel lines no longer appear parallel but instead would point to a vanishing point.

The **Field of View** determines the amount of perspective. Larger values result in more perspective distortion.

The Render selection allows you to choose between OpenGL and Software renderers. On some computers with minimal graphics cards Software renderer may perform better or be more stable.

Auto Fit Scene: The choices here include:

- *On Significant Change:* This is the default behavior which causes the view to recenter and rescale if the extents of the view would change significantly. Otherwise the view is unaffected.
- *On Any Change:* This causes the view to recenter and rescale if the extents of the view changes at all
- *Never:* The view will not change if objects change.

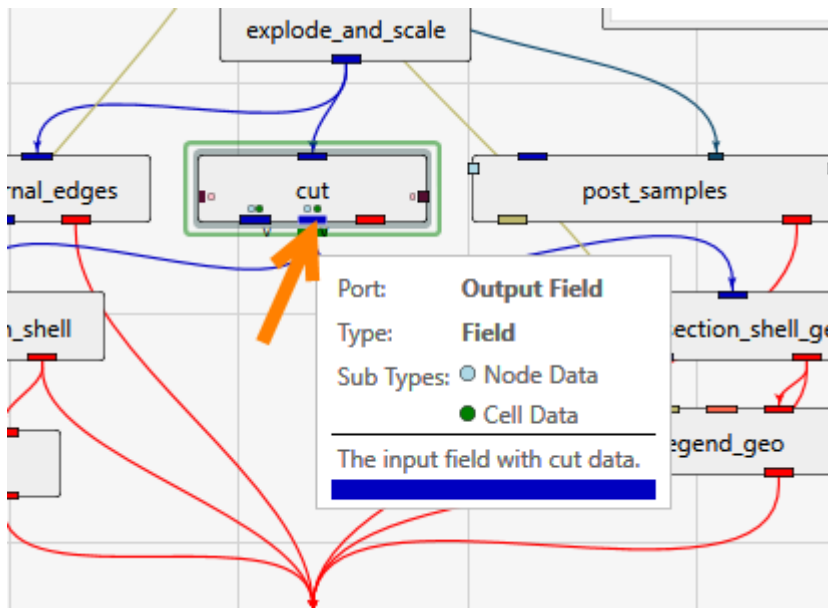
The **Window Sizing** options

- **Fit to Window:** The view size is determined by the size of the viewer window
- **Size Manually:** The view size is set in the Viewer Width and Height type-ins below to a specific size. The viewer then has scroll bars if the view size exceeds the window size.

Basic Statistics

At any time after modules have run, you can quickly obtain basic statistical and model extents data merely by double-clicking on any blue output port.

Let's demonstrate this by using the second output port of the cut module



When we double-click here, the following information appears in the Properties window.

Properties

Choose Object to Edit

Editing: cut: Output Field

Search for Property... (Ctrl+P)

Port Information

Statistics

Number Of Nodes	1598400
Number Of Cell Sets	5
Total Number Of Cells	1469769
Number Of Node Data	3
Number Of Cell Data	1

Coordinate Extents

	Min	Max
X	11,055	11,617
Y	12,680	13,111
Z	-280	57.5
Z (Scale Adj)	-55.9	11.5

Extents

X	562
Y	431
Z	337
Z (Scale Adj)	67.4

This quickly tells us that this port has a model with the following data and coordinate extents

- 1.59 million nodes
- 1.47 million cells
- 3 nodal data
- 1 cell data
- The X, Y & Z Minimum, Maximum and Extents are provided
 - The Z data is provided both for the Z Scaled model (5 times exaggerated in this case) and actual (shown highlighted in yellow)

For more comprehensive statistical analysis of the nodal data, connect the output port to the [statistics](#) module.

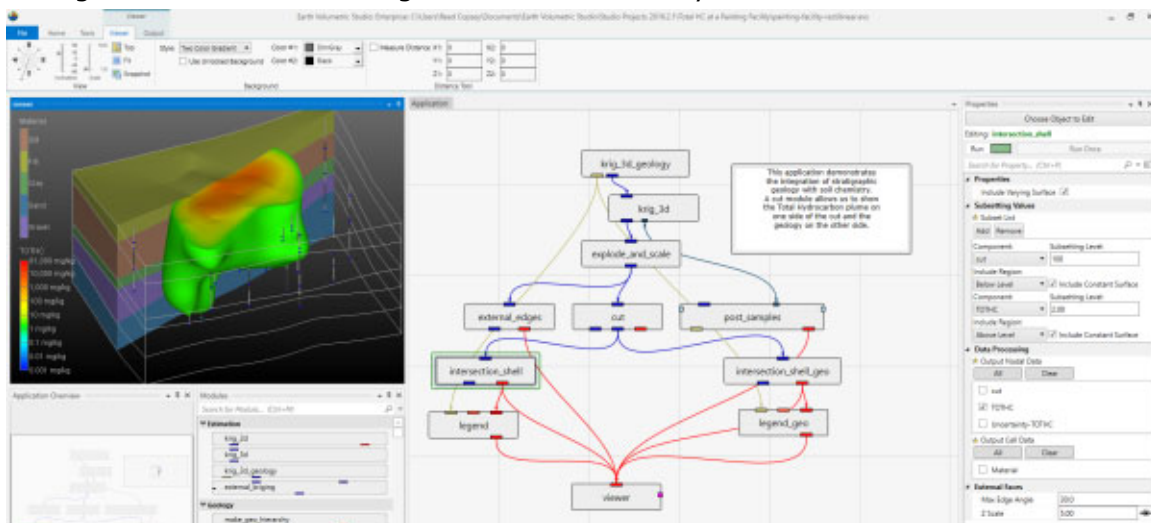
Change Plume Level

Before we end this first workbook, let's interact with this application in another way.

In the Application window, double click on the intersection_shell module, and you will see a green border around it. This green border designates the selected module whose properties are available for editing.

This will open its Properties in the Properties window in the upper right. In this application, intersection_shell is performing two tasks. It is cutting the model using information provided by the cut module and it is also subsetting what remains by Total Hydrocarbon (TOTH) level. It might seem strange at first that the cut module isn't actually cutting the model. But if it did, it would only provide one side or the other. By giving us data that is the "signed" distance from the specified cutting plane, we are able to use cut's data to create the cut for the front side giving us the plume and the back side giving us the geologic layers. We can also offset any distance from the theoretical cutting plane without actually moving the cutting plane, but only changing the "cut" Subsetting level. In fact, in this application we're cutting 100 feet from the specified cutting plane.

Change the TOTHC Subsetting Level to be 2.0 and your view should look like this:



You can continue to experiment and see that you can view any subsetting level in less than a second.

Migration from Studio-32 to Studio-64

There are a number of profound changes which are likely to impact every prior (32 bit) studio application (.evs file) you have made. All of these changes also affect your importation of EVS-Pro and MVS applications as well (though a significant number of additional issues apply in those cases). We have grouped the errors into categories, with the last one being trivial (meaning you can ignore these errors completely).

1. Estimation Method: This is the new global term for the method to be used for estimation. In previous versions of our software it has had several names including *Interpolation Type*, *Interpolation Method* and *Method Type*. Since the estimation process typically includes both interpolation and extrapolation of the data, these prior names were inappropriate and inconsistent, and all modules which perform estimation now include a selector titled Estimation Method.

This will result in one error for each of the affected modules. In the Output Log shown below, these errors have been highlighted in yellow.

The affected modules include:

- krig_3d_geology
- krig_3d
- krig_2d
- indicator_geology
- adaptive_indicator_krig

The consequence of this change is that any prior application will now open with the default Estimation Method for all affected modules, rather than the method you might have specified in the individual modules. It is important that you carefully confirm that you have the correct method selected.

2. Geology Cell Data: All geology data (Geo_Layer and Material_ID) has been moved from being Nodal Data to Cell Data. This affects all applications with stratigraphic layers and the modules:

- 3d_geology_map
- krig_3d (since it creates layers from krig_3d_geology's surfaces)
- geologic_surface
- geologic_surfaces
- post_samples
- make_geo_hierarchy
- any module that is coloring geology data from the above modules

The consequences of this change are the most pervasive, and require the most effort on your part to adjust the settings in the individual modules to carefully confirm that you have the correct outputs in your application. Custom datamaps that were previously applied to nodal data must be copied to cell data (but this is easy).

3. Kriging Data Component Enhancements: krig_3d and krig_2d have new data component selection options which allow you to simultaneously choose any or all of the statistics, min-max plume and other data (e.g. depth, elevation, layer_thickness, etc.) for output. By default all are ON. This means that you will always have more data components coming out of the kriging modules than you did before, likely in a different order.

The consequences of this change are also quite pervasive. Any module that operates on nodal data components downstream of kriging modules will likely select the wrong data components when you load the application unless they are selecting the first one.

You can resolve this problem in a combination of ways:

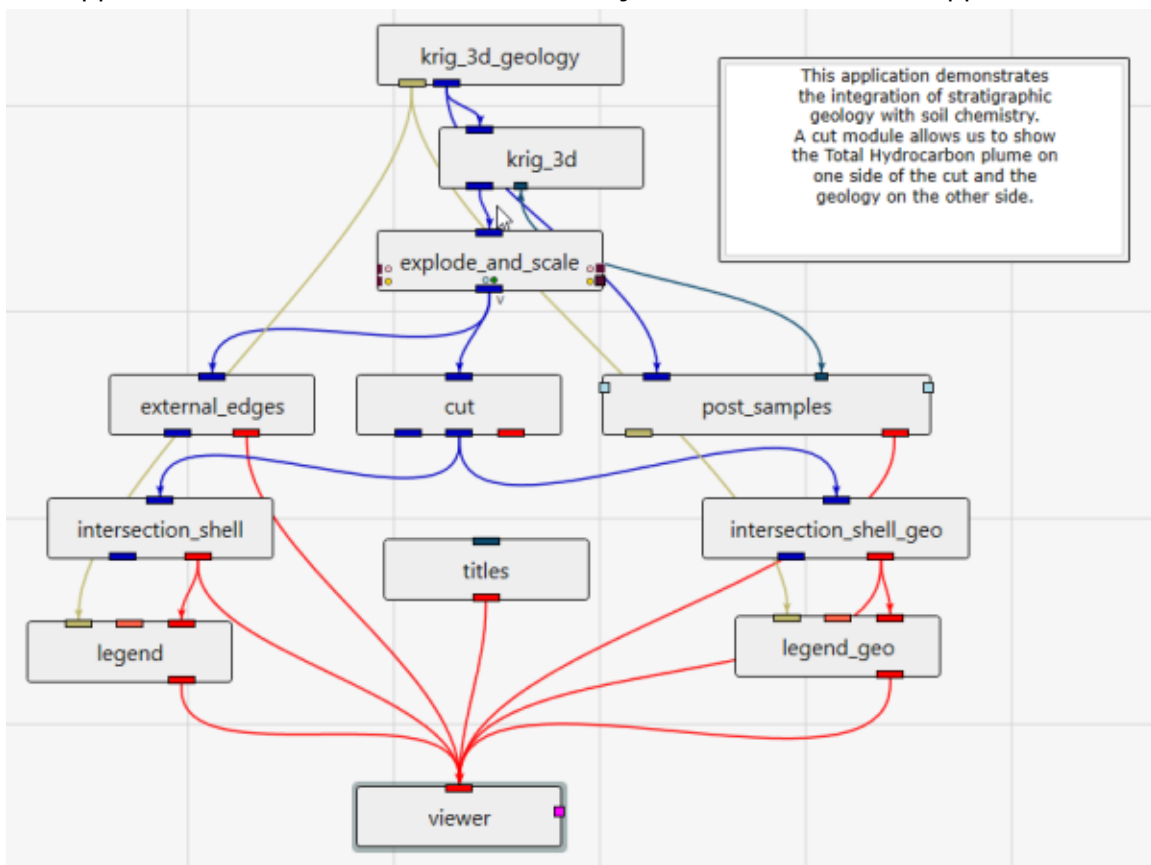
- You can choose only the data components that you need in your output. Even though you may choose the same data you had before, the order may be different, and you will not have the geologic data which is now cell data. For that reason, downstream modules will almost certainly be affected. Unless you have limited memory on your computer, you can leave all of the data selected and deal with the issue downstream.
- If the output in the viewer is not correct look to each module connected to the viewer, and potentially to those upstream to make sure the data you need is selected.

4. Cache Size: In all prior versions of our software (which were all 32-bit applications), any object which connected to the viewer had a default Cache Size in MB. If your model was large and required a larger cache of memory, it would not display and an error would alert you that you needed to increase the cache. The 64-bit application has allowed us to eliminate any cache restrictions and therefore this parameter has been eliminated. Since it is gone, it is creating an error message, but **the error is completely harmless and can be ignored**. In the Output Log shown below, these errors have been highlighted in **Orange**.

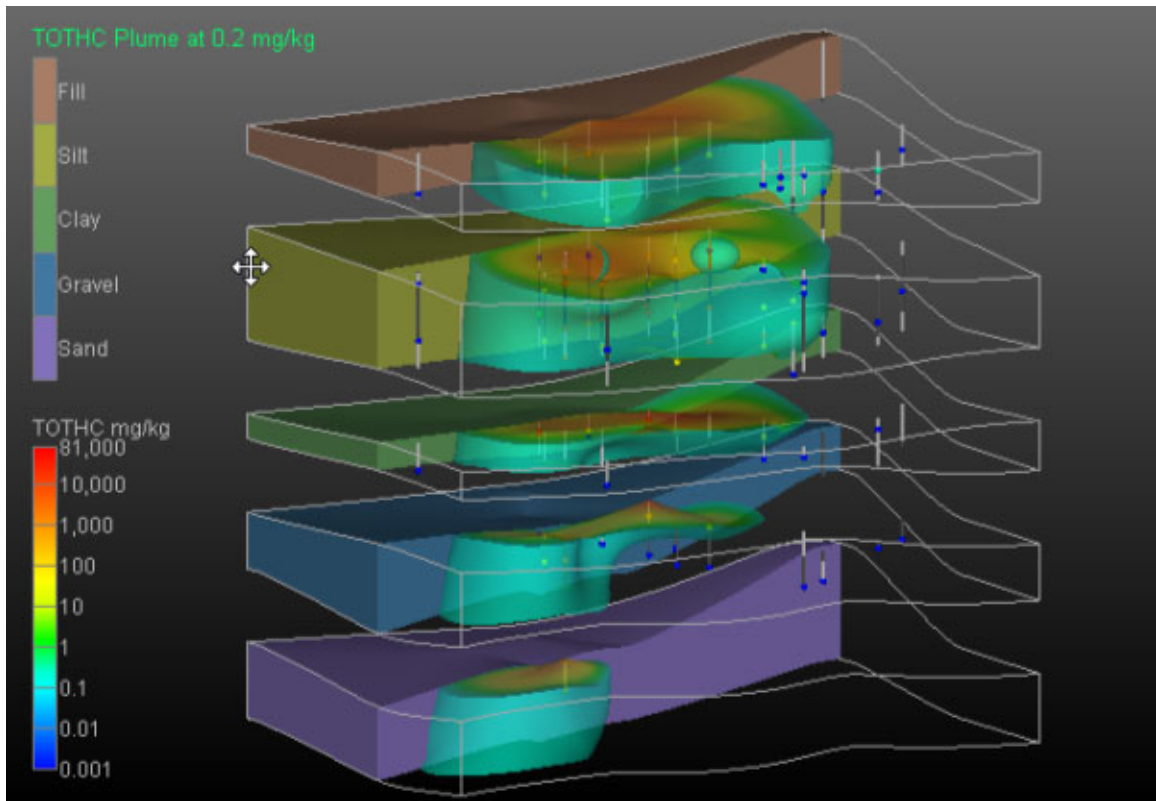
All of these issues are relatively simple to correct, but there will be a significant number of warnings and errors when you load your prior Studio-32 applications. Once you make the adjustments to your modules and re-save, the errors should be gone on the next load.

Please back-up your old applications first.

The application below is one of the Studio Projects version 2016.2.1 applications.



which created the following output in Earth Volumetric Studio version 2016.2.1

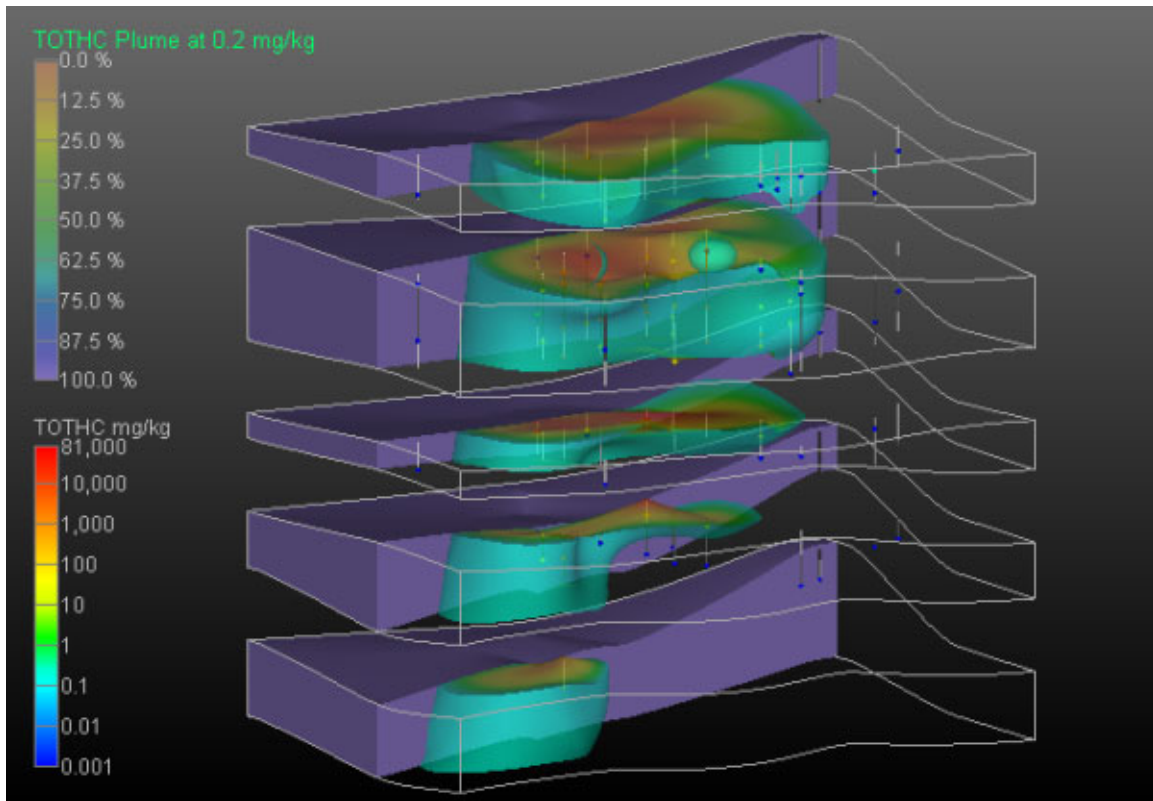


Upon loading this application in Studio-64, the output log displays the following errors and warnings:



Please notice that the bulk of the errors for this application are related to krig_3d. However, the changes to krig_3d that result in these errors do not necessarily require making changes in krig_3d, but rather require making changes in all affected modules downstream of krig_3d. And since krig_3d is near the top of our application, these changes can affect many modules.

As you can see, when we import this application, the viewer is not showing the model correctly.



The plume is correct, and the reason that it is OK is because the data component for TOTHC was and still is the first data component. But the coloring of the geologic layers on the back side of the cutting plane are now being colored by Confidence with a (leftover node data) datamap that was intended for geologic layers.

It is important that you understand what has changed in krig_3d. As stated above, one of the biggest changes that will affect your applications is the quantity of nodal data and cell data components which krig_3d now creates. In Studio-32 there were two fundamentally different kriging options which we referred to as Statistics or MinMax. These options are now gone and the default kriging output now includes the outputs of both options and more. This was not done in the 32-bit software to conserve memory, but the ability to have all options simultaneously is very powerful and as a 64-bit application, the memory limitations are effectively gone. However, you have the option to selectively include any or all of these data.

In Studio-64 there are three categories of data which are output with kriging and by default all are included. The first two categories (marked by blue arrows) are nodal data, and the third is cell data.

Data To Export

Auxillary Kriging Data ←

☒ Automatically Select Everything

Max Plume

Min Plume

Statistical Confidence

Standard Deviation

Statistical Uncertainty

Geologic Nodal Data ←

☒ Automatically Select Everything

Layer Thickness

Depth

Elevation

Geologic Cell Data ←

☒ Automatically Select Everything

Material ID

Geo Layer

The first affected module in our application is cut. Previously nodal data components of TOTH, Layer Thickness and Material_ID were selected, but as you can see below, that is no longer the case.

Properties ▾ ⏏

Choose Object to Edit

Editing: **cut**

Run ☐

Search for Property... (Ctrl+P) 🔍 ▾

Heading: Dip:

★ Reset

▲ **Data Mapping**

Nodal Data

☐ Automatically Select Everything

☒ TOTHC

☐ 90%Max:TOTHC

☐ 90%Min:TOTHC

☒ Confidence-TOTHC

☐ StdDev-TOTHC

☒ Uncertainty-TOTHC

☐ Layer Thickness

☐ Depth

☐ Elevation

Cell Data

☐ Material

☐ Layer

I've unchecked Confidence and Uncertainty and have checked Layer Thickness and Depth (don't really need it, but it can be useful). We also want to select Material from the Cell Data.

Properties

Choose Object to Edit

Editing: **cut**

Run ☐

Search for Property... (Ctrl+P)

Heading: Dip:

★ Reset

Data Mapping

Nodal Data

☐ Automatically Select Everything

☒ TOTHC

☐ 90%Max:TOTHC

☐ 90%Min:TOTHC

☐ Confidence-TOTHC

☐ StdDev-TOTHC

☐ Uncertainty-TOTHC

☒ Layer Thickness

☒ Depth

☐ Elevation

Cell Data

☒ Material

☐ Layer

Now we have to modify the module that was coloring the geologic layers. Since we want to color by Material, we need to clear all nodal data and select material. If there were pinched layers, we would need to use the Layer_Thickness for subsetting.

Properties

Choose Object to Edit

Editing: **intersection_shell_geo**

Run

Run Once

Search for Property... (Ctrl+P)

Properties

Include Varying Surface

Subsetting Values

★ Subset List

Add

Remove

Component:

cut

Subsetting Level:

100

Include Region:

Above Level

Include Constant Surface

Data Processing

★ Output Nodal Data

All

Clear

cut

TOTHC

Layer Thickness

Depth

★ Output Cell Data

All

Clear

Material

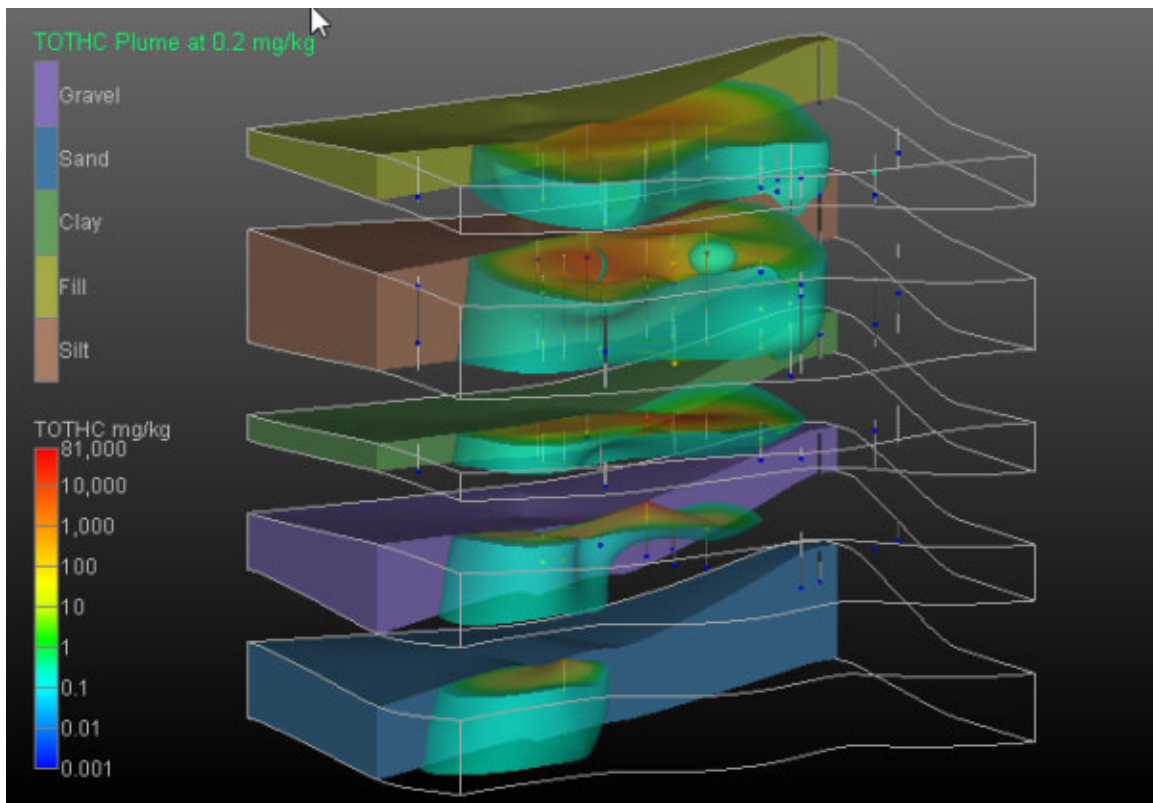
External Faces

Max Edge Angle

30.0

Z Scale

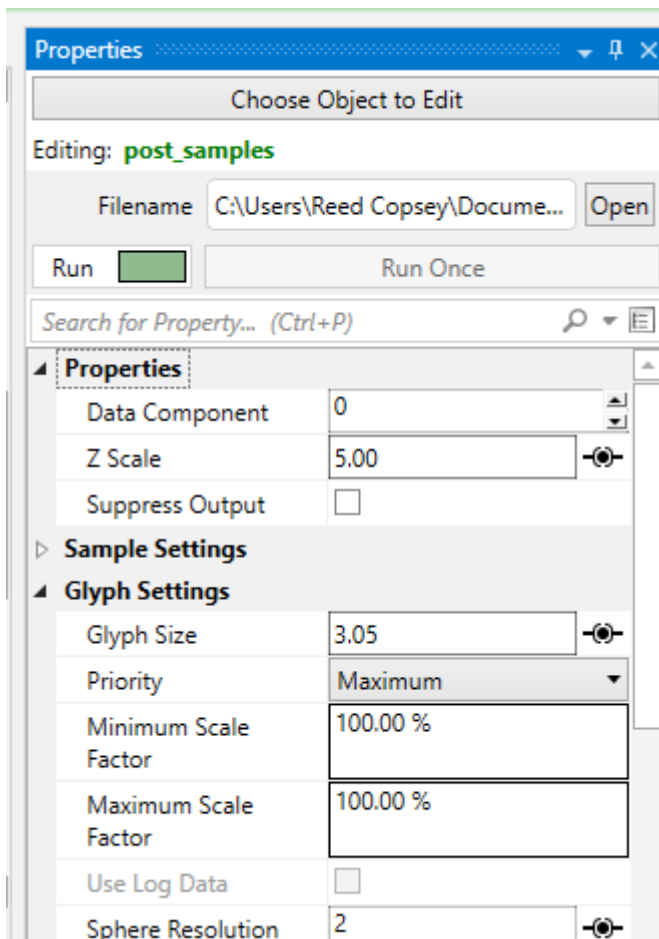
5.00



Below you can see the primary issues related to the post_samples errors. Instead of having a Radius Min and Radius Max, we now have:

- Glyph Size
- Priority
- Minimum Scale Factor
- Maximum Scale Factor

These parameters give you complete control over the sizes of spheres or glyphs and their correlation to data.

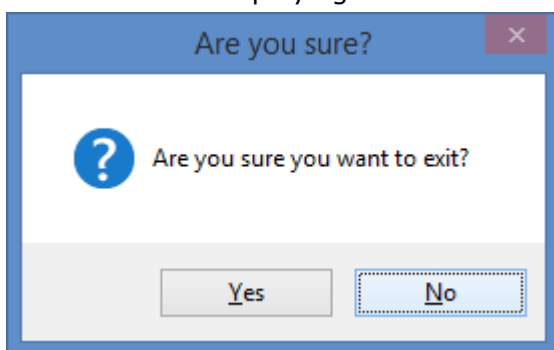


Exit EVS

Let's exit EVS.

Select the File->Exit pull-down command.

EVS exits after displaying a confirmation message.



If you close the main window using the X in the upper left, it will prompt you similarly.

You have now completed Workbook 1.

EVS Project Structure for Maximum Portability

Create a "project" folder with all of your data in one or more subfolders under that folder (any number of levels deep). As long as you don't put your applications more

than 2 levels deep inside of the project folder, everything will be relative, and moving the project folder (as a whole) will always "just work".

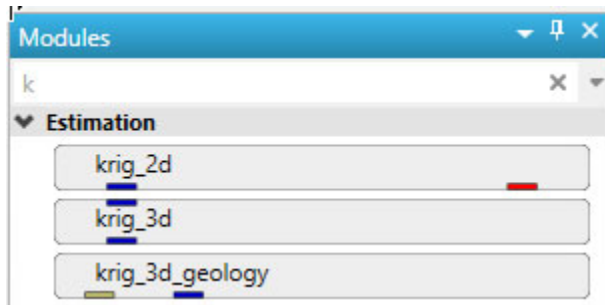
An example would be:

- drive\some\path
 - project
 - applications
 - data
 - data sub 1
 - data sub 2

Alternatively, the most portable EVS Application is one where all of the data is packaged.

Instance Modules

Now let's see just how fast we can instance the modules to create a useful application. In the Modules window, type k



This will show all modules beginning with the letter k. From this filtered list we can instance any of these modules by double-clicking on them. However, we can get the first one, krig_2d by hitting Enter. Do that.

When you hit enter, it also clears the filter (search) field.

Now type p. Double-click on plume, ~4th in the list.

Since we didn't hit enter, we need to clear the p and now type e. Double-click on external_edges, ~3rd in the list.

Finally, backspace or clear the e and type l for legend, finding it as the second module and double click on it too.

You may need to pan in the application to see our application should be:



Connect the modules

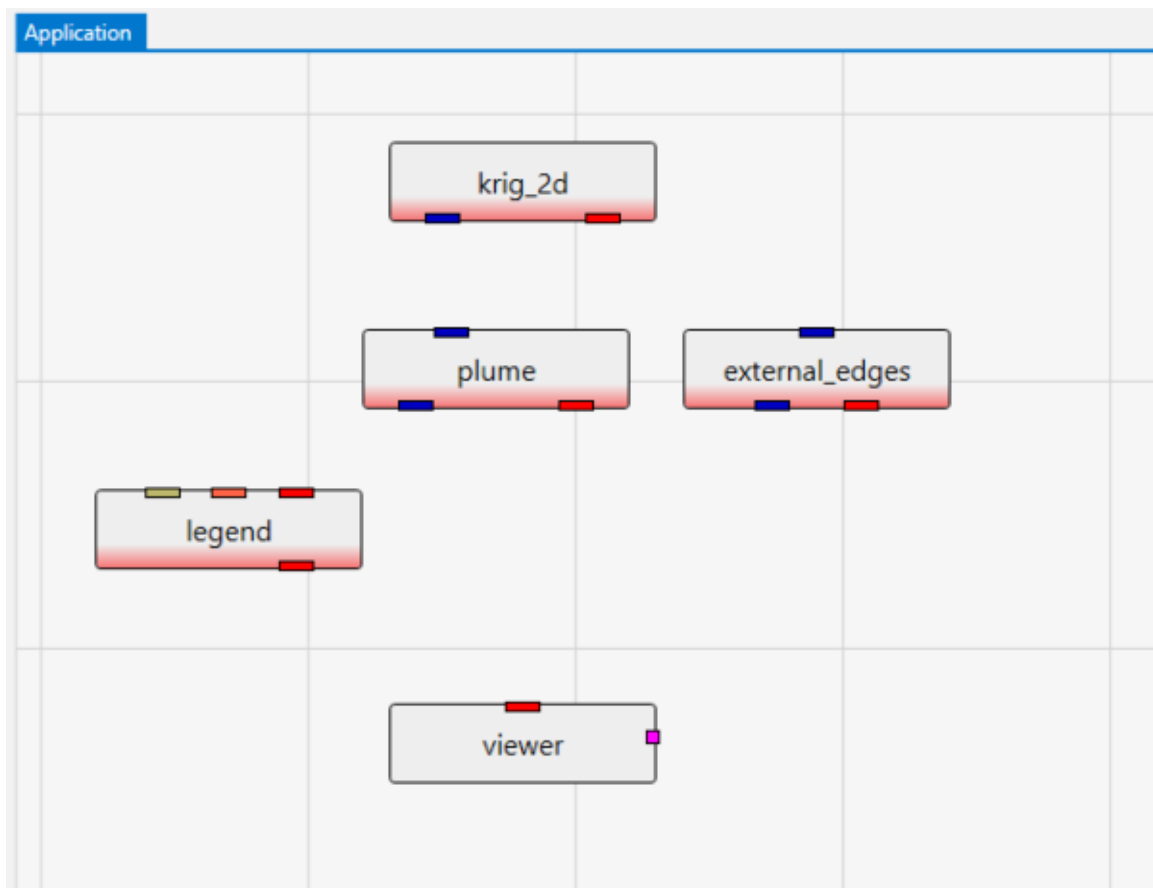
Connections

We'll now connect these modules. Connections determine how data flows or is shared among modules, and affects the modules' order of execution.

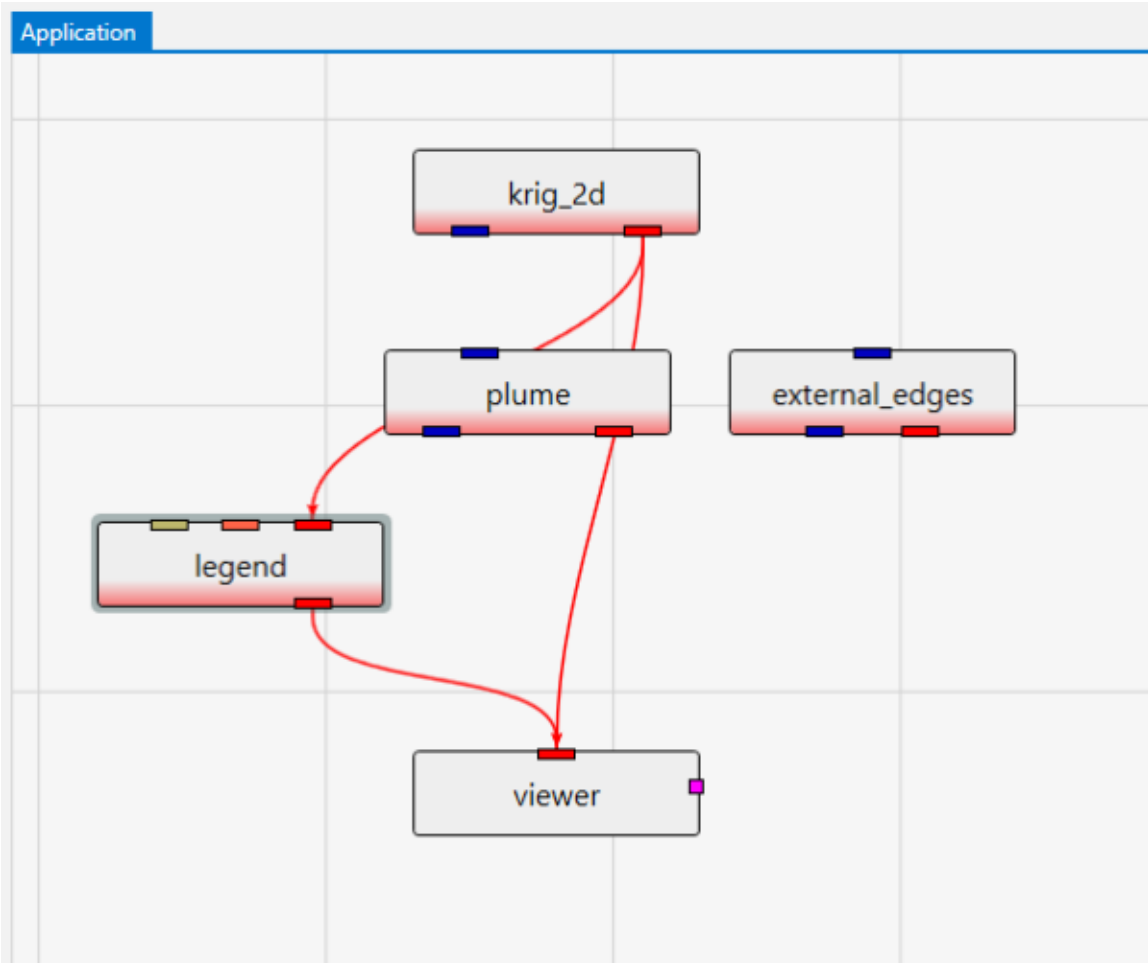
(Note: the order in which we instance and connect modules is, with the exception of certain array connections, unimportant. We could have instanced and connected these modules in any order.)

The method of connecting modules was covered in the [second workbook](#).

We could leave these modules in their current positions, but let's move them around so they better match how we want the data to flow. Adjust the positions to approximately match:



and let's just connect a few of them as shown below

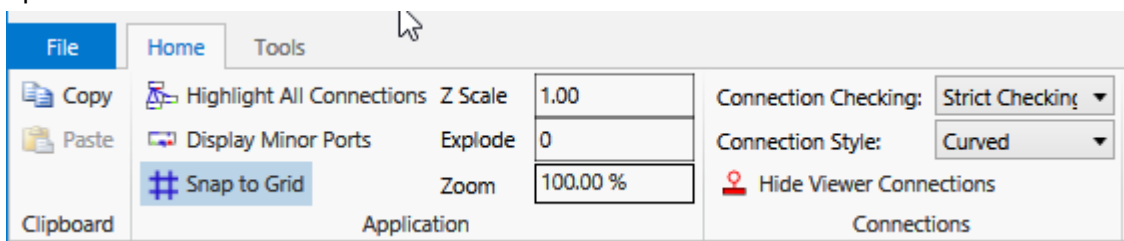


We are not connecting all of the modules at this time for two reasons:

- We want to examine the simplest 2D kriging applications first and then make it more complex.
- krig_2d will not connect to plume_area until krig_2d has been run. More on this in the next topic..[Home Tab Basic Options](#).

Home Tab Basic Options

The Home tab has several important options which allow you to customize the behavior of Earth Volumetric Studio as a user and/or for a specific project. These options are shown below.



There are many important options. Most are quite obvious, but a few may not be and they are worth understanding:

Application Options affect the construction, function and appearance of your application:

- **Highlight All Connections** when off suppresses showing potential connections to minor ports.
- **Display Minor Ports**: when on makes minor ports visible on all modules including those in the Modules window. Otherwise, the minor ports are only visible when hovering over a module.
- **Snap to Grid**: When on, modules will align with a grid. This is a recommended setting, but when off, modules can be positioned anywhere.
- **Z Scale**: This is the global (application wide) vertical exaggeration. This controls the Z Scale in all modules with this parameter (e.g. explode_and_scale, post_samples, etc.) Each module inherits this value by default, but can be set with its own unique value if desired.
- **Explode**: This is the global (application wide) layer explode factor. This controls the Explode parameter in all modules with this parameter (e.g. explode_and_scale, post_samples, etc.) Each module inherits this value by default, but can be set with its own unique value if desired.
- **Zoom**: When the Viewer renormalizes, this is the default scale applied to all objects.

Connections Options affect how and when connections can be made and the appearance of connections between modules.

- **Connection Checking**: In general we will not allow you to make completely inappropriate connections. Port connections are restricted to be ports of similar type (color). However, many output ports have content that is dependent on the type of data passed to the module and/or the options selected in the module. For example, many modules can work with lines, points and volumes. Others can only process volumetric data. Prior to a module's running, we cannot know the content of its output, and therefore we cannot know precisely which ports should be connected, and which ones should not.
 - **Strict Checking**: will not allow any connections that are not definitely appropriate. This means that many connections cannot be made until a module is run.
 - **Basic Checking**: relaxes the rules of strict checking, but still prevents most inappropriate actions.
 - **Allow Unsafe Connections**: allows you to make any connections that can potentially be appropriate. This option should only be used by experienced users who know in advance what connections are appropriate.
- **Connection Style**:
 - Curved connections are much less likely to be ambiguous.
 - Straight Connections give your application the legacy appearance of EVS Pro and MVS.
- **Hide Viewer Connections** is useful to keep applications simpler looking. The red connections are suppressed but modules which are connected are obvious.

The Property Display Options control the level of detail and display of module properties

- Display Expert Properties should be off for novice users as it reduces the choices and helps prevent changing parameters that should normally not be changed.
- Always Show Critical Properties is recommended to be on since it will always show those properties which we feel must always be considered and adjusted for all applications.
- Automatically Collapse Properties is a personal preference. When on, each module's properties groups are collapsed and easy to see. When off, all properties are displayed, but for complex modules you must scroll to see them.

The Module Library Options control the level of detail and display of modules

- Include Expert Modules should be off for novice users as it reduces the choices
- Include Deprecated Modules should be off for novice users as it reduces the choices.
- Automatically Collapse Module Groups is a personal preference. When on, each module group is collapsed and easy to see. When off, all modules are displayed, but you must scroll to see them.

Gridding Defaults: These options affect many modules such as krig_3d_geology, krig_2d, krig_3d, etc.

Display Defaults:

- These options affect many modules such as explode_and_scale, geologic_surfaces, post_samples, axes, etc.
 - Z Scale
 - Explode

Estimation Defaults: These options affect modules such as krig_2d, krig_3d, etc.

- *Pre clip min* and *Post clip min* affect how non-detects are used in kriging
- *H/V Anisotropy* is an important parameter in *krig_3d*
- *Use all samples if # samples below: This toggle and the upper limit to "use all"* affects how kriging is performed
- *Number of points: affects how kriging is performed*
- *Log Process*: is an important option which affects krig_2d and krig_3d. For data which spans several orders of magnitude like contaminant concentrations, log processing the data before kriging dramatically improves the quality of the results. It is not appropriate for many other analytes such as porosity.
- See the [Geostatistics Workbook](#) for more information on the options related to Confidence.

Options Application

Option Presets:

Set to Beginner Mode Set to Intermediate Mode Set to Expert Mode

Application Display Options

- ☐ Use Straight Connections
- ☐ Hide Viewer Connections
- ☐ Always Display Minor Ports

Connection Options

- ☒ Prevent Inappropriate Connections
- ☐ Highlight Minor Ports

Property Display Options

- ☐ Display Expert Properties
- ☐ Always Show Critical Properties
- ☐ Automatically Collapse Properties

Module Library Options

- ☒ Include Expert Modules
- ☐ Include Deprecated Modules
- ☒ Automatically Collapse Module Groups

Gridding Defaults

Grid X Resolution: 81

Grid Y Resolution: 81

Grid Z Resolution: 70

Rect Offset: 0.1000

- ☒ Use Convex Hull

Display Defaults

Z Scale: 5.000

Explode: 0.000

Renderer: OpenGL

- ☒ Autonormalize

Background Style: Two Color Gradient

- ☐ Use Simple Background

Background Color 1: DimGray

Background Color 2: Black

Window Layout

Reed Center Viewer

+ -

Load Selected Layout

Save Current Layout

Revert to Default

User

Author: Reed Copsey

Organization: C Tech Development Corporation

Custom Paths

+ -

Custom Script Load Paths

+ -

Estimation Defaults

Pre clip min: 0.0001

Post clip min: 0.0010

HV Anisotropy: 10.00

- ☒ Use all data if # samples below: 1000

Number of points: 20

- ☒ Log Process

Confidence Bound: 10.00

Confidence Max for Plume: 80.00

Run the Application

Let's execute the analysis module, krig_2d, in order to produce a model based on the data file we will select. You will need to have installed the Studio Projects specific to the version of Studio you have installed.

- First, double-left-click on krig_2d to open krig_2d's properties so you can see the window below
- Click on the Open button to the right of Filename and browse to Studio Projects/Railyard Facility Complex Python Scripting and choose railyard.apdv
- Then click "Execute".

Properties

Choose Object to Edit

Editing: **krig_2d**

Filename:

Search for Property... (Ctrl+P)

Properties

Data Component:

Data Priority:

Grid Settings

Grid Type:

Grid Coordinates

	Min	Max
X	<input type="text" value="630,430"/>	<input type="text" value="630,950"/>
Y	<input type="text" value="4,271,900"/>	<input type="text" value="4,272,234"/>

X Resolution:

Y Resolution:

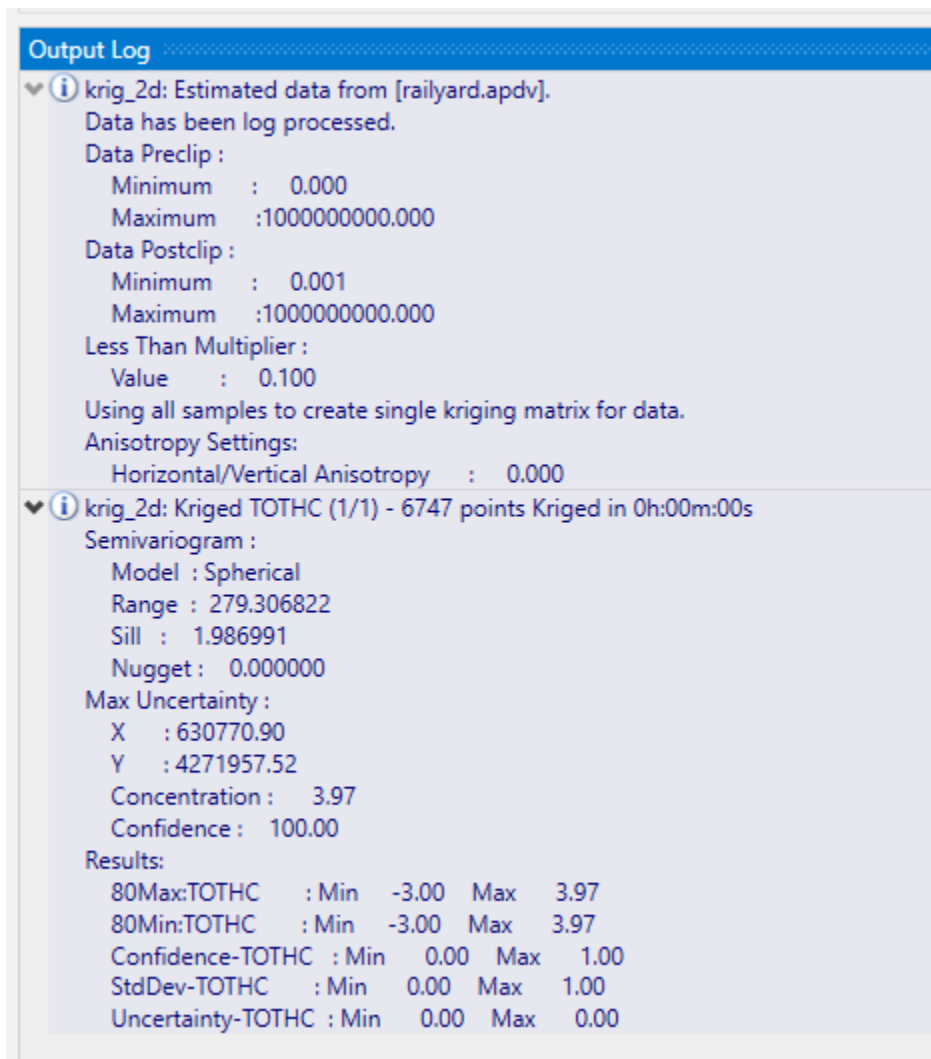
Adaptive Gridding: ☒

Boundary Offset:

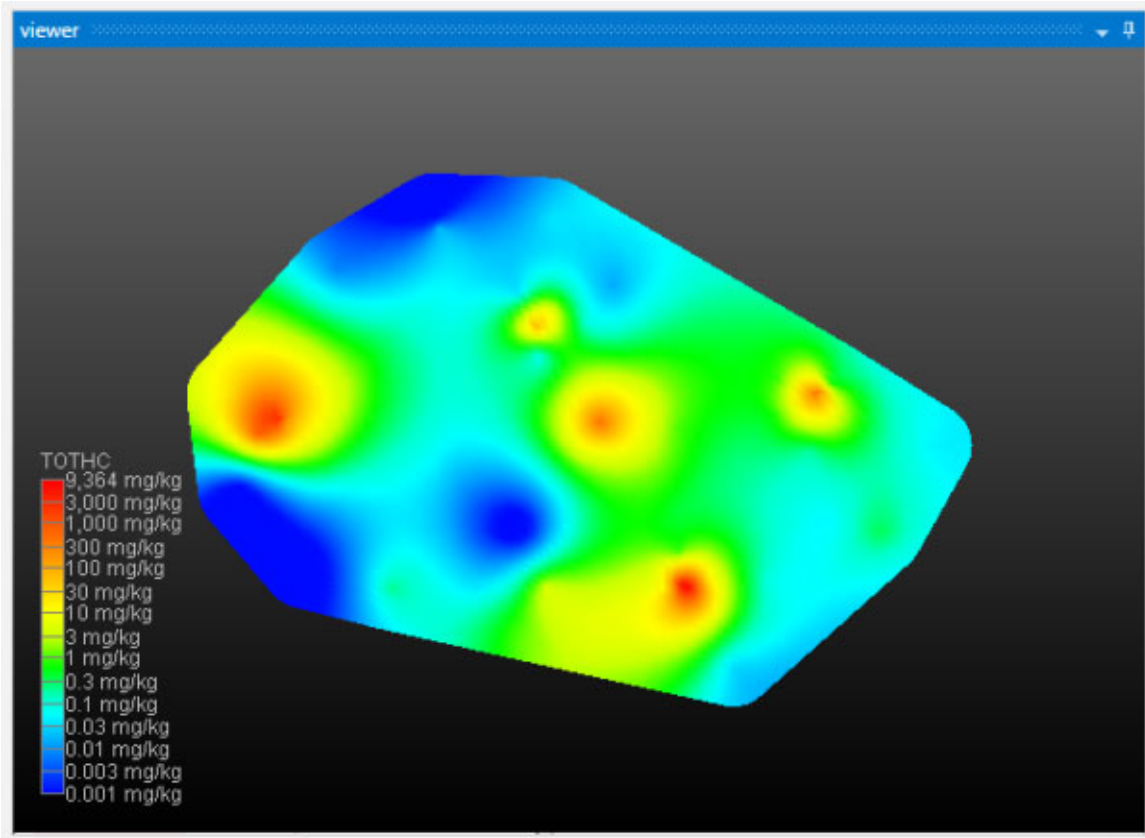
Data Processing

krig_2d reads the analyte (e.g. chemistry) data and begins the kriging process. In a very short time, it calculates the estimated concentrations for the grid we selected. While it runs, krig_2d prints messages to the Information Window such as percentage completion.

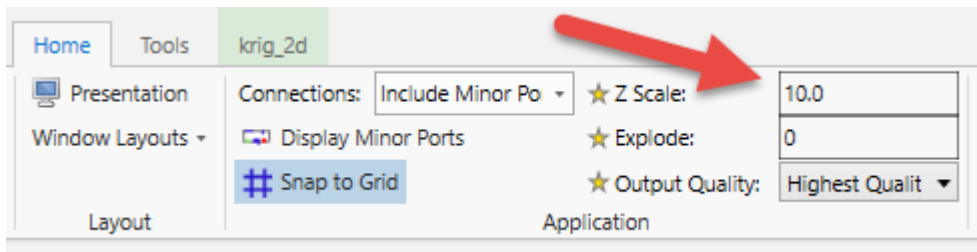
When it is done, the Output Log will show two lines, which when expanded will display:



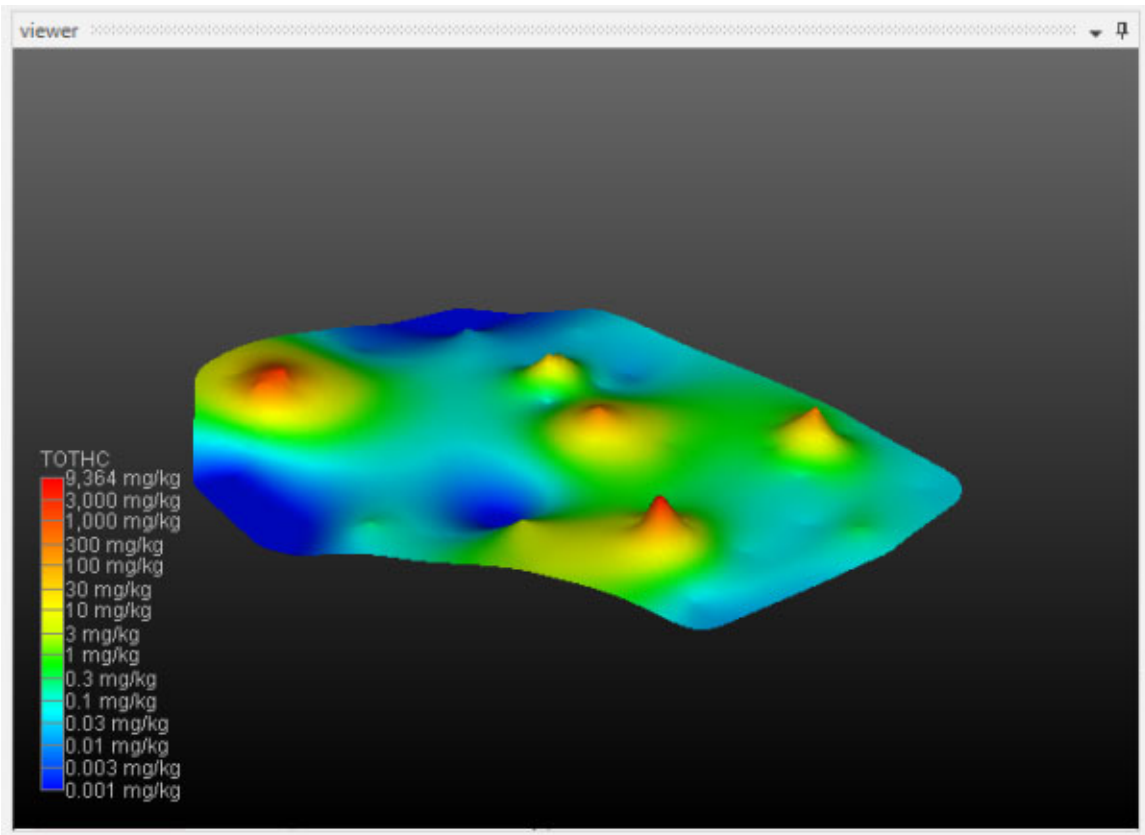
The viewer will promptly display a top view of the surface we have estimated. Your viewer should look like this:



On the Home tab, please change the Z Scale to 10. This will create an artificial topography to our surface where elevations correlate to concentration.



With a simple rotation of our model we now have

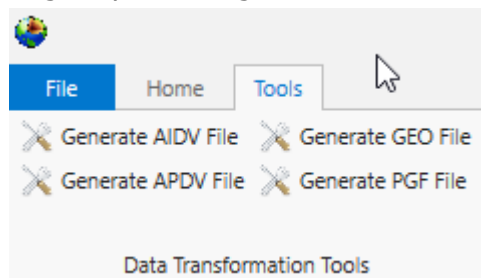


Exporting from Excel to C Tech File Formats

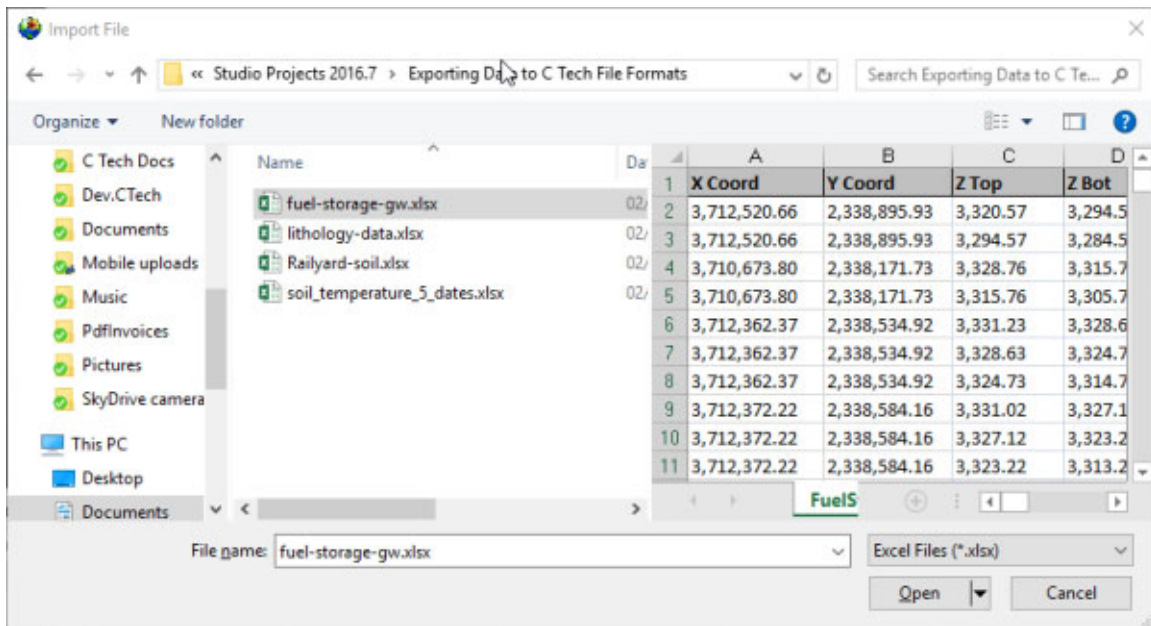
- [Creating PGF Files - Lithology](#)
- [Creating GEO Files - Stratigraphic Horizons from Vertical Borings](#)
- [Creating APDV Files - Analyte Data Measured at Points](#)
- [Creating AIDV Files - Analyte Data Measured over Intervals](#)

Creating AIDV Files - Analyte Data Measured over Intervals

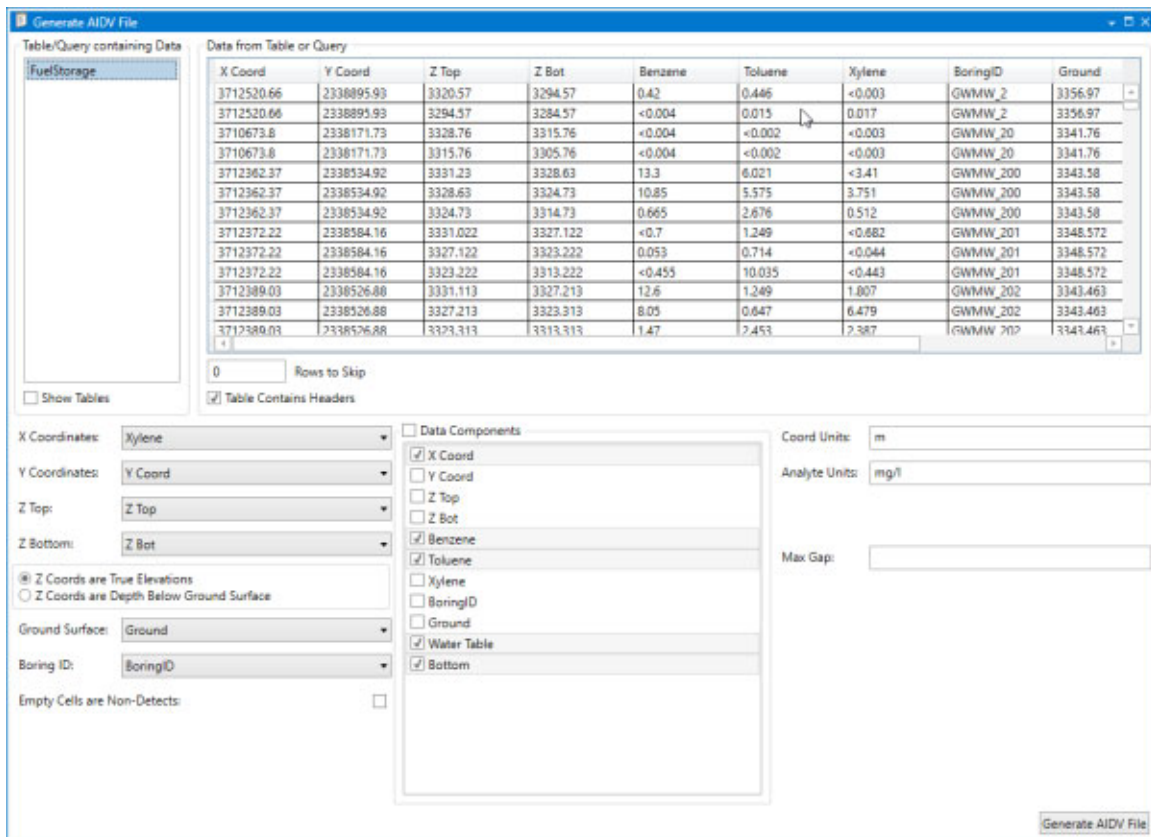
Begin by selecting the Tools tab, and select Generate AIDV File.



Let's browse to the folder shown and select the file fuel-storage-gw.xlsx



You'll need to select the appropriate table in the file. Some may have several, this one has only one named FuelStorage. Once you do, the program will attempt to automatically choose settings for you, but as you can see below, it isn't perfect.



Since Xylene starts with the letter "X", it was chosen as the X Coordinate. This is clearly wrong, however, everything else along the left side is correct. By default, the Data Components list will select whatever is left over. Sometimes this is handy, but often, and in this case it is excessive. This excel table includes water table and

bottom of model elevations that we will want for a .GEO file, but not for the AIDV file. So we'll need to make quite a few changes.

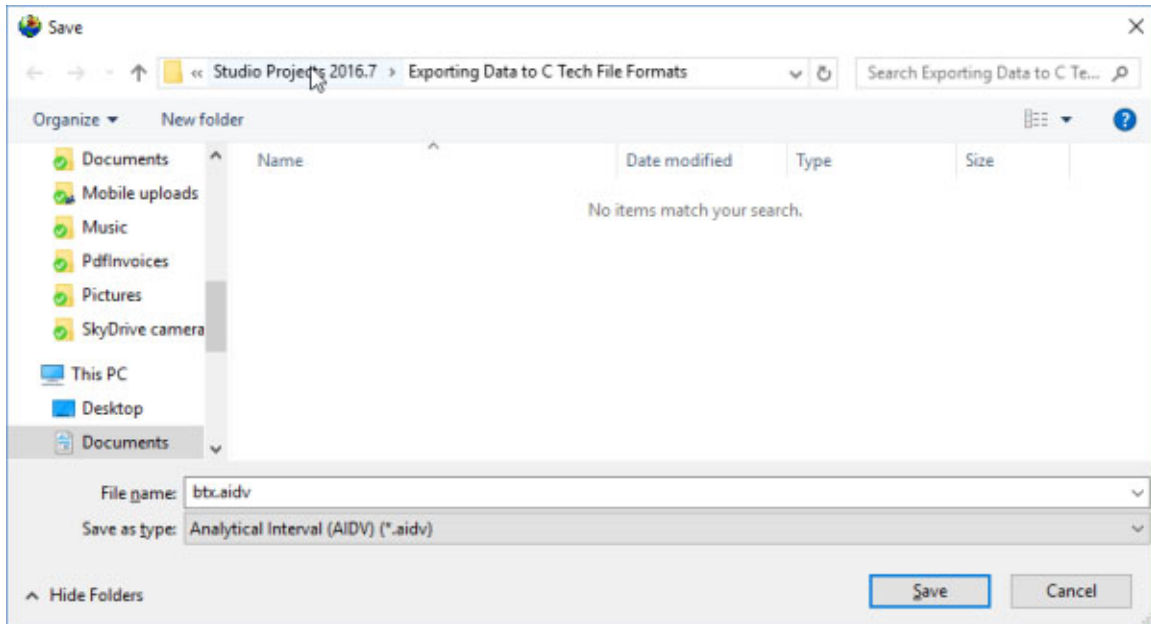
It also can't know the correct units for your analytes nor your coordinate units. It is your responsibility to make sure these are correct or change them.

The last thing you MUST do is determine and choose a Max Gap parameter. This parameter takes some understanding to properly determine. I've looked at this excel file in detail and the screen intervals vary from 0.26 to 35.1 meters in length. The Max Gap parameter is the longest length we will allow to be converted into a single point when we convert intervals to points for kriging. I would recommend setting it to 5 for this data file. That means that any interval less than 5 meters will be represented by a single point at the center of the interval. Intervals longer than 5 meters will be represented by two or more points. Choosing a value too small will create oversampling along the Z direction and too large can result in plumes which become disconnected in Z. Fortunately there tends to be a large range of reasonable values. For this dataset, I expect that good results can be obtained with values ranging from 1 to 12.

X Coord	Y Coord	Z Top	Z Bot	Benzene	Toluene	Xylene	BoringID	Ground
3712520.66	2338895.93	3320.57	3294.57	0.42	0.446	<0.003	GWMW_2	3356.97
3712520.66	2338895.93	3294.57	3284.57	<0.004	0.015	0.017	GWMW_2	3356.97
3710673.8	2338171.73	3328.76	3315.76	<0.004	<0.002	<0.003	GWMW_20	3341.76
3710673.8	2338171.73	3315.76	3305.76	<0.004	<0.002	<0.003	GWMW_20	3341.76
3712362.37	2338534.92	3331.23	3328.63	13.3	6.021	<3.41	GWMW_200	3343.58
3712362.37	2338534.92	3328.63	3324.73	10.85	5.575	3.751	GWMW_200	3343.58
3712362.37	2338534.92	3324.73	3314.73	0.665	2.676	0.512	GWMW_200	3343.58
3712372.22	2338584.16	3331.022	3327.122	<0.7	1.249	<0.682	GWMW_201	3348.572
3712372.22	2338584.16	3327.122	3323.222	0.053	0.714	<0.044	GWMW_201	3348.572
3712372.22	2338584.16	3323.222	3313.222	<0.455	10.035	<0.443	GWMW_201	3348.572
3712389.03	2338526.88	3331.113	3327.213	12.6	1.249	1.807	GWMW_202	3343.463
3712389.03	2338526.88	3327.213	3323.313	8.05	0.647	6.479	GWMW_202	3343.463
3712389.03	2338526.88	3323.313	3313.313	1.47	2.453	2.387	GWMW_202	3343.463

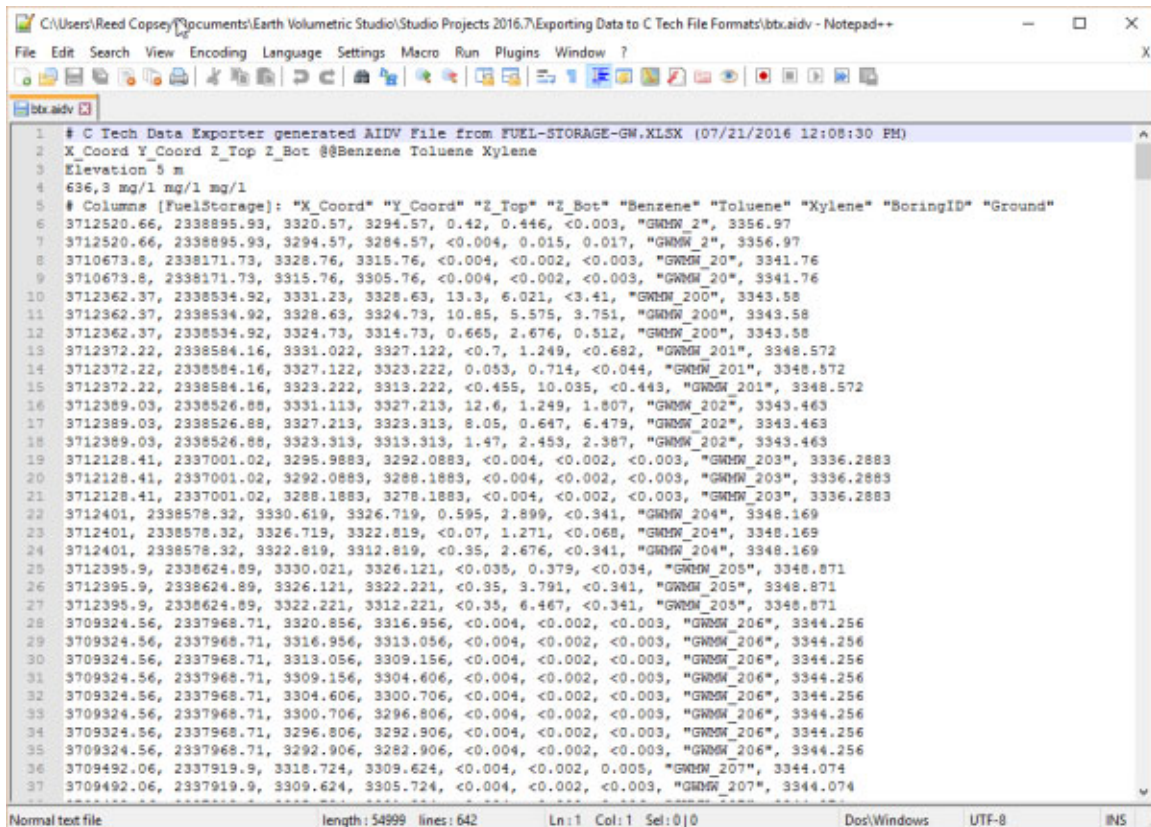
With all of our settings correct as shown above, all we need to do is click the Generate AIDV File button, and let's call the file btx.aidv.

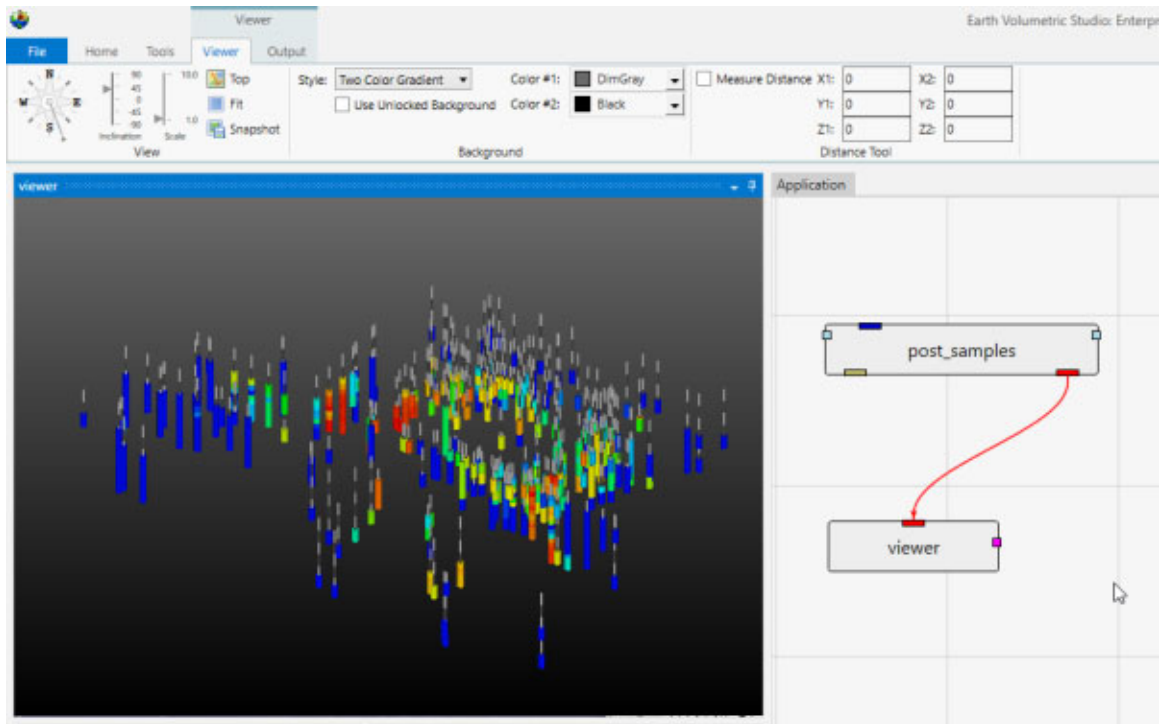
NOTE: I also want to point out the option "Empty Cells are Non-Detects". In general this toggle should be off. Normally empty cells are interpreted as being Not-Measured. It is rare that an empty cell should be a [non-detect](#), which also means that you have no information about detection limits.



Our last two tasks will be to take a look at the file in a text editor and confirm that it works in Earth Volumetric Studio.

Although Windows comes with Notepad, it is really a very poor text editor since it lacks line numbers, column numbers, and the ability to handle large files. There are many freeware text editors, but the one we like is Notepad++.

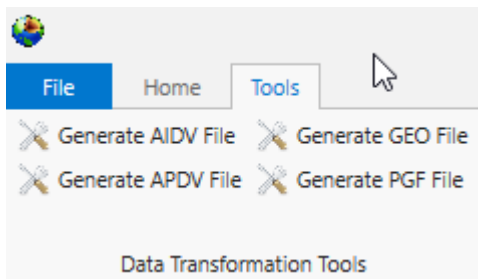




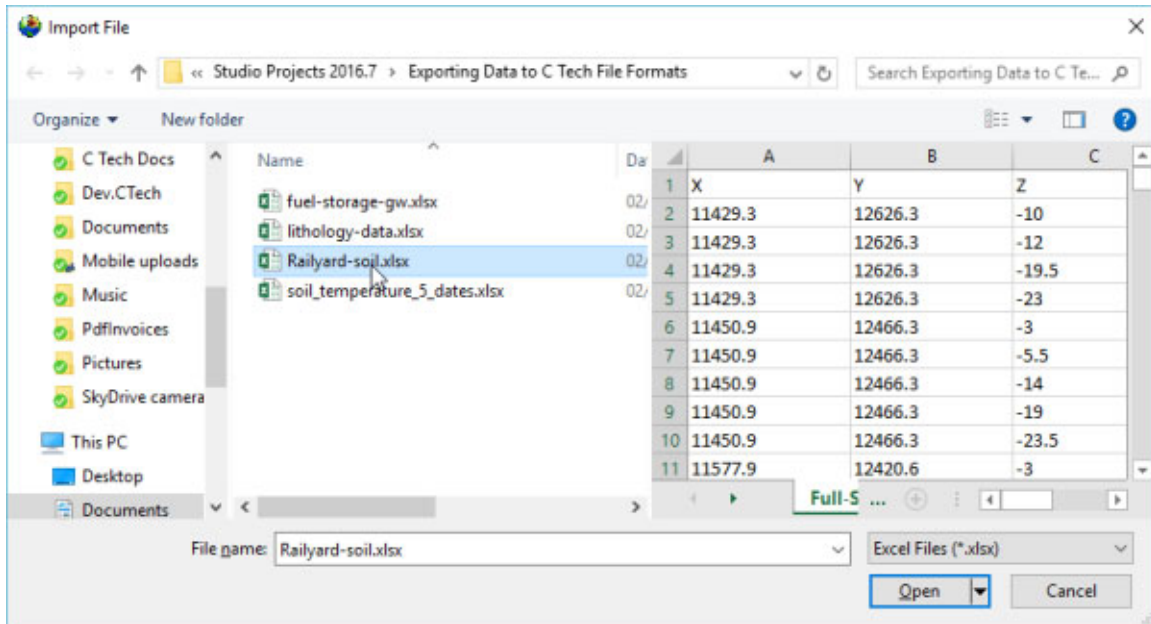
Creating APDV Files - Analyte Data Measured at Points

Begin by selecting the Tools tab, and select Generate APDV File.

Note: this topic builds upon [Creating AIDV Files](#) and assumes that you have completed that topic.

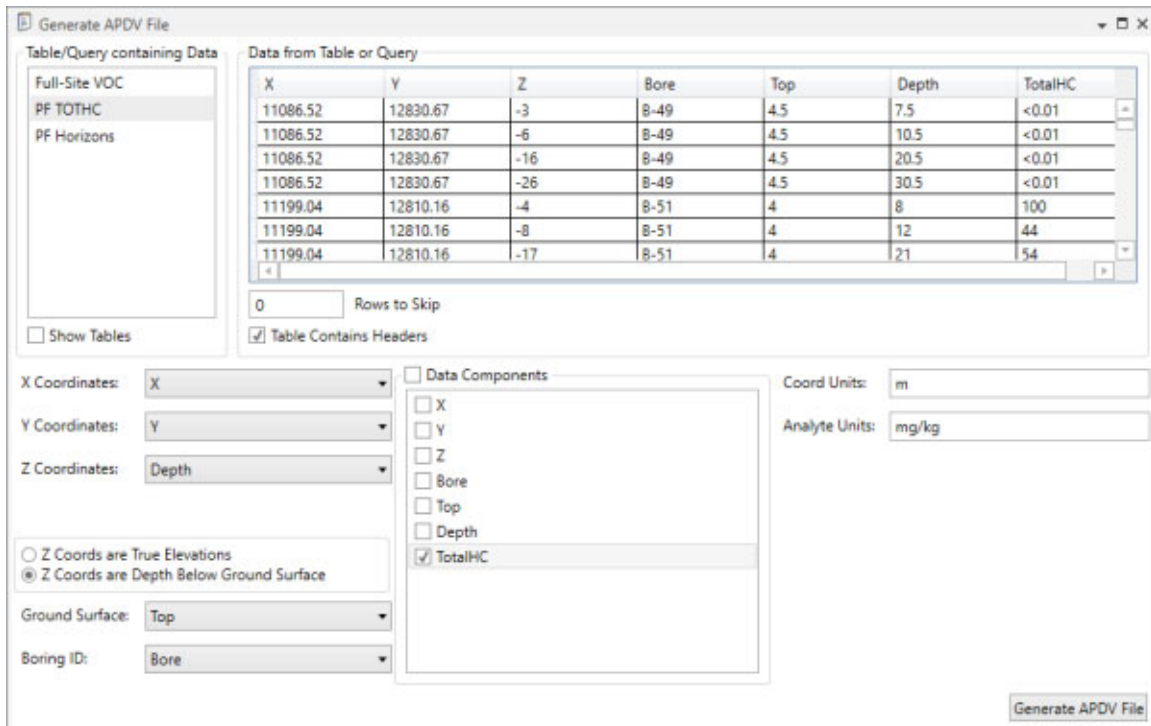


Let's browse to the folder shown and select the file Railyard-soil.xlsx



This file has three sheets and for this example, we'll choose the second one. This particular sheet has Z coordinates represented as both true Elevation and Depth below ground surface. Both are commonly used and it is not uncommon to see both in a database as a convenience for people working with the data. Our exporter can use either one and there is no technical advantage of one over the other. However, the data file created will retain the Z coordinate option selected.

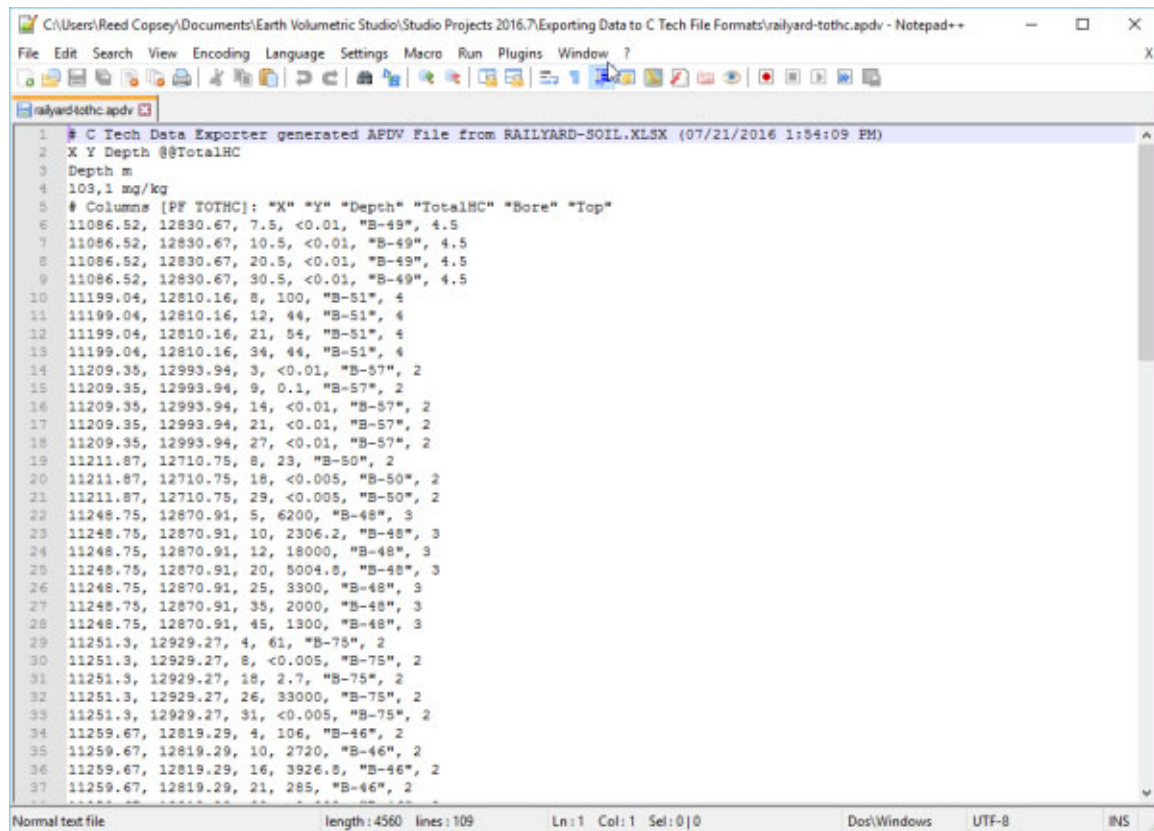
Since we used True Elevations for AIDV files, let's work with Depths this time. The correct settings are:



Please note that Top, which is our Ground Surface must be in true elevation since it is the reference surface used to define depths. Depths are always positive numbers with greater depth corresponding to lower elevations.

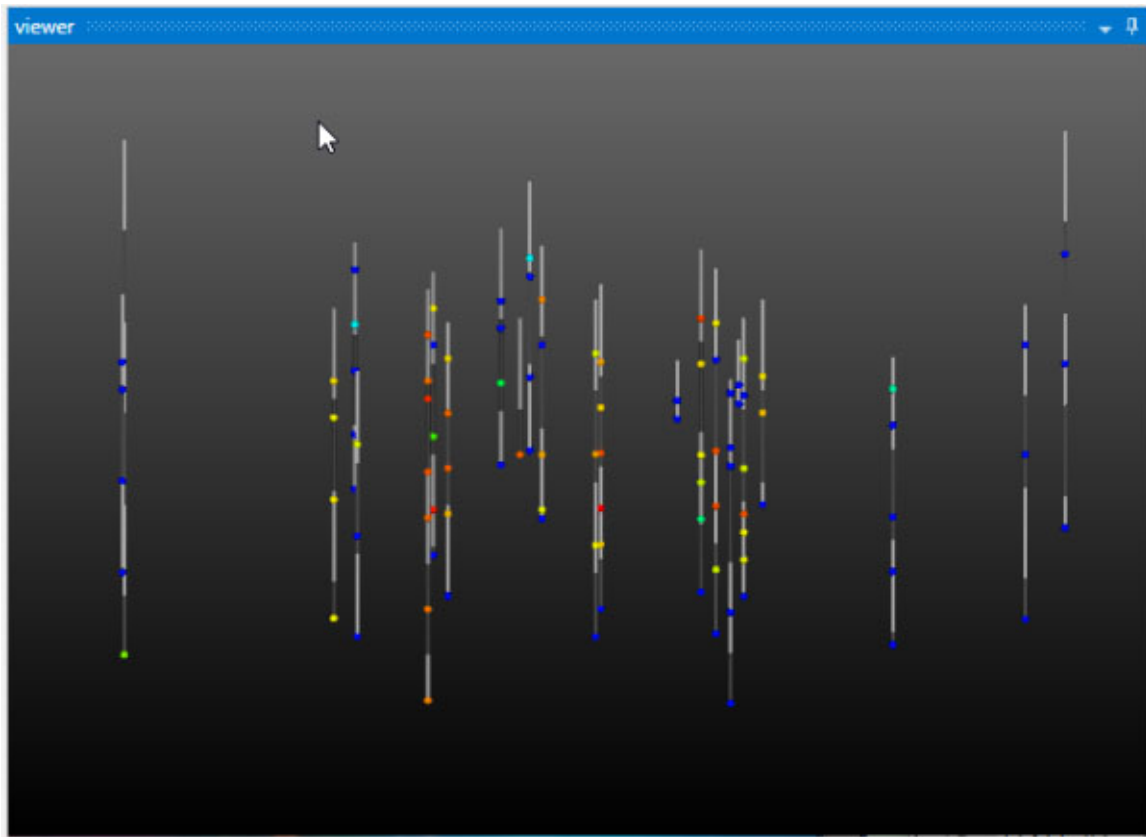
With all of our settings correct as shown above, all we need to do is click the Generate APDV File button, and let's call the file railyard-tothc.apdv.

In notepad++ our file looks like:



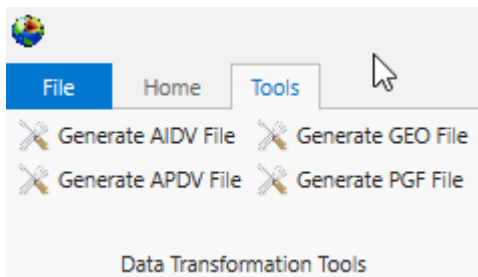
```
1 # C Tech Data Exporter generated APDV File from RAILYARD-SOIL.XLSX (07/21/2016 1:54:09 PM)
2 X Y Depth @@TotalHC
3 Depth m
4 103.1 sq/kg
5 # Columns [PF TOTHC]: "X" "Y" "Depth" "TotalHC" "Bore" "Top"
6 11086.52, 12830.67, 7.5, <0.01, "B-49", 4.5
7 11086.52, 12830.67, 10.5, <0.01, "B-49", 4.5
8 11086.52, 12830.67, 20.5, <0.01, "B-49", 4.5
9 11086.52, 12830.67, 30.5, <0.01, "B-49", 4.5
10 11199.04, 12810.16, 8, 100, "B-51", 4
11 11199.04, 12810.16, 12, 44, "B-51", 4
12 11199.04, 12810.16, 21, 54, "B-51", 4
13 11199.04, 12810.16, 34, 44, "B-51", 4
14 11209.35, 12993.94, 3, <0.01, "B-57", 2
15 11209.35, 12993.94, 9, 0.1, "B-57", 2
16 11209.35, 12993.94, 14, <0.01, "B-57", 2
17 11209.35, 12993.94, 21, <0.01, "B-57", 2
18 11209.35, 12993.94, 27, <0.01, "B-57", 2
19 11211.87, 12710.75, 8, 23, "B-50", 2
20 11211.87, 12710.75, 18, <0.005, "B-50", 2
21 11211.87, 12710.75, 29, <0.005, "B-50", 2
22 11248.75, 12870.91, 5, 6200, "B-48", 3
23 11248.75, 12870.91, 10, 2306.2, "B-48", 3
24 11248.75, 12870.91, 12, 18000, "B-48", 3
25 11248.75, 12870.91, 20, 5004.8, "B-48", 3
26 11248.75, 12870.91, 25, 3300, "B-48", 3
27 11248.75, 12870.91, 35, 2000, "B-48", 3
28 11248.75, 12870.91, 45, 1300, "B-48", 3
29 11251.3, 12929.27, 4, 61, "B-75", 2
30 11251.3, 12929.27, 8, <0.005, "B-75", 2
31 11251.3, 12929.27, 18, 2.7, "B-75", 2
32 11251.3, 12929.27, 26, 33000, "B-75", 2
33 11251.3, 12929.27, 31, <0.005, "B-75", 2
34 11259.67, 12819.29, 4, 106, "B-46", 2
35 11259.67, 12819.29, 10, 2720, "B-46", 2
36 11259.67, 12819.29, 16, 3926.8, "B-46", 2
37 11259.67, 12819.29, 21, 285, "B-46", 2
```

and if we look at this file in Studio with Z-Scale of 5 it is:

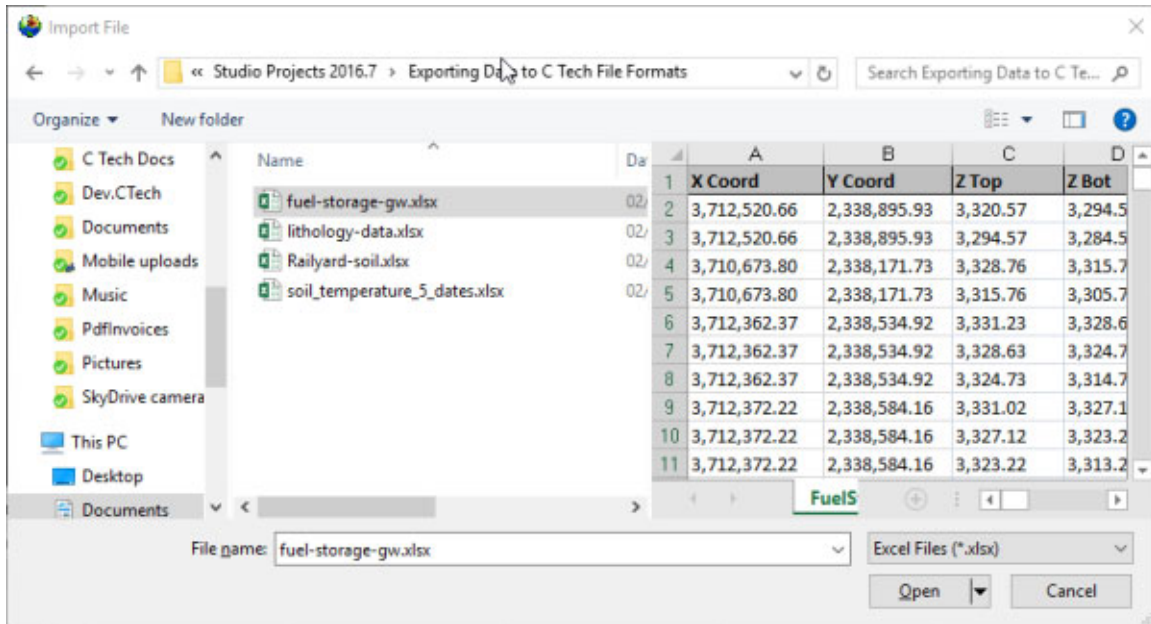


Creating GEO Files - Stratigraphic Horizons from Vertical Borings

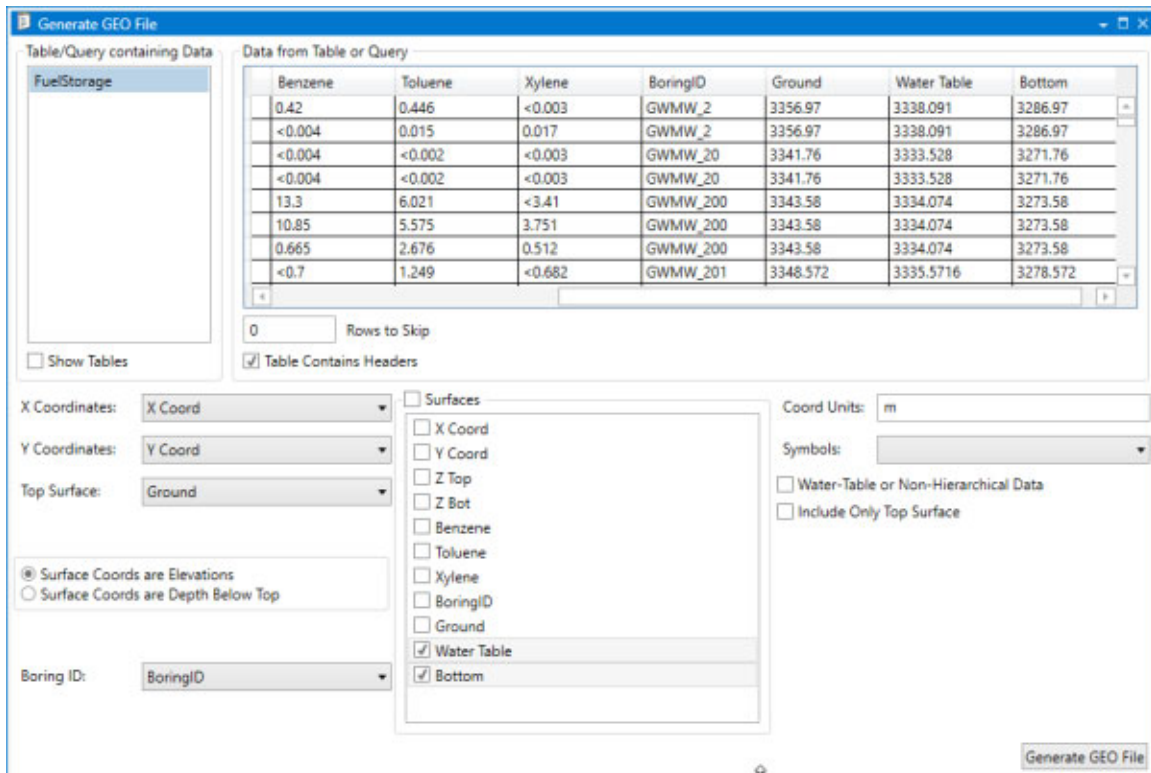
Begin by selecting the Tools tab, and select Generate GEO File.



Let's browse to the folder shown and select the file fuel-storage-gw.xlsx since we mentioned that this file had three surface which we can use for stratigraphic geology. In this case the three surfaces define just two layer which correspond to the vadose and saturated regions, however, that is an important minimal geology file for working with groundwater data.



If we select the only table, choose the correct settings and scroll to the far right we can see the fields that represent our bottom two surfaces:



Based on the values for both surfaces, it is clear they are Elevations and not Depths. For the Surfaces selectors, we don't choose Ground because it is already selected as the Top Surface. This file will have three surfaces defining two layers.

With all of our settings correct as shown above, all we need to do is click the Generate AIDV File button, and let's call the file btx.geo.

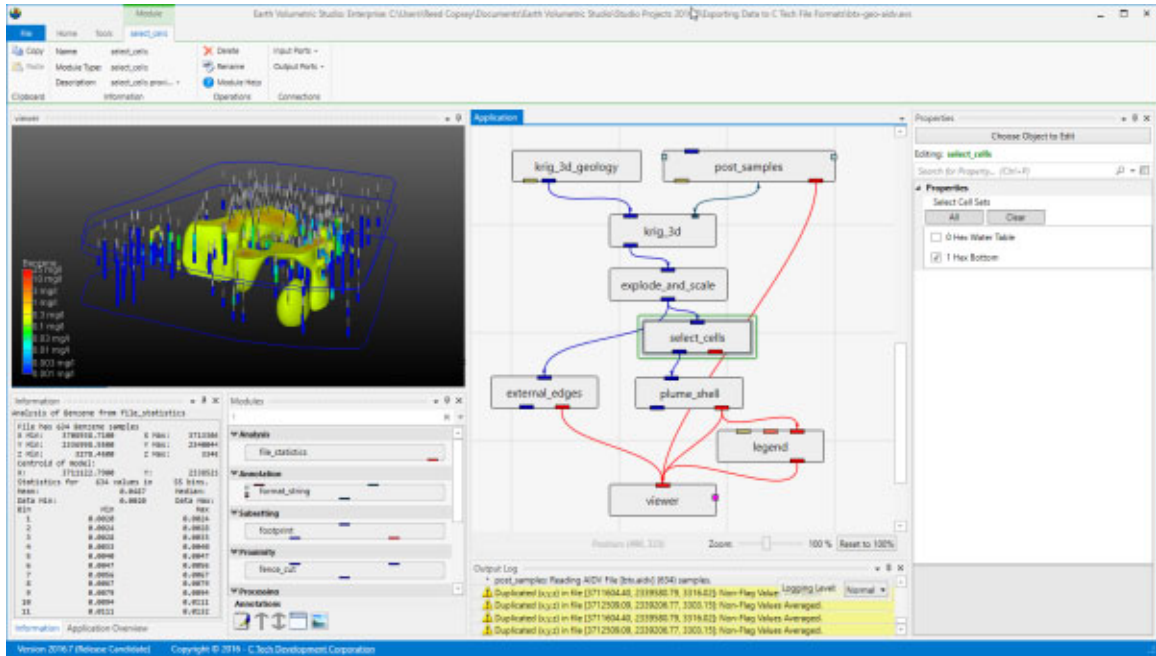
```

C:\Users\Reed Copey\Documents\Earth Volumetric Studio\Studio Projects 2016\7\Exporting Data to C Tech File Formats\btv.geo - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
btv.geo
1 "X Coord" "Y Coord" "Ground" "Water Table" "Bottom" "BoringID"
2 Elevation "Ground" "Water Table" "Bottom" m
3 # C Tech Data Exporter generated GEO File from FUEL-STORAGE-GW.XLSX (07/21/2016 2:16:40 PM)
4 # Columns [FuelStorage]: "X_Coord" "Y_Coord" "Ground" "Water_Table" "Bottom" "BoringID"
5 636 3 1 1 2
6 3712520.66, 2338895.93, 3356.97, 3338.091, 3286.97, "GWHW_2"
7 3712520.66, 2338895.93, 3356.97, 3338.091, 3286.97, "GWHW_2"
8 3710673.8, 2338171.73, 3341.76, 3333.528, 3271.76, "GWHW_20"
9 3710673.8, 2338171.73, 3341.76, 3333.528, 3271.76, "GWHW_20"
10 3712362.37, 2338534.92, 3343.58, 3334.074, 3273.58, "GWHW_200"
11 3712362.37, 2338534.92, 3343.58, 3334.074, 3273.58, "GWHW_200"
12 3712362.37, 2338534.92, 3343.58, 3334.074, 3273.58, "GWHW_200"
13 3712372.22, 2338584.16, 3348.572, 3335.5716, 3278.572, "GWHW_201"
14 3712372.22, 2338584.16, 3348.572, 3335.5716, 3278.572, "GWHW_201"
15 3712372.22, 2338584.16, 3348.572, 3335.5716, 3278.572, "GWHW_201"
16 3712389.03, 2338526.88, 3343.463, 3334.0389, 3273.463, "GWHW_202"
17 3712389.03, 2338526.88, 3343.463, 3334.0389, 3273.463, "GWHW_202"
18 3712389.03, 2338526.88, 3343.463, 3334.0389, 3273.463, "GWHW_202"
19 3712128.41, 2337001.02, 3336.2883, 3331.88649, 3266.2883, "GWHW_203"
20 3712128.41, 2337001.02, 3336.2883, 3331.88649, 3266.2883, "GWHW_203"
21 3712128.41, 2337001.02, 3336.2883, 3331.88649, 3266.2883, "GWHW_203"
22 3712401, 2338578.32, 3348.169, 3335.4507, 3278.169, "GWHW_204"
23 3712401, 2338578.32, 3348.169, 3335.4507, 3278.169, "GWHW_204"
24 3712401, 2338578.32, 3348.169, 3335.4507, 3278.169, "GWHW_204"
25 3712395.9, 2338624.89, 3348.871, 3335.6613, 3278.871, "GWHW_205"
26 3712395.9, 2338624.89, 3348.871, 3335.6613, 3278.871, "GWHW_205"
27 3712395.9, 2338624.89, 3348.871, 3335.6613, 3278.871, "GWHW_205"
28 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
29 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
30 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
31 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
32 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
33 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
34 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
35 3709324.56, 2337968.71, 3344.256, 3334.2768, 3274.256, "GWHW_206"
36 3709492.06, 2337919.9, 3344.074, 3334.2222, 3274.074, "GWHW_207"
37 3709492.06, 2337919.9, 3344.074, 3334.2222, 3274.074, "GWHW_207"

```

Since geo files are rather boring in post_samples, let's do something a bit more interesting with this data.

Below is our application and its output. We cheated a bit and I want to explain where and why.



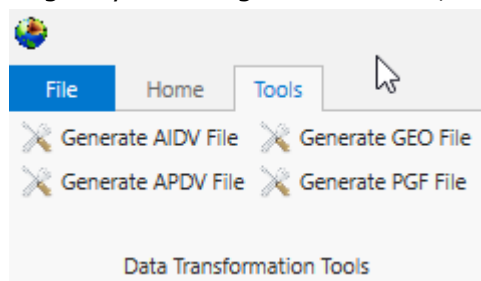
We've kriged groundwater data into both layers of our model. However it doesn't make sense to ever display or do any volumetric analysis of groundwater data in the

vadose zone. We could have used the subset_layers module to get only the single bottom layer corresponding to the saturated zone (aquifer) but if we did that, we wouldn't have both stratigraphic layers which we are displaying with the external_edges module and could display with a variety of other techniques. In that case we would need to create a parallel path in our application where we would use 3d_geology_map to create either the top layer only or both layers in order to display the geology separate from the groundwater chemistry.

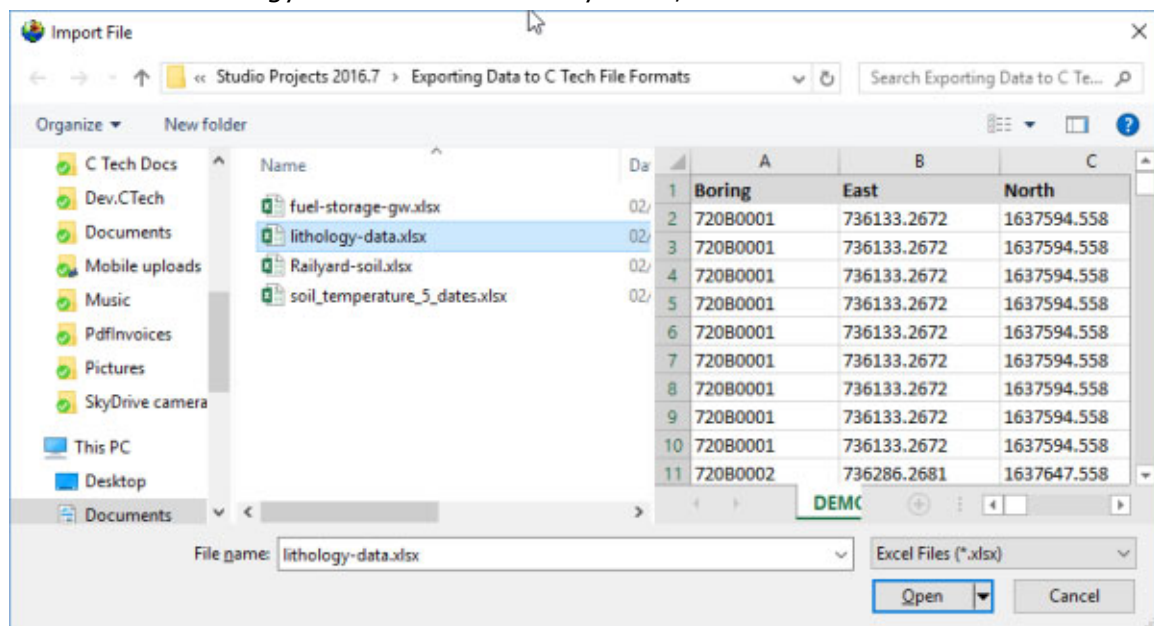
So we cheated and kriged into both layers, but we used the select_cells module to turn off the upper layer before we display the plume with plume_shell. If we wanted to do volumetrics, we would be sure to only do so for the bottom layer. Other than a few seconds used to krig into the vadose layer we've managed to get by with a simpler application.

Creating PGF Files - Lithology

Begin by selecting the Tools tab, and select Generate PGF File.



We'll choose lithology-data.xlsx and its only table, DEMO.



When you look at the table, it is clear that we have a Start and End (Top and Bottom), which means that we need to select the toggle in the upper left. This toggle allows us to select separate X-Y coordinates for the Start and End to handle non-vertical borings, but if the borings are vertical, both can and should be the same (as in this case).

Generate PGF File

Table/Query containing Data: DEMO

Data from Table or Query

Boring	East	North	Ground Surface	Elev-Top	Elev-Bot	Lithology	Depth-Top	Depth-Bot
72080001	736133.267249	1637594.55844	1190.2	1190.2	1189.8	SAND	0	0.4
72080001	736133.267249	1637594.55844	1190.2	1189.8	1188.2	SANDSTONE	0.4	2
72080001	736133.267249	1637594.55844	1190.2	1188.2	1187.8	SAND	2	2.4
72080001	736133.267249	1637594.55844	1190.2	1187.8	1186.7	SANDSTONE	2.4	3
72080001	736133.267249	1637594.55844	1190.2	1186.7	1185.6	SAND	3.5	4
72080001	736133.267249	1637594.55844	1190.2	1185.6	1184.2	GRAVEL	4.6	6
72080001	736133.267249	1637594.55844	1190.2	1184.2	1181.2	SANDSTONE	6	9
72080001	736133.267249	1637594.55844	1190.2	1181.2	1176.2	SANDSTONE	9	11

0 Rows to Skip

☐ Show Tables

☒ Table Contains Headers

☒ Rows Are Intervals (Material Start and End)

Coord Units: ft

X Start Coordinates: East

Y Start Coordinates: North

Z Start Coordinates: Elev-Top

X End Coordinates: East

Y End Coordinates: North

Z End Coordinates: Elev-Bot

☒ Z Coords are True Elevations
☐ Z Coords are Depth Below Ground Surface

Lithology: Lithology

Boring ID: Boring

Generate PGF File

This is another table where we could work in Depths or Elevations. However for a PGF file, the file itself is always in Elevation, so if you choose depth, it just does the conversion before creating the file. We'll just use the elevation fields directly. However, always make sure you've selected the right ones and be consistent.

With all of our settings correct as shown above, all we need to do is click the Generate PGF File button, and let's call the file litho.pgf.

Save

« Studio Projects 2016.7 » Exporting Data to C Tech File Formats

Organize New folder

Documents Mobile uploads Music PdfInvoices Pictures SkyDrive camera This PC Desktop Documents

Name Date modified Type Size

No items match your search.

File name: litho.pgf

Save as type: Lithology (PGF) Files (*.pgf)

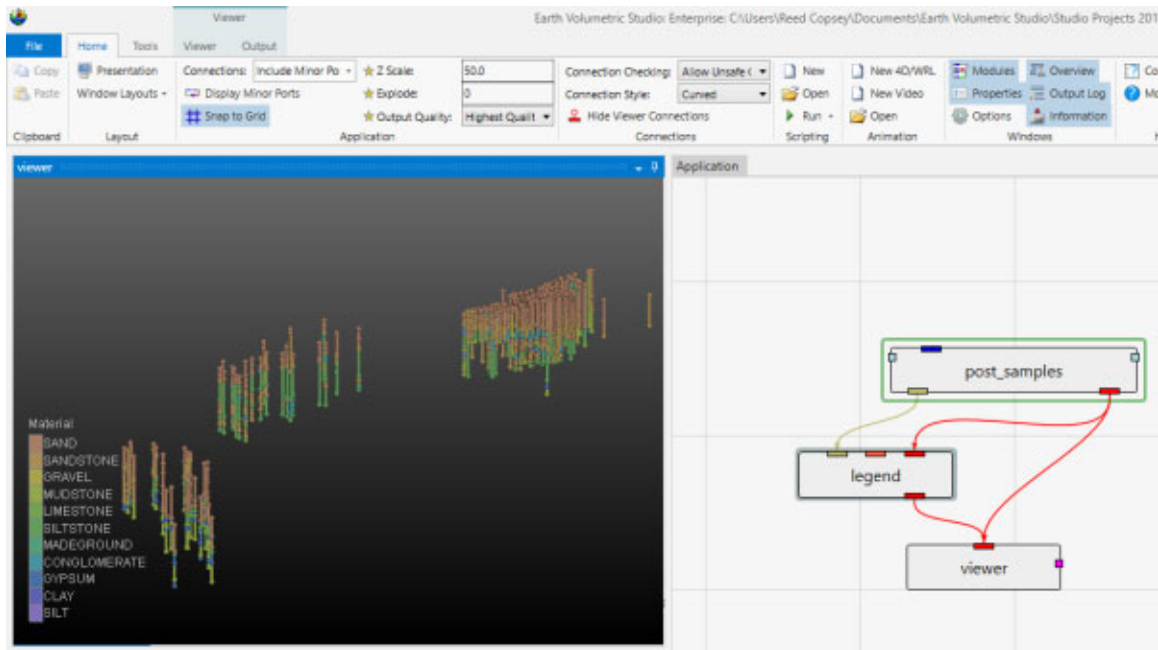
Hide Folders

Save Cancel

Below is the file in Notepad++

```
C:\Users\Reed Copsey\Documents\Earth Volumetric Studio\Studio Projects 2016\7\Exporting Data to C Tech File Formats\litho.pgf - Notepad++
File Edit Search View Encoding Language Settings Macro Run Plugins Window ?
litho.pgf
1 C Tech Data Exporter generated PGF File from LITHOLOGY-DATA.XLSX (07/21/2016 3:08:06 PM)
2 Elevation 0|SAND "1|SANDSTONE "2|GRAVEL "3|MUDSTONE "4|LIMESTONE "5|SILTSTONE "6|MADEGROUND "7|CONGLOMERATE "
3 1699,1
4 # Columns [DEMO]: "East" "North" "Elev-Top" "East" "North" "Elev-Bot" "Lithology" "Boring" "Ground_Surface"
5 736133.267249, 1637594.55844, 1190.2, 0, 720B0001
6 736133.267249, 1637594.55844, 1189.8, 0, 720B0001
7 736133.267249, 1637594.55844, 1188.2, 1, 720B0001
8 736133.267249, 1637594.55844, 1187.8, 0, 720B0001
9 736133.267249, 1637594.55844, 1186.7, 1, 720B0001
10 736133.267249, 1637594.55844, 1185.6, 0, 720B0001
11 736133.267249, 1637594.55844, 1184.2, 2, 720B0001
12 736133.267249, 1637594.55844, 1181.2, 1, 720B0001
13 736133.267249, 1637594.55844, 1176.2, 1, 720B0001
14 736133.267249, 1637594.55844, 1174.9, 1, 720B0001
15 736286.268053, 1637647.55834, 1191.2, 0, 720B0002
16 736286.268053, 1637647.55834, 1190.2, 0, 720B0002
17 736286.268053, 1637647.55834, 1189.8, 0, 720B0002
18 736286.268053, 1637647.55834, 1187.2, 1, 720B0002
19 736286.268053, 1637647.55834, 1186.2, 2, 720B0002
20 736286.268053, 1637647.55834, 1184.1, 1, 720B0002
21 736286.268053, 1637647.55834, 1182.2, 1, 720B0002
22 736286.268053, 1637647.55834, 1175.9, 1, 720B0002
23 737193.272266, 1637709.55665, 1190.2, 0, 720B0003
24 737193.272266, 1637709.55665, 1189.2, 0, 720B0003
25 737193.272266, 1637709.55665, 1188.2, 0, 720B0003
26 737193.272266, 1637709.55665, 1184.2, 0, 720B0003
27 737193.272266, 1637709.55665, 1181.9, 0, 720B0003
28 737193.272266, 1637709.55665, 1179.2, 1, 720B0003
29 737193.272266, 1637709.55665, 1178.9, 0, 720B0003
30 737193.272266, 1637709.55665, 1178.2, 1, 720B0003
31 737193.272266, 1637709.55665, 1177.9, 0, 720B0003
32 737193.272266, 1637709.55665, 1177.2, 1, 720B0003
33 737193.272266, 1637709.55665, 1174.7, 1, 720B0003
34 736918.269877, 1637179.55485, 1194.2, 0, 720B0004
35 736918.269877, 1637179.55485, 1190.8, 0, 720B0004
36 736918.269877, 1637179.55485, 1185.2, 1, 720B0004
Normal text file length: 86780 lines: 1705 Ln: 1 Col: 1 Sel: 0|0 Dos\Windows ANSI INS
```

And in post_samples with a legend we can see that this dataset spans a very large set with borings in three distinct groupings. We need a Z-Scale of 50 to be able to see the borings well.



Data Requirements Overview

The collection and formatting of data for volumetric modeling is often the most challenging task for novice EVS users. This tutorial covers the instructions for

preparing and reviewing all types of data commonly used in Earth Science modeling projects.

The next topics will demonstrate how to visualize these file formats, helping to ensure the quality and consistency of your data.

The following guidelines will simplify your data preparation:

- Use a single [consistent coordinate projection](#) (e.g. UTM, State Plane, etc.) for all data files used on a project, ensuring that X, Y and Z coordinate units are the same (e.g. meters or feet).
- For each file, you must know whether your Z coordinates represent Elevation or Depth below ground surface (most EVS data formats will accommodate both)
- Understand the data formats and what they represent. Below is a list of C Tech's primary ASCII input file formats:
 - **Geologic Data**
 - [PGF](#): A PGF file can be considered a **group of file sections** where each section represents **the lithology for individual borings** (wells). Typical borings logs can be easily converted to PGF format, and many boring log software programs export C Tech's PGF format directly.
 - [GEO](#): This file format represents a series of stratigraphic horizons which define geologic layers. GEO files are limited to data collected from vertical borings and require interpretation to handle pinched layers and dipping strata. The [make_geo_hierarchy](#) module may be used to create GEO files from PGF files, though they can be created in other ways.
 - [GMF](#): This file format represents a series of stratigraphic horizons which define geologic layers. GMF files are not limited to vertical borings as GEO files are. Each horizon can have any number of X-Y-Z coordinates, however interpretation is still required to handle pinched layers and dipping strata. The [make_geo_hierarchy](#) module may be used to create GMF files from PGF files.
 - **Analytical Data**
 - Analytical Data files can be used for many types of data and industries including:
 - Chemical or assay measurements
 - Geophysical data (density, porosity, conductivity, gravity, temperature, seismic, resistance, etc.)
 - Oceanographic & Atmospheric data (conductivity, temperature, salinity, plankton density, etc.)
 - Time domain data representing any of the above analytes
 - [APDV](#): The Analytical Point Data Values (.apdv) format should be used for all analytical data which is (effectively) measured at a point. Even data which is measured over small consistent (less than 1-2% of vertical model extent) intervals should normally be represented as being measured at a single point (X-Y-Z coordinate) at the midpoint of the interval. Time domain data for a single analyte should use this format.

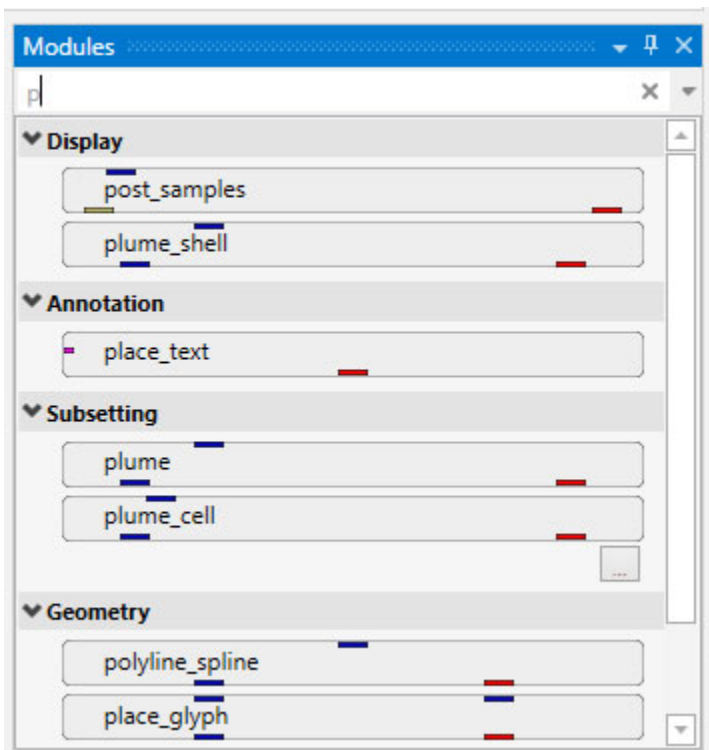
- [AIDV](#): The Analytical Interval Data Values (.aidv) format should be used for all analytical data which is measured over a range of elevations (depths). Data which is measured over variable intervals, usually exceeding 2% of vertical model extent should use this format. Time domain data for a single analyte should use this format.
- The C Tech Data Exporter will export the above formats for data in Excel files and Microsoft Access databases. In all cases, the data source must contain sufficient information to create the desired output.

It is important to view your data prior to using it to build a model. There are many common file errors that can be quickly detected by viewing your raw data files, including:

- Transposing X & Y (Easting and Northing) coordinates
- Using Depth or Elevations incorrectly
- Consistency of geologic and analytical data

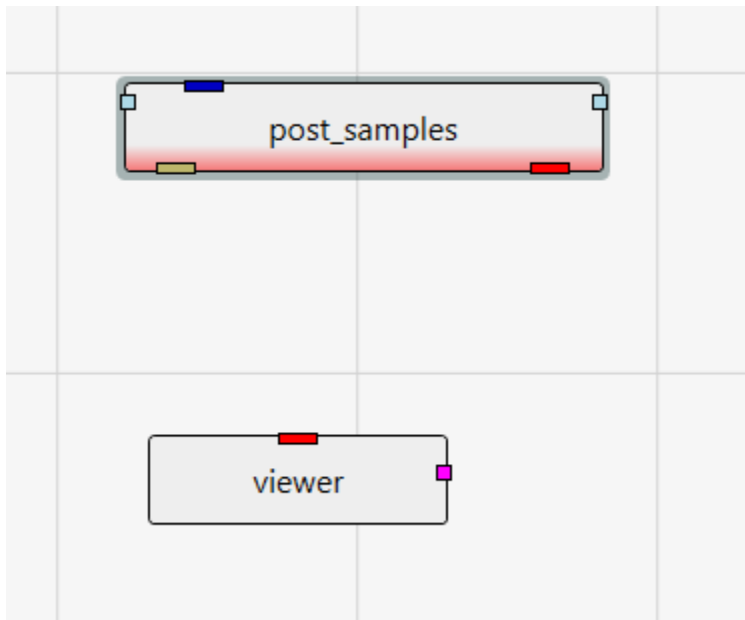
Creating a Simple Application

Let's begin by creating a very simple application. In the *Modules* window, type **p** in the *Search for Module* section.



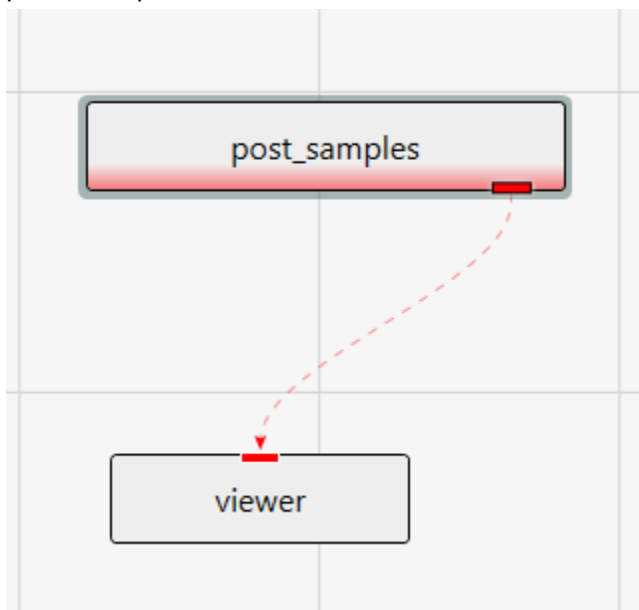
Notice that as soon as you type **p**, only those modules which start with this letter are displayed. The one we want in the first one listed, *post_samples*.

We now want to copy the *post_samples* module into our *Application* window. We do this using the mouse. Left-click on *post_samples* in the *Modules* window and hold the mouse down. Drag *post_samples* to the *Application* window and place it above the viewer as shown below.

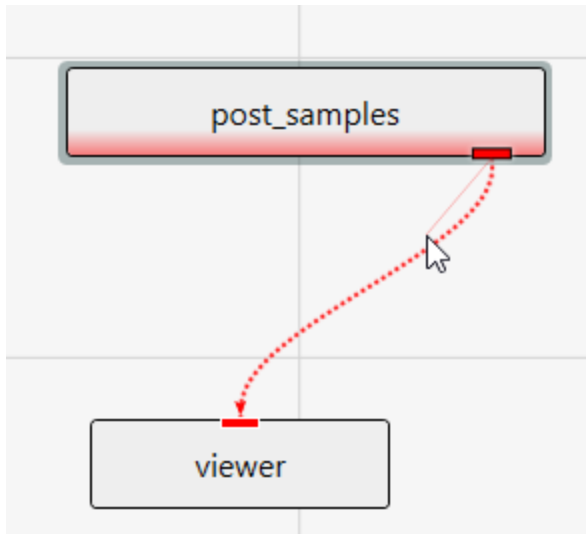


Note that `post_samples` has a red border along the bottom. This tells us that the module has not yet run. This visual indication is very useful, especially with complex applications.

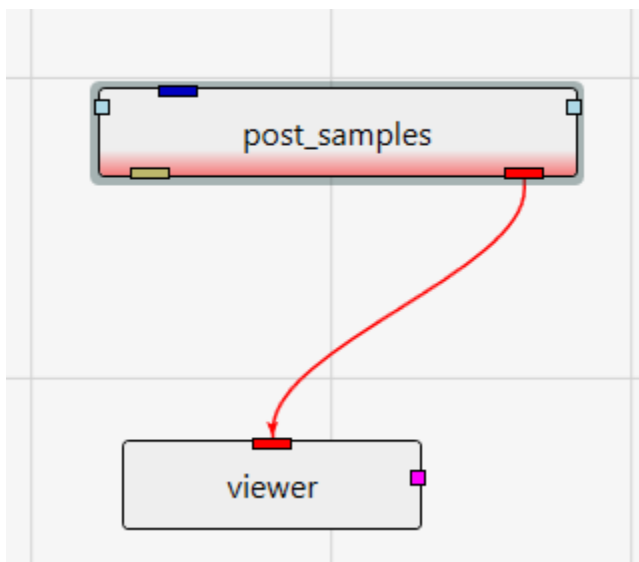
The next step is to connect `post_samples` and the viewer. You can see that the only port color they have in common is red. Left-click in the red output port of `post_samples`:



Then, while holding down the left-mouse, drag a short distance from the port, but near the red-dashed connection, until the dashes turn to dots.



At this point, release the left mouse button and the connection is made. The reason for the dashed and dotted lines is that there are often multiple modules that *can* be connected. All will be shown dashed, but only the connection which is closest to the cursor will be dotted.



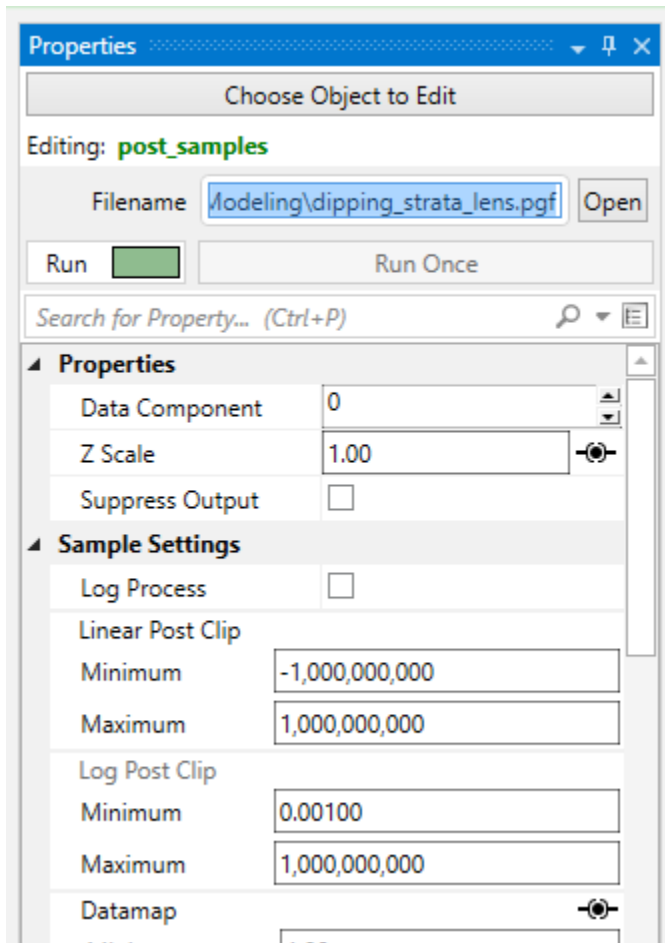
Deleting a connection

If we make an incorrect connection, we can delete the connection. To delete the connection, merely click on it to highlight it and then press the Delete key on your keyboard.

Viewing PGF Files

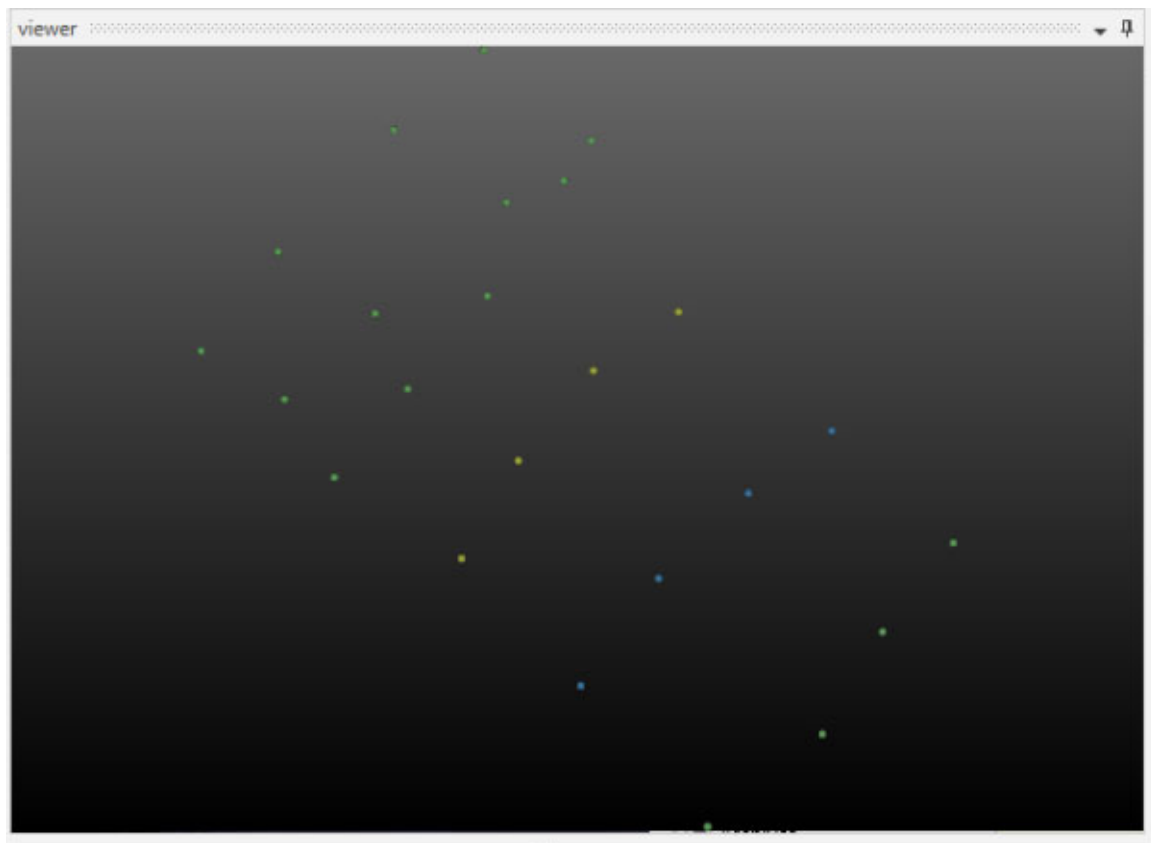
With the simple application from the previous topic, let's read a [PGF](#) file and see that data represented in the viewer.

Double-left-click on `post_samples` in the *Application* window to make its settings editable in the *Properties* window.

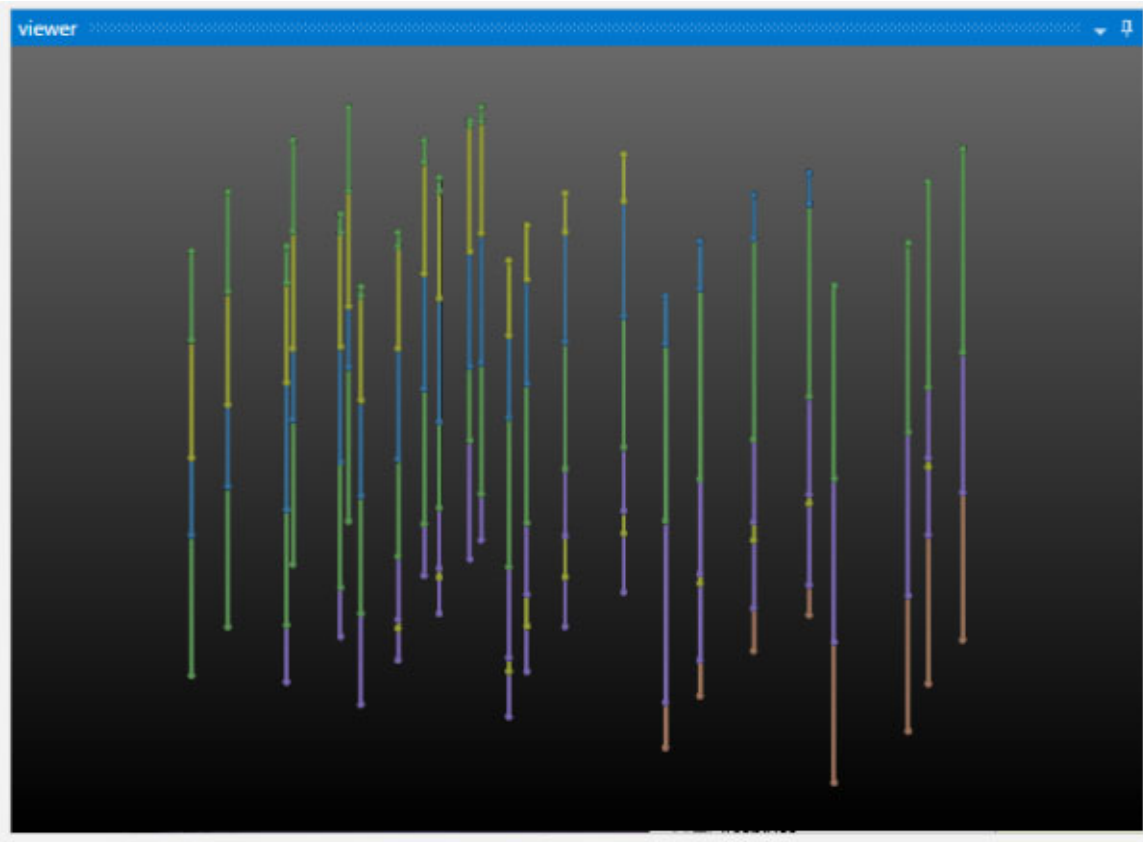


post_samples will automatically adjust many of its settings based on the type of file read. Click on the Open button and browse to the *Lithologic Geologic Modeling* folder in Studio Projects and select dipping_strata_lens.pgf.

post_samples will automatically run and your viewer should show a top view of:



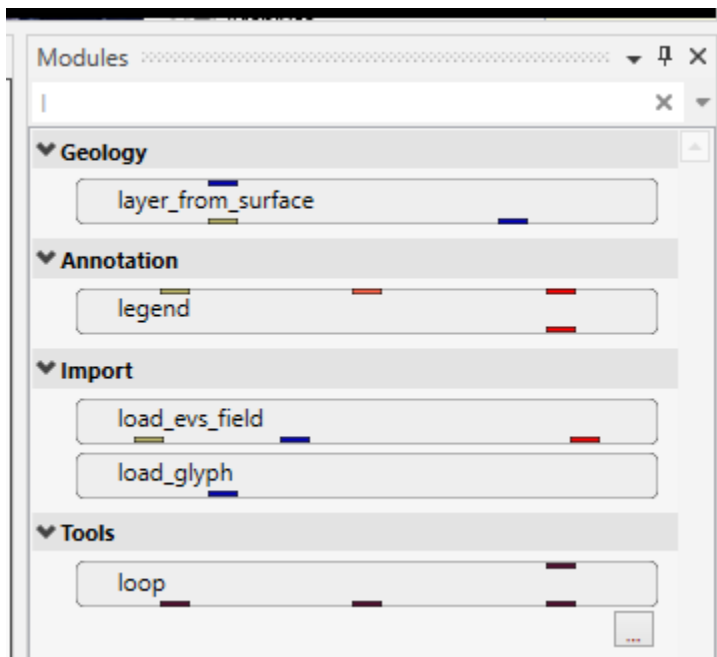
By default, we are seeing a top view of the borings represented in the PGF file. Using the left mouse button, rotate the view so you can see the 3D borings which are colored by lithology (geologic material).



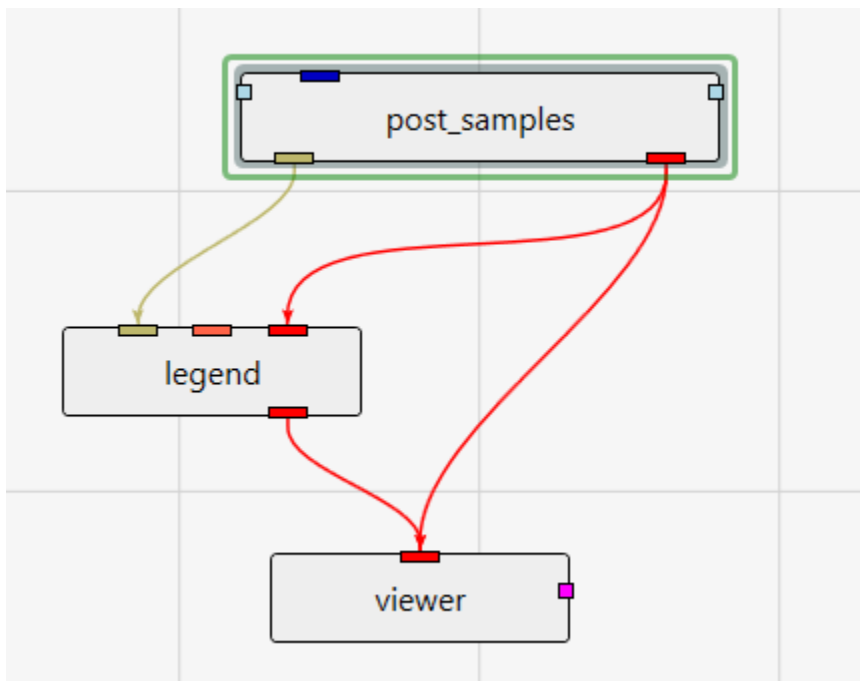
The image above demonstrates the default display of PGF (pregeology) files. The lithology intervals are colored by material and spheres are located at the beginning and end of each interval.

The colors represent material and range from purple (low) to orange-brown (high). Since this is geology, let's add a legend to make it clear what materials correspond to our colors.

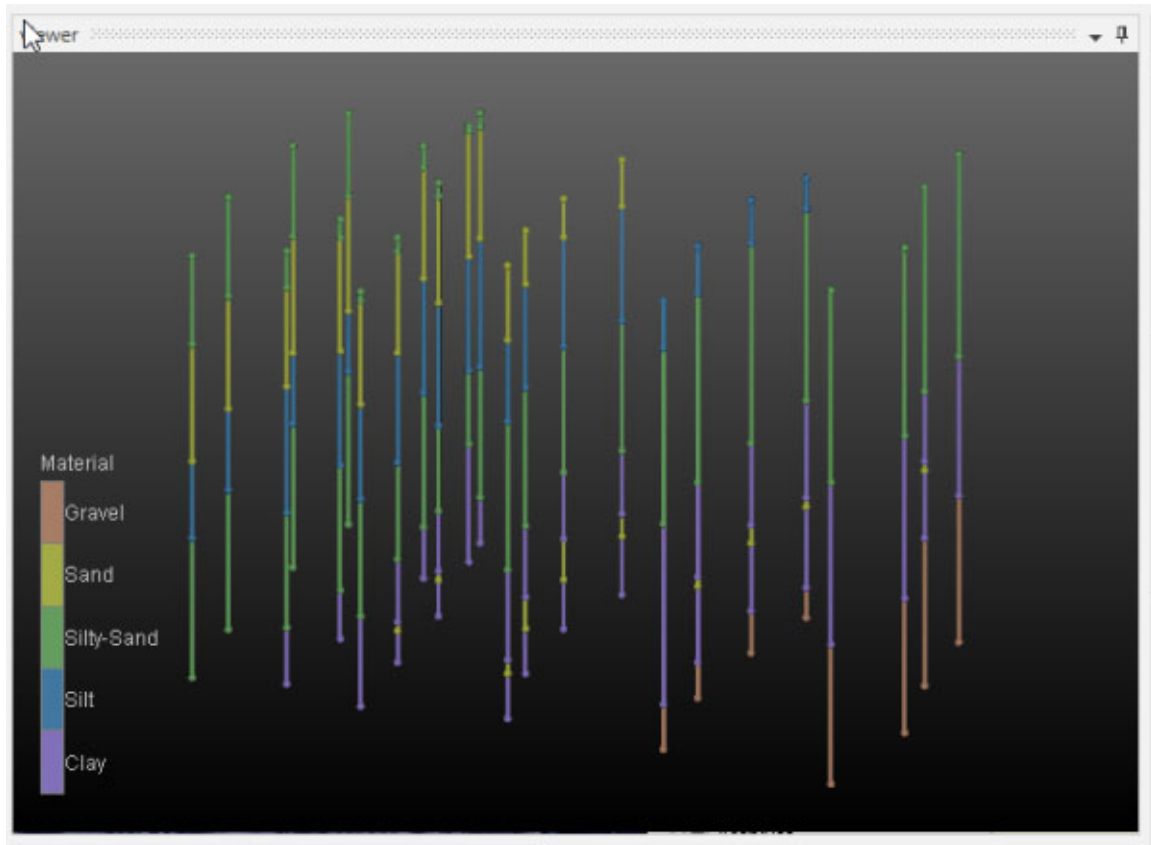
Type "l" in the Modules window and it will display



Copy legend to the Application (left-click and drag) and make the three new connections as shown below



You can move the modules around so that your application and the associated connections between modules is as clear as possible. However, the arrangement (placement) of the modules does not affect how the application behaves. With legend our view becomes:

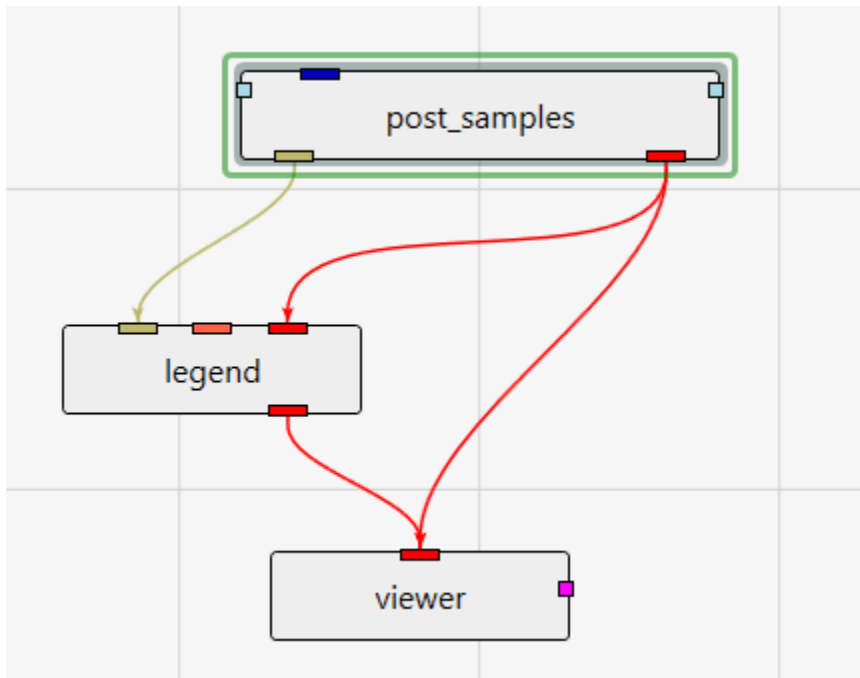


In the next topic, [Viewing GEO Files](#), we'll adjust colors

Viewing GEO Files

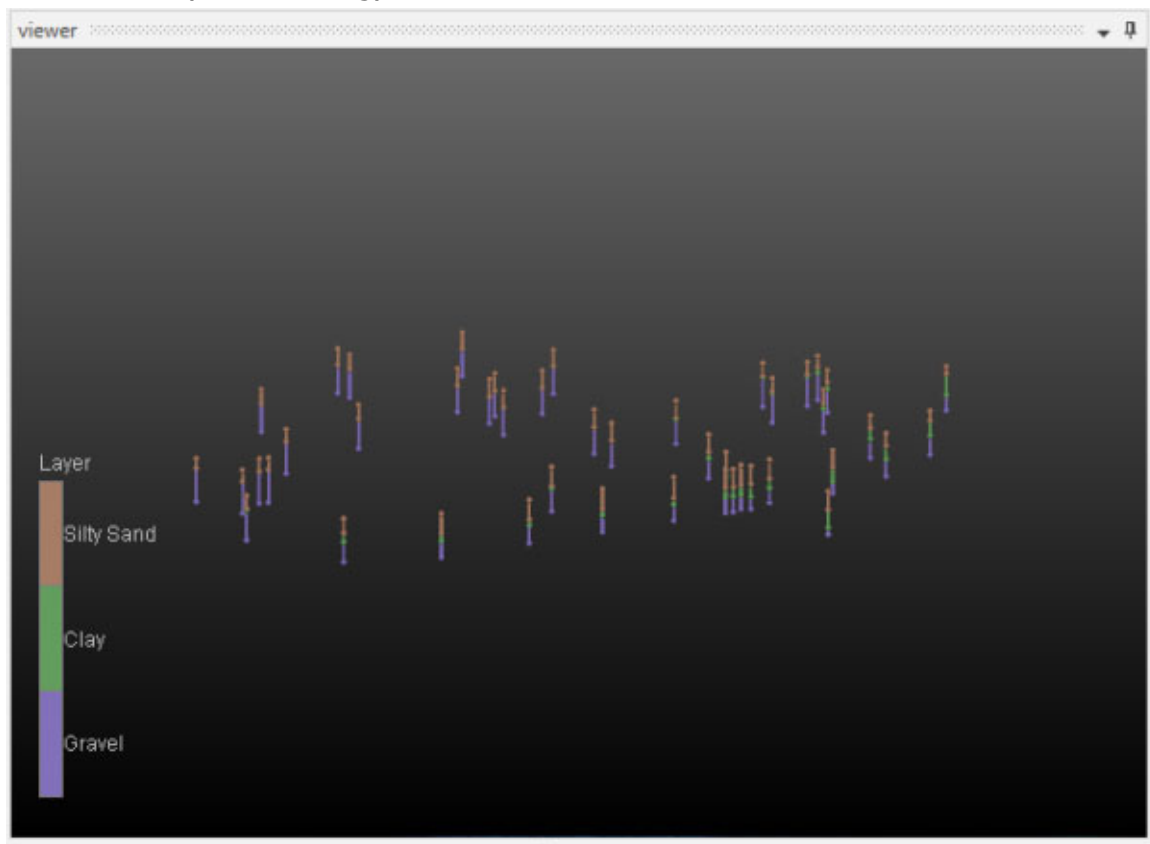
To view a [GEO](#) file, the process is nearly identical as with PGF.

As long as you didn't change some of the linked parameters in post_samples' Properties we can just change the filename from our last topic. Otherwise replace post_samples with a fresh instance:



Click on the Open button and browse to the *Lithologic Geologic Modeling* folder in Studio Projects and select *railyard_pgf.geo*.

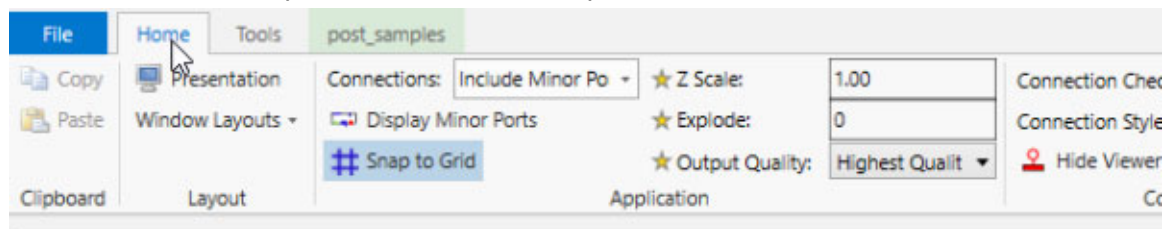
Your viewer (after rotating) should show:



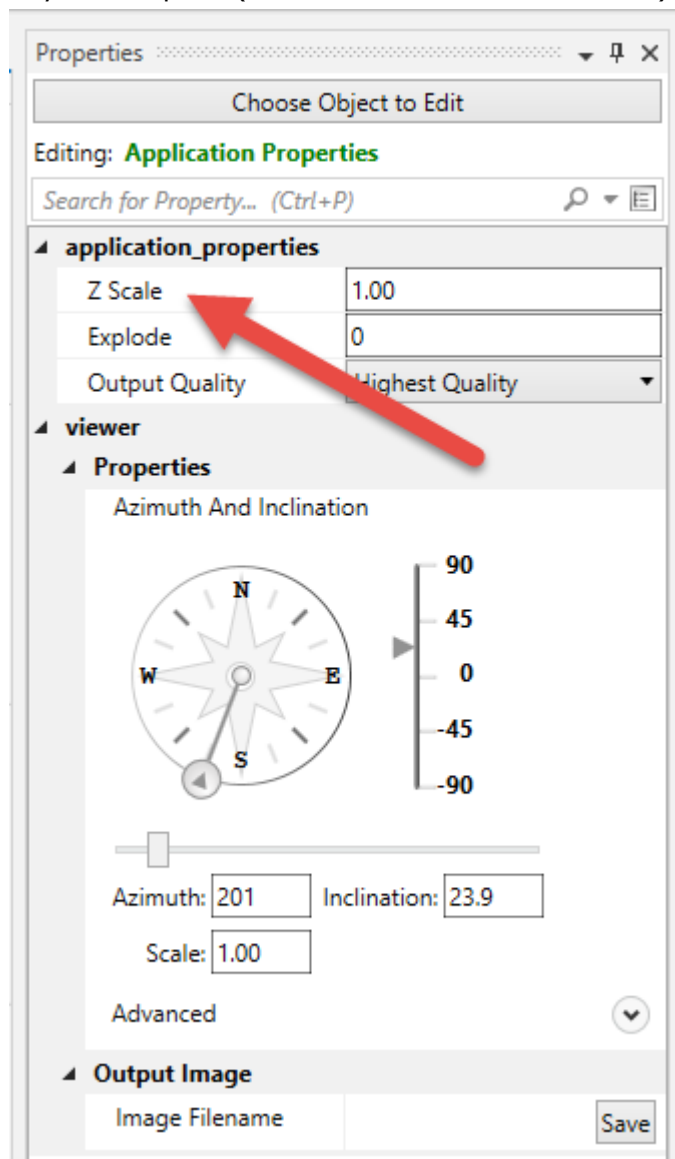
Your first question might be, why are the borings so short?

Welcome to the real world. In the last topic we were dealing with a site where the z-extent was comparable to the x & y extents. But for this site, the z extent is 5-10% of the x-y extent. In order to better see the Stratigraphy represented by our .GEO file, we need to apply some vertical exaggeration, which we also refer to as Z-Scale.

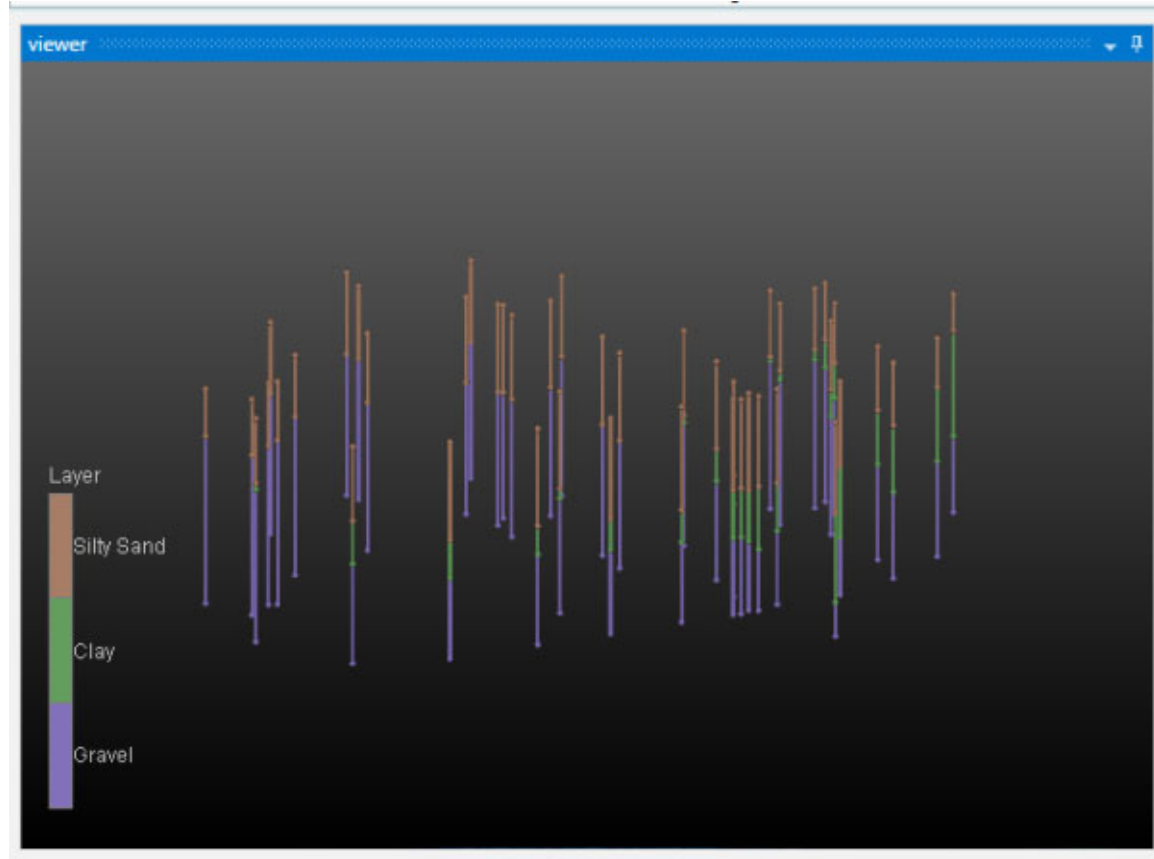
We find the Z-Scale parameter in one of 2 places. Either on the Home Tab



or in the Application Properties. To get to the Application Properties, double click on any blank space (not on a module or connection) in the Application.



Notice if we change it here, to be 5, it changes on the Home tab and in every module which has a Z-Scale. Our viewer now shows:

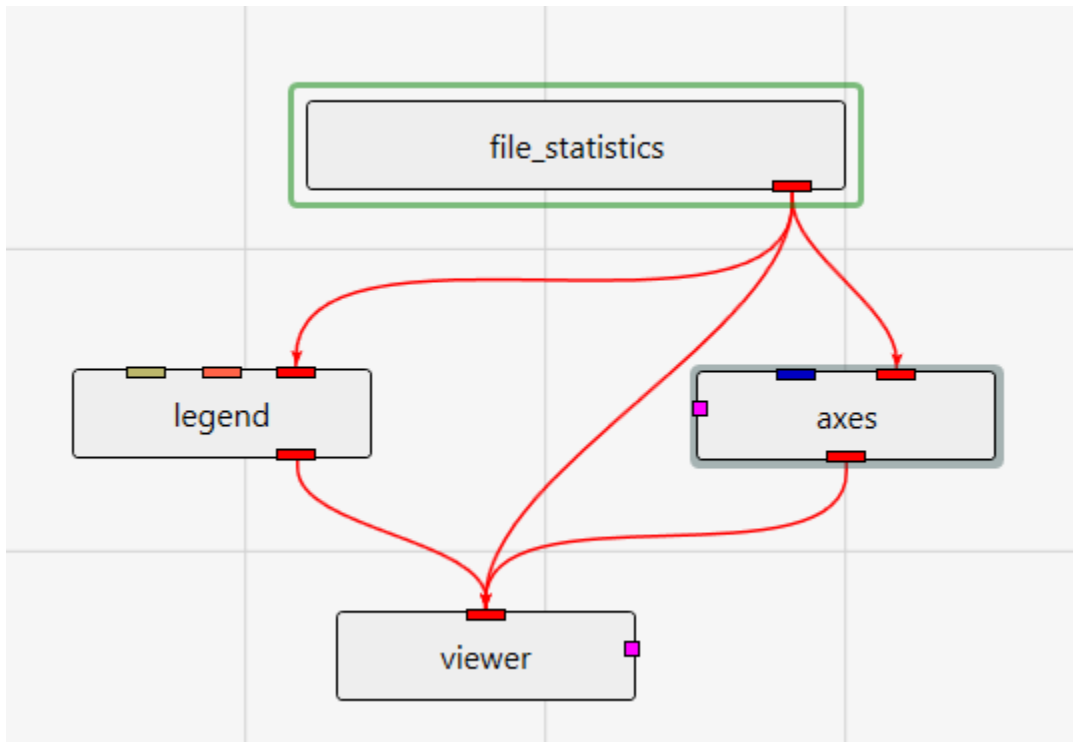


Please note: We could have changed the Z-Scale in `post_samples`, but by doing so, we would have broken its link to the Global Z-Scale on the Home tab and Application Properties. In general you want all modules to share the Global Z-Scale, but there are times when you want control on a module-by-module basis. That is why we allow both.

Viewing GMF Files

[GMF](#) files are different than most other C Tech file formats in that the data is specifically NOT associated with borings. GMF files can be viewed using `post_samples`, but `file_statistics` can often be more useful, especially when dealing with large datasets.

Let's build a new application:



file_statistics (and post_samples) will only display a single surface of a GMF file at one time. The advantage of file_statistics is that it will provide the extents and basic statistics information. The Data Component parameter determines which surface is displayed. 0 (zero) is the first surface.

file_statistics outputs points which are colored by elevation (for GMF files).

Double click on file_statistics and select the file Reference\bathymetry-with-fault.gmf.

The viewer should show:

Properties ▾ 🔍 ✕

Choose Object to Edit

Editing: **axes**

Run Run Once

Search for Property... (Ctrl+P) 🔍 📋

	Z:	200
Interval Reference Point	X:	285,000
	Y:	613,000
	Z:	-1,200
Max Intervals		100

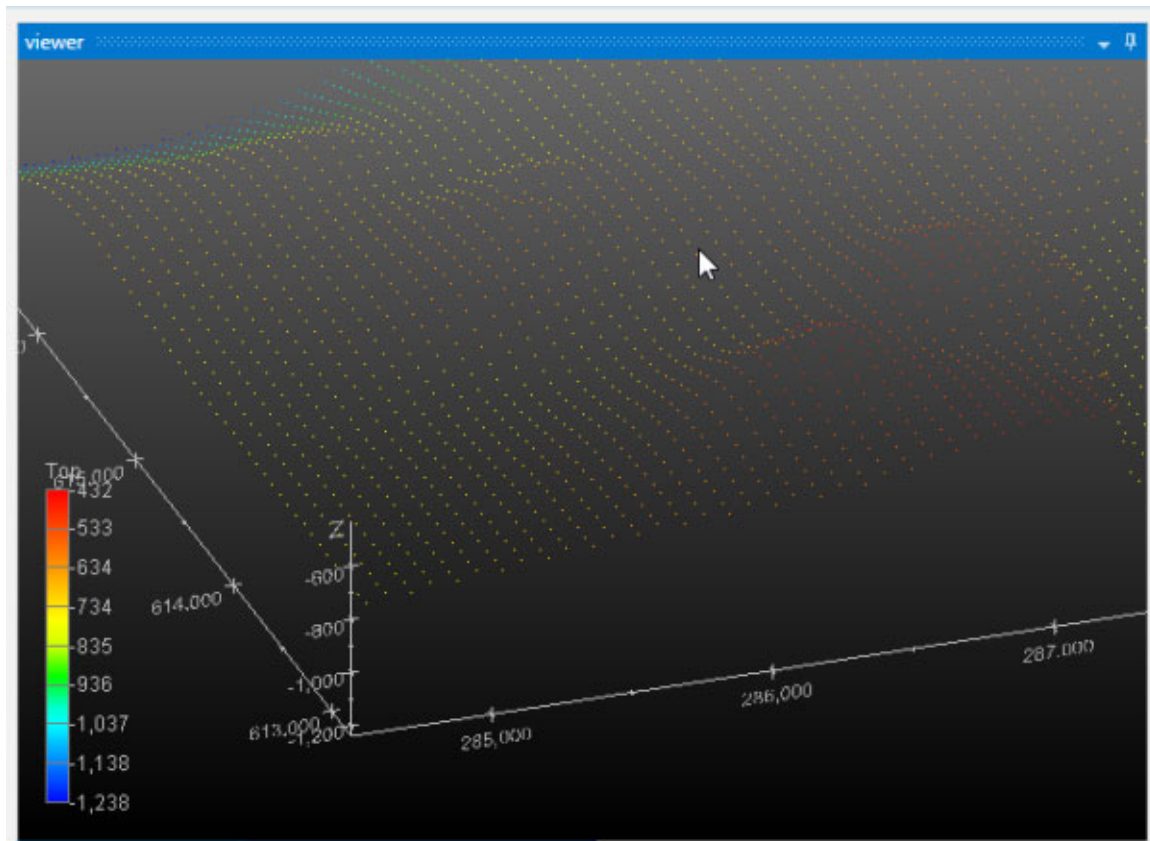
▾ **Display Settings**

Draw Box	<input type="checkbox"/>
Draw Ticks	<input checked="" type="checkbox"/>
Box Line Width	0 ⬆ ⬇ ⬆
Display Major	<input checked="" type="checkbox"/>
Display Minor	<input checked="" type="checkbox"/>
Major Gridline Style	Dashed ▾
Minor Gridline Style	Dotted ▾
Major Line Width	0 ⬆ ⬇ ⬆
Minor Line Width	0 ⬆ ⬇ ⬆
Display XY Grid At Min	<input type="checkbox"/>
Display XY Grid At Max	<input type="checkbox"/>
Display XZ Grid At Min	<input type="checkbox"/>
Display XZ Grid At Max	<input type="checkbox"/>
Display YZ Grid At Min	<input type="checkbox"/>
Display YZ Grid At Max	<input type="checkbox"/>

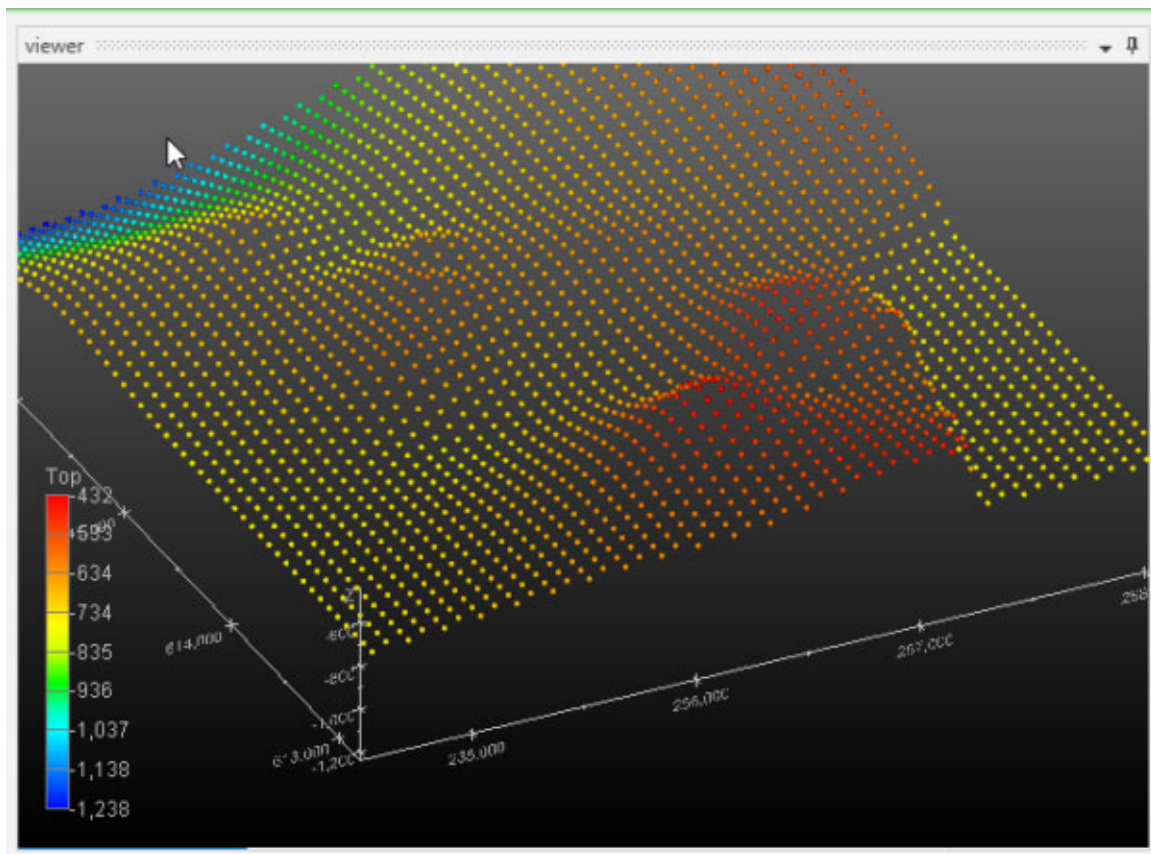
▾ **All Axes Settings**

Display Axes	<input checked="" type="checkbox"/>
Axes Color	 White ▾
Label Type	TrueType Fonts ▾
TrueType Font Style	⬇
Line Font Style	Single Line ▾

Now we have:

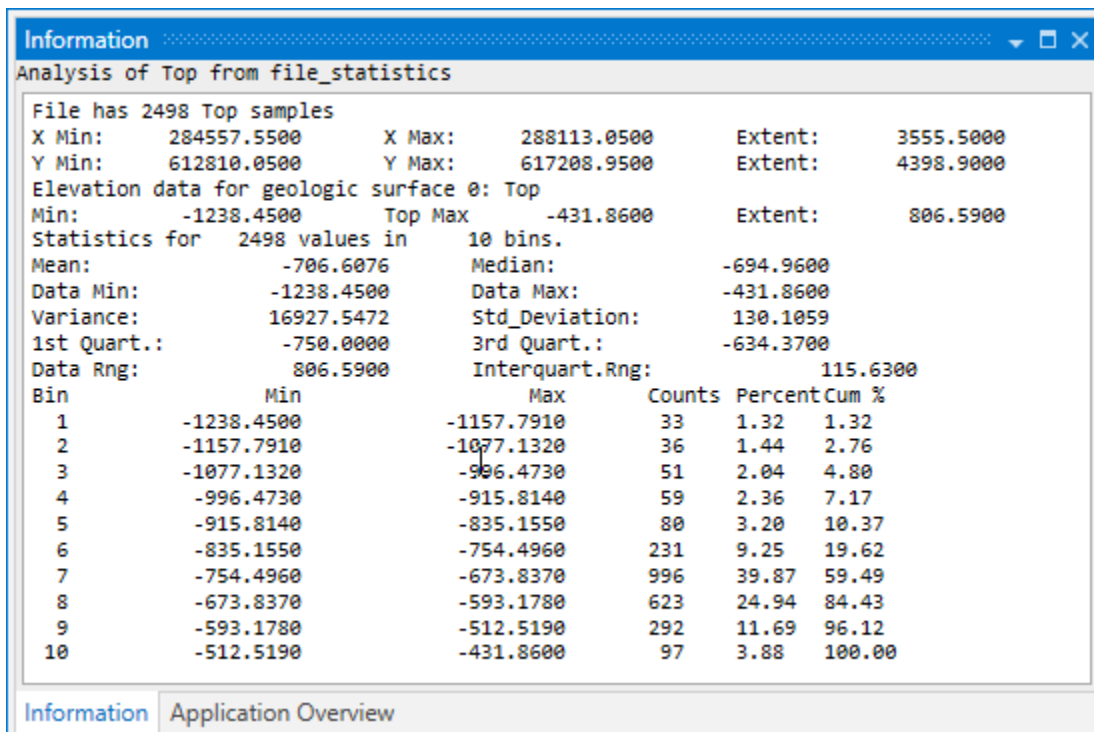


In this view, each point is displayed as a single pixel point. You can increase the size to be a square of 2x2 pixels or larger using the *Point Width* parameter. Alternatively you can display *Points As Spheres* and make them any size (this is the computed default of 12.6).



When file_statistics runs, it provides the following information to the *Information* window.

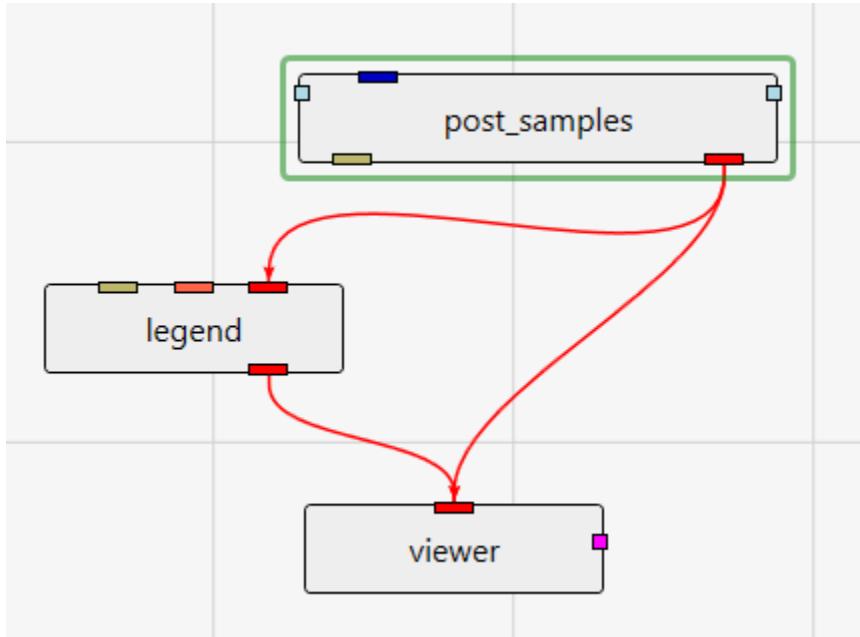
Note that *Number of Bins* was set to 10, and Detailed Statistics was turned on.



Viewing APDV Files

[APDV](#) files represent analyte data which is measured at points. The data can be collected at scattered locations or along borings. When boring IDs are included in the file, post_samples will draw the borings as well as the samples.

Create the following application. It is nearly identical to the application used for PGF files, but we do not need to connect the yellow port which contains geology or lithology names, as those are not applicable to APDV (or [AIDV](#)) files. However, if you do connect it, it won't hurt anything.



Double click on **post_samples** to open its Properties window. Select *Railyard Facility Complex Python Scripting\railyard.apdv* and change the Z Scale to 5 on the Home Tab or Application Properties.

Properties

Choose Object to Edit

Editing: **post_samples**

Filename

Railyard Facility Complex Python Scripting\railyard.apdv

Open

Run

Run Once

Search for Property... (Ctrl+P)

Properties

Data Component

0

Z Scale

5.00

Suppress Output

☐

Sample Settings

Log Process

☒

Linear Post Clip

Minimum

-1,000,000,000

Maximum

1,000,000,000

Log Post Clip

Minimum

0.00100

Maximum

1,000,000,000

Datamap

Minimum

0.00100

Maximum

9,364

Detection Limit

0.000100

Less Than Multiplier

0.100

Default Coordinate Units

m

Display Samples

☒

Sphere Count Limit

10000

Maximum Sample Threshold

50000

Synthetic Only

☐

Display Interval As

Tubes

Tube Scale

1.00

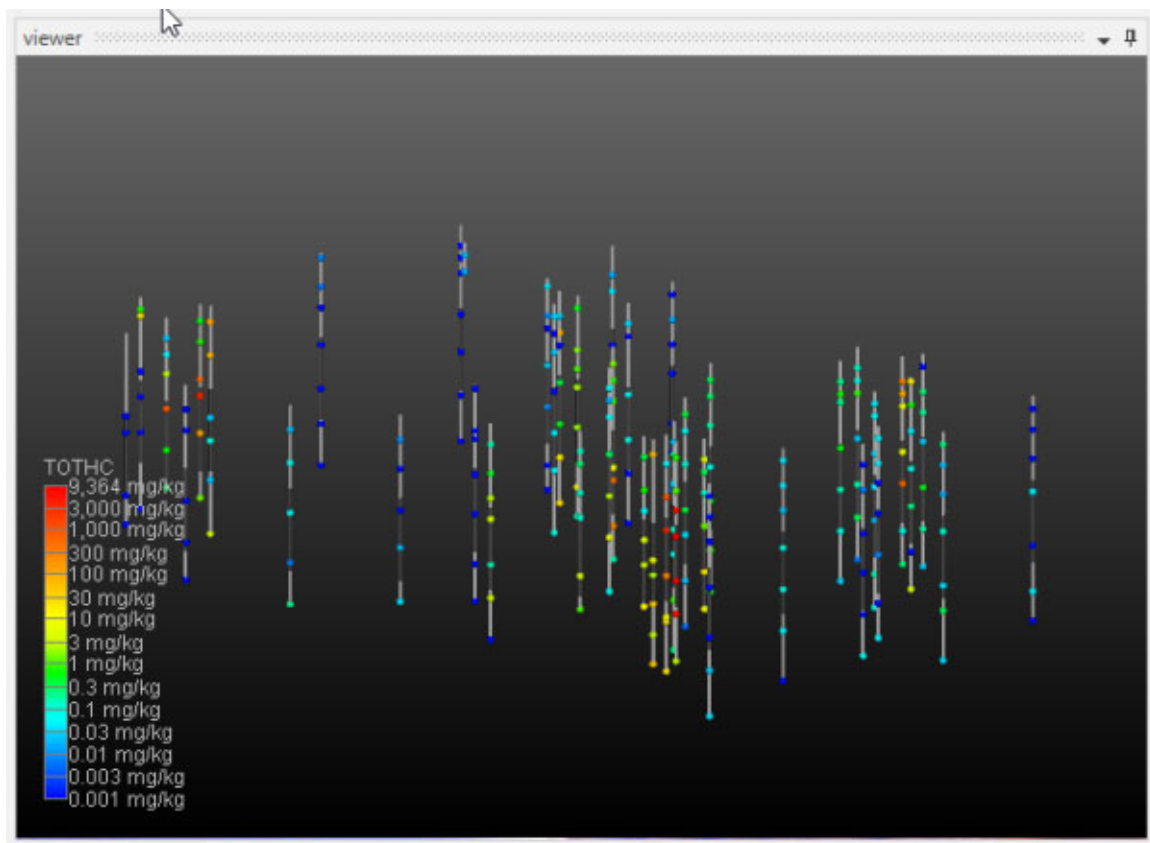
Tube Resolution

8

Close Tubes

☒

to show the following in the viewer.



post_samples has many options for displaying this type of data (also applicable to PGF, GEO, AIDV). These include (but are not limited to):

- displaying the data as colored tubes (with or without spheres/glyphs)
- using different glyphs to represent each sample (a sphere is the default glyph)
- changing the diameter of glyphs or tubes based on the data magnitudes
- labeling the samples and/or borings

Let's see the four options above:

It is easy to display colored tubes. You can scroll down to the *Color Tube Settings* or better yet collapse the other settings until you get to it.

Change *Display As* from None to **Tubes**.

Properties

Choose Object to Edit

Editing: **post_samples**

Filename

Run ☐

Search for Property... (Ctrl+P)

Properties

Data Component	0
Z Scale	5.00
Suppress Output	<input type="checkbox"/>

▶ Sample Settings

▶ Glyph Settings

▶ Subsetting Settings

▶ Collapse To 2D

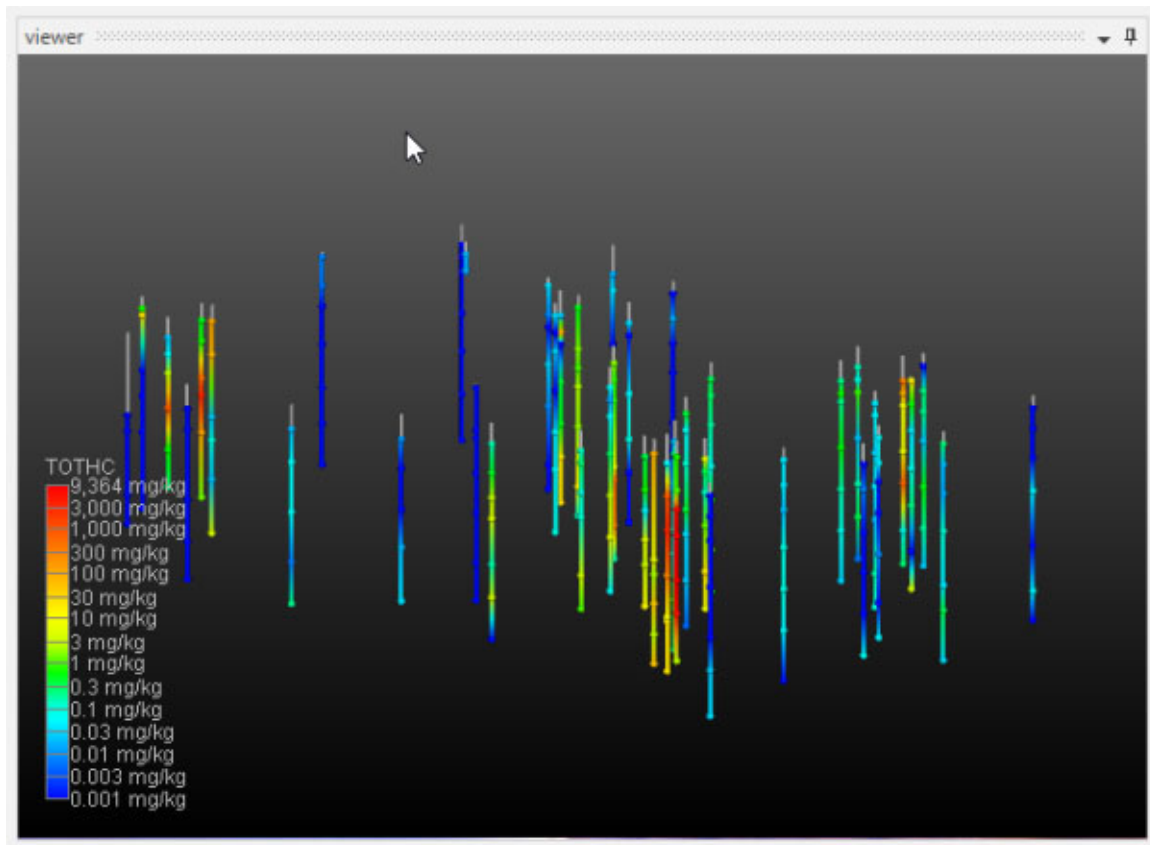
▶ Geology Settings

▶ Time Settings

▶ Boring Tube Settings

▶ Color Tube Settings

Display As	Tubes
Tube Ratio	90.00 %
Tube Resolution	8
Max Variation	33.00 %
Phase	0
Close Tubes	<input checked="" type="checkbox"/>
Connect Lines	<input type="checkbox"/>



To change glyphs is incredibly simple. We just go to the Glyph Settings, and we'll change the Generated Glyph to be Cube instead of the default Sphere, and we'll also set the Maximum Scale Factor to be 200%

Properties ▾ ▴ ✕

Choose Object to Edit

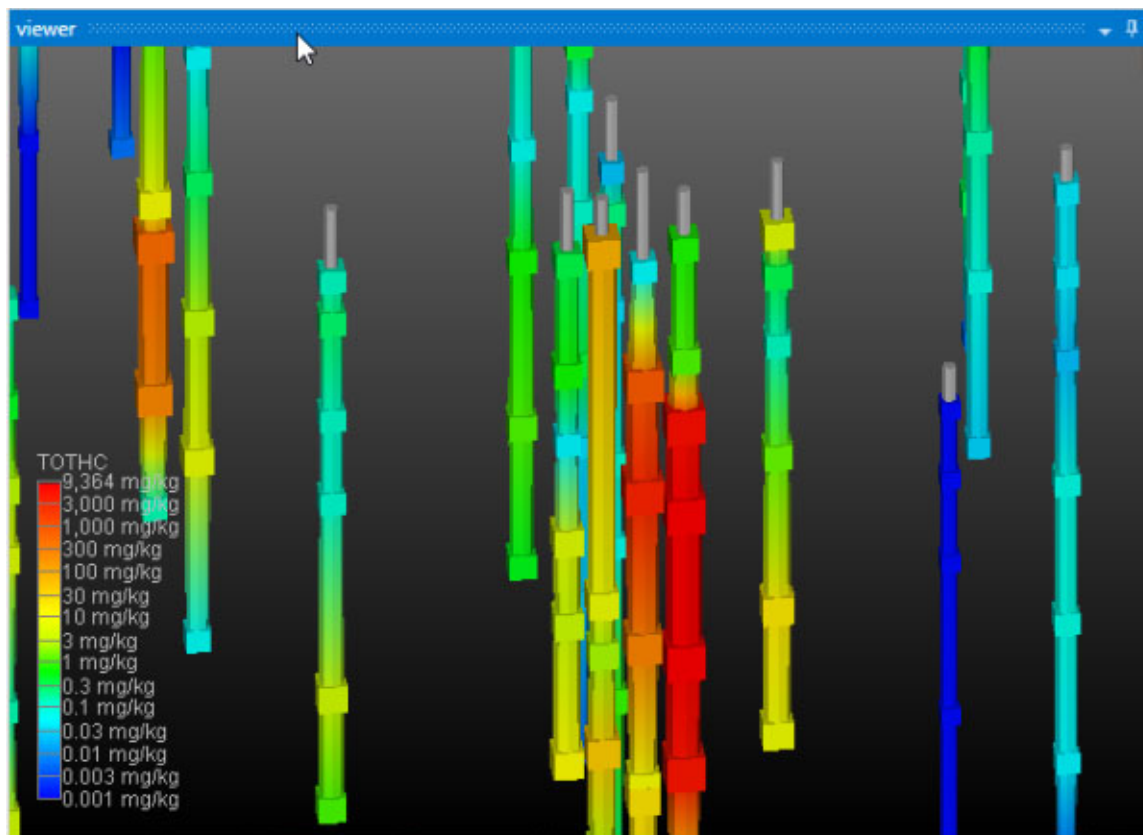
Editing: **post_samples**

Filename

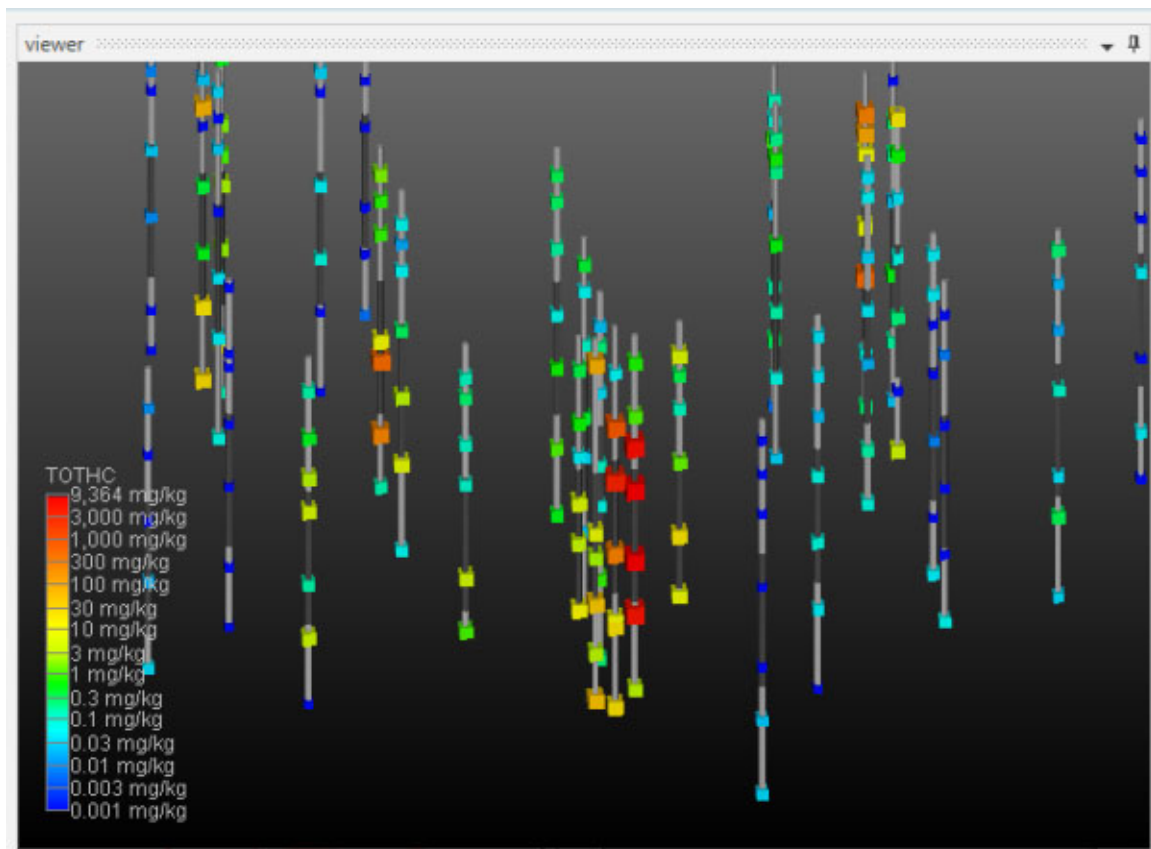
Search for Property... (Ctrl+P) 🔍 ▾

Tube Resolution	<input type="text" value="8"/>
Close Tubes	<input checked="" type="checkbox"/>
Phase	<input type="text" value="0"/>
▲ Glyph Settings	
Glyph Size	<input type="text" value="3.03"/> 🔍
Priority	<input type="text" value="Maximum"/>
Minimum Scale Factor	<input type="text" value="100.00 %"/>
Maximum Scale Factor	<input type="text" value="200.00 %"/>
Use Log Data	<input type="checkbox"/>
Generated Glyph	<input type="text" value="Cube"/>
Sphere Subdivisions	<input type="text" value="2"/> 🔍
Glyph Resolution	<input type="text" value="8"/>
Primary Axis Factor	<input type="text" value="100.00 %"/>
Secondary Axis Factor	<input type="text" value="100.00 %"/>
Roll	<input type="text" value="0"/>
▲ Subsetting Settings	

Since we've still left colored tubes on, our viewer shows:



Before we make any other changes let's turn off Colored Tubes which will change our view to be:



Finally, we'll add labels at each sample and the top of the borings:

Properties ▾ 🔍 ✕

Choose Object to Edit

Editing: **post_samples**

Filename

Search for Property... (Ctrl+P) 🔍 ▾ 📋

Tube Ratio	<input type="text" value="90.00 %"/>	▲
Tube Resolution	<input type="text" value="8"/>	
Max Variation	<input type="text" value="33.00 %"/>	
Phase	<input type="text" value="0"/>	
Close Tubes	<input checked="" type="checkbox"/>	
Connect Lines	<input type="checkbox"/>	

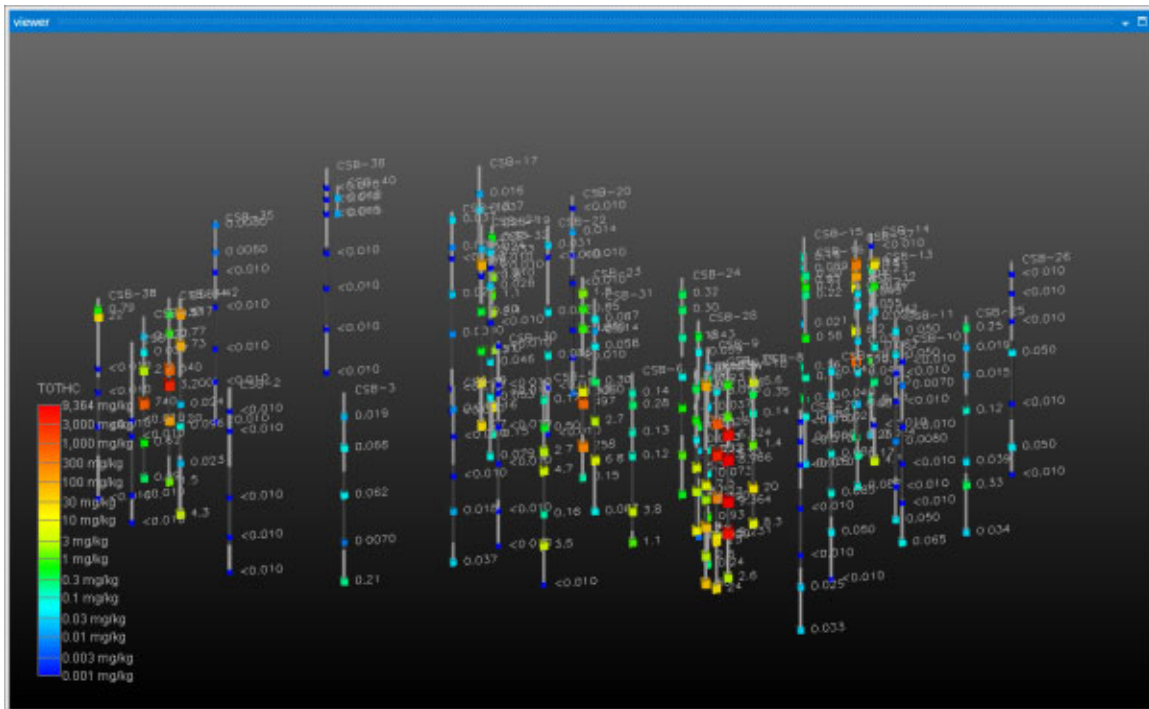
▲ **Label Settings**

Features To Label	<input type="text" value="Well and Value"/>	▾
Label Type	<input type="text" value="Line Fonts"/>	▾
TrueType Font Style		▾
Line Font Style	<input type="text" value="Single Line"/>	▾
3D Label Settings		▾
Forward Facing Style		▾
Label Color	<input type="color" value="DarkGray"/>	▾
Label Angle	<input type="text" value="0"/>	

Format Label Numbers ▲

Format Specifier	<input type="text" value="Numeric"/>	▾
Precision Determined By	<input type="text" value="Automatically"/>	▾
Decimal Points	<input type="text" value="2"/>	▲ ▾
Threshold for Zero Dec. Points	<input type="text" value="10"/>	▾

Top Offset	<input type="text" value="1.00"/>
Radial Offset	<input type="text" value="1.00"/>
Z Offset	<input type="text" value="0"/>
X Blank	<input type="text" value="0"/>
Y Blank	<input type="text" value="0"/>
Z Blank	<input type="text" value="0"/>
Favor Min Value	<input type="checkbox"/>



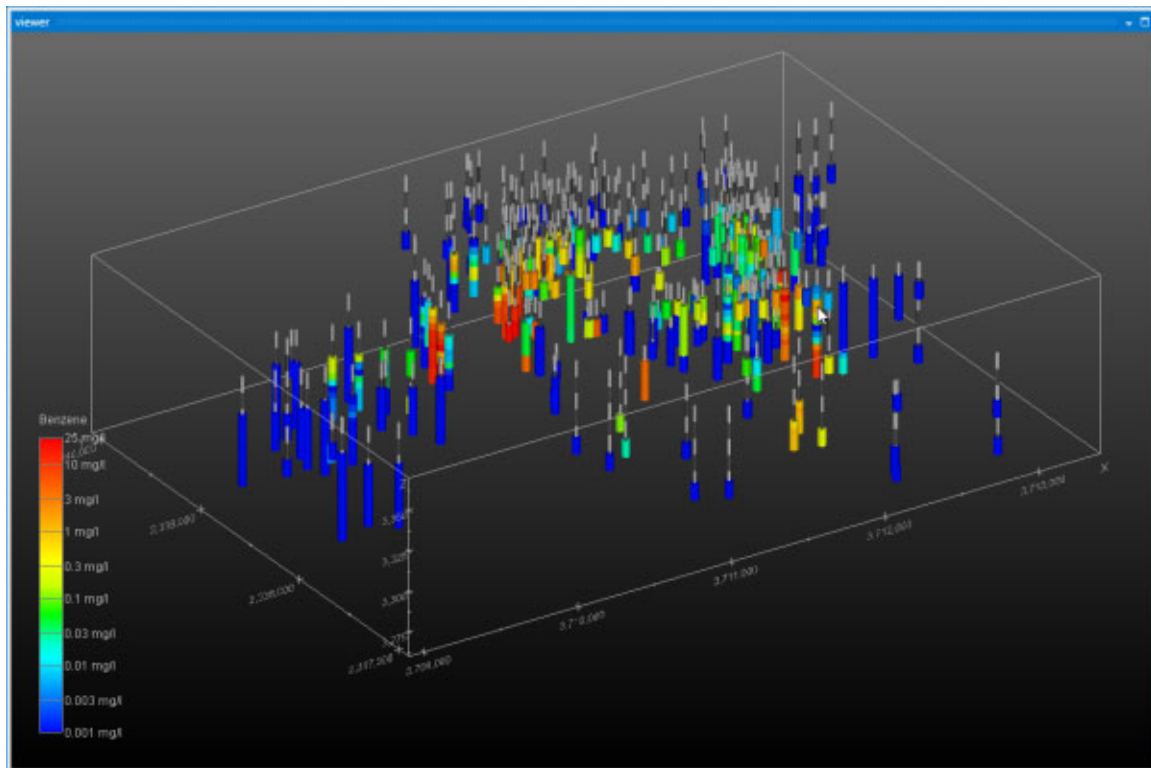
As you can see above, the labels get very dense and it can be hard to see the most important (hottest) samples. `post_samples` provides functionality to allow you to blank out lower value labels within some distance of hotter samples. Set the X Blank, Y Blank and Z Blank settings to match below and zoom in to see the labels intelligently culled.

Label Settings

Features To Label	Well and Value
Label Type	Line Fonts
TrueType Font Style	
Line Font Style	Single Line
3D Label Settings	
Forward Facing Style	
Label Color	DarkGray
Label Angle	0
Format Label Numbers	
Format Specifier	Numeric
Precision Determined By	Automatically
Decimal Points	2
Threshold for Zero Dec. Points	10
Top Offset	1.00
Radial Offset	1.00
Z Offset	0
X Blank	20.0
Y Blank	20.0
Z Blank	3.00
Favor Min Value	<input type="checkbox"/>

Note that the blanking can favor minimum values, such as when you are kriging pH levels.

In the axes module, we've turned off all of the grids (as we did for viewing GMF files) and have set the axes color to be light grey. However we left the box on.



Packaging Data Into Your Applications

Packaging data into your Applications has many advantages including:

- Integrating your application into a single file
- Making your application easier to share with others
- Ensuring the correct version of data files are associated with the application
- Minimizing or eliminating the possibility of application corruption should one or more files become modified or lost
- Packaged applications generally load faster

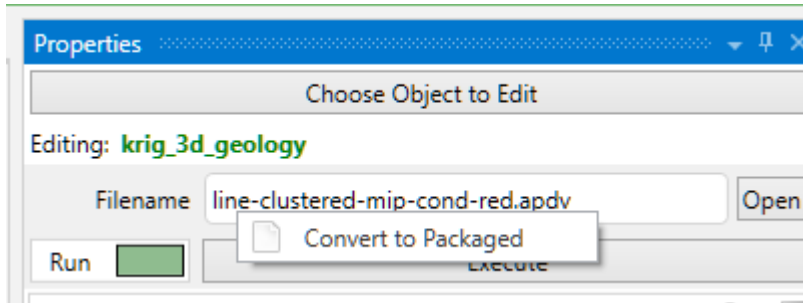
Generally you would not package data into an application during the early development of your project models. As we teach in our video tutorials we recommend that you frequently save your applications with a modified name (such as a serial number or -letter) so if you find you've gone down a wrong path you can go back to your last correct version.

I often find that it is best to work with a coarser resolution to keep compute times low and segregate my tasks depending on the scope of the project. Only once your work is nearing a final stage or you need to make interim deliverables to teammates would it make sense to package data with your application.

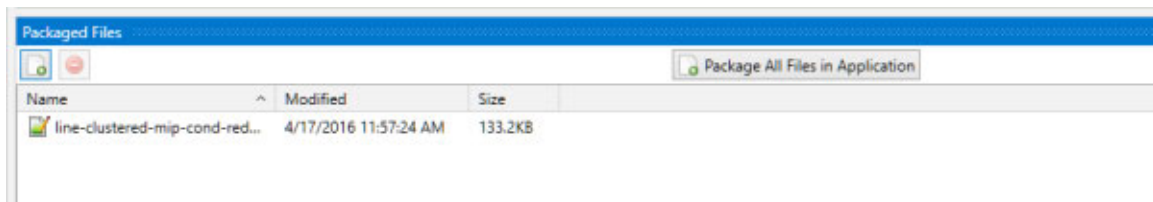
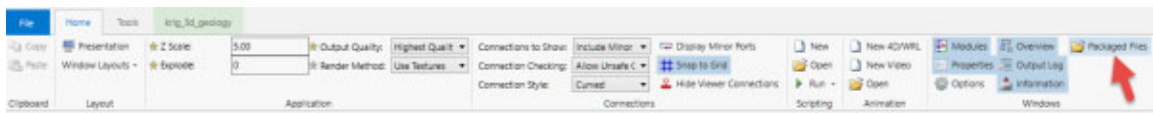
How to Package a Single File

Packaging a single file is very simple, but is seldom necessary since you will generally use the option to package all of the files in your application.

In every module with a file browser, you merely right-click on the filename, and a *Convert to Packaged* button will appear.

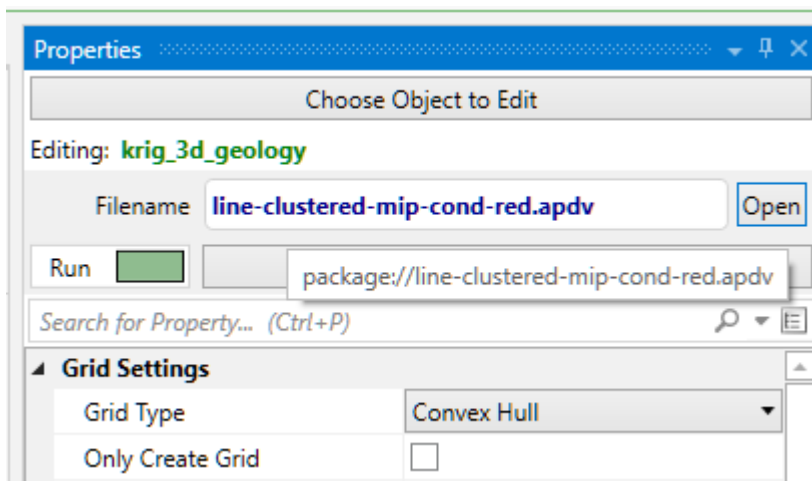


To view your Packaged Files, Click on the Home tab and in the Windows Group turn on *Packaged Files*



When a module is reading a packaged file, the file appears in the browser as **bold-blue text** with no *apparent* path. However, if you hover over the file, you will see the path as:

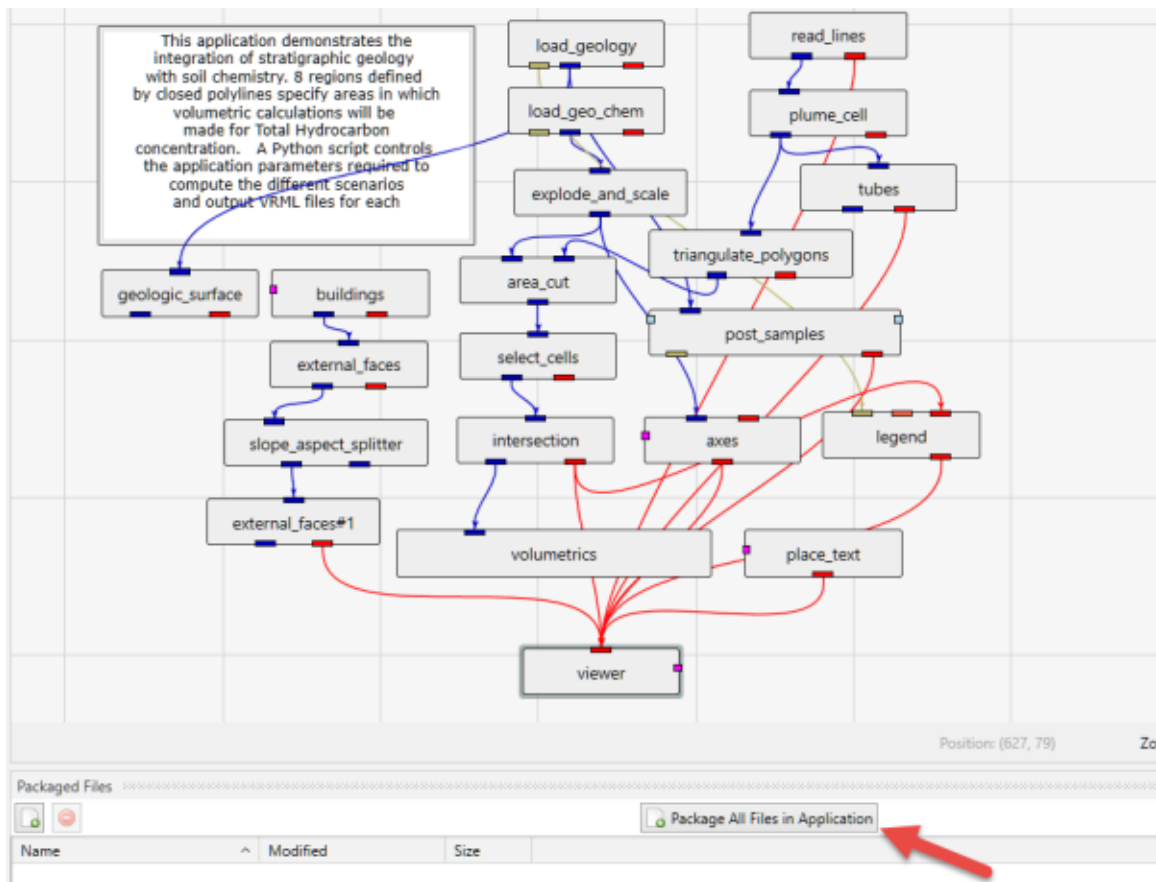
package://



Once one or all files are packaged in a .EVS application, when you save the application, the data files will be saved "in" the .EVS file.

Packaging All Files in an Application

To Package all files in an Application, open the Packaged Files window and merely press the *Package All Files in an Application* button.



For the application *railyard-volumetrics-efb-for-vrml.intermediate.evs*, the list of packaged files are:

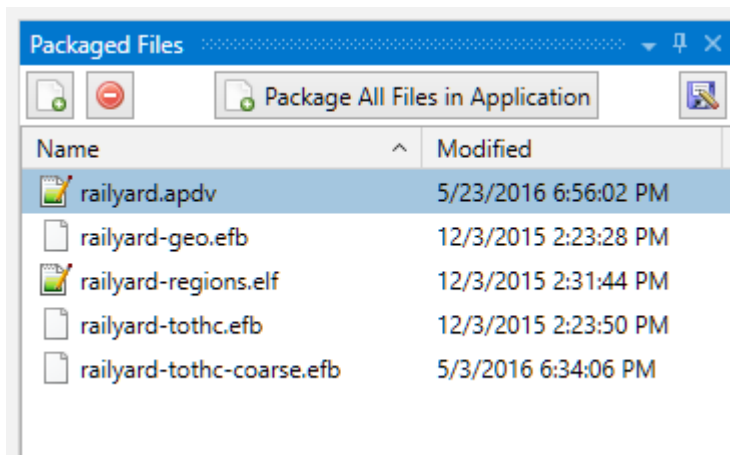
Packaged Files			Package All Files in Application	
Name	Modified	Size		
railyard.apdv	5/23/2016 6:56:02 PM	12.8KB		
railyard-geo.efb	12/3/2015 2:23:28 PM	186.8KB		
railyard-regions.elf	12/3/2015 2:31:44 PM	2.8KB		
railyard-tothc.efb	12/3/2015 2:23:50 PM	23.6MB		
railyard-tothc-coarse.efb	5/3/2016 6:34:06 PM	6.9MB		

Once one or all files are packaged in a .EVS application, when you save the application, the data files will be saved "in" the .EVS file.

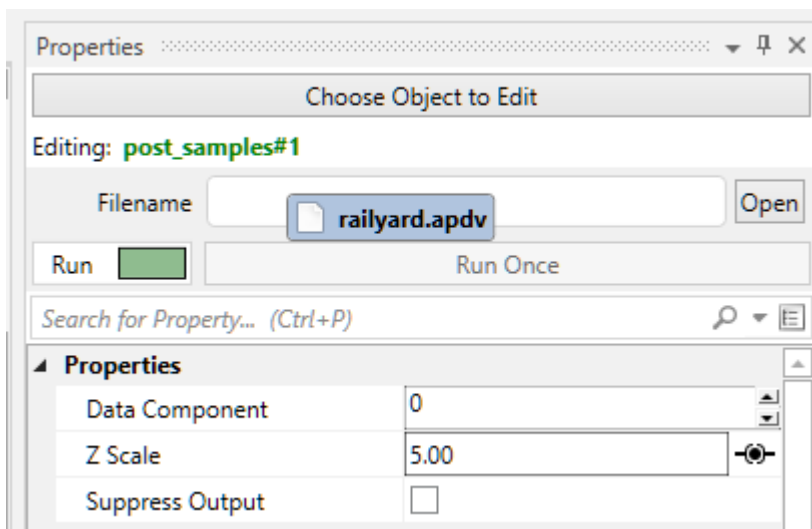
Note: The EVS modules requiring special treatment will be properly and automatically handled when using the *Package All Files in an Application* button.

How to Read a Packaged File in a New Module

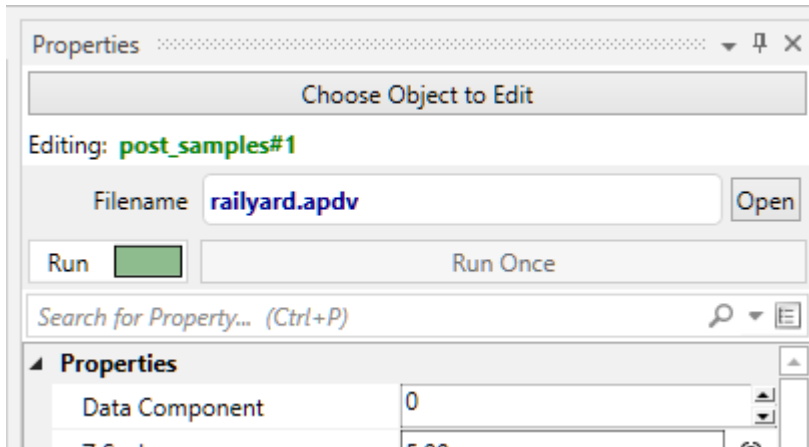
Using a file that has been added to an application's packaged data is easy, but is a bit different. The process involves selecting the file in the *Packaged Files* window with the left mouse and dragging and dropping it onto the file browser of the module where you wish to use it.



Below, we drop the railyard.apdv file from the *Packaged Files* onto the file browser of post_samples#1



When we drop (release) the file it appears in the browser as **bold-blue text** with no *apparent* path. However, if you hover over the file, you will see the path as: *package://*



Modules Requiring Special Packaging Treatment

Several EVS modules require special treatment in order to package their data. We won't go into all of the reasons why this is required for each module, but we will explain some of the reasons and advantages of the post-treatment application.

The affected modules are:

1. [read_vector_gis](#)
2. [read_cad](#)
3. [overlay_aerial](#)

The first two modules share a similar post-treatment application solution and benefit. For both, the advantages are similar and can be profound. Reading both shapefiles and CAD files can be quite slow, and the files tend to be large and inefficient.

read_vector_gis

The packaging process converts these files to binary EVS Field Files (.efb files) and requires replacing the original modules with a new load_evs_field module. In this simple application:

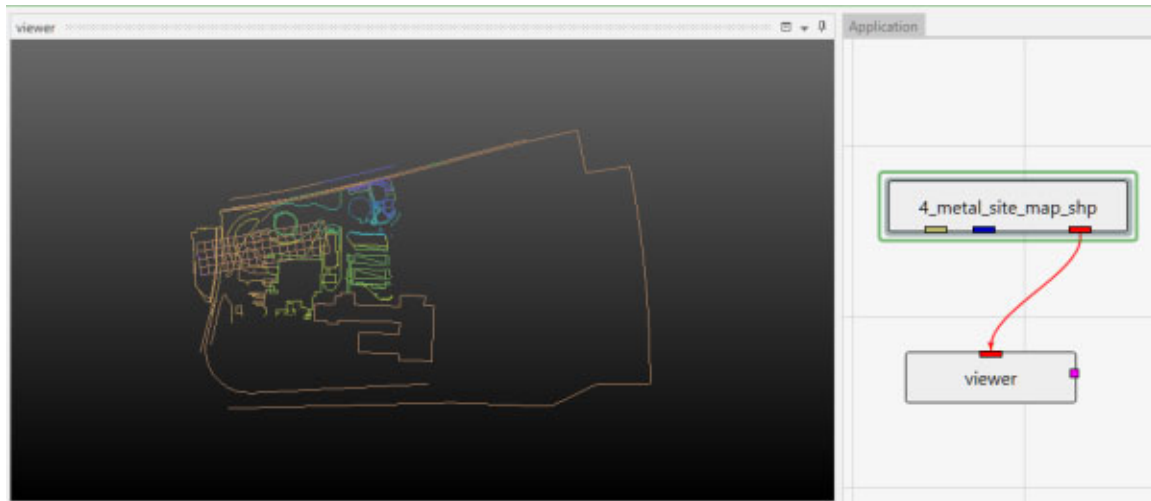


The shapefile actually consists of a set of 5 files which total 749 KB.

Name	Date modified	Type	Size
4_metal_site_map.dbf	04/20/2004 8:19 AM	DBF File	585 KB
4_metal_site_map.sbn	04/20/2004 8:19 AM	SBN File	7 KB
4_metal_site_map.sbx	04/20/2004 8:19 AM	Adobe Illustrator ...	1 KB
4_metal_site_map.shp	04/28/2004 2:30 AM	SHP File	152 KB
4_metal_site_map.shx	04/20/2004 8:19 AM	SHX File	6 KB

The read_vector_gis module, now has a "Convert to Packaged" button, which when we click on it does the following:

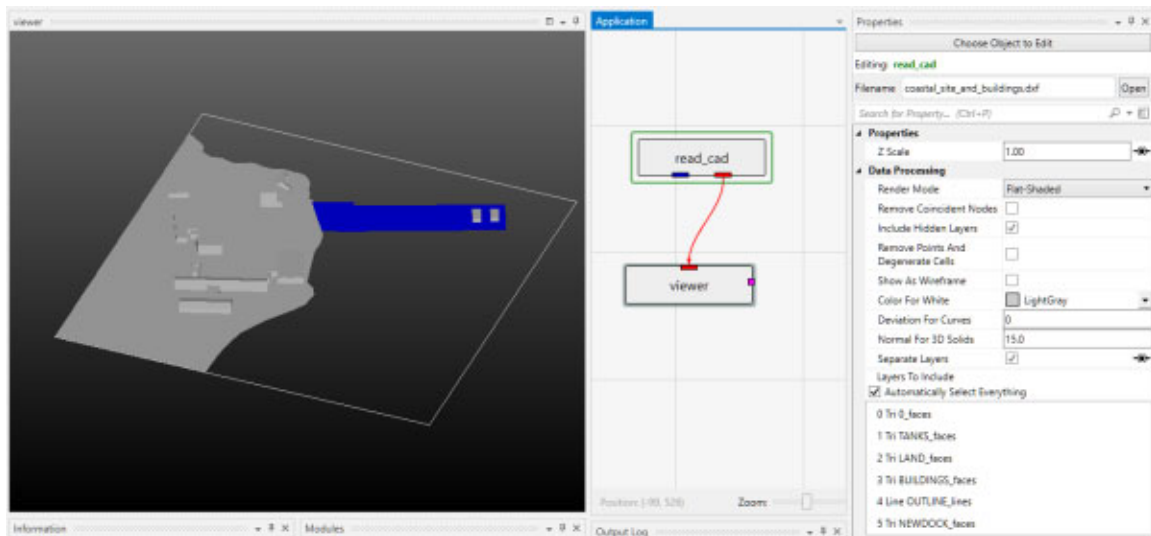
- Automatically replaces read_vector_gis with a load_evs_field module which is named based on the data file being read.
- Creates the efb file for you and adds it to your packaged data files
 - btw: it is half as big as the total shapefiles, and will load in less than 1/10th of the time
- Oh....that's all. What else is there!



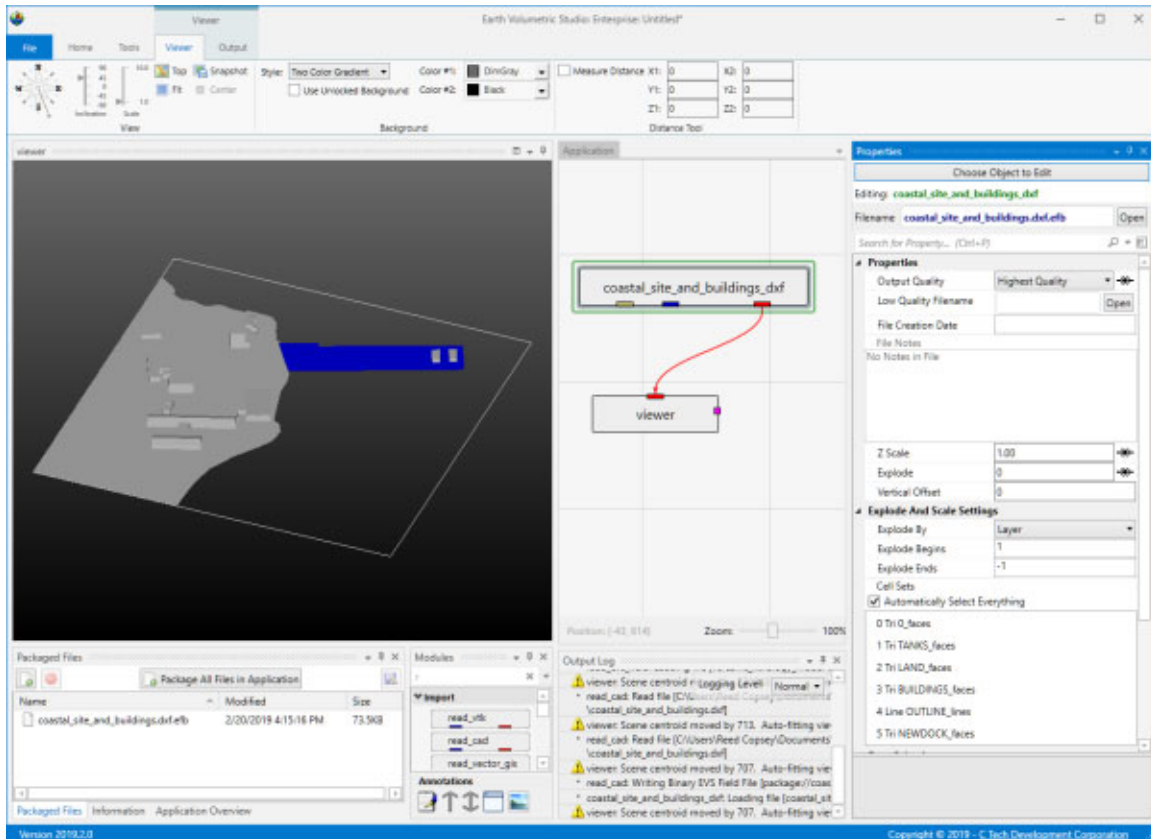
read_cad

When you convert `read_cad` to packaged, the module is replaced with a `load_evs_field` module which is named based on the data file being read, and the contents of the CAD file is converted to an EFB file automatically.

In the following application:



After packaging, the application is changed to:



overlay_aerial

The special treatment for `overlay_aerial` is quite different. Packaging is problematic when a module reads a file, but that process results in reading additional files (as with shapefiles). This also occurs with imagery when orthorectified images include an image file and a georeferencing file (e.g. world file, .gcp file, etc.).

To resolve this, `overlay_aerial`'s "Convert to Packaged" button creates a GeoTIFF file that is both cropped and matched to the specified resolution in `overlay_aerial`. This creates a new *single* image file which is generally dramatically smaller than the original files which were read. Your application is unchanged except that the Filename specified in `overlay_aerial` will reference the new packaged geotiff file created by this process.

Geostatistics Overview

When a volumetric model is created, we generally use geostatistics to estimate (interpolate and extrapolate) data into the volume based on sparse measurements. The algorithm used is called kriging, which is named after a South African statistician and mining engineer, Danie G. Krige who pioneered the field of geostatistics. Kriging is not only one of the best estimation methods, but it also is the only one that provides statistical measures of quality of the estimate.

The basic methodology in kriging is to predict the value of a function at a given point by computing a weighted average of the known values of the function in the

neighborhood of the point. The method is mathematically related to regression analysis. Both derive a best linear unbiased estimator, based on assumptions on covariances and make use of Gauss-Markov theorem to prove independence of the estimate and error.

The combination of kriging and volumetric modeling provides a much more feature rich model than is possible with any model that is limited to external surfaces and/or simpler estimation methods such as IDW or FastRBF. It allows us to perform volumetric subsetting operations and true volumetric analysis, and we can defend the quality of our models based on the limitations of our data.

In the coal mining industry, we can determine the quantity and quality of coal and its financial value. We can assess the amount and extraction cost of excavating overburden layers that must be removed or whether it is more cost effective to use tunneling to access the coal.

In the field of environmental engineering, where our software was born, volumetric modeling allows us to determine the spatial extent of the contamination at various levels as well as compute the mass of contaminant that is present in the soil, groundwater, water or air. During remediation efforts, this is critical, since we must confirm that the mass of contaminant being removed matches the reduction seen in the site, otherwise it is a clue that during the site assessment we have not found all the sources of contamination. This can result in remediation efforts which create contamination in some otherwise clean portions of the site.

The kriging algorithm provides us with only one direct statistical measure of quality, and that is [Standard Deviation](#). However, C Tech uses Standard Deviation to compute three additional metrics which are often more meaningful. These are:

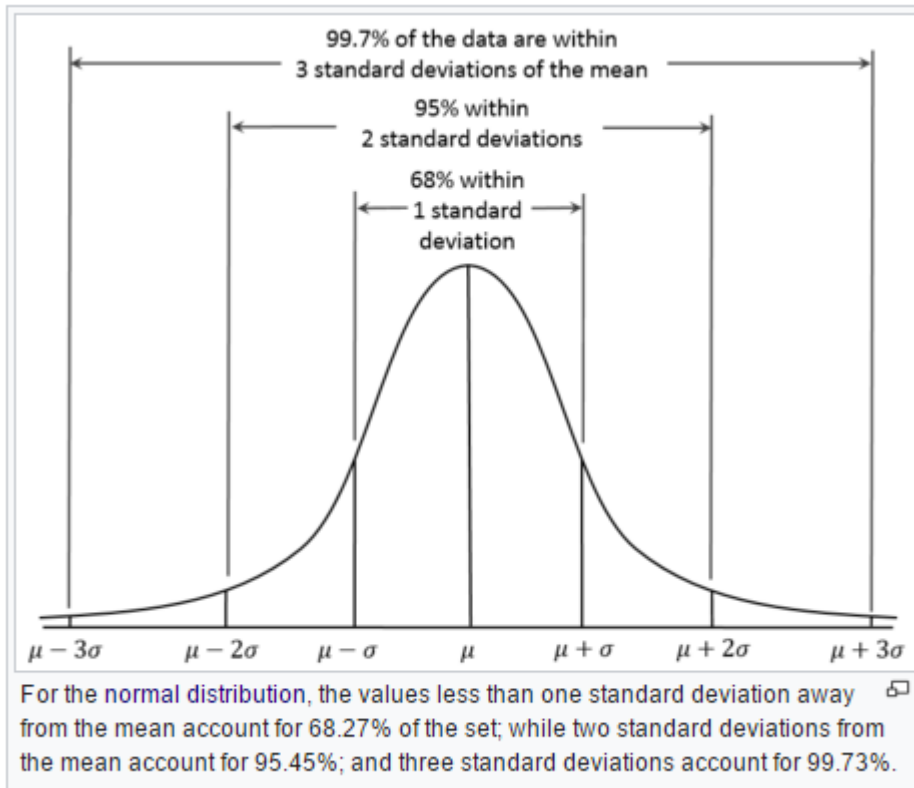
- [Confidence](#)
- [Uncertainty](#)
- [Min & Max \(Plume\) Estimate](#)

Standard Deviation

Inherent in the kriging process is the determination of the expected error or Standard Deviation at each estimated point. As we approach the location of our samples, the standard deviation will approach zero (0.0) since there should be no error or deviation at the measured locations.

The units of standard deviation are the same as the units of your estimated analyte.

The figure below shows why one standard deviation corresponds to 68% of the occurrences, whereas three sigma (standard deviations) covers 99.7%



At a particular node in our grid, if we predict a concentration of 50 mg/kg and have a standard deviation of 7 mg/kg, then we can say that we have a ~68% confidence that the actual value will fall between 43 and 57 mg/kg.

The computation of the expected Minimum and Maximum estimates for a given Confidence level is what our [Min/Max Estimate](#) provides.

Confidence

The Confidence values are the answer to a question, and the wording of the question depends on whether you are Log Processing your data or not.

- For the "Log Processing" case the question is: What is the "Confidence" that the predicted value will fall within a factor of "Statistical Confidence Factor" of the actual value?
- For the "Linear Processing" case, the question is: What is the "Confidence" that the predicted value will fall within a +/- tolerance "Statistical Confidence Tolerance" of the actual value?

Properties

Choose Object to Edit

Editing: **krig_3d**

Filename Open

Run Execute

Search for Property... (Ctrl+P)

- Properties
- Grid Settings
- Data Processing
- Krig Settings**

Estimation Type	Kriging
Reach	0
Points In Reach	20
Octant Search	<input type="checkbox"/>
Use All Points	<input checked="" type="checkbox"/>
All Points Limit	15000
Statistical Confidence Tolerance	1.00
Statistical Confidence Factor	2.00
Min Max Plume Confidence	80.0
IDW Power	2.00
Variogram And Anisotropy	
Anisotropy Mode:	Simple

So if your "Statistical Confidence Factor" is 2.0 as shown for a Log Processing case above, the question is:

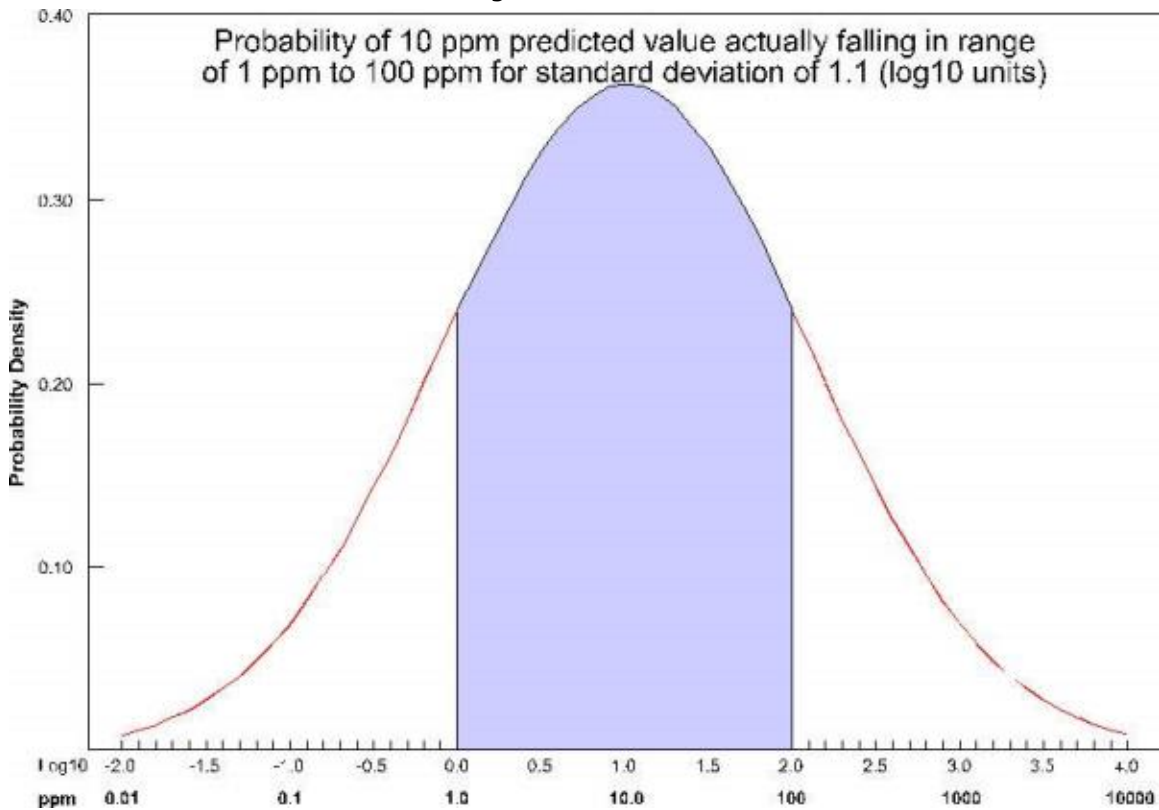
What is the "Confidence" that the predicted value will fall within a factor of 2.0 of the actual value?

The confidence is affected by your variogram and the quality of fit, but also by the range of data values and the local trends in the data where the Confidence estimate is being determined.

If your data spans several orders of magnitude, the confidences will be lower and if your data range is small the confidences will be higher depending also on the settings you use.

If the "Statistical Confidence Factor" were set to 10.0, because we are working on a log10 scale, EVS would take the log10 of the Statistical Confidence Factor (the value was 10, so the log is 1.0). It then compares the log concentration values and a corresponding standard deviation that was calculated for every node in our domain. For log concentrations, one unit is a factor of ten, therefore we are asking what is the probability that we will be within one unit. Above, where the Statistical Confidence Factor is 2.0, the questions would have been: What is the confidence that the predicted concentration will be within a factor of 2 of the actual concentration?

The actual calculation to determine confidence requires the standard deviation of the estimate at a node, and the Statistical Confidence Factor. The figure below shows the confidence (as the shaded area under the "bell" curve) for a Statistical Confidence Factor of 10 at a node where the predicted concentration was 10 ppm (1.0 as a log10 value) and the standard deviation for this point was 1.1 (in log10 units). For this example, the confidence would be ~64%, which means that 64% of the time, the value would lie in the shaded region.



Data Processing	
Data Processing	Linear Processing
Data Scaling	1.00
Pre Clip Min	0.000100
Pre Clip Max	1,000,000,000
Less Than Multiplier	0.100
Detection Limit	0
Post Clip Min	0.00100
Post Clip Max	1,000,000,000
Auto Exponentiate External Log Data	<input checked="" type="checkbox"/>
External Data	
<input type="button" value="All"/> <input type="button" value="Clear"/>	
(Sequence Contains No Elements)	
Krig Settings	
Estimation Type	Kriging
Reach	0
Points In Reach	20
Octant Search	<input type="checkbox"/>
Use All Points	<input checked="" type="checkbox"/>
All Points Limit	15000
Statistical Confidence Tolerance	0.0100
Statistical Confidence Factor	2.00
Min Max Plume Confidence	80.0

For example, consider the case where we are estimating soil porosity, and the input data values are ranging from 0.12 to 0.29. We would want to use "Linear Processing", and since our values fall within a tight range of numbers we might want to use a "Statistical Confidence Tolerance" that was 0.01. The confidence values we would compute would then be based upon the following question:

What is the "Confidence" that the predicted porosity value will be within 0.01 of the actual value?

If we were careless and used a "Statistical Confidence Tolerance" of 1.0 all of our confidences would be 100% since it would be impossible to predict any value that would be off by 1.0.

However, if we used a "Statistical Confidence Tolerance" of 0.0001, it is likely that our confidence values would drop off very quickly as we move away from the locations where measurements were taken.

Uncertainty

At first glance, confidence seems to be a reasonable measure of site assessment quality. If the confidence is high (and we are asking the right question), we can be

assured of the reasonableness of the predicted values. You might be tempted to collect samples everywhere that the confidence was low, and if you did, your site would be well characterized.

But, there is a better, more cost-effective way. Instead of focusing on every place where confidence was low, we could focus on only those locations where there was low confidence and where the predicted concentration was reasonably high. We make that easy by providing the Uncertainty.

In EVS, uncertainty is high where concentrations are predicted to be relatively high (above the Clip Min), but the confidence in that prediction is low. If the goal is to find the contamination, using uncertainty allows for more rapid, cost effective site assessment. Uncertainty is the core of our DrillGuide™ technology which performs successive analyses using the location of Maximum Uncertainty to select new locations for sampling on each analysis iteration.

NOTICE: Uncertainty values should be considered unitless and their magnitudes cannot directly be used to assess the quality of a site assessment. Please observe the following precautions:

- Use Uncertainty as it was intended, as a guide to locations needing additional characterization.
- Do not use Uncertainty values directly to assess the quality of a site assessment
- A 50% reduction in Uncertainty magnitude cannot be construed as a 50% improvement in site assessment.

Our training videos cover the use of DrillGuide and how to properly use and interpret Uncertainty.

Min-Max Estimate

Both krig_2d and krig_3d include the ability to compute the Minimum and Maximum Estimate, which is computed using the nominal estimates and standard deviations at every grid node based upon the user input *Min-Max Plume Confidence*.

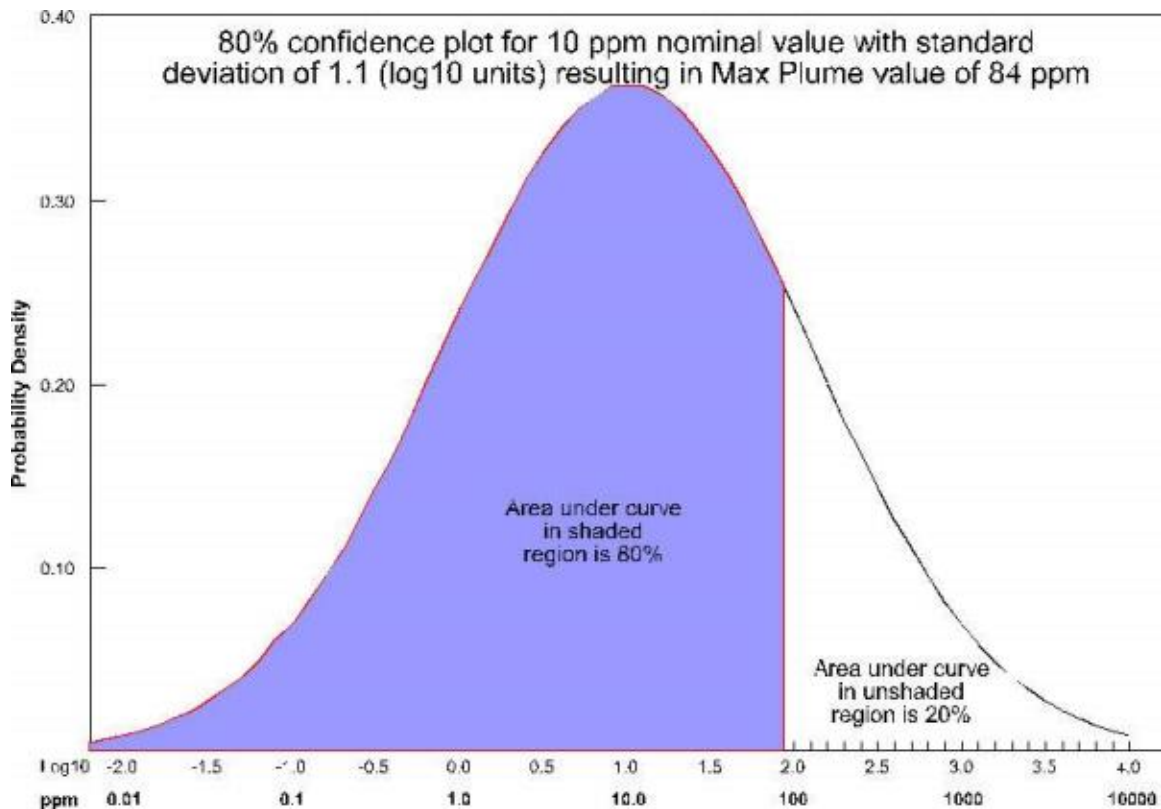
The **issue** with our MIN or MAX plumes is that they represent the statistical Min or Max at every point in the grid. It is quite unrealistic to believe that you could possibly have a **case** where you'd find the actual concentration would trend towards either the Min or Max at all locations.

[C Tech's Fast Geostatistical Realizations®](#) (FGR®) creates more plausible cases (realizations) which allow the Nominal concentrations to deviate from the peak of the bell curve (equal probability of being an under-prediction as an over-prediction) by the same user defined Confidence. However, FGR® allows the deviations to be both positive (max) and negative (min), and to fluctuate in a more realistic randomized manner.

Krig Settings	
Estimation Type	Kriging
Reach	0
Points In Reach	20
Octant Search	<input type="checkbox"/>
Use All Points	<input checked="" type="checkbox"/>
All Points Limit	15000
Statistical Confidence Tolerance	1.00
Statistical Confidence Factor	2.00
Min Max Plume Confidence	80.0
IDW Power	2.00
Variogram And Anisotropy	

For the case of Max Plume and 80% confidence, at each node, a maximum value is determined such that 80% of the time, the actual values will fall below the maximum value (for that nominal concentration and standard deviation). This process is shown below pictorially for the case of a nominal value of 10 ppm with a standard deviation of 1.1 (log units). For this case, the maximum value at that node would be ~84 ppm. This process is repeated for every node (tens or hundreds of thousands) in the model.

Note that for this plot, the entire left portion of the bell curve is shaded. If we were assessing the minimum value, it would be the right side. Statistically, we are asking a different type of question than when we calculate confidence for our nominal concentrations.



If this Confidence value were set to ~81% then we would be adding one standard deviation to the nominal estimate to create the Max and subtracting one standard deviation to create the Min. The higher you set the *Min-Max Plume Confidence* the greater the multiplier for standard deviations which are added/subtracted to create the Max/Min.

Even though Min & Max Estimates may not be realistic "realizations" of a likely site state, they still provide the best technique to determine when your site is adequately characterized. Some sites may have very complex contaminant distributions and high gradients while others may be very simple. Applying a single standard for sampling based on fixed spacing will never be optimal.

It is up to the regulators and property owners to determine the ultimate criteria, but generally having the ability to assess the variation in the expected plume volume and the corresponding variation in analyte mass within, provides the best metric for assessing when a site has been sufficiently characterized.

Visualization Fundamentals

- **Data Content Requirements**
- **Direct Data Visualization**
- **Gridding and Dimensionality**
- **Rectilinear Grids**
- **Convex Hull**
- **Triangular Networks**
- **Estimation Methods**
- **Surfaces**
- **Color**
- **Model Subsetting**

Data Content Requirements

As defined above, our discussion of environmental data will be limited to data that includes spatial information. When spatial data is collected with a GPS (Global Positioning Satellite) system, the spatial information is often represented in latitude and longitude (Lat-Lon). Generally, before this data is visualized or combined with other data, it is converted to a Cartesian coordinate system. The process of converting from Lat-Lon to other coordinate systems is called projection. Many different projections and coordinate systems can be used. The single most important thing is maintaining consistency. Projecting this data is especially necessary for three-dimensional visualization because we want to maintain consistent units for x, y, and z coordinates. Latitude and longitude angle units (degrees, minutes and seconds) do not represent equal lengths and there is no equivalent unit for depth. Projections convert the angles into consistent units of feet or meters.

analyte (e.g. chemistry)

analyte (e.g. chemistry) data files must contain the spatial information (x, y, and optional z coordinates) as well as the measured analytical data. The file should specify the name of the analyte and should include information about the detection limits of the measured parameter. The detection limit is necessary because samples where the analyte was not detected are often reported as zero or "nd". It is generally not adequate (especially when logarithmically processing this data) to merely use a value of 0.0.

If we want to be able to create a graphical representation of the borings or wells from which the samples were taken, the analyte (e.g. chemistry) data file should also include the boring or well name associated with each sample and the ground surface elevation at the location of that boring.

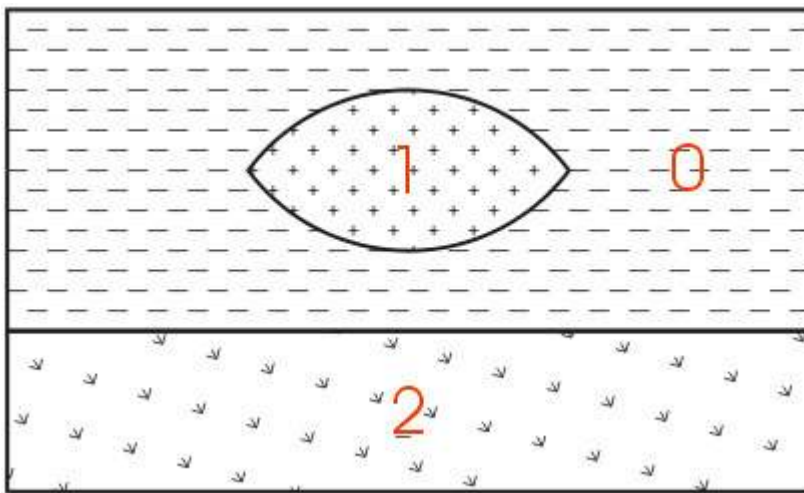
[The chapter on analyte \(e.g. chemistry\) Data Files](#) includes an in-depth look at the format used by C Tech Development Corporation's Environmental Visualization System (EVS).

Geology

Geologic information is considerably more difficult to represent in a single, unified data format because of its nature and complexity. Geologic data files can be grouped into one of two classes, those representing interpreted geology and those representing boring logs. By some definitions, boring logs are interpreted since a geologist was required to assign materials based on core samples or some other

quantitative measurements. However, for this discussion interpreted geology data will be defined as data organized into a geologic hierarchy.

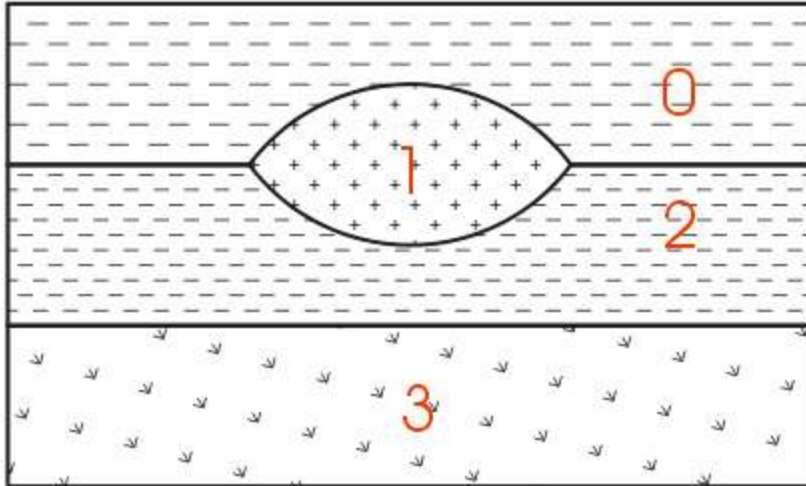
C Tech's software utilizes one of two different ASCII file formats for interpreted geologic information. These two file formats both describe points on each geologic surface (ground surface and bottom of each geologic layer), based on the assumption of a geologic hierarchy. Simply stated, geologic hierarchy requires that all geologic layers throughout the domain be ordered from top to bottom and that a consistent hierarchy be used for all borings. At first, it may not seem possible for a uniform layer hierarchy to be applicable for all borings. Layers often pinch out or exist only as localized lenses. Also layers may be continuous in one portion of the domain, but are split by another layer in other portions of the domain. However, all of these scenarios and many others can be usually be modeled using a hierarchical approach.



The easiest way to describe geologic hierarchy is with an example. Consider the example above of a clay lens in sand with gravel below.

Imagine borings on the left and right sides of the domain and one in the center. Those outside the center would not detect the clay lens. On the sides, it appears that there are only two layers in the hierarchy, but in the middle there are three materials and four layers.

EVS's & MVS's hierarchical geologic modeling approach accommodates the clay lens by treating every layer as a sedimentary layer. Because we can accommodate "pinching out" layers (making the thickness of layers ZERO) we are able to produce most geologic structures with this approach. Geologic layer hierarchy requires that we treat this domain as 4 geologic layers. These layers would be Upper Sand (0), Clay (1), Lower Sand (2) and Gravel (3).



If desired, both Upper and Lower Sand can have identical colors or hatching patterns in the final output.

Figure 0.1 Geologic Hierarchy of Clay Lens in Sand

When this geologic model is visualized in 3D, both Upper and Lower Sand can have identical colors or hatching patterns. Since the layers will fit together seamlessly, dividing a layer will not change the overall appearance (except when layers are exploded).

For sites that can be described using the above method, it is generally the best approach for building a 3D geologic model. Each layer has smooth boundaries and the layers (by nature of hierarchy) can be exploded apart to reveal the individual layer surface features. An example of a much more complex site is shown below in Figure 1.3. Sedimentary layers and lenses are modeled within the confines of a geologic hierarchy.

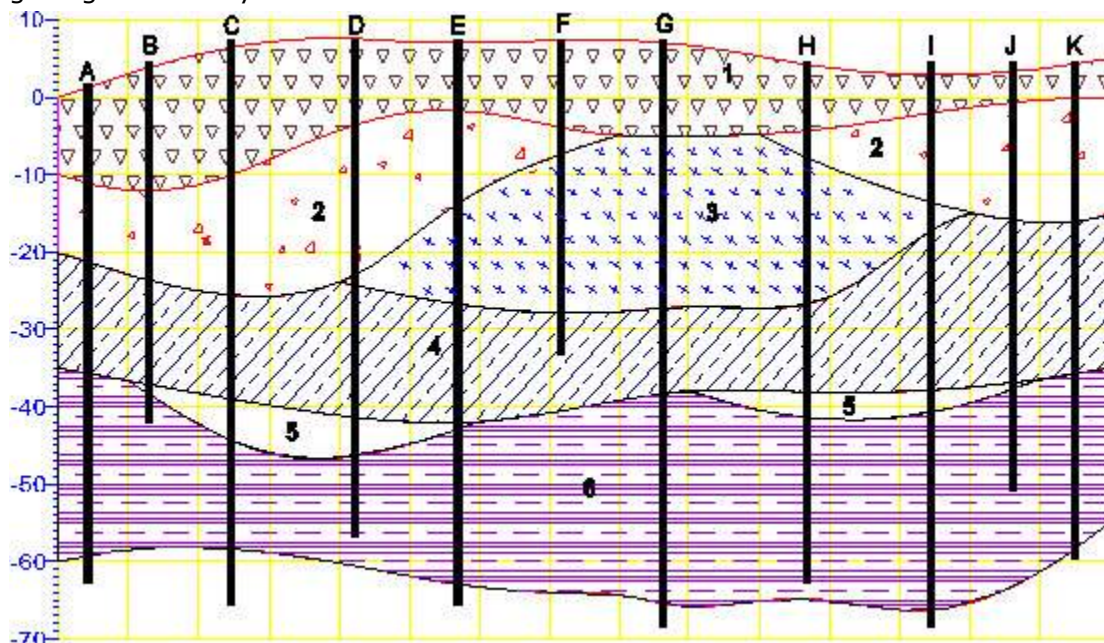


Figure 0.2 Complex Geologic Hierarchy

The hierarchical borehole based geology file format used for Figure 1.3 is described in the chapter on [Borehole Geology Files](#).

With C Tech's EVS software, there are two other geology file formats. One of them is a more generic format for interpreted (hierarchical) geologic information. With that format; x, y, and z coordinates are given for each surface in the model. There is no requirement for the points on each surface to have coincident x-y coordinates or for each surface to be defined with the same number of points. The borehole geology file format described above could always be represented with this more generic file format.

The last file format is used to represent the materials observed in each boring. Borings are not required to be vertical, nor is there any requirement on the operator to determine a geologic hierarchy. C Tech refers to this file format as Pregeology referring to the fact that it is used to represent raw 3D boring logs. This format is also considered to be "uninterpreted". This is not meant to imply that no form of geologic evaluation or interpretation has occurred. On the contrary, it is required that someone categorizes the materials on the site and in each boring.

In C Tech's EVS software, the raw boring data can be used to create complex geologic models directly using a process called Geologic Indicator Kriging (GIK). The GIK process begins by creating a high-resolution grid constrained by ground surface and a constant elevation floor or some other meaningful geologic surface such as rockhead. For each cell in the grid, the most probable geologic material is chosen using the surrounding nearby borings. Cells of common material are grouped together to provide visibility and rendering control over each material.

[The pregeology file format is discussed in this chapter.](#)

Direct Data Visualization

Many methods of environmental data visualization require mapping (interpolation and/or extrapolation) of sparse measured data onto some type of grid. Whenever this is done, the visualization includes assumptions and uncertainties introduced by both the gridding and interpolation processes. For these reasons, it is crucial to incorporate direct visualization of the data as a part of the entire process. It becomes the operator's responsibility to ensure that the gridding and interpolation methods accurately represent the underlying data.

A common means for directly visualizing environmental data is to use glyphs. A "glyph" refers to a graphical object that is used as a symbol to represent an object or some measured data. For the purposes of this paper, glyphs will be positioned properly in space and may be colored and/or sized according to some data value. For a graphics display, the simplest of all glyphs would be a single pixel. A pixel is a dot that is drawn on the computer screen or rendered to a raster image. The issue of pixel size often creates confusion. Pixels (by definition) do not have a specific size. Their apparent size depends on the display (or printer) characteristics. On a computer screen, the displayed size of a pixel can be determined by dividing the screen width in inches or millimeters by the screen resolution in pixels. For example, a 19" computer monitor has a screen width of about 14.5 inches. If the "Desktop Area" is set to 1280 by 1024, the width of a pixel would be approximately 0.011 inches (~0.29 mm). If the "Desktop Area" were reduced, the apparent size of a pixel would increase.

There are virtually no limits to the type of glyph objects that may be used. Glyphs can be simple geometric objects (e.g. triangles, spheres, and cubes) or they can be representations of real-world objects like people, trees or animals.

Glyphs in 2D

For two-dimensional displays we generally use glyph objects which are two-dimensional (having no depth or z-coordinate information). Figure 1.4 is an example of such a display.

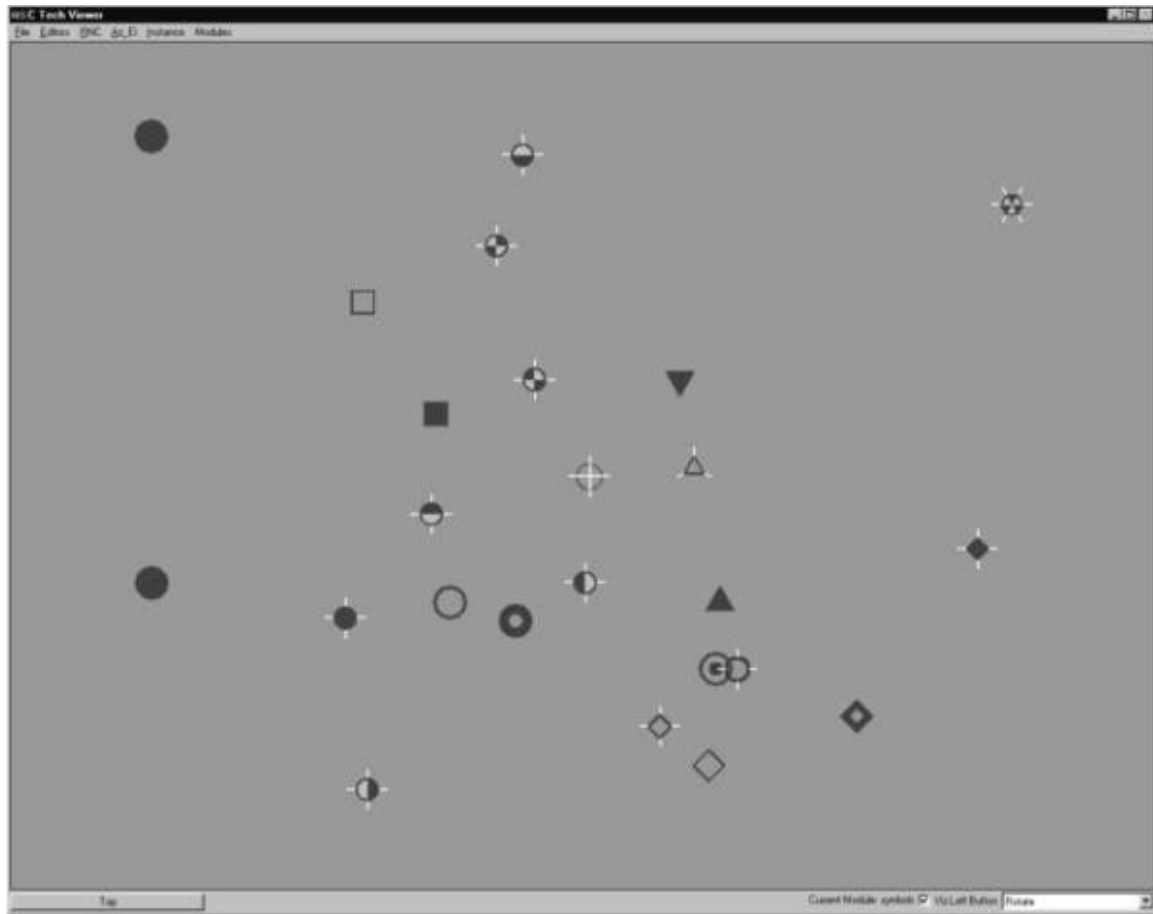


Figure 0.3 Two-Dimensional Glyphs

Glyphs in 3D

It is once we move to the three-dimensional world that glyphs become much more interesting. In Figure 1.5, cubes (hexahedron elements) are positioned, sized and colored to represent chemical measurements made in soil at a railroad yard in Sacramento, California. Axes were added to provide coordinate references and this picture was rendered with perspective effects turned on. This results in a visualization where parallel lines do not remain parallel and objects in the foreground appear larger than those in the background.

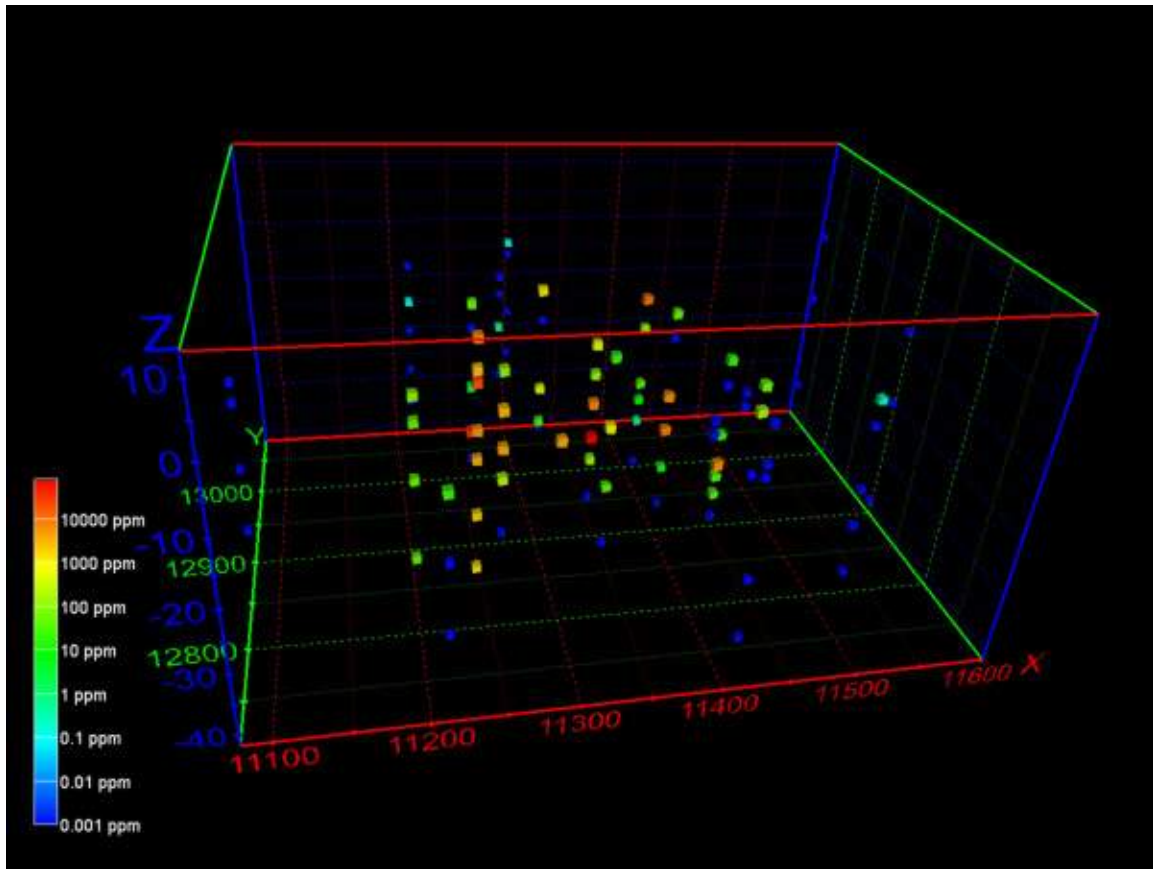


Figure 0.4 Three-Dimensional Cubic Glyphs

When representations of the borings are added, the figure becomes much more useful. Figure 1.6 shows the sample represented by colored spheres and tubes represent the borings. The tubes are colored alternating dark and light gray where the color changes on ten-foot intervals. This provides a reference to allow the viewer to quickly determine the approximate depth of the samples. The borings are also labeled with their designation. These last two figures both represent the same data, however it is clear which one provides the most useful information.

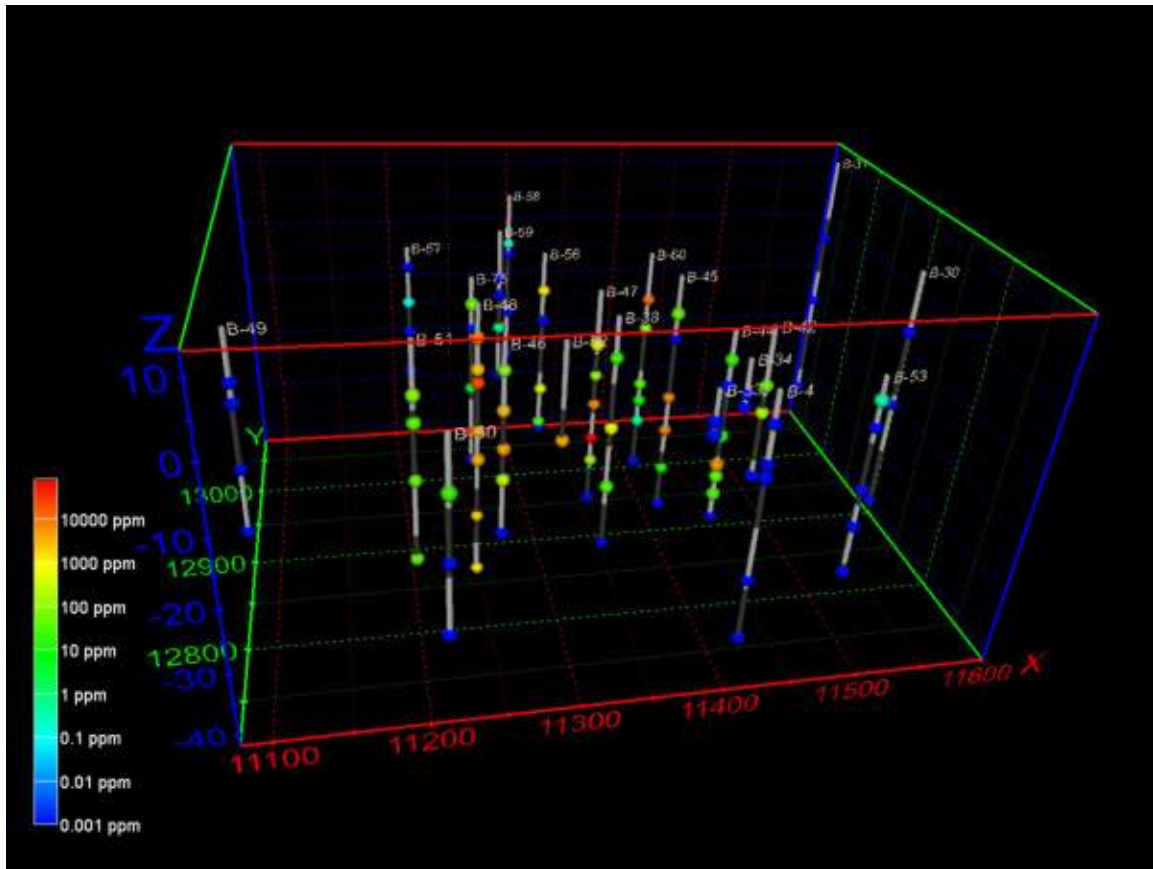


Figure 0.5 Three-Dimensional Glyphs with Boring Tubes

Glyphs can also be used to represent vector data. The most commonly encountered vector data represents ground water flow velocity. In this case, the glyph is not only colored and sized according to the magnitude of the velocity vector, but the glyph can also be oriented to point in the vector's direction. For this type of application, an assymetric glyph (as opposed to a sphere or cube) is used. Figure 1.7 uses a glyph that is referred to as "jet". It is an elongated tetrahedron that points in the direction of the vector. The data represented in this figure is predicted velocities output from a MODFLOW simulation to predict the groundwater flow field resulting from the dewatering of a gold mine pit.

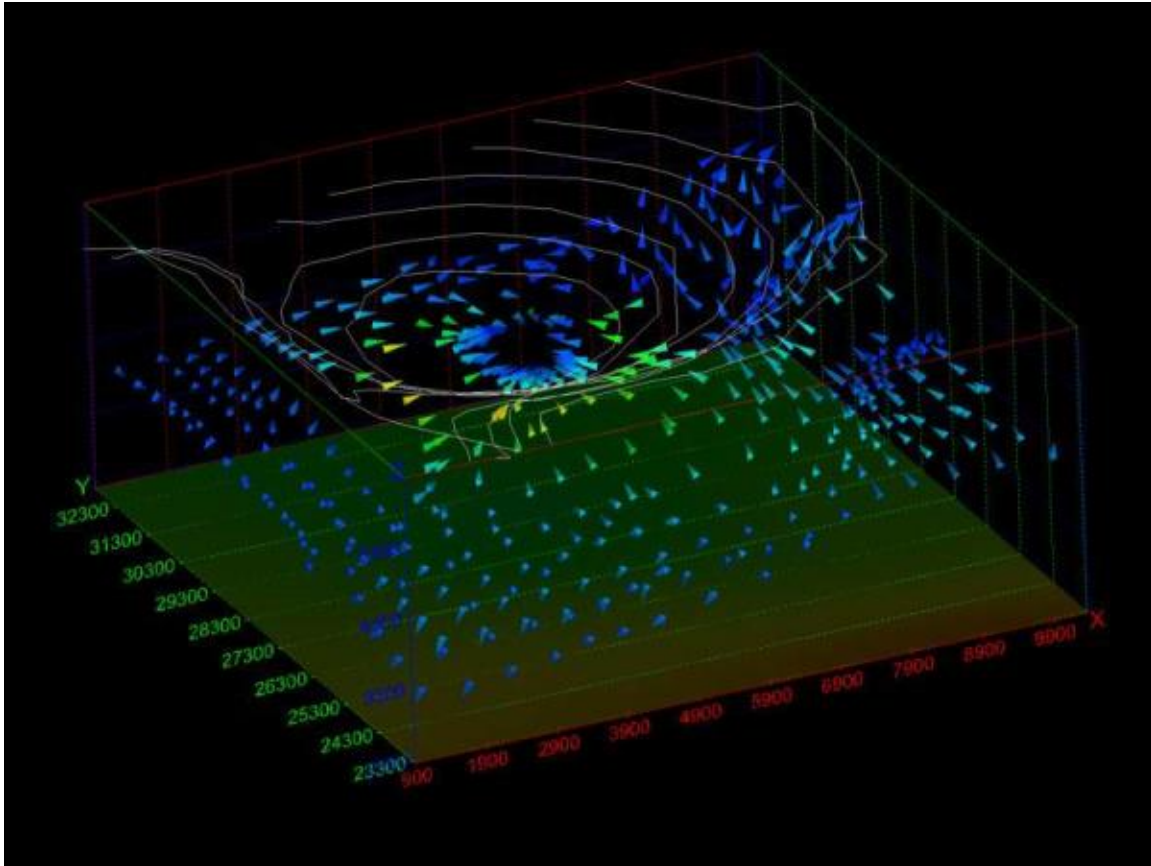


Figure 0.6 Three-Dimensional Glyphs Representing Vector Data

Gridding and Dimensionality

Although there is great value in directly visualizing measured data; it does have many limitations. Without mapping sparse measured data to a grid, computation of contaminant areas or volumes is not possible. Further, the techniques available for visualizing the data are very limited. For these reasons and more, significant attention should be paid to the process of creating a grid into which the data will be interpolated and extrapolated.

For this paper, a grid is defined as a collection of nodes and cells. Nodes are points in two or three-dimensions with coordinates and usually one or more data values. The word "cell" and "element" are both used as a generic term to refer to geometric objects. The cell type and the nodes that comprise their vertices define these objects. Commonly used cell types are described in Table 1.1 and Figure 1.8.

Cell Type	Number of Nodes	Dimensionality
Point	1	0
Line	2	1
Triangle	3	2
Quadrilateral	4	2
Tetrahedron	4	3
Pyramid	5	3

Prism	6	3
Hexahedron	8	3

Table 0.1 Common Cell Types

Dimensionality refers to the space occupied by the cell. Points have do not have length, width, or height, therefore their dimensionality is zero (0). Lines are dimensionality "1" because they have length. Dimensionality 2 objects such as quadrilaterals (quad) and triangles have area and dimensionality 3 objects ranging from tetrahedrons (tet) to hexahedrons (hex) are volumetric. When creating a two-dimensional grid, areal cells are used and for three-dimensional grids, volumetric cells are used.

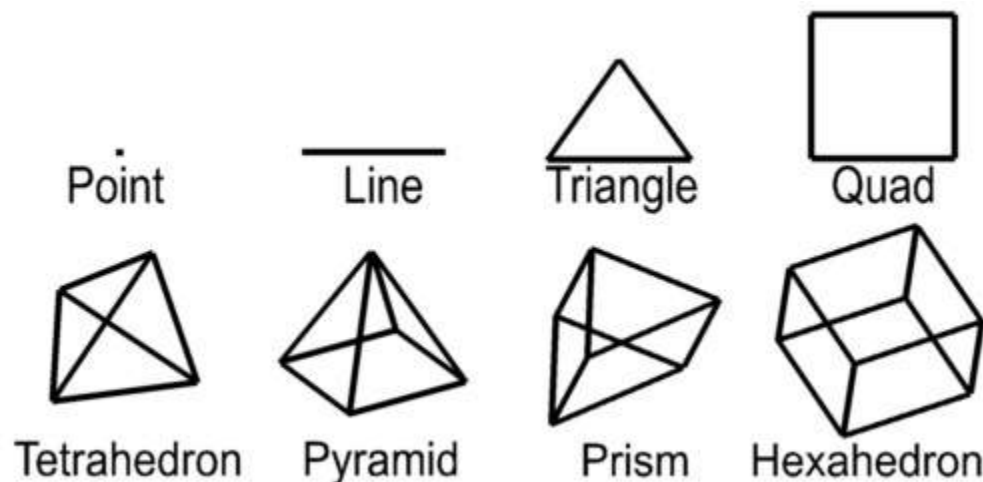


Figure 0.7 Common Cell Types

Rectilinear Grids

Rectilinear (a.k.a. uniform) grids are among the simplest type of grid. The grid axes are parallel to the coordinate axes and the cells are always rectangular in cross-section. The positions of all the nodes can be computed knowing only the coordinate extents of the grid (minimum and maximum x, y and optionally z). Two-dimensional rectilinear grids are comprised of quadrilateral cells. For a 2D grid with i nodes in the x direction and j nodes in the y direction, there will be a total of $(i - 1) * (j - 1)$ cells.

The connectivity of the cells (the nodes that define each cell) can be implicitly determined because the nodes and cells are numbered in an orderly fashion. The advantages of rectilinear grids include the ease of creating them and the uniformity of cell area in 2D and cell volume in 3D. The disadvantages are that grid nodes are generally not coincident with the sample data locations and large areas of the grid may fall outside of the bounds of the data. A simple two-dimensional rectilinear grid is shown in Figure 1.9.

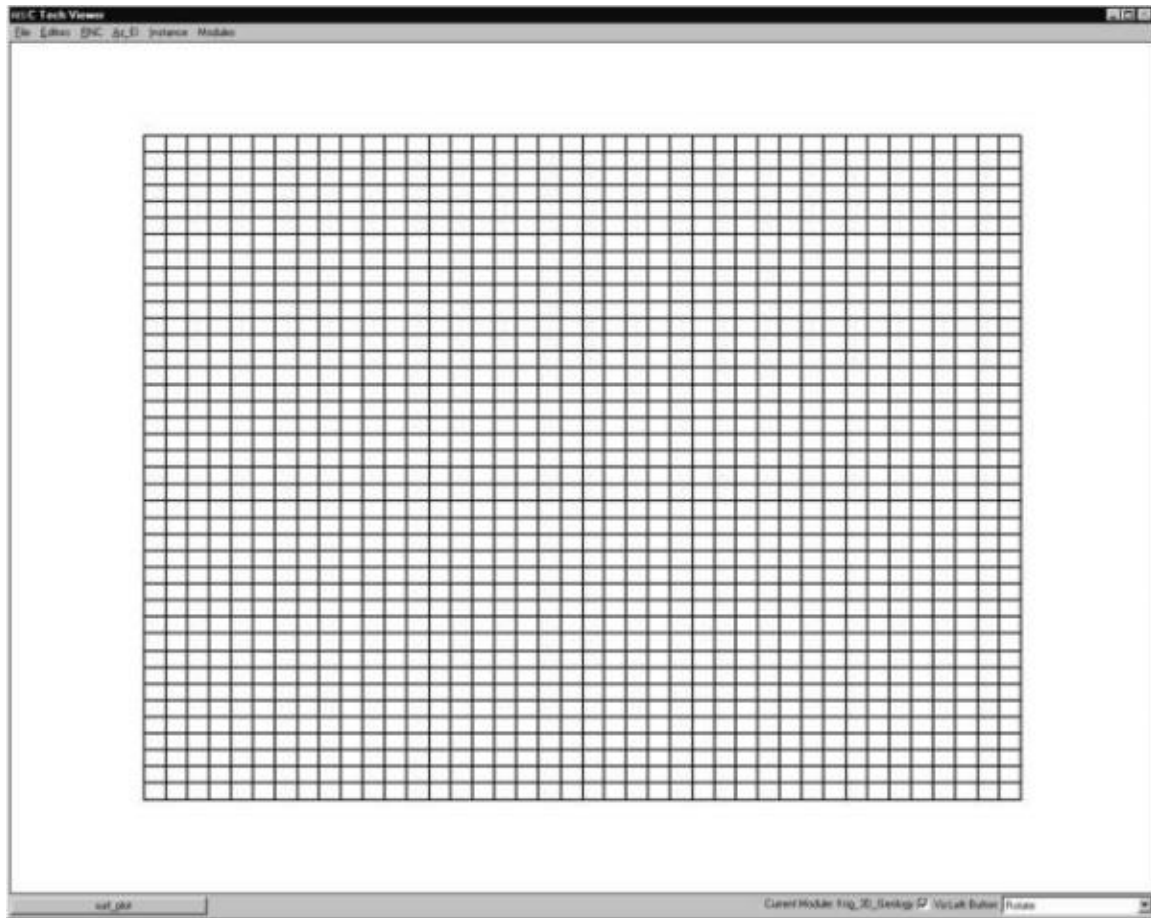


Figure 0.8 Two-Dimensional Rectilinear Grid

Three-dimensional rectilinear grids offer the simplest method for gridding a volume. They are constrained to rectangular parallel piped volumes and have hexahedral cells of constant size. (See Figure 1.10) For some processes and visualization techniques such as volume rendering, this is advantageous and may even be required. For a grid having i by j by k nodes there will be $(i-1) * (j-1) * (k-1)$ hexahedron cells whose connectivity can be implicitly derived.

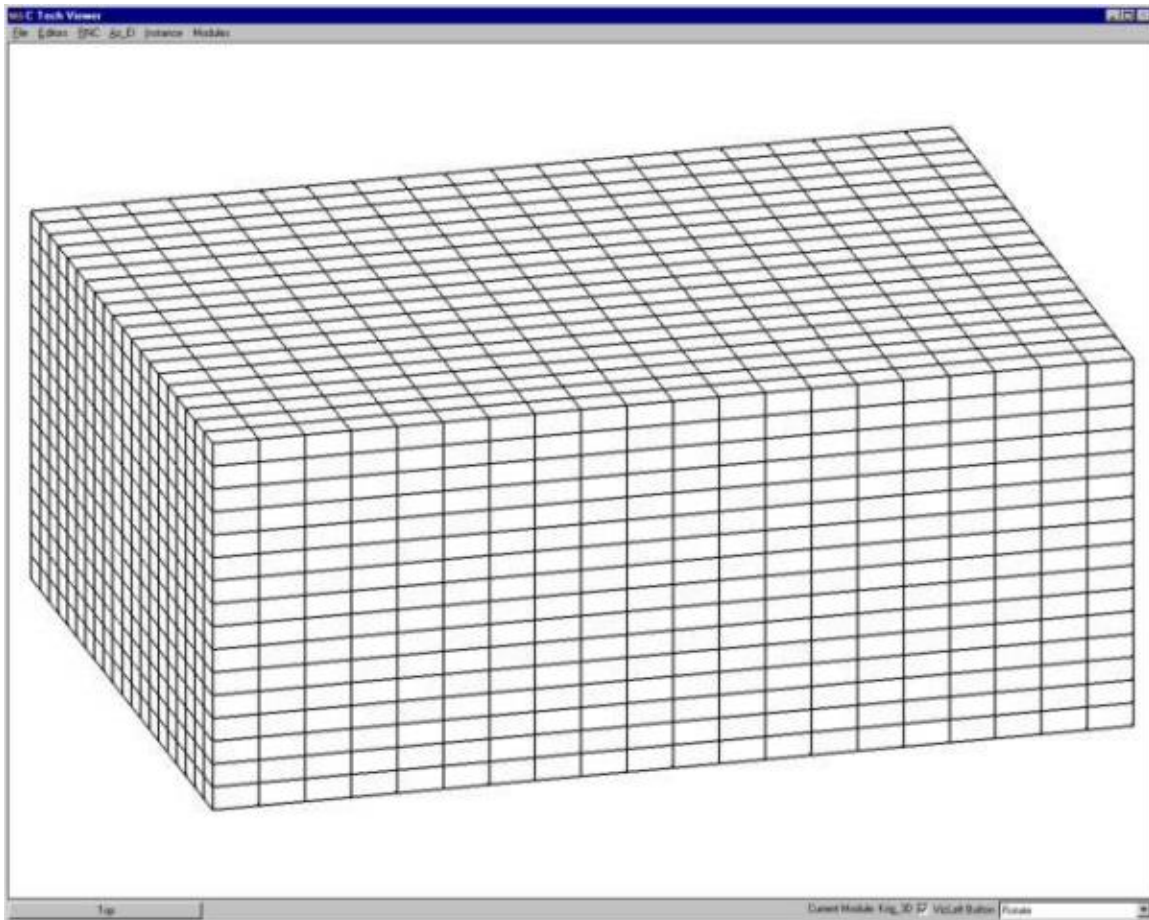


Figure 0.9 Three-Dimensional Rectilinear Grid

Finite Difference

The following type of grid derives its name from the numerical methods that it employs. Simulation software such as the USGS's MODFLOW utilizes a finite difference numerical method to solve equilibrium and transient ground water flow problems. This solution method requires a grid that contains only rectangular cells. However the cells need not be uniform in size. For two-dimensional grids, this results in rectangular cells, however it is possible that no two cells are precisely the same size. Some simulation software requires that finite difference grids be aligned with the coordinate axes. EVS does not impose this restriction, but it does provide a means to export the grid transformed so that the grid axes are aligned. Figure 1.11 shows a rotated 2D finite difference grid. Smaller cells are concentrated in areas of the model where there are significant gradients in the data. For groundwater simulations this is usually where wells are located. For environmental contamination it should be the location of spills or areas where DNAPL (dense non-aqueous phase liquids) contaminant plumes were detected. The smaller cells provide greater accuracy in estimating the parameter(s) of interest.

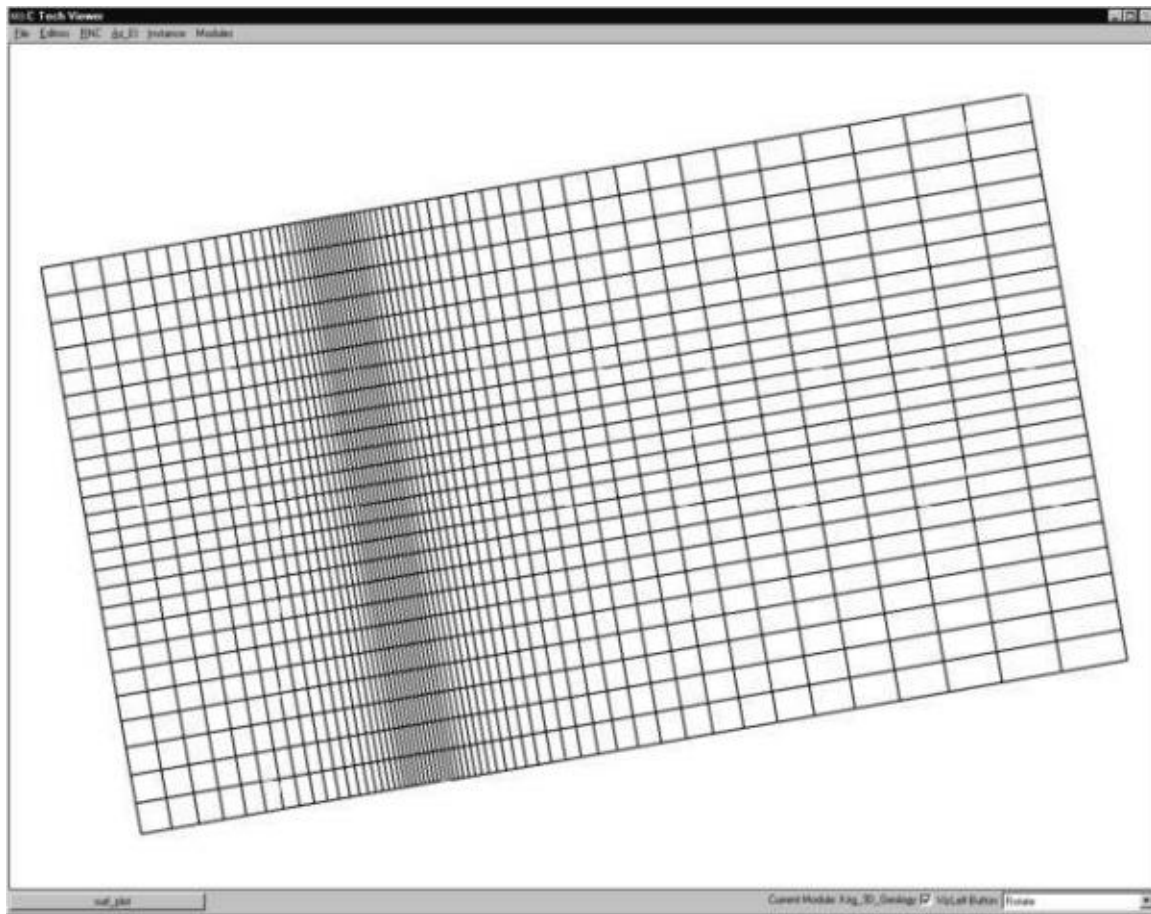


Figure 0.10 Two-Dimensional Rotated Finite Difference Grid

Three-dimensional finite difference grids have the same restrictions as 2D grids with respect to their x and y coordinates (cell width and length). However, the z coordinates of the grid (which define the cell thicknesses) are allowed to vary arbitrarily. This allows for creation of a grid that follows the contours of geologic surfaces. For a grid having i by j by k nodes there will be $(i-1) * (j-1) * (k-1)$ hexahedron cells whose connectivity can be implicitly derived. However the coordinates of the nodes for this grid must be explicitly specified. Figure 1.12 shows the grid created to model the migration of a contaminant plume in a tidal basin.

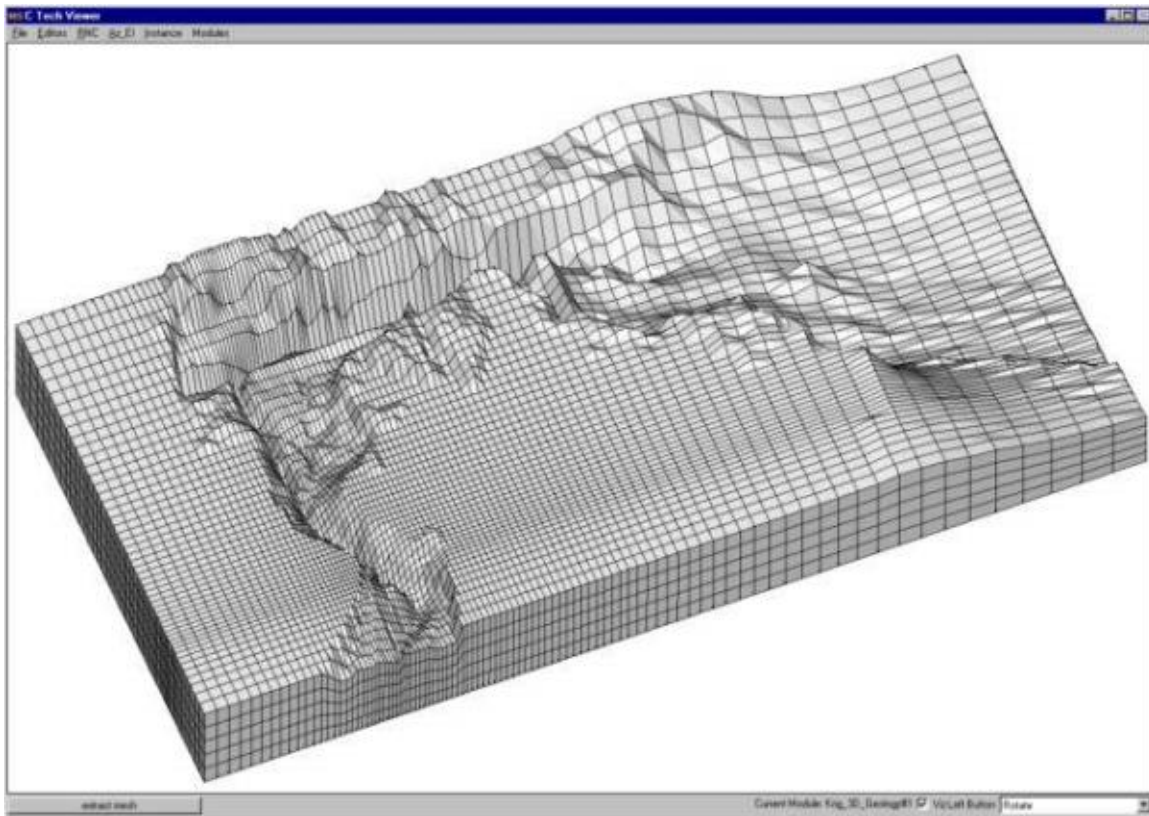


Figure 0.11 Three-Dimensional Finite Difference Grid

Convex Hull

The convex hull of a set of points in two-dimensional space is the smallest convex area containing the set. In the x-y plane, the convex hull can be visualized as the shape assumed by a rubber band that has been stretched around the set and released to conform as closely as possible to it. The area defined by the convex hull offers significant advantages. Within the convex hull all parameter estimates are interpolations. The convex hull best fits the spatial extent of the data. Remember that the convex hull defines an area. That area can be gridded in many ways. EVS grids convex hull regions with quadrilaterals. Smoothing techniques are used to create a grid that has reasonably equal area cells. A two-dimensional example of a convex hull grid is shown in Figure 1.13. In this example, the domain of the model was offset by a constant amount from the theoretical convex hull. This results in rounded corners and a model region that is larger than the convex hull.

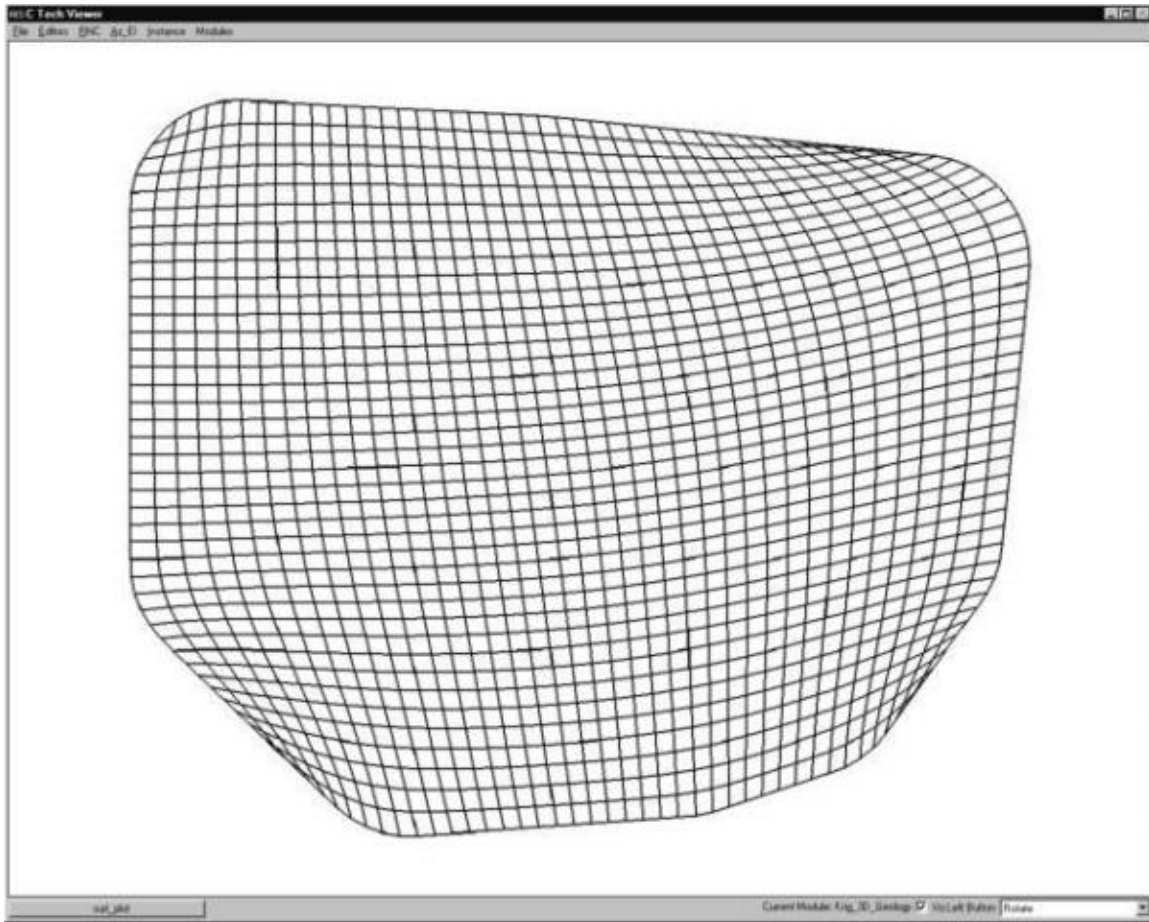


Figure 0.12 Convex Hull Grid with Offset

Adaptive Gridding

Adaptive gridding is the localized refinement of a grid to provide higher resolution in the areas or volumes surrounding measured sample data. Adaptive gridding or grid refinement can be accomplished in many different ways. In EVS, rectilinear, finite difference and convex hull grids can all be refined using a similar method. In two-dimensions a new node is placed precisely at the measured sample data location. Three additional nodes are placed to create a small quadrilateral cell within the cell to be refined. The corners of the small cell are connected to the corresponding corners of the cell being refined creating a total of five cells where the one previously was. The resulting nodal locations and grid connectivity must be explicitly defined.

Adaptively gridding offers many advantages. It assures that there will always be nodes at the precise coordinates of the sample data. This insures that the data minimum and maximum in the gridded model will match the sample data. It also provides greater fidelity in defining data trends in regions with high gradients. Figure 1.14 shows a two-dimensional adaptively gridded convex hull model. This model's area was also offset from the convex hull. Since each sample data point results in a refined region, and the sample points define the convex hull, the regions in each corner of the model contain adaptively gridded cells.

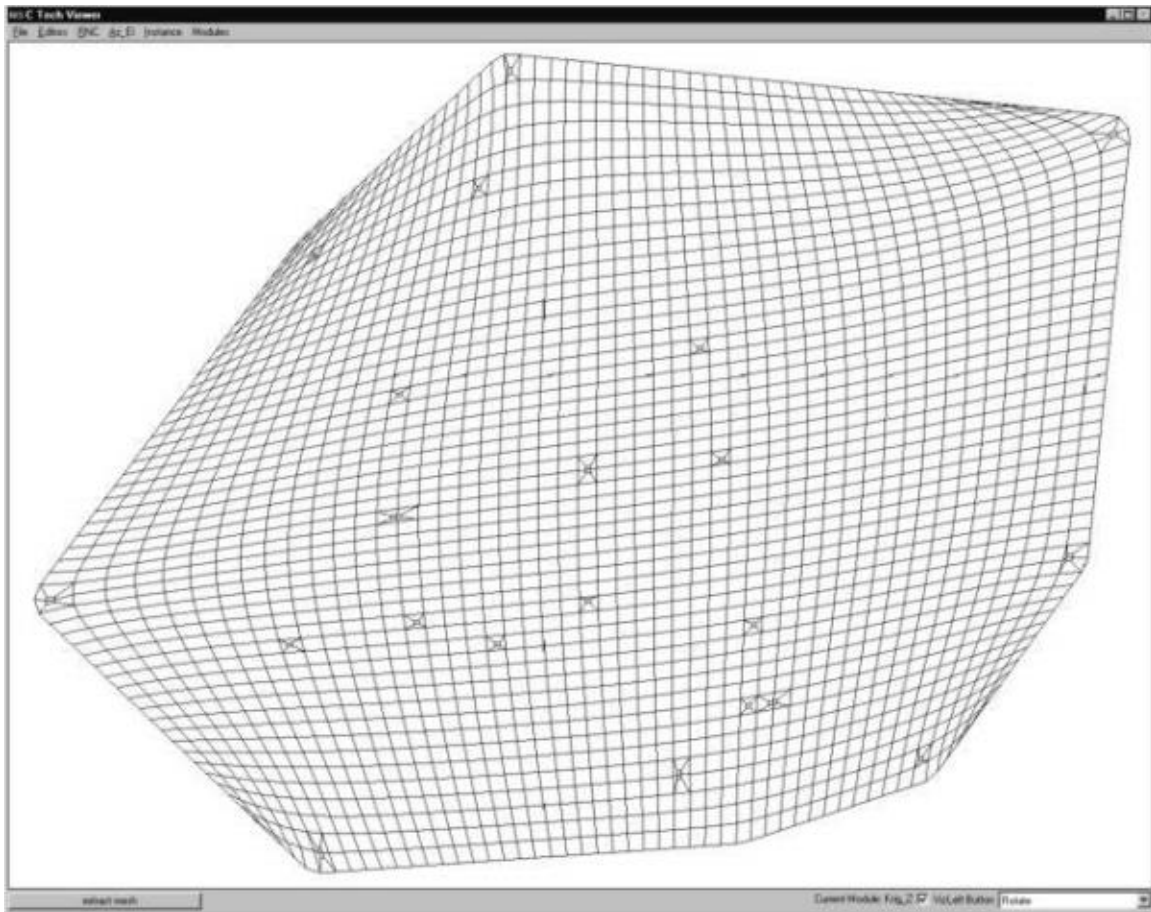


Figure 0.13 Adaptively Gridded Convex Hull Grid

Figure 1.15 is a close-up view of some refined cells near the lower right in Figure 1.14. It shows one of the special cases. If the point to be refined falls very near an existing cell edge, that edge is refined and the cells on either side of the edge are symmetrically refined. Since the edge must be broken into three segments, the cells on both sides must be affected.

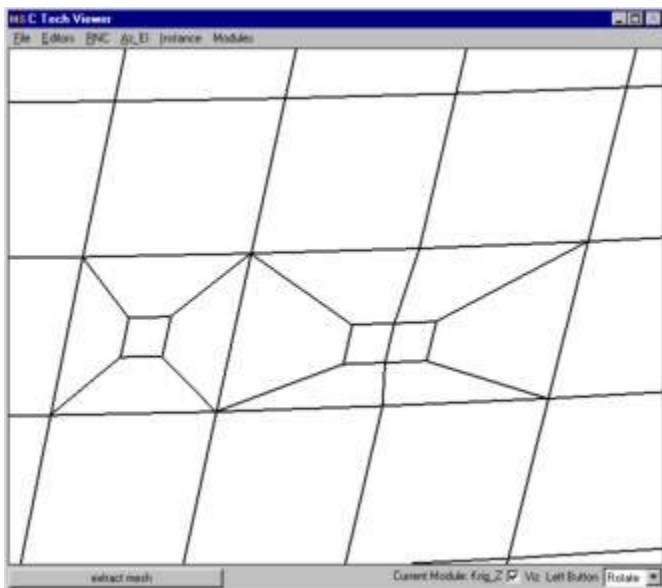


Figure 0.14 Close-up of Figure 1.14

The refinement process can also be applied to all types of 3D grids. When a sample falls in a hexahedron (hex) cell, a new much smaller hex cell is created with one of its' corners located precisely at the coordinates of the sample point. The eight corners of the small cell are connected to the corresponding corners of the parent cell. This creates 7 hex cells that fully occupy the volume of the original cell. Since the 3D-refinement process occurs internal to the volume of the model, it is more difficult to visualize the process. In order to see the refined cells, removing all cells in the grid with any nodes that were below a thresholded concentration level created Figure 1.16. By choosing the threshold properly, several of the refined cells become visible.

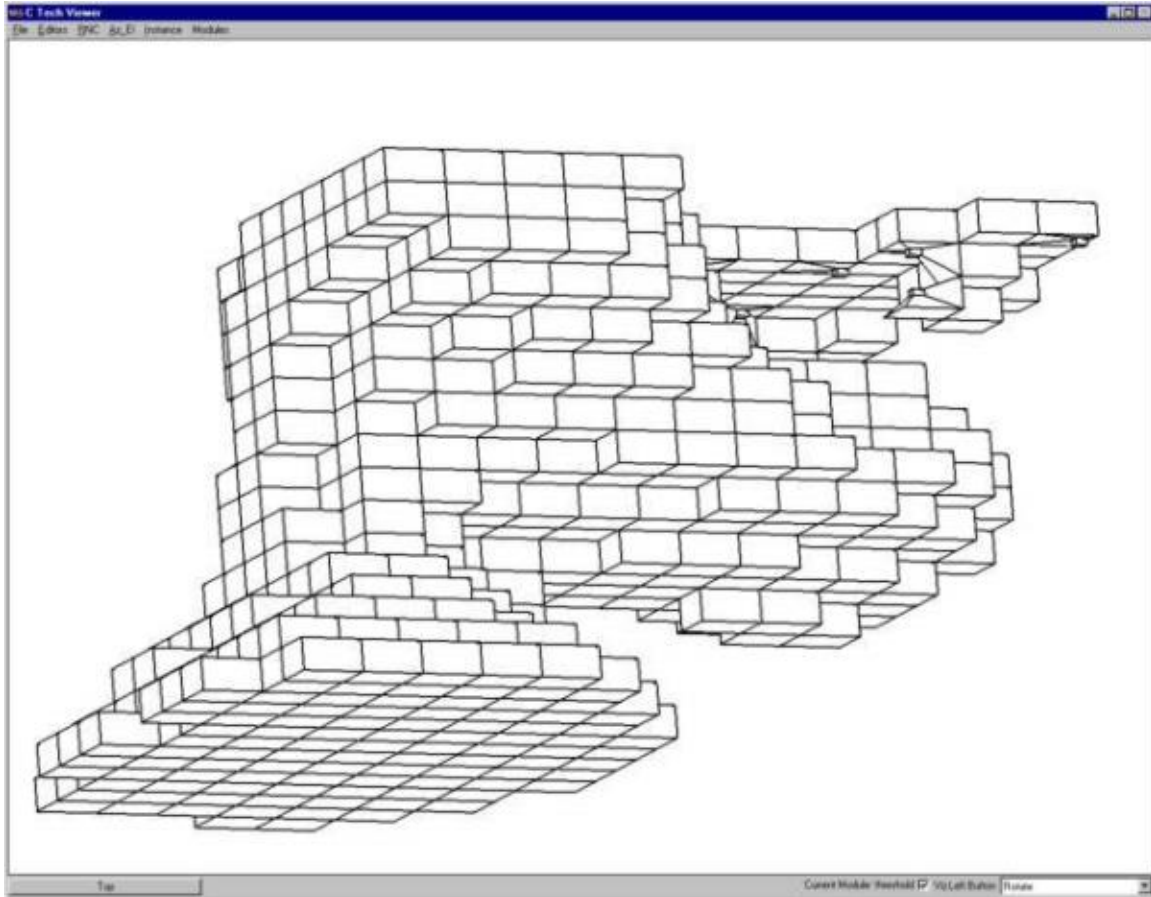


Figure 0.15 3D Adaptively Gridded Model

This figure (Figure 1.17) is an enlarged view of the upper right hand corner. It reveals the structure, relative sizes and connectivity resulting from 3D adaptive gridding.

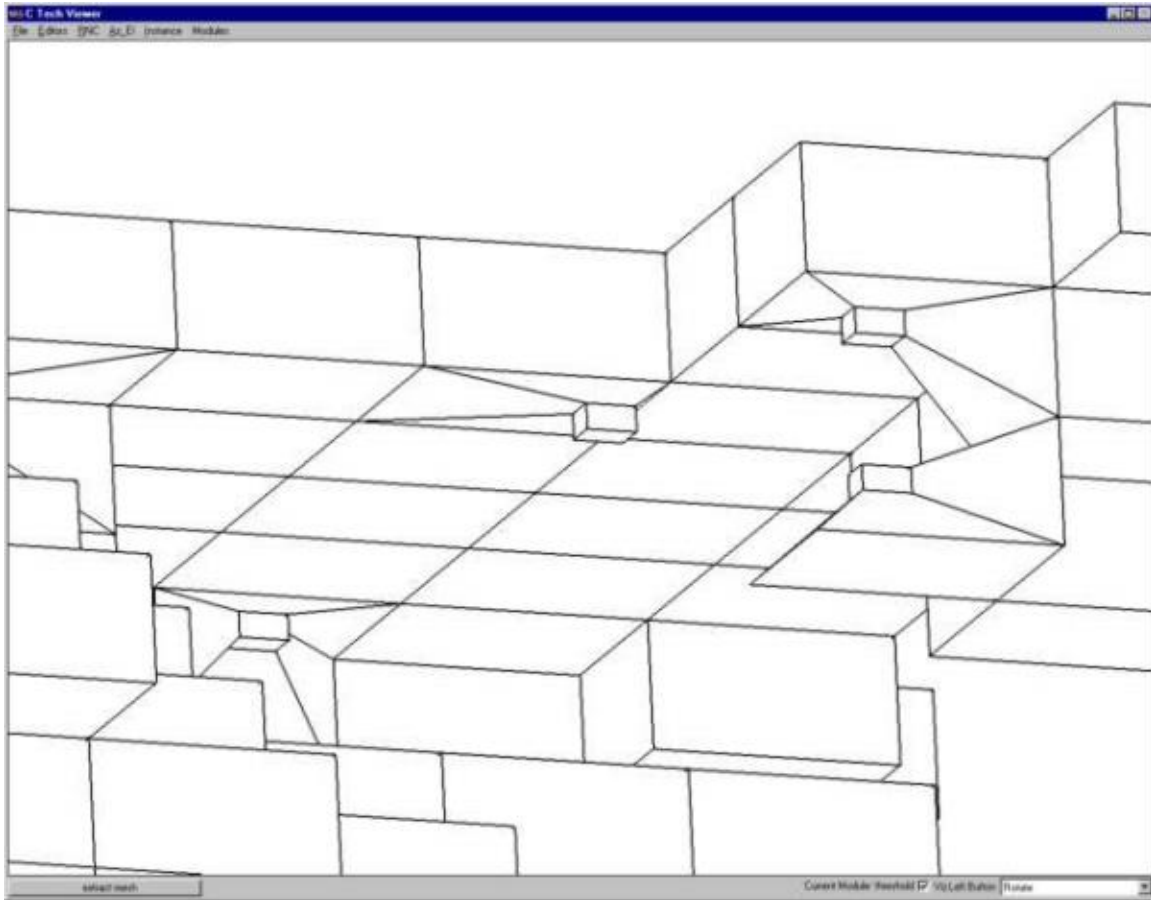


Figure 0.16 Close-up of Figure 1.16

Triangular Networks

Triangular networks are defined as grids of triangle or tetrahedron cells where all of the nodes in the grid are exclusively those in the sample data. For these types of grids, the cell connectivity must be explicitly defined. In two dimensions, these grids are referred to as Triangulated Irregular Networks or TINs. The 3D equivalent grids are Tetrahedral Irregular Networks.

Triangulated Irregular Networks – 2D

Delaunay triangulation is one of the most commonly used methods for creating TINs. By definition, 3 points form a Delaunay triangle if and only if the circle defined by them contains no other point. Focusing on creating Delaunay triangles produces triangles with fat (large) angles that have preferred rendering characteristics. The boundary edges on the Delaunay network form the convex hull, which is the smallest area convex polygon to contain all of the vertices.

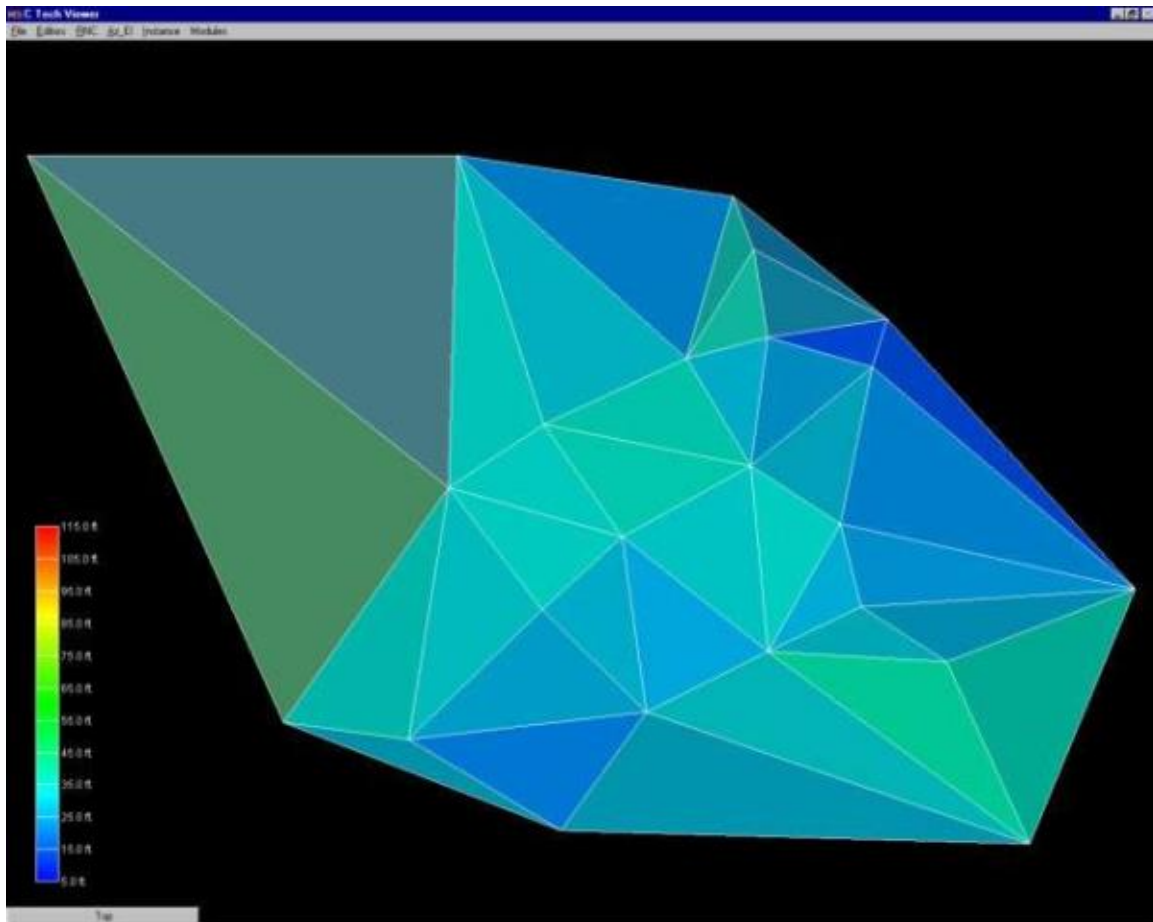


Figure 0.17 Flat-Shaded Delaunay TIN of Geologic Surface

The TIN surface above (Figure 1.18) has significant variation in the size of the triangles. This is a natural consequence of the grid's being created using only nodes from the input data file. When such a surface is rendered with data, having very large triangles can result in very objectionable visualization anomalies. These anomalies result from rendering large triangles that have a range of data values that span a significant fraction of the total data range. There are many methods that could be used to assign color to each triangle. These methods are referred to as surface rendering modes.

Two of the most commonly used rendering modes are flat shading and Gouraud shading. Flat shading assigns a single color to the entire triangle. The color is computed based on the average elevation (data value) for that triangle, lighting parameters and orientation to the viewer camera. In the upper left corner we have a large single triangle that spans a significant range of elevations. When it is assigned a color that corresponds to the mean elevation for that triangle, that color will be wrong. More precisely, the color does not fall within the color scale. Note the color of the triangle in the upper right corner of Figure 1.18 and the one below it. The color of these triangles is outside the range of our color scale.

The problem of large triangles is no better when using Gouraud shading. Gouraud shading assigns colors to each node of the triangle based on the data values. This assures that the colors at the nodes (vertices of the triangles) **will** be correct. Colors are then interpolated over the area of the triangle based on lighting parameters and orientation to the viewer camera. Consider the triangle in the upper right hand

corner of Figure 1.19. The upper right node is assigned the color blue (corresponding to a low value) and the upper left node is assigned the color red (corresponding to a high value). The color scale for this problem ranges from blue to cyan to green to yellow to red. However, for this anomalous situation the color that will be interpolated between blue and red along the uppermost edge will be magenta. Magenta is **not** a color in our range of colors.

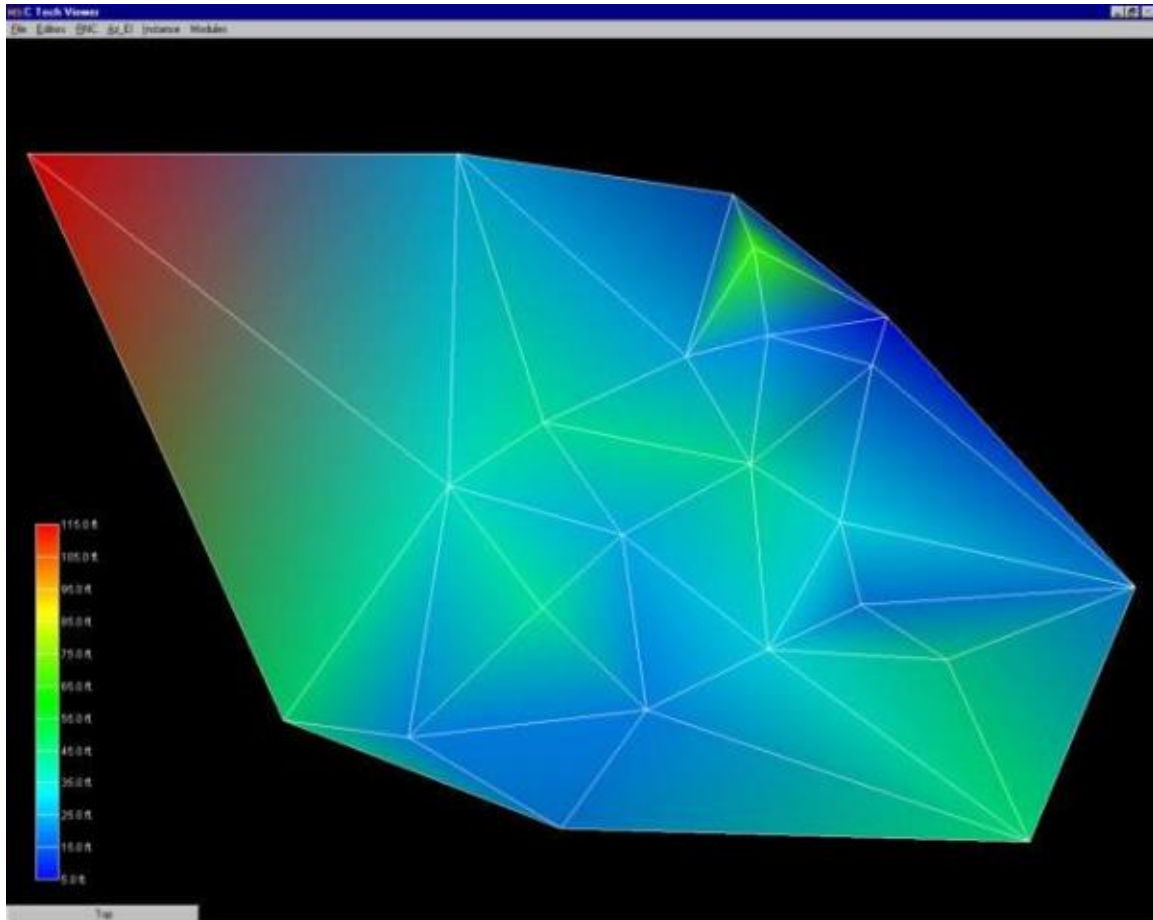


Figure 0.18 Gouraud-Shaded Delaunay TIN of Geologic Surface

To overcome the problems caused by large triangles, the triangles can be refined (subdivided) to create a grid that still contains points that honor the original input nodes, but has more uniform cell sizes. In Figure 1.20 (which has a spatial extent of 500 feet in x and 380 feet in y) it was specified that no triangle's edge may exceed 45 feet in length. We must interpolate the elevation values (or our data values) to these new nodes created as a result of the triangle subdivision. The simplest means of doing this is bilinear interpolation. The refined TIN grid with bilinear interpolation and flat shaded triangles is shown in Figure 1.21. Note that the all of the triangles have appropriate colors. To avoid the large cell coloring problem (this is a problem with all cell types except points), no single cell should have data values at its nodes that span more than about 20 percent of the total data range.

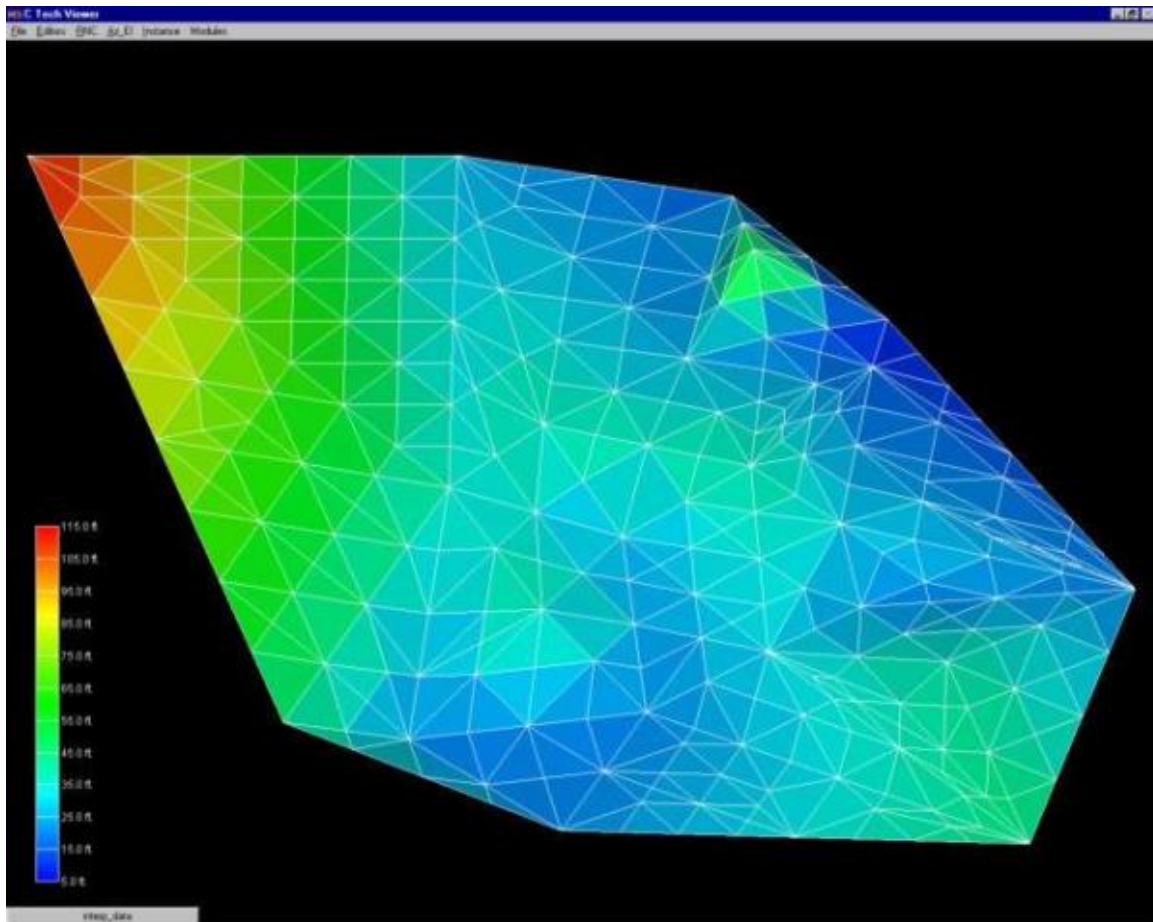


Figure 0.19 Flat-Shaded Subdivided TIN of Geologic Surface

If Gouraud shading is employed instead of flat shading, the resultant surface has a smoother appearance, however the fundamental linear interpolation along cell edges is still evident in the colors. If the maximum triangle size were made much smaller, the flat shaded model would approach the appearance of the Gouraud shaded model. However, without using a different interpolation approach the Gouraud-shaded model would not change dramatically.

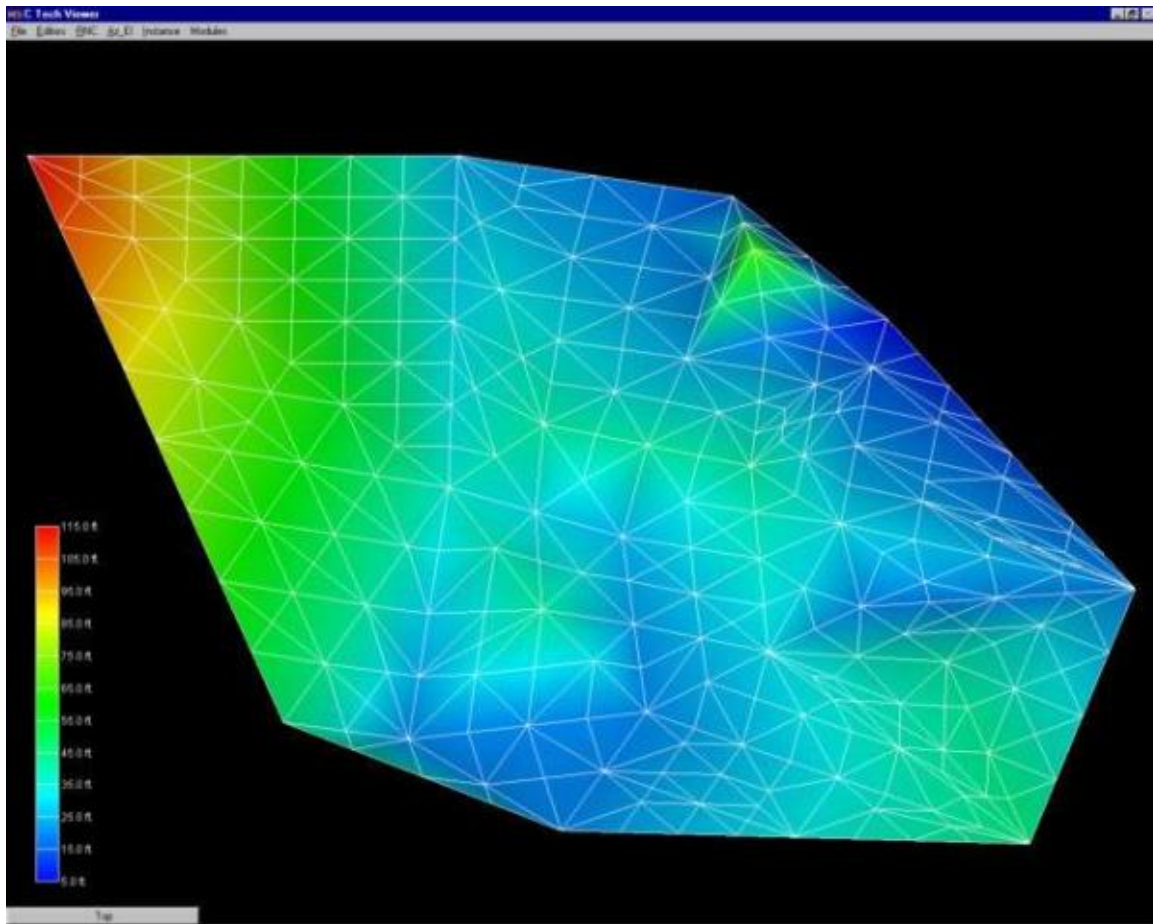


Figure 0.20 Gouraud-Shaded Subdivided TIN of Geologic Surface

EVS includes another technique for coloring surfaces. This method, called solid contours, assigns uniform color bands based on the data values. Figure 1.22 demonstrates this method that subdivides cells using bilinear interpolation. Because this method inherently includes triangle subdivision using bilinear interpolation, the figure would be identical whether the input grid was the large triangles from the original TIN surface or the refined smaller triangles. The boundaries of the colored bands are effectively isopachs (isolines) of constant elevation.

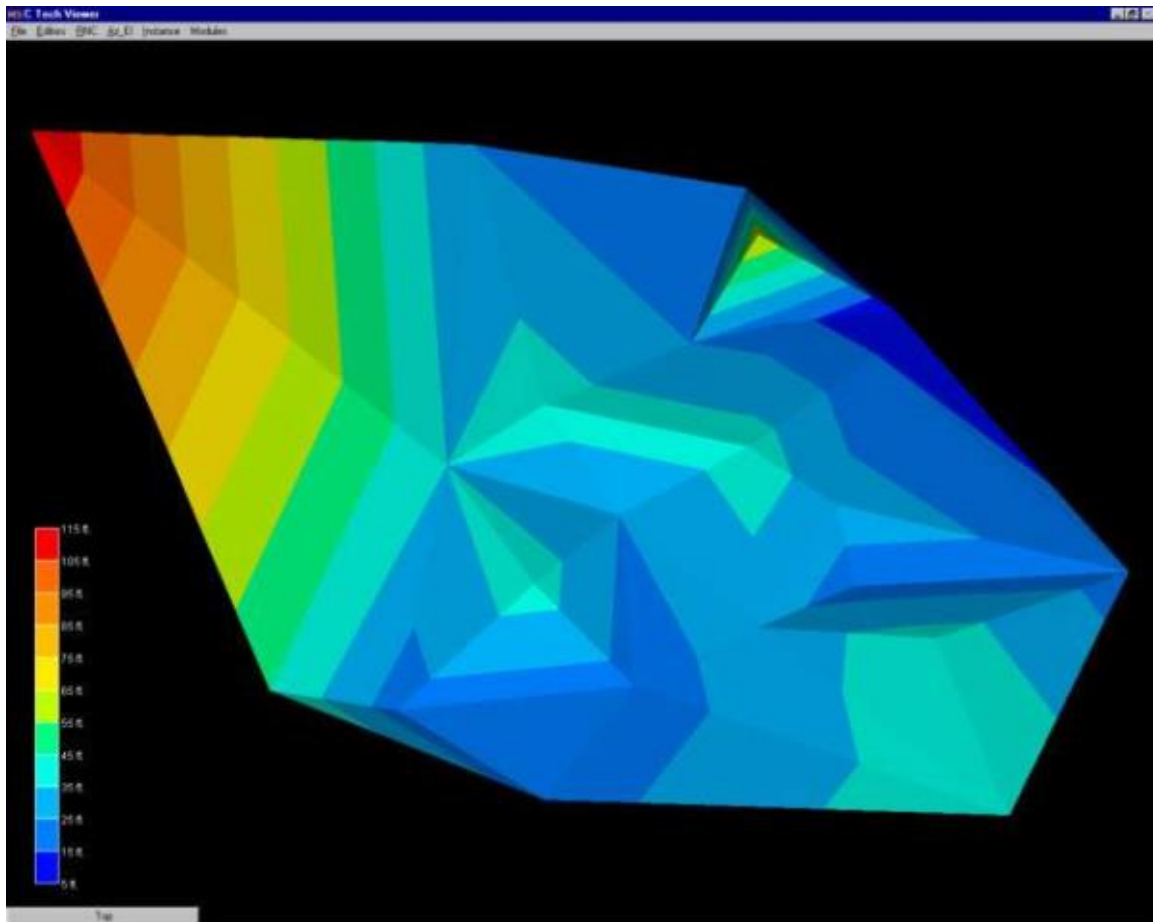


Figure 0.21 Solid Contour TIN of Geologic Surface

To complete this discussion and comparison of gridding and interpolation methods, the same data file was used to create a convex hull grid and the elevation data was estimated using EVS's two-dimensional kriging software. Kriging will be discussed in more detail in section 1.3.3. This technique honors all of the original data points, but creates much smoother distributions between the values. The result shown in Figure 1.23 is a more realistic and aesthetically superior surface. Labeled isolines on 10 foot intervals were added to this figure. Note that these isolines are similar, but much smoother than those in Figure 1.22.

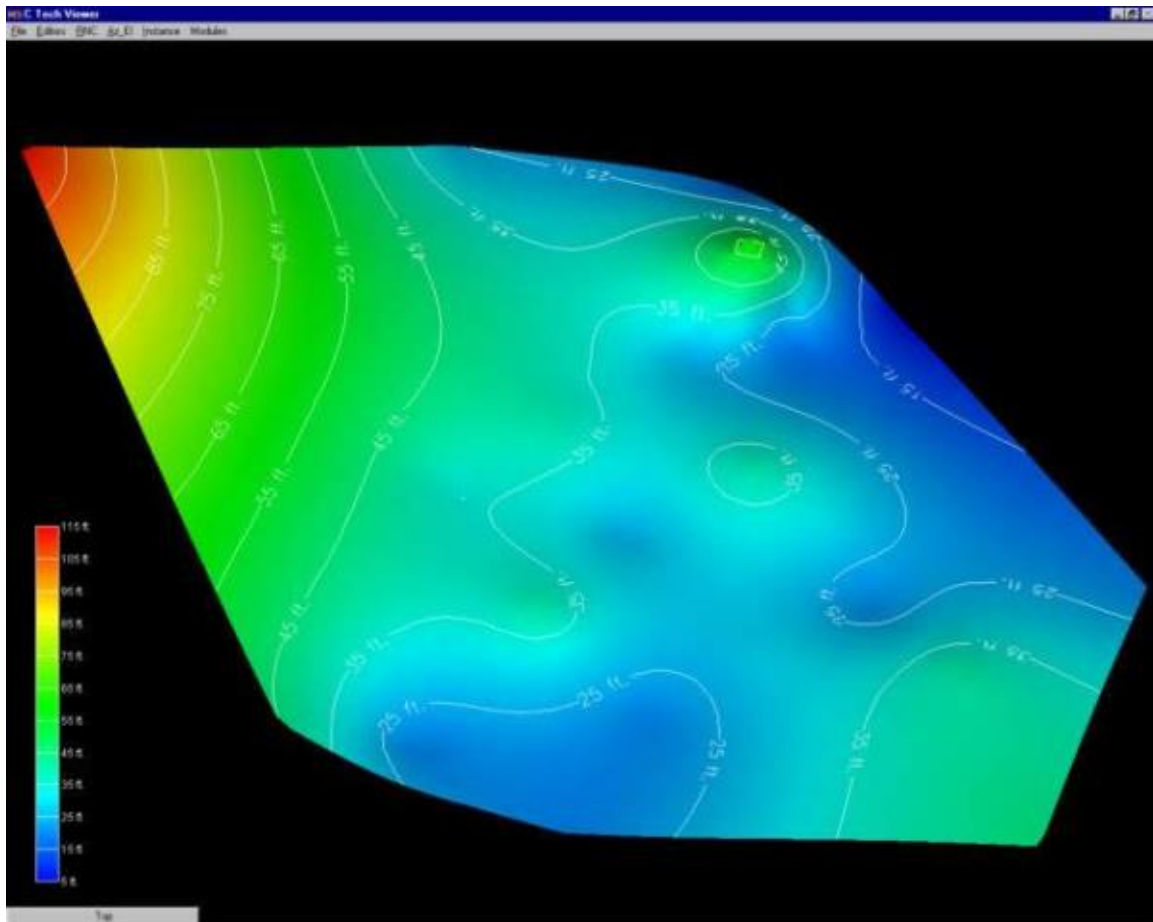


Figure 0.22 Kriged 2D Convex Hull Grid
Tetrahedral Irregular Networks – 3D

Tetrahedral Irregular Networks provide a method to create a volumetric representation of a three-dimensional set of points. As with a TIN, the nodes in the resulting grid are exclusively those in the original measured sample data. Tetrahedral Irregular Networks use tetrahedron cells to fill the three-dimensional convex hull of the data as shown in Figure 1.24. The result often contains cells of widely varying volumes having potentially large data variation across individual cells. For this and other reasons, this approach is not often used.

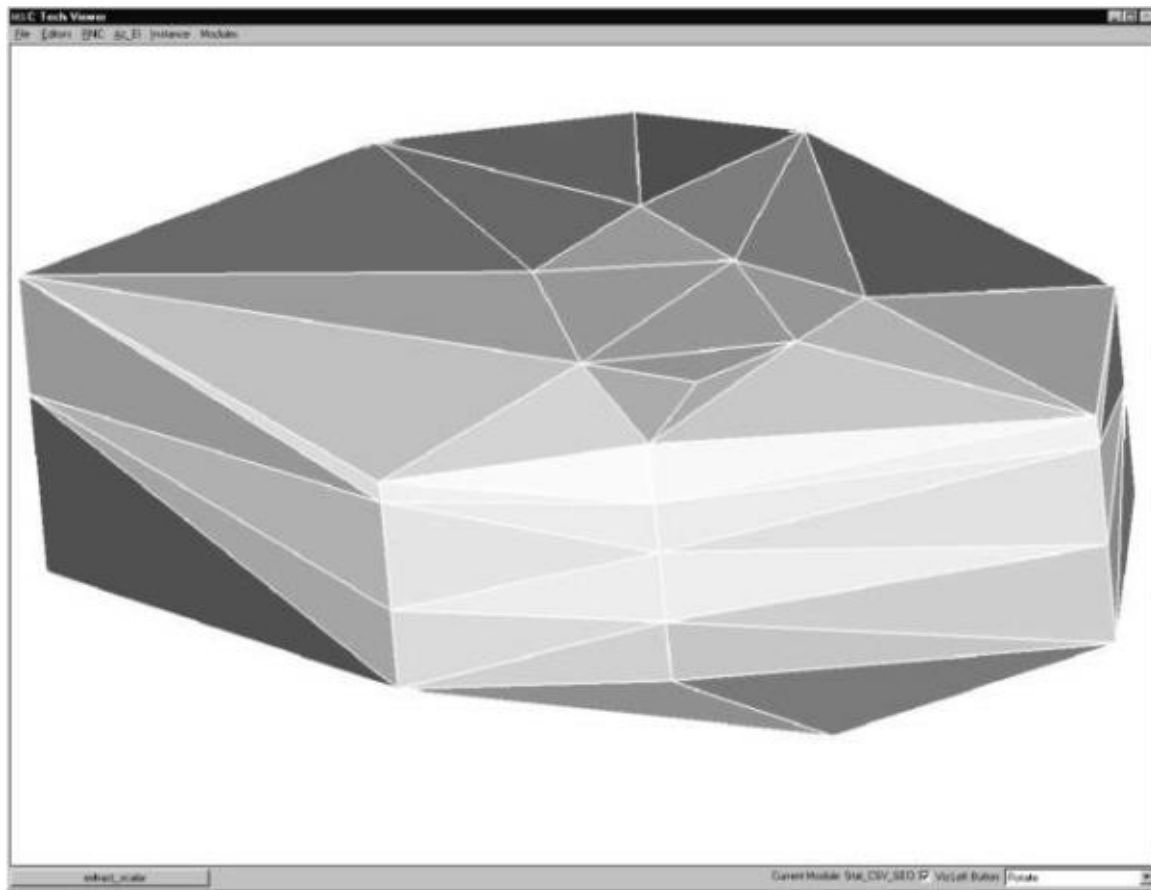


Figure 0.23 Tetrahedral Irregular Network

Estimation Methods

Spatial interpolation methods are used to estimate measured data to the nodes in grids that do not coincide with measured points. The spatial interpolation methods differ in their assumptions, methodologies, complexity, and deterministic or stochastic nature.

Inverse Distance Weighted

Inverse distance weighted averaging (IDWA) is a deterministic estimation method where values at grid nodes are determined by a linear combination of values at known sampled points. IDWA makes the assumption that values closer to the grid nodes are more representative of the value to be estimated than samples further away. Weights change according to the linear distance of the samples from the grid nodes. The spatial arrangement of the samples does not affect the weights. IDWA has seen extensive implementation in the mining industry due to its ease of use. IDWA has also been shown to work well with noisy data. The choice of power parameter in IDWA can significantly affect the interpolation results. As the power parameter increases, IDWA approaches the nearest neighbor interpolation method where the interpolated value simply takes on the value of the closest sample point. Optimal inverse distance weighting is a form of IDWA where the power parameter is chosen on the basis of minimum mean absolute error.

Splining

Splining is a deterministic technique to represent two-dimensional curves on three-dimensional surfaces. Splining may be thought of as the mathematical equivalent of fitting a long flexible ruler to a series of data points. Like its physical counterpart, the

mathematical spline function is constrained at defined points. Splines assume smoothness of variation. Splines have the advantage of creating curves and contour lines that are visually appealing. Some of splining's disadvantages are that no estimates of error are given and that splining may mask uncertainty present in the data. Splines are typically used for creating contour lines from dense regularly spaced data. Splining may, however, be used for interpolation of irregularly spaced data.

Geostatistical Methods (Kriging)

Kriging is a stochastic technique similar to inverse distance weighted averaging in that it uses a linear combination of weights at known points to estimate the value at the grid nodes. Kriging is named after D.L. Krige, who used kriging's underlying theory to estimate ore content. Kriging uses a variogram (a.k.a. semivariogram) which is a representation of the spatial and data differences between some or all possible "pairs" of points in the measured data set. The variogram then describes the weighting factors that will be applied for the interpolation. Unlike other estimation procedures investigated, kriging provides a measure of the error and associated confidence in the estimates. Cokriging is similar to kriging except it uses two correlated measured values. The more intensely sampled data is used to assist in predicting the less sampled data. Cokriging is most effective when the covariates are highly correlated. Both kriging and cokriging assume homogeneity of first differences. While kriging is considered the best linear unbiased spatial predictor (BLUP), there are problems of nonstationarity in real-world data sets.

Surfaces

The choice of surface rendering technique has a dramatic impact on model visualizations. Figure 1.25 is a dramatization that incorporates many common surface-rendering modes. These include Gouraud Shading, Flat Shading, Solid Contours, Transparency and Background Shading. In this figure, a plume is represented in each geologic layer of this model. The geologic layers are exploded and a unique rendering mode is used for each layer. This allows demonstrating five different surface rendering techniques. Section 1.2.5 included some discussion on surface rendering techniques. In the model, a very fine grid (in the x-y plane) was used and the flat shaded plume looks similar to the Gouraud shaded one. The solid contoured plume provides sharp color discontinuities at specific plume levels, however it provides no information about the variation of values within each interval.

The transparent plume was Gouraud shaded. Transparency could be applied to any of the surface rendering techniques except background shading. Transparency provides a means to see features or objects inside of the plume while still providing the basic shape of the plume. Objects inside a colored transparent object will have altered colors and the colors of the transparent object are affected by the color of the background and any other objects inside or behind the plume.

Background shading is a rather different approach. Each cell of the plume is colored the same color as the background. This makes the cell invisible, however the cell is still opaque. Objects that are behind the background shaded cells are not visible. In this example, the cell outlines are shown as lines colored by the concentration values. Background shading of the surfaces provides a "hidden line" rendering where the cells behind are not shown.

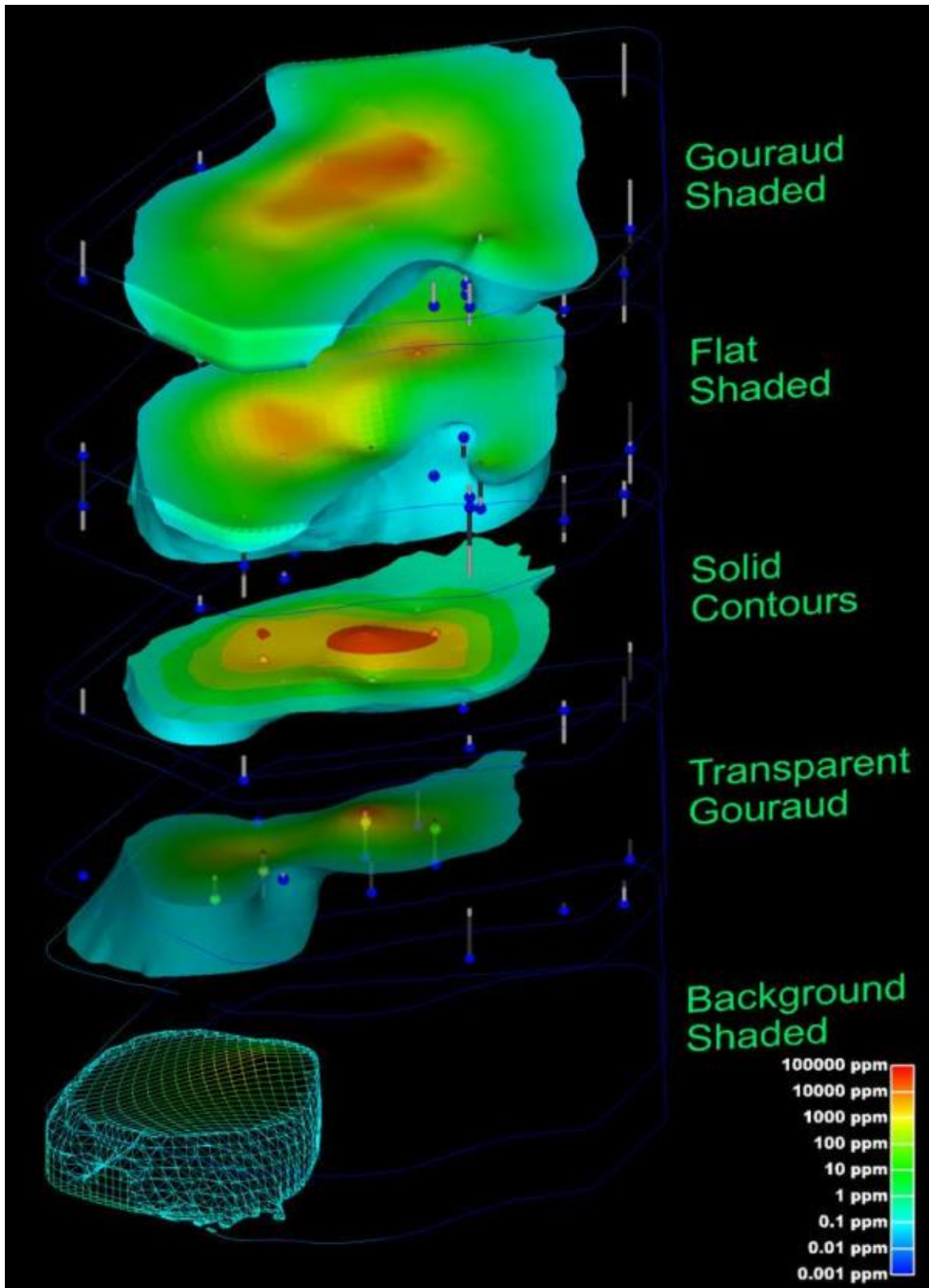


Figure 0.24 plume_shell Showing Various Shading Methods

An example of the rendering mode called "no lighting" has not been included in this paper. This technique renders cells as a single color (similar to flat shading), but with

no lighting or shading effects. This eliminates all three-dimensional clues about the surface and usually produces an undesirable affect.

Texture mapping is a process of projecting a raster image onto one or more surfaces. The images should be geo-referenced (see section 1.1.1.5) to ensure that the image's features are placed in the correct spatial location. In Figure 1.26, a chlorinated hydrocarbon contaminant plume is shown at an industrial facility on the coast. Sand and rock geologic layers are displayed below the ocean layer. A color aerial photograph of the actual site was used to texture map and render the geologic layer that represents the ocean and was also applied to the three-dimensional representations of the site buildings as well as the ground surface.

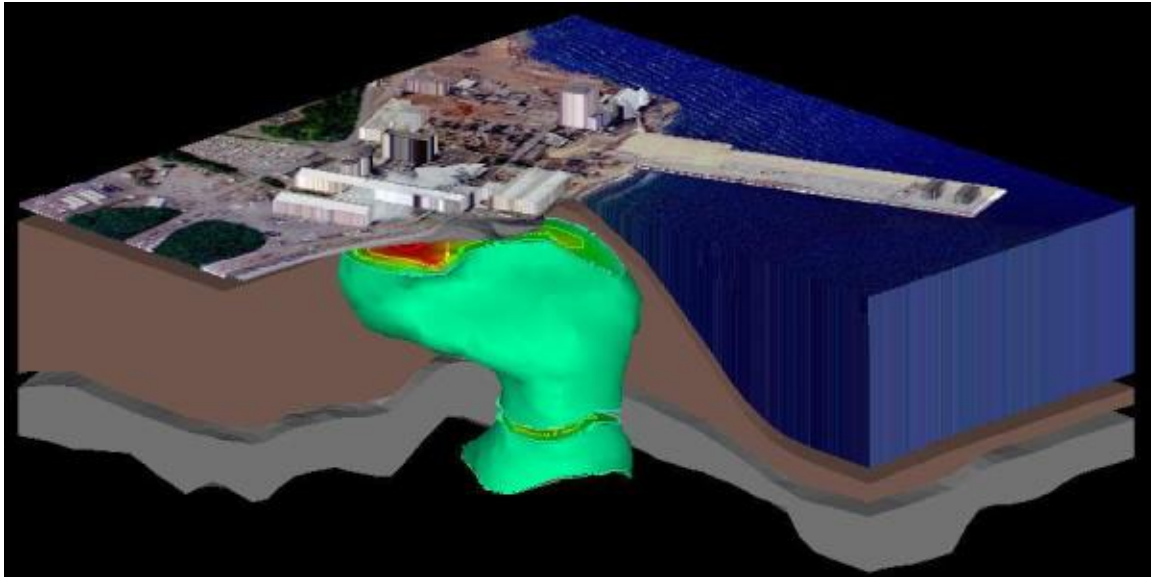


Figure 0.25 Coast Facility Showing Contaminant Plume, Geology with Texture Mapping

Color

The choice of color(s) to be used in a visualization affects the scientific utility of the visualization and has a large psychological impact on the audience. Throughout this paper, a consistent color scale (a.k.a. datamap) has been used. This color scale associates low data values with the color blue and high data values with the color red. Values between the data minimum and maximum are mapped to hues that transition from red to yellow to green to cyan (light blue) to blue. People are accustomed to interpreting blue as a "cold" color and red as a "hot" color. For this reason, lay persons more easily understand this color spectrum. It also provides a reasonably high degree of color fidelity, allowing discrimination of small changes in data values.

However, many times color scales with vivid colors like red are deemed too alarming. Since there is not a universally (or even scientifically) accepted standard for color spectrums used for data presentation, the use of softer shades of color and the elimination of red or other garish colors from the spectrum cannot be challenged on a scientific or legal basis. The consequence of this is the distinct possibility of two different visualizations that both communicate the same information with completely different colors. Often the choice of colors is made on aesthetic or political grounds, governed more by the party being represented and their role in the site than by scientific reasons.

Model Subsetting

Once the model of the site has been created, visually communicating the information about that site generally requires subsetting the model. Subsetting is a generic term used to convey the process of displaying only a portion of the information based on some criteria. The criteria could be "display all portions of the model with a y coordinate of 12,700. This would result in a slice at $y = 12,700$ through the model orthogonal to the y (or North) axis. As this slice passes through geologic layers and/or contaminated volumes, a cross-section of those objects would be visible on the slice. Without subsetting, only the exterior faces of the model will be visible.

When evaluating subsetting operations, the dimensionality of input and output should be considered. As an example, consider the slice described above. If a slice is passed through a volume, the output is a 2D planar surface. If that same slice passes through a surface, the result is a line. Slices reduce the dimensionality of the input by one. The sections below will discuss a few of the more common subsetting techniques.

Plume Visualization

Contaminant plume visualization employs one of the most frequently used subsetting operations. This is accomplished by taking the subset of all regions of a model where data values are above or below a threshold. This subset is also referred to as a volumetric subset and its threshold value as the subsetting level. When creating the objects that represent the plumes, two fundamentally different approaches can be employed. One approach creates one or more surfaces corresponding to all regions in the volume with data values exactly equal to the subsetting level and all portions of the external surfaces of the model where the data values exceed the subsetting level. This results in a closed but hollow representation of the plume. This method, which was used in Figure 1.26, has a dimensionality one less than the input dimensionality.

The other approach subsets the volumetric grid outputting all regions of the model (cells or portions thereof) that exceed the subsetting level. This method has the same dimensionality output as input. The disadvantages of this approach are the need to compute and deal with the all interior volumetric cells and nodes. The advantages include the ability to perform additional subsetting and to compute volumetric or mass calculations on the subset volume.

Cutting and Slicing

Within C Tech's EVS software there is a significant distinction between the terms cut and slice. Slices create objects with dimensionality one less than the input dimensionality. If a volume is sliced the result is a plane. If a surface is sliced the result is one or more lines. If a line is sliced, one or more points are created. Figure 1.29 has three slice planes passing through a volume which has total hydrocarbon concentrations on a fine 3D grid. The horizontal slice plane is transparent and has isolines on $\frac{1}{2}$ decade intervals.

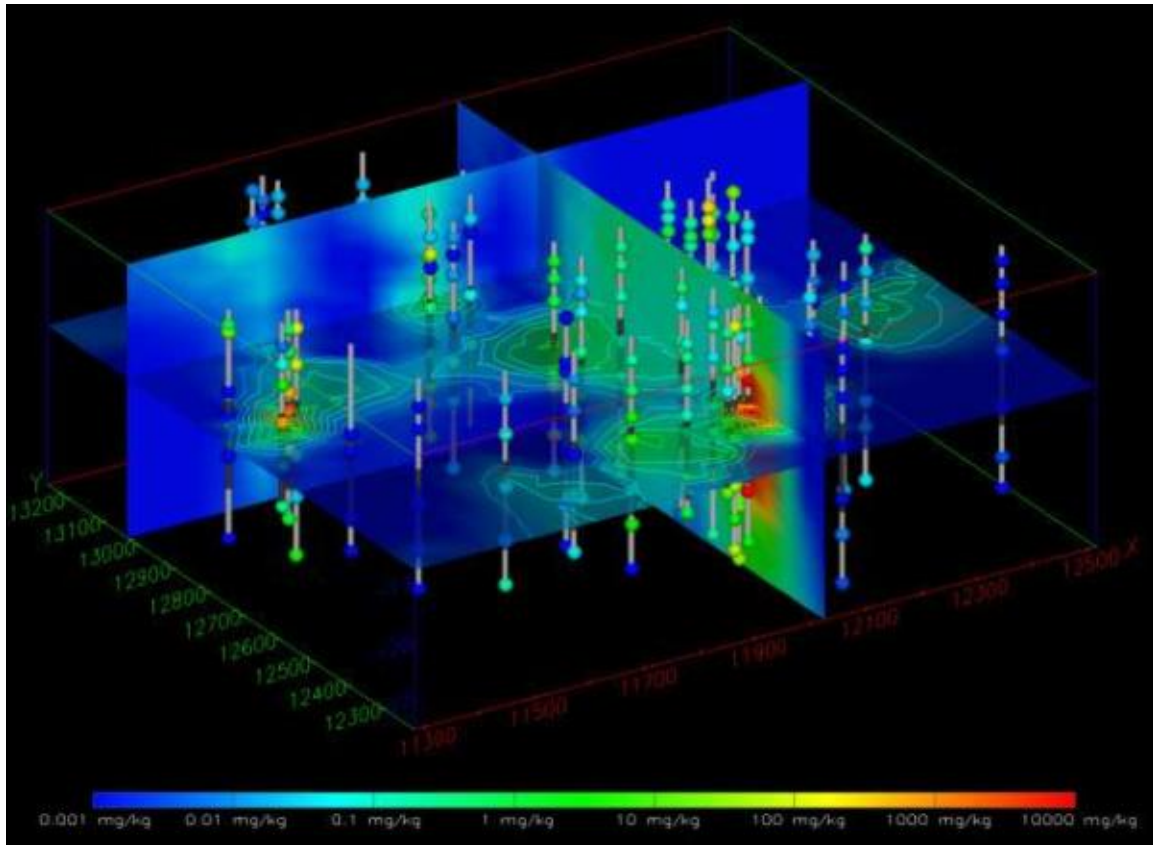


Figure 0.28 Three Slice Planes Passing Through a 3D Kriged Model

By comparison, cutting still uses a plane, but the dimensionality of input and output are the same. Cutting outputs all portions of the objects on one side of the cutting plane. If a volume is cut, a smaller volume is output. In Figure 1.30, the top half of the grid was cut away, but the plume at 1000 ppm is displayed in this portion of the volume. The lower half of the model also has labeled isolines on $\frac{1}{2}$ decade intervals.

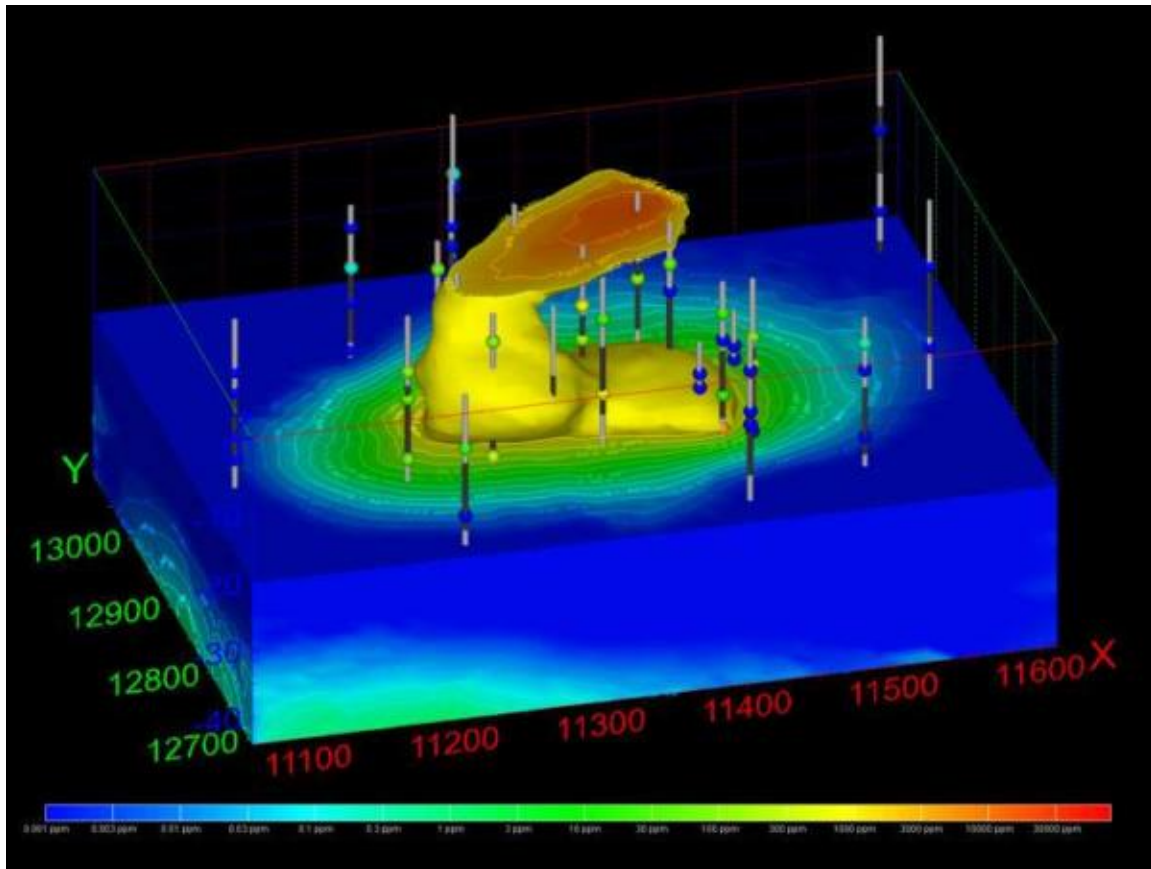


Figure 0.29 Cut 3D Kriged Model with Plume and Labeled Isolines
Isolines

Isolines (sometimes referred to as isopachs) have output dimensionalities that are one less than the input dimensionality. Surfaces with data result in isolines or contour lines that are paths of constant value on the surface(s). Isolines can be labeled or unlabeled. Various labeling techniques can be employed ranging from values placed beside the lines to labels that are incorporated into a break in the line path and mapped to the three-dimensional contours of the underlying surface. Examples of visualizations using isolines are shown in Figures 1.30 and 1.26.

Studio Projects Reference Applications

Each major release of Earth Volumetric Studio will include a corresponding release of Sample Projects, which we tend to refer to as "Studio Projects".

You must be a registered user in order to download the Studio and Studio Projects.

Earth Volumetric Studio 2019.9 (September 19th, 2019)			
File	Description	Type	Size
Earth Volumetric Studio 2019.9	This download will install the September, 2019 Release (2019.9) of Earth Volumetric Studio. This release works with a valid license, or in demonstration mode.	.exe	139.1 MB
Earth Volumetric Studio Sample Projects	This download will install the 2019.10 Sample Projects, appropriate for the September, 2019 Release (Version 2019.9) of Earth Volumetric Studio.	.exe	468 MB

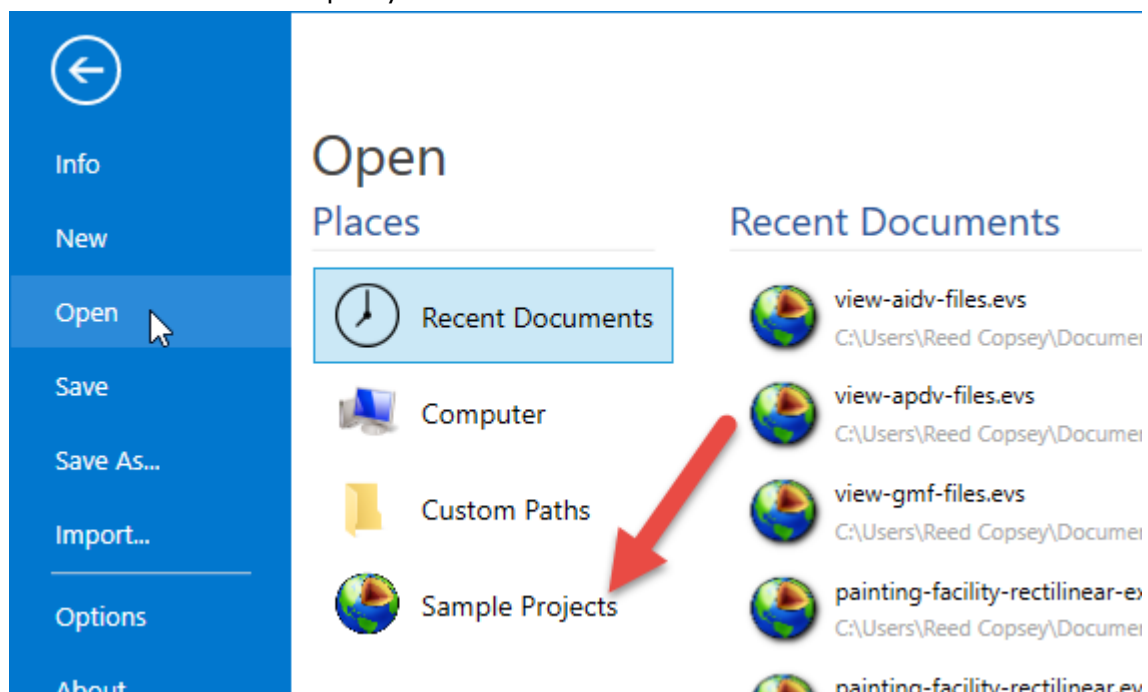
We strongly recommend that all users download the Sample Projects with each new release since these will be a major reference that we will use as a part of technical support. If you call or email Support, asking how to do something, the odds are very high that the answer will often be to take a look at one of the sample applications in Studio Projects. This will give you more than a quick answer, it will provide you with

a detailed example, with real world data which will allow you to see precisely how to create the model and output that you require.

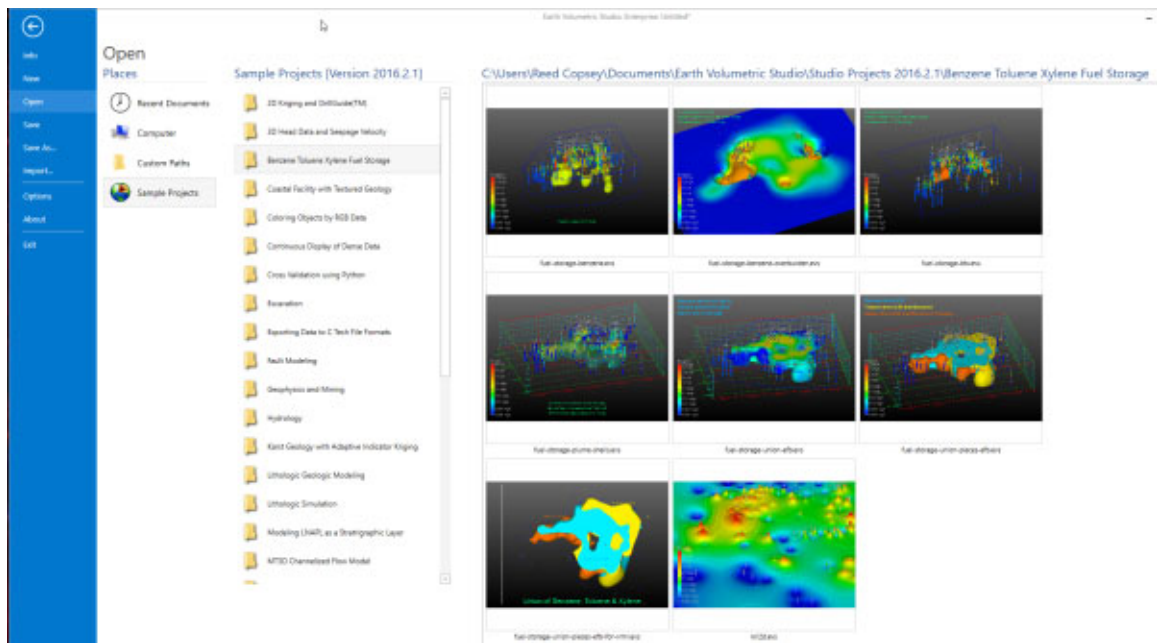
Some of our sample applications include very advanced topics such as:

- Automation using Python
- Time domain data: both geology and chemistry
- Creation of multi-frame vector outputs
 - 4DIMs
 - 3D PDFs
 - 3D Web-published

When you install Studio Projects, they are included in a special way. If you are in Studio and select File.Open you will see



When you select Sample Projects and then select any folder in the list, you will see a large thumbnail image of the output created by each application to quickly allow you to select applications based on their output.



Over time, we expect the number of project folders to grow, and each includes real world data and applications to address the challenges of that data. There is some redundancy among the applications, since some are intentionally simple, while others are increasingly complex to provide more advanced examples.

There is a great deal to be learned by a self-paced exploration of these projects.

EVS Data Input & Output Formats

Input

EVS conducts most of its analysis using input data contained in a number of ASCII files. These files can generally be created using the *Data Transformation Tools*, which are on the *Tools* tab of EVS. These tools will create C Tech's formats from from Microsoft Excel files.

- **Requirement for Consistent Coordinate Systems**
- **Projecting File Coordinates**
- **3D analyte (e.g. chemistry) (.apdv) Format**
- **3D Groundwater analyte (e.g. chemistry) (.aidv) Format**
- **analyte (e.g. chemistry) Time Files (.sct and .gwt) Format**
- **Handling Non-Detects**
- **Pre-Geology File Format**
- **Borehole (.geo) Geology Format**
- **Geology File Example: Sedimentary Layers and Lenses**
- **Geology File Example: Outcrop of Dipping Strata**
- **Geology Multi-File (.gmf) Format**
- **Geology Files for Fence Diagrams**
- **Time Control File (TCF)**
- **EVS Field File Formats (.eff, .efz & .efb)**

.apdv, .aidv and .pgf files can be used to create a single geologic layer model. This is not preferred alternative to creating/representing your valid site geology. However, most sites have some ground surface topography variation. If krig_3d is used without geology input, the resulting output will have flat top and bottom surfaces. The flat top surface may be below or above the actual ground surface at various locations. This can result in plume volumes that are inaccurate.

When a .apdv, .aidv, or .pgf is read by krig_3d_geology the files are interpreted as geology as follows:

1. If **Top** of boring elevations are provided in the file, these values are used to create the ground surface.
2. If **Top** of boring elevations **are not** provided in the file, the elevations of the highest sample in each boring are used to create the ground surface.
3. The bottom surface is created as a flat surface slightly below the lowest sample in the file. The elevation of the surface is computed by taking the lowest sample and subtracting 5% of the total z-extent of the samples.

Output

Because EVS runs under all versions of Microsoft Windows operating systems, there are numerous options for creating output.

Bitmap: EVS renders objects in the viewer in a user defined resolution. That resolution refers to the number of pixels in the horizontal and vertical directions.

Images: EVS also includes the output_images module, which will produce virtually all types of bitmap images supported by Windows. The most common types are .png; .bmp; .tga; .jpg; and .tif. PNG is the recommended format because it has high quality **lossless** compression.

Bitmap Animations: By using output_images with the Animator module, EVS can create bitmap animations. Once a sequence of images is created, the Images_to_Animation module is used to convert these to a bitmap animation format such as .AVI, .MPG, or a proprietary format called .HAV.

Printed Output: The viewer provides the ability to directly output to any Windows printer at a user defined resolution. Alternatively, images may be created (as in **a**) above) and printed.

Vector: EVS offers several vector output options. These include:

VRML: EVS creates VRML files which are a vector output format that allows for creation of 3D modules that model can be zoomed, panned and rotated and can represent most of the objects in the C Tech viewer. VRML files must be played in a VRML viewer or used for creating 3D PDFs or 3D printing.

4DIM: EVS creates 4DIMs, which unlike bitmap (image) based animations contain a **complete 3D model** at each frame of the animation. Each frame can be thought of as a VRML model (*though it is not*) and has similar functionality. Each frame of the model can be zoomed, panned and rotated as a static 3D model **or** you can interact with the 4DIM animation as it is playing.

2D and 3D Shapefiles: Shapefiles that are compatible with ESRI's ArcGIS program can be created in full three-dimensions. Nearly any object in your applications can be output as a shapefile. The primary limitations are associated with the limitations of shapefile. The most significant limitation is the lack of any volumetric elements.

AutoCAD .DXF Files: AutoCAD compatible DXF files can be created in full three-dimensions. Nearly any object in your applications can be output as a DXF file.

Archive: EVS offers several output options for archiving kriged results and/or geologic models. The preferred format is C Tech's fully documented EFF or EFB formats. Both of these file types can be read back into EVS eliminating the need to recreate the models by kriging or re-gridding. This saves time and provides a means to archive the data upon which analysis or visualization was based.

Handling Non-Detects

It is important to understand how to properly handle samples that are classified as non-detects. A non-detect is an analytical sample where the concentration is deemed to be lower than could be detected using the method employed by the laboratory. Non-detects are accommodated in EVS for analysis and visualization using a few **very** important parameters that should be well understood and carefully considered. These parameters control the clipping non-detect handling in all of the EVS modules that read chemistry (.apdv, or .aidv) files. The affected modules are krig_3d, krig_2d, post_samples, and file_statistics.

Non-detects should "almost" never be left out of the data file. They are critically important in determining the spatial extent of the contamination. Furthermore, it is important to understand what it means to have a sample that is *not-detected*. It is not the same as truly ZERO, or perfectly clean. In some cases samples may be non-detects but the detection limit may be so high that the sample should not be used in your data file. If the lab (for whatever reason) reports "Not detected to less than XX.X" where that value XX.X is above your contaminant levels of interest, that sample should not be included in the data file because doing so may create an indefensible "bubble" of high concentration.

As for WHY to use a fraction of the detection limit. At each point where a measurement was made and the result was a non-detect, we should use a fraction of the detection limit (such as one-half to one-tenth). If we were to use the detection limit, we would dramatically overestimate the actual concentrations. From a statistical point of view, when we have a non-detect on a site where the range of measurements varies over several orders of magnitude, it is far more probable that the actual measurement will be dramatically lower than the detection limit rather than just below it. Statistically, if the data spans 6 orders of magnitude, then we would actually expect a non-detect to be 2-3 orders of magnitude below the detection limit! Using ONE-HALF is inane conservative and is a throwback to linear (vs log) interpolation and thinking.

When you might drop a specific Non-Detect: If your target MCL was 1.0 mg/l, and the laboratory reporting limit for a sample were 0.5 mg/l, you would be on the **edge** of whether this sample should be included in your dataset. If you plan to use a multiplier of one-half, it would make the sample 0.25, which is far too close to your MCL given that the only information you really have is that the lab was unable to detect the analyte. If you use a multiplier of one-tenth, it is probably acceptable to include this sample, however if the nearby samples are already lower than this value, we would still recommend dropping it.

Recommended Method: The recommended approach for including non-detects in your data files is the use of Less Than signs "<" preceding the laboratory detection limit for that sample. In this case, the Less Than Multiplier affects each value, making it less by the corresponding fraction.

Otherwise, you can enter either 0.0 or ND for each non-detect in which case, you need to understand (and perhaps modify) the following parameters:

- The number entered into the **Pre-Clip Min** input field will be used during preprocessing to replace any nodal property value that is less than the

specified number. When log processing is being used, the value of Clip Min must be a positive, non-zero value. Generally, Clip Min should be set to a value that is one-half to one-tenth of the global detection limit for the data set. If individual samples have varying detection limits, use the Recommended Method with "<" above. As an example, if the lowest detection limit is 0.1 (which is present in the data set as a 0), and the user sets Clip Min to 0.001, the clipped non-detected values forces two orders of magnitude between any detected value and the non-detected values.

- The **Less Than Multiplier** value affects any file value with a preceeding "<" character. It will multiply these values by the set value.
- The **Detection Limit** value affects any file values set with the "ND" or other non-detect flags (for a list of these flags open the help for the APDV file format). When the module encounters this flag in the file it will insert the a value equal to (Detection Limit * LT Multiplier).

Consistent Coordinate Systems

C Tech's software is designed to work with many types of data. However, because you are creating objects in a three-dimensional domain (x, y, and z extents) you must have all objects defined in a consistent coordinate system. Any coordinate projection may be used, but it is essential that all of your data files (including world files to georeference images) be in the same coordinate system.

Furthermore, if volumes are to be calculated the units for all three axes (x, y, and z) must be the same. We strongly recommend working in feet or meters. Other units may be used (even microns!), but you may have to perform your own unit conversions when computing volumes with [volumetrics](#).

Though all of your analysis must be performed in a consistent coordinate system, we do allow you to have data files with different units. If you choose to do this you must use the **reprojection** capabilities of the [Projecting File Coordinates](#) options in your data files.

Projecting File Coordinates

Discussion of File Coordinate Projection

Each file contains horizontal and vertical coordinates, which can be projected from one coordinate system to another given that the user knows which coordinates systems to project from and to. This is accomplished by adding the REPROJECT tag to the file. This tag is used in place of the coordinate unit definition and causes the file reader to look at the end of the file for a block of text describing the projection definitions. The definitions are a series of flags that listed below. **NOTE:** GMF files do not need the REPROJECT tag, the projection definitions can occur in a continuous block anywhere in the file.

NOTE: When projecting from Geographic to Projected coordinates, please note that Latitude corresponds to Y and Longitude corresponds to X. Since we expect X coordinates before Y coordinates we expect Longitude (then) Latitude (Lon-Lat). If the order in your data file is Lat-Lon you must use the "SWAP_XY" tag at the bottom of the file.

Format (for REPROJECT flag):

APDV and AIDV files:

Line 2:Elevation/Depth Specifier:This line must contain the word *Elevation* or *Depth* (case insensitive) to denote whether sample elevations are true elevation or depth below ground surface. This should be followed by the ASCII string REPROJECT.

AN EXAMPLEFOLLOWS:

```
# This is a comment line....not the header line - the next line is
X Y Z@@TOTH C Bore Top
Elevation 6.0 REPROJECT
```

PGF files:

- **Line 2:** Line 2 contains the declaration of Elevation or Depth, the definitions of Lithology IDs and Names, and coordinate units.
 - **Elevation/Depth Specifier:** This line must contain the word *Elevation* or *Depth* (case insensitive) to specify whether well screen top and bottom elevations are true elevation or depth below ground surface.
 - **Depth** forces the otherwise optional ground surface elevation column to be required. Depths given in column 3 are distances below the ground surface elevation in the last column (column 6). If the top surface is omitted, a value of 0.0 will be assumed and a warning message will be printed to the EVS Information Window.
 - **IDs and Names:** Line 2 should contain Lithology IDs and corresponding names for each material. Each Name is explicitly associated with its corresponding Lithology ID and the pairs are delimited by a pipe symbol "|".
 - Though it is generally advisable, IDs need not be sequential and may be any integer values. This allow for a unified set of Lithology IDs and Names to be applied to a large site where models create for sub-sites may not have all materials.
 - The number of (material) IDs and Names MUST be equal to the number of Lithology IDs specified in the data section. Each material ID present in the data section must have corresponding Lithology IDs and Names. If there are four materials represented in your .pgf file, there should be at least four IDs and Names on line two.
 - The order of Lithology IDs and Names will determine the order that they appear in legends. The IDs do not need to be sequential.
 - You can specify additional IDs and Names, which are not in the data and those will appear on legends.
 - **Coordinate Units:** You should include the units of your coordinates (e.g. *feet* or *meters*). If this is included it must follow the names associated with each Lithology ID.
 - The Btag must follow the IDs & names for the materials.

The first two lines of a PGF EXAMPLEFOLLOWS:

```
Pregeology file
Elevation 1|Silt 2|Fill 3|Clay 4|Sand 5|Gravel REPROJECT
```

GEO files:

Line 2: Elevation/Depth Specifier:

- The only REQUIRED item on this line in the Elevation or Depth Specifier.
 - This line should contain the word *Elevation* or *Depth* (case insensitive) to denote whether sample elevations are true elevation or depth below ground surface.
 - If set to Depth all surface descriptions for layer bottoms are entered as depths relative to the top surface. This is a common means of collecting sample coordinates for borings.
 - Note that the flags such as pinch or short are not modified.
- Line 2 SHOULD contain names for each geologic surface (and therefore the layers created by them).
 - There are some rules that must be observed.
 - The number of surface (layer) names MUST be equal to the number of surfaces. Therefore, if naming layers, the first name should correspond to the top surface and each subsequent name will refer to the surface that defines the bottom of that layer.
 - A name containing a space MUST be enclosed in quotation marks example ("Silty Sand"). Names should be limited to upper and lower case letters, numerals, hyphen "-" and underscore "_". The names defined on line two will appear as the cell set name in the explode_and_scale or select_cells modules. Names should be separated with spaces, commas or tabs.
- The REPROJECT tag must follow the names for the material numbers. It replaces the COORDINATE UNITS

AN EXAMPLE FOLLOWS:

```
X Y TOP BOT_1 BOT_2 BOT_3 BOT_4 BOT_5 BOT_6 BOT_7 Boring
-1 Top Fill SiltySand Clay Sand Silt Sand GravelREPROJECT
```

GMF files:

GMF files can have the projection block placed anywhere in the file.

Projection Block Flags:

****NOTE:** Most flags defined below include arguments denoted by the '[' and ']' characters. These characters should not be included in the file. (Example: IN_XY meters)

PROJECTION: Indicates the start of the coordinate projection block

SWAP_XY: This will swap all coordinates in the x and y columns

UNITS[*string*]: This defines what your final coordinates for x, y, and z, will be. These units will be checked for in the file \data\special\unit_conversions.txt. If they are not found there they will be treated as equivalent to meters.

UNIT_SCALE*[double]*: The UNIT_SCALE flag sets the conversion factor between the final coordinates and meters. This is only necessary if you are defining units with the UNITS flag that are not listed in the \data\special\unit_conversions.txt file.

IN_Z*[string]*: This flag sets what units your z or depth coordinates are. These units if different than the defined UNITS will be converted to the UNIT type. If UNITS are not set then this will generate an error.

IN_X*[string]*: This flag sets what units your x coordinates are. These units if different than the defined UNITS will be converted to the UNIT type. If UNITS are not set then this will generate an error.

IN_Y*[string]*: This flag sets what units your y coordinates are. These units if different than the defined UNITS will be converted to the UNIT type. If UNITS are not set then this will generate an error.

IN_XY*[string]*: This flag sets what units your x and y coordinates are. These units if different than the defined UNITS will be converted to the UNIT type. If UNITS are not set then this will generate an error.

PROJECT_FROM_ID*[int]*: This flag sets the EPSG ID value you wish to project from, you can look up what ID is appropriate for your location using the project_field module. To use this flag you must set the PROJECT_TO_ID or PROJECT_TO flag as well.

PROJECT_TO_ID*[int]*: This flag sets the EPSG ID value you wish to project to, you can look up what ID is appropriate for your location using the project_field module. To use this flag you must set the PROJECT_FROM_ID or PROJECT_FROM flag as well.

PROJECT_FROM*[string]*: This flag sets the NAME of the location you wish to project from, you can look up what NAME is appropriate for your location using the project_field module. To use this flag you must set the PROJECT_TO_ID or PROJECT_TO flag as well. **IMPORTANT**: The full name should be enclosed in quotation marks so that the full name will be read.

PROJECT_TO*[string]*: This flag sets the NAME of the location you wish to project to, you can look up what NAME is appropriate for your location using the project_field module. To use this flag you must set the PROJECT_FROM_ID or PROJECT_FROM flag as well. **IMPORTANT**: The full name should be enclosed in quotation marks so that the full name will be read.

TRANSLATE*[double double double]*: This flag will translate each coordinate in the file by these values. It will translate x by the first value, y by the second, and all z values by the third.

END_PROJECTION: Denotes the end of the projection block and is required.

Example 1:

```
PROJECTION
PROJECT_FROM_ID 4267
PROJECT_TO "NAD83 / UTM zone 10N"
UNITS "meters"
SWAP_XY
END_PROJECTION
```

Example 2:

```
PROJECTION
UNITS "meters"
IN_XY "km"
IN_Z "ft"
END_PROJECTION
```

Analytical Data

All analytical data can be represented in one of two formats:

- [Data collected at points APDV](#)
 - Additionally, 2D or 3D point shapefiles with analytical data can be used as analytical input data.
- [Data collected over intervals AIDV](#)

These two file formats can support many different types of data including:

- Soil, groundwater and air contaminant concentrations
- Ore data
- Data collected at multiple dates and times
- MIP (semi-continuous)
- Geophysical data
 - Porosity, transmissivity
 - Hydraulic head
 - Flow velocity
 - Electrical Resistivity
 - Ground Penetrating Radar
 - Seismic
- Oceanographic data
 - CTD
 - Plankton density
 - Other water quality
 - Sub-bottom sediment measurements

APDV: Analyte Point Data File Format

Discussion of analyte (e.g. chemistry) or Property Files

Analyte (e.g. chemistry) or property files contain horizontal and vertical coordinates, which describe the 3-D locations and values of properties of a system. For simplicity, these files will generally be referred to in this manual as analyte (e.g. chemistry) files, although they can actually contain any scalar property value of interest.

Analyte (e.g. chemistry) files must be in ASCII format and can be delimited by commas, spaces, or tabs. They must have a .apdv suffix to be selected in the file browsers of EVS modules. The content and format of analyte (e.g. chemistry) files are the same, except that fence diagram files require some special subsetting and ordering. Each line of the analyte (e.g. chemistry) file contains the coordinate data for one sampling location and any number of (columns of) analyte (e.g. chemistry) or property values. There are no computational restrictions on the number of borings and/or samples that can be included in a analyte (e.g. chemistry) file, except that run times for execution of kriging do increase with the number of samples in the file.

Analyte (e.g. chemistry) data can be visualized independently or within a domain bounded by a geologic system. When a geologic domain is utilized for a 3-D visualization, a consistent coordinate system must be used in both the analyte (e.g. chemistry) and geology files. The boring and sample locations in 3-D analyte (e.g. chemistry) files do not have to correspond to those in the geology files, except that they must be contained within the spatial domain of the geology, or they will not be displayed in the visualization. If the posting of borings and sample locations are to honor the topography of a site, the analyte (e.g. chemistry) files also must contain the top surface elevation of the boring. As will be described in later sections, EVS uses tubes to show actual boring locations and depths, and spheres to show actual sample locations in three-space. In order for these entities to be correctly positioned in relation to a variable topography, the top elevation of the boring must be supplied to the program.

Format:

You may insert comment lines in C Tech analyte (e.g. chemistry) (.apdv) input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

Line 1: You may include any header message here (that does not start with a '#' character) **unless** you wish to include analyte names for use by other EVS modules (e.g. data component name). The format for line 1 to enable chemical names is as follows

A. Placing a pair of '@' symbols triggers the use and display of chemical names (example @@VOC). Any characters up to the @@ characters are ignored, and only the first analyte name needs @@, after that the chemical names must be delimited by spaces,

B. The following rules for commas are implemented to accommodate comma delimited files and also for using chemical names which have a comma within (example 1,1-DCA). Commas following a name will not become a part of the name, but a comma in the middle of a text string will be included in the name. The recommended approach is to put a space before the names.

C. If you want a space in your analyte name, you may use underscores and EVS will convert underscores to spaces (example: Vinyl_Chloride in a .aidv file will be converted to 'r;Vinyl Chloride.' Or you may surround the entire name in quotation marks (example: "Vinyl Chloride").

The advantages of using chemical names (attribute names of any type) are the following:

- many modules use analyte names instead of data component numbers,

- when writing EVS Field files (.eff, .efb, etc.), you will get analyte names instead of data component numbers.
- when querying your data set with post_sample's mouse interactivity, the analyte name is displayed.
- time-series data can be used and the appropriate time-step can be displayed.

Line 2: Specifications

- **Elevation/Depth Specifier:** The first item on line 2 must be the word *Elevation* or *Depth* (case insensitive) to denote whether well screen top and bottom elevations are true elevation or depth below ground surface.
- **Coordinate Units:** After Depth/Elevation, include the units of your coordinates (e.g. *feet* or *meters*)

Line 3: Specifications

- The first integer (n) is the number of samples (rows of data) to follow. **You may specify "All" instead to use all data lines in the file.**
- The second integer is the number of analyte (chemistry) values per sample.
- The units of each data analyte column (e.g. *ppm* or *mg/kg*).

Line 4: The first line of analyte point data must contain:

- X
- Y
- Elevation (or Depth) of sample
- (one or more) Analyte Value(s) (chemistry or property)
- Well or Boring name. The boring name cannot contain spaces (recommend underscore "_" instead).
- Elevation of the top of the boring.

Boring name and top are optional parameters, but are used by many modules and it is highly recommended that you include this information in your file if possible. They are used by post_samples for posting tubes along borehole traces and for generating tubes which start from the ground surface of the borehole. Both krig_3d and krig_3d_geology will use this information to determine the Z spatial extent of your grids (krig_3d_geology will create a layer that begins at ground surface if this information is provided). Numbers and names can be separated by one comma and/or any number of spaces or tabs.

BLANK ENTRIES (CELLS) ARE NOT ALLOWED.

Please see the section on [Handling Non-Detects](#) for information on how to deal with samples whose concentration is below the detection limit. For any sample that is not detected you may enter any of the following. Please note that the first three flag words are not case sensitive, but must be spelled exactly as shown below.

- Prepend a less than sign < to the actual detection limit for that sample. This allows you to set the "Less Than Multiplier" in all modules that read .apdv files to a value such as 0.1 to 0.5 (10 to 50%). This is the preferred and most rigorous method.

- nondetect
- non-detect
- nd
- 0.0 (zero)

For files with multiple analytes such as the example below, if an analyte was not measured at a sample location, use any of the flags below to denote that this sample should be skipped for this analyte. Please note that these flag words are not case sensitive, but must be spelled exactly as shown below.

- missing
- unmeasured
- not-measured
- nm
- unknown
- unk
- na

[Example Files are here:](#)

APDV: Analyte Point Data File Format

Discussion of analyte (e.g. chemistry) or Property Files

Analyte (e.g. chemistry) or property files contain horizontal and vertical coordinates, which describe the 3-D locations and values of properties of a system. For simplicity, these files will generally be referred to in this manual as analyte (e.g. chemistry) files, although they can actually contain any scalar property value of interest.

Analyte (e.g. chemistry) files must be in ASCII format and can be delimited by commas, spaces, or tabs. They must have a .apdv suffix to be selected in the file browsers of EVS modules. The content and format of analyte (e.g. chemistry) files are the same, except that fence diagram files require some special subsetting and ordering. Each line of the analyte (e.g. chemistry) file contains the coordinate data for one sampling location and any number of (columns of) analyte (e.g. chemistry) or property values. There are no computational restrictions on the number of borings and/or samples that can be included in a analyte (e.g. chemistry) file, except that run times for execution of kriging do increase with the number of samples in the file.

Analyte (e.g. chemistry) data can be visualized independently or within a domain bounded by a geologic system. When a geologic domain is utilized for a 3-D visualization, a consistent coordinate system must be used in both the analyte (e.g. chemistry) and geology files. The boring and sample locations in 3-D analyte (e.g. chemistry) files do not have to correspond to those in the geology files, except that they must be contained within the spatial domain of the geology, or they will not be displayed in the visualization. If the posting of borings and sample locations are to honor the topography of a site, the analyte (e.g. chemistry) files also must contain the top surface elevation of the boring. As will be described in later sections, EVS uses tubes to show actual boring locations and depths, and spheres to show actual sample locations in three-space. In order for these entities to be correctly positioned in relation to a variable topography, the top elevation of the boring must be supplied to the program.

Format:

You may insert comment lines in C Tech analyte (e.g. chemistry) (.apdv) input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

Line 1: You may include any header message here (that does not start with a '#' character) **unless** you wish to include analyte names for use by other EVS modules (e.g. data component name). The format for line 1 to enable chemical names is as follows

A. Placing a pair of '@' symbols triggers the use and display of chemical names (example @@VOC). Any characters up to the @@ characters are ignored, and only the first analyte name needs @@, after that the chemical names must be delimited by spaces,

B. The following rules for commas are implemented to accommodate comma delimited files and also for using chemical names which have a comma within (example 1,1-DCA). Commas following a name will not become a part of the name, but a comma in the middle of a text string will be included in the name. The recommended approach is to put a space before the names.

C. If you want a space in your analyte name, you may use underscores and EVS will convert underscores to spaces (example: Vinyl_Chloride in a .aidv file will be converted to 'r;Vinyl Chloride.' Or you may surround the entire name in quotation marks (example: "Vinyl Chloride").

The advantages of using chemical names (attribute names of any type) are the following:

- many modules use analyte names instead of data component numbers,
- when writing EVS Field files (.eff, .efb, etc.), you will get analyte names instead of data component numbers.
- when querying your data set with post_sample's mouse interactivity, the analyte name is displayed.
- time-series data can be used and the appropriate time-step can be displayed.

Line 2: Specifications

- **Elevation/Depth Specifier:** The first item on line 2 must be the word *Elevation* or *Depth* (case insensitive) to denote whether well screen top and bottom elevations are true elevation or depth below ground surface.
- **Coordinate Units:** After Depth/Elevation, include the units of your coordinates (e.g. *feet* or *meters*)

Line 3: Specifications

- The first integer (n) is the number of samples (rows of data) to follow. **You may specify "All" instead to use all data lines in the file.**
- The second integer is the number of analyte (chemistry) values per sample.
- The units of each data analyte column (e.g. *ppm* or *mg/kg*).

Line 4: The first line of analyte point data must contain:

- X
- Y
- Elevation (or Depth) of sample
- (one or more) Analyte Value(s) (chemistry or property)

- Well or Boring name. The boring name cannot contain spaces (recommend underscore "_" instead).
- Elevation of the top of the boring.

Boring name and top are optional parameters, but are used by many modules and it is highly recommended that you include this information in your file if possible. They are used by post_samples for posting tubes along borehole traces and for generating tubes which start from the ground surface of the borehole. Both krig_3d and krig_3d_geology will use this information to determine the Z spatial extent of your grids (krig_3d_geology will create a layer that begins at ground surface if this information is provided). Numbers and names can be separated by one comma and/or any number of spaces or tabs.

BLANK ENTRIES (CELLS) ARE NOT ALLOWED.

Please see the section on [Handling Non-Detects](#) for information on how to deal with samples whose concentration is below the detection limit. For any sample that is not detected you may enter any of the following. Please note that the first three flag words are not case sensitive, but must be spelled exactly as shown below.

- [Prepend a less than sign < to the actual detection limit for that sample. This allows you to set the "Less Than Multiplier" in all modules that read .apdv files to a value such as 0.1 to 0.5 \(10 to 50%\). This is the preferred and most rigorous method.](#)
- [nondetect](#)
- [non-detect](#)
- [nd](#)
- [0.0 \(zero\)](#)

For files with multiple analytes such as the example below, if an analyte was not measured at a sample location, use any of the flags below to denote that this sample should be skipped for this analyte. Please note that these flag words are not case sensitive, but must be spelled exactly as shown below.

- [missing](#)
- [unmeasured](#)
- [not-measured](#)
- [nm](#)
- [unknown](#)
- [unk](#)
- [na](#)

[Example Files are here:](#)

Three Dimensional Analyte Point Data File Example

An actual .apdv file could look like the following:

X	Y	ELEV	@@1-DCA	1-DCE	TCE	VC	SITE_ID
Elevation	feet						
50	4		mg/kg	ug/kg	ug/kg	mg/kg	

12008	12431	22.9	22	missing	500	<0.01	CSB-39
12008	12431	18.9	<0.01	<0.01	2800	<0.01	CSB-39
12008	12431	13.4	<0.01	<0.01	290	<0.01	CSB-39
12008	12431	8.4	<0.01	<0.01	9.7	<0.01	CSB-39
12008	12431	7.9	<0.01	<0.01	23	<0.01	CSB-39
12008	12431	1.9	<0.01	<0.01	24	<0.01	CSB-39
11651	13184	28.5	<0.01	<0.01	<0.01	<0.01	CSB-40
11651	13184	26	<0.01	<0.01	<0.01	<0.01	CSB-40
11427	12781	28.8	0.28	0.02	0.78	<0.01	CSB-42
11427	12781	24.8	<0.01	0.02	0.76	<0.01	CSB-42
11427	12781	17.3	<0.01	<0.01	0.01	<0.01	CSB-42
11427	12781	14.6	<0.01	<0.01	0.01	<0.01	CSB-42
11427	12781	9.8	<0.01	<0.01	<0.01	<0.01	CSB-42
11427	12781	3.3	0.64	0.14	1.5	0.19	CSB-42
11410	12725	29.6	0.01	<0.01	0.01	<0.01	CSB-43
11410	12725	23.6	0.08	<0.01	0.02	<0.01	CSB-43
11410	12725	21.6	0.04	<0.01	0.01	<0.01	CSB-43
11410	12725	12.1	0.1	<0.01	<0.01	0.13	CSB-43
11410	12725	6.1	0.06	<0.01	<0.01	0.05	CSB-43
11417	12819	28.2	0.01	<0.01	0.03	<0.01	CSB-44
11417	12819	24.2	0.04	<0.01	0.04	<0.01	CSB-44
11417	12819	16.2	0.43	0.04	0.04	<0.01	CSB-44
11417	12819	11.2	1.1	<0.01	<0.01	<0.01	CSB-44
11417	12819	9.2	<0.01	<0.01	<0.01	<0.01	CSB-44
11417	12819	6.2	<0.01	<0.01	<0.01	<0.01	CSB-44
11417	12819	2.2	0.06	<0.01	<0.01	<0.01	CSB-44
11402	12898	28.5	<0.01	<0.01	<0.01	<0.01	CSB-45
11402	12898	24.5	<0.01	<0.01	<0.01	<0.01	CSB-45
11402	12898	14.5	0.79	<0.01	1.7	<0.01	CSB-45
11402	12898	9	<0.01	<0.01	11	<0.01	CSB-45
11402	12898	2	0.18	<0.01	0.01	0.11	CSB-45
11260	12819	28.4	<0.01	<0.01	<0.01	<0.01	CSB-46
11260	12819	22.4	<0.01	<0.01	<0.01	<0.01	CSB-46
11260	12819	16.9	<0.01	<0.01	<0.01	<0.01	CSB-46
11260	12819	11.9	<0.01	<0.01	<0.01	<0.01	CSB-46
11260	12819	2.9	<0.01	<0.01	<0.01	<0.01	CSB-46

11340	12893	24.6	<0.01	<0.01	<0.01	<0.01	CSB-47
11340	12893	20.1	<0.01	<0.01	<0.01	<0.01	CSB-47
11340	12893	14.6	0.15	<0.01	<0.01	<0.01	CSB-47
11340	12893	9.1	<0.01	<0.01	<0.01	1.1	CSB-47
11340	12893	5.1	<0.01	<0.01	<0.01	<0.01	CSB-47
11249	12871	27.8	90	0.07	0.32	<0.01	CSB-48
11249	12871	23.3	0.16	<0.01	<0.01	<0.01	CSB-48
11249	12871	21.3	2.1	<0.01	<0.01	<0.01	CSB-48
11249	12871	13.3	<0.01	<0.01	<0.01	<0.01	CSB-48
11249	12871	8.3	<0.01	<0.01	<0.01	<0.01	CSB-48
11087	12831	28.3	<0.01	<0.01	0.01	<0.01	CSB-49
11087	12831	24.8	<0.01	<0.01	<0.01	<0.01	CSB-49
11087	12831	14.8	<0.01	<0.01	<0.01	<0.01	CSB-49
11087	12831	4.8	<0.01	<0.01	<0.01	<0.01	CSB-49

This file uses z coordinates (versus depth) for all samples, therefore line 2 has the word Elevation. There are 50 samples a<0.01 5 analytes (chemicals) per sample.

Another example using depths from the top surface is:

X Coord	Y Coord	Depth	@@TOTHC	Boring	Top
Depth	feet				
37	1		ppm		
11856.72	12764.01	1	.057	CSB_67	1.7
11856.72	12764.01	8	.134	CSB_67	1.7
11856.72	12764.01	16	.081	CSB_67	1.7
11856.72	12764.01	20	.292	CSB_67	1.7
11856.72	12764.01	26	.066	CSB_67	1.7
11889.60	12772.20	2	1.762	CSB_23	1.3
11889.60	12772.20	4	.853	CSB_23	1.3
11889.60	12772.20	7	.941	CSB_23	1.3
11889.60	12772.20	15	10.467	CSB_23	1.3
11889.60	12772.20	16	488.460	CSB_23	1.3
11889.60	12772.20	22	410.900	CSB_23	1.3
11889.60	12772.20	26	.140	CSB_23	1.3
11939.19	12758.45	6	.175	CSB_70	3.7
11939.19	12758.45	15	.100	CSB_70	3.7
11939.19	12758.45	18	.430	CSB_70	3.7
11939.19	12758.45	26	.100	CSB_70	3.7

12002.80	12759.80	2	.321	CSB_24	1.2
12002.80	12759.80	4	.296	CSB_24	1.2
12002.80	12759.80	8	.179	CSB_24	1.2
12002.80	12759.80	13	0.000	CSB_24	1.2
12002.80	12759.80	17	.711	CSB_24	1.2
12002.80	12759.80	23	.864	CSB_24	1.2
12002.80	12759.80	28	.311	CSB_24	1.2
12085.15	12749.01	2	.104	CSW_71	4.6
12085.15	12749.01	6	.154	CSW_71	4.6
12085.15	12749.01	16	.732	CSW_71	4.6
12085.15	12749.01	26	.065	CSW_71	4.6
12146.70	12713.21	1	.027	CSB-72	2.1
12146.70	12713.21	7	.251	CSB-72	2.1
12146.70	12713.21	23	1.176	CSB-72	2.1
12199.70	12709.80	2	.043	CSB-12	6.0
12199.70	12709.80	4	.055	CSB-12	6.0
12199.70	12709.80	8	.031	CSB-12	6.0
12199.70	12709.80	12	.014	CSB-12	6.0
12199.70	12709.80	16	.018	CSB-12	6.0
12199.70	12709.80	23	.466	CSB-12	6.0
12199.70	12709.80	27	.197	CSB-12	6.0

This file has 37 samples in 7 boreholes. Since depth below the top surface is used instead of "Z" coordinates, line 2 contains the word Depth. Note that in this example there is only one analyte (e.g. chemistry) (property) value per line, but up to 300 could be included in which case line three of the file would read "37 300" a<0.01 we would have 299 more columns of numbers in this file!.

A analyte (e.g. chemistry) fence diagram file has the exact same format, except that the samples from each boring must occur in the order of connectivity along the fence, a<0.01 they should be sorted by increasing depth at each sample location.

Discussion of analyte (e.g. chemistry) Files for Fence Sections

analyte (e.g. chemistry) files to be used to create fence diagrams using the older krig_fence module, must contain only those borings that the user wishes to include on an individual cross section of the fence, in the order that they will be connected along the section. The result is that one .apdv file is produced for each cross section that will be included in the fence diagram, a<0.01 the data for borings at which the fences will intersect are included in each of the intersecting cross section files. When geology is included on the fence diagrams, the order of the borings in the analyte (e.g. chemistry) files must be identical to those in the geology files for each section. Generally, it is easiest to create the analyte (e.g. chemistry) file for a complete dataset, a<0.01 then subset the fence diagram files from the complete file.

AIDV: Analyte Interval Data File Format

This format allows you to specify the top and bottom elevations of well screens and one or more concentrations that were measured over that interval. This new format (.aidv) will allow you to

quickly visualize well screens in post_samples and automatically convert well screens to intelligently spaced samples along the screen interval for 3D (and 2D) kriging.

Format:

You may insert comment lines in C Tech Groundwater analyte (e.g. chemistry) (.aidv) input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

Line 1: You may include any header message here (that does not start with a '#' character) **unless** you wish to include analyte names for use by other EVS modules (e.g. data component name). The format for line 1 to enable chemical names is as follows

A. Placing a pair of '@' symbols triggers the use and display of chemical names (example @@VOC). Any characters up to the @@ characters are ignored, and only the first analyte name needs @@, after that the chemical names must be delimited by spaces,

B. The following rules for commas are implemented to accommodate comma delimited files and also for using chemical names which have a comma within (example 1,1-DCA). Commas following a name will not become a part of the name, but a comma in the middle of a text string will be included in the name. The recommended approach is to put a space before the names.

C. If you want a space in your analyte name, you may use underscores and EVS will convert underscores to spaces (example: Vinyl_Chloride in a .aidv file will be converted to 'r;Vinyl Chloride." Or you may surround the entire name in quotation marks (example: "Vinyl Chloride").

The advantages of using chemical names (attribute names of any type) are the following:

- many modules use analyte names instead of data component numbers,
- when writing EVS Field files (.eff, .efb, etc.), you will get analyte names instead of data component numbers.
- when querying your data set with post_sample's mouse interactivity, the analyte name is displayed.
- time-series data can be used and the appropriate time-step can be displayed.

Line 2: Specifications

- **Elevation/Depth Specifier:** The first item on line 2 must be the word *Elevation* or *Depth* (case insensitive) to denote whether well screen top and bottom elevations are true elevation or depth below ground surface.
- **Maximum Gap:** The second parameter in this line is a real number (not an integer) specifying the Max-Gap. Max-gap is the maximum distance between samples for kriging. When a screen interval's total length is less than max-gap, a single sample is placed at the center of the interval. If the screen interval is longer than max-gap, two or more equally spaced samples are distributed within the interval. The number of samples is equal to the interval divided by max-gap rounded up to an integer.
 - [note: if you set max gap too small, you effectively create over-sampling in z (relative to x-y) for your data. On the other hand, if you have multiple screen intervals with different z extents and depths, choosing the proper value for max-gap will ensure better 3D distributions. If max-gap is set very large, only one sample is placed at the center of each screen interval. If the screens are small relative to the thickness of the aquifer, a large max gap is OK. If the screens are long (30% or more) of the local thickness and there are nearby screens with different depths/lengths, you will need a smaller max-gap value. Viewing your screen intervals with the spheres ON will help assess the optimal value.

- **Coordinate Units:** After Depth/Elevation, include the units of your coordinates (e.g. *feet* or *meters*)

Line 3: Specifications

- **The first integer (n) is the number of well screens (rows of data) to follow. You may specify "All" instead to use all data lines in the file.**
- The second integer is the number of analyte (chemistry) values per well screen.
- The units of each data analyte column (e.g. *ppm* or *mg/l*).

Line 4: The first line of analyte interval (well screen) data must contain:

- X
- Y
- Well Screen Top
- Well Screen Bottom
- (one or more) Analyte Value(s) (chemistry or property)
- Well or Boring name. The boring name cannot contain spaces (recommend underscore "_" instead).
- Elevation of the top of the boring.

Boring name and top are optional parameters, but are used by many modules and it is highly recommended that you include this information in your file if possible. They are used by post_samples for posting tubes along borehole traces and for generating tubes which start from the ground surface of the borehole. Both krig_3d and krig_3d_geology will use this information to determine the Z spatial extent of your grids (krig_3d_geology will create a layer that begins at ground surface if this information is provided). Numbers and names can be separated by one comma and/or any number of spaces or tabs.

BLANK ENTRIES (CELLS) ARE NOT ALLOWED.

Please see the section on [Handling Non-Detects](#) for information on how to deal with samples whose concentration is below the detection limit. For any sample that is not detected you may enter any of the following. Please note that the first three flag words are not case sensitive, but must be spelled exactly as shown below.

- Prepend a less than sign < to the actual detection limit for that sample. This allows you to set the "Less Than Multiplier" in all modules that read .apdv files to a value such as 0.1 to 0.5 (10 to 50%). This is the preferred and most rigorous method.
- nondetect
- non-detect
- nd
- 0.0 (zero)

For files with multiple analytes such as the example below, if an analyte was not measured at a sample location, use any of the flags below to denote that this sample should be skipped for this analyte. Please note that these flag words are not case sensitive, but must be spelled exactly as shown below.

- missing
- unmeasured

- not-measured
- nm
- unknown
- unk
- na

[Example Files are here:](#)

AIDV: Analyte Interval Data File Format

This format allows you to specify the top and bottom elevations of well screens and one or more concentrations that were measured over that interval. This new format (.aidv) will allow you to quickly visualize well screens in post_samples and automatically convert well screens to intelligently spaced samples along the screen interval for 3D (and 2D) kriging.

Format:

You may insert comment lines in C Tech Groundwater analyte (e.g. chemistry) (.aidv) input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

Line 1: You may include any header message here (that does not start with a '#' character) **unless** you wish to include analyte names for use by other EVS modules (e.g. data component name). The format for line 1 to enable chemical names is as follows

A. Placing a pair of '@' symbols triggers the use and display of chemical names (example @@VOC). Any characters up to the @@ characters are ignored, and only the first analyte name needs @@, after that the chemical names must be delimited by spaces,

B. The following rules for commas are implemented to accommodate comma delimited files and also for using chemical names which have a comma within (example 1,1-DCA). Commas following a name will not become a part of the name, but a comma in the middle of a text string will be included in the name. The recommended approach is to put a space before the names.

C. If you want a space in your analyte name, you may use underscores and EVS will convert underscores to spaces (example: Vinyl_Chloride in a .aidv file will be converted to 'r;Vinyl Chloride." Or you may surround the entire name in quotation marks (example: "Vinyl Chloride").

The advantages of using chemical names (attribute names of any type) are the following:

- many modules use analyte names instead of data component numbers,
- when writing EVS Field files (.eff, .efb, etc.), you will get analyte names instead of data component numbers.
- when querying your data set with post_sample's mouse interactivity, the analyte name is displayed.
- time-series data can be used and the appropriate time-step can be displayed.

Line 2: Specifications

- **Elevation/Depth Specifier:** The first item on line 2 must be the word *Elevation* or *Depth* (case insensitive) to denote whether well screen top and bottom elevations are true elevation or depth below ground surface.
- **Maximum Gap:** The second parameter in this line is a real number (not an integer) specifying the Max-Gap. Max-gap is the maximum distance between samples for kriging. When a screen interval's total length is less than max-gap, a single sample is placed at the

center of the interval. If the screen interval is longer than max-gap, two or more equally spaced samples are distributed within the interval. The number of samples is equal to the interval divided by max-gap rounded up to an integer.

- [note: if you set max gap too small, you effectively create over-sampling in z (relative to x-y) for your data. On the other hand, if you have multiple screen intervals with different z extents and depths, choosing the proper value for max-gap will ensure better 3D distributions. If max-gap is set very large, only one sample is placed at the center of each screen interval. If the screens are small relative to the thickness of the aquifer, a large max gap is OK. If the screens are long (30% or more) of the local thickness and there are nearby screens with different depths/lengths, you will need a smaller max-gap value. Viewing your screen intervals with the spheres ON will help assess the optimal value.
- **Coordinate Units:** After Depth/Elevation, include the units of your coordinates (e.g. *feet* or *meters*)

Line 3: Specifications

- **The first integer (n) is the number of well screens (rows of data) to follow. You may specify "All" instead to use all data lines in the file.**
- The second integer is the number of analyte (chemistry) values per well screen.
- The units of each data analyte column (e.g. *ppm* or *mg/l*).

Line 4: The first line of analyte interval (well screen) data must contain:

- X
- Y
- Well Screen Top
- Well Screen Bottom
- (one or more) Analyte Value(s) (chemistry or property)
- Well or Boring name. The boring name cannot contain spaces (recommend underscore "_" instead).
- Elevation of the top of the boring.

Boring name and top are optional parameters, but are used by many modules and it is highly recommended that you include this information in your file if possible. They are used by post_samples for posting tubes along borehole traces and for generating tubes which start from the ground surface of the borehole. Both krig_3d and krig_3d_geology will use this information to determine the Z spatial extent of your grids (krig_3d_geology will create a layer that begins at ground surface if this information is provided). Numbers and names can be separated by one comma and/or any number of spaces or tabs.

BLANK ENTRIES (CELLS) ARE NOT ALLOWED.

Please see the section on [Handling Non-Detects](#) for information on how to deal with samples whose concentration is below the detection limit. For any sample that is not detected you may enter any of the following. Please note that the first three flag words are not case sensitive, but must be spelled exactly as shown below.

- Prepend a less than sign < to the actual detection limit for that sample. This allows you to set the "Less Than Multiplier" in all modules that read .apdv files to a value such as 0.1 to 0.5 (10 to 50%). This is the preferred and most rigorous method.
- nondetect

- non-detect
- nd
- 0.0 (zero)

For files with multiple analytes such as the example below, if an analyte was not measured at a sample location, use any of the flags below to denote that this sample should be skipped for this analyte. Please note that these flag words are not case sensitive, but must be spelled exactly as shown below.

- missing
- unmeasured
- not-measured
- nm
- unknown
- unk
- na

[Example Files are here:](#)

AIDV File Examples

An actual .aidv file could look like the following:

```
# This is a comment line....any line that starts with # is ignored
```

X	Y	Ztop	Zbot	@@TOHC	Bore	Top
Elevation	6.0	feet				
10	1	mg/l				
11086.52	12830.67	-13	-26	2.000	W-49	4.5
11199.04	12810.16	-18	-30	2.000	W-51	4
11298.00	12808.63	-12	-38	3600.	W-52	3
11566.34	12850.59	-14	-25	0.000	W-30	7.5
11251.30	12929.27	-24	-30	33000	W-75	2
11248.75	12870.91	-17	-22	5004.8	W-48	3
11340.49	12892.61	-11	-16	120.0	W-47	2.5
11340.49	12892.61	-22	-28	320.0	W-47	2.5
11338.00	12830.80	-13	-20	640.0	W-38	4
11401.73	12897.77	-36	-40	<0.300	W-45	4

This example file above (10_well_screens.aidv) has 10 well screens in 9 boreholes. Well W-47 has two different screen intervals. Note that line 2 contains the word Elevation and the number 6.0 which is the max-gap parameter. There are 10 rows of data and there is only one analyte value per line, but up to 300 could be included in a single file.

Analyte Time Files Format

Discussion of Analyte Time Files

Analyte time files contain 3-D coordinates (x, y, z) describing the locations of samples and values of one or more analytes or properties taken over a series of different times. Time files must conform to the ASCII formats described below and individual entries (coordinates or measurements) can be delimited by commas, spaces, or tabs. They must have either a **.sct** (Soil Chemistry Time) or **.gwt** (Ground Water Time) suffix to be selected in the file browsers of EVS modules. Each line of the file contains the coordinate data for one sampling location, or well screen, and any number of chemistry or property values. There are no limits on the number of borings and/or samples that can be included in these files, except that run times for execution of kriging do increase with a greater number of samples in the file.

Time data can be visualized independently (without geology data) or within a domain bounded by a geologic system. When a geologic domain is utilized for a 3-D visualization, a consistent coordinate system (the same projection and overlapping spatial extents) must be used for both the chemistry and geology. The boring and sample locations in the time files do not have to correspond to those in the geology files, except that only those contained within or proximal to the spatial domain of the geology will be used for the kriging.

If the posting of borings and sample locations are to honor the topography of the site, the chemistry files also must contain the top surface elevation of each boring.

Format:

You may insert comment lines anywhere in Analyte time files. Comments must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

The format of chemistry time files is substantially different from other analyte file formats (.apdv or .aidv) used in EVS. These differences include **required** analyte name and unit on line one (no other information allowed), and no need to specify the number of samples or number of analytes and times.

Line 1: This line contains the name of each analyte. After every analyte has been listed the analyte units are then required for each analyte. Analyte Units are **REQUIRED** for time chemistry files.

Line 2: This line contains the mapping of the analytes to a specific date. This is done by listing the analyte name followed by a pipe character "|" and then followed by the sampling date. There should be one of these mappings for every column of data in the file. If you want a space in your analyte name you may enclose the entire name and date in quotation marks (example: "Vinyl Chloride|6/1/2004"). Optionally the analyte name may be omitted and just a date used, in this case the first analyte name listed on line one will be used.

It is required that the order of analyte-date columns be from oldest to newest for each analyte.

The date format is dependent on your REGIONAL SETTINGS on your computer (control panel).

C Tech uses the SHORT DATE and SHORT TIME formats.

If the date/time works in Excel it will likely work in EVS.

For most people in the U.S., this would not be 24 hour clock so you would need:

"m/d/yyyy hh:mm:ss AM" or "m/d/yyyy hh:mm:ss PM"

Also, you MUST put the date/time in quotes if you use more than just date (i.e. if there are spaces in the total date/time).

Line 3: This line must contain the word Elevation or Depth to denote whether sample elevations are true elevation or depth below ground surface. If actual elevations are used (a right-handed coordinate system), then this parameter should be *Elevation*; if depths below the top surface elevation are used, then this parameter should be *Depth*.

FOR GWT FILESONLY: the second parameter in this line is a real number (not an integer) specifying the Max-Gap in the same units as your coordinate data. Max-gap is the maximum distance between samples for kriging. When a screen interval's total length is less than max-gap, a single sample is placed at the center of the interval. If the screen interval is longer than max-gap, two or more equally spaced samples are distributed within the interval. The number of samples is equal to the interval divided by max-gap rounded up to an integer.

The last value on this line should be the units of your coordinates (e.g. feet or meters), or the flag word reproject.

Lines 4+: *The lines of sample data:* The content of these lines varies whether the file is a SCT or GWT file. GWT files have an additional column of elevation (Z) data to allow for specification of the top and bottom of each screen interval, whereas SCT files specify the location of a POINT sample (requiring only a single elevation).

X, Y, Z (for Chemistry files or Well Screen Top), **Well Screen Bottom** for groundwater chemistry files), (one or more) **Analyte Value(s)** (chemistry or property), **Boring name**, and **Elevation of the Top Of The Boring** (optional).

There are several flag words available for **missing values** these include:

- a. unmeasured
- b. not-measured
- c. nm
- d. missing
- e. unknown
- f. unk
- g. na

For **non-detect** samples the following flag words are available:

- a. Prepend a less than sign < to the actual detection limit for that sample. This allows you to set the "Less Than Multiplier" in all modules that read .apdv files to a value such as 0.1 to 0.5 (10 to 50%). This is the preferred and most rigorous method.
- b. nondetect or
- c. non-detect
- d. nd

The boring name cannot contain spaces (recommend underscore "_" instead), unless surrounded by quotation marks (example: "B 1"). The optional boring name and top are needed only by the post_samples module for posting tubes along borehole traces and for generating tubes which start from the ground surface of the borehole.

Numbers and names can be separated by one comma and/or any number of spaces or tabs. **BLANK ENTRIES (CELLS) ARE NOT ALLOWED.**

When Top of Boring elevations are given, they must be provided for all lines of the file.

```
#Soil Chemistry Time File Example (SCT)
"ethane""ethylene""mg/kg""ug/kg"
"ethane|6/8/1976""ethylene|6/8/1976""ethane|1/12/1979" "ethylene|1/12/1979"
"ethylene|3/16/1981"
Elevation meters
12008 12431 22.9 22 Unk 21 500 0 CSB-39 30.4
11271 13105 18.9 0 0 0 2800 0 CSB-40 35.9
10652 13857 23.4 0 0 0 290 0 CSB-41 28.1
9904 14522 18.4 0 0 0 Unk Unk CSB-42 22.8
9029 15283 37.9 0 0 0 23 0 CSB-43 30.1
```

For the GWT file below, those items that are unique to GWT (vs. SCT) are in [BLUE](#).

```
#Ground WaterChemistry Time File Example (GWT)
"ethane""ethylene""mg/kg""ug/kg"
"ethane|6/8/1976""ethylene|6/8/1976""ethane|1/12/1979" "ethylene|1/12/1979"
"ethylene|3/16/1981"
Elevation3.0meters
12008 12431 22.9 15.2 22 Unk 21 500 0 CSB-39 30.4
11271 13105 18.9 12.5 0 0 0 2800 0 CSB-40 35.9
10652 13857 23.4 19.0 0 0 0 290 0 CSB-41 28.1
9904 14522 18.4 11.8 0 0 0 Unk Unk CSB-42 22.8
9029 15283 37.9 30.3 0 0 0 23 0 CSB-43 30.1
```

Time Domain Analyte Data

We recommend that analyte files which represent data collected over time use either the APDV or AIDV format and include data for only a single analyte

We do not recommend using the [SCT or GWT](#) formats.

When using APDV or AIDV files for time domain data, the following rules apply:

- Include data for only a **single** analyte
- Group measurements taken over a few days or even weeks into the same DATE GROUP. If your entire site is re-sampled every 3 months, do not separately list each day when a particular well is sampled.
- The "analyte name" for each column of data representing a Date Group should be the average date for that sampling event. The date must be in the Windows standard short date format. In the United States that is typically MM/DD/YYYY (e.g. 11/08/2003 for November 8, 2003)
- The data file cannot specify the actual analyte name (e.g. benzene). However, the modules which deal with time domain data have the ability to specify the actual name and units.
- Date groups need not be at equal time intervals.

Time Domain AIDV Example File

x	y	ztop	zbot	@@1/1/2001	5/1/2001	8/1/200
Elevation	10	m				
98	3			mg/l	mg/l	mg/l
2772536.7	331635.8	886.5	866.5	6	5	5
2772554.6	331635.2	987.4	967.4	0.71	5	5
2772601.5	333091.7	862.1	852.1	0.71	5	5
2772610.4	333100.5	950.6	930.6	0.71	1	1
2772830.1	336800.0	853.5	833.5	190	130	125
2772982.4	333214.1	955.3	935.3	5	5	5
2773014.8	331825.0	954.0	934.0	180	nm	nm
2773014.8	331825.0	881.9	861.9	150	nm	nm
2773069.9	332631.8	888.1	868.1	35	36	40
2773076.0	332138.7	959.5	949.5	48	48	55
2773087.1	332138.3	994.4	974.4	0.71	1	10
2773091.3	332611.7	784.4	684.4	5	5	5
2773104.2	332134.5	887.6	867.6	440	480	500
2773129.1	332136.9	736.0	686.0	0.71	5	5
2773146.2	333741.7	862.5	842.5	300	330	240
2773149.9	333225.7	1020.1	990.1	2650	2500	2350
2773156.3	333244.4	1017.8	987.8	750	690	13500
2773156.6	333219.8	1002.0	982.0	200	200	200
2773157.7	333579.1	946.1	941.1	0.71	2	5
2773159.4	333587.1	1006.4	986.4	0.71	1	1
2773165.1	333262.3	1013.1	993.1	10000	10000	30000
2773182.8	333309.7	1009.2	989.2	45000	43000	53500
2773192.1	333368.0	796.2	779.2	5	5	5
2773192.5	333361.4	870.7	853.7	19	11	22
2773196.2	333647.9	936.4	921.4	29	100	130
2773236.4	333568.8	1016.6	1016.6	10	9	nm
2773253.6	333567.2	1017.0	1017.0	800	800	770
2773266.3	335344.6	908.3	888.3	6	nm	nm
2773290.3	335351.9	833.0	813.0	610	nm	nm
2773307.6	333207.6	1005.5	985.5	2000	1900	1500
2773308.9	333198.4	945.6	940.6	180	180	200
2773323.3	333554.5	1016.3	996.3	750	510	7700
2773324.5	333353.1	947.0	942.0	750	750	675
2773325.8	333349.2	1009.5	989.5	100	91	85
2773326.6	333529.3	1012.4	992.4	1100	1000	810
2773328.0	333518.5	1021.1	1001.1	800	730	700
2773439.9	333202.0	994.0	974.0	90	88	80
2773441.7	333077.6	1009.3	989.3	410	410	400

2773446.4	333203.9	946.0	941.0	5	5	5
2773457.6	333081.2	890.2	870.2	400	380	275
2773462.8	333364.4	1000.7	980.7	11000	11000	10550
2773477.3	333524.2	941.8	936.8	5	5	5
2773480.4	333449.2	1010.0	980.0	7000	6600	5750
2773480.5	333522.5	1006.9	986.9	350	350	375
2773482.1	333669.2	946.5	931.5	0.71	1	5
2773541.1	333784.9	876.4	826.4	230	240	290
2773570.2	333713.2	1013.2	989.9	0.71	1	5
2773571.6	333770.9	853.5	833.5	100	110	160
2773572.2	332825.6	1008.8	988.8	25	26	27
2773573.4	332844.1	903.4	883.4	125	120	175
2773575.8	333740.1	738.3	688.3	0.71	5	5
2773620.0	332116.7	1019.5	996.5	5	5	5
2773630.2	332116.9	959.4	939.4	1	1	5
2773663.4	332966.1	1003.8	983.8	700	610	625
2773672.4	332971.5	889.9	869.9	75	65	240
2773688.4	332956.9	743.3	693.3	5	5	5
2773689.4	333385.8	997.9	977.9	370	190	420
2773692.6	333066.4	882.0	862.0	800	750	950
2773708.8	333065.2	1007.8	987.8	250000	220000	260000
2773713.9	333494.8	860.6	849.1	100	270	190
2773714.1	333523.8	1006.5	986.5	36	36	35
2773717.9	333532.7	941.2	936.2	31	31	30
2773730.5	331660.3	906.0	886.0	0.71	nm	nm
2773732.8	331687.1	950.3	930.3	0.71	nm	nm
2773735.5	333543.7	784.5	734.5	0.71	5	5
2773760.8	333319.1	936.3	931.3	8	8	8
2773763.3	333330.4	997.1	977.1	59262	57805	56348
2773765.6	333309.4	1013.0	963.0	770	820	890
2773797.1	333060.9	1008.8	988.8	97	97	95
2773899.9	333080.3	967.1	957.1	10	12	12
2773902.7	333097.7	915.8	905.8	5	9	12
2774022.9	333742.9	882.9	832.9	46	95	77
2774033.8	333513.5	986.9	974.9	2	2	2
2774051.8	333512.9	1027.5	1005.5	2100	2100	2100
2774065.2	333730.6	983.5	963.5	5	250	5
2774073.1	333738.4	858.5	838.5	0.71	0.71	3
2774073.7	334671.8	947.1	937.1	0.71	1	4
2774076.5	333728.3	823.7	823.7	0.71	2	2
2774083.0	332103.9	866.4	856.4	98	85	100
2774085.3	333736.6	996.9	973.5	16	25	37
2774087.2	334674.8	792.4	782.4	22	20	19

2774094.7	333745.8	936.3	924.5	16	14	50
2774186.2	331604.2	873.9	853.9	0.71	5	5
2774187.3	333087.0	911.3	891.3	16	22	25
2774194.8	333100.9	973.6	953.6	5	5	5
2774324.1	334101.7	922.3	912.3	0.71	1	5
2774332.3	333623.1	881.4	861.4	0.71	3	5
2774338.3	333327.8	998.8	981.5	0.71	2	5
2774341.9	333638.3	1022.6	999.4	5	5	5
2774344.3	333870.5	862.2	852.2	5	5	6
2774352.8	333882.0	898.1	888.1	0.71	1	4
2774664.2	334463.8	845.0	835.0	0.71	1	5
2774677.0	334462.1	961.0	951.0	130	120	135
2774820.0	333352.3	883.5	863.5	0.71	5	5
2774995.8	336287.5	694.9	644.9	0.71	5	5
2774995.9	336310.6	831.8	811.8	30	31	34
2775092.1	334397.8	946.4	936.4	10	9	10
2777126.6	336231.0	809.7	789.7	0.71	5	5

Analyte Time Files Format

Discussion of Analyte Time Files

Analyte time files contain 3-D coordinates (x, y, z) describing the locations of samples and values of one or more analytes or properties taken over a series of different times. Time files must conform to the ASCII formats described below and individual entries (coordinates or measurements) can be delimited by commas, spaces, or tabs. They must have either a **.sct** (Soil Chemistry Time) or **.gwt** (Ground Water Time) suffix to be selected in the file browsers of EVS modules. Each line of the file contains the coordinate data for one sampling location, or well screen, and any number of chemistry or property values. There are no limits on the number of borings and/or samples that can be included in these files, except that run times for execution of kriging do increase with a greater number of samples in the file.

Time data can be visualized independently (without geology data) or within a domain bounded by a geologic system. When a geologic domain is utilized for a 3-D visualization, a consistent coordinate system (the same projection and overlapping spatial extents) must be used for both the chemistry and geology. The boring and sample locations in the time files do not have to correspond to those in the geology files, except that only those contained within or proximal to the spatial domain of the geology will be used for the kriging.

If the posting of borings and sample locations are to honor the topography of the site, the chemistry files also must contain the top surface elevation of each boring.

Format:

You may insert comment lines anywhere in Analyte time files. Comments must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

The format of chemistry time files is substantially different from other analyte file formats (.apdv or .aidv) used in EVS. These differences include **required** analyte

name and unitson line one (no other information allowed), and no need to specify the number of samples or number of analytesandtimes.

Line 1: This line contains the name of each analyte. After every analyte has been listed the analyte units are then required for each analyte. Analyte Units are **REQUIRED** for time chemistry files.

Line 2: This line contains the mapping of the analytes to a specific date. This is done by listing the analyte name followed by a pipe character "|" and then followed by the sampling date. There should be one of these mappings for every column of data in the file. If you want a space in your analyte name you may enclose the entire name and date in quotation marks (example: "Vinyl Chloride|6/1/2004"). Optionally the analyte name may be omitted and just a date used, in this case the first analyte name listed on line one will be used.

It is required that the order of analyte-date columns be from oldest to newest for each analyte.

The date format is dependent on your REGIONAL SETTINGS on your computer (control panel).

C Tech uses the SHORT DATE and SHORT TIME formats.

If the date/time works in Excel it will likely work in EVS.

For most people in the U.S., this would not be 24 hour clock so you would need:

"m/d/yyyy hh:mm:ss AM" or "m/d/yyyy hh:mm:ss PM"

Also, you **MUST** put the date/time in quotes if you use more than just date (i.e. if there are spaces in the total date/time).

Line 3: This line must contain the word Elevation or Depth to denote whether sample elevations are true elevation or depth below ground surface. If actual elevations are used (a right-handed coordinate system), then this parameter should be *Elevation*; if depths below the top surface elevation are used, then this parameter should be *Depth*.

FOR GWT FILESONLY:the second parameter in this line is a real number (not an integer) specifying the Max-Gap in the same units as your coordinate data. Max-gap is the maximum distance between samples for kriging. When a screen interval's total length is less than max-gap, a single sample is placed at the center of the interval. If the screen interval is longer than max-gap, two or more equally spaced samples are distributed within the interval. The number of samples is equal to the interval divided by max-gap rounded up to an integer.

The last value on this line should be the units of your coordinates (e.g. feet or meters), or the flag word reproject.

Lines 4+: *The lines of sample data:* The content of these lines varies whether the files is a SCT or GWT file. GWT files have an additional column of elevation (Z) data to allow for specification of the top and bottom of each screen interval, whereas SCT files specify the location of a POINT sample (requiring only a single elevation).

X, Y, Z (for Chemistry files or Well Screen Top), **Well Screen Bottom** for groundwater chemistry files) , (one or more) **Analyte Value(s)** (chemistry or property), **Boring name**, and **Elevation of the Top Of The Boring** (optional).

There are several flag words available for **missing values** these include:

- a. unmeasured

- b. not-measured
- c. nm
- d. missing
- e. unknown
- f. unk
- g. na

For **non-detect** samples the following flag words are available:

- a. Prepend a less than sign < to the actual detection limit for that sample. This allows you to set the "Less Than Multiplier" in all modules that read .apdv files to a value such as 0.1 to 0.5 (10 to 50%). This is the preferred and most rigorous method.
- b. nondetect or
- c. non-detect
- d. nd

The boring name cannot contain spaces (recommend underscore "_" instead), unless surrounded by quotation marks (example: "B 1"). The optional boring name and top are needed only by the post_samples module for posting tubes along borehole traces and for generating tubes which start from the ground surface of the borehole. Numbers and names can be separated by one comma and/or any number of spaces or tabs. **BLANK ENTRIES (CELLS) ARE NOT ALLOWED.**

When Top of Boring elevations are given, they must be provided for all lines of the file.

```
#Soil Chemistry Time File Example (SCT)
"ethane""ethylene""mg/kg""ug/kg"
"ethane|6/8/1976""ethylene|6/8/1976""ethane|1/12/1979" "ethylene|1/12/1979"
"ethylene|3/16/1981"
Elevation meters
12008 12431 22.9 22 Unk 21 500 0 CSB-39 30.4
11271 13105 18.9 0 0 0 2800 0 CSB-40 35.9
10652 13857 23.4 0 0 0 290 0 CSB-41 28.1
9904 14522 18.4 0 0 0 Unk Unk CSB-42 22.8
9029 15283 37.9 0 0 0 23 0 CSB-43 30.1
```

For the GWT file below, those items that are unique to GWT (vs. SCT) are in **BLUE**.

```
#Ground WaterChemistry Time File Example (GWT)
"ethane""ethylene""mg/kg""ug/kg"
"ethane|6/8/1976""ethylene|6/8/1976""ethane|1/12/1979" "ethylene|1/12/1979"
"ethylene|3/16/1981"
Elevation3.0meters
12008 12431 22.9 15.2 22 Unk 21 500 0 CSB-39 30.4
11271 13105 18.9 12.5 0 0 0 2800 0 CSB-40 35.9
10652 13857 23.4 19.0 0 0 0 290 0 CSB-41 28.1
```

```
9904 14522 18.4 11.8 0 0 0 Unk Unk CSB-42 22.8
9029 15283 37.9 30.3 0 0 0 23 0 CSB-43 30.1
```

Pre Geology File: Lithology

The ASCII pregeology file name must have a .pgf suffix to be selected in the module's file browser. This file type represents raw (uninterpreted) 3D boring logs representing lithology. This format is used by:

- make_geo_hierarchy
- post_samples
- krig_3d_geology (to extract a top and bottom surface to build a single layer)
- indicator_geology for [Geologic Indicator Kriging](#) (GIK).
- adaptive_indicator_krig

You may insert comment lines in C Tech Pre Geology (.pgf) input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

The pre-geology file format is used to represent raw 3D boring logs. We also refer to this geologic data format as "uninterpreted". This is not meant to imply that no form of geologic evaluation or interpretation has occurred. On the contrary, it is required that someone categorizes the materials on the site and in each boring.

Data Concept:

- A PGF file can be considered a **group of file sections** where each section represents **the lithology for individual borings** (wells).
- It is essential to use the same ID for the ground surface (first line) as for the bottom of the first observed material (second line) in each section (boring). If a different material ID is used a synthetic point will be added between the ground and first observed material. This will be reported for the first five occurrences.
 - Think about the PGF file as a shorthand way of specifying intervals. The first line is the FROM. The second is the TO.
- Please note that the data for each boring must be sorted (by you) from beginning to end (normally top to bottom).
- We cannot sort this data for you because some borings may turn to horizontal or even upwards.
 - **It is your responsibility to make sure that the data is in the proper order.**
 - **It is your responsibility to make sure that each boring ID corresponds to a unique X, Y location if there would be overlapping Z (or depth) intervals. In other words, there cannot be overlapping boring definitions.**
 - If the data is unsorted, and within a boring the direction between two values varies by more than 90 degrees, an error will be reported.

FILE FORMAT:

- **Line 1:** May contain any header message, but cannot be left blank or commented. There is no information content in this line.
- **Line 2:** Line 2 contains the declaration of Elevation or Depth, the definitions of Lithology IDs and Names, and coordinate units.
 - **Elevation/Depth Specifier:** This line must contain the word *Elevation* or *Depth* (case insensitive) to specify whether well screen top and bottom elevations are true elevation or depth below ground surface.
 - **Depth** forces the otherwise optional ground surface elevation column to be required. Depths given in column 3 are distances below the ground surface elevation in the last column (column 6). If the top surface is omitted, a value of 0.0 will be assumed and a warning message will be printed to the EVS Information Window.
 - **IDs and Names:** Line 2 should contain Lithology IDs and corresponding names for each material. Each Name is explicitly associated with its corresponding Lithology ID and the pairs are delimited by a pipe symbol "|".
 - Though it is generally advisable, IDs need not be sequential and may be any integer values. This allow for a unified set of Lithology IDs and Names to be applied to a large site where models create for sub-sites may not have all materials.
 - The number of (material) IDs and Names **MUST** be equal to the number of Lithology IDs specified in the data section. Each material ID present in the data section must have corresponding Lithology IDs and Names. If there are four materials represented in your .pgf file, there should be at least four IDs and Names on line two.
 - The order of Lithology IDs and Names will determine the order that they appear in legends. The IDs do not need to be sequential.
 - You can specify additional IDs and Names, which are not in the data and those will appear on legends.
 - **Coordinate Units:** You should include the units of your coordinates (e.g. *feet* or *meters*). If this is included it must follow the names associated with each Lithology ID.
- **Line 3:** Must be the number of lines of data (n) to follow. For each boring, there is one line for the ground surface and one line for the bottom of each observed lithologic unit. Therefore the total number of lines in the file should be equal to the number of borings PLUS the sum of the number of materials observed in each boring.
- **Line 4:** First line of sample data. X, Y, Z, "Lithology ID", Boring name, and Ground surface elevation. The Ground surface elevation is an optional parameter which is required if *Depth* is specified on line 2. If depths are used (instead of elevations) the top surface should be in the same coordinate system. Depths are relative to the Ground surface (which is assumed at 0.0 if the Ground surface is not defined). The boring name cannot contain spaces unless the entire name is surrounded in quotation marks (example "Boring 1D"). One comma and/or any number of spaces or tabs can separate numbers and name.
- **Line 3+n:** is the last line of the file.

AN EXAMPLE FILE FOLLOWS:

```

Pregeology      file
Elevation      1|Silt      2|Fill      3|Clay      4|Sand      5|Gravel      ft

```

17

11086.5	12830.7	4.5	1	B-49
11086.5	12830.7	-3.8	1	B-49
11086.5	12830.7	-21	2	B-49
11086.5	12830.7	-26	3	B-49
11086.5	12830.7	-42	4	B-49
11086.5	12830.7	-55	5	B-49
11199	12600	4	1	B-51
11199	12600	-5	1	B-51
11199	12600	-20	2	B-51
11199	12600	-25	3	B-51
11199	12600	-39	4	B-51
11199	12600	-53	5	B-51
11259.7	12819.3	2	1	B-46
11259.7	12819.3	-7.5	1	B-46
11259.7	12819.3	-27	3	B-46
11259.7	12819.3	-40	4	B-46
11259.7	12819.3	-53	5	B-46

Pre Geology File: Lithology

The ASCII pregeology file name must have a .pgf suffix to be selected in the module's file browser. This file type represents raw (uninterpreted) 3D boring logs representing lithology. This format is used by:

- make_geo_hierarchy
- post_samples
- krig_3d_geology (to extract a top and bottom surface to build a single layer)
- indicator_geology for [Geologic Indicator Kriging](#) (GIK).
- adaptive_indicator_krig

You may insert comment lines in C Tech Pre Geology (.pgf) input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

The pre-geology file format is used to represent raw 3D boring logs. We also refer to this geologic data format as "uninterpreted". This is not meant to imply that no form of geologic evaluation or interpretation has occurred. On the contrary, it is required that someone categorizes the materials on the site and in each boring.

Data Concept:

- A PGF file can be considered a **group of file sections** where each section represents **the lithology for individual borings** (wells).
- It is essential to use the same ID for the ground surface (first line) as for the bottom of the first observed material (second line) in each section (boring). If a different material ID is used a synthetic point will be added between the ground and first observed material. This will be reported for the first five occurrences.
 - Think about the PGF file as a shorthand way of specifying intervals. The first line is the FROM. The second is the TO.
- Please note that the data for each boring must be sorted (by you) from beginning to end (normally top to bottom).
- We cannot sort this data for you because some borings may turn to horizontal or even upwards.
 - **It is your responsibility to make sure that the data is in the proper order.**
 - **It is your responsibility to make sure that each boring ID corresponds to a unique X, Y location if there would be overlapping Z (or depth) intervals. In other words, there cannot be overlapping boring definitions.**
 - If the data is unsorted, and within a boring the direction between two values varies by more than 90 degrees, an error will be reported.

FILE FORMAT:

- **Line 1:** May contain any header message, but cannot be left blank or commented. There is no information content in this line.
- **Line 2:** Line 2 contains the declaration of Elevation or Depth, the definitions of Lithology IDs and Names, and coordinate units.
 - **Elevation/Depth Specifier:** This line must contain the word *Elevation* or *Depth* (case insensitive) to specify whether well screen top and bottom elevations are true elevation or depth below ground surface.
 - **Depth** forces the otherwise optional ground surface elevation column to be required. Depths given in column 3 are distances below the ground surface elevation in the last column (column 6). If the top surface is omitted, a value of 0.0 will be assumed and a warning message will be printed to the EVS Information Window.
 - **IDs and Names:** Line 2 should contain Lithology IDs and corresponding names for each material. Each Name is explicitly associated with its corresponding Lithology ID and the pairs are delimited by a pipe symbol "|".
 - Though it is generally advisable, IDs need not be sequential and may be any integer values. This allow for a unified set of Lithology IDs and Names to be applied to a large site where models create for sub-sites may not have all materials.
 - The number of (material) IDs and Names **MUST** be equal to the number of Lithology IDs specified in the data section. Each material ID present in the data section must have corresponding Lithology IDs and Names. If there are

four materials represented in your .pgf file, there should be at least four IDs and Names on line two.

- The order of Lithology IDs and Names will determine the order that they appear in legends. The IDs do not need to be sequential.
- You can specify additional IDs and Names, which are not in the data and those will appear on legends.
- **Coordinate Units:** You should include the units of your coordinates (e.g. *feet* or *meters*). If this is included it must follow the names associated with each Lithology ID.
- **Line 3:** Must be the number of lines of data (n) to follow. For each boring, there is one line for the ground surface and one line for the bottom of each observed lithologic unit. Therefore the total number of lines in the file should be equal to the number of borings PLUS the sum of the number of materials observed in each boring.
- **Line 4:** First line of sample data. X, Y, Z, "Lithology ID", Boring name, and Ground surface elevation. The Ground surface elevation is an optional parameter which is required if *Depth* is specified on line 2. If depths are used (instead of elevations) the top surface should be in the same coordinate system. Depths are relative to the Ground surface (which is assumed at 0.0 if the Ground surface is not defined). The boring name cannot contain spaces unless the entire name is surrounded in quotation marks (example "Boring 1D"). One comma and/or any number of spaces or tabs can separate numbers and name.
- **Line 3+n:** is the last line of the file.

AN EXAMPLE FILE FOLLOWS:

```

Pregeology      file
Elevation      1|Silt      2|Fill      3|Clay      4|Sand      5|Gravel      ft
17
11086.5        12830.7      4.5        1          B-49
11086.5        12830.7      -3.8       1          B-49
11086.5        12830.7      -21        2          B-49
11086.5        12830.7      -26        3          B-49
11086.5        12830.7      -42        4          B-49
11086.5        12830.7      -55        5          B-49
11199          12600        4          1          B-51
11199          12600        -5         1          B-51
11199          12600        -20        2          B-51
11199          12600        -25        3          B-51
11199          12600        -39        4          B-51
11199          12600        -53        5          B-51
11259.7        12819.3      2          1          B-46
11259.7        12819.3      -7.5       1          B-46
11259.7        12819.3      -27        3          B-46

```

11259.7	12819.3	-40	4	B-46
11259.7	12819.3	-53	5	B-46

PGF File Examples

In the (very short) example file below, please note that the Lithology IDs and Names are not ordered by increasing ID number. The order that you specify the Lithology IDs and Names determines the order that is used for exploding the lithologic materials and the ordering in legends. Also notice that Lithology ID 3 is specified in line 2, but not present in the data. Silty-Sand will be shown in the legend, but will not be present in the borings displayed with post_samples nor any model created with this data.

```
EAST NORTH TOP-BOT MATERIAL-ID WELL_ID
Elevation 4|Sand 5|Gravel 1|Clay 2|Silt 3|Silty-sand meters
11
2085487.12 322869.95 31 4 AW-3
2085487.12 322869.95 -1 4 AW-3
2085487.12 322869.95 -3 2 AW-3
2085108.47 323363.89 32 4 MW-10A
2085108.47 323363.89 20 4 MW-10A
2085108.47 323363.89 12 5 MW-10A
2085079.22 323361.25 32 4 MW-10B
2085079.22 323361.25 20 4 MW-10B
2085266.93 323410.05 32 4 MW-11A
2085266.93 323410.05 14 4 MW-11A
2085266.93 323410.05 7 1 MW-11A
```

In the realistic example below, IDs are listed in ascending order and this order on Line 2 will be the order used for exploding materials and legends.

```
Pre-Geology File for Initial
Painting Facility soil investigation
Elevation1|Silt 2|Fill13|Clay4|Sand 5|Gravel ft
144
11086.5 12830.74.5 2 B-49
11086.5 12830.7-3.8 2 B-49
11086.5 12830.7-21.0 1 B-49
11086.5 12830.7-26.0 3 B-49
11086.5 12830.7-42.0 5 B-49
11086.5 12830.7-55.0 4 B-49
11199.0 12810.24.0 2 B-51
11199.0 12810.2-5.0 2 B-51
11199.0 12810.2-20.0 1 B-51
11199.0 12810.2-25.0 3 B-51
11199.0 12810.2-39.0 5 B-51
11199.0 12810.2-53.0 4 B-51
```

11259.7	12819.32.0	2	B-46
11259.7	12819.3-7.5	2	B-46
11259.7	12819.3-20.5	1	B-46
11259.7	12819.3-27.0	3	B-46
11259.7	12819.3-40.0	5	B-46
11259.7	12819.3-53.0	4	B-46
11298.0	12808.63.0	2	B-52
11298.0	12808.6-6.0	2	B-52
11298.0	12808.6-19.0	1	B-52
11298.0	12808.6-25.8	3	B-52
11298.0	12808.6-41.8	5	B-52
11298.0	12808.6-55.0	4	B-52
11414.4	12781.12.0	2	B-34
11414.4	12781.1-6.0	2	B-34
11414.4	12781.1-20.5	1	B-34
11414.4	12781.1-28.0	3	B-34
11414.4	12781.1-42.0	5	B-34
11414.4	12781.1-55.0	4	B-34
11427.0	12780.96.5	2	B-42
11427.0	12780.9-7.0	2	B-42
11427.0	12780.9-23.0	1	B-42
11427.0	12780.9-28.5	3	B-42
11427.0	12780.9-38.5	5	B-42
11427.0	12780.9-51.0	4	B-42
11496.3	12753.61.5	2	B-53
11496.3	12753.6-7.5	2	B-53
11496.3	12753.6-20.0	1	B-53
11496.3	12753.6-28.8	3	B-53
11496.3	12753.6-38.8	5	B-53
11496.3	12753.6-51.0	4	B-53
11209.4	12993.92.0	2	B-57
11209.4	12993.9-3.0	2	B-57
11209.4	12993.9-23.0	1	B-57
11209.4	12993.9-27.5	3	B-57
11209.4	12993.9-37.5	5	B-57
11209.4	12993.9-51.0	4	B-57
11251.3	12929.32.0	2	B-75
11251.3	12929.3-2.5	2	B-75
11251.3	12929.3-22.0	1	B-75
11251.3	12929.3-28.0	3	B-75
11251.3	12929.3-40.0	5	B-75
11251.3	12929.3-53.0	4	B-75
11248.8	12870.93.0	2	B-48

11248.8	12870.9-3.5	2	B-48
11248.8	12870.9-22.0	1	B-48
11248.8	12870.9-28.5	3	B-48
11248.8	12870.9-36.3	5	B-48
11248.8	12870.9-50.0	4	B-48
11211.9	12710.82.0	2	B-50
11211.9	12710.8-6.5	2	B-50
11211.9	12710.8-22.5	1	B-50
11211.9	12710.8-27.5	3	B-50
11211.9	12710.8-37.5	5	B-50
11211.9	12710.8-51.0	4	B-50
11302.0	13079.74.5	2	B-58
11302.0	13079.7-3.5	2	B-58
11302.0	13079.7-21.9	1	B-58
11302.0	13079.7-26.0	3	B-58
11302.0	13079.7-38.0	5	B-58
11302.0	13079.7-51.0	4	B-58
11286.8	13026.72.0	2	B-59
11286.8	13026.7-5.0	2	B-59
11286.8	13026.7-23.0	1	B-59
11286.8	13026.7-29.0	3	B-59
11286.8	13026.7-37.0	5	B-59
11286.8	13026.7-50.0	4	B-59
11309.0	12949.04.0	2	B-56
11309.0	12949.0-2.5	2	B-56
11309.0	12949.0-22.0	1	B-56
11309.0	12949.0-28.3	3	B-56
11309.0	12949.0-38.3	5	B-56
11309.0	12949.0-52.0	4	B-56
11340.5	12892.62.5	2	B-47
11340.5	12892.6-2.5	2	B-47
11340.5	12892.6-20.0	1	B-47
11340.5	12892.6-23.0	3	B-47
11340.5	12892.6-38.0	5	B-47
11340.5	12892.6-52.0	4	B-47
11338.0	12830.84.0	2	B-38
11338.0	12830.8-8.8	2	B-38
11338.0	12830.8-23.0	1	B-38
11338.0	12830.8-26.5	3	B-38
11338.0	12830.8-36.5	5	B-38
11338.0	12830.8-50.0	4	B-38
11393.5	12948.93.5	2	B-60
11393.5	12948.9-3.8	2	B-60

11393.5	12948.9-23.0	1	B-60
11393.5	12948.9-27.0	3	B-60
11393.5	12948.9-39.0	5	B-60
11393.5	12948.9-52.0	4	B-60
11401.7	12897.84.0	2	B-45
11401.7	12897.8-2.0	2	B-45
11401.7	12897.8-22.0	1	B-45
11401.7	12897.8-27.5	3	B-45
11401.7	12897.8-37.5	5	B-45
11401.7	12897.8-51.0	4	B-45
11416.9	12819.52.5	2	B-44
11416.9	12819.5-5.0	2	B-44
11416.9	12819.5-21.0	1	B-44
11416.9	12819.5-28.5	3	B-44
11416.9	12819.5-38.5	5	B-44
11416.9	12819.5-51.0	4	B-44
11381.7	12747.51.5	2	B-33
11381.7	12747.5-4.0	2	B-33
11381.7	12747.5-21.5	1	B-33
11381.7	12747.5-25.8	3	B-33
11381.7	12747.5-42.8	5	B-33
11381.7	12747.5-56.0	4	B-33
11410.3	12724.70.5	2	B-43
11410.3	12724.7-4.5	2	B-43
11410.3	12724.7-22.9	1	B-43
11410.3	12724.7-25.0	3	B-43
11410.3	12724.7-36.0	5	B-43
11410.3	12724.7-49.0	4	B-43
11566.3	12850.62.5	2	B-30
11566.3	12850.6-5.0	2	B-30
11566.3	12850.6-21.0	1	B-30
11566.3	12850.6-28.5	3	B-30
11566.3	12850.6-38.5	5	B-30
11566.3	12850.6-51.0	4	B-30
11586.3	13050.611.5	2	B-31
11586.3	13050.61.0	2	B-31
11586.3	13050.6-11.0	1	B-31
11586.3	13050.6-18.5	3	B-31
11586.3	13050.6-26.5	5	B-31
11586.3	13050.6-47.0	4	B-31
11086.3	13090.68.5	2	B-32
11086.3	13090.6-1.0	2	B-32
11086.3	13090.6-14.0	1	B-32

11086.3	13090.6-23.5	3	B-32
11086.3	13090.6-32.5	5	B-32
11086.3	13090.6-48.0	4	B-32

PGF File Example with Depth

Easting	Northing	Depth	Lithology_ID		Boring_ID
Ground					
Depth	0 Overburden	1 Lava	2 Sulfide	3 Rhyolite	4 Mafic_Intrusion
m					
29					
192731.10	1389503.04	0.00	0	1	2132.53
192731.10	1389503.04	6.75	0	1	2132.53
192731.10	1389503.04	101.00	1	1	2132.53
192731.10	1389503.04	383.10	3	1	2132.53
192731.10	1389503.04	403.70	2	1	2132.53
192731.10	1389503.04	490.00	4	1	2132.53
192674.55	1389639.67	0.00	0	22	2126.28
192674.55	1389639.67	4.30	0	22	2126.28
192674.55	1389639.67	100.60	1	22	2126.28
192674.55	1389639.67	156.00	3	22	2126.28
192674.55	1389639.67	209.40	2	22	2126.28
192674.55	1389639.67	496.20	4	22	2126.28
192987.12	1389624.87	0.00	0	13	2130.64
192987.12	1389624.87	6.98	0	13	2130.64
192987.12	1389624.87	91.40	1	13	2130.64
192987.12	1389624.87	397.40	2	13	2130.64
192987.12	1389624.87	425.80	4	13	2130.64
192930.95	1389745.48	0.00	0	14	2128.68
192930.95	1389745.48	6.70	0	14	2128.68
192930.95	1389745.48	80.40	1	14	2128.68
192930.95	1389745.48	246.40	3	14	2128.68
192930.95	1389745.48	250.60	2	14	2128.68
192930.95	1389745.48	459.60	4	14	2128.68
192582.47	1389677.63	0.00	0	23	2123.62
192582.47	1389677.63	6.80	0	23	2123.62
192582.47	1389677.63	101.20	1	23	2123.62
192582.47	1389677.63	138.70	3	23	2123.62
192582.47	1389677.63	160.00	2	23	2123.62
192582.47	1389677.63	499.60	4	23	2123.62

LPDV Lithology Point Data Value File Format

The LPDV lithology file format is the most general, free-form format to represent lithology information.

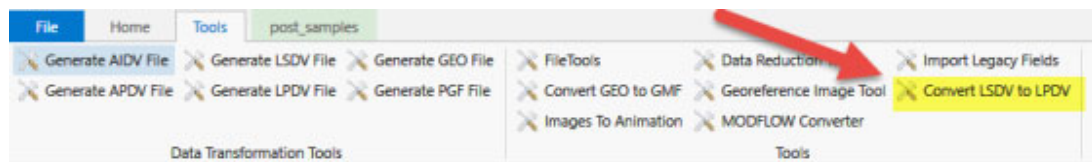
To understand the rationale for its existence, you must understand that when creating lithologic models (smooth or block) with indicator_geology, the internal

kriging operations require lithologic data in point format. Therefore all other lithology file formats (.PGF and .LSDV) are converted to points based on the *PGF Refine Distance*. LPDV files are not refined since we use the point data directly.

LPDV files have the following advantages and disadvantages:

- **Advantages**

- Is not based on borings or screens
- It can represent surficial lithology data (material definitions at ground without depth)
- LSDV formats can be converted to LPDV and merged with other LPDV data. This is done with this Tool



- **Disadvantages**

- Files tend to be larger since a single screen can represent many points
- Displaying boring based data is more limited
- LPDV files cannot be further refined.
 - If your points are too coarse or too fine, you cannot easily change this.

An explanation of the file format follows:

- Any line beginning with # is a comment (in the file below, the first and third lines are comments and could be deleted without loss of function)
- Entries on lines can be separated by commas, spaces and/or tabs.
- The First (uncommented) line:
 1. Must begin with Elevation or Depth
 - For the data section shown below, **when Depth is specified, replace Z with Depth and columns 5 & 6 are required**
 2. Then each material specified in the file is listed as:
"Material|number|Material|name"
 3. The end of the line has the coordinate units (typically m [*meters*] or ft [*feet*]), OR the [REPROJECT](#) tag.
- The next line begins the data section. *You do not need to specify the number of data lines.* The 9 entries in each line are all **required** and therefore must be:
 - Columns 1-3: X, Y, Z
 - Column 4: Material-number (these are integers which should begin with zero on line 1)
 - Column 5: Boring ID : OPTIONAL, however, if any line has this then all lines must have it.

- Column 6: Ground Surface Elevation: OPTIONAL, however, it can only be included if Boring_ID is included and if any line has this then all lines must have it.

Below is a snippet of the file "lithology.lpdv" in the "Exporting Data to C Tech File Formats" folder of Studio Projects. This file was converted from lithology.lsdv.

```
Elevation "0|SAND" "1|SANDSTONE" "2|GRAVEL" "3|MUDSTONE" "4|LIMESTONE"
"5|SILTSTONE" "6|MADEGROUND" "7|CONGLOMERATE" "8|GYPSUM" "9|CLAY"
"10|SILT" m
736133.267249 1637594.558440 1190.194000 0 720B0001 1190.200000
736133.267249 1637594.558440 1189.806000 0 720B0001 1190.200000
736133.267249 1637594.558440 1189.794000 1 720B0001 1190.200000
736133.267249 1637594.558440 1189.006000 1 720B0001 1190.200000
736133.267249 1637594.558440 1188.206000 1 720B0001 1190.200000
736133.267249 1637594.558440 1188.194000 0 720B0001 1190.200000
736133.267249 1637594.558440 1187.806000 0 720B0001 1190.200000
736133.267249 1637594.558440 1187.794000 1 720B0001 1190.200000
736133.267249 1637594.558440 1187.256000 1 720B0001 1190.200000
736133.267249 1637594.558440 1186.706000 1 720B0001 1190.200000
736133.267249 1637594.558440 1186.694000 0 720B0001 1190.200000
736133.267249 1637594.558440 1186.156000 0 720B0001 1190.200000
736133.267249 1637594.558440 1185.606000 0 720B0001 1190.200000
736133.267249 1637594.558440 1185.594000 2 720B0001 1190.200000
736133.267249 1637594.558440 1184.906000 2 720B0001 1190.200000
736133.267249 1637594.558440 1184.206000 2 720B0001 1190.200000
736133.267249 1637594.558440 1184.194000 1 720B0001 1190.200000
736133.267249 1637594.558440 1183.453000 1 720B0001 1190.200000
736133.267249 1637594.558440 1182.706000 1 720B0001 1190.200000
736133.267249 1637594.558440 1181.959000 1 720B0001 1190.200000
736133.267249 1637594.558440 1181.206000 1 720B0001 1190.200000
736133.267249 1637594.558440 1181.194000 1 720B0001 1190.200000
```

LSDV Lithology Screen Data Value File Format

The LSDV lithology file format can be used as a more feature rich replacement for the older PGF format. It has the following advantages:

- Fully supports non-vertical borings
- Supports missing intervals and lithology data which does not begin at ground surface
- Provides an Explicit definition of each lithologic interval

An explanation of the file format follows:

- Any line beginning with # is a comment (in the file below, the first and third lines are comments and could be deleted without loss of function)

- Entries on lines can be separated by commas, spaces and/or tabs.
- The First (uncommented) line:
 1. Must begin with Elevation or Depth
 - For the data section shown below, when Depth is specified, replace Z with Depth
 2. Then each material specified in the file is listed as:
"Material|^{number}|Material|^{name}"
 3. The end of the line has the coordinate units (typically m [*meters*] or ft [*feet*]), OR the [REPROJECT](#) tag.
- The next line begins the data section. *You do not need to specify the number of data lines.* The 9 entries in each line are all **required** and therefore must be:
 - Columns 1-3: X_{top}, Y_{top}, Z_{top}
 - Columns 4-6: X_{bottom}, Y_{bottom}, Z_{bottom}
 - Column 7: Material-number (these are integers which should begin with zero on line 1)
 - Column 8: Boring ID
 - Column 9: Ground Surface Elevation
- We cannot sort this data for you because some borings may turn to horizontal or even upwards.
 - **It is your responsibility to make sure that the data is in the proper order.**
 - **It is your responsibility to make sure that each boring ID corresponds to a unique X, Y location if there would be overlapping Z (or depth) intervals. In other words, there cannot be overlapping boring definitions**

Below is a snippet of the file "lithology.lsdv" in the "Exporting Data to C Tech File Formats" folder of Studio Projects.

```
# C Tech Data Exporter generated LSDV File from LITHOLOGY-DATA.XLSX
(05/01/2020 15:31:03)
Elevation "0|SAND" "1|SANDSTONE" "2|GRAVEL" "3|MUDSTONE" "4|LIMESTONE"
"5|SILTSTONE" "6|MADEGROUND" "7|CONGLOMERATE" "8|GYPSUM" "9|CLAY"
"10|SILT" m
# Columns [DEMO]: "East" "North" "Elev-Top" "East" "North" "Elev-Bot"
"Lithology" "Boring" "Ground_Surface"
736133.267249, 1637594.55844, 1190.2, 736133.267249, 1637594.55844,
1189.8, 0, "720B0001", 1190.2
736133.267249, 1637594.55844, 1189.8, 736133.267249, 1637594.55844,
1188.2, 1, "720B0001", 1190.2
736133.267249, 1637594.55844, 1188.2, 736133.267249, 1637594.55844,
1187.8, 0, "720B0001", 1190.2
736133.267249, 1637594.55844, 1187.8, 736133.267249, 1637594.55844,
1186.7, 1, "720B0001", 1190.2
```

736133.267249, 1637594.55844, 1186.7, 736133.267249, 1637594.55844,
1185.6, 0, "720B0001", 1190.2
736133.267249, 1637594.55844, 1185.6, 736133.267249, 1637594.55844,
1184.2, 2, "720B0001", 1190.2
736133.267249, 1637594.55844, 1184.2, 736133.267249, 1637594.55844,
1181.2, 1, "720B0001", 1190.2
736133.267249, 1637594.55844, 1181.2, 736133.267249, 1637594.55844,
1176.2, 1, "720B0001", 1190.2
736133.267249, 1637594.55844, 1176.2, 736133.267249, 1637594.55844,
1174.9, 1, "720B0001", 1190.2
736286.268053, 1637647.55834, 1191.2, 736286.268053, 1637647.55834,
1190.2, 0, "720B0002", 1191.2
736286.268053, 1637647.55834, 1190.2, 736286.268053, 1637647.55834,
1189.8, 0, "720B0002", 1191.2
736286.268053, 1637647.55834, 1189.8, 736286.268053, 1637647.55834,
1187.2, 1, "720B0002", 1191.2
736286.268053, 1637647.55834, 1187.2, 736286.268053, 1637647.55834,
1186.2, 2, "720B0002", 1191.2
736286.268053, 1637647.55834, 1186.2, 736286.268053, 1637647.55834,
1184.1, 1, "720B0002", 1191.2
736286.268053, 1637647.55834, 1184.1, 736286.268053, 1637647.55834,
1182.2, 1, "720B0002", 1191.2
736286.268053, 1637647.55834, 1182.2, 736286.268053, 1637647.55834,
1175.9, 1, "720B0002", 1191.2
737193.272266, 1637709.55665, 1190.2, 737193.272266, 1637709.55665,
1189.2, 0, "720B0003", 1190.2
737193.272266, 1637709.55665, 1189.2, 737193.272266, 1637709.55665,
1188.2, 0, "720B0003", 1190.2
737193.272266, 1637709.55665, 1188.2, 737193.272266, 1637709.55665,
1184.2, 0, "720B0003", 1190.2
737193.272266, 1637709.55665, 1184.2, 737193.272266, 1637709.55665,
1181.9, 0, "720B0003", 1190.2
737193.272266, 1637709.55665, 1181.9, 737193.272266, 1637709.55665,
1179.2, 1, "720B0003", 1190.2
737193.272266, 1637709.55665, 1179.2, 737193.272266, 1637709.55665,
1178.9, 0, "720B0003", 1190.2
737193.272266, 1637709.55665, 1178.9, 737193.272266, 1637709.55665,
1178.2, 1, "720B0003", 1190.2
737193.272266, 1637709.55665, 1178.2, 737193.272266, 1637709.55665,
1177.9, 0, "720B0003", 1190.2
737193.272266, 1637709.55665, 1177.9, 737193.272266, 1637709.55665,
1177.2, 1, "720B0003", 1190.2
737193.272266, 1637709.55665, 1177.2, 737193.272266, 1637709.55665,
1174.7, 1, "720B0003", 1190.2

GEO: Borehole Geology Stratigraphy

Geology data files basically contain horizontal and vertical coordinates, which describe the geometry of geologic features of the region being modeled. The files must be in ASCII format and can be delimited by commas, spaces, or tabs. Borehole Geology files must have a .geo suffix to be selected in the file browsers of EVS modules. The z values in .geo files can represent either elevation or depth, although elevation is generally the easiest to work with. When chemistry or property data is to be utilized along with geologic data for a 3-D visualization, a consistent coordinate system must be used in both sets of data.

Geology files should also specify the geologic layer material (color) number and layer names. This provides a mechanism to color multiple (not necessarily adjacent) layers as the same material.

Borehole Geology files (.geo suffix) must have the same number of entries for each boring location, so that every geologic layer in the system is represented in each boring. However, EVS allows flags to be included in the .geo files to allow automated processing of data in systems where geologic layers are not present in all locations (i.e., the layers "pinch out"). Also, EVS accommodates borings that were not extended deep enough to encounter layers that the scientist knows are present in the system. The use of these flags greatly facilitates the production of .geo data files, and minimizes the amount of manual interpretation the scientist must do before using EVS to analyze, understand, and refine a geologic model. For layers that pinch out, a flag of *pinch* can be used for automated estimation of the "depth" to the bottom of that layer. Entering this flag is essentially equivalent to entering the bottom depth of the layer directly above the pinched out layer (which is also an acceptable way to prepare the file). When EVS encounters this flag in a file, it assigns the pinched out layer a zero thickness at this location. For borings that do not extend to the depths of geologic layers in the system, a flag of *short* is included in the file for all layers below the depth of the boring. Including this flag notifies EVS to ignore the presence of this boring when kriging the surface of the layers below the total depth of the boring.

Format:

The file name must have a .geo suffix to be selected in the module's file browser. The format below is the same for all EVS modules which read geology files:

You may insert comment lines in C Tech Geology (.geo) files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file. There is an important exception. The first non-commented line of the file is the header line (line 1 described below).

Line 1: Any header message: Except that:

- \$W or \$G as the first two characters signifies a special geology file which contains unrelated surfaces such as historical water tables. These flags turn off checking for corrupt geology file formats (situations where lower surfaces are above higher surfaces) and automatically turn off kriging in thickness space.
- Line one cannot be BLANK

Line 2: Elevation/Depth Specifier:

- The only REQUIRED item on this line in the Elevation or Depth Specifier.

- This line should contain the word *Elevation* or *Depth* (case insensitive) to denote whether sample elevations are true elevation or depth below ground surface.
- If set to Depth all surface descriptions for layer bottoms are entered as depths relative to the top surface. This is a common means of collecting sample coordinates for borings.
- Note that the flags such as pinch or short are not modified.
- Line 2 SHOULD contain names for each geologic surface (and therefore the layers created by them).
 - There are some rules that must be observed.
 - The number of surface (layer) names MUST be equal to the number of surfaces. Therefore, if naming layers, the first name should correspond to the top surface and each subsequent name will refer to the surface that defines the bottom of that layer.
 - A name containing a space MUST be enclosed in quotation marks example ("Silty Sand"). Names should be limited to upper and lower case letters, numerals, hyphen "-" and underscore "_". The names defined on line two will appear as the cell set name in the explode_and_scale or select_cells modules. Names should be separated with spaces, commas or tabs.
- Line 2: After the names, include the units of your coordinates (e.g. *feet* or *meters*). It must follow the names for each material number.

Line 3: The first integer (n) is the number of lines to follow. The second integer (m) is the number of geologic layer depths plus one (for the top surface). The 3rd and subsequent numbers are the colors for each surface in your model. Layers are colored by the color of the surface that defines their bottoms. The first two color numbers should be the same (top and bottom of the first layer).

When used with fence_geology, the order of the borings determines the connectivity of the fence diagram and must match the chemistry file for krig_fence.

Note that X and Y corresponding to Eastings and Northings are used. Be careful not to reverse these.

Line 4: First line of sample data. X, Y, top surface, and "m" depths or elevations to the bottom of each geologic layer. Coordinates, elevations (depths) and boring name can be separated by one comma and/or any number of spaces or tabs.

Two different flag parameters are included to accommodate special conditions. These flags are

A: Boring terminates early or surface information is missing. This flag class is used to identify that a boring did not continue deep enough to find the bottom of a geologic layer, OR that a section of a core sample is missing (lost, damaged, etc.) and that no determination of the location of this surface can be made from this boring. This is distinctly different than a surface (layer) that is not present because it has been determined that it has pinched out. The flags that are used for this class are [note: all flags are case insensitive, but spelling is critical]:

- missing
- unknown
- unk
- na
- short
- terminated
- term

In the sample file below, BOR-24 was not deep enough to reach to the bottom of the Lsand (lower sand) layer or the gravel layer. Rather than use the bottom of the boring (a meaningless number), the short flag is used so that this boring will not be used to determine the bottom of these two layers. Similarly BOR-72 is not deep enough to be used in determining the bottom of the last (Gravel) layer. The flags that are used for this class are [note: all flags are case insensitive, but spelling is critical]:

B: This flag class is used to identify that a geologic layer is not present because it has pinched out for this particular boring. It can be "thought of" as numerically equivalent to using the value one column to the left.

However, now that krig_3d_geology includes special treatment for the *pinchflag*, using the value to the left is not strictly equivalent.

- pinch
- pinched
- pinch-out

Note that several layers pinch out in borings WEL-67, BOR-23, BOR-70 and BOR-24, so the ***pinch*** flag was used for these layer's entries instead of any numerical value.

IMPORTANT: There are two important things to consider when using the flags above:

1. It is wholly inappropriate to have a pinch follow a short. Pinch denotes that the layer above is zero thickness. It is equivalent to using the numeric value to the left. However if it were to follow a short (unknown) it would be meaningless since the short is interpreted to be missing information.
2. If your last defined surface has fewer than 3 numeric values (with all the rest being missing/short), it will be poorly defined since it takes 3 points to define a plane. **If there are no numeric values the surface cannot be created.**

...

Line 3+n is the last line of the file.

AN EXAMPLE FILE FOLLOWS:

X	Y	TOP	BOT_1	BOT_2	BOT_3	BOT_4	BOT_5	BOT_6	BOT_7	Boring
Depth	Top	Fill	SiltySand	Clay	Sand	Silt	Sand	Gravel	feet	
7	8	5	5	3	1	4	2	4	6	

11856.7	12764.0	0	5.0	18.2	23.5	pinch	pinch	69.0	105.0	WEL-67
11889.6	12772.2	0	1.5	17.0	22.0	pinch	pinch	63.0	105.0	BOR-23
11939.1	12758.4	0	2.5	16.0	25.5	pinch	pinch	63.0	105.0	BOR-70
12002.8	12759.8	0	1.0	17.0	27.0	pinch	47.0	short	short	BOR-24
12085.1	12749.0	0	1.0	17.5	25.7	45.7	pinch	68.0	105.0	WEL-71
12146.7	12713.2	0	1.0	18.0	26.5	32.5	39.5	65.0	short	BOR-72
12199.7	12709.8	0	1.0	16.5	22.5	27.5	35.5	70.0	105.0	WEL-12

This file has 7 boreholes with 8 entries (surfaces) per borehole, corresponding to the top surface and the bottom depths of 7 geologic layers. Note that the fourth and sixth layers are both designated to be material 4. This allows you to easily create layers with the same material the same color.

Other Examples of Geologic Input Files

Example of a .geo file for sedimentary layers and lenses (containing pinchouts)

[Example of a .geo file for Dipping Strata](#)

Geologic_File_Example_Outcrop_of_Dipping_Strata

GEO: Borehole Geology Stratigraphy

Geology data files basically contain horizontal and vertical coordinates, which describe the geometry of geologic features of the region being modeled. The files must be in ASCII format and can be delimited by commas, spaces, or tabs. Borehole Geology files must have a .geo suffix to be selected in the file browsers of EVS modules. The z values in .geo files can represent either elevation or depth, although elevation is generally the easiest to work with. When chemistry or property data is to be utilized along with geologic data for a 3-D visualization, a consistent coordinate system must be used in both sets of data.

Geology files should also specify the geologic layer material (color) number and layer names. This provides a mechanism to color multiple (not necessarily adjacent) layers as the same material.

Borehole Geology files (.geo suffix) must have the same number of entries for each boring location, so that every geologic layer in the system is represented in each boring. However, EVS allows flags to be included in the .geo files to allow automated processing of data in systems where geologic layers are not present in all locations (i.e., the layers "pinch out"). Also, EVS accommodates borings that were not extended deep enough to encounter layers that the scientist knows are present in the system. The use of these flags greatly facilitates the production of .geo data files, and minimizes the amount of manual interpretation the scientist must do before using EVS to analyze, understand, and refine a geologic model. For layers that pinch out, a flag of *pinch* can be used for automated estimation of the "depth" to the bottom of that layer. Entering this flag is essentially equivalent to entering the bottom depth of the layer directly above the pinched out layer (which is also an acceptable way to prepare the file). When EVS encounters this flag in a file, it assigns the pinched out layer a zero thickness at this location. For borings that do not extend to the depths of geologic layers in the system, a flag of *short* is included in the file for all layers below the depth of the boring. Including this flag notifies EVS to ignore the presence of this boring when kriging the surface of the layers below the total depth of the boring.

Format:

The file name must have a .geo suffix to be selected in the module's file browser. The format below is the same for all EVS modules which read geology files:

You may insert comment lines in C Tech Geology (.geo) files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file. There is an important exception. The first non-commented line of the file is the header line (line 1 described below).

Line 1: Any header message: Except that:

- \$W or \$G as the first two characters signifies a special geology file which contains unrelated surfaces such as historical water tables. These flags turn off checking for corrupt geology file formats (situations where lower surfaces are above higher surfaces) and automatically turn off kriging in thickness space.
- Line one cannot be BLANK

Line 2: Elevation/Depth Specifier:

- The only REQUIRED item on this line in the Elevation or Depth Specifier.
 - This line should contain the word *Elevation* or *Depth* (case insensitive) to denote whether sample elevations are true elevation or depth below ground surface.
 - If set to Depth all surface descriptions for layer bottoms are entered as depths relative to the top surface. This is a common means of collecting sample coordinates for borings.
 - Note that the flags such as pinch or short are not modified.
- Line 2 SHOULD contain names for each geologic surface (and therefore the layers created by them).
 - There are some rules that must be observed.
 - The number of surface (layer) names MUST be equal to the number of surfaces. Therefore, if naming layers, the first name should correspond to the top surface and each subsequent name will refer to the surface that defines the bottom of that layer.
 - A name containing a space MUST be enclosed in quotation marks example ("Silty Sand"). Names should be limited to upper and lower case letters, numerals, hyphen "-" and underscore "_". The names defined on line two will appear as the cell set name in the explode_and_scale or select_cells modules. Names should be separated with spaces, commas or tabs.
- Line 2: After the names, include the units of your coordinates (e.g. *feet* or *meters*). It must follow the names for each material number.

Line 3: The first integer (n) is the number of lines to follow. The second integer (m) is the number of geologic layer depths plus one (for the top surface). The 3rd and subsequent numbers are the colors for each surface in your model. Layers are colored by the color of the surface that defines their bottoms. The first two color numbers should be the same (top and bottom of the first layer).

When used with `fence_geology`, the order of the borings determines the connectivity of the fence diagram and must match the chemistry file for `krig_fence`.

Note that X and Y corresponding to Eastings and Northings are used. Be careful not to reverse these.

Line 4: First line of sample data. X, Y, top surface, and "m" depths or elevations to the bottom of each geologic layer. Coordinates, elevations (depths) and boring name can be separated by one comma and/or any number of spaces or tabs.

Two different flag parameters are included to accommodate special conditions. These flags are

A: Boring terminates early or surface information is missing. This flag class is used to identify that a boring did not continue deep enough to find the bottom of a geologic layer, OR that a section of a core sample is missing (lost, damaged, etc.) and that no determination of the location of this surface can be made from this boring. This is distinctly different than a surface (layer) that is not present because it has been determined that it has pinched out. The flags that are used for this class are [note: all flags are case insensitive, but spelling is critical]:

- missing
- unknown
- unk
- na
- short
- terminated
- term

In the sample file below, BOR-24 was not deep enough to reach to the bottom of the Lsand (lower sand) layer or the gravel layer. Rather than use the bottom of the boring (a meaningless number), the short flag is used so that this boring will not be used to determine the bottom of these two layers. Similarly BOR-72 is not deep enough to be used in determining the bottom of the last (Gravel) layer. The flags that are used for this class are [note: all flags are case insensitive, but spelling is critical]:

B: This flag class is used to identify that a geologic layer is not present because it has pinched out for this particular boring. It can be "thought of" as numerically equivalent to using the value one column to the left.

However, now that `krig_3d_geology` includes special treatment for the *pinch* flag, using the value to the left is not strictly equivalent.

- pinch
- pinched
- pinch-out

Note that several layers pinch out in borings WEL-67, BOR-23, BOR-70 and BOR-24, so the ***pinch*** flag was used for these layer's entries instead of any numerical value.

IMPORTANT: There are two important things to consider when using the flags above:

1. It is wholly inappropriate to have a pinch follow a short. Pinch denotes that the layer above is zero thickness. It is equivalent to using the numeric value to the left. However if it were to follow a short (unknown) it would be meaningless since the short is interpreted to be missing information.
2. If your last defined surface has fewer than 3 numeric values (with all the rest being missing/short), it will be poorly defined since it takes 3 points to define a plane. **If there are no numeric values the surface cannot be created.**

...

Line 3+n is the last line of the file.

AN EXAMPLE FILE FOLLOWS:

X	Y	TOP	BOT_1	BOT_2	BOT_3	BOT_4	BOT_5	BOT_6	BOT_7	Boring
Depth	Top	Fill	SiltySand	Clay	Sand	Silt	Sand	Gravel	feet	
7	8	5	5	3	1	4	2	4	6	
11856.7	12764.0	0	5.0	18.2	23.5	pinch	pinch	69.0	105.0	WEL-67
11889.6	12772.2	0	1.5	17.0	22.0	pinch	pinch	63.0	105.0	BOR-23
11939.1	12758.4	0	2.5	16.0	25.5	pinch	pinch	63.0	105.0	BOR-70
12002.8	12759.8	0	1.0	17.0	27.0	pinch	47.0	short	short	BOR-24
12085.1	12749.0	0	1.0	17.5	25.7	45.7	pinch	68.0	105.0	WEL-71
12146.7	12713.2	0	1.0	18.0	26.5	32.5	39.5	65.0	short	BOR-72
12199.7	12709.8	0	1.0	16.5	22.5	27.5	35.5	70.0	105.0	WEL-12

This file has 7 boreholes with 8 entries (surfaces) per borehole, corresponding to the top surface and the bottom depths of 7 geologic layers. Note that the fourth and sixth layers are both designated to be material 4. This allows you to easily create layers with the same material the same color.

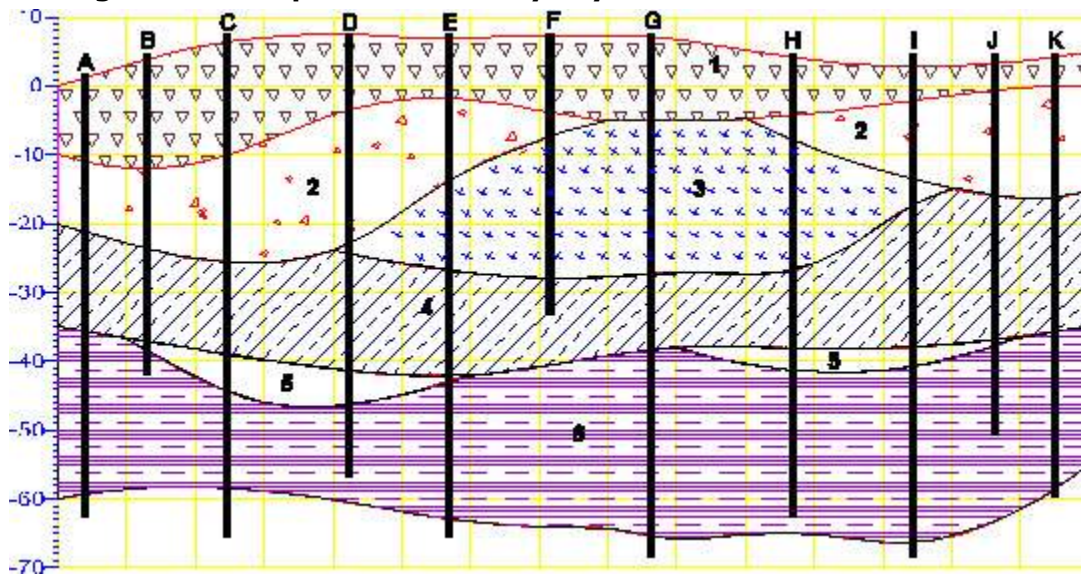
Other Examples of Geologic Input Files

Example of a .geo file for sedimentary layers and lenses (containing pinchouts)

[Example of a .geo file for Dipping Strata](#)

Geologic_File_Example_Outcrop_of_Dipping_Strata

Geologic File Example: Sedimentary Layers & Lenses



Both example files below represent valid forms for the geology file associated with the above figure. For file 1, line 2 of the file is "1", therefore all surface elevations are entered as actual elevations relative to a fixed reference such as sea level (not depths) and the relationship between x, y, and elevation must be a right handed coordinate system. Note that X and Y corresponding to Eastings and Northings are used. Be careful not to reverse these.

Two special flags are used to accommodate special conditions. These flags are pinch and short. Pinch is used to identify that a geologic layer is not present (pinched out) for a particular boring. It is equivalent to using the value one column to the left. Short is used to identify that a boring did not extend to the bottom of a geologic layer. In the sample file below, boring C was not deep enough to reach to the bottom of the layer 3 or any subsequent layers. Rather than use the bottom of the boring (a meaningless number), the *short* flag is used so that this boring will not be used to determine the bottom of these layers.

File 1:

X	Y	TOP	BOT_1	BOT_2	BOT_3	BOT_4	BOT_5	BOT_6	NAME
Elev		Top	FILL	SH	SS	SD	SLS	GR	feet
11	7	1	1	2	3	4	5	6	
5	3	3	-11.5	-22	pinch	-36	pinch	-59	A
13	5	3.5	-12	-22.5	pinch	-36.8	-37.5	short	B
24	7	5	-11	-24	pinch	-38.5	-43	-58.6	C
42	2	8	-3	-22	-23	-41.5	-46	short	D
57	11	7	-2	-13	-26.5	-42	-43.5	-63	E
72	14	7	-3	-8	-27.6	short	short	short	F
85	19	5.7	-5	pinch	-26.6	-38.3	pinch	-65	G
107	23	4.2	-5	-8	-26	-38	-41	short	H
123	35	2.2	-3	-13	-16.9	-37.5	-41	-66	I

136	24	3	-1.5	-15	pinch	-37	-37.5	short	J
145	18	4	0	-15.7	pinch	-36.3	pinch	-58	K

For file 2 line 2 of the file is Depth", therefore all surface descriptions for layer bottoms are entered as depths relative to the top surface elevations. This is a common means of collecting sample coordinates for borings. Note that the flags (pinch and short) are not affected by using depths versus elevations.

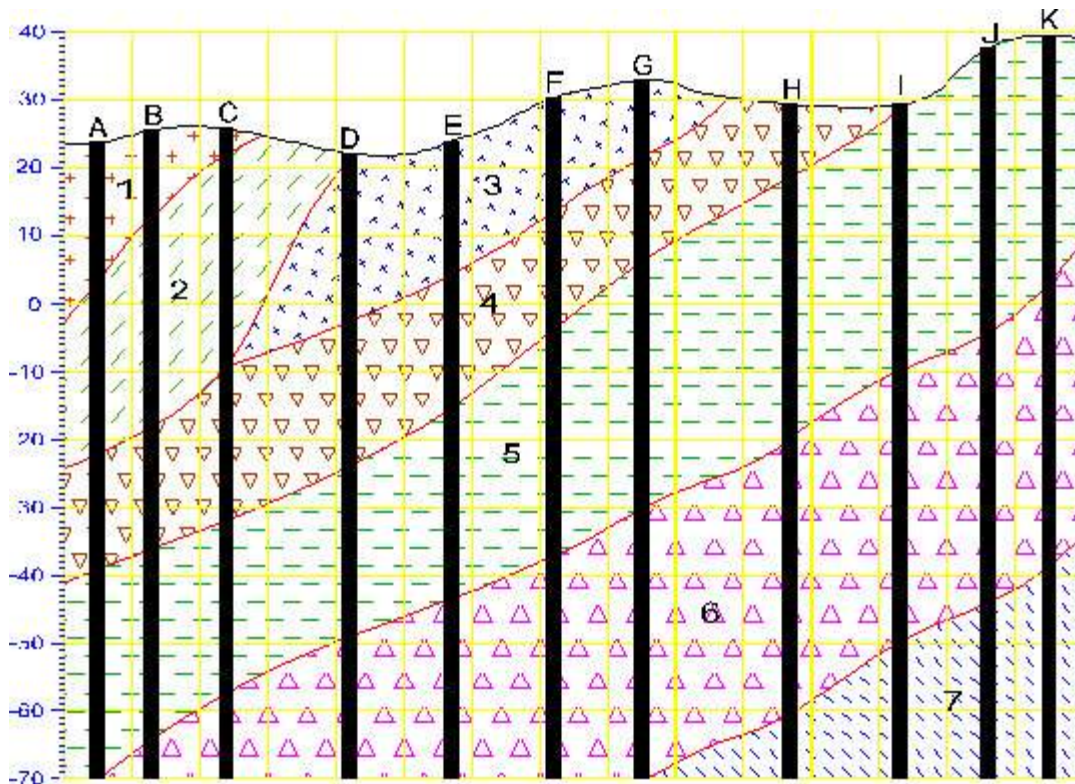
File 2:

X	Y	TOP	BOT_1	BOT_2	BOT_3	BOT_4	BOT_5	BOT_6	NAME
Depth		Top	FILL	SH	SS	SD	SLS	GR	feet
11	7	1	1	2	3	4	5	6	
5	3	3	14.5	25	pinch	39	pinch	62	A
13	5	3.5	15.5	26	pinch	40.3	41	short	B
24	7	5	16	29	pinch	43.5	48	63.6	C
42	2	8	11	30	31	49.5	54	short	D
57	11	7	9	20	33.5	49	50.5	70	E
72	14	7	10	15	34.6	short	short	short	F
85	19	5.7	10.7	pinch	32.3	44	pinch	70.7	G
107	23	4.2	9.2	12.2	30.2	42.2	45.2	short	H
123	35	2.2	5.2	15.2	19.1	39.7	43.2	68.2	I
136	24	3	4.5	18	pinch	40	40.5	short	J
145	18	4	4	19.7	pinch	40.3	pinch	62	K

There is no numerical equivalent to using the `short` flag. It causes the kriging modules to select only those borings with valid data for computing the surfaces of each layer.

Geologic File Example: Outcrop of Dipping Strata

EVS is not limited to sedimentary layers or lenses. The figure below shows a cross-section through an outcrop of dipping geologic strata. EVS easily model the layers truncating on the top ground surface.



The file below represents the geology file associated with the above figure. Line 2 of the file is "Elevation", therefore all surface elevations are entered as elevations (not depths) and the relationship between x, y, and elevation must be a right handed coordinate system. The *pinch* flag is used extensively to identify that a geologic layer is not present (pinched out) for a particular boring. It is equivalent to using the value one column to the left. The file was created with the assumption that there was no desire to model any layers below -70 foot elevation and that all borings extend to/beyond that depth.

Also, we have assigned the following material layer colors (numbers) to the 7 layers.

Layer # Material Abbreviation Material Color

- 1 Shale SH 5
- 2 Silty-sand SS 2
- 3 Sand SD 1
- 4 Sandy-silt SLS 3
- 5 Silty-sand SS 2
- 6 Sandy-silt SLS 3
- 7 Silt SL 4

X	Y	TOP	BOT_1	BOT_2	BOT_3	BOT_4	BOT_5	BOT_6	BOT_7	NAME
Elevation	Top	SH	SH	SS	SD	SLS	SS	SLS	SL	feet
44	8	5	5	2	1	3	2	3	4	
5	3	23.5	4	-22	pinch	-39	-70	-70	-70	A
13	5	26	13	-18	pinch	-36	-64	-70	-70	B

24	7	26	22	-9	-9.5	-32	-57.5	-70	-70	C
42	2	22	pinch	pinch	-3	-24	-50	-70	-70	D
57	6	24	pinch	pinch	4	-15	-43.5	-70	-70	E
72	7	30.5	pinch	pinch	14	-4	-37	-70	-70	F
85	3	33	pinch	pinch	21.5	6	-30	-70	-70	G
107	4	29.5	pinch	pinch	pinch	19	-20	-60	-70	H
123	6	29.5	pinch	pinch	pinch	28.5	-10	-49.5	-70	I
136	3	38	pinch	pinch	pinch	pinch	-4	-44	-70	J
145	0	39.5	pinch	pinch	pinch	pinch	-3	-39	-70	K
3.11	28.18	25.93	3.96	-20.99	pinch	-39.01	-70	-70	-70	A1
16.85	37.97	24.85	15.61	-20.7	pinch	-35.7	-61.92	-70	-70	B1
25.99	32.02	23.05	23.34	-6.11	-6.41	-31.53	-59.17	-70	-70	C1
41.05	25.13	24.26	pinch	pinch	-1.22	-25.57	-47.06	-70	-70	D1
54.43	34.94	26.56	pinch	pinch	1.36	-14.66	-45.49	-70	-70	E1
67.29	29.3	28.3	pinch	pinch	16.45	-6.49	-37.22	-70	-70	F1
88.89	25.31	32.92	pinch	pinch	19.17	6.16	-27.28	-70	-70	G1
104.17	30.58	30.13	pinch	pinch	pinch	19.76	-22.25	-62.18	-70	H1
121.87	30.26	30.76	pinch	pinch	pinch	27.84	-7.81	-49.67	-70	I1
136.99	29.61	35.95	pinch	pinch	pinch	pinch	-6.02	-44.8	-70	J1
149.67	29.33	37.59	pinch	pinch	pinch	pinch	-4.09	-40.17	-70	K1
4.06	62.03	23.47	3.46	-22.43	pinch	-38.05	-70	-70	-70	A2
12.09	64.15	25.26	13.42	-19.11	pinch	-33.89	-59.06	-70	-70	B2
30.73	66.42	25.81	26.1	-3.46	-3.76	-28.81	-58.62	-70	-70	C2
40.43	49.79	26.12	pinch	pinch	-0.5	-27.73	-46.67	-70	-70	D2
54.5	65.51	27.88	pinch	pinch	1.79	-15.51	-43.8	-70	-70	E2
66.41	52.9	25.48	pinch	pinch	16.96	-7.18	-35.22	-70	-70	F2
93.58	50.18	34.29	pinch	pinch	21.62	6.46	-28.76	-70	-70	G2
106.13	55.44	30.39	pinch	pinch	pinch	20.9	-23.47	-60.65	-70	H2
126.19	63.43	28.78	pinch	pinch	pinch	27.64	-8.31	-48.85	-70	I2
138.39	62.4	36.52	pinch	pinch	pinch	pinch	-5.72	-47.12	-70	J2
144.91	52.79	40.49	pinch	pinch	pinch	pinch	-4.66	-37.23	-70	K2
6.77	86.15	21.09	2.83	-22.62	pinch	-36.05	-70	-70	-70	A3
16.91	98.53	22.86	10.95	-17.19	pinch	-31.33	-57.46	-70	-70	B3
35.07	87.05	25.39	25.81	-2.37	-2.67	-30.89	-59.85	-70	-70	C3
36.37	77.38	26.62	pinch	pinch	-2.19	-27.56	-43.87	-70	-70	D3
51.5	94.86	27.26	pinch	pinch	4.57	-15.51	-46.35	-70	-70	E3
71.23	73.19	26.45	pinch	pinch	16.19	-9.22	-38.04	-70	-70	F3
93.09	79.15	33.93	pinch	pinch	19.64	9.37	-28.16	-70	-70	G3
110.18	76.02	27.4	pinch	pinch	pinch	20.63	-21.81	-63.39	-70	H3
127.9	90.62	31.64	pinch	pinch	pinch	29.56	-8.26	-45.96	-70	I3
139.27	96.26	37.57	pinch	pinch	pinch	pinch	-8.29	-47.67	-70	J3

143.52	75.62	38.22	pinch	pinch	pinch	pinch	-6.59	-37.51	-70	K3
--------	-------	-------	-------	-------	-------	-------	-------	--------	-----	----

Geology Files for Production of a Fence Diagram

Discussion of Geology Files for Fence Sections

Files used to create fence diagrams contain only those borings that the user wishes to include on an individual cross section of the fence, in the order that they will be connected along the section. The resulting set of files includes one .geo file for each cross section that will be included in a fence diagram. The order of the boring listings determines the connectivity of the fence diagram, and must match the order of the borings in the associated chemistry file when chemistry is to be displayed on the diagram. The data for the boring(s) at which individual sections will be joined to produce the fence diagram are included in each of the cross section files that will intersect. Generally, it is easiest to create the geology file for the complete 3-D dataset, and then cut and paste the individual section files from the complete file. Examples of a 3-D geology file and a typical set of fence diagram files are presented below.

The format of the data in the file is exactly the same as for 3-D geology files.

Material colors are not supported for fence diagrams.

An example set of files for producing a fence diagram with two merged cross sections are shown below:

Geology File for Cross Section A-A'

Elevation feet

7	8									
11086.52	12830.67	2500	2496	2484	2479	2420	2417.5	2415	2395	BOR-49
11199.04	12810.16	2501	2492	2482	2473	2420	2414.5	2409	2397	BOR-51
11259.67	12819.29	2502	2492	2479	2467	2425	2419.5	2414	2399	BOR-46
11298	12808.63	2503	2492	2492	2480	2424	2413.5	2403	2392	BOR-52
11414.4	12781.1	2504	2491	2482	2471	2420	2416.3	2412	2396	BOR-34
11427	12780.9	2501	2493	2477	2467	2424	2415.0	2406	2397	BOR-42
11496.34	12753.59	2502	2492	2480	2465	2422	2416.5	2411	2400	BOR-53

Geology File for Cross Section B B'

Elevation feet

5	8									
11209.35	12993.94	2502	2492	2481	2462	2423	2415	2410	2400	BOR 57
11251.30	12929.27	2503	2493	2474	2465	2422	2414	2406	2397	BOR 75
11248.75	12870.91	2501	2492	2483	2472	2421	2416	2411	2396	BOR 48
11199.04	12810.16	2501	2492	2482	2473	2420	2414	2409	2397	BOR 51
11211.87	12710.75	2503	2493	2480	2468	2422	2420	2415	2399	BOR 50

This example fence diagram contains two cross sections, with elevations for the surface and the bottoms of seven layers of geology in each. Section A-A' has seven borings that will be used to define it, and Section B-B' has five borings. Neither of the sections contains layers that pinch out, and all of the borings extend to the depth of the fence. Note that the entries for location BOR-51 are identical in each file, and

are placed such that the sections will cross at the second location in the A-A' file, and the fourth location in the B-B' file. The user will typically use a basemap to plan the orientations and intersections of the fences. EVS does not impose any restrictions on the number of borings in or placement of sections in fence diagrams, but planning should be done to assure that most sections of the fence can be viewed from a chosen viewpoint.

Geology Multi-File

Geology Multi-Files: Unlike the .geo file format, the .gmf format is not based on boring observations with common x,y coordinates. The multi-file format allows for description of individual geologic surfaces by defining a set of x,y,z coordinates (separated by spaces, tabs, and/or commas). Geologic hierarchy still applies for definition of complex geologic structures.

This file format allows for creation of geologic models when the data available for the top surface and one or more of the subsurface layers are uncorrelated (in number or x,y location). For example, a gmf file may contain 1000 x,y,z measurements for the ground surface, but only 12 x,y,z measurements for other lithologic surfaces. This format also allows for specification of the geologic material color (layer material number).

You **SHOULD** include the units of your coordinates (e.g. *feet* or *meters*). If this is included it must be on a line following the word *units*.

Note: there are no special flags (e.g. short, pinch, etc.) used in GMF files. Since each surface stands on its own (does not refer to a prior surface) pinched-out layers are accomplished by duplicating the elevations (x,y,z points) on two consecutive surfaces. The "short" flags are not needed since those points are merely excluded from a surface's definition.

The name for a surface can be a date or date & time if the data represents surface points at different times (e.g. changing groundwater elevations. The date format is dependent on your REGIONAL SETTINGS on your computer (control panel).

C Tech uses the SHORT DATE and SHORT TIME formats.

If the date/time works in Excel it will likely work in EVS.

For most people in the U.S., this would not be 24 hour clock so you would need:

"m/d/yyyy hh:mm:ss AM" or "m/d/yyyy hh:mm:ss PM"

Also, you **MUST** put the date/time in quotes if you use more than just date (i.e. if there are spaces in the total date/time).

Format: The following is a geology multi-file which is included with EVS. This file begins with the line starting with a "#".

```
# Lines beginning with a "#" character are comments.
# Each geologic surface begins with a line: surface x
# The number after surface is the layer material color number.
# Each surface can have different x,y coords and number of points
units ft
surface 2 Top
11086.5 12830.7 4.5
11199.0 12810.2 4
# Comment lines can be placed anywhere in a multi-file
11259.7 12819.3 2
```

11298.0 12808.6 3
11414.4 12781.1 2
11427.0 12780.9 6.5
11496.3 12753.6 1.5
11209.4 12993.9 2
11251.3 12929.3 2
11248.8 12870.9 3
11211.9 12710.8 2
11302.0 13079.7 4.5
11286.8 13026.7 2
11309.0 12949.0 4
11340.5 12892.6 2.5
11338.0 12830.8 4
11393.5 12948.9 3.5
11401.7 12897.8 4
11416.9 12819.5 2.5
11381.7 12747.5 1.5
11410.3 12724.7 0.5
11566.3 12850.6 2.5
11586.3 13050.6 11.5
11086.3 13090.6 8.5
surface 2 Fill
11086.5 12830.7 -3.8
11199.0 12810.2 -5
11259.7 12819.3 -7.5
11298.0 12808.6 -6
11414.4 12781.1 -6
11427.0 12780.9 -7
11496.3 12753.6 -7.5
11209.4 12993.9 -3
11251.3 12929.3 -2.5
11248.8 12870.9 -3.5
11211.9 12710.8 -6.5
11302.0 13079.7 -3.5
11286.8 13026.7 -5
11309.0 12949.0 -2.5
11340.5 12892.6 -2.5
11338.0 12830.8 -8.8
11393.5 12948.9 -3.8
11401.7 12897.8 -2
11416.9 12819.5 -5
11381.7 12747.5 -4
11410.3 12724.7 -4.5
11566.3 12850.6 -5
11586.3 13050.6 1

11086.3 13090.6 -1
surface 1 Silt
11086.5 12830.7 -21
11199.0 12810.2 -20
11259.7 12819.3 -20.5
11298.0 12808.6 -19
11414.4 12781.1 -20.5
11427.0 12780.9 -23
11496.3 12753.6 -20
11209.4 12993.9 -23
11251.3 12929.3 -22
11248.8 12870.9 -22
11211.9 12710.8 -22.5
11302.0 13079.7 -21.9
11286.8 13026.7 -23
11309.0 12949.0 -22
11340.5 12892.6 -20
11338.0 12830.8 -23
11393.5 12948.9 -23
11401.7 12897.8 -22
11416.9 12819.5 -21
11381.7 12747.5 -21.5
11410.3 12724.7 -22.9
11566.3 12850.6 -21
11586.3 13050.6 -11
11086.3 13090.6 -14
surface 3 Clay
11086.5 12830.7 -26
11199.0 12810.2 -25
11259.7 12819.3 -27
11298.0 12808.6 -25.8
11414.4 12781.1 -28
11427.0 12780.9 -28.5
11496.3 12753.6 -28.8
11209.4 12993.9 -27.5
11251.3 12929.3 -28
11248.8 12870.9 -28.5
11211.9 12710.8 -27.5
11302.0 13079.7 -26
11286.8 13026.7 -29
11309.0 12949.0 -28.3
11340.5 12892.6 -23
11338.0 12830.8 -26.5
11393.5 12948.9 -27
11401.7 12897.8 -27.5

11416.9 12819.5 -28.5
11381.7 12747.5 -25.8
11410.3 12724.7 -25
11566.3 12850.6 -28.5
11586.3 13050.6 -18.5
11086.3 13090.6 -23.5

surface 5 Gravel

11086.5 12830.7 -42
11199.0 12810.2 -39
11259.7 12819.3 -40
11298.0 12808.6 -41.8
11414.4 12781.1 -42
11427.0 12780.9 -38.5
11496.3 12753.6 -38.8
11209.4 12993.9 -37.5
11251.3 12929.3 -40
11248.8 12870.9 -36.3
11211.9 12710.8 -37.5
11302.0 13079.7 -38
11286.8 13026.7 -37
11309.0 12949.0 -38.3
11340.5 12892.6 -38
11338.0 12830.8 -36.5
11393.5 12948.9 -39
11401.7 12897.8 -37.5
11416.9 12819.5 -38.5
11381.7 12747.5 -42.8
11410.3 12724.7 -36
11566.3 12850.6 -38.5
11586.3 13050.6 -26.5
11086.3 13090.6 -32.5

surface 4 Sand

11086.5 12830.7 -55
11199.0 12810.2 -53
11259.7 12819.3 -53
11298.0 12808.6 -55
11414.4 12781.1 -55
11427.0 12780.9 -51
11496.3 12753.6 -51
11209.4 12993.9 -51
11251.3 12929.3 -53
11248.8 12870.9 -50
11211.9 12710.8 -51
11302.0 13079.7 -51
11286.8 13026.7 -50

```

11309.0 12949.0 -52
11340.5 12892.6 -52
11338.0 12830.8 -50
11393.5 12948.9 -52
11401.7 12897.8 -51
11416.9 12819.5 -51
11381.7 12747.5 -56
11410.3 12724.7 -49
11566.3 12850.6 -51
11586.3 13050.6 -47
11086.3 13090.6 -48
end

```

Geology Multi-File

Geology Multi-Files: Unlike the .geo file format, the .gmf format is not based on boring observations with common x,y coordinates. The multi-file format allows for description of individual geologic surfaces by defining a set of x,y,z coordinates (separated by spaces, tabs, and/or commas). Geologic hierarchy still applies for definition of complex geologic structures.

This file format allows for creation of geologic models when the data available for the top surface and one or more of the subsurface layers are uncorrelated (in number or x,y location). For example, a gmf file may contain 1000 x,y,z measurements for the ground surface, but only 12 x,y,z measurements for other lithologic surfaces. This format also allows for specification of the geologic material color (layer material number).

You **SHOULD** include the units of your coordinates (e.g. *feet* or *meters*). If this is included it must be on a line following the word *units*.

Note: there are no special flags (e.g. short, pinch, etc.) used in GMF files. Since each surface stands on its own (does not refer to a prior surface) pinched-out layers are accomplished by duplicating the elevations (x,y,z points) on two consecutive surfaces. The "short" flags are not needed since those points are merely excluded from a surface's definition.

The name for a surface can be a date or date & time if the data represents surface points at different times (e.g. changing groundwater elevations. The date format is dependent on your REGIONAL SETTINGS on your computer (control panel).

C Tech uses the SHORT DATE and SHORT TIME formats.

If the date/time works in Excel it will likely work in EVS.

For most people in the U.S., this would not be 24 hour clock so you would need:

"m/d/yyyy hh:mm:ss AM" or "m/d/yyyy hh:mm:ss PM"

Also, you **MUST** put the date/time in quotes if you use more than just date (i.e. if there are spaces in the total date/time).

Format: The following is a geology multi-file which is included with EVS. This file begins with the line starting with a "#".

```

# Lines beginning with a "#" character are comments.
# Each geologic surface begins with a line: surface x
# The number after surface is the layer material color number.
# Each surface can have different x,y coords and number of points

```

```
units ft
surface 2 Top
11086.5 12830.7 4.5
11199.0 12810.2 4
# Comment lines can be placed anywhere in a multi-file
11259.7 12819.3 2
11298.0 12808.6 3
11414.4 12781.1 2
11427.0 12780.9 6.5
11496.3 12753.6 1.5
11209.4 12993.9 2
11251.3 12929.3 2
11248.8 12870.9 3
11211.9 12710.8 2
11302.0 13079.7 4.5
11286.8 13026.7 2
11309.0 12949.0 4
11340.5 12892.6 2.5
11338.0 12830.8 4
11393.5 12948.9 3.5
11401.7 12897.8 4
11416.9 12819.5 2.5
11381.7 12747.5 1.5
11410.3 12724.7 0.5
11566.3 12850.6 2.5
11586.3 13050.6 11.5
11086.3 13090.6 8.5
surface 2 Fill
11086.5 12830.7 -3.8
11199.0 12810.2 -5
11259.7 12819.3 -7.5
11298.0 12808.6 -6
11414.4 12781.1 -6
11427.0 12780.9 -7
11496.3 12753.6 -7.5
11209.4 12993.9 -3
11251.3 12929.3 -2.5
11248.8 12870.9 -3.5
11211.9 12710.8 -6.5
11302.0 13079.7 -3.5
11286.8 13026.7 -5
11309.0 12949.0 -2.5
11340.5 12892.6 -2.5
11338.0 12830.8 -8.8
11393.5 12948.9 -3.8
```

11401.7 12897.8 -2
11416.9 12819.5 -5
11381.7 12747.5 -4
11410.3 12724.7 -4.5
11566.3 12850.6 -5
11586.3 13050.6 1
11086.3 13090.6 -1
surface 1 Silt
11086.5 12830.7 -21
11199.0 12810.2 -20
11259.7 12819.3 -20.5
11298.0 12808.6 -19
11414.4 12781.1 -20.5
11427.0 12780.9 -23
11496.3 12753.6 -20
11209.4 12993.9 -23
11251.3 12929.3 -22
11248.8 12870.9 -22
11211.9 12710.8 -22.5
11302.0 13079.7 -21.9
11286.8 13026.7 -23
11309.0 12949.0 -22
11340.5 12892.6 -20
11338.0 12830.8 -23
11393.5 12948.9 -23
11401.7 12897.8 -22
11416.9 12819.5 -21
11381.7 12747.5 -21.5
11410.3 12724.7 -22.9
11566.3 12850.6 -21
11586.3 13050.6 -11
11086.3 13090.6 -14
surface 3 Clay
11086.5 12830.7 -26
11199.0 12810.2 -25
11259.7 12819.3 -27
11298.0 12808.6 -25.8
11414.4 12781.1 -28
11427.0 12780.9 -28.5
11496.3 12753.6 -28.8
11209.4 12993.9 -27.5
11251.3 12929.3 -28
11248.8 12870.9 -28.5
11211.9 12710.8 -27.5
11302.0 13079.7 -26

11286.8 13026.7 -29
11309.0 12949.0 -28.3
11340.5 12892.6 -23
11338.0 12830.8 -26.5
11393.5 12948.9 -27
11401.7 12897.8 -27.5
11416.9 12819.5 -28.5
11381.7 12747.5 -25.8
11410.3 12724.7 -25
11566.3 12850.6 -28.5
11586.3 13050.6 -18.5
11086.3 13090.6 -23.5
surface 5 Gravel
11086.5 12830.7 -42
11199.0 12810.2 -39
11259.7 12819.3 -40
11298.0 12808.6 -41.8
11414.4 12781.1 -42
11427.0 12780.9 -38.5
11496.3 12753.6 -38.8
11209.4 12993.9 -37.5
11251.3 12929.3 -40
11248.8 12870.9 -36.3
11211.9 12710.8 -37.5
11302.0 13079.7 -38
11286.8 13026.7 -37
11309.0 12949.0 -38.3
11340.5 12892.6 -38
11338.0 12830.8 -36.5
11393.5 12948.9 -39
11401.7 12897.8 -37.5
11416.9 12819.5 -38.5
11381.7 12747.5 -42.8
11410.3 12724.7 -36
11566.3 12850.6 -38.5
11586.3 13050.6 -26.5
11086.3 13090.6 -32.5
surface 4 Sand
11086.5 12830.7 -55
11199.0 12810.2 -53
11259.7 12819.3 -53
11298.0 12808.6 -55
11414.4 12781.1 -55
11427.0 12780.9 -51
11496.3 12753.6 -51

```
11209.4 12993.9 -51
11251.3 12929.3 -53
11248.8 12870.9 -50
11211.9 12710.8 -51
11302.0 13079.7 -51
11286.8 13026.7 -50
11309.0 12949.0 -52
11340.5 12892.6 -52
11338.0 12830.8 -50
11393.5 12948.9 -52
11401.7 12897.8 -51
11416.9 12819.5 -51
11381.7 12747.5 -56
11410.3 12724.7 -49
11566.3 12850.6 -51
11586.3 13050.6 -47
11086.3 13090.6 -48
end
```

ctech_example.gmf

```
# Database Generated GMF File (Creation at 7/22/2003 5:36:07 PM)
```

```
#
```

```
#
```

```
# Surface 1: 25 Coordinates
```

```
# Database Columns [GMF_Surface0 (Ground Surface)]: X, Y, Top
```

```
surface 1 Sand
```

```
11566.34 12850.59 2.5
```

```
11586.34 13050.59 11.5
```

```
11086.3 13090.6 8.5
```

```
.
```

```
.
```

```
.
```

```
.
```

```
11393.47 12948.9 3.5
```

```
11251.3 12929.27 2
```

```
# Surface 1 Complete
```

```
#
```

```
# Surface 2: 24 Coordinates (Added at 7/22/2003 5:37:04 PM)
```

```
# Database Columns [GMF_Surface1]: X, Y, Z
```

```
surface 1 Sand
```

```
11566.34 12850.59 -5
```

```
11586.34 13050.59 1
```

```
11086.3 13090.6 -1
```

```
.
```

```
.
```

```
.
```

```

.
11393.47 12948.9 -3.8
11251.3 12929.27 -2.5
# Surface 2 Complete
#
# Surface 3: 24 Coordinates (Added at 7/22/2003 5:38:18 PM)
# Database Columns [GMF_Surface2]: X, Y, Z
surface 1 Sand
11566.34 12850.59 -21
11586.34 13050.59 -11
11086.3 13090.6 -14
.
.
.
11393.47 12948.9 -23
11251.3 12929.27 -22
# Surface 3 Complete
#
units ft
end
# Database Generated GMF File (Finalization at 7/22/2003 5:39:06 PM)

```

.PT File Format

The .PT (Place-Text) format is used to place 3D text (labels) with user adjustable font and alignment.

The format is:

- Lines beginning with "#" are comments
- Lines beginning with "FONT" are font specification lines (more later)
- Lines beginning with "END" specify the end of the file (this is optional, but if you want to have anything after the last command or data line, precede it with an "END" statement.
- All other lines are DATA lines specifying the x-y-z coordinates of a string and the text for that string.
- Blank lines are ignored.
- The FONT specification lines contain the following information **in this order**:
 - Height: The font height of a typical capitol letter is in true user units
 - Justification: The justification (Horizontal & Vertical Alignment) options are the same as in [post samples](#)
 - Azimuth: refers to the same Azimuth specified in the viewer's View tab or *Azimuth and Inclination* section under viewer Properties.
 - Inclination: refers to the same Inclination specified in the viewer's View tab or *Azimuth and Inclination* section under viewer Properties.
 - Roll: refers to the same Roll specified in the viewer's *Advanced* portion of the *Azimuth and Inclination* section under viewer Properties.

- Red, Green, Blue: These 3 numbers determine the font color.
- Resolution: The resolution parameter is the same as in [post_samples](#)
- Depth: The parameter is the same as in [post_samples](#)
- Bevel%: The Bevel percentage is the same as in [post_samples](#)
- Font Style Flags: None, Bold, Italic in quotes e.g. "Bold, Italics"
- Font Name: The Font Name must be in quotes.
- The DATA lines contain five columns of information:
 1. X coordinate
 2. Y coordinate
 3. Z coordinate
 4. Explode_ID: This is equivalent to the (Stratigraphic) cell data "Layer" information. The uppermost ID (layer) is ZERO (0) and does not move. If you don't want your text to move with changing *Explode Distance*, use a value of ZERO. Otherwise, by assigning an appropriate ID value your text string can move properly with both stratigraphic layers or lithologic materials as they are exploded.
 5. Text: Everything on the line after the z coordinate (and trailing spaces) is the text to be placed at the above coordinate, and must be in quotes.

Overview of Module Libraries

EVS modules can each be considered software applications that can be combined together by the user to form high level customized applications performing analysis and visualization. These modules have input and output ports and user interfaces.

The library of module are grouped into the following categories:

- Estimation modules take sparse data and map it to surface and volumetric grids
- Geology modules provide methods to create surfaces or 3D volumetric grids with lithology and stratigraphy assigned to groups of cells
- Display modules are focused on visualization functions
- Analysis modules provide quantification and statistical information
- Annotation modules allow you to add axes, titles and other references to your visualizations
- Subsetting modules extract a subset of your grids or data in order to perform boolean operations
- Proximity modules create new data which can be used to subset or assess proximity to surfaces, areas or lines.
- Processing modules act on your data
- Import modules read files that contain grids, data and/or archives
- Export modules write files that grids, data and/or archives
- Modeling modules are focused on functionality related to simulations and vector data
- Geometry modules create or act upon grids and geometric primitives
- Projection modules transform grids into other coordinates or dimensionality
- Image modules are focused on aerial photos or bitmap operations
- Time modules provide the ability to deal with time domain data
- Tools are a collection of modules to make life easier
- Cell Data modules have functionality specific to cell data (vs. nodal data)
- View modules are focused on visualization and output of results

Overview of Module Libraries

EVS modules can each be considered software applications that can be combined together by the user to form high level customized applications performing analysis and visualization. These modules have input and output ports and user interfaces.

The library of module are grouped into the following categories:

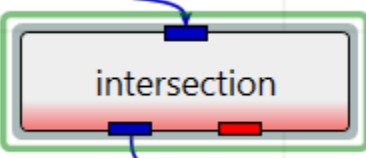
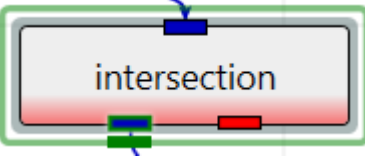
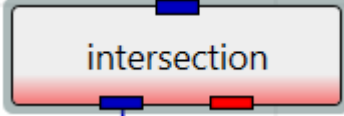
- Estimation modules take sparse data and map it to surface and volumetric grids
- Geology modules provide methods to create surfaces or 3D volumetric grids with lithology and stratigraphy assigned to groups of cells
- Display modules are focused on visualization functions
- Analysis modules provide quantification and statistical information

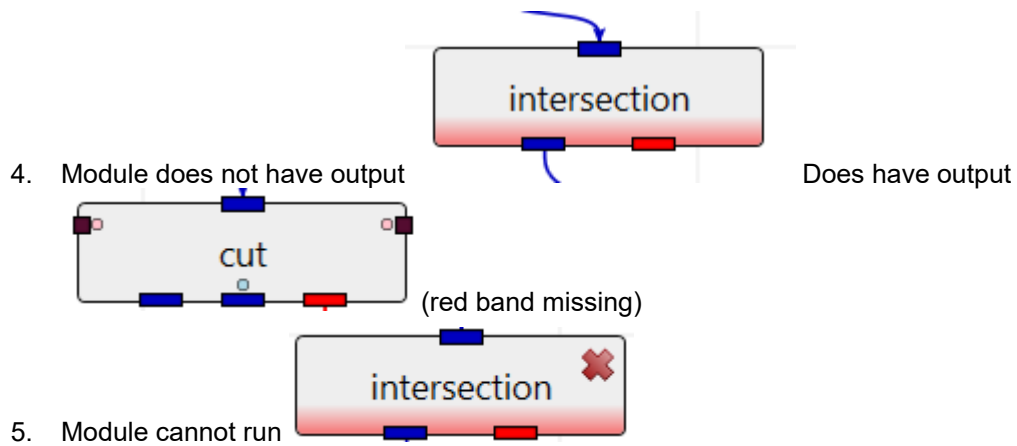
- Annotation modules allow you to add axes, titles and other references to your visualizations
- Subsetting modules extract a subset of your grids or data in order to perform boolean operations
- Proximity modules create new data which can be used to subset or assess proximity to surfaces, areas or lines.
- Processing modules act on your data
- Import modules read files that contain grids, data and/or archives
- Export modules write files that grids, data and/or archives
- Modeling modules are focused on functionality related to simulations and vector data
- Geometry modules create or act upon grids and geometric primitives
- Projection modules transform grids into other coordinates or dimensionality
- Image modules are focused on aerial photos or bitmap operations
- Time modules provide the ability to deal with time domain data
- Tools are a collection of modules to make life easier
- Cell Data modules have functionality specific to cell data (vs. nodal data)
- View modules are focused on visualization and output of results

Module Status Indicators

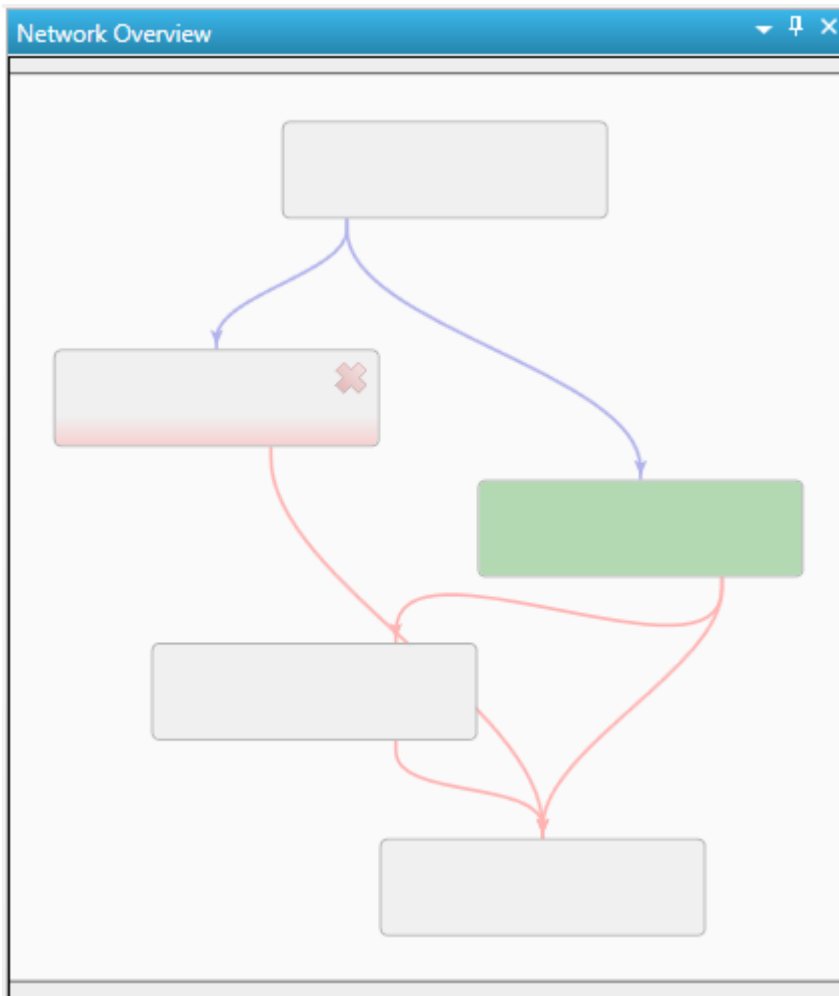
EVS modules have four visual indicators which provide status of two important criteria:

1. Whether it is the current module whose parameters can be changed in the Properties window.
2. Whether a particular output port of the module is being edited in the Properties window.
3. Whether the module is (one of) the currently selected module
4. Whether the module has output
5. Whether the module can run (execute) in an automatic manner (such as when new data comes to their input ports or a property is changed)











1. Current editable module: (grey border)  Note: Also is current selected
2. Current editable module's port selected (grey border)  Note: Also is current
3. Current selected module 



Also, in the Network Overview window, much of the above status information is still available. In the image below, four modules have run, one cannot run and the (green) selected for editing module is obvious.



Module Input and Output Ports

- **Renderable object**  This commonly used port connects various modules generally to the viewer. It contains the grids, data and rendering information.
- **Field**  This is the most common port which passes your grids and data (nodal or cell) between modules that create these fields and those that subset or modify them.
- **String**  Used to pass strings which can range from single words to phrases and file names and paths.
- **Geologic Legend Information**  This data port contains material names from geology modules
- **Vistas Data** Used to pass geologic surface information to Groundwater Vistas to initialize MODFLOW models.
- Brown: **Number**  This port passes a real number. format_string has number input ports
 - Brown side port: **Z-Scale**  This port passes the z-exaggeration factor. Used by many modules such as explode_and_scale.
 - Brown side port: **Explode Factor**  This port passes the explode factor. Used by many modules such as explode_and_scale.
- Light Blue port: **Date-Time**  This port passes date & time
- **Image** Port:  passes images
- Purple side port: **View**:  This port is a viewer output port and input to many other modules. It is used to pass information to render images, allow for

When trying to determine which port on the module icon corresponds to which documented item, remember:

- Input Ports are documented left side top to bottom and the top side from left to right.
- Output ports are documented rights side top to bottom and the bottom side from left to right.

With the cursor on the module, hover over a port for details or click the right mouse button to activate the menu.

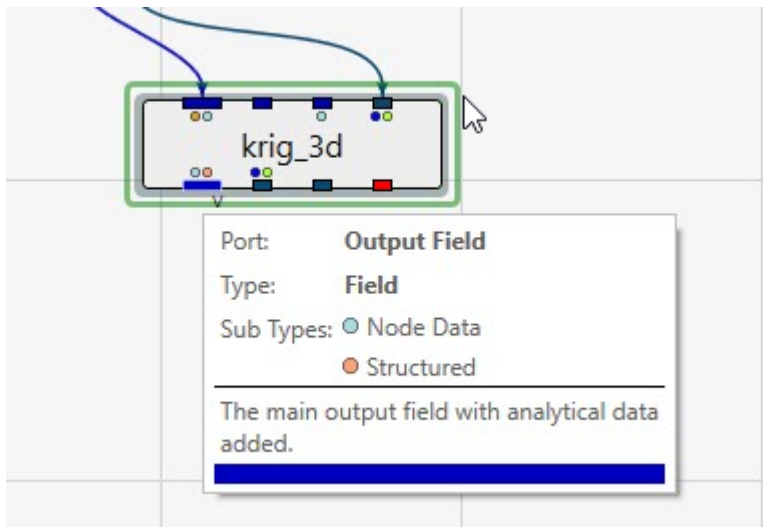
PORT SUBTYPES:

In addition to the primary colors of ports, some ports have subtypes designated by one or more small colored circular dots. These dots designate port subtypes which will further restrict how ports connect based on the type of data that the ports contain. The port subtypes depend on the types of ports and are in three primary classes:

1. Fields
 - Geology: Goldenrod

- Structured: LightSalmon
 - Uniform: Purple
 - Node Data: LightBlue
 - Cell Data: Green
2. Number
 - Z Scale: Pink
 - Explode: Gold
 3. String
 - Filename: Blue
 - Analytical File: GreenYellow
 - Stratigraphy File: LightGoldenrodYellow
 - Lithology File: DarkGray

Below is an example of krig_3d's output field where its output has two subtypes: Node Data and Structured. Please note that krig_3d's output can have different subtypes depending on its inputs and settings.



krig_3d_geology

The krig_3d_geology module uses data files containing geologic horizons or surfaces (usually .geo, .gmf and other ctech formats containing surfaces) to model the surfaces bounding geologic layers that will provide the framework for three-dimensional geologic modeling and parameter estimation. Conversion of scattered points to surfaces uses kriging (default) or spline (previously in the spline_geology module), IDW or nearest neighbor algorithms.

krig_3d_geology creates a 2D grid containing one or more elevations at each node. Each elevation represents a geologic surface at that point in space. The output of krig_3d_geology is a data field that can be sent to several modules (e.g. krig_3d, 3d_geology_map, geology_to_structured, geologic_surfaces, etc.)

Those modules which create volumetric models convert the quadrilateral elements into layers of hexahedral (8-node *brick*) elements. The output of krig_3d_geology can also be sent to the geologic_surface(s) module(s) which allow visualization of the individual layers of quadrilateral elements (the surfaces) that comprise the surfaces.

krig_3d_geology has the capability to produce layer surfaces within the convex hull of the data domain, within a rectilinear domain with equally spaced nodes, or within a rectilinear domain with specified cell sizes such as a finite-difference model grid. The finite-difference gridding capabilities allows the user to visually design a grid with variable spacing, and then krig the geologic layer elevations directly to the finite difference grid nodes. krig_3d_geology also provides geologic surface definitions to the post_samples module to allow exploding of boreholes and samples by geologic layer.

Note: krig_3d_geology has the ability to read .apdv, .aidv and .pgf file to create a single geologic layer model. This was not done as a preferred alternative to creating/representing your valid site geology. However, most sites have some ground surface topography variation. If krig_3d is used without geology input, the resulting output will have flat top and bottom surfaces. The flat top surface may be below or above the actual ground surface at various locations. This can result in plume volumes that are inaccurate.

When a .apdv or .pgf is read by krig_3d_geology the files are interpreted as geology as follows:

- 1) If **Top** of boring elevations are provided in the file, these values are used to create the ground surface.
- 2) If **Top** of boring elevations **are not** provide in the file, the elevations of the highest sample in each boring are used to create the ground surface.
- 3) The bottom surface is created as a flat surface slightly below the lowest sample in the file. The elevation of the surface is computed by taking the lowest sample and subtracting 5% of the total z-extent of the samples.

When reading these files, you will get a single layer which goes to either the Top column (if it exists) otherwise, the top sample in each boring, and 5% below the lowest sample in the file (flat bottom). This allows you to create a convex hull around data without having geology info. It also provide a topographic top surfaces if your analyte (e.g. chemistry) or PGF file has Tops (grounds surface elevations). Also nice for doing indicator kriging (since a single, well-defined pgf can give you an entire indicator model now). Be aware that if Top is specified, but all values are exactly 0.0, the top sample elevation for each boring will be used.

Module Input [Ports](#)

- **Filename** [String / minor] Receives the filename from other modules.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Geologic Field** [Field] Can be connected to the krig_3d, 3D_Geology Map, and geologic_surface(s) modules.
- **Filename** [String / minor] Outputs a string containing the file name and path. This can be connected to other modules to share files.
- **Status Information** [String / minor] Outputs a string containing module parameters. This is useful for connection to save_evs_field to document the settings used to create a grid.
- **Geology Export Output** [Vistas Data / minor] Provides input to the Geology_to_Vistas and other modules which create raster output.

- **Grid** [Renderable / minor] Outputs the geometry of 2D grid.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Grid Settings: control the grid type, position and resolution
- Krig Settings: control the estimation methods
- Computational Settings: define computational surfaces included in the output. This allows a single surface file to define a layer specified by elevation or depth.

krig_3d

krig_3d performs parameter estimation using kriging and other methods to map 3D analytical data onto volumetric grids defined by the limits of the data set, or by the convex hull, rectilinear, or finite-difference grid extents of a geologic system modeled by krig_3d_geology. krig_3d provides several convenient options for pre- and post-processing the input parameter values, and allows the user to consider anisotropy in the medium containing the property.

krig_3d also has the ability to create uniform fields, and the ability to choose which data components you want to include in the output. There are a couple significant requirements for uniform fields. First, there cannot be geologic input (otherwise the cells could not be rectangular blocks). Second, Adaptive_Gridding must be turned off (otherwise the connectivity is not implicit).

Module Input [Ports](#)

- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krig data into geologic layers.
- **Input External Grid** [Field / minor] Allows the user to import a previously created grid. All data will be kriged to this grid.
- **Input External Data** [Field / minor] Allows the user to import a field contain data. This data will be kriged to the grid instead of using file data.

Module Output [Ports](#)

- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.
- **Status Information** [String / minor] Outputs a string containing module parameters. This is useful for connection to save_evs_field to document the settings used to create a grid.
- **Uncertainty Sphere** [Renderable / minor] Outputs a sphere to the viewer. This sphere represents the location of maximum uncertainty.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Grid Settings: control the grid type, position and resolution
- Data Processing: controls clipping, processing (Log) and clamping of input data and kriged outputs.
- Time Settings: controls how the module deals with time domain data
- Krig Settings: control the estimation methods
- Data To Export: specify which data is included in the output
- Display Settings: applies to maximum uncertainty sphere
- Drill Guide: parameters association with DrillGuide computations for analytically guided site assessment

For additional information on kriging speed, memory requirements and performance please see the [Performance Benchmarks](#) page.

Variogram Options:

There are three variogram options:

1. Spherical: Our default and recommended choice for most applications
2. Exponential: Generally gives similar results to Spherical and may be superior for some datasets
3. Gaussian: Notoriously unstable, but can "smooth" your data with an appropriate nugget.

I specifically want to discuss the pros and cons of Gaussian. Without a nugget term, Gaussian is generally unusable. When using Autofit, our expert system will apply a modest nugget (~1% of sill) to maintain stability. If you're committed to experimenting with Gaussian, it is recommended that you experiment with the nugget term after EVS computes the Range and Sill. Below are some things to look for:

- If you find that Gaussian kriging is overshooting the plume in various directions, your nugget is likely too small.
- However, if the plume looks overly smooth and is too far from honoring your data, your nugget is likely too big.

Advanced Variography Options:

It is far beyond the scope of our Help to attempt an advanced Geostatistics course. The terminology and variogram plotting style that we use is industry standard and we do so because we will not provide detailed technical support nor complete documentation on these features, which would effectively require a geostatistics textbook, in our help.

However, we have offered an online course on how to take advantage of the complex, directional anisotropic variography capabilities in `krig_3d` (which applies equally well to `indicator_geology` and `adaptive_indicator_krig`), and that course is available as a recorded video class. This class is focused on the mechanics of how to employ and refine the variogram anisotropy with respect to your data and the physics of your project such as contaminated sediments in a river bottom. The variogram is displayed as an ellipsoid which can be distorted to represent the

Primary and Secondary anisotropies and rotated to represent the Heading, Dip and Roll. Overall scale and translation are also provided as additional visual aids to compare the variogram to the data, though these do not affect the actual variogram. We are not hiding this capability from you as the *Anisotropic Variography Study* folder of *Earth Volumetric Studio Projects* contains a number of sample applications which demonstrate exactly what is described above. However, we assure you that understanding how to apply this to your own projects will be quite daunting and really does require a number of prerequisites:

- A thorough explanation of these complex applications
- A reasonable background in Python and how to use Python in Studio
- An understanding of all of the variogram parameters and their impact on the estimation process on both theoretical datasets as well as real-world datasets.

This 3 hour course addresses these issues in detail.

krig_2d

krig_2d performs parameter estimation using kriging and other methods to map 2D analytical data onto surface grids defined by the limits of the data set as rectilinear or convex hull extents of the input data.

Its *Adaptive Gridding* further subdivides individual elements to place a "kriged" node at the location of each input data sample. This guarantees that the output will accurately reflect the input at all measured locations (i.e. the maximum in the output will be the maximum of the input).

The DrillGuide functionality produces a new input data file with a synthetic boring at the location of maximum uncertainty calculated from the previous kriging estimates, which can then be rerun to find the next area of highest uncertainty. The naming of the "DrillGuide©" file which is created when krig_2d is run with all types of analyte (e.g. chemistry) files ends in apdv1, apdv2, apdv3, etc. the output file name will be .apdv2, apdv3, apdv4.... There are no limits to the number of cycles that may be run.

The use of krig_2d to perform analytically guided site assessment is covered in detail in Workbook 2: DrillGuide© Analytically Guided Site Assessment.

This process can be continued as many times as desired to define the number and placement of additional borings that are needed to reduce the maximum uncertainty in the modeled domain to a user specified level. The features of krig_2d make it particularly useful for optimizing the benefits obtained from environmental sampling or ore drilling programs. krig_2d also provides some special data processing options that are unique to it, which allow it to extract 2-dimensional data sets from input data files that contain three-dimensional data. This functionality allows it to use the same .apdv files as all of the other EVS input and kriging modules, and allows detailed analyses of property characteristics along 2-dimensional planes through the data set. krig_2d also provides the user with options to magnify or distort the resulting grid by the kriged value of the property at each grid node. krig_2d also allows the user to automatically clamp the data distribution to a specified level along a boundary that can be offset from the convex hull of the data domain by a user defined amount.

Module Input [Ports](#)

- **Input External Grid** [Field / minor] Allows the user to import a previously created grid. All data will be kriged to this grid.

- **Input External Data** [Field / minor] Allows the user to import a field contain data. This data will be kriged to the grid instead of using file data.
- **Filename** [String / minor] Allows the sharing of file names between similar modules.

Module Output [Ports](#)

- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules which have the same color port
- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Status Information** [String / minor] Outputs a string containing module parameters. This is useful for connection to save_evs_field to document the settings used to create a grid.
- **Surface** [Renderable] Outputs the kriged surface to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Grid Settings: control the grid type, position and resolution
- Data Processing: controls clipping, processing (Log) and clamping of input data and kriged outputs.
- Time Settings: controls how the module deals with time domain data
- Krig Settings: control the estimation methods
- Data To Export: specify which data is included in the output
- Display Settings: applies to maximum uncertainty sphere
- Drill Guide: parameters association with DrillGuide computations for analytically guided site assessment

Variogram Options:

There are three variogram options:

1. Spherical: Our default and recommended choice for most applications
2. Exponential: Generally gives similar results to Spherical and may be superior for some datasets
3. Gaussian: Notoriously unstable, but can "smooth" your data with an appropriate nugget.

I specifically want to discuss the pros and cons of Gaussian. Without a nugget term, Gaussian is generally unusable. When using Autofit, our expert system will apply a modest nugget (~1% of sill) to maintain stability. If you're committed to experimenting with Gaussian, it is recommended that you experiment with the nugget term after EVS computes the Range and Sill. Below are some things to look for:

- If you find that Gaussian kriging is overshooting the plume in various directions, your nugget is likely too small.

- However, if the plume looks overly smooth and is too far from honoring your data, your nugget is likely too big.

analytical_realization

The analytical_realization module is one of three similar modules (the other two are indicator_realization and stratigraphic_realization), which allows you to very quickly generate statistical realizations of your 2D and 3D kriged models based upon C Tech's Proprietary Extended *Gaussian Geostatistical Simulation* (GGS) technology, which we refer to as Fast Geostatistical Realizations® or FGR®. Our extensions to GGS allow you to:

- Create realizations very rapidly
- Exercise greater control over the frequency and magnitude of *noise* typical in GGS.
- Control deviation magnitudes from the nominal kriged prediction based on a *Min Max Confidence Equivalent*.
 - Deviations are the absolute value of the changes to the analytical prediction (in user units)
- Apply Simple or Advanced Anisotropy control over 2D or 3D wavelengths

C Tech's FGR® creates more plausible cases (realizations) which allow the Nominal concentrations to deviate from the peak of the bell curve (equal probability of being an under-prediction as an over-prediction) by the same user defined Confidence. However, FGR allows the deviations to be both positive (max) and negative (min), and to fluctuate in a more realistic randomized manner.

Module Input [Ports](#)

- **Realization** [Special Field] Accepts outputs from krig_3d and krig_2d to allow for EGGs models to be created

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting level
- **Deviations Field** [Field] Outputs the deviations from the nominal kriged model

Important Parameters

There are several parameters which affect the realizations. A brief description of each follows:

- Randomness Generator Type
 - There are four types, each of which create a different 2D/3D random distribution
- Anisotropy Mode
 - Two options here are Simple or Advanced. These are equivalent to the variogram settings in krig_3d or krig_2d
- Seed
 - The "Seed" is used in the random number generator, and makes it reproducible.

- Unique seeds create unique realizations
- Wavelength
 - The 2D or 3D random distribution is governed by a mean wavelength that determines the apparent frequency of deviations from the nominal kriged result.
 - Wavelength is in your project coordinates (e.g. meters or feet)
 - Longer wavelengths create smoother realizations
 - Shorter wavelengths create more "noisy" variations in the realizations
 - Very short wavelengths will give results more similar to GGS (aka Sequential Gaussian Simulations)
- Min Max Confidence Equivalent
 - This parameter determines the magnitude of the deviations.
 - Values close to 50% result in outputs that deviate very little from the nominal kriged result.
 - (we do not allow values below 51% for algorithm stability reasons)
 - Values at or approaching 99.99% will result in the greatest (4 sigma) variations (more similar to GGS)

indicator_realization

The `indicator_realization` module is one of three similar modules (the other two are `analytical_realization` and `stratigraphic_realization`), which allows you to very quickly generate statistical realizations of your 2D and 3D lithologic models based upon C Tech's Proprietary Extended *Gaussian Geostatistical Simulation* (GGS), which we refer to as Fast Geostatistical Realizations® or FGR®. Our extensions to GGS allow you to:

- Create realizations rapidly:
 - Though `indicator_realizations` are the slowest of the three because:
 - The material probabilities must be additionally processed to assign materials
 - When Smooth option is on, this process often takes nearly as long as the original kriging
- Exercise greater control over the frequency and magnitude of *visual noise* typical of GGS.
- Control deviation magnitudes from the nominal kriged probability prediction based on a *Min Max Confidence Equivalent*.
 - Deviations are the absolute value of the changes to each material's probability
- Apply Simple or Advanced Anisotropy control over 2D or 3D wavelengths

Module Input [Ports](#)

- **Realization** [Special Field] Accepts outputs from `krig_3d` and `krig_2d` to allow for EGGs models to be created

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting level
- **Deviations Field** [Field] Outputs the deviations from the nominal kriged probabilities

Important Parameters

There are several parameters which affect the realizations. A brief description of each follows:

- Randomness Generator Type
 - There are four types, each of which create a different 2D/3D random distribution
- Anisotropy Mode
 - Two options here are Simple or Advanced. These are equivalent to the variogram settings in indicator_geology
- Seed
 - The "Seed" is used in the random number generator, and makes it reproducible.
 - Unique seeds create unique realizations
- Wavelength
 - The 2D or 3D random distribution is governed by a mean wavelength that determines the apparent frequency of deviations from the nominal kriged probabilities results.
 - Wavelength is in your project coordinates (e.g. meters or feet)
 - Longer wavelengths create smoother realizations
 - Shorter wavelengths create more "noisy" variations in the realizations
 - Very short wavelengths will give results more similar to GGS (aka Sequential Gaussian Simulations)
- Min Max Confidence Equivalent
 - This parameter determines the magnitude of the deviations.
 - Values close to 50% result in outputs that deviate very little from the nominal kriged probabilities results.
 - (we do not allow values below 51% for algorithm stability reasons)
 - Values at or approaching 99.99% will result in the greatest (4 sigma) variations (more similar to GGS)

stratigraphic_realization

The stratigraphic_realization module is one of three similar modules (the other two are analytical_realization and indicator_realization), which allows you to very quickly generate statistical realizations of your stratigraphic horizons based upon C Tech's Proprietary Extended *Gaussian Geostatistical Simulation* (GGS), which we refer to as Fast Geostatistical Realizations® or FGR®. Our extensions to GGS allow you to:

- Create realizations rapidly
- Exercise greater control over the frequency and magnitude of *noise* typical in GGS.

- Control deviation magnitudes from the nominal kriged prediction based on a *Min Max Confidence Equivalent*.
 - Deviations are the absolute value of the changes to surface elevations for each stratigraphic horizon.
- Apply Simple or Advanced Anisotropy control over 2D wavelengths
- **For stratigraphic realizations only: we support Natural Neighbor as well as kriging for the input model.**

Module Input [Ports](#)

- **Realization** [Special Field] Accepts outputs from krig_3d_geology to allow for FGR® models to be created

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting level
- **Deviations Field** [Field] Outputs the deviations from the nominal kriged model

Important Parameters

There are several parameters which affect the realizations. A brief description of each follows:

- Randomness Generator Type
 - There are four types, each of which create a different 2D/3D random distribution
- Anisotropy Mode
 - Two options here are Simple or Advanced. These are equivalent to the variogram settings in krig_3d_geology
- Seed
 - The "Seed" is used in the random number generator, and makes it reproducible.
 - Unique seeds create unique realizations
- Wavelength
 - The 2D or 3D random distribution is governed by a mean wavelength that determines the apparent frequency of deviations from the nominal kriged (or Natural Neighbor) result.
 - Wavelength is in your project coordinates (e.g. meters or feet)
 - Longer wavelengths create smoother realizations
 - Shorter wavelengths create more "noisy" variations in the realizations
 - Very short wavelengths will give results more similar to GGS (aka Sequential Gaussian Simulations)
- Min Max Confidence Equivalent
 - This parameter determines the magnitude of the deviations.
 - Values close to 50% result in outputs that deviate very little from the nominal kriged (or Natural Neighbor) result.

- (we do not allow values below 51% for algorithm stability reasons)
- Values at or approaching 99.99% will result in the greatest (4 sigma) variations (more similar to GGS)

external_kriging

The external_kriging module allows users to perform estimation using grids created in EVS (with or without layers or lithology) in GeoEAS which supports very advanced variography and kriging techniques. Grids and data are kriged externally from EVS and the results can then be read into EVS and treated as if they were kriged in EVS.

This is an advanced module which should be used only by persons with experience with GeoEAS and geostatistics. C Tech does not provide tech support for the use of GeoEAS.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Input Data** [Field] Allows the user to import a field contain data. This data will be kriged to the grid instead of using file data.
- **Input Grid** [Field] Allows the user to import a previously created grid. All data will be kriged to this grid.

Module Output [Ports](#)

- **Output** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: defines Z Scale and grid translation(s)
- Export Data: controls the file names and data processing for creation of GeoEAS inputs.
- Export Grid: Exports the grid and data to GeoEAS formats. A grid and data must be connected to the import ports
- Import Data: Imports the grid and data to GeoEAS formats. A grid and data must be connected to the import ports

make_geo_hierarchy

The make_geo_hierarchy module reads a special input file format called a pgf file, and then allows the user to build geologic surfaces based on the input file's geologic surface intersections. This process is carried out visually (in the EVS viewer) with the use of the make_geo_hierarchy user interface. The surface hierarchy can either be generated automatically for simple geology models or for every layer for complex models. When the user is finished creating surfaces the gmf file can be finalized and converted into a *.GEO file.

3d_geology_map

The 3d_geology_map module creates 3-dimensional solid layers from the 2-dimensional surfaces produced by krig_3d_geology, to allow visualizations of the geologic layering of a system. It accomplishes this by creating a user specified distribution of nodes in the Z dimension between the top and bottom surfaces of each geologic layer.

The number of nodes specified for the Z Resolution may be distributed (proportionately) over the geologic layers in a manner that is approximately proportional to the fractional thickness of each layer relative to the total thickness of the geologic domain. In this case, at least three layers of nodes (2 layers of elements) will be placed in each geologic layer.

Please note that if any portions of the input geology is NULL, these cells will be omitted from the grid that is created. This can save memory and provide a means to cut (in a Lego fashion) along boundaries.

Module Input [Ports](#)

- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krige data into geologic layers.

Module Output [Ports](#)

- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls Z Scale and Explode distance
- Layer Settings: resolution and layer settings
- Data To Export: controls what data to outputs.

geology_to_structured

The geology_to_structured module creates 3-dimensional solid layers from the 2-dimensional surfaces produced by krig_3d_geology, to allow visualizations of the geologic layering of a system. It accomplishes this by creating a user specified distribution of nodes in the Z dimension between the top and bottom surfaces of each geologic layer.

This module is similar to 3d_geology_map, but does not duplicate nodes at the layer boundaries and therefore the model it creates cannot be exploded into individual layers. However, this module has the advantage that its output is substantially more memory efficient and can be used with modules like crop_and_downsize or ortho_slice.

The number of nodes specified for the Z Resolution may be distributed (proportionately) over the geologic layers in a manner that is approximately proportional to the fractional thickness of each layer relative to the total thickness of the geologic domain.

Module Input [Ports](#)

- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krige data into geologic layers.

Module Output [Ports](#)

- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls Z Scale and Explode distance
- Layer Settings: resolution and layer settings
- Data To Export: controls what data to outputs.

layer_from_surface

The layer_from_surface module will create a single geo layer based upon an existing surface and a constant elevation value.

The Surface Defines option will allow the user to set whether the selected surface defines the top or the bottom of the layer. For example if the Top Of Layer is chosen the selected surface will define the top, while the Constant Elevation for Layer will define the bottom of the layer. The 'Material Name / Number' will define the geologic layer name and number for the newly created layer.

geologic_surfaces

The geologic_surfaces module provides complete control of displaying, scaling and exploding one or more geologic surfaces from the set of surfaces output by [krig_3d_geology](#). This module allows visualization of the topology of any or all surfaces and/or the interaction of a set of individual surfaces.

geologic_surfaces can explode geologic surfaces analogous to how explode_and_scale explodes layers created by 3d_geology_map or krig_3d. The ability to explode the surfaces is integral to this module.

geologic_surfaces also allows the user to either color the surface according to the surface Elevation or any other data component exported by krig_3d_geology.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Explode** [Number] Accepts the Explode distance from other modules
- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krig data into geologic layers.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Explode** [Number] Outputs the Explode distance to other modules
- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.
- **Surface** [Renderable]: Outputs to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls Z Scale and Explode distance
- Surface Settings: controls translation, hierarchy and surface selection
- Data Settings: controls clipping, processing (Log) and clamping of input data and kriged outputs.

geologic_surface

This module allows visualization of the topology of any single surface.

geologic_surface can explode the geologic surface analogous to how explode_and_scale explodes layers created by 3d_geology_map or krig_3d. The ability to explode the surface is integral to this module.

geologic_surface also allows the user to either color the surface according to the surface Elevation or any other data component exported by krig_3d_geology.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Explode** [Number] Accepts the Explode distance from other modules
- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krig data into geologic layers.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Explode** [Number] Outputs the Explode distance to other modules
- **Surface Name** [String / minor] Outputs a string containing the selected surface's name
- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.
- **Surface** [Renderable]: Outputs to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls Z Scale and Explode distance
- Surface Settings: controls translation, hierarchy and surface selection
- Data Settings: controls clipping, processing (Log) and clamping of input data and kriged outputs.

indicator_geology

indicator_geology is an alternative geologic modeling concept that uses geostatistics to assign each cell's lithologic material as defined in a pregeology (.pgf) file, to cells in a 3D volumetric grid.

There are two **Estimation Types**:

- Nearest Neighbor is a quick method that merely finds the nearest lithology sample interval among all of your data and assigns that material. It is very fast, but generally should not be used for your final work.
- Kriging provides the rigorous probabilistic approach to geologic indicator kriging. The probability for each material is computed for each cell center of your grid. The material with the highest probability is assigned to the cell. All of the individual material probabilities are provided as additional cell data components. This will allow you to identify regions where the material assignment is somewhat ambiguous. Needless to say, this approach is much slower (especially with many materials), but often yields superior results and interesting insights.

There are also two Lithology Methods when Kriging is selected.

- The default method is block. This method is the quickest since probabilities are assigned directly to cells, and lithology is therefore determined based on the highest probability among all materials. However the resulting model is "lego-like" and therefore requires high grid resolutions in x, y & z in order to give good looking results.
- The other method is Smooth. With Smooth, probabilities are assigned to nodes. In much the same way as analytical data, nodal data for probabilities provides an inherently higher effective grid resolution because after kriging probabilities to the nodes, there is an additional step where we "Smooth" the grid by interpolating between the nodes, cutting the blocky grid and forming a new smooth grid. MUCH lower grid resolutions can be used, often achieving superior results.

Module Input [Ports](#)

- **Input Geologic Field** [Field] Accepts a data field from krig_3d_Geology to krig data into geologic layers.
- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Refine Distance** [Number] Accepts the distance used to discretize the lithologic intervals into points used in kriging.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Field** [Field] Contains the volumetric cell based indicator geology lithology (cell data representing geologic materials).
- **Filename** [String / minor] Outputs a string containing the file name and path. This can be connected to other modules to share files.
- **Refine Distance** [Number] Outputs the distance used to discretize the lithologic intervals into points used in kriging or displayed in post_samples as spheres.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Grid Settings: control the grid type, position and resolution
- Krig Settings: control the estimation methods
 - NOTE: The *Quick Method* assigns the lithologic material cell data based on the nearest lithologic material (in anisotropic space) to your PGF borings. This is done based on the cell center (coordinates) and an enhanced refinement scheme for the PGF borings. *In general the Quick Method should not be used for final results*

Advanced Variography Options:

It is far beyond the scope of our Help to attempt an advanced Geostatistics course. The terminology and variogram plotting style that we use is industry standard and we do so because we will not provide detailed technical support nor complete documentation on these features, which would effectively require a geostatistics textbook, in our help.

However, we have offered an online course on how to take advantage of the complex, directional anisotropic variography capabilities in krig_3d (which applies equally well to indicator_geology and adaptive_indicator_krig), and that course is available as a recorded video class. This class is focused on the mechanics of how to employ and refine the variogram anisotropy with respect to your data and the physics of your project such as contaminated sediments in a river bottom. The variogram is displayed as an ellipsoid which can be distorted to represent the Primary and Secondary anisotropies and rotated to represent the Heading, Dip and Roll. Overall scale and translation are also provided as additional visual aids to compare the variogram to the data, though these do not affect the actual variogram.

We are not hiding this capability from you as the *Anisotropic Variography Study* folder of *Earth Volumetric Studio Projects* contains a number of sample applications which demonstrate exactly what is described above. However, we assure you that understanding how to apply this to your own projects will be quite daunting and really does require a number of prerequisites:

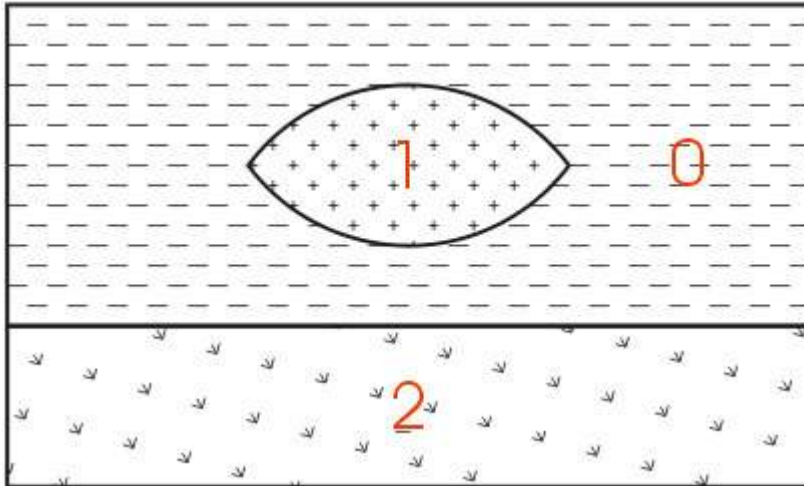
- A thorough explanation of these complex applications
- A reasonable background in Python and how to use Python in Studio
- An understanding of all of the variogram parameters and their impact on the estimation process on both theoretical datasets as well as real-world datasets.

This 3 hour course addresses this issues in detail.

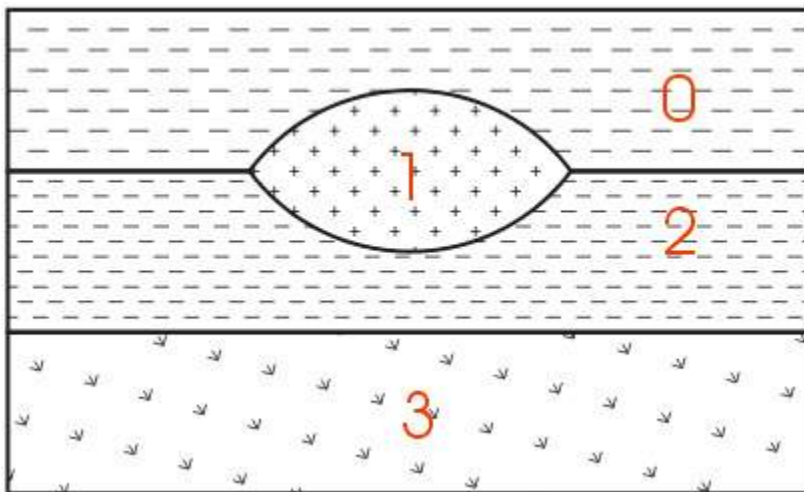
Discussion of Lithologic (Geologic Indicator Kriging) vs. Stratigraphic (Hierarchical) Geologic Modeling

Stratigraphic geologic modeling utilizes one of two different ASCII file formats (.geo and .gmf) which contain "interpreted" geologic information. These two file formats both describe points on each geologic surface (ground surface and bottom of each geologic layer), based on the assumption of a geologic hierarchy.

The easiest way to describe geologic hierarchy is with an example. Consider the example below of a clay lens in sand with gravel below. Some borings will see only sand above the gravel, while others will reveal an upper sand, clay, and lower sand.

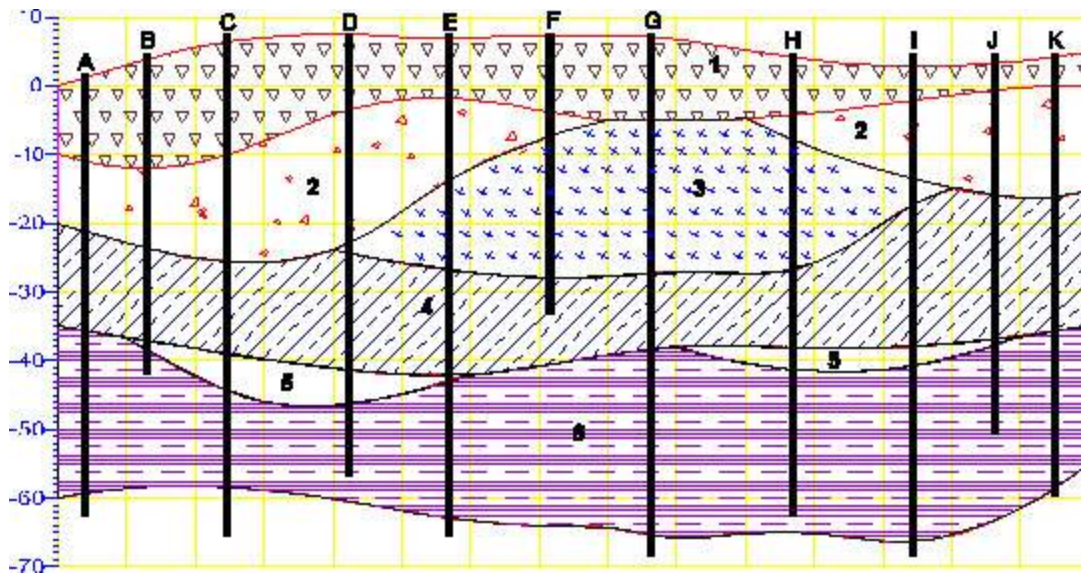


The geologic hierarchy for this site will be upper sand, clay, lower sand, and gravel. This requires that the borings with only sand (above the gravel) be described as upper sand, clay, and lower sand, with the clay described as being zero thickness. For this simple example, determining the hierarchy is straightforward. For some sites (as will be discussed later) it is very difficult or even impossible.



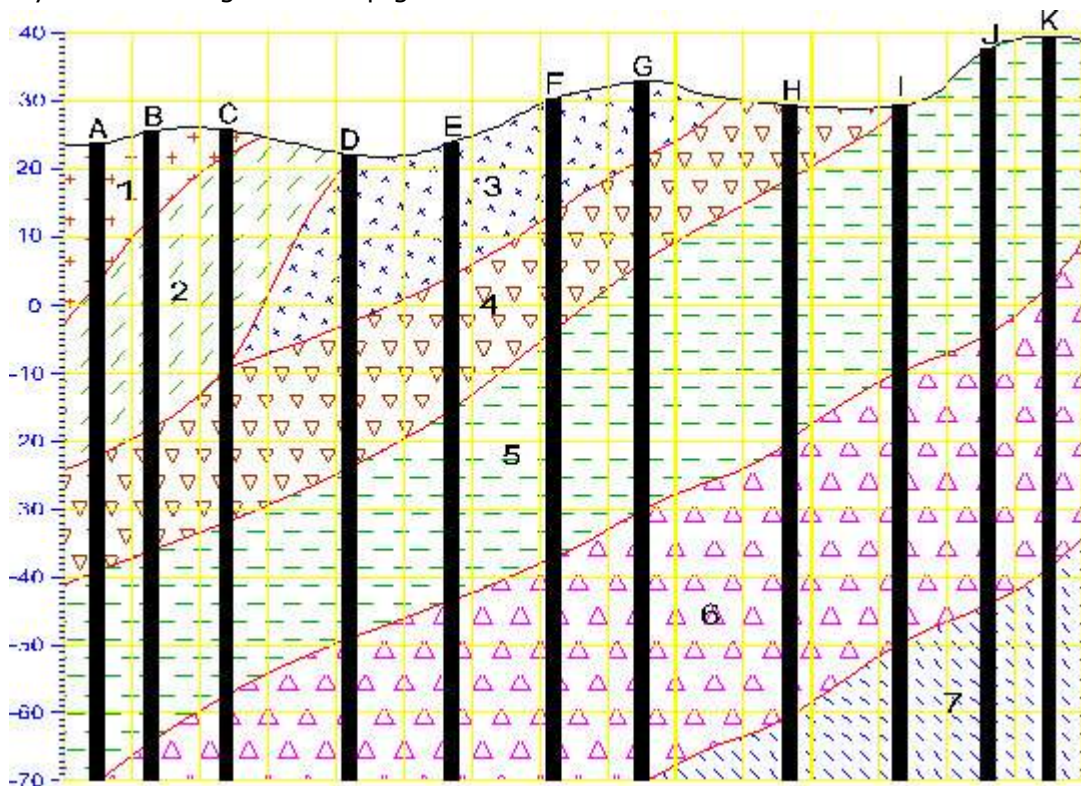
For those sites that can be described using the above method, it remains the best approach for building a 3D geologic model. Each layer has smooth boundaries and the layers (by nature of hierarchy) can be exploded apart to reveal the individual layer surface features. In the above example, the numbers represent the layer numbers for this site (even though layers 0 and 2 are both sand). Two examples of much more complex sites that are best described by this original approach are shown below.

Geologic Example: Sedimentary Layers and Lenses



Geology Example & Figure: Outcrop of Dipping Strata

EVS is not limited to sedimentary layers or lenses. The figure below shows a cross-section through an outcrop of dipping geologic strata. EVS can easily model the layers truncating on the top ground surface.



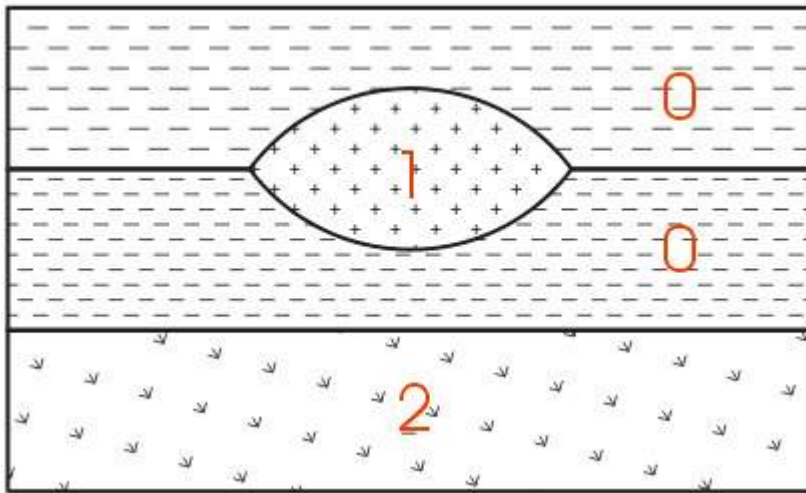
However, many sites have geologic structures (plutons, karst geology, sand channels, etc.) that do not lend themselves to description within the context of hierarchical layers. For these sites, Geologic Indicator Kriging (GIK) offers the ability to build extremely complex models with a minimum of effort (and virtually no interpretation) on the part of the geologist. GIK can also be a useful check of

geologic hierarchies developed for sites that do lend themselves to a model based upon hierarchical layers.

GIK uses raw, uninterpreted 3D borings logs as the input file. The .pgf (pre-geology file) format is used to represent these logs. PGF files contain descriptions of each boring with x,y, & z coordinates for ground surface and the bottom of each observed geologic unit. Consecutive integer values (e.g. 0 through n-1, for n total observed units in the site) are used to describe each material observed in the entire site.

NOTE: It is important to start your material ID numbering at zero (0) instead of 1.

Usually, materials are numbered based upon a logical classification (such as porosity or particle size), however the numbering can be arbitrary as long as the numbers are consecutive (don't leave numbers out of the sequence). For the example given above, we could number the materials as shown in the figure below (even though it is not a numbering sequence based on porosity or particle size).

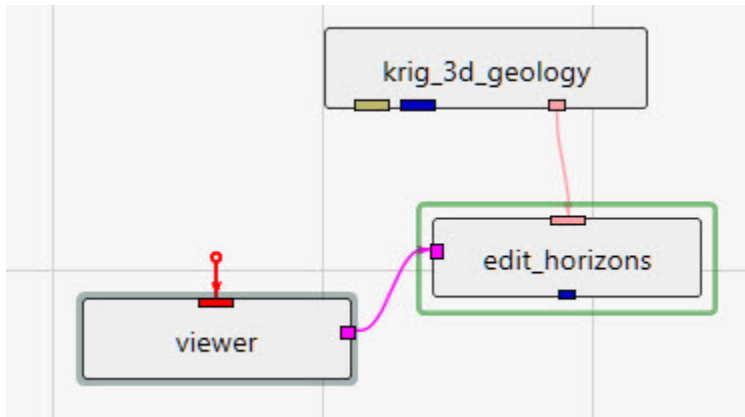


For a .pgf file, borings that do not see the clay (material 2 in the figure) would not need to consider the sand as being divided into upper and lower. Rather, every boring is merely a simple ASCII representation of the raw borings logs. The only interpretation involves classification of the observed soil types in each boring and assigning an associated numbering scheme.

edit_horizons

edit_horizons is an interactive module which allows you to probe points to be selectively added to the creation of each and every stratigraphic horizon. This provides the ability to manual edit horizon surfaces prior to the creation of geologic models.

The method of connecting edit_horizons is unique among our modules. It uses the pink output port from krig_3d_geology as its primary input, and it also requires the purple side port from viewer since it requires interactive probing. Its blue output port then becomes equivalent to the blue output of krig_3d_geology, but with edited surfaces.



Regardless of the estimation method used originally, `edit_horizons` uses Natural Neighbor to perform its near-real-time modifications. For this reason, there is a *Use Gradients* toggle at the top of the user interface, which is identical in function to the one in [krig_3d_geology](#).

The other important parameter at the top of the user interface is the Horizon Point Radius. The default (linked) value for this parameter is computed for you as 2% of the X-Y diagonal extents of your input geology. If any of the original data points for the selected horizon being edited fall within the Horizon Point Radius, then we don't use your probed point based on the assumption that the original data is more defensible and should take precedence.

Next there is the *Probe Action*, which has 3 options:

1. None (default state when the module is instanced)
2. Reset Position (allows you to move points)
3. Add Point (allows you to add new surface control points for the selected horizon)

The *Horizons* list shows all of your geologic horizons. Here, you select the horizon surface you wish to modify. The points that you add only affect the selected horizon. When you change the selected horizon, you can add new points for that surface. You are able to add as many points as you need for any or all of the horizons.

The Horizon Point List is the list of points that you have added by probing in your model. You can only probe on actual objects. These objects can be `geologic_surfaces`, `slices`, `tubes`, or whatever objects you've added to your viewer. Slices are very useful since you can move them where you need them so you can probe points at specific coordinates. You are also able to manually change the X, Y, and/or Z coordinates or any point as needed. For each point, a Note: box is provided so you can keep a record of your actions and reasons.

horizon_ranking

The `horizon_ranking` module is used to give the user control over individual surface priorities and rankings. This allows the user to fine tune their hierarchy in ways much more complex than a simple top-down or bottom-up approach.

Module Input Ports

`horizon_ranking` has one input port which receives geologic input from modules like `krig_3d_geology`

Module Output Ports

`horizon_ranking` has one output port which outputs the geologic input with re-prioritized hierarchy

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field from krig_3d or other similar modules.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting field as edges
- **Geologic legend Information** [Geology legend] Outputs the geologic material information

material_mapping

This module can re-assign data corresponding to:

- Geologic Layer
- Material ID
- Indicator
- Adaptive Indicator

for the purpose of grouping. This provides great flexibility for exploding models or coloring.

Groups are processed from Top to Bottom. You can have overlapping groups or groups whose range falls inside a previous group. In that event, the lower groups override the values mapped in a higher group.

For example, if you have ten material ids (0 through 9) and you want to have them all be 0 except for 5 & 6 which should be 1, this can be accomplished with two groups:

1. From 0 to 9 Map to 0
2. From 5 to 6 Map to 1

Please note that in the animator, you can animate these values. Each group has From, To and Map To values that are numbered zero through eleven (e.g. From0, MapTo5)

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the processed field.

combine_geology

The combine_geology module is used to merge up to six geologic surface (per combine_geology module) to create a field representing multiple geologic layers.

The mesh (x-y coordinates) from the first input field, will be the mesh in the output. The input fields should have the same scale and origin, and number of nodes in order for the output data to have any meaning.

It also has a Run toggle (to prevent downstream modules from firing during input setting changes).

combine_geology provides an important ability to merge sets of surfaces or add additional surfaces to geologic models. It is important to understand the consequences of doing so and the steps that must be taken. The Brown-Grey-Light Brown-Beige port contains the material_ID numbers and names and it is important that the content of this port reflect the current set of surfaces/layers reflected in the geology. When Material_ID or Geo_Layer is presented in a legend, this port is necessary to automatically provide the layer names. When combine_geology is used to construct modified geologic horizons, its Geologic legend Information port MUST be used vs. the same port in [krig_3d_geology](#)

Module Input [Ports](#)

- **Input Geologic Field [Field]** Accepts a field with data whose grid will be exported.
- **Input Field 1 [Field]** Accepts a data field.
- **Input Field 2 [Field]** Accepts a data field.
- **Input Field 3 [Field]** Accepts a data field.
- **Input Field 4 [Field]** Accepts a data field.
- **Input Field 5 [Field]** Accepts a data field.

Module Output [Ports](#)

- **Geologic legend Information [Geology legend]** Supplies the geologic material information for the legend module.
- **Output Geologic Field [Field]** Outputs the field with selected data
- **Output Object [Renderable]:** Outputs to the viewer.

subset_layers

The subset_layers module allows you to subset the output of krig_3d_geology so that downstream modules (krig_3d, 3d_geology_map, Geologic Surface) act on only a portion of the layers kriged.

subset_layers is used to select a subset of the layers (and corresponding surfaces) export from krig_3d_geology. This is useful if you want (need) to krig parameter data in each geologic layer separately.

This is not normally needed with contaminant data, but when you are kriging data such as porosity that is inherently discontinuous across layer boundaries, it is essential that each layer be kriged with data collected ONLY within that layer.

Subset_layers eliminates the need for multiple krig_3d_geology modules reading data files that are subsets of a master geology. Inserting subset_layers between krig_3d_geology and krig_3d allows you to select one or more layers from the geology.

This functionality is very useful when you want to krig groundwater and soil data using a single master geology file that represents both the saturated and unsaturated zones.

Module Input [Ports](#)

- **Input Geologic Field [Field]** Accepts a data field from krig_3d_geology to krig data into geologic layers.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Geologic Field** [Field] Can be connected to the krig_3d, 3D_Geology Map, and geologic_surface(s) modules.

make_single_layer

The make_single_layer module allows you to subset the output of krig_3d_geology so that downstream modules (krig_3d, 3d_geology_map, Geologic Surface) act on only a single merged layer.

make_single_layer is used to merge all layers (and corresponding surfaces) export from krig_3d_geology into a single layer (topmost and bottommost surfaces).

Make_single_layer eliminates the need for multiple krig_3d_geology modules reading data files that are single layer subset of a master geology. Inserting make_single_layer between krig_3d_geology and krig_3d krige all data into a single geologic layer. When used with subset_layers it allows for creating a single layer that represents a only a portion (subset) of the master geology file.

Module Input [Ports](#)

- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krige data into geologic layers.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Geologic Field** [Field] Can be connected to the krig_3d, 3D_Geology Map, and geologic_surface(s) modules.

displace_block

displace_block receives any 3D field into its input port and outputs the same field translated in z according to a selected nodal data component of an input surface allowing for non-uniform fault block translation.

This module allows for the creation of tear faults and other complex geologic structures. Used in conjunction with surf_cut it makes it possible to easily model extremely complex deformations.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a volumetric field
- **Input Surface** [Field] Accepts a 2D surface grid with elevation nodal data. This type of grid is created by krig_3d_geology and raster_to_geology.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the displaced field
- **Output Object** [Renderable]: Outputs to the viewer.

post_samples

The post_samples module is used to visualize:

- Sampling locations and the values of the properties in .apdv files
- The lithology specified in a pgf or .geo files
- The location and values of well screens in a .aidv file

along with a representation of the borings from which the samples/data were collected. The post_samples module has the capability to process property values to make the posted data values consistent with data used in kriging modules. Data can be represented as spheres or any user specified glyph. The sampling locations may be colored and sized according to the magnitude of the property value, and labels can be applied to the sampling locations with several different options.

Each sampling location can be probed for data by holding the alt button and left-clicking on the sample location.

The post_samples module can also represent downhole geophysical logs or Cone Penetration Test (CPT) logs with tubes which are colored and/or sized according to the magnitude of the data. It can display nonvertical borings and data values collected along their length, and can also explode borings and sample locations to show their correct position within exploded geologic layering.

When used to read geology files, post_samples will place surface indicators at the top (ground) surface and the bottom of each geologic layer that are colored according to the layer they depict. When a geology file (.geo or .gmf) is exploded without using geologic surface input from krig_3d_geology there will be surface indicators at the top and bottom of each layer. You may color the borings by lithology.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Explode** [Number] Accepts the Explode distance from other modules
- **Date** [Number] Accepts a date to interpolate time domain data
- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krig data into geologic layers.
- **Subsetting Feature** [Field] Accepts a (1D) line or (2D) surface to be used to subset the borings.
- **Sample Glyph** [Field] Allows the user to import a field containing a geometric object which will be the glyph displayed at each sample location.
- **Filename** [String / minor] Allows the sharing of file names between similar modules.

Module Output [Ports](#)

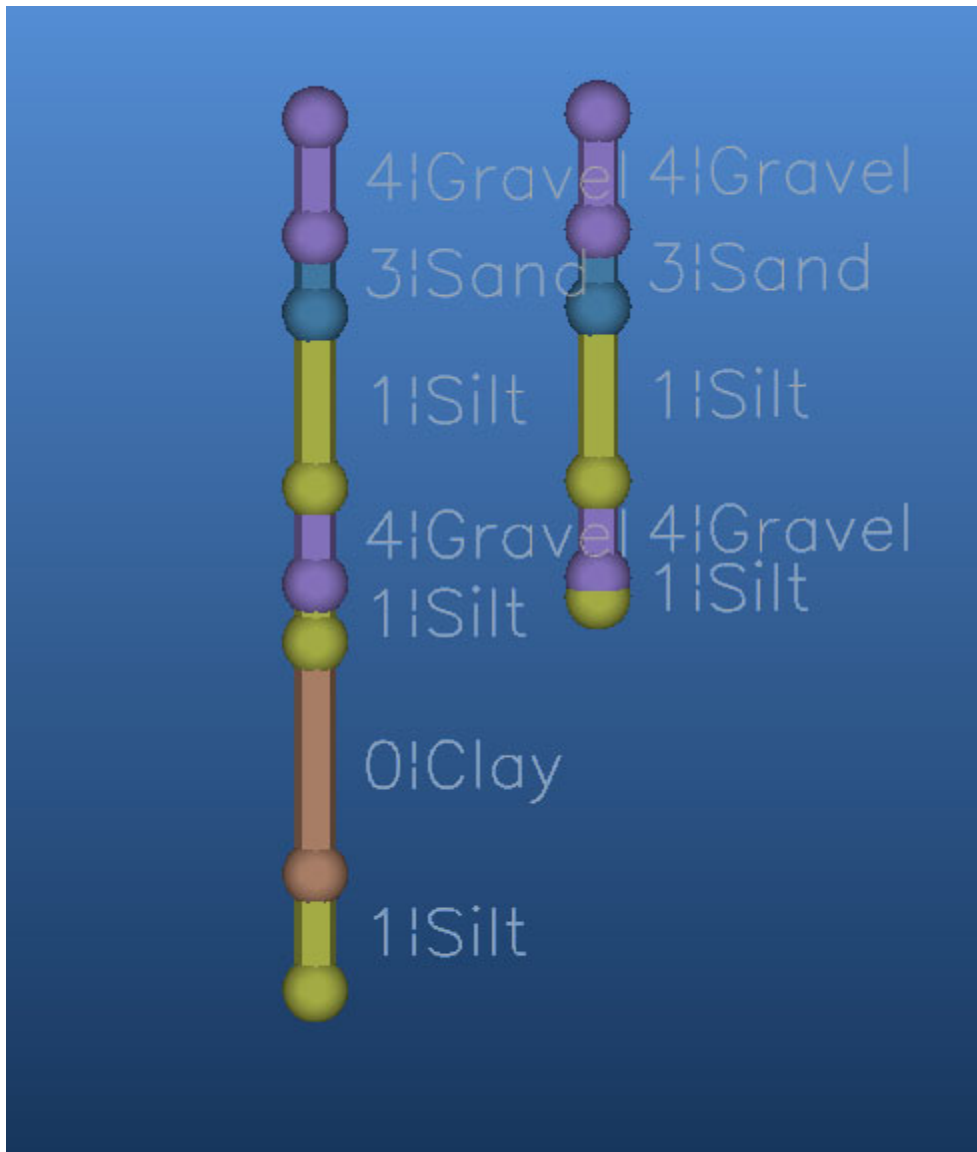
- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Explode** [Number] Outputs the Explode distance to other modules
- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Boring Tubes** [Field / minor] Outputs the tube paths as lines with data
- **Boring Data** [Field / minor] Outputs the tube paths as lines with data

- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Analytes Name** [String / minor] Outputs a string containing the name of the currently selected analyte or date
- **Sample Data** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the Z Scale and selected data component
- Sample Settings: controls clipping, processing (Log) and clamping of data.
- Collapse to 2D: controls how 3D data is subset to 2D
- Geology Settings: controls the display of geologic data
- Time Settings: controls how the module deals with time domain data
- Boring Tube Settings: controls how borings are displayed
- Color Tube Settings: controls the display of colored tubes as an alternative representation (vs. spheres or glyphs)
- Label Settings: parameters association with labeling of borings and samples
 - PGF Files have some unique labeling options so that the lithology names and/or numbers can be labeled at appropriate locations
 - The *PGF Labeling* option determines where labels will be placed along the boring. The options are:
 - At sample
 - At mid-interval
 - The PGF Format option determines what label will be applied. The options are:
 - Material Number
 - Material Name
 - Number|Name (e.g. Material number 'Pipe symbol' Material Name) as shown below with the "mid interval" option.



explode_and_scale

The explode_and_scale module is used to separate (or explode) and apply a scaling factor to the vertical dimension (z-coordinate) of objects in a model.

explode_and_scale can also translate the fields in the z direction, and control the visibility of individual cell sets (e.g. geologic layers).

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Explode** [Number] Accepts the Explode distance from other modules
- **Input Field** [Field] Accepts a data field from krig_3d or other similar modules.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules

- **Explode** [Number] Outputs the Explode distance to other modules
- **Output Field** [Field / minor] Outputs the field with the scaling and exploding applied.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the scaling, exploding and Z translation
- Explode And Scale Settings: controls layer exploding and cell sets

plume_shell

The plume_shell module creates the external faces of a volumetric subset of a 3D input. The resulting closed volume "shell" generally is used only as a visualization of a plume and would not be used as input for further subsetting or volumetric computations since it is hollow (empty). This module creates a superior visualization of a plume as compared with other modules such as plume passing to external_faces and is quicker and more memory efficient.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Isolevel** [Number] Accepts the subsetting level.

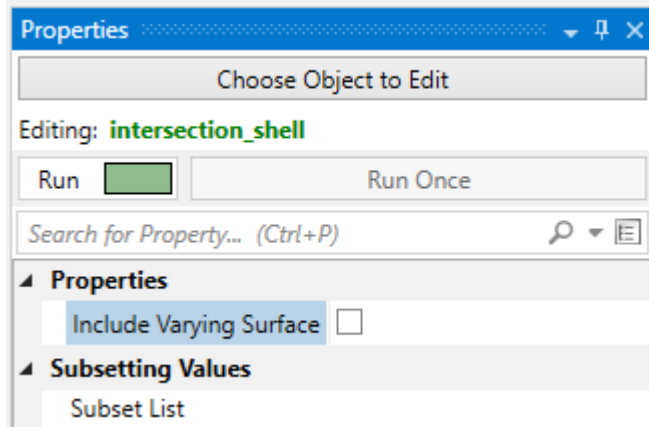
Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting field as a closed surface.
- **Status** [String / minor] Outputs a string containing a description of the operation being performed (e.g. TCE plume above 4.00 mg/kg)
- **Isolevel** [Number] Outputs the subsetting level.
- **Plume** [Renderable]: Outputs to the viewer.

intersection_shell

The intersection_shell is a powerful module that incorporates some of the characteristics of plume_shell, yet allows for a large number of sequential (serial) subsetting operations, just like [intersection](#).

To get the functionality of (the now deprecated) constant_shell module, you would turn off *Include Varying Surface*.



Because this module has "intersection" in its name, it allows you to add any number of subsetting operations.

Each operation can be "Above" or "Below" the specified *Threshold* value, which in Boolean terms corresponds to:

- **A and B** where both the A & B operations are set to Above or
- **A and (NOT B)** where the A operation is set to above and the B operation is set to Below.

However the operator is always **"and"** for *intersection* modules. If you need an **"or"** operator to achieve your subsetting, you need the [union](#) module.

This module creates an efficient and superior visualization of a plume that can be sent directly to the viewer for rendering. The *intersection_shell* module outputs a specialized version of a sequentially subset plume that is suitable for VRML export for 3D printing to create full color physical models.

For output to 3D printing, please jump to the [Issues for 3D Printing](#) topic.

Without *intersection_shell* it is very difficult if not impossible to create a VRML file suitable for printing, especially with complex models.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

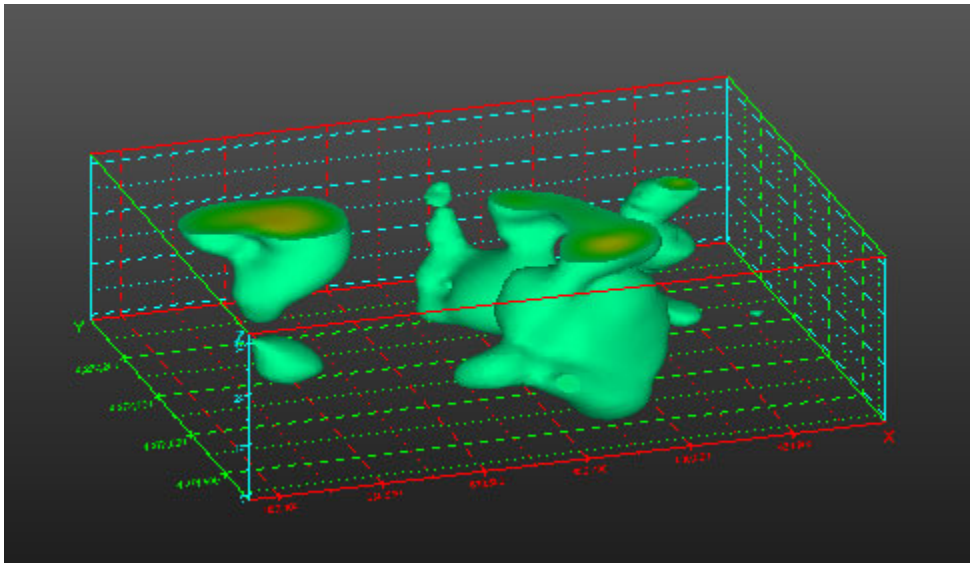
- **Output Field** [Field] Outputs the subsetted field as a closed surface.
- **Output Object** [Renderable]: Outputs to the viewer.

intersection_shell is the module that can create an ISOSURFACE. In other words, a surface (not volume) representing part(s) of your plume.

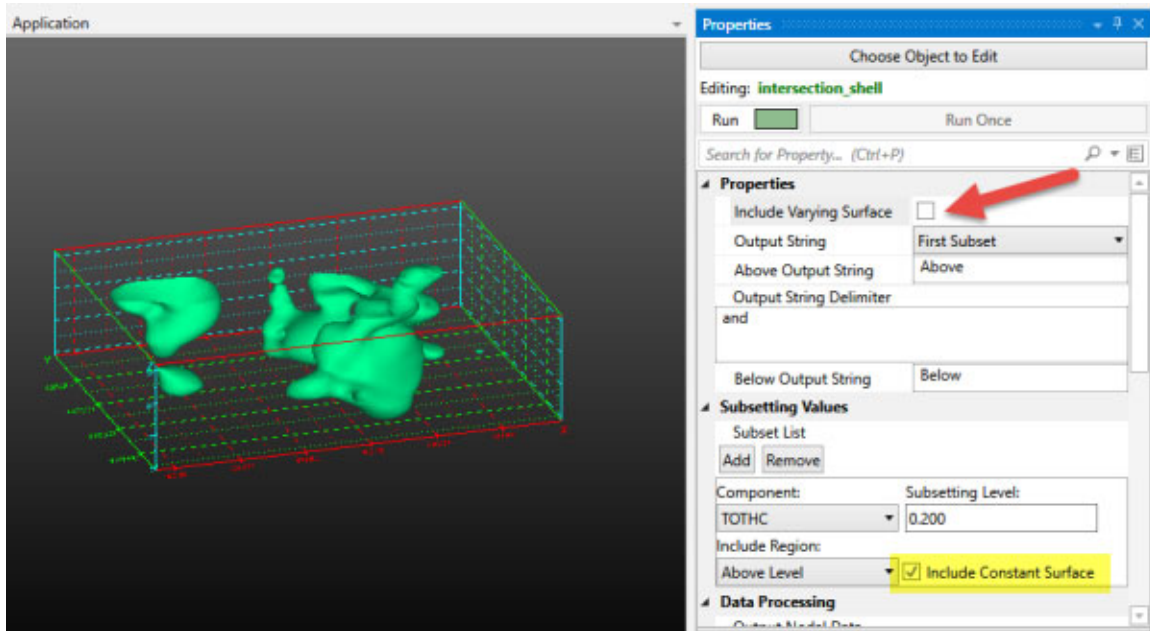
It has two (+) toggles which control the visibility of a plume "shell".

In general a plume external shell has two components: That portion which is exactly EQUAL to the Subsetting Level That portion which is greater than the Subsetting Level

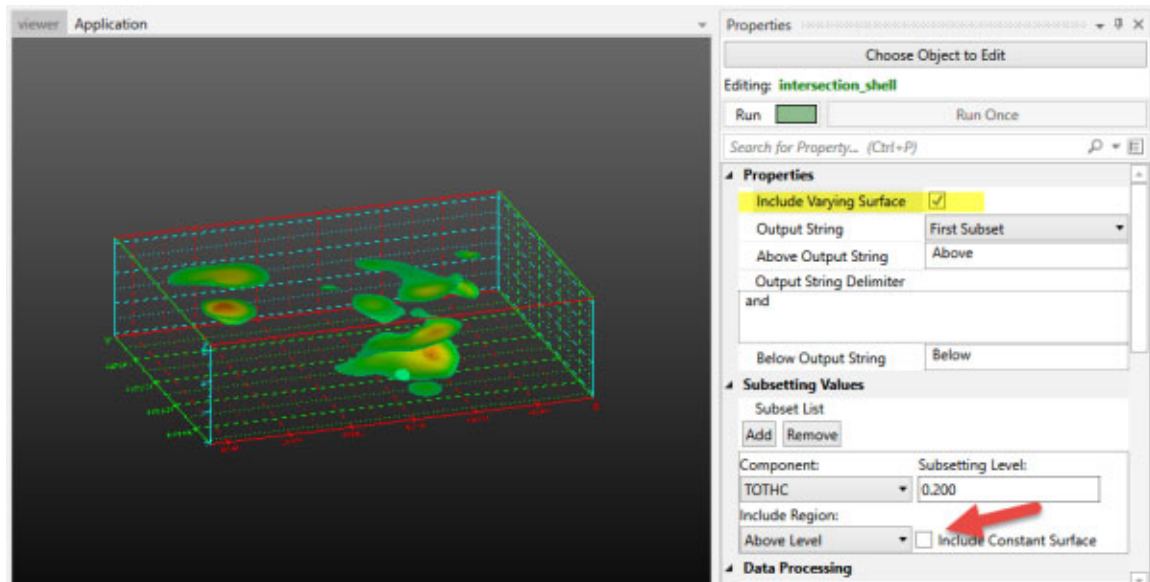
When both toggles are on (default) the plume is



If you display only the Constant Surface (component 1) you get this



If you display only the Varying Surface (component 2) you get this



change_minmax

The change_minmax module extends the capabilities of the now deprecated set_minmax module by allowing setting of max values above the true maximum data range and min values below the true minimum data range. This functionality is commonly needed for color mapping of time-series data. For example, the user can set the minmax values to bracket the widest range achieved for many datasets thus allowing consistent mapping from dataset to dataset during a time-series animation. This way 100 ppm would always be red throughout the animation, and if one dataset did not reach a maximum of 100 ppm, there would be no red color mapping for those time-steps.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with altered data min/max values
- **Output Object** [Renderable]: Outputs to the viewer.

contour_data

contour_data provides a means to color surface and volumetric objects in solid colored bands vs. the default Gouraud shading (smoothly changing colors). contour_data can contour by both nodal and cell data.

This module does not do subsetting like plume_shell , plume. It is used in conjunction with these modules to change the way their output is colored.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Contour Levels** [Contours]: Accepts an array of values representing values to place contours

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with altered data min/max values
- **Output Contour Levels** [Contours]: Outputs an array of values representing values to be labeled in the legend.
- **Output Object** [Renderable]: Outputs to the viewer.

volume_renderer

Volume_renderer directly renders a 3D uniform field using either the Back-to-Front (BTF) or Ray-tracing volume rendering techniques. The Ray-tracing mode is available to both OpenGL and the software renderer. The BTF renderer, which is configured as the default, is available only in the OpenGL renderer.

The basic concept of volume rendering is quite different than anything other rendering technique in EVS. Volume_renderer converts data into a fuzzy transparent cloud where data values at each point in a 3D grid are represented by a particular color and opacity.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Object** [Renderable]: Outputs to the viewer.

adjust_opacity

adjust_opacity provides a means to adjust the opacity (1 - transparency) of any object based on its data values using a simple ramp function which assigns a starting opacity to values less than or equal to the Level Start and an ending opacity to values greater than or equal to the Level End. The appearance of the resulting output is often similar in appearance to volume rendering. adjust_opacity converts data into partially transparent surfaces where data values at each point in a grid are represented by a particular color and opacity.

NOTE: Any module connected after adjust_opacity MUST have Normals Generation set to Vertex (if there is a Normals Generation toggle on the module's panel, it must be OFF).

1. The leftmost port accepts an input field

Module Output Ports

1. The output field which passes the original data with a special new "opacity" data component for use with downstream modules (e.g. slice, plume_shell, etc.)
2. The (red) port for connection to the viewer.

illuminated_lines

Display of Illuminated Lines using texture mapped illumination model on polylines with line halo and animation effects.

Prerequisites

This module requires OpenGL rendering to be selected. This module utilizes special OpenGL calls to implement the illuminated line technique. If this module is used with another renderer, such as the software renderer or the output_images module (not set to Automatic), lines will be drawn in the default mode with illuminated line features disabled.

This module requires the input mesh to contain one Polyline cell set. Any other type of cell set will be rejected, and any additional cell sets will be ignored. Any scalar node data may be present, or none for purely geometric display.

Animation Effects

Ramped/Stepped This choice selects the style of effect variation. Stair creates a linearly increasing or decreasing value, while step makes a binary chop effect. In *Ramped* mode, the blending can be selected to start small then get big, or the reverse or both. The values are *down*, *up*, *up&down* respectively. *Stepped* causes abrupt changes in effect.

AnimatedLength This slider sets the length of the effect along the polyline.

AnimationSpacing This slider sets the spacing between effects along the line.

ModulateOpacity In this mode the line segment varies in transparency from completely transparent to opaque.

ModulateWidth In this mode the line width is varied between 1 (very thin) to fat, based on the effect modes and shape controls.

Reverse Effect As the animation effect is applied between two zones, such as the dash and the space between the dash, this toggle reverses the area where the effect is applied.

Halo Parameters

Halo Width The width control for the halo effect defines the size of the transparent mask region added to the edge of each line. A value of zero turns off the halo effect.

Illuminated Lines Shading Model

AmbientLighting This value provides a base shadow value, a constant added to all shading values.

DiffuseLighting Pure diffuse reflection term, amount of shading dependent on light angle

SpecularHighlights Amount of specular reflection hi-lights based on light and viewer angle

Specular Focus Tightness of specular reflection, low values are dull, wide reflections, high values are small spot reflections.

Line Width Controls line width. Normal 1-pixel lines are 1, can be increased in whole increments. Wide lines are drawn in 2D screen space, not full 3D ribbons. If you want full ribbons, use streamline module ribbon mode.

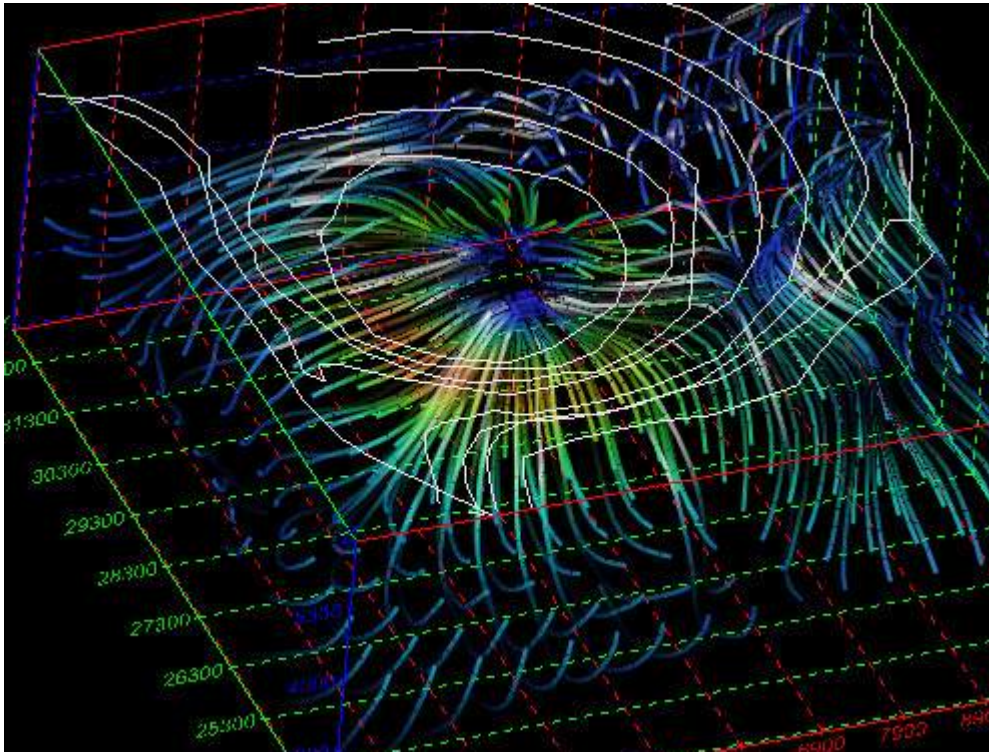
Line Opacity Variable transparency of all lines. A value of 1.0 is fully opaque, while a value of zero makes lines invisible.

DataColor Blending If node data is present, this controls the relative mix of data color and shading color. A value of zero sets full contribution of data color, while at 1.0 no data color is used and the line shade is dominated by illumination effects.

Smooth Shading This enables an additional interpolation mode for blended node data colors. In the off state, data is sampled once per line segment. When enabled, linear interpolation is used between end points of each segment. This can be helpful if large gradients are present on low resolution polylines.

Antialias This effect, sometimes called "smooth lines" blends the drawing of lines to create a smooth effect, reducing the effects of "jaggies" at pixel resolution.

Sort Trans This mode assists visual quality when transparency or antialiasing modes are used, helping to reduce artifacts caused by non-depth sorted line crossings.



texture_wave

The texture_wave module utilizes transparency and texture mapping similar to [texture_colors](#) and [illuminated_lines](#) technology to create an animated effect. However, unlike illuminated_lines, this module works with both OpenGL and Software Rendering.

texture_wave has a single input port that accepts the grid with nodal data that you want to color with this technique. This would normally be tubes or streamribbons.

The **Phase** is the parameter that changes during the animation loop.

Number of Steps: determines the number of steps in the animation.

Texture Resolution is the internal resolution of the image used for texture-coloring.

Min Amplitude is the minimum opacity of the objects.

Max Amplitude is the maximum opacity of the objects.

Contrast affects the contrast (similar to color saturation).

In the image below, we used streamlines which are passed to tubes, which are then connected to texture_wave. The transparency, colors, and animation effects on the tubes is all performed by texture_wave.

The viewer window is shown below.

slope_and_aspect

The slope_and_aspect module determines the slope and aspect of a surface. The slope is the angle between the surface and the horizon. The aspect is the cardinal direction in degrees (rotating clockwise with 0° being North) that the slope is facing.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field with scalar or vector data.

Module Output [Ports](#)

- **Output Field** [Field] Outputs both slope and aspect data as a field
- **Output Slope Object** [Renderable]: Outputs to the viewer.
- **Output Aspect Object** [Renderable]: Outputs to the viewer.

select_data

The select_data module extracts a single data component from a field. Select_data can extract scalar data components or vector components. Scalar components will be output as scalar components and vector components will be output as vector components.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as faces.
- **Output Object** [Renderable]: Outputs to the viewer.

read_wavefront_obj

The read_wavefront_obj module will only read Wavefront Technologies format .OBJ files which include object textures which are represented (included) as a single image file. Each file set is actually a set of 3 files which must always include the following 3 file types with the same base file name, which must be in the same folder:

1. The .obj file (this is the file that we browse for)
2. A .mtl (Material Template Library) file
3. An image file (e.g. .jpg) which is used for the texture. Note: there must be only ONE image/texture file. We do not support multiple texture files.

This module provides the user with the capability to integrate complex photo-realistic site plans, buildings, and other 3D features into the EVS visualization, to provide a frame of reference for understanding the three dimensional relationships between the site features, and characteristics of geologic, hydrologic, and chemical features.

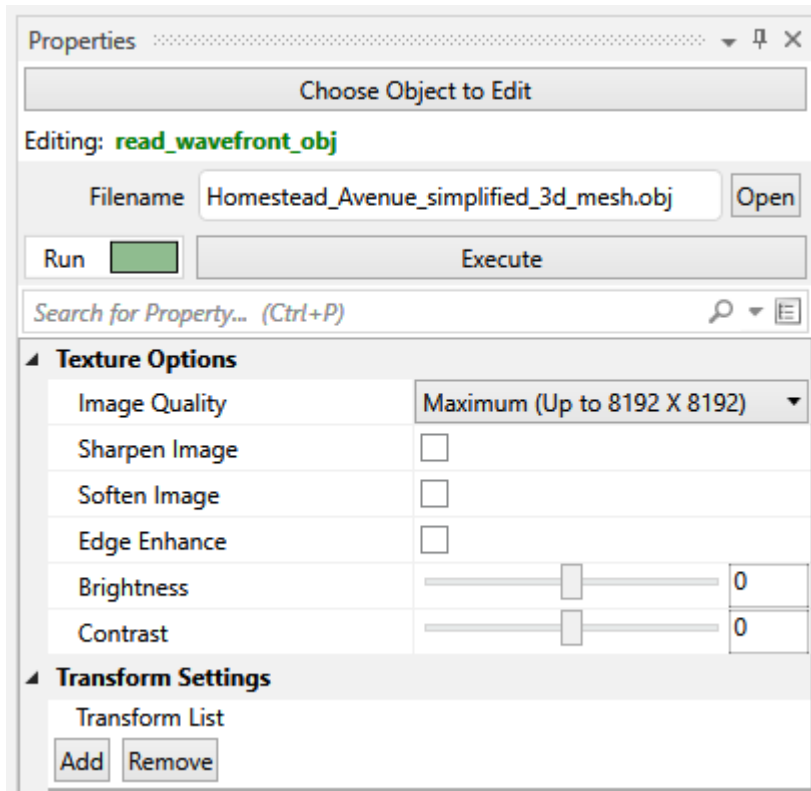
NOTE: This module intentionally does not have a Z-Scale port since this class of files are so often not in a user's model projected coordinate system. Instead we are providing a Transform Settings group that allows for a much more complex set of transformations including scaling, translations and rotations.

Module Output [Ports](#)

- **Output Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window includes the following parameters:



Texture Options: These allow you to enhance the image used for texturing to achieve the best looking final output.

Transform Settings: This allows you to add any number of Translation or Scale transformations in order to place your Wavefront Object in the same coordinate space as the rest of your "Real-World" model. It is very typical that Wavefront Objects are in a rather arbitrary local coordinate system that will have no defined transformation to any standard coordinate projection.

Generally you should know if the coordinates are feet or meters and if those are not correct, do that scaling as your first set of transforms.

It will be up to you to determine the set of translations that will properly place this object in your model. Hopefully rotations will not be required, but they are possible with the Transform List.

volumetrics

The volumetrics module is used to calculate the volumes and masses of soil, and chemicals in soils and ground water, within a user specified constant_shell (surface

of constant concentration), and set of geologic layers. The user inputs the units for the nodal properties, model coordinates, and the type of processing that has been applied to the nodal data values, specifies the subsetting level and soil and chemical properties to be used in the calculation, and the module performs an integration of both the soil volumes and chemical masses that are within the specified constant_shell. The results of the integration are displayed in the EVS Information Window, and in the module output window.

The volumetrics module computes the volume and mass of everything passed to it. To compute the volume/mass of a plume, you must first use a module like plume or intersection to subset your model.

NOTE: Do not use plume_shell or intersection_shell upstream of volumetrics since their output is a hollow shell without any volume.

The volumetrics module computes volumes and masses of analytes using the following method:

- Each cell within the selected geologic units is analyzed
- The mass of analyte within the cell is integrated based on concentrations at all nodes (and computed cell division points)
- The volumes and masses of all cells are summed
- Centers of mass and eigenvectors are computed
- For soil calculations the mass of analyte is directly computed from the computed mass of soil (e.g. mg/kg). This is affected by the soil density parameter (all densities should be entered in gm/cc).
- For groundwater calculations, the mass of analyte (Chemical Mass) is computed by first determining the volume of water in each cell. This uses the porosity parameter and each individual cell's volume. From the cell's water volume, the mass of analyte is directly computed (e.g. mg/liter).
- The volume of analyte (Chemical Volume) is computed from the Chemical Mass using the "Chem Density" parameter (all densities should be entered in gm/cc).

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Explode** [Number] Accepts the Explode distance from other modules
- **Input Field** [Field] Accepts a field with data.
- **String for Output** [String]
- **Input Subsetting Level** [Number] Accepts the subsetting level

Module Output [Ports](#)

- **Output Subsetting Level** [Number] Outputs the subsetting level
- **Soil Volume Level** [Number] Outputs the computed soil volume
- **Soil Mass Level** [Number] Outputs the computed soil mass
- **Chemical Volume Level** [Number] Outputs the computed chemical volume
- **Chemical Mass Level** [Number] Outputs the computed chemical mass

- **Nodal Data Component** [String] The name of the analyte
- **Volume Units** [String] The units of the volume calculations (e.g. m3)
- **Result Value** [Number] The final output
- **Output Second Moment Object** [Renderable]: Outputs to the viewer

You can use the **Geologic Layers** selection list which allows you to choose the cell sets (geologic layers) that you want to perform computations on.

The Soil Density and Porosity inputs allow the user to input the properties of the soil matrix in which the chemicals reside. Note that if the mass of chemicals in a combined soil and ground water plume are to be estimated, one of the geologic layers should be set up to have a boundary within it that corresponds to the water table position. In essence, this will create two layers out of one geologic unit that can be used to separate the soil domain from the ground water domain. The user can then choose the appropriate Nodal Data Units for each layer in the two domains, and obtain volumetrics estimates by summing the results in individual layers. There are several other alternative methods for completed volumetrics estimates in continuous soil and ground water plumes, which involve either setting up separate soil and ground water models, or using the Field Math module to remove and include specified areas of the domains.

The Chemical Density input allows the user to input the density of the chemical constituent for which mass estimates are being completed. Note that this value is used to calculate the volume of chemical in the specified constant_shell, as the mass units are calculated directly from the nodal data.

Volume Dollars is used along with the total volume of the chemical to indicate the cost of the removal of the chemical.

Mass Dollars is used, along with the total chemical mass, to determine the value of the chemical mass.

Volume Units is used to select which units the volume should be calculated in. For the Specified Unit Ratio the units to convert to are liters. For example if your units were Cubic Meters the ratio would be 1000.

Mass Units is used to select which units the mass should be calculated in. For the Specified Unit Ratio the units to convert to are Kilograms.

The **Output Results File** toggle causes volumetrics to write a file to the ctech folder (volumetrics_results.txt) that contains all volumetrics information in a format suitable for input to programs like Excel (tab delimited .txt file). This file is written to in an append mode. It will grow in size as you use volumetrics. You should delete or move the file when you're done with it.

The **Run Automatically** toggle, when selected, causes the module to run as soon as any of the input parameters have changed. When not selected the accept button must be pushed for the module to run.

There is an advanced window that can be opened by checking the Advanced Output Options toggle.

The advance panel provides many capabilities including Spatial Moment Analysis.

- Spatial Moment Analysis involves computing the zeroth, first, and second moments of a plume to provide measures of the mass, location of the center of mass, and spread of the plume.

- The zeroth moment is a mass estimate for each sample event and COC. The estimated mass is used to evaluate the change in total mass of the plume over time.
- The first moment estimates the center of mass of the plume (as coordinates X_c , Y_c , & Z_c).
- The second moment indicates the spread of the contaminant about the center of mass (σ_{xx} , σ_{yy} and σ_{zz}), or the distance of contamination from the center of mass. This is somewhat analogous to the standard deviation of the plume along three orthogonal axes represented as an ellipsoid created using the eigenvalues as the ellipsoid major and minor axes, and the eigenvectors to orient the ellipsoid. The orientation of the ellipsoid is aligned with the primary axis of the plume (not the coordinate axes).
- The Second Moment ellipsoid represents the spread of the plume in the x, y and z directions. Freyberg (1986) describes the second moment about the center of mass as the spatial covariance tensor.
- The components of the covariance tensor are indicative of the spreading of the contaminant plume about the center of mass. The values of σ_{xx} , σ_{yy} and σ_{zz} represent the axes of the covariance ellipsoid. The volumetrics module provides a scaling parameter that allows you to view the ellipsoid corresponding to the one-sigma (default) or higher sigma (higher confidence) representation of the contaminant spread.

The **Water Density** type in window allows the user to specify the density of water. The default of 0.9999720 g/mL (gm/cc) is the Density of Water at 4.5 degrees Celsius.

The **Output Filetype** radio list is used to select the format of the output file. The default is a tab spaced single line output, the second choice will format the output the same as the display window, and the third option will format the output separated by tabs on multiple lines. Changing these options will not cause the module to run, you must hit accept or change an input value for the module to run.

Overwrite causes the output file to be overwritten instead of appended to. This toggle will only be selected for one run and then will unselect itself and begin appending again, unless it is rechecked. Selecting this toggle will not cause the module to run, you must hit accept or change an input value for the module to run.

The **Date** type in allows you to set the date, which is output only in the Tabbed Multi-Line file.

Connecting the **Red Output Port** of volumetrics to the viewer will display the Second Moment Ellipsoid and the Eigenvectors (if turned on).

The three toggles:

1. **Display Mass Along Major Eigen Vector**
2. **Display Mass Along Minor Eigen Vector**
3. **Display Mass Along Interm(ediate) Eigen Vector**

allow you turn on and off the lines lying along the Major, Minor, and Intermediate Eigenvectors. These vectors represent the second moment of mass, and by default have chemical data mapped to them. These lines are of the same orientation as the second moment ellipse but they stretch only to the extents of the model. To output these lines the Export Results button must be pushed.

The **Segments In Lines** type in allows you to control the number of segments making up each line, the larger the number of segments the closer the node data along the line will match the node data of the model.

The **Color Lines by Axis** toggle strips the node data from the lines leaving them colored by the axis they represent.

EllipsoidResolution is an integer value determines the number of faces used to approximate the analytically smooth ellipsoid. The higher the resolution the smoother the ellipsoid.

EllipsoidScale is a scaling factor for the second moment ellipsoid. A value of 1.0 (default) is analogous to one-sigma (67%) statistical confidence. Higher values would provide an indication of the size of the eigenvalues with a higher statistical confidence.

cell_volumetrics

The cell_volumetrics module provides cell by cell volumetrics data. It creates an extremely large output file with volume, contaminant mass and cell centers for every cell in the grid.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Explode** [Number] Accepts the Explode distance from other modules
- **Input Field** [Field] Accepts a field with data.
- **String for Output** [String]
- **Input Subsetting Level** [Number] Accepts the subsetting level

Module Output [Ports](#)

- **Output Subsetting Level** [Number] Outputs the subsetting level

area_integrate

The area_integrate module is used to calculate the areas of the entire field input. The input data to area_integrate must be a two dimensional data field output from krig_2d, slice, or any subsetting module which outputs two-dimensional data (slice, plume with 2D input, or plume_shell). The results of the integration are updated each time the input changes.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field which is a surface.

Module Output [Ports](#)

- **Output Area** [Number] The area in user units squared
- **Units** [String] The units (e.g. ft or m)

file_statistics

The file_statistics module is used to check the format of: *.apdv; *.aidv; *.geo; *.gmf; *.vdf; and *.pgf files, and to calculate and display statistics about the data contained in these files. This module also calculates a frequency distribution of properties in the file. During execution, file_statistics reads the file, displays an error

message if the file contains errors in format or numeric values, and then displays the statistical results in the EVS Information window

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Filename** [String / minor] Allows the sharing of file names between similar modules.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Sample Data** [Field / minor] Outputs the data as points (size of points can be controlled).
- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Mean Level** [Number] Outputs the mean data value
- **Median Level** [Number] Outputs the median data value
- **Min Level** [Number] Outputs the minimum data value
- **Max Level** [Number] Outputs the maximum data value
- **Number Of Points** [Number] Outputs the number of points
- **Statistics** [String / minor] Outputs a string containing the full output normally sent to the Information window
- **Sample Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Data Processing: controls clipping, processing (Log) and clamping of input data
- Time Settings: controls how the module deals with time domain data

statistics

The statistics module is used to analyze the statistical distribution of a field with nodal data. The data field can contain any number of data components. Statistical analyses can only be performed on scalar nodal data components. An error occurs if a statistical analysis is attempted on vector data. Output from the statistics module appears in the EVS Information Window. Output consist of calculated min and max values, the mean and standard deviation of the data set, the distribution of the data set, and the coordinate extents of the model.

The first port (the leftmost one) should contain a mesh with nodal data. If no nodal data is present, statistics will only report the extents and centroid of your mesh. Data sent to the statistics module for analysis will reflect any data transformation or manipulation performed in the upstream modules. Any mesh data sent to the port is used for calculating the X, Y and Z coordinate ranges. The mesh coordinates have no affect on the data distribution. Cell based data is not used.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Input Geologic Field** [Field] Accepts a data field from upon which statistics are computed

Module Output [Ports](#)

- **Mean Level** [Number] Outputs the mean data value
- **Median Level** [Number] Outputs the median data value
- **Min Level** [Number] Outputs the minimum data value
- **Max Level** [Number] Outputs the maximum data value
- **Number Of Points** [Number] Outputs the number of points
- **Statistics** [String / minor] Outputs a string containing the full output normally sent to the Information window

well_decommission

Groundwater contamination sites worldwide are engaged in regular sampling of monitoring wells with a typical cost of over \$1,000 per well per sampling event. Many of these wells are redundant or geostatistically insignificant and can be decommissioned. The well_decommission module analyzes all available data and quantifies the impact to site assessment quality of removing each well. The well_decommission module offers the following functionality:

1. Provides an easy to use method to determine which, if any, wells can be decommissioned.
2. Performs baseline analysis using all data.
3. Provides a justifiable approach for determining candidate wells for decommissioning.
4. Can save thousands of dollars per year for each well identified for decommission.

Module Input [Ports](#)

- **Input External Grid** [Field / minor] Allows a grid to be imported into the module for the purposes of kriging. If a grid is imported the selected file will be ignored for gridding purposes.
- **Filename** [String / minor] Allows the sharing of file names between similar modules.

Module Output [Ports](#)

- **Output Field** [Field / minor] Outputs the subsetted field as faces.
- **Filename** [String / minor] Allows for the sharing of the name of the well decommission file that is created after the analysis at each cycle is complete.
- **Status Information** [String / minor] Outputs a string containing module parameters. This is useful for connection to save_evs_field to document the module operation.

- **Analytes Name** [String / minor] Outputs a string containing the name of the currently selected analyte or date
- **Target Concentration** [Number] Outputs the target concentration level.
- **Sphere Display Component** [String / minor] Outputs the name of the displayed data component
- **Output Object** [Renderable]: Outputs to the viewer.

Outputs

The primary output of well_decommission are 8 metrics which are assigned to each well. The metrics are listed below.

Area Deviation is the absolute value of the difference between the baseline plume area and the area of the plume with each of the wells dropped.

(Pseudo) Mass is the integral of concentration * area over the surface of the plume. It is not a true mass because that would require a volume vs. an area.

Mass Deviation is the absolute value of the difference between the baseline plume mass and the mass of the plume with each of the wells dropped. It is indicative of the amount of contaminant mass error that would result from dropping each well.

- Maximum Area Deviation (the greater of the three below)
 1. Deviation of Min (plume) area
 2. Deviation of Nominal (plume) area
 3. Deviation of Max (plume) area
- Maximum Mass Deviation (the greater of the three below)
 1. Deviation of Min (plume) Mass
 2. Deviation of Nominal (plume) Mass
 3. Deviation of Max (plume) Mass

Legend

The legend module is used to place a color scale bar in the viewer window. The legend shows the relationship between the selected data component for a particular module and the colors shown in the viewer. For this reason, the legend's RED input port must be connected to the RED output port of a module which is connected to the viewer and is generally the dominant colored object in view.

Many modules with red output ports have a selector to choose which ONE of the nodal or cell data components are to be used for coloring. The name of the selected data component will be displayed as the Title of the legend if the Label Options are set to Automatic (default).

If the data component to be viewed is either Geo_Layer or Material_ID (for models where the grid is based upon geology), the Geologic legend Information port from krig_3d_geology must also be connected to legend to provide the Geologic Layer (or material) names for automatic labeling. When this port is connected it will have no affect if any other data component is selected.

The minimum and maximum values are taken from the data input as defined in the datamap. Labels can be placed at user defined intervals along the color scale bar. Labels can consist of user input alphanumerical values or automatically determined numerical values.

Module Input [Ports](#)

- **Geologic legend Information** [Geology legend] Accepts the geologic material information from modules that read geologic data.
- **Contour Levels** [Contours]: Accepts an array of values representing values to be labeled in the legend.
- **Input Object** [Renderable]: Accepts the output of a module to which the legend corresponds.

Module Output [Ports](#)

- **Output legend** [Field] Outputs the legend as a field to allow texturing
- **Title Output** [String] Can be connected to the krig_3d, 3D_Geology Map, and geologic_surface(s) modules.
- **Output Object** [Renderable]: Outputs to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Label Options: controls the legend labeling
- Scale Options: controls the legend size and placement

axes

General Module Function

The axes module is used to place 3D axes in the viewer scaled by the model data and/or user defined limits. Axes accepts data from many of the Subsetting and Processing modules and outputs directly to the viewer. Data passed to Axes should come from modules which have scaled or transformed the mesh data, for example [explode and scale](#). Axes generated by axes and displayed in the viewer are transformable with other objects in the viewer.

The User interface to axes is very comprehensive. Each coordinate direction axis can be individually controlled. Axis labels and tick marks for each axes can be specified. The label font, label precision, label orientation, and other label parameters are all user specified. Many of the parameters do not have default values that will produce the desired results because many variables control how the axes should be defined.

axes requires a field input to position and size the axes. If you disconnect the (blue/black) field input port, you no longer lose the axes bounds values and your axes remain in place. This is useful when field data changes in an animation so that you don't constantly recreate the axes.

Also, the size of text and tick marks is based on a percentage of the x-y-z extent of the input field. This now allows you to set the extent of one or more axes to zero so you can have a scale of only one or two dimensions.

Module Input [Ports](#)

- **View** [View] This is the primary Purple port which connects to the viewer to receive the extent of all objects in the viewer AND outputs the axes.
 - This port can be used as your only connection from axes to the viewer and no other connections are needed.
- **Input Geologic Field** [Field] Accepts a field to receive the extent

- **Input Objects** [Renderable]: Accepts a renderable output port to receive the extent
- **Minor Ports not needed for most all cases**
 - **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
 - **Explode** [Number] Accepts the Explode distance from other modules

Module Output [Ports](#)

- **Output Object** [Renderable] Outputs the axes to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the scaling and exploding
- Spatial Definition: Controls the extents and grid densities
- Display Settings: controls layer exploding and cell sets
- All Axes Settings: Controls parameters for XYZ simultaneously
- X Axes Settings: Controls parameters for X axis
- Y Axes Settings: Controls parameters for Y axis
- Z Axes Settings: Controls parameters for Z axis

in_view (Purple) : This port accepts the output of the viewer directly. It will draw the axes around everything displayed in the viewer. This port will only cause the module to run when the port is connected or when the "Accept Current Values" button is pressed. If the models coordinate extents are going to change often then another input port should be used.

objects_in (Red) : This port accepts any number of (Red) output ports from other modules. When any of those modules are run the axes module will run as well.

meshs_in (Blue/Black) : This port accepts any number of (Blue/Black) output ports from other modules. When any of those modules are run the axes module will run as well.

explode (Grey/Green) : This port accepts a float value representing the explode distance from **explode_and_scale**. If you have an explode distance set to anything but 0, the Z axis tick labels are not printed.

z_scale (Grey/Brown) : This port accepts a float value representing Z exaggeration of the model from modules like **explode_and_scale** to ensure that the Z axis is correctly labeled.

north

The north module is used to place a 3D North Arrow or Rose Compass in the 3D viewer scaled by the model data and/or user defined parameters.

Module Input [Ports](#)

- **View** [View] This is the primary Purple port which connects to the viewer to receive the extent of all objects in the viewer AND outputs the north arrow or compass rose.

- This port can be used as your only connection from north to the viewer and no other connections are needed.
- **Minor Ports not needed for most all cases**
 - **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
 - **Explode** [Number] Accepts the Explode distance from other modules

Module Output [Ports](#)

- **Output For Transform** [Renderable] Provides an additional output port if you want to duplicate place_text's output via a transform_group module.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the scaling and positioning
- North Arrow Settings:
- Compass Rose Settings:

add_logo

The add_logo module is used to place a logo or other graphic object in the viewer's non-transformable overlay. It is extremely easy to use. There are sliders to adjust size and position and a button to select the image file to use as a logo.

Module Input [Ports](#)

- **View** [View] Connects to the viewer

Module Output [Ports](#)

- **Output Object** [Renderable] Outputs the logo as a 2D overlay in the viewer.

Titles

Titles connects to the red port on the viewer (as does Color_legend) and provides a straightforward means to place text in the viewer. By using the red port, the text is not transformed by viewer transformations and is positioned using sliders in the Titles user interface.

Module Input [Ports](#)

- **View** [View] This is the primary Purple port which connects to the viewer to receive the extent of all objects in the viewer AND outputs the titles.
 - This port can be used as your only connection from titles to the viewer and no other connections are needed.
- **Input String** [String] Accepts the string to display.
- **Input Objects** [Renderable]: Accepts a renderable output port to receive the extent

Module Output [Ports](#)

- **Output Object** [Renderable]: Outputs to the viewer. NOT REQUIRED when the View port is used.

place_text

place_text replaces both Text3D and MultiText3D and provides a means to interactively place 2D and 3D renderable text strings or to read a [.PT File](#) (or legacy [.EMT file](#)) to place the text.

Module Input [Ports](#)

- **View** [View] This is the primary Purple port which connects to the viewer to receive the extent of all objects in the viewer AND outputs the test.
- This port can be used as your only connection from place_text to the viewer and no other connections are needed.
- **Minor Ports not needed for most all cases**
 - **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
 - **Explode** [Number] Accepts the Explode distance from other modules

Module Output [Ports](#)

- **Output For Transform** [Renderable] Provides an additional output port if you want to duplicate place_text's output via a transform_group module.
- **Minor Ports not needed for most all cases**
 - **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
 - **Explode** [Number] Outputs the Explode distance to other modules

interactive_labels

The interactive_labels module allows the user to place formatted labels at probed locations within the viewer. The data displayed is the data at the probed location

Module Input [Ports](#)

- **Z Scale** [Number / minor] Accepts Z Scale (vertical exaggeration) from other modules
- **Number Variable** [Number / minor] Accepts a number to be used in the expression
- **Input String Variable** [String / minor] Accepts a string to be used in the expression
- **View** [View / minor] Connects to the viewer to allow probing on all objects.

Module Output [Ports](#)

- **Z Scale** [Number / minor] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Number Variable** [Number / minor] Outputs a number to be used in the expression
- **Output String Variable** [String / minor] Outputs a string to be used in the expression

- **Output Object** [Renderable] Outputs to the viewer.

format_string

format_string allows you to construct a complex string (for use in titles or as file names) using multiple string and numeric inputs. An expression determines the content of the output.

Module Input [Ports](#)

- **Date** [Number] Accepts a date
- **Number 1** [Number] Accepts a number
- **Number 2** [Number] Accepts a number
- **Number 3** [Number] Accepts a number
- **Number 4** [Number] Accepts a number
- **String 1** [String] An input string
- **String 2** [String] An input string
- **String 3** [String] An input string

Module Output [Ports](#)

- **Output String** [String] The resultant string output

Note: Strings cannot be formatted or subsetted

NUMBER FORMATTING

The available floating point presentation types are:

- 'e' - Exponent notation. Prints the number in scientific notation using the letter 'e' to indicate the exponent.
- 'E' - Exponent notation. Same as 'e' except it converts the 'e+XX' to uppercase 'E+XX'.
- 'f' - Fixed point. Displays the number as a fixed-point number.
- 'g' - General format. For a given precision $p \geq 1$, this rounds the number to p significant digits and then formats the result in either fixed-point format or in scientific notation, depending on its magnitude.
 - The precise rules are as follows: suppose that the result formatted with presentation type 'e' and precision $p-1$ would have exponent exp . Then if $-4 \leq exp < p$, the number is formatted with presentation type 'f' and precision $p-1-exp$. Otherwise, the number is formatted with presentation type 'e' and precision $p-1$. In both cases insignificant trailing zeros are removed from the significant, and the decimal point is also removed if there are no remaining digits following it.
 - Positive and negative infinity, positive and negative zero, and nans, are formatted as inf, -inf, 0, -0 and nan respectively, regardless of the precision.
 - A precision of 0 is treated as equivalent to a precision of 1.
 - The default precision is 6.

- 'G' - General format. Same as 'g' except switches to 'E' if the number gets to large.
- 'n' - Number. This is the same as 'g', except that it uses the current locale setting to insert the appropriate number separator characters.
- '%' - Percentage. Multiplies the number by 100 and displays in fixed ('f') format, followed by a percent sign.
- '' (None) - similar to 'g', except that it prints at least one digit after the decimal point.

The following are example formats and the resultant output:

- *N1 = 3.141592654 | Expression set to {N1:.4f} | Result is 3.1416*
- *N1 = 12345.6789 | Expression set to {N1:.6e} | Result is 1.234568e+04*
- *N1 = 123456789.0123 | Expression set to {N1:.6G} | Result is 1.23457E+08*
- *N1 = 123456789.0123 | Expression set to {N1:.6g} | Result is 1.23457e+08*
- *N1 = 123456.0123 | Expression set to {N1:.6G} | Result is 123456*
- *N1 = 123456.0123 | Expression set to {N1:.9G} | Result is 123456.012*
- *N1 = 123456.0123 | Expression set to {N1:.5f} | Result is 123456.01230*
- *N1 = 0.893 | Expression set to {N1:.2%} | Result is 89.30%*
- *N1 = 3.141592654 | Expression set to {N1} | Result is 3.141592654*

REFERENCE: <https://www.python.org/dev/peps/pep-3101/>

DATE FORMATTING

Syntax	Description	Example	Notes
%a	Weekday as locale's abbreviated name.	Sun, Mon, ..., Sat (en_US); So, Mo, ..., Sa (de_DE)	(1)
%A	Weekday as locale's full name.	Sunday, Monday, ..., Saturday (en_US); Sonntag, Montag, ..., Samstag (de_DE)	(1)
%w	Weekday as a decimal number, where 0 is Sunday and 6 is Saturday.	0, 1, ..., 6	-
%d	Day of the month as a zero-padded decimal number.	01, 02, ..., 31	-

Syntax	Description	Example	Notes
%b	Month as locale's abbreviated name.	Jan, Feb, ..., Dec (en_US); Jan, Feb, ..., Dez (de_DE)	(1)
%B	Month as locale's full name.	January, February, ..., December (en_US); Januar, Februar, ..., Dezember (de_DE)	(1)
%m	Month as a zero-padded decimal number.	01, 02, ..., 12	-
%y	Year without century as a zero-padded decimal number.	00, 01, ..., 99	-
%Y	Year with century as a decimal number.	0001, 0002, ..., 2013, 2014, ..., 9998, 9999	(2)
%H	Hour (24-hour clock) as a zero-padded decimal number.	00, 01, ..., 23	-
%I	Hour (12-hour clock) as a zero-padded decimal number.	01, 02, ..., 12	-
%p	Locale's equivalent of either AM or PM.	AM, PM (en_US); am, pm (de_DE)	(1), (3)
%M	Minute as a zero-padded decimal number.	00, 01, ..., 59	-
%S	Second as a zero-padded decimal number.	00, 01, ..., 59	(4)
%f	Microsecond as a decimal number, zero-padded on the left.	000000, 000001, ..., 999999	(5)
%z	UTC offset in the form +HHMM or -HHMM (empty string if the object is naive).	(empty), +0000, -0400, +1030	(6)
%Z	Time zone name (empty string if the object is naive).	(empty), UTC, EST, CST	-
%j	Day of the year as a zero-padded decimal number.	001, 002, ..., 366	-
%U	Week number of the year (Sunday as the first day of the week) as a zero padded decimal number. All days in a new year preceding the first Sunday are considered to be in week 0.	00, 01, ..., 53	(7)

Syntax	Description	Example	Notes
<code>%W</code>	Week number of the year (Monday as the first day of the week) as a decimal number. All days in a new year preceding the first Monday are considered to be in week 0.	00, 01, ..., 53	(7)
<code>%c</code>	Locale's appropriate date and time representation.	Tue Aug 16 21:30:00 1988 (en_US); Di 16 Aug 21:30:00 1988 (de_DE)	(1)
<code>%x</code>	Locale's appropriate date representation.	08/16/88 (None); 08/16/1988 (en_US); 16.08.1988 (de_DE)	(1)
<code>%X</code>	Locale's appropriate time representation.	21:30:00 (en_US); 21:30:00 (de_DE)	(1)
<code>%%</code>	A literal '%' character.	%	-

Notes:

1. Because the format depends on the current locale, care should be taken when making assumptions about the output value. Field orderings will vary (for example, "month/day/year" versus "day/month/year"), and the output may contain Unicode characters encoded using the locale's default encoding (for example, if the current locale is ja_JP, the default encoding could be any one of eucJP, SJIS, or utf-8; use `locale.getlocale()` to determine the current locale's encoding).
2. The `strptime()` method can parse years in the full [1, 9999] range, but years < 1000 must be zero-filled to 4-digit width.
Changed in version 3.2: In previous versions, `strptime()` method was restricted to years >= 1900.
Changed in version 3.3: In version 3.2, `strptime()` method was restricted to years >= 1000.
3. When used with the `strptime()` method, the `%p` directive only affects the output hour field if the `%I` directive is used to parse the hour.
4. Unlike the `time` module, the `datetime` module does not support leap seconds.
5. When used with the `strptime()` method, the `%f` directive accepts from one to six digits and zero pads on the right. `%f` is an extension to the set of format characters in the C standard (but implemented separately in `datetime` objects, and therefore always available).

6. For a naive object, the %z and %Z format codes are replaced by empty strings.

For an aware object:

%z

utcoffset() is transformed into a 5-character string of the form +HHMM or -HHMM, where HH is a 2-digit string giving the number of UTC offset hours, and MM is a 2-digit string giving the number of UTC offset minutes. For example, if utcoffset() returns timedelta(hours=-3, minutes=-30), %z is replaced with the string '-0330'.

%Z

If tzname() returns None, %Z is replaced by an empty string. Otherwise %Z is replaced by the returned value, which must be a string.

Changed in version 3.2: When the %z directive is provided to the strptime() method, an aware datetime object will be produced. The tzinfo of the result will be set to a timezone instance.

7. When used with the strptime() method, %U and %W are only used in calculations when the day of the week and the year are specified.

Copyright © 2001-2014 Python Software Foundation; All Rights Reserved

external_faces

The external_faces module extracts external faces from a 2D or 3D field for rendering. external_faces produces a mesh of only the external faces of each cell set of a data set. Because each cell set's external faces are created there may be faces that are seemingly internal (vs. external). This is especially true when external faces is used subsequent to a plume module on 3D (volumetric) input.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as faces.
- **Output Object** [Renderable]: Outputs to the viewer.

external_edges

The external_edges module produces a wireframe representation of of an unstructured cell data mesh. This is generally used to visualize the skeletal shape of the data domain while viewing output from other modules, such as plumes and surfaces, inside the unstructured mesh. external_edges produces a mesh of only the external edges which meet the edge angle criteria below for each cell set of a data set. Because each cell set's external faces are used there may be edges that are seemingly internal (vs. external). This is especially true when external edges is used subsequent to a plume module on 3D (volumetric) input.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).

- **Input Field** [Field] Accepts a data field from krig_3d or other similar modules.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field] Outputs the subsetting field as edges
- **Output Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the Z scaling and edge angle used to determine what edges should be displayed
- Data Selection: controls the type and specific data to be output or displayed

thin_fence

Thin_fence creates a fence diagram along a user defined (x, y) path. The fence cross-section has no thickness (because it is composed of areal elements such as triangles and quadrilaterals), but can be created in either true 3D model space or projected to 2D space.

It receives a 3D field (with volumetric elements) into its left input port and it receives lines or polylines (from draw_lines, polyline_spline, read_cad, isolines, read_vector_gis, or other sources) into its right input port. Its function is similar to buffer, however it actually creates a new grid and does not rely on any other modules (e.g. plume or plume_shell) to do the "cutting". Only the x and y coordinates of the input (poly)lines are used because thin_fence cuts a projected slice that is z invariant. Thin_fence recalculates when either input field is changed (and Run Automatically is on) or when the "Run Once" button is pressed.

If you select the option to "Straighten to 2D", thin_fence creates a straightened fence that is projected to a new 2D coordinate system of your choice. The choices are XZ or XY. For output to ESRI's ArcMAP, XY is required.

NOTE: The beginning of straightened (2D) fences is defined by the order of the points in the incoming line/polyline. This is done to provide the user with complete control over how the cross-section is created. However, if you are provided a CAD file and you do not know the order of the line points, you can export the CAD file using the write_lines module which provides a simple text file that will make it easy to see the order of the points.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a volumetric data field.
- **Input Line** [Field] Accepts a field with one or more line segments for the creation of the fence cross-section. Only the XY coordinates are used. Data is not used.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field
- **Output Object** [Renderable]: Outputs to the viewer.

slice

The slice module allows you to create a subset of your input which is of reduced dimensionality. This means that volumetric, surface and line inputs will result in surface, line and point outputs respectively. This is unlike cut which preserves dimensionality.

The slice module is used to slice through an input field using a slicing plane defined by one of four methods

1. A vertical plane defined by an X or **Easting** coordinate
2. A vertical plane defined by a Y or **Northing** coordinate
3. A **Horizontal** plane defined by a Z coordinate
4. An arbitrarily positioned **Rotatable** plane which requires:
 1. A 3D point through which the slicing plane passes. This point can be displayed using the *Reference Sphere* whose size, visibility and transparency can be controlled. Please note that the same slicing result can be achieved with an infinite number of 3D points, all of which would be on the same slicing plane.
 2. A **Dip** direction
 3. A **Strike** direction

Module Input [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field] Outputs the field
- **Output Object** [Renderable]: Outputs to the viewer.

isolines

The isolines module is used to produce lines of constant (iso)value on a 2D surface (such as a slice plane), or the external faces of a 3D surface, such as the external faces of a plume. The input data for isolines must be a surface (faces), it cannot be a volumetric data field. If the input is the faces of a 3D surface, then the isolines will actually be 3D in nature. Isolines can automatically place labels in the 2D or 3D isolines. By default isolines are on the surface (within it) and they have an elevated jitter level (1.0) to make them preferentially visible. However they can be offset to either side of the surface.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Contour Levels** [Contours]: Accepts an array of values representing values to place isolines

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with altered data min/max values
- **Output Contour Levels** [Contours]: Outputs an array of values representing values to be labeled in the legend.
- **Output Object** [Renderable]: Outputs to the viewer.

cut

The cut module allows you to create a subset of your input which is of the same dimensionality. This means that volumetric, surface, line and point inputs will have subsetting outputs of the same object type. This is unlike slice which decreases dimensionality.

The cut module is used to cut away part of the input field using a cutting plane defined by one of four methods

The cut module cuts through an input field using a slicing plane defined by one of four methods

1. A vertical plane defined by an X or **Easting** coordinate
2. A vertical plane defined by a Y or **Northing** coordinate
3. A **Horizontal** plane defined by a Z coordinate
4. An arbitrarily positioned **Rotatable** plane which requires:
 1. A 3D point through which the slicing plane passes. This point can be displayed using the *Reference Sphere* whose size, visibility and transparency can be controlled. Please note that the same slicing result can be achieved with an infinite number of 3D points, all of which would be on the same slicing plane.
 2. A **Dip** direction
 3. A **Strike** direction

The cutting plane essentially cuts the data field into two parts and sends only the part above or below the plane to the output ports (above and below are terms which are defined by the normal vector of the cutting plane). The output of cut is the subset of the model from the side of the cut plane specified.

Module Input [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Cut Field** [Field] Outputs the field with "cut" data to later use for subsetting
- **Output Field** [Field] Outputs the subsetting field
- **Output Object** [Renderable]: Outputs to the viewer.

plume

The plume module creates a (same dimensionality) subset of the input, regardless of dimensionality. What this means, in other words, is that plume can receive a field (blue port) model with cells which are points, lines, surfaces and/or volumes and its output will be a subset of the same type of cells.

This module should not normally be used when you desire a visualization of a 3D volumetric plume but rather when you wish to do subsequent operations such as analysis, slices, etc.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Isolevel** [Number] Accepts the subsetting level.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting field as a volume.
- **Status** [String / minor] Outputs a string containing a description of the operation being performed (e.g. TCE plume above 4.00 mg/kg)
- **Isolevel** [Number] Outputs the subsetting level.
- **Plume** [Renderable]: Outputs to the viewer.

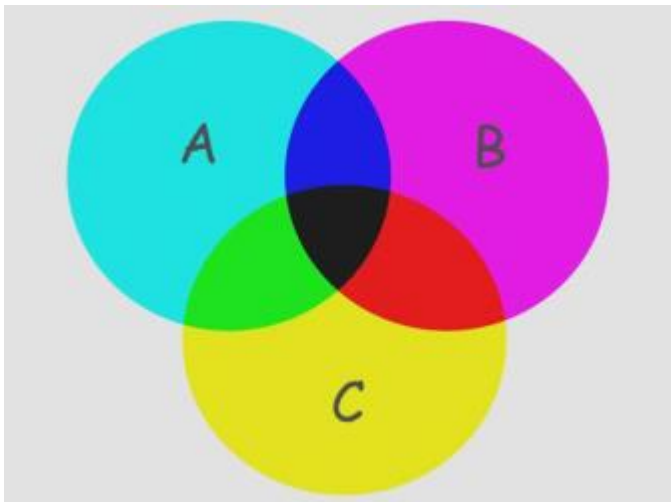
intersection

intersection is a powerful module that incorporates some of the characteristics of plume, yet allows for any number of volumetric sequential (serial) subsetting operations.

The functionality of the intersection module can be obtained by creating a network of serial plume modules. The number of analytes in the intersection is equal to the number of plume modules required.

The intersection of multiple analytes and threshold levels can be equated to the answer to the following question (example assumes three analytes A, B & C with respective subsetting levels of a, b and c):

"What is the volume within my model where A is above a, **AND** B is above b, **AND** C is above c?"



The figure above is a Boolean representation of 3 analyte plumes (A, B & C). The intersection of all three is the black center portion of the figure. Think of the image boundaries as the complete extents of your models (grid). The "A" plume is the circle colored cyan and includes the green, black and blue areas. The intersection of just A & C would be both the green and black portions.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field
- **Output Object** [Renderable]: Outputs to the viewer.

union

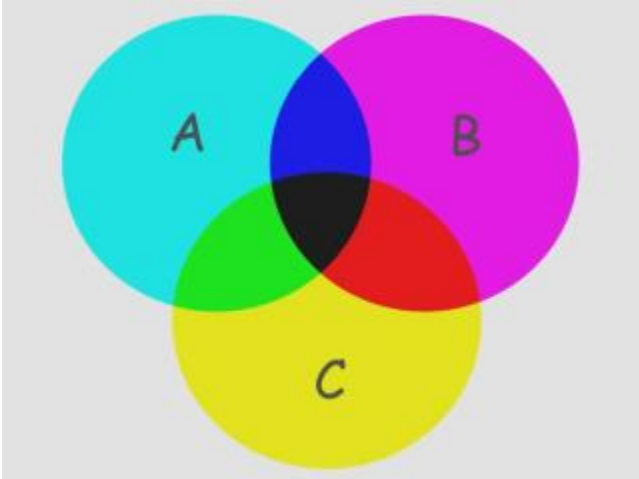
union is a powerful module that automatically performs for a large number of complex serial and parallel subsetting operations required to compute and visualize the union of multiple analytes and threshold levels. The functionality of the union module can be obtained by creating a network fragment composed of only plume modules. However as the number of analytes in the union increases, the number of plume modules increases very dramatically. The table below lists the number of plume modules required for several cases:

Number of Analytes	Number of plume Modules
2	3
3	6
4	10
5	15
6	21
7	28
n	$(n * (n+1)) / 2$

From the above table, it should be evident that as the number of analytes in the union increases, the computation time will increase dramatically. Even though union appears to be a single module, internally it grows more complex as the number of analytes increases.

The union of multiple analytes and threshold levels can be equated to the answer to the following question (example assumes three analytes A, B & C with respective subsetting levels of a, b and c):

"What is the volume within my model where A is above a, **OR** B is above b, **OR** C is above c?"



The figure above is a Boolean representation of 3 analyte plumes (A, B & C). The union of all three is the entire colored portion of the figure. Think of the image boundaries as the complete extents of your models (grid). The "A" plume is the circle colored cyan and includes the green, black and blue areas. The union of just A & C would be all colored regions EXCEPT the magenta portion of B.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field
- **Output Object** [Renderable]: Outputs to the viewer.

plume_cell

The plume_cell module creates a subset of the input grid with the same dimensionality. What this means, in other words, is that plume can receive a field (blue port) model with cells which are points, lines, surfaces and/or volumes and its output will be a subset of the same type of cells.

plume_cell is different from plume in that it outputs entire cells making its output *lego-like*.

It uses a mathematical expression allowing you to do complex subsetting calculations on coordinates and MULTIPLE data components with a single module, which can dramatically simplify your network and reduce memory usage. It has 2 floating point variables (N1,N2) which are setup with ports so they can be easily animated.

Subset By: You can specify whether the subsetting is based on either Nodal data or Cell data.

Expression Cells to Include: Is a Python expression where you can specify whether the subsetting of cells requires all nodes to match the criteria for a cell to be included or if ANY nodes match, then the cell will be included. The second option includes more cells.

Operators:

- == Equal to
- < Less than

- > Greater Than
- <= Less than or Equal to
- >= Greater Than or Equal to
- or
- and
- in (as in list)

Example Expressions:

- If Nodal data is selected:
 - $D0 \geq N1$ All nodes with the first analyte greater than or equal to N1 will be used for inclusion determination.
 - $(D0 < N1) \text{ or } (D1 < N2)$ All nodes with the first analyte less than or equal to N1 OR the second analyte less than or equal to N2 will be used for inclusion determination.
- If Cell data is selected:
 - $D1 \text{ in } [0, 2]$ where D1 is Layer will give you the uppermost and third layers.
 - $D1 \text{ in } [1]$ where D1 is Layer will give you the middle layer.
 - $D1 == 0$ where D1 is Layer will give you the uppermost layer
 - $D1 \geq 1$ where D1 is Layer will give you all but the uppermost layer

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as a volume.
- **Status** [String / minor] Outputs a string containing a description of the operation being performed (e.g. TCE plume above 4.00 mg/kg)
- **Isolevel** [Number] Outputs the subsetting level.
- **Plume** [Renderable]: Outputs to the viewer.

footprint

The footprint module is used to create the 2D footprint of a plume_shell. It creates a surface at the specified Z Position with an x-y extent that matches the 3D input. The footprint output does not contain data, but data can be mapped onto it with external kriging.

NOTE: Do not use adaptive gridding when creating the 3D grid to be footprinted and mapping the maximum values with krig_2d (as in the example shown below).

Footprint will produce the correct area, but krig_2d will map anomalous results when used with krig_3d's adaptive gridding.

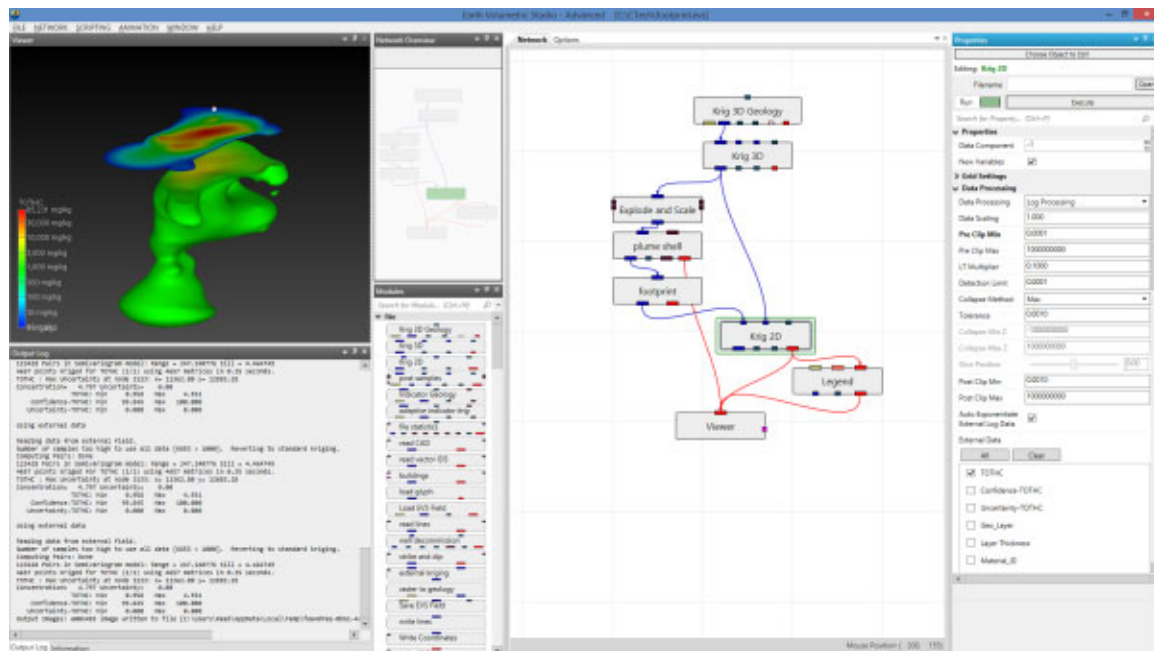
Module Input Ports

- **Input Field** [Field] Accepts a data field.

Module Output Ports

- **Output Field** [Field] Outputs the subsetted field.
- **Output Object** [Renderable]: Outputs to the viewer.

NOTE: Creating a 2D footprint with the maximum data within the plume volume mapped to each x-y location requires the external data and external gridding options in krig_2d. A typical network and output is shown below.

**slope_aspect_splitter**

The `slope_aspect_splitter` module will split an input field into two output fields based upon the slope and/or aspect of the external face of the cell and the subset expression used. The input field is split into two fields one for which all cells orientations are true for the subset expression, and another field for which all cells orientations are false for the subset expression.

All data from the original input is preserved in the output.

Flat Surface Aspect: If you have a flat surface then a realistic aspect can not be generated. This field lets you set the value for those sells.

1) **To output all upward facing surfaces:** use the default subset expression of **SLOPE < 89.9**. If your object was a perfect sphere, this would give you most of the upper hemisphere. Since the equator would be at slope of 90 degrees and the bottom would >90 degrees.

(Notice there is potential for rounding errors use 89.9 instead of 90)

Note: If your ground surface is perfectly flat and you wanted only it, you could use $SLOPE < 0.01$, however in the real world where topography exists, it can be difficult

if not impossible to extract the ground surface and not get some other bits of surfaces that also meet your criteria.

2) General expression (assuming a standard cubic building)

A) $\text{SLOPE} > 0.01$ (Removes the top of the building)

B) $\text{SLOPE} > 0.01$ and $\text{SLOPE} < 179.9$ (Removes the top and bottom of the building)

3) Since ASPECT is a variable it must be defined for each cell. In cells with a slope of 0 or 180 there would be no aspect without our defining it with the *flat surface aspect* field

4) Units are always degrees. You could change them to radians if you want inside the expression. ($\text{SLOPE} * \text{PI}/180$)

Module Input [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Input Field** [Field] Accepts a data field.
- **Number Variable 1** [Number] Accepts the first numeric value for the slope or aspect expression
- **Number Variable 2** [Number] Accepts the second numeric value for the slope or aspect expression

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output True Field** [Field] Outputs the field which matches the subsetting expression
- **Output False Field** [Field] Outputs the opposite of the true field

crop_and_downsize

The crop_and_downsize module is used to subset an image, or structured 1D, 2D or 3D mesh (an EVS "field" data type with implicit connectivity). Similar to cropping and resizing a photograph, crop_and_downsize sets ranges of cells in the I, J and K directions which create a subset of the data. When used on an image (which only has two dimensions), crop removes pixels along any of the four edges of the image. Additionally, crop_and_downsize reduces the resolution of the image or grid by an integer downsize value. If the resolution divided by this factor yields a remainder, these cells are dropped.

crop_and_downsize refers to I, J, and K dimensions instead of x-y-z. This is done because grids are not required to be parallel to the coordinate axes, nor must the grid rows, columns and layers correspond to x, y, or z. You may have to experiment with this module to determine which coordinate axes or model faces are being cropped or downsize.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting field
- **Output Object** [Renderable]: Outputs to the viewer.

select_cells

select_cells provides the ability to select individual geologic layers for output. If connected to explode_and_scale multiple select_cells modules will allow selection of specific geologic layers for downstream processing. One example would be to texture map the top layer with an aerial photo after one select_cells and to color the other layers by geologic layer with a parallel select_cells path. This can be accomplished by multiple explode_and_scale modules, but that would take much more memory.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field
- **Output Object** [Renderable]: Outputs to the viewer.

orthoslice

The orthoslice module is similar to the slice module, except limited to only displaying slice positions north-south (vertical), east-west (vertical) and horizontal. orthoslice subsets a structured field by extracting one slice plane and can only be orthogonal to the X, Y, or Z axis. Although less flexible in terms of capability, orthoslice is computationally more efficient.

The **axis** selector chooses which axis (I, J, K) the orthoslice is perpendicular to. The default is I. If the field is 1D or 2D, three values are still displayed. Select the values meaningful for the input data.

The **plane** slider selects which plane to extract from the input. This is similar to the position slider in slice but, since the input is a field, the selection is based on the nodal dimensions of the axis of interest. Therefore, the range is 0 to the maximum nodal dimension of the axis. For example, for an orthoslice through a grid with dimension 20 x 20 x 10, the range in the x and y directions would be 0 to 20.

edges

The edges module is similar to the External_Edges module in that it produces a wireframe representation of the nodal data making up an unstructured cell data mesh. There is however, no adjustment of edge angle and therefore only allows viewing of all grid boundaries (internal AND external) of the input mesh. The edges module is useful in that it is able to render lines around adaptive gridding locations whereas external_edges does NOT render lines around this portion of the grid.

bounds

bounds generates lines and/or surfaces that indicate the bounding box of a 3D structured field. This is useful when you need to see the shape of an object and the structure of its mesh. This module is similar to external_edges (set to edge angle = 60), except, bounds allows for placing faces on the bounds of a model.

bounds has one input ports. Data passed to the first port (closest to the left) must contain any type of structured mesh (a grid definable with IJK resolution and no separable layers). Node_Data can be present, but is only used if you switch on Data.

area_cut

Area_cut receives any 3D field into its left input port and it receives triangulated polygons (from triangulate_polygon, or other sources) into its right input port. Its

function is similar to buffer or shape_cut. It adds a data component to the input 3D field and using plume_shell, you can cut structures inside or outside of the input polygons. Only the x and y coordinates of the polygons are used because area_cut cuts a projected slice that is z invariant. Area_cut recalculates when either input field is changed or the "Accept" button is pressed.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Area** [Field] Accepts a field with the area to include/exclude

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with area data to allow subsetting

The first thing to know, is that area_cut does not cut.

It provides data with which you can then subset using other modules like plume or intersection.

Without the subsetting modules AFTER area_cut, you would see no affect of having area_cut in your application other than it adds a new nodal data component called area_cut (or whatever you've renamed your module to be).

area_cut needs a SURFACE as its input. It does not care where that surface comes from and it certainly does not need to be from a DWG file. The surface can be complex, meaning that it can have holes in it, or it can be separate disjoint pieces of surface(s).

If you're starting with lines, it is required that the lines form a closed polyline. It is not enough that the lines appear to be a closed path, they must be truly closed, with each successive segment precisely connected to the last and next. CAD files are often poorly drawn and are not closed (though they can be well drawn and properly closed also).

Our draw_lines module can certainly be used to create a Closed polyline, but you must make sure to turn on the "Closed" toggle for each line segment to ensure it is closed.

Once you have one or more closed polylines, you will need to pass those through triangulate_polylines modules to create a TIN surface from the closed polylines. You should confirm (by connecting it to the viewer) that you are getting the correct surface before proceeding to area_cut. If triangulate_polylines will not run, your lines are not closed.

Once you have your surface(s) and you pass that to the right input port of area_cut, the output of area_cut is data with which you can subset your original model. The data is zero (0.0) at the boundaries of your surface: is less than zero (negative) inside the surface; and is greater than zero (positive) outside of the surface. To get everything inside, you need to choose "Below Level" in the subsetting modules rather than the Default "Above Level".

surf_cut

surf_cut receives any 3D field into its left input port and it receives a surface (from scat_to_tin, geologic_surface, slice, etc.) into its right input port. Its function is similar to shape_cut. It adds a data component to the input 3D field referencing the cutting surface. With this new data component you can use a subsetting module like plume to pass either side of the 3D field as defined by the cutting surface, thereby allowing cutting of structures along any surface. The surface can originate from a TIN

surface, a slice plane or a geologic surface. The cutting surface can be multi-valued in Z, which means the surface can have instances where there are more one z value for a single x, y coordinate. This might occur with a wavy fault surface that is nearly vertical, or a fault surface with recumbent folds.

surf_cut recalculates when either input field is changed or the "Accept" button is pressed.

The general approach with surf_cut is:

Create a cutting surface representing either a fault plane, a scouring surface (unconformity), or an excavation.

Create a 3D model of the object you wish to cut.

Pass the 3D model into the left port of surf_cut, and the cutting surface to the right port of surf_cut and hit accept.

Module Input [Ports](#)

- **Input 3D Field** [Field] Accepts a data field.
- **Input Surface** [Field] Accepts a field with the surface to cut the input volume/surface

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with distance to surface data to allow subsetting

surf cut example images

shape_cut

shape_cut receives any 3D field into its input port and outputs the same field with an additional data component. Using plume_shell, you can cut structures with either a cylinder or rotated rectangle. The cutting action is z invariant (like a cookie cutter). Depending on the resolution of the input field, rectangles may not have sharp corners. With rectilinear fields (and non-rotated rectangles), the threshold module can replace plume_shell to produce sharp corners (by removing whole cells). plume can be used to output 3D fields for additional filtering or mapping.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with data to allow subsetting.

buffer

buffer receives any 3D field into its left input port and it receives polylines (from read_lines, read_vector_gis, read_cad, isolines, or other sources) into its right input port. Its function is similar to shape_cut. It adds a data component to the input 3D field and using plume_shell, you can cut structures along the path of the input polylines. Only the x and y coordinates of the polylines are used because buffer cuts a projected slice that is z invariant. buffer recalculates when either input field is changed or the "Execute" button is pressed. "Thick Fences" can be produced with the output of this module.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Fence Line** [Field] Accepts a field with the line(s) to cut the input volume/surface

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with distance to path(s) data to allow subsetting

tunnel_cut

The tunnel_cut module is similar to the [surf_cut](#) module in that it receives any 3D field into its left input port, BUT instead of a surface, it receives a line (along the trajectory of a tunnel, boring or mineshaft) into its right input port. The tunnel_cut module then cuts a cylinder, of user defined radius, along the line trajectory. The algorithm is identical in concept to surf_cut in that it adds a data component to the input 3D field referencing the distance from the line (trajectory). With this new data component you can use a subsetting module like plume_volume to pass either portion of the 3D field (inside the cylinder or outside the cylinder), thereby allowing cutting tunnels along any trajectory. The trajectory line can originate from a DXF file or a NetCDF file.

The general approach is to subset the tunnel_cut data component with either constant_shellor plume_volume. The choice of 1.0 for the subsetting level will result in cutting AT the user radius, while less than 1.0 is inside the cylinder wall and greater than 1.0 is outside the cylinder wall.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Tunnel Line** [Field] Accepts a field with the surface to cut the input volume/surface

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with distance to tunnel line data to allow subsetting

overburden

The overburden module computes the complete volume required to excavate a plume or ore body given the pit wall slope (measured from vertical) and the excavation digging accuracy (we refer to as buffer size).

overburden receives any 3D field into its input port and outputs the same field with an additional data component. Its function is similar to shape_cut, but instead involves computing a new data component based on the nodal values in the 3D field and two user defined parameter values called Wall Slope and buffer size (addressing excavation accuracy). The data component is subset according to a concentration input (based on the subsetting level you want excavated). For example, once overburden has been run for GOLD at a 45 degree pit wall slope, the user would select 45-deg:overburden_GOLD and subset all data below 1 ppm to render a 45 degree slope pit which would excavate everything higher than 1 ppm concentration. A volumetrics calculation could be made on these criteria which would encompass the excavation and the ore body above 1 ppm.

NOTE: overburden must be placed before any scaling modules (such as `explode_and_scale`) to ensure an accurate slope angle during computations and subsequent visualizations.

Note on angles: Angles are defined from the vertical and are specified in degrees.

- A vertical wall pit is created with an angle of Zero (0.0) degrees
- A 2:1 pitch slope from horizontal would be an angle whose arctangent = 2.0. This is 63.4 degree from horizontal and therefore you would enter 26.6 degrees (from vertical)

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the enhanced field with overburden data

Create Buffer Around Plume - This toggle determines if the overburden computations are rigorous and determine the buffer on all side of the plume (ore body). If this is off, the module runs much quicker.

Buffer Size - An accuracy level resulting in the amount of excavation outside the subsetting level of interest. For example, a type-in of 10.0 would result in 10 feet of over-excavation from the subsetting level of interest.

Overburden creates a data component name that includes the wall slope, module name (including #1 or #2 if there are more than one copy in your application), and original data component (analyte) name. (i.e. 30-deg:overburden#1 of Benzene)

The overburden data component may be subset by modules such as plume, isosurface, plume_shell, etc.

mask_geology

mask_geology receives geologic input into its left input port and an optional input masking surface into its right port.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Area** [Field] Accepts a field defining a surface of the area for masking

Module Output [Ports](#)

- **Output Field** [Field] Outputs the processed field.

node_computation

The node_computation module is used to perform mathematical operations on nodal data fields **and** coordinates. Data values can be used to affect coordinates (x, y, or z) and coordinates can be used to affect data values.

Up to two fields can be input to node_computation. Mathematical expressions can involve one or both of the input fields. **Fields must be identical grids. This means they must have the same number of nodes and cells, otherwise the results will not make sense.**

Nodal data input to each of the ports is normally scalar, however if a vector data component is used, the values in the expression are automatically the magnitude of the vector (which is a scalar). If you want a particular component of a vector, insert an `extract_scalar` module before connecting a vector data component to `node_computation`. The output is always a scalar. If a data field contains more than one data component, you may select from any of them.

Module Input [Ports](#)

- **Input Field 1** [Field] Accepts a data field.
- **Input Field 2** [Field / minor] Accepts a data field.
- **Input Value N1** [Number / minor] Accepts a number to be used in the field computations.
- **Input Value N2** [Number / minor] Accepts a number to be used in the field computations.
- **Input Value N3** [Number / minor] Accepts a number to be used in the field computations.
- **Input Value N4** [Number / minor] Accepts a number to be used in the field computations.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as faces.
- **Output Value N1** [Number / minor] Outputs a number used in the field computations.
- **Output Value N2** [Number / minor] Outputs a number used in the field computations.
- **Output Value N3** [Number / minor] Outputs a number used in the field computations.
- **Output Value N4** [Number / minor] Outputs a number used in the field computations.
- **Output Object** [Renderable]: Outputs to the viewer.

Module Parameters

- **Data Definitions:** You can have more than one new data component computed from each pass of `node_computation`. By default there is only Data0.
 - Add/Remove buttons allow you to add or remove Data Definitions
- **Name:** The data component name (e.g. Total Hydrocarbons)
- **Units :** The units of the data component (e.g. mg/kg)
- **Log Process:** When your input data is log processed, the values within `node_computation` will always be exponentiated.
 - In other words, even when your data is log processed, you will always see actual (not log) values.
 - This toggle should be ON whenever you are dealing with Log data.
 - If you want to perform math operations on the "Log" data, you must take the log of the A_n^* or B_n^* values within `node_computation`

- If you do take the log of those values, you should always exponentiate the end results before exiting node_computation.

Each nodal data component from Input Field 1 is assigned as a variable to be used in the script. For example:

- An0 : First input data component
- An1 : Second input data component
- An2 : Third input data component
- An^N : Nth input data component

The min and max of these components are also added as variables :

- Min_An0 : Minimum of An0 data
- Max_An0 : Maximum of An0 data
- Min_An* : Minimum of An* data

For Input Field 2 the variable names change to:

- Bn0 : First input data component
- Bn1 : Second input data component
- Bn2 : Third input data component
- Bn^N : Nth input data component

An interesting and simple example of using node_computation can be found [here](#).

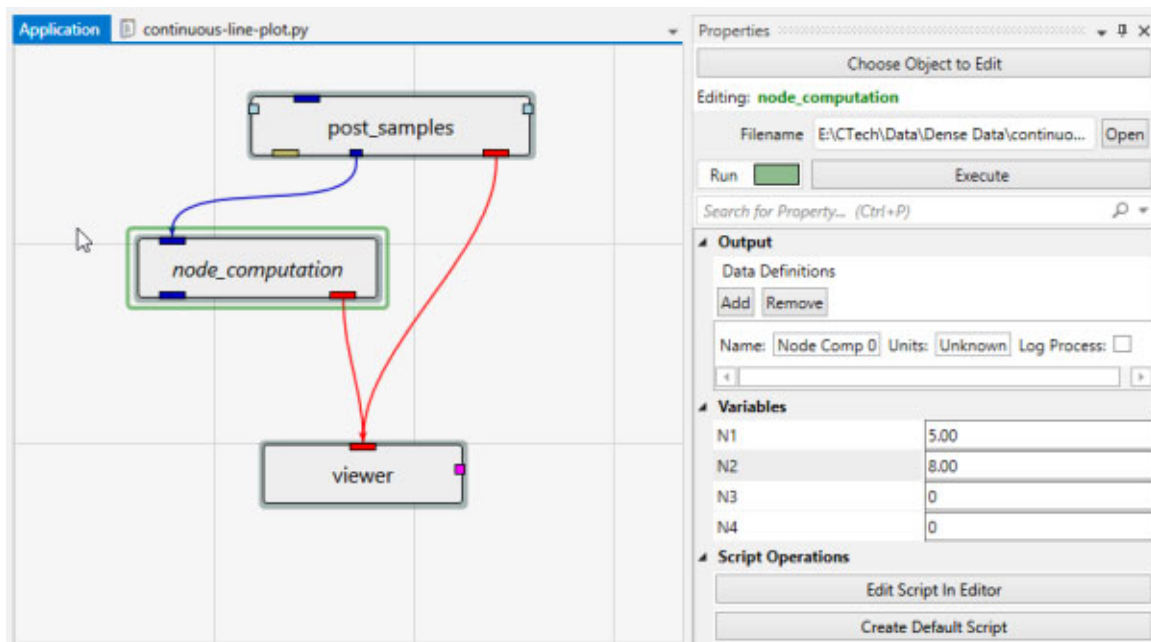
The equation(s) used to modify data and/or coordinates must be input as part of a Python Script. The module will generate a default script and by modifying only one line (for the X coordinate)we get:


```

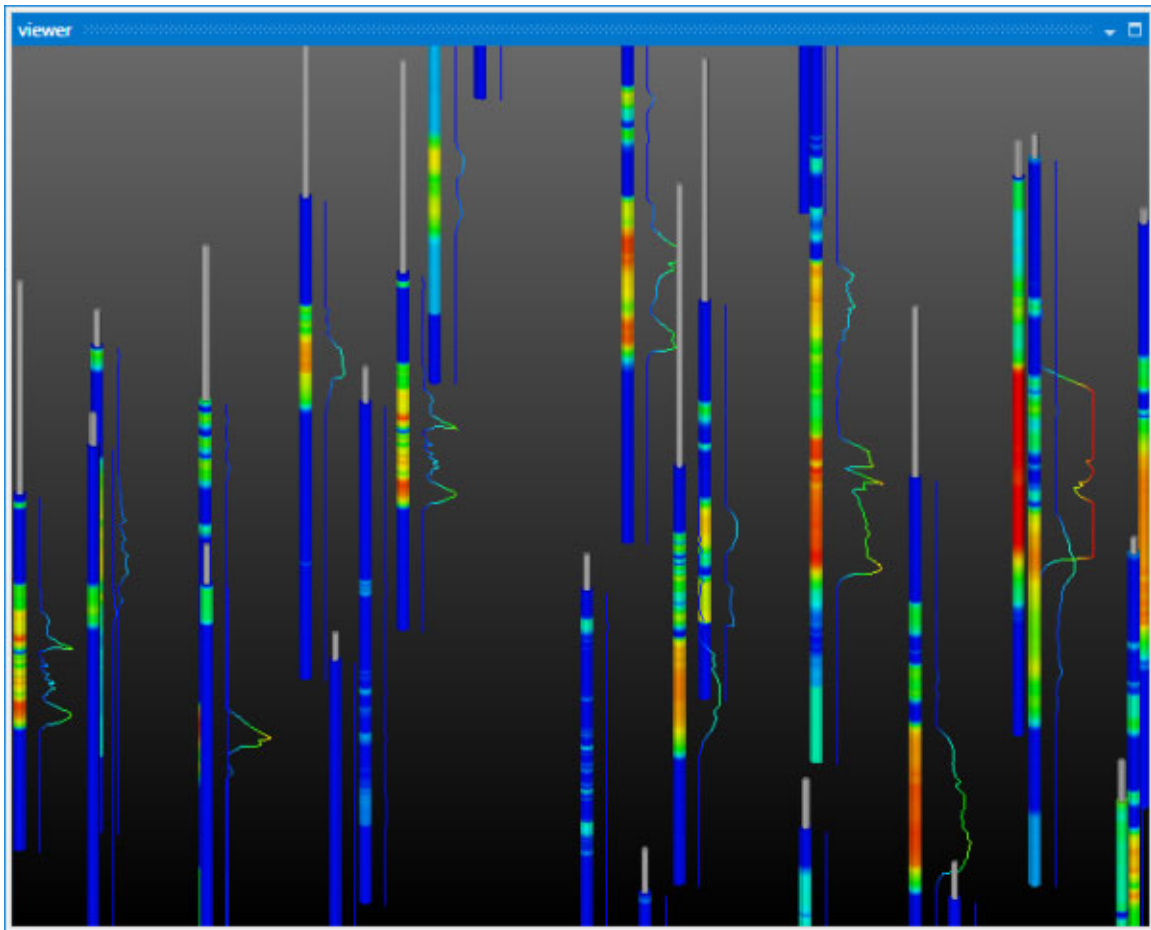
Application  continuous-line-plot.py X
1  from math import *
2
3  # Outputs to generate
4  # X : New X coordinate
5  # Y : New Y coordinate
6  # Z : New Z coordinate
7  X = Ax + (An0-Min_An0)*N1+N2
8  Y = Ay
9  Z = Az
10
11 # Data0 : New data for output data 1
12 Data0 = An0
13
14
15 # Available variables:
16 # N1 : User defined floating point number
17 # N2 : User defined floating point number
18 # N3 : User defined floating point number
19 # N4 : User defined floating point number
20 # An0 : First input data component
21 # An1 : Second input data component
22 # An2 : Third input data component
23 # An* : Nth input data component
24 # Min_An0 : Minimum of An0 data
25 # Max_An0 : Maximum of An0 data
26 # Min_An* : Minimum of An* data
27

```

which with the following application:



Gives us the ability to view densely sampled data as line plots beside each boring



combine_components

The combine_components module is used to create a new set of nodal data components by selecting components from up to six separate input data fields. The mesh (x-y-z coordinates) from the first input field, will be the mesh in the output. The input fields should have the same scale and origin, and number of nodes in order for the output data to have any meaning. This module is useful for combining data contained in multiple field ports or files, or from different Kriging modules.

Module Input [Ports](#)

- **Model Field [Field]** Accepts a field with data whose grid will be exported.
- **Input Field 1 [Field]** Accepts a data field.
- **Input Field 2 [Field]** Accepts a data field.
- **Input Field 3 [Field]** Accepts a data field.
- **Input Field 4 [Field]** Accepts a data field.
- **Input Field 5 [Field]** Accepts a data field.

Module Output [Ports](#)

- **Output Field [Field]** Outputs the field with selected data
- **Output Object [Renderable]:** Outputs to the viewer.

interp_data

The Interp data module interpolates nodal data from a 3D or 2D field to either a 2D mesh or 1D line. Typical uses of this module are mapping of nodal data from a 3D mesh onto a geologic surface or a 2D fence section. In these applications the 2D surface(s) simply provide the new geometry (mesh) onto which the adjacent nodal values are interpolated. The primary requirement is that the nodal data be equal or higher dimensionality than the mesh to be interpolated onto. For instance, if the user has a 2D surface with nodal data (perhaps z values), then a 1D line may be input and the nearest nodal values from the 2D surface will be interpolated onto it.

Module Input [Ports](#)

- **Input Data Field** [Field] Accepts a data field.
- **Input Destination Field** [Field] Accepts a field onto which the data will be interpolated

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field Destination Field with new data
- **Output Object** [Renderable]: Outputs to the viewer.

thickness

The thickness module allows you to compute the thickness of complex plumes or cell sets such as indicator_geology's materials.

Module Input [Ports](#)

- **Input Field** The field to map thickness data onto
- **Input Volume** The (volumetric) field to determine thickness data from

Module Output [Ports](#)

- **Output Field** The surface (or 3D object) with mapped thickness data

Important Features and Considerations

The right input port must have a 3D field as input.

- There is no concept of thickness associated with 2D or 3D surfaces
- Volumetric inputs can be plume_shell or intersection_shell objects which are hollow.
 - Thickness will be determined based upon the apparent thickness of the plume elements.
 - When 3D Shells are input, they must be closed objects.

Determining thickness of arbitrary volumetric objects is a very computationally intensive operation. You can use this module to compute thickness in two primary ways:

- Compute the thickness distribution of a 3D object and project that onto a 2D surface (generally at the ground surface)

- A surface (such as from geologic surface) would connect to the first (left) input port
- The volumetric object connects to the second (right) input port
- Compute the thickness distribution of a 3D object and project that onto the same object
 - The volumetric object connects to the first (left) input port
 - The same volumetric object connects to the second (right) input port

Note: *In all cases run times can be long. Coarser grids and the first option will run faster, but the complexity and resolution of the volumetric object will be the major factor in the computation time.*

data_translate

The data_translate module accepts **nearly** any mesh and translates the grid in x, y, or z based upon either a nodal or cell data component or a constant.

The interface enables changing the Scale Factor for z translates to accommodate an overall z exaggeration in your applications. This module is most useful when used with the read_vector_gis module to properly place polygonal shapefile cells at the proper elevation.

Warning: The scale factor is always applied. If translating along any axis other than z, it is unlikely that you want to use the Z Exaggeration factor used elsewhere in your application.

- When translating by a Constant, the amount is affected by the Z Scale Factor.
- When translating by Cell Data, a radio box appears to allow specification of the cell data component
- When translating by Node Data, a radio box appears to allow specification of the nodal data component

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a data field from krig_3d or other similar modules.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field] Outputs the subsetted field
- **Scale Link**
- **Output Object** [Renderable]: Outputs to the viewer

load_evs_field

load_evs_field reads a dataset from any of six different EVS compatible file formats (as created by [save_evs_field](#)) into an EVS field, including the new EVS field formats:

- .eff ASCII format, best if you want to be able to open the file in an editor or print it

- .efz GNU Zip compressed ASCII, same as .eff but in a zip archive
- .efb binary compressed format, the smallest & fastest format due to its binary form

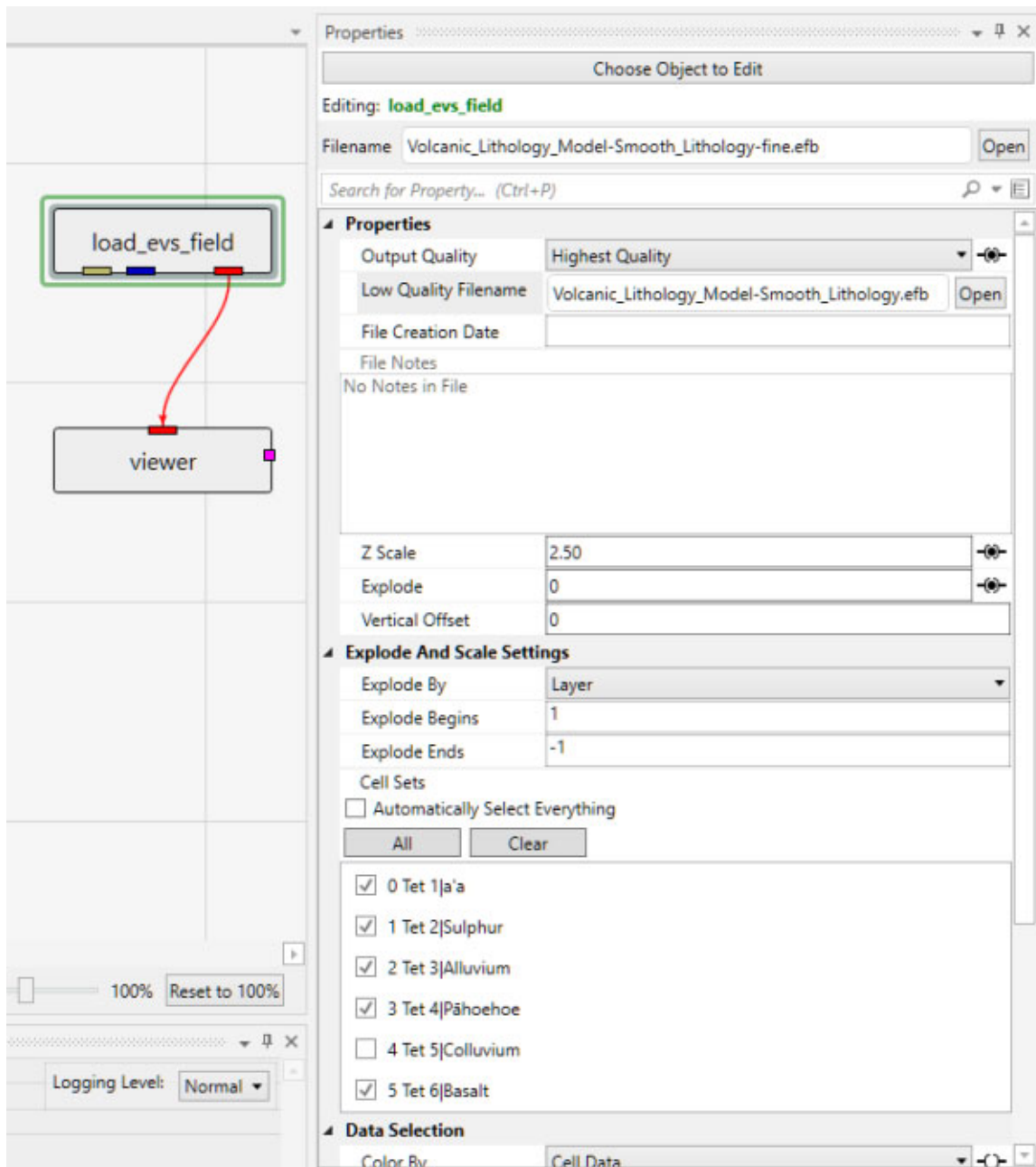
The EVS Field Formats *.eff;*.efz;and *.efb support **all** types of EVS field output including:

1. Uniform fields
2. Geology (from krig_3d_geology)
3. Structured fields (including legacy .FLD format)
4. Unstructured Cell Data (including legacy UCD format) general grids with nodal and/or cell data
5. Special fields containing spheres (which are points with radii)
6. Special fields containing color data (such as from Read_DXF)

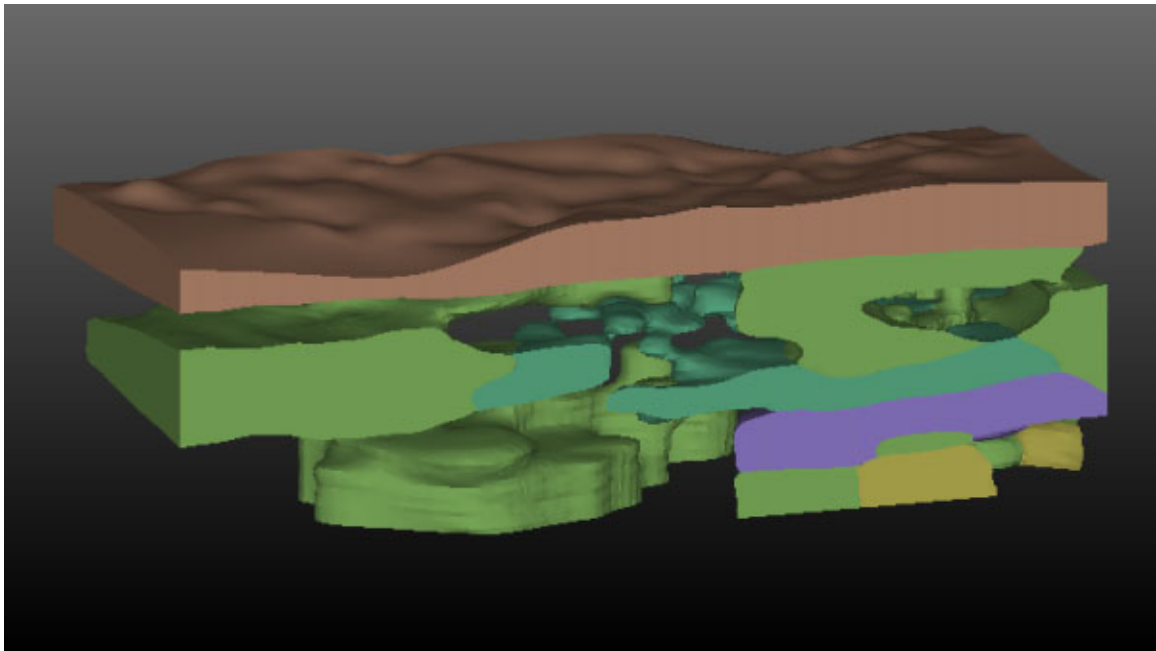
Note: Because the .eff, .efz and .efb formats better handle all types of EVS output, these three formats are recommended for use over UCD, netCDF or Field.

For a description of the [.EFF file formats click here](#).

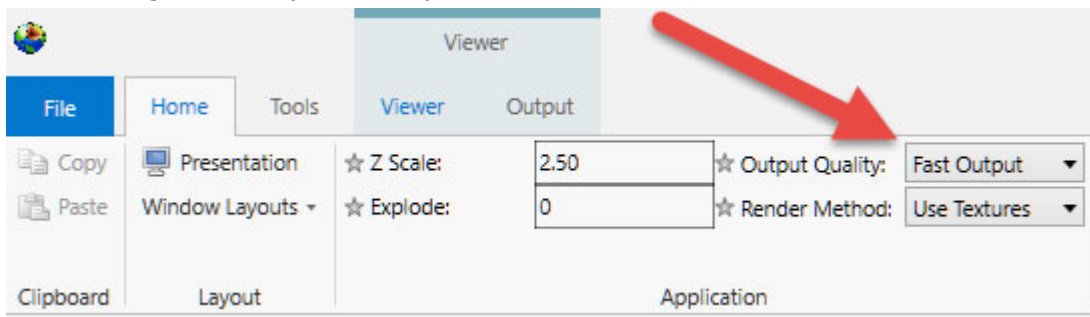
Output Quality: An important feature of load_evs_field is the ability to specify two separate files which correspond to High Quality (e.g. fine grids) and Low Quality (e.g. coarse grids a.k.a. fast). The example application below demonstrates this:



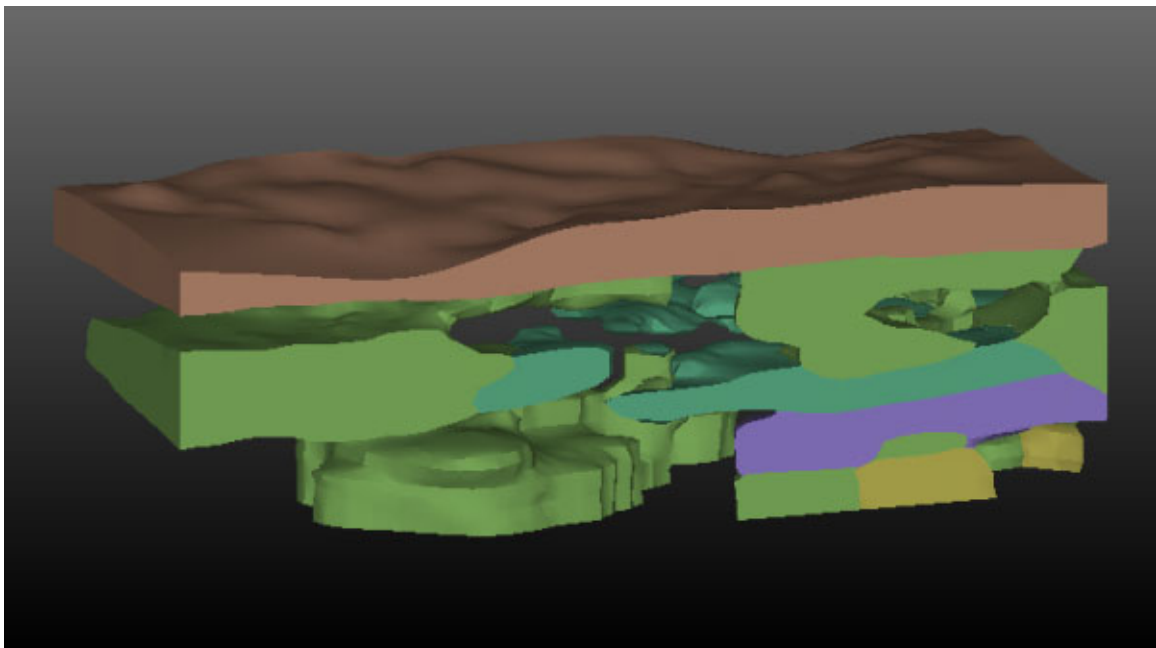
You can see that load_evs_field is specifying two different EFB files. The *Output Quality* is set to *Highest Quality* and is Linked (black circle). The viewer shows:



If we change the Output Quality on the Home Tab



It changes the setting in load_evs_field and the viewer changes to show:



Though you "can" change the Output Quality in load_evs_field, it is best to change it on the Home Tab to make sure that all load_evs_field modules in your application will have the same setting. This is not relevant to this simple application, but if we were using a cutting surface (saved as fine and coarse EFBs) and doing surf_cut operations on a very large grid, this synchronization would be important.

load_evs_field effectively has explode_and_scale and an external_faces module built in. This allows the module to perform:

- Z Scaling
- Exploding
- Nodal or Cell data selection
- Selection of cell_sets

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output** [Field] Outputs the saved field.
- **File Notes** [String / minor] Outputs a string to document the settings used to create the field.
- **Output Object** [Renderable]: Outputs to the viewer.

EVS Field File Formats and Examples

EVS Field file formats supplant the need for UCD, netCDF, Field (.fld), EVS_Geology by incorporating all of their functionality and more in a new file format with three mode options.

- 1) .eff ASCII format, best if you want to be able to open the file in an editor or print it
- 2) .efz GNU Zip compressed ASCII, same as .eff but in a zip archive
- 3) .efb binary compressed format, the smallest & fastest format due to its binary form

Here are the tags available in an EVS field file, in the appropriate order. Note that **no** file will contain ALL these tags, as some are specific to the type of field (based on definition). The binary file format is undocumented and exclusively used by C Tech's save_evs_field module.

If the file is written compressed, the .efz file (and any split, extra data files) will all be compressed. The compression algorithm is compatible with the free gzip/gunzip programs or WinZip, so the user can uncompress a .efz file and get an .eff file at will. The .efb file is also compressed (hence its very small size), but uncompressing this file will not make it human-readable.

EVS Field Files

EVS Field Files consist of file tags that delineate the various sections of the file(s) and data (coordinates, nodal and/or cell data, and connectivity). The file tags are discussed below followed by portions of a few example files.

FILE TAGS:

The file tags for the ASCII file formats (shown in ***Bold Italics***) are discussed below with a representative example. They are given in the appropriate order. If you need assistance creating software to write these file formats, please contact support@ctech.com.

DATE_CREATED(optional) 7/16/2004 1:57:55 PM

The creation date of the file.

EVS_FIELD_FILE_NOTES_START (optional)

Insert your Field file notes here.

EVS_FIELD_FILE_NOTES_END

This is the file description block. These notes are used to describe the contents of the Field file. The entire block is optional, however if you wish to use notes then both the starting and end tag are required.

DEFINITION Mesh+Node_Data

This is the type of field we are creating. Typically options are:

- 1) Mesh+Node_Data
- 2) Mesh+Cell_Data
- 3) Mesh+Node_Data+Cell_Data
- 4) Mesh_Struct+Node_Data (Geology)
- 5) Mesh_Unif+Node_Data (Uniform field)

NSPACE 3

nspace of the output field. Typically 3, but 2 in the case of geology or an image

NNODES 66355

Number of nodes. Not used for Mesh_Struct or Mesh_Unif

NDIM 2

Number of dimensions in a Mesh_Struct or Mesh_Unif

DIMS 41 41

The dimensions for a mesh_struct or uniform field

POINTS 11061.528999 12692.304504 -44.049999 11611.330994
13098.105469 11.500000

The lower left and upper right corner of a uniform field (Mesh_Unif only)

COORD_UNITS "ft"

Coordinate Units

NUM_NODE_DATA 7

Number of nodal data components

NUM_CELL_DATA 1

Number of cell data components

NCELL_SETS 5

Number of cell sets

NODES FILE "test_split.xyz" ROW 1 X 1 Y 2 Z 3

Nodes section is starting. If it says "NODES IN_FILE", the nodes follow (x/y/z) on the next nnodes rows, otherwise, the line will say FILE "filename" ROW 1 X 1 Y 2 Z 3, which is the file to get the coordinates, the row to start at (1 is first line of file), and the columns containing your X, Y, and Z values

NODE_DATA_DEF 0 "TOTHC" "log_ppm" MINMAX -3 4.592 FILE

"test_split.nd" ROW 1 COLS 1

NODE_DATA_DEF specifies the definition of a nodal data component. The second word is the data component number, the third is the name, the 4th is the units, then it will either say IN_FILE (which means that it will start after a NODE_DATA_START tag) or the file information. Other options are:

- 1) MINMAX - two numbers follow which are the data minimum and maximum. This behaves much like the set_min_max module.
- 2) If this is vector data, there will be a VECLLEN 3 tag in there, and COLS will need to have 3 numbers following it (for each component of the vector)
- 3) NODE_DATA_START. All the node data components that are specified IN_FILE are listed in order after this tag.

CELL_SET_DEF 0 8120 Hex "Fill" MINMAX 1 14 FILE "test_split.conn" ROW 1

Definition of a cell set. 2nd word is cell set number, 3rd is number of cells, 4th is type, 5th is the name, then its either IN_FILE (which means they will be listed in order by cell set), or the FILE "filename" section and a row to begin reading from. Other options are:

- 1) MINMAX - two numbers follow which are the data minimum and maximum. This behaves much like the cell_set_min_max module.
- 2) CELL_START. Start of all the cell set definitions that are specified IN_FILE.

CELL_DATA_DEF 0 "Indicator" "Discreet Unit" FILE "test_split.cd" ROW 1 COLS 1

Definition of cell data. Same options as NODE_DATA_DEF

CELL_DATA_START

Start of all cell data that is specified as IN_FILE

LAYER_NAMES "Top" "Fill" "Silt" "Clay" "Gravel" "Sand"

Allows you to specify the names associated with surfaces (layers)

MATERIAL_MAPPING "1|Silt" "2|Fill" "3|Clay" "4|Sand" "5|Gravel"

Allows you to specify the Material_ID and the associated material names. Note that each number/name pair is in quotes, with the name separated from the number by the pipe "|" symbol.

END

Marks the end of the data section of the file. (Allows us to put a password on .eff files)

EVS Field File Examples:

Because EVS Field Files can contain so many different types of grids, it is beyond the scope of our help system to include every variant.

krig_3d - EFF file representing a uniform field: The file below is an abbreviated example of writing the output of krig_3d having kriged a uniform field (which can be volume rendered). Large sections of the data regions of this file are omitted to save space. This is represented by sections of the file with "**** omitted ****" replacing many lines of data.

```

DEFINITION Mesh_Unif+Node_Data
NSPACE 3
NDIM 3
DIMS 41 41 35
COORD_UNITS "ft"
NUM_NODE_DATA 7
POINTS 11281.910004 12211.149994 -29.900000 12515.890015
13259.449951 0.900000
NODE_DATA_DEF 0 "VOC" "log_ppm" IN_FILE
NODE_DATA_DEF 1 "Confidence-VOC" "linear_%" IN_FILE
NODE_DATA_DEF 2 "Uncertainty-VOC" "linear_Unc" IN_FILE
NODE_DATA_DEF 3 "Geo_Layer" "linear_" IN_FILE
NODE_DATA_DEF 4 "Elevation" "linear_ft" IN_FILE
NODE_DATA_DEF 5 "Layer Thickness" "linear_ft" IN_FILE
NODE_DATA_DEF 6 "Material_ID" "linear_" IN_FILE
NODE_DATA_START
-2.357487 34.455845 2.325005 0.000000 -29.900000 30.799999
0.000000
-3.000000 34.977974 0.000000 0.000000 -29.900000 30.799999
0.000000
-3.000000 35.603794 0.000000 0.000000 -29.900000 30.799999
0.000000

```

```

***** OMITTED *****
-3.000000 30.056839 0.000000 0.000000 0.900000 30.799999 0.000000
-3.000000 29.858747 0.000000 0.000000 0.900000 30.799999 0.000000
-3.000000 29.673925 0.000000 0.000000 0.900000 30.799999 0.000000
END

```

krig_3d - EFF Split file representing a uniform field: The file below is a complete example of writing the output of krig_3d having kriged a uniform field (which can be volume rendered). Note that the .EFF file is quite small, but references the data in a separate file named krig_3d_uniform_split.nd.

```

DEFINITION Mesh_Unif+Node_Data
NSPACE 3
NDIM 3
DIMS 41 41 35
COORD_UNITS "ft"
NUM_NODE_DATA 7
POINTS 11281.910004 12211.149994 -29.900000 12515.890015
13259.449951 0.900000
NODE_DATA_DEF 0 "VOC" "log_ppm" FILE "krig_3d_uniform_split.nd"
ROW 1 COLS 1
NODE_DATA_DEF 1 "Confidence-VOC" "linear_%" FILE
"krig_3d_uniform_split.nd" ROW 1 COLS 2
NODE_DATA_DEF 2 "Uncertainty-VOC" "linear_Unc" FILE
"krig_3d_uniform_split.nd" ROW 1 COLS 3
NODE_DATA_DEF 3 "Geo_Layer" "linear_" FILE
"krig_3d_uniform_split.nd" ROW 1 COLS 4
NODE_DATA_DEF 4 "Elevation" "linear_ft" FILE
"krig_3d_uniform_split.nd" ROW 1 COLS 5
NODE_DATA_DEF 5 "Layer Thickness" "linear_ft" FILE
"krig_3d_uniform_split.nd" ROW 1 COLS 6
NODE_DATA_DEF 6 "Material_ID" "linear_" FILE
"krig_3d_uniform_split.nd" ROW 1 COLS 7
END

```

Large sections of the data regions of the data file krig_3d_uniform_split.nd are omitted below to save space. This is represented by sections of the file with "**** omitted ****" replacing many lines of data.

```

-2.357487 34.455845 2.325005 0.000000 -29.900000 30.799999
0.000000
-3.000000 34.977974 0.000000 0.000000 -29.900000 30.799999
0.000000
-3.000000 35.603794 0.000000 0.000000 -29.900000 30.799999
0.000000
***** OMITTED *****
-3.000000 30.056839 0.000000 0.000000 0.900000 30.799999 0.000000
-3.000000 29.858747 0.000000 0.000000 0.900000 30.799999 0.000000
-3.000000 29.673925 0.000000 0.000000 0.900000 30.799999 0.000000

```

krig_3d_Geology & krig_3d - EFF file representing multiple geologic layers with analyte (e.g. chemistry): The file below is an abbreviated example of writing the output of krig_3d having kriged analyte (e.g. chemistry) data with geology input. Large sections of the data regions of this file are omitted to save space. This is represented by sections of the file with "**** omitted ****" replacing many lines of data.

```
NSPACE 3
NNODES 66355
COORD_UNITS "ft"
NUM_NODE_DATA 7
NCELL_SETS 5
NODES IN_FILE
11153.998856 12722.725708 2.970446
11161.871033 12715.198792 2.783408
11169.743210 12707.671875 2.594242
***** OMITTED *****
11250.848221 12865.266907 -42.575920
11248.750000 12870.909973 -42.000000
11243.389938 12870.020935 -42.474934
NODE_DATA_DEF 0 "TOTHc" "log_mg/kg" IN_FILE
NODE_DATA_DEF 1 "Confidence-TOTHc" "linear_%" IN_FILE
NODE_DATA_DEF 2 "Uncertainty-TOTHc" "linear_Unc" IN_FILE
NODE_DATA_DEF 3 "Geo_Layer" "Linear_" IN_FILE
NODE_DATA_DEF 4 "Elevation" "Linear_ft" IN_FILE
NODE_DATA_DEF 5 "Layer Thickness" "Linear_ft" IN_FILE
NODE_DATA_DEF 6 "Material_ID" "Linear_" IN_FILE
NODE_DATA_START
-0.777059 27.239126 15.861248 0.000000 2.970446 8.270601 2.000000
-0.661227 27.349216 16.503609 0.000000 2.783408 8.270658 2.000000
-0.288564 27.512394 18.822187 0.000000 2.594242 8.261375 2.000000
***** OMITTED *****
2.886921 69.551514 1.128253 4.000000 -42.575920 13.628321
4.000000
3.113943 99.999977 0.000000 4.000000 -42.000000 13.654032
4.000000
3.070153 72.869553 0.841437 4.000000 -42.474934 13.646055
4.000000
CELL_SET_DEF 0 8120 Hex "Fill" IN_FILE
CELL_SET_DEF 1 14680 Hex "Silt" IN_FILE
CELL_SET_DEF 2 6502 Hex "Clay" IN_FILE
CELL_SET_DEF 3 11284 Hex "Gravel" IN_FILE
CELL_SET_DEF 4 14412 Hex "Sand" IN_FILE
CELL_START
0 1 42 41 1681 1682 1723 1722
1 2 43 42 1682 1683 1724 1723
```

```

2 3 44 43 1683 1684 1725 1724
***** OMITTED *****
54462 54503 66349 66348 56143 56184 66353 66352
54503 54502 66350 66349 56184 56183 66354 66353
54502 54461 66347 66350 56183 56142 66351 66354
END

```

Post_samples - EFF file representing spheres: The file below is a complete example of writing the output of post_samples' blue-black field port having read the file initial_soil_investigation_subsite.apdv. This data file has 99 samples with data that was log processed. If this file is read by load_evs_field. It creates all 99 spheres colored and sized as they were in Post_samples. The tubes and any labeling are not included in the field port from which this file was created.

```

DEFINITION Mesh+Node_Data
NSPACE 3
NNODES 99
COORD_UNITS "units"
NUM_NODE_DATA 2
NCELL_SETS 1
NODES IN_FILE
11566.340027 12850.590027 -10.000000
11566.340027 12850.590027 -70.000000
11566.340027 12850.590027 -160.000000
11586.340027 13050.589966 -10.000000
11586.340027 13050.589966 -70.000000
11586.340027 13050.589966 -160.000000
11381.700012 12747.500000 -15.000000
11381.700012 12747.500000 -25.000000
11414.399994 12781.099976 -15.000000
11414.399994 12781.099976 -25.000000
11338.000000 12830.799988 -10.000000
11338.000000 12830.799988 -65.000000
11338.000000 12830.799988 -115.000000
11338.000000 12830.799988 -165.000000
11410.290009 12724.690002 -5.000000
11410.290009 12724.690002 -35.000000
11410.290009 12724.690002 -45.000000
11410.290009 12724.690002 -125.000000
11410.290009 12724.690002 -175.000000
11427.000000 12780.900024 -10.000000
11427.000000 12780.900024 -30.000000
11427.000000 12780.900024 -80.000000
11416.899994 12819.450012 -10.000000
11416.899994 12819.450012 -30.000000
11416.899994 12819.450012 -70.000000
11416.899994 12819.450012 -95.000000

```

11416.899994	12819.450012	-105.000000
11416.899994	12819.450012	-120.000000
11416.899994	12819.450012	-140.000000
11401.730011	12897.770020	-10.000000
11401.730011	12897.770020	-30.000000
11401.730011	12897.770020	-80.000000
11401.730011	12897.770020	-110.000000
11401.730011	12897.770020	-145.000000
11401.730011	12897.770020	-180.000000
11259.670013	12819.289978	-10.000000
11259.670013	12819.289978	-40.000000
11259.670013	12819.289978	-70.000000
11259.670013	12819.289978	-95.000000
11259.670013	12819.289978	-140.000000
11340.489990	12892.609985	-30.000000
11340.489990	12892.609985	-55.000000
11340.489990	12892.609985	-80.000000
11340.489990	12892.609985	-110.000000
11340.489990	12892.609985	-130.000000
11340.489990	12892.609985	-165.000000
11248.750000	12870.909973	-10.000000
11248.750000	12870.909973	-35.000000
11248.750000	12870.909973	-45.000000
11248.750000	12870.909973	-85.000000
11248.750000	12870.909973	-110.000000
11248.750000	12870.909973	-160.000000
11248.750000	12870.909973	-210.000000
11086.519997	12830.669983	-15.000000
11086.519997	12830.669983	-30.000000
11086.519997	12830.669983	-80.000000
11086.519997	12830.669983	-130.000000
11211.869995	12710.750000	-30.000000
11211.869995	12710.750000	-80.000000
11211.869995	12710.750000	-135.000000
11199.039993	12810.159973	-20.000000
11199.039993	12810.159973	-40.000000
11199.039993	12810.159973	-85.000000
11199.039993	12810.159973	-150.000000
11298.000000	12808.630005	-60.000000
11496.339996	12753.590027	-10.000000
11496.339996	12753.590027	-30.000000
11496.339996	12753.590027	-80.000000
11496.339996	12753.590027	-110.000000
11496.339996	12753.590027	-150.000000
11309.029999	12948.989990	-10.000000

```
11309.029999 12948.989990 -35.000000
11309.029999 12948.989990 -95.000000
11309.029999 12948.989990 -125.000000
11309.029999 12948.989990 -130.000000
11209.350006 12993.940002 -5.000000
11209.350006 12993.940002 -35.000000
11209.350006 12993.940002 -60.000000
11209.350006 12993.940002 -95.000000
11209.350006 12993.940002 -125.000000
11301.970001 13079.660034 -20.000000
11301.970001 13079.660034 -30.000000
11301.970001 13079.660034 -85.000000
11301.970001 13079.660034 -125.000000
11286.769989 13026.699951 -30.000000
11286.769989 13026.699951 -45.000000
11286.769989 13026.699951 -75.000000
11286.769989 13026.699951 -120.000000
11393.470001 12948.900024 -20.000000
11393.470001 12948.900024 -45.000000
11393.470001 12948.900024 -95.000000
11393.470001 12948.900024 -110.000000
11393.470001 12948.900024 -130.000000
11393.470001 12948.900024 -170.000000
11251.300003 12929.270020 -10.000000
11251.300003 12929.270020 -30.000000
11251.300003 12929.270020 -80.000000
11251.300003 12929.270020 -120.000000
11251.300003 12929.270020 -145.000000
NODE_DATA_DEF 0 "TOTHC" "log_mg/kg" IN_FILE
NODE_DATA_DEF 1 "" "" ID 668 IN_FILE
NODE_DATA_START
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
1.322219 4.998203
2.806180 4.998203
1.602060 4.998203
-3.000000 4.998203
```


-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
1.845098 4.998203
2.278754 4.998203
-3.000000 4.998203
1.296665 4.998203
-3.000000 4.998203
1.278754 4.998203
3.716003 4.998203
1.623249 4.998203
1.505150 4.998203
-3.000000 4.998203
1.707570 4.998203
-3.000000 4.998203
3.770852 4.998203
3.869232 4.998203
1.113943 4.998203
-3.000000 4.998203
2.025306 4.998203
3.434569 4.998203
3.594039 4.998203
2.454845 4.998203
-3.000000 4.998203
2.740363 4.998203
2.079181 4.998203
3.806180 4.998203
4.908485 4.998203
2.176091 4.998203
-3.000000 4.998203
3.792392 4.998203
3.362897 4.998203
4.255272 4.998203
3.699387 4.998203
3.518514 4.998203
3.301030 4.998203
3.113943 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
1.361728 4.998203
-3.000000 4.998203

```
-3.000000 4.998203
2.000000 4.998203
1.643453 4.998203
1.732394 4.998203
1.643453 4.998203
3.556303 4.998203
-0.522879 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
3.079181 4.998203
-3.000000 4.998203
2.633468 4.998203
1.505150 4.998203
-3.000000 4.998203
-3.000000 4.998203
-0.920819 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-0.886057 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-3.000000 4.998203
-0.096910 4.998203
-3.000000 4.998203
4.000000 4.998203
2.000000 4.998203
1.602060 4.998203
1.000000 4.998203
-0.301030 4.998203
-3.000000 4.998203
1.785330 4.998203
-3.000000 4.998203
0.431364 4.998203
4.518514 4.998203
-3.000000 4.998203
CELL_SET_DEF 0 99 Point "" IN_FILE
CELL_START
0
1
2
```

3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47

48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92

93
94
95
96
97
98
END

read_vtk

read_vtk reads a dataset from any of the following 9 [VTK](#) file formats. Please note that VTK's file formats do not include coordinate units information, not analyte units. There is a parameter which allows you to specify coordinate units (meters are the default).

- vtk: legacy format
- vtr: Rectilinear grids
- vtp: Polygons (surfaces)
- vts: Structured grids
- vtu: Unstructured grids
- pvtp: Partitioned Polygons (surfaces)
- pvtr: Partitioned Rectilinear grids
- pvts: Partitioned Structured grids
- pvtu: Partitioned Unstructured grids

Module Output [Ports](#)

- **Output** [Field] Outputs the saved field.
- **Output Object** [Renderable]: Outputs to the viewer.

read_cad

General Module Function

The read_cad module will read the following versions of CAD files:

- AutoCAD DWG and DXF files through AutoCAD 2021 (version 24.0)
- Bentley Microstation DGN files through Version 8.

This module provides the user with the capability to integrate site plans, buildings, and other 2D or 3D features into the EVS visualization, to provide a frame of reference for understanding the three dimensional relationships between the site features, and characteristics of geologic, hydrologic, and chemical features. The drawing entities are treated as three dimensional objects, which provides the user with a lot of flexibility in the placement of CAD objects in relation to EVS objects in the visualization. The [surfmap](#) and geologic_surfmap modules allow the user to drape CAD line-type entities (not 3D-Faces) onto three dimensional surfaces.

Virtually all AutoCAD object types are supported including points, lines (of all types), 3D surface objects and 3D volumetric objects.

AutoCAD drawings can be drawn in model space (MSPACE) or paper space (PSPACE). Drawings in paper space have a defined viewport which has coordinates near the origin. When read into EVS this creates objects which are far from your true model coordinates. For this reason, all drawings for use in our software should be in model space.

Module Side [Port](#)

- **Z Scale Use the Global Z Scale and avoid using this port in general:**
[Number] Accepts Z Scale (vertical exaggeration) from other modules.

Module Output [Ports](#)

- **Output** [Field] Outputs the CAD layers.
- **Output Object** [Renderable]: Outputs to the viewer

read_geometry

General Module Function

The read_geometry module will read STL, PLY, OBJ and .G files containing object geometries.

This module provides the user with the capability to integrate site plans, topography, buildings, and other 3D features into the EVS visualizations.

NOTE: This module intentionally does not have a Z-Scale port since this class of files are so often not in a user's model projected coordinate system. Instead we are providing a Transform Settings group that allows for a much more complex set of transformations including scaling, translations and rotations.

Module Output [Ports](#)

- **Output** [Field] Outputs the CAD layers.
- **Output Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

Transform Settings: This allows you to add any number of Translation or Scale transformations in order to place your Wavefront Object in the same coordinate space as the rest of your "Real-World" model. It is very typical that Wavefront Objects are in a rather arbitrary local coordinate system that will have no defined transformation to any standard coordinate projection.

Generally you should know if the coordinates are feet or meters and if those are not correct, do that scaling as your first set of transforms.

It will be up to you to determine the set of translations that will properly place this object in your model. Hopefully rotations will not be required, but they are possible with the Transform List.

read_vector_gis

The read_vector_gis module reads the following vector file formats: ESRI Shapefile (*.shp); Arc/Info E00 (ASCII) Coverage (*.e00); Atlas BNA file (*.bna); GeoConcept text export (*.gxt); GMT ASCII Vectors (*.gmt); and the MapInfo TAB (*.tab) format.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output** [Field] Outputs the GIS data.
- **Output Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties controls Z Scale
- Data Processing: controls clipping, processing (Log) and clamping of input data

raster_to_geology

The raster_to_geology module reads several different raster format files in EVS Geology format. These formats include DEMs, Surfer grid files, Mr. Sid files, ADF files, etc.. Multiple raster_to_geology can be combined with combine_geology into a 3D geologic model. Alternatively, a single file can be displayed as a surface (with geologic_surfaces) or you can export its coordinates (with write_coordinates) to use the values in a GMF file.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Geologic Field** [Field / minor] Outputs a 2D grid with data similar in functionality to krig_3d_geology

buildings

The buildings module reads C Tech's .BLDG file and creates various 3D objects (boxes, cylinders, wedge-shapes for roofs, simple houses etc.), and provides a means for scaling the objects and/or placing the objects at user specified locations. The objects are displayed based on x, y & z coordinates supplied by the user in a .bldg file, with additional scaling option controls on the buildings user interface.

Each object is made up of 3D volumetric elements. This allows for the output of buildings to be cut or sliced to reveal a cross section through the buildings.

Selecting the "Edit Buildings" toggle will open an additional section which provides the ability to interactively create 3D buildings in your project.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **View** [View] Connects to the viewer to allow interactive building creation.

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output** [Field] Outputs the buildings as a field which can be sliced, cut or further subsetting.
- **Output Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties controls Z Scale and file input and output
- Default Building Settings: Defines the default values when a building is interactive created
- Building Settings: Shows the parameters for all buildings.

Sample Buildings File

Below is an example buildings file. Note that the last 4 columns are optional and contain RGB color values (three numbers from zero to 1.0) and/or a building ID number that can be used for coloring. If only color values are supplied (3 numbers) the ID is automatically determined by the row number. If four numbers are provided it is assumed that the last one is the ID. If only one number is provided it is the ID.

The file below is shown in a table (with dividing lines) for clarity only. The first uncommented line is the number 16 which defines the number of rows of buildings data. The actual file is a simple ASCII file with separators of *space, comma and/or tab*.

```
# EVS
# Copyright (c) 1994-2008 by
# C Tech Development Corporation
# All Rights Reserved
#
# This software comprises unpublished confidential information of
# C Tech Development Corporation and may not be used, copied or made
# available to anyone, except in accordance with the license
# under which it is furnished.
#
#
# C Tech 3D Building file
# Building 0 is a unit box with base at z=0.0 centered at origin x,y
# Building 1 is a gabled roof for the unit box
# (to make it a house) with base at z=0.0 centered at origin x,y
```



```

# Building 2 is a wedge roof for the unit box
# (to make it a house) with base at z=0.0 centered at origin x,y
# Building 3 is a Equilateral (or Isosceles) Triangular Building 3 side
# Building 4 is a Right Triangular Building 3 side
# Building 5 is a Hexagonal (6 side) cylinder
# Building 6 is a Octagonal (8 side) cylinder
# Building 7 is a 16 side cylinder
# Building 8 is a 32 side cylinder
# Building 9 is a 16 sided horiz. cylindrical tank (Height & Width
equal diameter, Length is along x)
# Building 10 is a 32 sided horiz. cylindrical tank (Height & Width
equal diameter, Length is along x)
# Building 11 is a right angle triangle, height only at right angle
# Building 12 is a right angle triangle, height at non-right angle
# Building 13 is a right angle triangle, height at right angle and 1
non-right angle
# Lines beginning with "#" are comments
# First uncommented line is number of buildings
# X   Y Z LengthWidthHeight Angle Bldg_Type Color and/orID
16
0     0     10     50     50     20     0     0     1
0     100    0     50     50     30     30     0     2
0     100    30     60     50     20     30     1     2
0     200    0     50     50     30     10     0     3
0     200    30     50     50     25     10     2     3
200   0     0     50     50     50     0     3     4
100   100    0     40     40     20     15     4     5
200   100    0     40     40     30     30     5     6
200   200    0     50     50     50     0     6     7
100   200    0     40     60     20     -45    7     8
100   0     0     50     50     40     0     8     9
300   0     0     60     20     20     -45    9     0.8     0.6     0.4     10
300   100    0     50     50     30     0     10     0.4     0.6     0.4     11
0     300    0     50     50     50     0     11     1.0     0.4     0.4     12
100   300    0     50     50     50     0     12     0.4     1.0     0.4     13
200   300    0     50     50     50     0     13     0.4     0.4     1.0     14

```

read_lines

The read_lines module is used to visualize a series of points with data connected by lines. read_lines accepts three different file formats, with the APDV file format the lines are connected by boring ID, with the ELF (EVS Line File) format each line is

made by defining the points that make up the line, and with the SAD (Strike and Dip) file format, there is a choice to connect each sample by ID or by Data Value. SAD files connect by ID – If a *.sad file has been read the lines will be connected by ID.

SAD files connect by Data – If a *.sad file has been read the lines will be connected by the data component.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field] Outputs the subsetted field as faces.
- **Output Object** [Renderable]: Outputs to the viewer.

EVS Line File Example

Discussion of EVS Line Files

EVS line files contain horizontal and vertical coordinates, which describe the 3-D locations and values of properties of a system. Line files must be in ASCII format and can be delimited by commas, spaces, or tabs. They must have an .elf suffix to be selected in the file browsers of EVS modules. Each line of the EVS line file contain the coordinate data for one sampling location and up to 300 (columns of) property values. There are no computational restrictions on the number of lines that can be included in a file.

EVS Line Files

EVS **Line** Files consist of file tags that delineate the various sections of the file(s) and data (coordinates, nodal and/or cell data). The file tags are discussed below followed by portions of a few example files.

FILE TAGS:

The file tags for the ASCII file formats (shown in ***Bold Italics***) are discussed below with a representative example. They are given in the appropriate order. If you need assistance creating software to write these file formats, please contact support@ctech.com.

COORD_UNITS "ft"

Defines the coordinate units for the file. These should be consistent in X, Y, and Z.

NUM__DATA 7 1

Number of nodal data components followed by the number of cell data components.

NODE_DATA_DEF 0 "TOTHC" "log_ppm"

NODE_DATA_DEF specifies the definition of a nodal data component. The second value is the data component number, the third is the name, and the 4th is the units.

CELL_DATA_DEF 0 "Indicator" "Discreet Unit"

Definition of cell data. Same options as NODE_DATA_DEF

LINE 12 1

Beginning of a line segment is followed on the same line by the cell data values.

Following this line should be the points making up the line in the following format:

X, Y, Z coordinates followed by nodal data values.

64718.310547 37500.000000 -1250.000000 1 -1250.000000

63447.014587 35101.682129 -2000.000000 2 -2000.000000

CLOSED

This flag is used at the end of a line definition to indicate the end of the line should be connected to the beginning of the line.

END

Marks the end of the data section of the file. (Allows us to put a password on .eff files)

EXAMPLE FILE

NUM_DATA 2 0

NODE_DATA_DEF 0 "Node_Number" "Linear_ID"

NODE_DATA_DEF 1 "Distance" "Linear_ft"

LINE

1900297.026154 677367.319824 72.000000 0.000000 0.000000

1900314.256775 677438.611328 72.000000 1.000000 73.344208

1900314.687561 677442.703522 72.000000 2.000000 77.459015

1900316.410645 677447.011261 72.000000 3.000000 82.098587

1900319.641266 677447.442018 72.000000 4.000000 85.357796

1900345.487030 677441.411530 72.000000 5.000000 111.897774

1900360.563782 677439.472870 72.000000 6.000000 127.098656

1900363.579193 677447.226807 72.000000 7.000000 135.418289

1900365.517822 677447.226807 72.000000 8.000000 137.356918

1900365.948608 677438.396118 72.000000 9.000000 146.198105

1900379.733032 677436.888245 72.000000 10.000000 160.064758

1900405.578766 677432.150055 72.000000 11.000000 186.341217

1900497.331879 677416.427002 72.000000 12.000000 279.431763

1900511.331512 677414.919464 72.000000 13.000000 293.512329

1900525.762268 677411.257721 72.000000 14.000000 308.400421

1900527.269775 677405.442444 72.000000 15.000000 314.407898

1900524.900696 677399.411926 72.000000 16.000000 320.887085

1900522.531311 677391.012024 72.000000 17.000000 329.614746

1900517.362366 677357.196808 72.000000 18.000000 363.822754
 1900501.854828 677266.951569 72.000000 19.000000 455.390686
 1900501.639282 677262.213379 72.000000 20.000000 460.133789
 1900500.777710 677255.321014 72.000000 21.000000 467.079773
 1900496.470306 677250.151733 72.000000 22.000000 473.808472
 1900487.208862 677241.751816 72.000000 23.000000 486.311798
 1900450.378204 677201.906097 72.000000 24.000000 540.572083
 1900403.568481 677152.368134 72.000000 25.000000 608.727478
 1900356.758759 677102.830177 72.000000 26.000000 676.882874
 1900309.949036 677053.292221 72.000000 27.000000 745.038269
 1900286.257172 677028.523243 72.000000 28.000000 779.313721
 1900278.718445 677022.923517 72.000000 29.000000 788.704651
 1900269.672546 677024.431061 72.000000 30.000000 797.875305
 1900217.334717 677035.200397 72.000000 31.000000 851.309631
 1900232.196075 677097.230453 72.000000 32.000000 915.095154
 1900247.057434 677159.260513 72.000000 33.000000 978.880615
 1900252.226715 677179.937317 72.000000 34.000000 1000.193787
 1900267.159851 677242.326401 72.000000 35.000000 1064.345215
 1900282.093018 677304.715485 72.000000 36.000000 1128.496460
 1900297.026154 677367.104584 72.000000 37.000000 1192.647827
 END

strike_and_dip

General Module Function

The strike_and_dip module is used to visualize sampled locations. It places a disk, oriented by strike and dip, at each sample location. Each disk is probable and can be colored by a picked color, by Id, or by data value. If an ID is present, such as a boring ID, then there is an option to place tubes between connected disks, or those disks with similar Id's.

Strike and dip refer to the orientation of a geologic feature. The strike is a line representing the intersection of that feature with the horizontal plane (though this is often the ground surface). Strike is represented with a line segment parallel to the strike line. Strike can be given as a compass direction (a single three digit number representing the azimuth) or basic compass heading (e.g. N, E, NW).

The dip gives the angle of descent of a feature relative to a horizontal plane, and is given by the number (0°-90°) as well as a letter (N,S,E,W, NE, SW, etc.) corresponding to the rough direction in which feature bed is dipping.

NOTE: We do not support the Right-Hand Rule, therefore all dip directions must have the direction letter(s).

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output** [Field] Outputs the subsetted field as edges

- **Output Object** [Renderable]: Outputs to the viewer

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the Z scaling and edge angle used to determine what edges should be displayed
- Display Settings: controls the type and specific data to be output or displayed

Strike and Dip File Example

Discussion of Strike and Dip Files

Strike and dip files consist of 3D coordinates along with two orientation values called strike and dip. A simple disk is placed at the coordinate location and then the disk is rotated about Z to match the strike and then rotated about Y to match the dip. An optional id and data value can be used to color the disk.

Format:

You may insert comment lines in C Tech Strike and Dip (.sad) input files. Comments can be inserted anywhere in a file and must begin with a '#' character.

Strike can be defined in the following ways :

1) For strikes running along an axis:

N, S, NS, SN are all equivalent to 0 or 180, and will always have a dip to E or W

E, W, EW, WE are all equivalent to 90 or 270, and will always have a dip to N or S

NE, SW are both equivalent to 135 or 315, and can have a dip specified to N, S, E, or W

NW, SE are both equivalent to 45 or 225, and can have a dip specified to N, S, E, or W

2) For all other strikes: any compass direction between 0 and 360 degrees can be specified, with the dip direction clarifying which side of the strike is downhill.

Dip can be defined only in degrees in the range of 0 to 90.0 followed by a direction such as 35.45E

There is no required header for this file type.

Each line of the file must contain:

X, Y, Z, Strike, Dip, ID (optional), and Data (optional).

NOTE: The ID can only contain spaces if enclosed in quotation marks (ex "ID 1").

EXAMPLE FILE

```
# x      y      z      strike    dip
51.967  10.948  26.127  35.205   59.8031E
50.373  33.938  26.127  13.048   68.49984E
```

```

51.654 60.213 26.127 139.18 76.74215E
50.529 83.203 26.127 213.50 62.94599E
64.358 76.634 11.471 114.23 80.38694E
66.430 33.938 -6.849 41.421 60.38837E
75.901 50.360 -21.505 60.141 72.88960E
72.943 7.663 -21.505 5.255 65.51247E
101.90 30.654 -72.801 77.675 65.9524E
81.339 50.360 -43.489 244.95 70.7079E
72.263 73.350 -21.505 82.929 69.3159E
89.897 73.350 -61.809 31.531 55.6570E
END

```

FILE TAGS:

The file tags for the ASCII file formats (shown in ***Bold Italics***) are discussed below with a representative example. They are given in the appropriate order. If you need assistance creating software to write these file formats, please contact support@ctech.com.

COORD_UNITS "ft"

Defines the coordinate units for the file. These should be consistent in X, Y, and Z.

END (this is optional, but should be used if any lines will follow your actual data lines)

load_glyph

load_glyph replaces the Glyphs sub-library that was in the tools library. It reads glyphs saved in any of the three primary EVS field file formats and allows you to modify the shape and orientation of the glyph to allow it to be used in various modules that employ glyphs in slightly different ways. These include glyph, geo_glyph, place_glyph, drive_glyph, drive_glyphs, advector, post_samples, etc. Most modules EXCEPT post_samples will use the glyphs without changing the default alignment. The supported file formats are:

- 1) .eff ASCII format, best if you want to be able to open the file in an editor or print it
- 2) .efz GNU Zip compressed ASCII, same as .eff but in a zip archive
- 3) .efb binary compressed format, the smallest & fastest format due to its binary form

For a description of the [.EFF file formats click here](#).

The objects saved in the .efx files should be simple geometric objects ideally designed to fit in a unit box centered at the origin (0,0,0). For optimal performance the objects should not include nodal or cell data. You may create your own objects or use any of the ones that C Tech supplies in the ctech\data\glyphs folder.

Module Output [Ports](#)

- **Output** [Field] Outputs the saved glyph.
- **Output Object** [Renderable]: Outputs to the viewer.

symbols

Symbols creates symbolic representations of different borehole identifiers based on a set of user defined parameters. The symbols are displayed at the top of the each borehole based on its x,y & z coordinates. A sample file with 48 predefined symbols is included, but it can be customized to produce special symbols.

Each symbol is made up of three components. The first shape is a fixed polygon with an outline. The thickness of the outline is selectable (via the control panel). A second polygon, which overlaps the first and has the same number of sides, has selectable minimum and maximum radial values (via the .SYM file). The third component is made up of a user defined set of lines (0 gives no lines). Each polygon has the same number of faces as defined in the #face parameter in the .SYM file. The area created by the difference between the Rmin value and the Rmax value is solid.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Input Geologic Field** [Field] Accepts a data field from krig_3d_geology to krig data into geologic layers.
- **Filename** [String / minor] Allows the sharing of file names between similar modules.

Module Output [Ports](#)

- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Sample Symbols** [Renderable]: Outputs to the viewer

EVS.SYM file:

The following is a listing of the file evs.sym in evs\data\special. This file can be customized to produce other symbols.

```
# rmin rmax lmin lmax #face #line bw rot lrot rvrs name
48
1 0. 1 1 1 12 0 1 0 0 0 solid fill circle
2 0. .7 .7 1.2 12 4 1 0 0 0 solid fill circle w/ line
3 .8 1 1 1 12 0 1 0 0 0 circle ring
4 .4 1 1 1 12 0 1 0 0 0 fat circle ring
5 .0 .4 1 1 12 4 1 0 0 0 circle ring w/lines
6 .8 .7 .7 1.2 12 4 1 0 0 0 circle ring w/lines
7 .4 1 1 1 4 0 1 0 0 0 fat square box
8 .8 1 1 1 4 0 1 45 0 0 thin square box
9 .0 1 1 1 4 0 1 45 0 0 solid square box
10 .0 .7 .7 1.2 12 4 2 30 -30 0 half moon bk top w/line
11 .0 .7 .7 1.2 12 4 2 300 -300 0 half moon bk rt w/line
12 .0 .7 .7 1.2 12 4 2 210 -210 0 half moon bk bot w/line
13 .0 .7 .7 1.2 12 4 4 30 -30 0 qrtr moon bk ul w/line
14 .0 .7 .7 1.2 12 4 4 120 -120 0 qrtr moon bk ur w/line
15 .8 .7 0 1.2 12 4 1 0 0 0 open bulls-eye
16 .0 .7 .7 1.2 12 4 2 120 -120 0 half moon bk lft w/line
17 .0 1 1 1. 3 0 1 30 0 0 solid black triangle
18 .8 .7 .7 1.2 3 3 1 90 0 0 hollow blk triangle w/line
19 .0 1 1 1. 3 0 1 90 0 0 solid black triangle
```

```

20 .8 .7 .7 1.2 4 4 1 0 0 0 diamond w/line
21 .8 1 1 1. 4 0 1 0 0 0 diamond
22 .0 .7 .7 1.2 4 4 1 0 0 0 solid diamond w/line
23 .0 .7 .7 1.2 6 6 4 0 0 0 hex moon bk ul w/line
24 .0 .7 .7 1.2 6 6 4 180 0 0 hex moon bk ul w/line
25 0. 1 1 1 12 0 1 0 0 1 solid fill circle
26 0. .7 .7 1.2 12 4 1 0 0 1 solid fill circle w/ line
27 .8 1 1 1 12 0 1 0 0 1 circle ring
28 .4 1 1 1 12 0 1 0 0 1 fat circle ring
29 .0 .4 1 1 12 4 1 0 0 1 circle ring w/lines
30 .8 .7 .7 1.2 12 4 1 0 0 1 circle ring w/lines
31 .4 1 1 1 4 0 1 0 0 1 fat square box
32 .8 1 1 1 4 0 1 45 0 1 thin square box
33 .0 1 1 1 4 0 1 45 0 1 solid square box
34 .0 .7 .7 1.2 12 4 2 30 -30 1 half moon bk top w/line
35 .0 .7 .7 1.2 12 4 2 300 -300 1 half moon bk rt w/line
36 .0 .7 .7 1.2 12 4 2 210 -210 1 half moon bk bot w/line
37 .0 .7 .7 1.2 12 4 4 30 -30 1 qrtr moon bk ul w/line
38 .0 .7 .7 1.2 12 4 4 120 -120 1 qrtr moon bk ur w/line
39 .8 .7 0 1.2 12 4 1 0 0 1 open bulls-eye
40 .0 .7 .7 1.2 12 4 2 120 -120 1 half moon bk lft w/line
41 .0 1 1 1. 3 0 1 30 0 1 solid black triangle
42 .8 .7 .7 1.2 3 3 1 90 0 1 hollow blk triangle w/line
43 .0 1 1 1. 3 0 1 90 0 1 solid black triangle
44 .8 .7 .7 1.2 4 4 1 0 0 1 diamond w/line
45 .8 1 1 1. 4 0 1 0 0 1 diamond
46 .0 .7 .7 1.2 4 4 1 0 0 1 solid diamond w/line
47 .0 .7 .7 1.2 6 6 4 0 0 1 hex moon bk ul w/line
48 .0 .7 .7 1.2 6 6 4 180 0 1 hex moon bk ul w/line

```

sym

Use to number(label) each symbols algorithm. This is the same number used in the last column of the APDV data file.

Rmin, Rmax, Lmin, and Lmax

These values determine the size of the three possible shapes used to create each symbol. The center point is at 0.0 and the outer edge of the polygons is at 1.0. The x/y lines can start at the center(0.0) or at any other position within the polygon. They can also be extended beyond 1.0 to a position of 1.7.

Rmin

Sets the minimum radius of the inside of the second polygon. With a setting of 0.0 the inside is fully minimized thus creating a solid polygon from the center out to Rmax. A setting of 0.8 will create a solid polygon, with an empty center, out to Rmax.

Rmax

Sets the maximum radius of the outside of the second polygon. A setting of 1.0, places the outside edge directly over the outside edge of the first, fixed polygon. A setting of 0.2 and a Rmin setting of 0.0 creates a small solid polygon centered in the middle of the first polygon.

Lmin

Sets the starting point for the x/y lines. 0.0 starts the lines from the center of the polygons. 1.0 starts the lines at the outer edge of the polygons.

Lmax

Determines how far the lines will extend from Lmin. If Lmax and Lmin equal 1.0 then no lines will be displayed. If Lmin is 0.0 and Lmax is 1.7 the lines will extend from the center past the outer edge of the polygons.

#face

This value determines the number of faces both polygons will display. A value of 12 displays a convincing circle.

#line

This value determines the number of lines.

bw

This parameter allows you to divide the second polygon into alternating light/dark solids with a x/y axis.

Valid values are 1, 2 and 4.

1 = full solid

2 = half solid

3 = alternating quarter solids

rot

Sets the rotation of the symbol in degrees.

lrot

Sets the rotation of the lines relative to the symbol in degrees.

rvrs

Use this parameter to reverse the symbols colors. A value of 0 is normally used but a value of 1 will reverse the colors.

name

an optional description of each symbol. This is only used for reference within the SYM file.

Sample Module Networks

The sample network shown below reads a GEO formatted data file, and a SYM formatted algorithm file. The output is displayed by the geometry viewer.

Symbols

|
|

EVS viewer

A test geology file is included in the evs\special directory called TEST_SYM.GEO. It displays all 48 of the default symbols defined in the file shown above. The symbols are oriented starting at the lower left hand corner and going left to right and bottom to top.

save_evs_field

The save_evs_field module creates a file in one of 5 different formats containing all the mesh and nodal and/or cell data component information sent to the input port.

This module is useful for writing the output of modules which manipulate or interpolate data ([krig_3d](#) , krig_2d , etc.) so that the data will not need to be

processed in the future. The processed data can be read using [load_evs_field](#), which is much faster than reprocessing the data.

For a description of the [.EFF file formats](#) [click here](#).

`save_evs_field` supplants the need for `Write_UCD`, `Write_netCDF` and `Write_EVS_Geology` by incorporating all of their functionality and more in a single module. It saves (writes) a dataset in any of five different EVS compatible file formats, including the new EVS Field Formats:

- `.eff` ASCII format, best if you want to be able to open the file in an editor or print it
- `.efz` GNU Zip compressed ASCII, same as `.eff` but in a zip archive
- `.efb` binary compressed format, the smallest & fastest format due to its binary form

The EVS Field Formats `*.eff`; `*.efz`; and `*.efb` support **all** types of EVS field output including:

1. Uniform fields
2. Geology (from `krig_3d_geology`)
3. Structured fields (such as irregular fields read in from `Read_Field`)
4. Unstructured Cell Data (UCD format) general grids with nodal and/or cell data
5. Special fields containing spheres (which are points with radii)
6. Special fields containing color data (such as from `Read_DXF`)

Note: Because the `.eff`, `.efz` and `.efb` formats better handle all types of EVS output, these three formats are recommended for use over UCD, netCDF or Field.

Module Input [Ports](#)

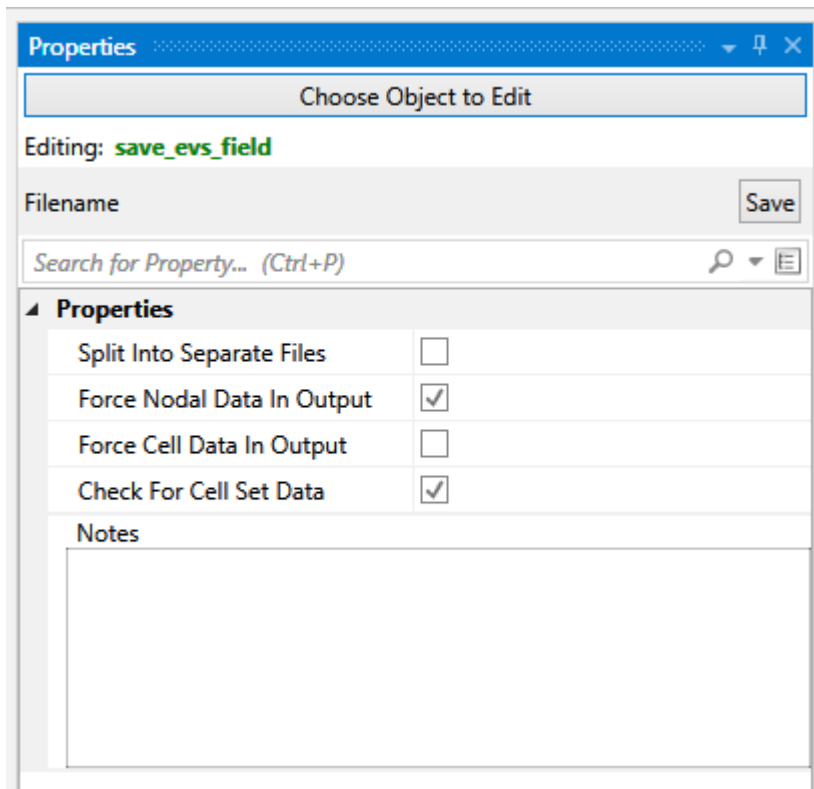
- **Geologic legend Information** [Geology legend] Accepts the geologic material information for the legend module.
- **Input Field** [Field] Accepts the field to be saved.
- **File Notes** [String / minor] Accepts a string to document the settings used to create the field.

Module Parameters

There are only a few parameters in `save_evs_field`, but they provide important functionality and should be understood.

- *Split Into Separate Files*: This toggle applies only to EFF format files and makes it easier to create your own EFF files from similar data. It separates the header file (`.eff`) from the coordinates, data and connectivity.
- *Force Nodal in Output*: This toggle is on by default and ensures that fields without data are tagged as having data because many EVS modules may not allow connections for fields without data. It does not add data, it only tags the file as having data (even if it doesn't)
- *Force Cell in Output*: Similar to the toggle above, but needed far less often.

- *Check For Cell Set Data*: This toggle converts cell data such as (lithology) Material, Geo Layer, etc. and converts it to "Cell Set Data" which results in a significantly smaller file. However, for backwards compatibility with older versions of EnterVol or MVS, this must be OFF.



write_coordinates

write_coordinates provides a means to export an ASCII file containing the coordinates (and optionally the data) of any object in EVS. The output contains a header line and one row for each node in the input field. Each row contains the x, y, & z coordinates and optionally node number and nodal data.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Input Field** [Field] Accepts a field with or without data

write_cad

General Module Function

write_cad will output one or more individual objects (red port) or your complete model (purple input port from the viewer). Volumetric objects in EVS are converted to surface and line type objects.

write_cad preserves the colors of all cells and objects by assigning **cell** colors to each AutoCAD surface or line entity according to the following procedure:

- a) If nodal data is present, the first nodal data component is averaged to the cells and that color is applied. This is equivalent to the appearance of surfaces in EVS with flat shading mode applied.

b) If no nodal data is present, but cell data is, that color is applied. This is equivalent to the appearance of surfaces in EVS with flat shading mode applied.

c) If neither nodal or cell data is present the object's color is used.

The results should look fairly similar to the viewer in EVS except:

- AutoCAD has a very [limited color palette](#) with only 256 total colors. With some datamaps this limitation will be more problematic and it is possible that the nearest AutoCAD color may apply to multiple colors used in a subtle geology datamap.
- AutoCAD lacks of Gouraud shading support (as mentioned above) so all cells are flat shaded.

All "objects" in EVS are converted to separate layers based upon the EVS object name (as shown in the viewer's Object_Selector).

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **View** [View] Connects to the viewer to receive all objects in the view
- **Input Object** [Renderable]: Receives inputs from one or more module's red port

write_vector_gis

The write_vector_gis module will create a file in one of the following vector formats: ESRI Shapefile (*.shp); GMT ASCII Vectors (*.gmt); and MapInfo TAB (*.tab).

Although C Tech allows non-ASCII analyte names, ESRI does not. Please see [this link](#) on acceptable shapefile field (attribute) names. It basically says that only A-Z, a-z, 0-9 and "_" are allowed. The only thing we can do when writing a shapefile is to change any unacceptable (non-ASCII) character to "_" and add a number if there are more than one.

If you plan to create a shapefile it will be better to change the analyte names to an ASCII equivalent that is more meaningful, but uses on the acceptable character set.

NOTE: Make sure to connect write_vector_gis **after** explode_and_scale to ensure that z-scaling is properly compensated.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Input Field** [Field] Accepts a field with or without data

geology_to_raster

geology_to_raster is used in conjunction with krig_3d_geology with rectilinear grids of geologic data. A large number of formats are supported such as Surfer and ESRI

grids. For some formats, each cell in your grid should be the same size. This will require you to adjust the extents of your grid and set the grid resolution according to:

Cell size = (Max:xy - Min:xy) / (grid-resolution -1)

NOTE: YOU MUST SELECT RECTILINEAR GRIDDING IN krig_3d_geology

Module Input [Ports](#)

- **Geology Export Output** [Vistas Data] Accepts output from krig_3d_geology for conversion to raster grids.

write_lines

The write_lines module is used to save a series of points with data connected by lines. These lines are stored in the EVS Line File format.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field with or without data which represents lines

geology_to_vistas

geology_to_vistas is used in conjunction with krig_3d_geology. krig_3d_geology can create finite difference grids based on your geologic data.

It writes the fundamental geologic grid information to a file format that Ground Water Vistas can read.

The output includes the x,y origin; rotation; and x-y resolutions in addition to descriptive header lines preceded by a "#".

Module Input [Ports](#)

- **Geology Export Output** [Vistas Data] Accepts output from krig_3d_geology for conversion to Groundwater Vistas format

export_scene

export_scene connects via the purple port and writes all objects in your view as a "C Tech Web Scene" (*.ctws), a single file which you and your customers can load and view at: <https://viewer.ctech.com/>

Anything in the 2D overlay (Forward Facing Text, 2D legends, add_logo, etc.) will not be written.

We strongly recommend renaming of modules (connected to the viewer) to create a more appropriate Model Tree for C Tech Web Scenes.

Look for many example applications with the suffix ".ctws.evs" in their names 2021 and later Studio Projects releases.

streamlines

The streamlines module is used to produce streamlines or stream-ribbons of a field which is a 2 or 3 element vector data component on any type of mesh. Streamlines, which are simply 3D polylines, represent the pathways particles would travel based on the gradient of the vector field. Stream-ribbons are 3D streamlines which can be rendered. At least one of the nodal data components input to streamlines must be a

vector. The direction of travel of streamlines can be specified to be forwards (toward high vector magnitudes) or backwards (toward low vector magnitudes) with respect to the vector field. Streamlines are produced by integrating a velocity field using the Runge-Kutte method of specified order with adaptive time steps.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Locations Field** [Field] Accepts the starting points for each line

Module Output [Ports](#)

- **Output Field** [Field] Outputs the streamlines or ribbons
- **Output Object** [Renderable]: Outputs to the viewer.

streamline_surface

The streamline surface module is used to produce streamlines on any surface based on its slopes. Streamlines are 3D polylines representing the paths particles would travel based on the slopes of the input surface. The direction of travel of streamlines can be specified to be downhill or uphill for the slope case. A physics simulation option is also available which employs a full physics simulation including friction and gravity terms to compute streamlines on the surface.

The *Physics* radio buttons allow the user to specify whether streamlines will be computed based on the slopes of the surface only or whether a full physics simulation including friction and gravity terms will be used to compute streamlines on the surface. When *Gravity* is selected *Segments perCell* and *Order* do not apply but additional parameters appear for the module. These are:

Integration Time Step is the time step for the numerical integration of the paths. For typical gravity units (like 32 feet per second-squared) this value is in seconds.

Gravity is the coefficient of gravity for your units. If your coordinate units are feet, the appropriate (default) value would be 32 feet per second-squared.

Viscosity Coefficient (v) is the friction term that depends on velocity.

Drag Coefficient (v^2) is the friction term that depends on velocity-squared.

Module Input [Ports](#)

- **Input Surface** [Field] Accepts a data field which must be a surface with elevation data.
- **Input Locations Field** [Field] Accepts the starting points for each line

Module Output [Ports](#)

- **Output Field** [Field] Outputs the streamlines
- **Output Object** [Renderable]: Outputs to the viewer.

drill_path

The drill_path module allows you to interactively create a complex drill path with multiple segments.

Each segment can be defined by one of three methods:

1. Continue Straight: for the specified "Total Length" along the current direction or Initial Drill Direction, if just starting.
2. Target Coordinate: Begin deviating with specified "Segment Length" and specified "Max Angle of Change" (per segment) until you reach the specified "(X,Y,Z)" coordinate.
3. Move to Heading: Begin deviating with specified "Segment Length" and specified "Max Angle of Change" (per segment) until you reach the specified "Heading" and "Dip"

modpath

The modpath module uses the cell by cell flow values generated from a MODFLOW project along with head values and other MODFLOW parameters to trace the path of a particle of water as it moves through the ground. The paths are calculated using the same algorithms used by U.S. Geological Survey MODPATH and the results should be similar.

The modpath module at this point does not handle transient simulations the same way that the U.S.G.S. MODPATH does. It treats each time step as a steady state model, and uses the parameters from the .dwr/.dwz file based on the starting time.

A valid modpath field file (.eff/.efz) should contain the following as cell data components: Head; CCF; ELEV_TOP; ELEV_BOT; and POROSITY. The Head component should contain the head value for each cell, the ELEV_TOP and ELEV_BOT should components should contain the elevation of the top of the cell, and the elevation of the bottom of the cell respectively, and the POROSITY should contain the flow due to porosity for that each cell. All other MODFLOW parameters (drains, wells, recharge, etc..) should be written into a .dwr/.dwz file.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration) from other modules
- **Input Field** [Field] Accepts a data field.
- **Input Starting Locations** [Field] Accepts the starting points for each line
- **Start Date** [Number] The starting time
- **Ending Date** [Number] The ending time

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field] Outputs the streamlines or ribbons
- **Start Date** [Number] The starting time
- **Ending Date** [Number] The ending time
- **Output Object** [Renderable]: Outputs to the viewer.

combine_vect

The combine vect (combine vector) module is used to create an n-length vector by combining n selected scalar data components. The vector length is determined by the Vector Type selector (2D or 3D).

Once the required number of components has been selected, any other data components are grayed out and not selectable. To change selections, first deselect

one of the vector components and then select a new component. If no components are selected, then all components are selectable. The order in which the components are selected will determine in which order they occur in the vector.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field with 2 or more nodal data components.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with selected data
- **Output Object** [Renderable]: Outputs to the viewer.

magnitude

The magnitude module calculates the magnitude of a vector field data component at every node in a mesh. Input to magnitude must contain a mesh of any type and nodal data. Nodal data components can be scalar or vector with up to 3 vector subcomponents.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a vector data field

Module Output [Ports](#)

- **Output Field** [Field] Outputs the scalar data field
- **Output Object** [Renderable]: Outputs to the viewer

Related Modules

[gradient](#)

gradient

The gradient module calculates the vector gradient field of a scalar data component at every node in a mesh. Input to gradient must contain a mesh of any type and nodal data, with at least one scalar nodal data component. Gradient uses a finite-difference method based on central differencing to calculate the gradient on structured (rectilinear) meshes. Shape functions and their derivatives are used to calculate the gradient on unstructured meshes.

Please note that the gradient of (pressure) head points in the direction of increasing head, not the direction that groundwater would flow. Please see the [seepage_velocity](#) module if you wish to compute groundwater flow

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field

Module Output [Ports](#)

- **Output Field** [Field] Outputs the vector data field
- **Output Object** [Renderable / Minor]: Outputs to the viewer

Related Modules

->[magnitude](#)

capture_zone

The capture_zone module utilizes [streamlines](#) technology to determine the volumetric regions of your model for which groundwater flow will be captured by one or more extraction wells.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field with vector data.
- **Well Nodes** [Field] Accepts a field of points representing the well locations

Module Output [Ports](#)

- **Output Field** [Field] Outputs the volumetric regions which are captured

seepage_velocity

The seepage_velocity module is used to compute the vector groundwater flow field visualizations of the vector field.

The input data requirements for the seepage_velocity module are:

1. A data component representing head (can have any name).
2. A Geo_Layer data component.
3. A Material_ID data component. If there is no Material_ID, we treat each layer as a separate material.
 - Layer 0 becomes material -1
 - Layer 1 becomes material -2
 - Layer 2 becomes material -3, etc.

Note: If you use krig_3d to krig head data with geologic input (in Version 6.0 or later) your output will meet these criteria (provided you toggle on these data components under *Kriging Parameters*).

The **Run toggle** determines if the module runs immediately when you change conductivity values.

Head Data Component determines which data component is used to scale and rotate the seepage_velocity velocity vectors. The default selection is the first data component. The Map component radio button list also displays all data components passed to seepage_velocity. Map component determines which data component is used to color the seepage_velocity velocity vectors. By default, the first (0th) data component is selected.

Head Data Component list displays all data components passed to seepage_velocity.

Current Material: allows you to select the Material (or geologic layer) to assign conductivity and porosity properties.

HeadUnits radio button list allows you to specify the units of your head data.

Output Conductivity Units: radio button list allows you to choose the units for specifying the conductivity in all three (x, y, z) directions for each geologic layer. You can choose any units (regardless of your head and coordinate units) and the appropriate conversions will be made for you.

The **Conductivity sliders** (with type-ins) allow you to change the log10 of the x, y, & z conductivity. These specify log values because conductivities vary over many orders of magnitude. These update when the (Linear) type-ins are changed.

The **Conductivity type-ins** allow you to change the x, y, & z conductivity. These are actual values and update when the sliders are changed.

The **Effective Porosity slider** (with type-in buttons) allows you to change the value of effective porosity.

Material (#/Name): allows you to specify the material type if it is not specified in your geologic layers. This is only to help you assign proper conductivities.

Data passed to the field port must be a 3D mesh with data representing heads and *normally* multiple Materials (or geologic layers).

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a data field with geologic and head data

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field] Outputs the vector data field

Technical Details

Inherent in the solution of seepage velocity implemented in this module is the assumption that within each geologic layer/material the conductivities are uniform. Clearly, this will never be completely accurate, however we would contend that there is seldom if ever a better measure of the site conductivities (true conductivity tensor) than the site heads because head is far easier to measure. Furthermore, geologic materials can be deposited such that their conductivities are very complex and directional and most groundwater models (e.g. MODFLOW) do not provide a way to reflect this EVEN IF IT COULD BE MEASURED.

This approach allows users to quickly investigate the impact on flow paths due to changes in the conductivity assigned to each layer/material, BASED ON THE MEASURED/KRIGED HEAD DISTRIBUTION. Clearly, the more accurately the head is characterized the better.

At this point, we don't propose to provide a mechanism to account for conductivity variations within a geologic layer. We obviously cannot account for natural or artificial barriers (low conductivity regions) UNLESS they are represented by the geologic materials.

Our approach is:

Compute the true seepage velocity (V_x , V_y , V_z) at each node, by taking the gradient of (kriged) head (without any z-exaggeration) and multiplying each component of head gradient by the component of conductivity at that node (based on its material) (K_x , K_y , K_z) and dividing by the Effective Porosity for that material.

$$V_x = dH/dx * K_x / Ne$$

$$V_y = dH/dy * K_y / Ne$$

$$V_z = dH/dz * K_z / Ne$$

regional_averages

The regional_average module averages nodal data values from the input field that fall into the input polygon regions. It then outputs a point for each region that contains the average x, y coordinates and the average of each selected nodal data component.

These polygons must contain at least 1 cell data component representing the regional ID.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Input Surface** [Field] Accepts a cell data field defining a regions

Module Output [Ports](#)

- **Output Field** [Field] Outputs the processed field.
- **Output Object** [Renderable]: Outputs to the viewer

modflow_converter

Note: This is unsupported technology. C Tech cannot provide tech support with issues of problems you may have with the use of this tool.

The modflow_converter module will import MODFLOW data from the following sources. For each, it is important that the reference files be present and any files referenced in those files also be in the same folder.

- MODFLOW98 :
 - Requires: *.BAS, *.BCF files
- MODFLOW2000
 - Requires *.NAM,*.DIS, *.BAS6 files
- GMS 8.1
 - Requires *.GPR file
- Visual MODFLOW
 - Requires *.VMG file
- Groundwater Vistas
 - Requires same files as MODFLOW

It will create time based EVS field files (.eff or .efz when Compress Files option is toggled) for the selected *Output Data*.

The MODFLOW simulation should be run to completion before the conversion is attempted.

For GMS and Visual MODFLOW the project files will contain links to files generated by the simulation.

Certain data will require additional files to be present.

For example head data will be created once the MODFLOW simulation is complete, usually generating a binary *.hed file.

This file should be referenced in the NAM file and then can be included in the generated EVS Field Files.

Concentration data is created when the MT3D simulation is run to successful completion and should be referenced in the NAM file as well.

modflow_converter

Note: This is unsupported technology. C Tech cannot provide tech support with issues of problems you may have with the use of this tool.

The modflow_converter module will import MODFLOW data from the following sources. For each, it is important that the reference files be present and any files referenced in those files also be in the same folder.

- MODFLOW98 :
 - Requires: *.BAS, *.BCF files
- MODFLOW2000
 - Requires *.NAM, *.DIS, *.BAS6 files
- GMS 8.1
 - Requires *.GPR file
- Visual MODFLOW
 - Requires *.VMG file
- Groundwater Vistas
 - Requires same files as MODFLOW

It will create time based EVS field files (.eff or .efz when Compress Files option is toggled) for the selected *Output Data*.

The MODFLOW simulation should be run to completion before the conversion is attempted.

For GMS and Visual MODFLOW the project files will contain links to files generated by the simulation.

Certain data will require additional files to be present.

For example head data will be created once the MODFLOW simulation is complete, usually generating a binary *.hed file.

This file should be referenced in the NAM file and then can be included in the generated EVS Field Files.

Concentration data is created when the MT3D simulation is run to successful completion and should be referenced in the NAM file as well.

Modpath DWR/DWZ File Example

Discussion of DWR/DWZ Files

DWR/DWZ files contain the package parameters from MODFLOW projects. These are stored as either a single record, for steady state simulations, or as a group of records based on date for transient simulations. The different packages supported are: DRAINS; WELLS; RECHARGE; ET (evapotranspiration); CONSTANT_HEAD; GENERAL_HEAD; RIVER_LEAKAGE; and STORAGE.

Format:

You may insert comment lines in DRW/DWZ input files. Comments can be inserted anywhere in a file and must begin with a '#' character. The line numbers that follow refer to all **non-commented** lines in the file.

Line 1: Should contain the word GRID.

Line 2: The GRID flag should be followed on the next line by the following grid parameters in order: the grid rotation about the z axis, the translation of the grid from the origin in the x direction, the translation of the grid from the origin in the y direction, the translation of the grid from the origin in the z direction.

Line 3: Should contain the word TIME followed on the same line by one of the following abbreviations indicating the time units of the file: "yr" – year, "d" – day, "h" – hours, "m" – minutes, "s" – seconds.

Line 4: Will contain either the word "STEADY_STATE" for steady state simulations, or the word "Date" followed by a date in the standard short date format.

Lines 5+: Should contain one of the package headers mention above (DRAINS,WELLS,etc..), followed on the subsequent lines with the coordinates of the center of the cell, the flow due to that package in that cell, and the face (if applicable) at which the flow is occurring. The faces are defined in the following order: top - 6, bottom - 5, right - 1, left - 2, front - 4, and back - 3. If the flow is not followed by a face number or is given a face number of 0 then the flow is applied to the entire cell and not to a cell face.

The word END on any line prevents further parsing of the file.

Steady State File example:

```
# EVS generated DWR file
GRID
0.000000 0.000000 0.000000 0.0
TIME d
STEADY_STATE
DRAINS
57500.000000 67500.000000 25.000000 -26006.757813 6.000000
WELLS
62500.000000 42500.000000 25.000000 100000.000000 0.000000
CONSTANT_HEAD
2500.000000 72500.000000 25.000000 -3394.514160
2500.000000 67500.000000 25.000000 -3415.331787
2500.000000 62500.000000 25.000000 -3453.412109
END
```

Transient File example:

```
# EVS generated DWR file
GRID
0.000000 0.000000 0.000000 0.0
TIME d
DATE 5/31/2146
DRAINS
57500.000000 67500.000000 25.000000 -26006.757813 6.000000
WELLS
```

```

62500.000000 42500.000000 25.000000 100000.000000 0.000000
CONSTANT_HEAD
2500.000000 72500.000000 25.000000 -3394.514160
2500.000000 67500.000000 25.000000 -3415.331787
2500.000000 62500.000000 25.000000 -3453.412109
DATE 10/28/2392
DRAINS
57500.000000 67500.000000 25.000000 -25082.052734 6.000000
WELLS
62500.000000 42500.000000 25.000000 90000.000000 0.000000
CONSTANT_HEAD
2500.000000 72500.000000 25.000000 -3022.231934
2500.000000 67500.000000 25.000000 -3042.281006
2500.000000 62500.000000 25.000000 -3079.266602

```

draw_lines

The draw_lines module enables you to create both 2D and 3D lines interactively with the mouse.

The mouse gesture for line creation is: depress the **Ctrl** key and then click the left mouse button on any pickable object in the viewer. The first click establishes the beginning point of the line segment and the second click establishes each successive point.

draw_lines allows adding of points that are outside the model extents, undoing of the last picked point, and the clearing of all picked points. Unlike most modules which create mesh data to be used by other modules, the draw_lines module receives input from the viewer, and also passes on field data to be used by other modules.

There are two drawing modes:

- 1) Top View Mode creates 2D lines which are always at Z=0.0. You must be in a Top View to draw with this mode, but you may pick points anywhere in the viewer screen.
- 2) Object Mode creates 3D lines which are drawn by probing objects in your model. You cannot draw at a point without having an object there or specifying a coordinate using the x-y-z type-ins.

NOTE: Because draw_lines saves your lines with your application, when an application is saved, the purple port is automatically disconnected from the viewer. This ensures that when you load an application the resulting objects (lines, fence-diagrams, etc.) will look exactly the same as when you saved the application. However, if you wish to draw new lines you will need to reconnect the purple port from the viewer.

Module Input [Ports](#)

- **View** [View] Connects to the viewer to receive the extent of all objects in the viewer for scaling lines and drawing on objects.

Module Output [Ports](#)

- **Output Field** [Field / minor] Outputs the field with the scaling and exploding applied.
- **Sample Data** [Renderable]: Outputs to the viewer.

polyline_spline

The polyline_spline module accepts a 3D polyline and can either increase or decrease the number of line segments of the polyline. A splining algorithm smooths the line trajectory once the number of points are specified. This module is useful for applications such as a fly over application (along a polyline path drawn by the user). If the user drawn line is jagged with erratically spaced line segments, polyline spline smooths the path and creates evenly spaced line segments along the path.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a 3D polyline field

Module Output [Ports](#)

- **Output Data** [Field] Outputs the splined lines
- **Output Object** [Renderable]: Outputs to the viewer

triangulate_polygons

triangulate_polygons converts a closed polyline into a triangulated surface. This surface can be extruded or used by the area_cut module to perform areal subsetting of 3D models.

Polylines with WIDTH in AutoCAD DWG files are converted by Read_CAD into triangle strips of the specified width. As you zoom in on polylines with width, the apparent width will change, whereas the apparent width of lines DOES NOT change. However, once they are triangles, they DO NOT define a closed area and therefore would not work with triangulate_polygons.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field representing closed polygon(s).

Module Output [Ports](#)

- **Output Field** [Field] Outputs the surface(s) field
- **Output Object** [Renderable]: Outputs to the viewer.

tri_tool

tri_tool is primarily for use with surf_cut. It can subdivide triangular and quadrilateral cells until none of the sides of the output triangles exceed a user specified length (a default value is calculated as 5% of the x-y extent of your input surface). This increases the accuracy of surf_cut especially when the input surface comes from scat_to_tin and the nodes used to create the TIN are poorly spaced. It can also correct the normals of a surface. It does this by organizing all of the triangles and quadrilaterals in a surface into disjoint patches, and then allowing the user to select which patches have normals that need to be flipped. The maximum number of triangles in a patch is 130,000, any triangles above this number will be considered to be in the next patch.

Removing small cells is used to remove extremely small cells (based on area in your coordinate units squared) that sometimes are generated with CAD triangulation routines that might have their normal vectors reversed and would contribute to poor cutting surface definition. Try this option if you find that surf_cut is giving anomalous results.

The maximum edge length allows the maximum length of each triangle side to be set for when the Split Cells option is set.

The ability to fix normals is used to check to that all of the triangles in selected patches of the surface have the same normal vector direction. If the normal is backwards, you can flip the normal of the patch in two ways. The first way is Alt + Right click on a cell in the patch that you wish to flip and then click the Add patch to flip list button. You only need to do this for one cell in each patch. Another way to do this is to set the Cell ID and Cell Data value of a cell in the patch you wish to flip. The Cell Id and Cell Data values must be obtained from the surface being output from tri_tool, and not the surface being input.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the refined grid.
- **Sample Data** [Renderable]: Outputs to the viewer.

tubes

The tubes module is used to produce open or closed tubes of constant or data dependent radius using 3D lines or polylines as input. Tube size, number of sides and data dependent coloring is possible.

Rotation of the tubes are done with the Phase slider (or type-in), which is specified in degrees. There are two methods used to maintain continuity of the tube orientation as the path meanders along a 3D path. These are specified as the Phase Determination method:

- **Force Z Up:** is the default and is most appropriate for paths that stay relatively horizontal. This option keeps the tube faces aligned with the Z axis and therefore with a slope of 30 degrees, the effective cross sectional area of the tube would be reduced by $\cos(30)$ which would be a 14% reduction. However for the typical slopes found with tunneling this effect is quite minimal and this option keeps the tube perfectly aligned.
- **Perpendicular Extrusions:** keeps the tube cross-section aligned with the tube (extrusion) path and therefore preserves the cross-section no matter what the path. However, tube rotation creep is possible.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field with or without data containing lines which represent the paths of the tubes.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field as tubes.
- **Output Object** [Renderable]: Outputs to the viewer.

volumetric_tunnel

The volumetric_tunnel module allows you to create a volumetric tunnel model that is defined by a polygonal surface cross-section along a complex 3D path. Once this volumetric grid is defined, it can be used as input to various modules to map analyte and/or geologic data onto the tunnel. These include:

- krig_3d: external grid port: to map analytical data
- indicator_geology: external grid port: to map lithologic data
- interp_data to map analytical data
- interp_cell_data: to map stratigraphic or lithologic material data

The requirements for the tunnel path and cross-section are:

- The path must be defined by a line input to the Right input port.
- The tunnel cross-section is defined by a surface input to the Left input port.
 - The cross-section should be defined in the X-Y plane at $Z = 0$ (2D)
 - The coordinates (size) of the cross-section should be actual scale in the same units as the tunnel path (generally feet or meters).
 - Do not use **cm** for cross-section and **meters** for path.
 - Generally, the X-Y Origin (0, 0) should lie within the cross-section and should represent where the tunnel path should be.

cross_section_tubes

The cross_section_tubes module is used to produce open or closed tubes of user defined cross-section and constant or data dependent radius using 3D lines or polylines as input for the centerline and a single 2D polyline as the cross-section of the tubes.

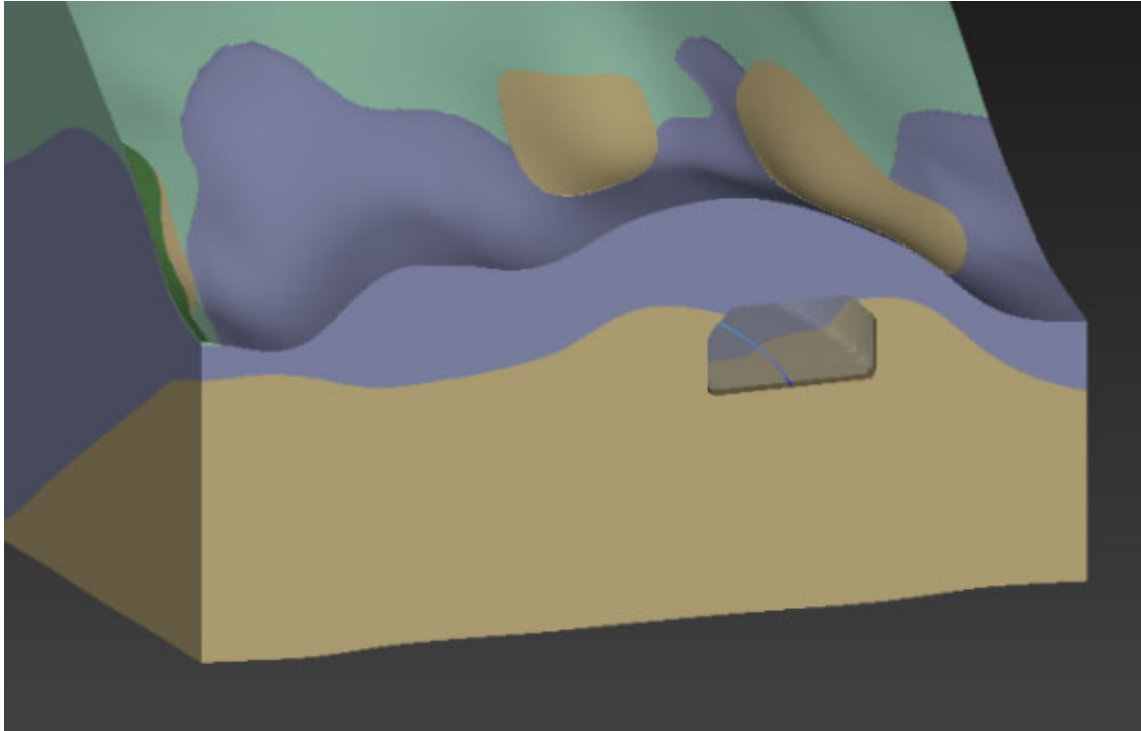
Module Input [Ports](#)

- **Input Field** [Field] Accepts a field with or without data containing lines which represent the paths of the tubes.
- **Input Cross Section Field** [Field] Accepts a field which has the cross-section of the tubes.

Rotation of the cross-section is done with the Phase slider (or type-in), which is specified in degrees. There are two methods used to maintain continuity of the tube orientation as the path meanders along a 3D path. These are specified as the Phase Determination method:

- Force Z Up: is the default and is most appropriate for paths that stay relatively horizontal. This option keeps the tube cross-section aligned with the Z axis and therefore with a slope of 30 degrees, the effective cross sectional area of the tube would be reduced by $\cos(30)$ which would be a 14% reduction. However for the typical slopes found with tunneling this effect is quite minimal and this option keeps the tube perfectly aligned.
- Perpendicular Extrusions: keeps the tube cross-section aligned with the tube (extrusion) path and therefore preserves the cross-section no matter what the path. However, cross-section rotation creep is possible.

The cross section field input must be a closed polyline that is drawn in the X-Y plane in the correct size. It should be balanced about the origin in X, usually with the Y axis ($X=0$) at the floor of the tunnel. This results in the tunnel being created such that the tunnel path will be at the centerline FLOOR of the tunnel as shown in the picture below.



This tube was created with an EVS Line File (.elf) that was very simple and is shown below:

```
LINE
-10 0 0
-10 7 0
-7 10 0
7 10 0
10 7 0
10 0 0
CLOSE
END
```

As you can see, all of the Z coordinates are zero since they are irrelevant. This shape is balanced about the Y axis and is all $Y \geq 0$

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as faces.
- **Output Object** [Renderable]: Outputs to the viewer.

extrude

The extrude module accepts any mesh and adds one to the dimensionality of the input by extruding the mesh in the Z direction. The interface enables changing the height scale for extruded cells and extruding by a constant, any nodal or cell data component. This module is often used with the [read_vector_gis](#) module to convert polygonal shapefiles into extruded volumetric cells.

When Node Data Component is chosen, the output cells will be extruded by the Scale Factor times the value of whichever nodal data component is selected on the right. With nodal data extrusion you must select "Positive Extrusions Only" or "Negative Extrusions Only". Since each node of a triangle or quadrilateral can have different values, it is possible for a single cell to have both positive and negative data values at its nodes. If this type of cell is extruded both directions, the cell topology can become tangled.

For this reason, nodal data extrusions must be limited to one direction. To extrude in both directions, merely use two extrude modules in parallel, one set to positive and the other to negative.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field with or without data

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the field
- **Output Object** [Renderable]: Outputs to the viewer

drive_glyphs

The drive_glyph module provides a way to move any object (glyph or object from Read_DXF, etc.) along multiple paths to create a "driving" animation.

Module Input Ports

drive_glyphs has three input ports.

Data passed to the first port is the paths to follow (normally from read_lines).

The second port accepts the glyph or vehicle to drive, usually read in with the [load_glyph](#) module.

The third port is a float parameter for the position of the glyphs.

Module Output Ports

drive_glyph has three output ports.

The leftmost output port is a float parameter for the position of the glyphs along the input paths.

The center port is the animated glyphs.

The right output port is the animated glyphs in a renderable form for the viewer.

place_glyph

General Module Function

The place_glyph module is used to place a single scalable geometric objects (glyph) at an interactively determined location.

glyph

The glyph module is used to place geometric objects (glyphs) at nodal locations. The glyphs can be scaled, rotated and colored based on the input data. If the input data is a vector, the glyph can be scaled and rotated to represent the direction and

absolute magnitude of the vector field. In a scalar data field, the objects can be scaled based on the magnitude of the scalar. The glyphs can represent the data field of one data component while being colored by another data component. Arrow glyphs are commonly used in vector fields to produce visualizations of the vector field.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field with scalar or vector data.
- **Input Glyph** [Field] Accepts a field representing the glyphs

Module Output [Ports](#)

- **Output Field** [Field] Outputs the glyphs
- **Output Object** [Renderable]: Outputs to the viewer.

create_fault_surface

The create_fault_surface module creates a 3D grid that is aligned to a specified strike and dip.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field to extract its extent

Module Output [Ports](#)

- **Z Scale** [Number] Outputs Z Scale (vertical exaggeration) to other modules
- **Output Field** [Field / Minor] Outputs the surface
- **Fault Surface** [Renderable]: Outputs to the viewer

create_grid

The create_grid module produces a 2D or 3D uniform grid that can be used for any purpose, however the primary application is as starting points for streamlines or advector. In 2D (default) mode it creates a rectangle of user adjustable grid resolution and orientation. In 3D mode it creates a box (3D grid of nodes).

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field to extract its extent

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the surface
- **Surface** [Renderable]: Outputs to the viewer

surfmap

surfmap provides a mechanism to drape lines onto surfaces. You should also investigate the geologic_surfmap module.

surfmap is similar to geologic_surfmap, but has one advantage and one disadvantage.

- The disadvantage is that data is not preserved
- The advantage is that lines are subsetting to match the size of the cells of the surface on which the lines are draped. In other words, draped lines will match the surface precisely.

Module Input [Ports](#)

- **Input Geologic Field** [Field] Accepts a geologic field
- **Input Lines** [Field] Accepts a field with the lines to be draped

Module Output [Ports](#)

- **Output Field** [Field] Outputs the draped lines
- **Surface** [Renderable]: Outputs the draped lines to the viewer.

transform_field

The transform_field module is used to translate, rotate or scale the coordinates any field. Uses for this module would be to rotate and translate a modflow or mt3d grid (having a grid origin of 0,0,0) to the actual coordinate system of the modeled area.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the transformed field.
- **Output Object** [Renderable]: Outputs to the viewer.

transform_group

transform_group is a special group object that allows all connected objects to be rotated (about a user defined center) and/or translated. This is useful if you wish to move objects that are complex, such as group objects like post_samples or axes and therefore cannot be contained in a single field (blue-black) port.

An example of this, would be the axes module. If you wanted an axes with an origin that did not match your data, it could be created separately and moved using the transform_group module.

Limitations

In some circumstances transform_group cannot be used with 4DIMs. It can cause the 4DIM extents to be different than they were in the EVS viewer. This has been noted when doing rotations.

In most cases, the [transform field](#) module can be used instead, however it does not allow for multiple objects to be connected to its input.

project_field

General Module Function

The project_field module is used to project the coordinates in any field, from one coordinate system to another.

Module Control Panel

The control panel for project_field is shown in the figure above.

Each coordinate system is divided into either Geographic or Projected coordinate systems. The coordinate system types are navigated by selecting the appropriate system type in the far left window. When a general coordinate system has been selected a specific coordinate system can be selected from the center window. If there are any details regarding the selected specific coordinate system, they will appear in the text window on the right. A specific coordinate system must be selected both to project from and to project to as in the picture below.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as faces.

overlay_aerial

The overlay_aerial module will take as input a field and then map an image onto the horizontal areas of the grid. The image can be projected from one coordinate system to another. It can also be georeferenced if it has an accompanying All vertical surfaces (Walls) can be included in the output but will not have image data mapped to them.

Note: If you need to georeference your image or adjust the georeferencing, you can do so with the *Georeference Image Tool* on the Tool Tab

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.
- **Filename** [String] The image filename

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetted field as faces.
- **Filename** [String] The image filename
- **Output Object** [Renderable]: Outputs to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the placement of the texture image
- Wall Properties: controls how walls are viewed
- Image Processing: allows for the alteration of the image brightness, contrast, etc.

Image Quality: This selector limits the max resolution of the image being read.

Most graphics cards support the **High** resolution of 2048, but relatively few support 4096 and only professional level cards and some of the newest DirectX 10 cards support 8192. Obviously higher resolution images will take more memory and more time to read, but will look much better when zoomed in.

Georeferencing Method: There are 8 different texture mapping modes as follows:

- 1) Map to Min/Max - Map image to the min/max extents of the input surface, or a user-defined value (can be typed into overlay_aerial directly).
- 2) Translate - Translate the image. Only requires a single GCP. No rotation or scaling is performed.
- 3) 2 pt: Trans./Rot. - Translate, Scale, and rotate the image. The image scaling is always the same in X&Y. Only a valid option if you have 2 GCP points. Good option if you only know 2 GCP points, and they are co-linear or near co-linear.
- 4) Translate/Scale - Translate and scale the image. Scale in X and Y are not the same. This keeps the image orthorectified. Can be used with 2 or more GCP points.
- 5) Affine - Perform a full affine transformation (1st order transformation) on the image. Requires a world file or 3 or more GCP points (from a gcp file). This is the default option which can be fully described with a World File.
- 6) 2nd Order - Perform a 2nd order polynomial transformation. This requires 6 or more GCP points (from a gcp file). It will map straight lines in the image into arcs. Allows an image that was georeferenced previously into LAT/LON coordinates to be "straightened" out and handled correctly. This can also be used to adjust for minor problems in the image due to topography. This option cannot be described with a World File because it uses a second order polynomial with more terms than are available in a world file. It requires the use of a GCP file.
- 7) 3rd Order - Perform a 3rd order polynomial transformation. Requires 10 or more GCP points. Allows you to adjust for drift in the image, "wedge" shaped photography, and more.
- 8) 4th Order - Perform a 4th order polynomial transformation. Requires 15 or more GCP points. Allows adjustments to be made where different portions of the image move in opposite directions. Requires many GCP points to use effectively.

Image Processing: These options allow for the adjustment of image brightness, sharpness, etc..

Image Projection Options: This toggle allows for the reprojection of the image. Each coordinate system is divided into either Geographic or Projected coordinate systems. The coordinate system types are navigated by selecting the appropriate system type in the far left window. When a general coordinate system has been selected a specific coordinate system can be selected from the center window. If there are any details regarding the selected specific coordinate system, they will appear in the text window on the right. A specific coordinate system must be selected both to project from and to project to, and then the Project Image toggle must be turned on.

texture_walls

General Module Function:

The texture_walls module provides a means to project an image onto surfaces such as walls of buildings to add more realism to your visualizations.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Object** [Renderable]: Outputs to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the placement and scale of the texture
- Image Processing: allows for the alteration of the image brightness, contrast, etc.

texture_geology

The texture_geology module will texture multiple images onto a field based on the geologic data in the field.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Object** [Renderable]: Outputs to the viewer.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Properties: controls the placement and scale of the textures
- Image Processing: allows for the alteration of the image brightness, contrast, etc.

georeferenced_output

This module will output a image in one of the following formats: BMP; TIF; JPG; and PNG. It will also output a world file that will allow the image to be placed correctly in applications that allow georeferencing.

Module Input [Ports](#)

- **Objects** [Renderable]: Receives one or more renderable objects similar to the viewer

read_image

read_image reads an image file of a specified type and converts it into an 2D uniform field. It can optionally perform some basic image manipulation upon the read image. It also automatically looks for a world file or GCP file associated with the image in the same folder. Since this is done automatically, the world file name must conform to industry standards.

The naming conventions for the world file require that the world file must have the same base name and the extension must be the first and last letter of the suffix followed by a 'w'. For example:

Image World file GCP file

801022_ot.jpg 801022_ot.jgw 801022_ot.gcp

coasth.png coasth.pgw coasth.gcp

sitecrop.tif sitecrop.tfw sitecrop.gcp

If your world or gcp file name does not conform to this standard, you need to rename the file.

World Files have six lines that should contain the following information:

FIRST LINE: The dimension of a pixel in map units in the x direction

SECOND LINE: 0 (the rotation factor for rows - in this case none)

THIRD LINE: the rotation factor for columns- in this case none)

FOURTH LINE: The dimension of a pixel in map units in the y direction. The y-scale is usually the negative value of the x-scale.

FIFTH LINE: The x coordinate of the center of the upper-left pixel in map units.

SIXTH LINE: The y coordinate of the center of the upper-left pixel in map units.

WARNING: If the image portion of a georeferenced file is cropped or resized, the corresponding World file must be appropriately edited.

The world file information is passed to the overlay_aerial module to georeference the image mapped on objects.

If you do not have a World file or GCP file, but are able to identify the coordinates (real world x-y) associated with objects (pixels) in your image, click on the

EditGeoreferencing button to open the standalone utility program [Georeference Image](#) which can create world files or .gcp (ground control point) files for images.

File types: The following image formats (with extensions) are currently supported:

- * Portable Network Graphics (png)
- * Windows Device Independent Bitmap 8/15/24 bit (bmp, rle, dib)
- * JPEG Joint Photographic Expert Group, JFIF 1.02 (jpg, jpeg, jif, jiff, jpe, J)
- * GIF Graphics Interchange Format (gif, giff)
- * TGA Truevision Targa 8/15/24/32 bit (tga)
- * TIFF Tagged Image File Format (tif, tiff)
- * XPM X-Pixmap (xpm)
- * BAY Bayer Image (bay)
- * FLC AutoDesk FLIC/FLIC-Pro, 8 bit (fli, flc)
- * PCX Zsoft Paintbrush, 8/24 bit (pcx)
- * PNM Portable Image 8/15/24 bit (pnm, ppm, pgm)
- * RAS Sun Raster Image 8/24/32 bit (ras, sun)
- * RGB Silicon Graphics Image 8/24/32 bit (rgb, rgba, bw, sgi)

AnimationFile types: Single frames can be read from any of the following animation formats (with extensions):

- * HAV High Quality Audio Video (hav)
- * AVI Windows Audio Video Interleaved (avi)
- * MPEG Motion Pictures Expert Group, Version 1 (mpg, mpeg, mpe, m1v)

fly_through

fly_through is an animation module which facilitates controlling the viewer or creating an animation in which the view follows a complex 3D path:

- on,
- through, or

- around your model.

The method by which this module controls fly-throughs allows the user to pause at any time and interact with the model using their mouse or the Az-Inc panel.

Az-Inc parameters (azimuth, elevation, scale, field of view, rotation/scaling center, etc.) are updated by fly_through in real time. This can be seen by running fly_through with the Az-Inc window open. However, please note that this will slow your animation substantially because of the need to continuously update the parameters in Az-Inc.

IMPORTANT NOTE: Be sure to **TURN OFF** "Animate viewer" in the Animator module if you're controlling fly_through with the Animator.

image_transition

image_transition receives two images, each coming from Read_Image and is used to transition from one image to another in any one of a variety of methods, including image fades and wipes.

The **Percent Transition to Right Image** slider adjusts how much of the right image will be visible. This slider is affected by the Immediate Mode toggle, which allows changes to the image to occur as the slider is changed in real time.

The Transition Method can be any of the following:

- **Fade to Right Image:** Fade from the left image to the right image, where any pixel is a percentage of the right image as specified by the slider.
- **Random Pixel Fade:** Replace a random percentage of pixels from the left image with ones from the right image, roughly following the percent transition slider. This is a reproducible random effect, so once a pixel appears as the right image, moving the slider to the right will never cause it to switch back to the left image.
- **White Noise Fade:** Pixels are randomly chosen from the right image, with the number of pixels in the right image roughly following the position of the slider. This method is non-reproducible, in that each time it is run, different pixels will be chosen. This is very similar to the above fade, but noticeably more "noisy".
- **Wipe from Left->Right:** Perform an image wipe, from the left edge of the image to the right edge, replacing the left image with the right.
- **Wipe from Top->Bottom:** Similar to above, except wipe from the top of the image down.
- **Wipe Rectangle:** Similar to above, except wipe outward from the center of the image in a rectangle following the aspect ratio of the original image.
- **Wipe Circle: Similar** to above, except wipe outward from the center of the image in a circle.
- **TheReverse Wipe Direction** toggle reverses the direction a wipe is performed. If Wipe from Left->Right is the current method, and this toggle is on, the wipe will be performed from right to left.

display_image

display_image: This module is a viewer for images. It should only be connected to Read_Image. When images are connected to the viewer (NOT RECOMMENDED), instead of display_image, each pixel is rendered as a quadrilateral. This is VERY inefficient.

texture_sphere

texture_sphere provides a means to (texture map) project images onto a sphere.

texture_cylinder

texture_cylinder provides a means to (texture map) project images onto a cylinder.

texture_cross_section

texture_cross_section allows you to apply images along a complex non-linear thin_fence (cross-section) path and compensate for the image scale and registration points at various points along the fence path.

This functionality provides the mechanism to accurately apply hand-drawn cross-sections to 3D fence diagrams. When combined in an application with [edit_horizons](#), texture_cross_section allows you to modify your 3D stratigraphic geology to accurately match your hand-drawn cross-sections.

load_eft

The load_eft module provides a mechanism to open saved OBJ file sets which require multiple files (geometry and textures) as a single file. This is required in order to [Package Files](#) which is a requisite step in the creation of EVS Presentations.

read_tcf

The read_tcf module is specifically designed to create models and animations of data that changes over time. This type of data can result from water table elevation and/or chemical measurements taken at discrete times or output from Groundwater simulations or other 3D time-domain simulations.

The read_tcf module creates a field using a Time Control File (.TCF) to specify the date/time, field and corresponding data component to read (in netCDF, Field or UCD format), for each time step of a time_data field. All file types specified in the TCF file must be the same (e.g. all netCDF or all UCD). The same file can be repeated, specifying different data components to represent different time steps of the output. read_tcf effectively includes internal interpolation between appropriate pairs of the files/data_components specified in the TCF file. Its internal structure only requires reading two successive time steps rather than the complete listing of time steps normally represented in a time_data field.

Module Input [Ports](#)

- **Date** [Number] Accepts a date

Module Output [Ports](#)

- **Start Date** [Number] Outputs the starting date
- **End Date** [Number] Outputs the ending date
- **Date** [Number] Output date
- **Output Field** [Field] Outputs the data field

TCF File Format and Example

The listing below is the full contents of the Time Control File **control_tce_cdf.tcf**. Blank lines or any lines beginning with a "#" are ignored. Valid lines representing time steps must be in order of ascending time and consisting of:

- a) a date and/or time in Windows standard format

b) a file name with an absolute path or just the filename (if the data files are in the same directory as the TCF file). **This is not a true relative path** (..\file.cdf and subdir\file.cdf don't work, but file.cdf does), but gives some of the relative path abilities.

c) the data component to use for that time step. (You can specify -1 in the third column, which causes ALL the data components to pass through.)

NOTE: These three items on each line **must** be separated with a comma ",".

```
# This file contains the list of control commands for the
# TCE time data in netCDF format.

# The format is a date/time, then the file, then the nodal data
component.
# The END on the last line is optional.

# Each line MUST be comma delimited
# (since spaces can exist in the time and filename)

6/1/1990 12:00 AM, $XP_PATH<0>/data/netcdf/time_data/tce_01.cdf, 0
12/1/1990, $XP_PATH<0>/data/netcdf/time_data/tce_02.cdf, 0
2/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_03.cdf, 0
5/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_04.cdf, 0
8/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_05.cdf, 0
11/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_06.cdf, 0
3/1/1992, $XP_PATH<0>/data/netcdf/time_data/tce_07.cdf, 0
6/1/1992, $XP_PATH<0>/data/netcdf/time_data/tce_08.cdf, 0
10/1/1992, $XP_PATH<0>/data/netcdf/time_data/tce_09.cdf, 0
3/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_10.cdf, 0
4/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_11.cdf, 0
8/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_12.cdf, 0
12/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_13.cdf, 0
3/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_14.cdf, 0
6/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_15.cdf, 0
9/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_16.cdf, 0
11/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_17.cdf, 0
3/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_18.cdf, 0
5/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_19.cdf, 0
8/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_20.cdf, 0
10/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_21.cdf, 0
1/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_22.cdf, 0
5/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_23.cdf, 0
9/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_24.cdf, 0
11/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_25.cdf, 0
12/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_26.cdf, 0
3/1/1997 12:00 AM, $XP_PATH<0>/data/netcdf/time_data/tce_27.cdf, 0
```

```

6/1/1997, $XP_PATH<0>/data/netcdf/time_data/tce_28.cdf, 0
9/1/1997, $XP_PATH<0>/data/netcdf/time_data/tce_29.cdf, 0
12/1/1997, $XP_PATH<0>/data/netcdf/time_data/tce_30.cdf, 0
3/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_31.cdf, 0
6/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_32.cdf, 0
9/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_33.cdf, 0
11/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_34.cdf, 0
5/1/1999, $XP_PATH<0>/data/netcdf/time_data/tce_35.cdf, 0
10/1/1999, $XP_PATH<0>/data/netcdf/time_data/tce_36.cdf, 0
3/1/2000, $XP_PATH<0>/data/netcdf/time_data/tce_37.cdf, 0
7/1/2000, $XP_PATH<0>/data/netcdf/time_data/tce_38.cdf, 0
11/1/2000, $XP_PATH<0>/data/netcdf/time_data/tce_39.cdf, 0
3/1/2001, $XP_PATH<0>/data/netcdf/time_data/tce_40.cdf, 0
5/1/2001, $XP_PATH<0>/data/netcdf/time_data/tce_41.cdf, 0
10/1/2001, $XP_PATH<0>/data/netcdf/time_data/tce_42.cdf, 0

```

END

read_multi_tcf

The read_multi_tcf module is one of a limited set of Time_Data modules. These modules are specifically designed to create models and animations of data that changes over time. This type of data can result from water table elevation and/or chemical measurements taken at discrete times or output from Groundwater simulations or other 3D time-domain simulations.

The read_multi_tcf module creates a field using one or more Time Control Files (.TCF). [Click here for an example of a TCF file and a description of the format.](#)

The read_multi_tcf module creates a mesh grid with the interpolated data from a user specified number of TCF files (n). It outputs the first data component from the first (n-1) TCF files and all of the time interpolated data components from the nth TCF file.

For example, if you were trying to create a time animation of the union of 3 analytes (e.g. Benzene, Toluene & Xylene), read_multi_tcf allows you to select all three separate TCF files. Only the first data component from Benzene.tcf (nominally the concentration of benzene) is output as the new first data component. The first data component from Toluene.tcf (nominally the concentration of toluene) is output as the new second data component. All of the data components from Xylene.tcf are then output (typically xylene, confidence_xylene, uncertainty_xylene, Geo_Layer, Material_ID, Elevation, etc.). This allows you to explode layers and do other typical subsetting and processing operations on the output of this module.

The TCF files should be created using identical grids with date ranges that overlap the time period of interest.

read_multi_tcf effectively includes an inter_time_step module internally in that it performs the interpolation between appropriate pairs of the files/data_components specified in the TCF file. Its internal structure only requires reading two successive time steps rather than the complete listing of time steps normally represented in a time_data field.

TCF File Format and Example

The listing below is the full contents of the Time Control File ***control_tce_cdf.tcf***. Blank lines or any lines beginning with a "#" are ignored. Valid lines representing time steps must be in order of ascending time and consisting of:

- a) a date and/or time in Windows standard format
- b) a file name with an absolute path or just the filename (if the data files are in the same directory as the TCF file). **This is not a true relative path** (..\file.cdf and subdir\file.cdf don't work, but file.cdf does), but gives some of the relative path abilities.
- c) the data component to use for that time step. (You can specify -1 in the third column, which causes ALL the data components to pass through.)

NOTE: These three items on each line **must** be separated with a comma ",".

```
# This file contains the list of control commands for the
# TCE time data in netCDF format.

# The format is a date/time, then the file, then the nodal data
# component.
# The END on the last line is optional.

# Each line MUST be comma delimited
# (since spaces can exist in the time and filename)

6/1/1990 12:00 AM, $XP_PATH<0>/data/netcdf/time_data/tce_01.cdf, 0
12/1/1990, $XP_PATH<0>/data/netcdf/time_data/tce_02.cdf, 0
2/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_03.cdf, 0
5/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_04.cdf, 0
8/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_05.cdf, 0
11/1/1991, $XP_PATH<0>/data/netcdf/time_data/tce_06.cdf, 0
3/1/1992, $XP_PATH<0>/data/netcdf/time_data/tce_07.cdf, 0
6/1/1992, $XP_PATH<0>/data/netcdf/time_data/tce_08.cdf, 0
10/1/1992, $XP_PATH<0>/data/netcdf/time_data/tce_09.cdf, 0
3/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_10.cdf, 0
4/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_11.cdf, 0
8/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_12.cdf, 0
12/1/1993, $XP_PATH<0>/data/netcdf/time_data/tce_13.cdf, 0
3/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_14.cdf, 0
6/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_15.cdf, 0
9/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_16.cdf, 0
11/1/1994, $XP_PATH<0>/data/netcdf/time_data/tce_17.cdf, 0
3/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_18.cdf, 0
5/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_19.cdf, 0
8/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_20.cdf, 0
10/1/1995, $XP_PATH<0>/data/netcdf/time_data/tce_21.cdf, 0
1/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_22.cdf, 0
```

```

5/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_23.cdf, 0
9/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_24.cdf, 0
11/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_25.cdf, 0
12/1/1996, $XP_PATH<0>/data/netcdf/time_data/tce_26.cdf, 0
3/1/1997 12:00 AM, $XP_PATH<0>/data/netcdf/time_data/tce_27.cdf, 0
6/1/1997, $XP_PATH<0>/data/netcdf/time_data/tce_28.cdf, 0
9/1/1997, $XP_PATH<0>/data/netcdf/time_data/tce_29.cdf, 0
12/1/1997, $XP_PATH<0>/data/netcdf/time_data/tce_30.cdf, 0
3/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_31.cdf, 0
6/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_32.cdf, 0
9/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_33.cdf, 0
11/1/1998, $XP_PATH<0>/data/netcdf/time_data/tce_34.cdf, 0
5/1/1999, $XP_PATH<0>/data/netcdf/time_data/tce_35.cdf, 0
10/1/1999, $XP_PATH<0>/data/netcdf/time_data/tce_36.cdf, 0
3/1/2000, $XP_PATH<0>/data/netcdf/time_data/tce_37.cdf, 0
7/1/2000, $XP_PATH<0>/data/netcdf/time_data/tce_38.cdf, 0
11/1/2000, $XP_PATH<0>/data/netcdf/time_data/tce_39.cdf, 0
3/1/2001, $XP_PATH<0>/data/netcdf/time_data/tce_40.cdf, 0
5/1/2001, $XP_PATH<0>/data/netcdf/time_data/tce_41.cdf, 0
10/1/2001, $XP_PATH<0>/data/netcdf/time_data/tce_42.cdf, 0

```

END

time_value

The time_value module is used to parse a TVF file consisting of dates, values, and (optional) labels. The starting and end dates are read from the file and the controls can be used to interpolate the values to the date and time of interest.

Module Input [Ports](#)

- **Date** [Number] Accepts a date

Module Output [Ports](#)

- **Start Date** [Number] Outputs the starting date
- **End Date** [Number] Outputs the ending date
- **Date** [Number] Output date
- **Current Date and Time Label** [String] Resulting string for the output date
- **Current Date and Time Value** [Number] Resulting value for the output date

TVF File Format

TVF files provide a way to generate a time varying numeric and option string (label). The file is similar to the TCF file, but does not reference information in external files. The file consists of two or more rows, each having 2 or 3 columns of information. The columns must contain:

1. Date and/or time in Windows standard format
2. A numeric (float) value (required)

3. A string consisting of one or more words. These need not be in quotes. Everything on the row after the numeric value will be used. (optional)

Dates must be in order from earliest to latest and not repeating. Only the label column is optional.

An example file follows:

06/01/12	-1.63	Spring Rains
06/04/12	-1.87	
06/07/12	-2.17	
06/10/12	-1.87	
06/13/12	-1.9	
06/16/12	-2.2	
06/19/12	-1.9	
06/22/12	-1.96	Summer
06/25/12	-1.81	
06/28/12	-1.84	
07/01/12	-1.69	
07/04/12	-1.39	
07/07/12	-1.33	
07/10/12	-1.12	
07/13/12	-0.85	
07/16/12	-1.03	
07/19/12	-1.06	
07/22/12	-0.76	
07/25/12	-0.61	Flood Event
07/28/12	-0.31	
07/31/12	-0.31	

08/03/12	-0.52	
08/06/12	-0.37	
08/09/12	-0.61	
08/12/12	-0.85	
08/15/12	-0.79	
08/18/12	-0.76	
08/21/12	-0.58	
08/24/12	-0.64	
08/27/12	-0.49	
08/30/12	-0.46	
09/02/12	-0.67	
09/05/12	-0.91	
09/08/12	-0.82	
09/11/12	-1.09	""
09/14/12	-1.27	
09/17/12	-1.3	
09/20/12	-1.33	
09/23/12	-1.51	Fall
09/26/12	-1.42	
09/29/12	-1.69	
10/02/12	-1.69	
10/05/12	-1.78	
10/08/12	-1.84	
10/11/12	-1.96	
10/14/12	-2.17	
10/17/12	-2.29	

10/20/12	-2.26	
10/23/12	-2.05	
10/26/12	-2.05	
10/29/12	-1.84	
11/01/12	-2.05	
11/04/12	-2.23	
11/07/12	-2.08	
11/10/12	-2.2	
11/13/12	-2.41	
11/16/12	-2.62	
11/19/12	-2.83	
11/22/12	-2.62	
11/25/12	-2.5	
11/28/12	-2.29	
12/01/12	-2.11	
12/04/12	-2.2	
12/07/12	-1.9	
12/10/12	-2.08	
12/13/12	-1.93	
12/16/12	-1.81	
12/19/12	-1.75	
12/22/12	-1.63	Winter
12/25/12	-1.36	
12/28/12	-1.45	
12/31/12	-1.24	
01/03/13	-1.21	

01/06/13	-1	
01/09/13	-1.27	
01/12/13	-1.21	
01/15/13	-1.18	
01/18/13	-1.15	
01/21/13	-1.12	
01/24/13	-1.33	
01/27/13	-1.39	
01/30/13	-1.24	
02/02/13	-1.3	
02/05/13	-1.57	
02/08/13	-1.66	
02/11/13	-1.81	
02/14/13	-1.69	
02/17/13	-1.78	
02/20/13	-1.78	
02/23/13	-1.84	
02/26/13	-1.72	
03/01/13	-2.02	
03/04/13	-2.23	
03/07/13	-2.08	
03/10/13	-2.02	
03/13/13	-2.32	
03/16/13	-2.11	
03/19/13	-2.41	
03/22/13	-2.65	Spring

03/25/13	-2.38
03/28/13	-2.47
03/31/13	-2.47
04/03/13	-2.32
04/06/13	-2.17
04/09/13	-2.14
04/12/13	-2.41
04/15/13	-2.65
04/18/13	-2.47
04/21/13	-2.35
04/24/13	-2.32
04/27/13	-2.38
04/30/13	-2.08
05/03/13	-1.93
05/06/13	-1.84
05/09/13	-1.57
05/12/13	-1.84
05/15/13	-1.57
05/18/13	-1.57
05/21/13	-1.69
05/24/13	-1.93
05/27/13	-1.78
05/30/13	-1.57
06/02/13	-1.84

time_geology

The time_geology module allows you to extract a surface from a set of time-based surfaces. The time for the extracted surface can be any time between the start and end of the surface set. It will interpolate between adjacent known times.

time_loop

General Module Function

The time_loop module is one of a limited set of Time_Data modules. These modules are specifically designed to create models and animations of data that changes over time. This type of data can result from water table elevation and/or chemical measurements taken at discrete times or output from Groundwater simulations or other 3D time-domain simulations.

The time_loop module allows you to loop through a series of times or specify a time for interpolation from a time field.

group_object

group_object is a renderable object that contains other subobjects that have the attributes that control how the rendering is done. Unlike DataObject, group_object does not include data. Instead, it is meant to be a node in the rendering hierarchy that groups other DataObjects together and supplies common attributes from them. This object is connected directly to one of the viewers (for example, Simpleviewer3D) or to another DataObject or to group_object. A group_object is included in all the standard viewers provided with the EVS applications chooses.

group_object combines the following:

- * DefaultDatamap to convert scalar node or cell data to RGB color values. By default, the datamap's minimum and maximum values are 0 and 255, respectively. This datamap is inherited by any children objects if they do not have their own datamaps.
- * DefaultProps to control color, material, line attribute, and geometrical attributes.
- * DefaultModes to control point, line, surface, volume, and bounds rendering modes.
- * DefaultPickInfo to contain information when this object is picked.
- * DefaultObject to control visibility, pickability, caching, transform mode, surface conversion, and image display attributes.

2d_overlay_group

2D_Overlay provides a module that applies any connected module's output to the viewer's 2D overlay. Objects in the overlay are not transformed (rotated, zoomed, panned). These objects are locked in position. This provides a mechanism to apply graphics like drawing title blocks or company logos.

However, you must ensure that the object sent to the 2D overlay fits inside its limited spatial extent. The 2D overlay is a window with an x-extent from -1.0 to 1.0. The y-extent is dependent on the aspect ratio of the viewer. With a default viewer having a 4:3 aspect ratio, it is three-quarters of the x-extent (e.g. -0.75 to 0.75).

trigger_script

The trigger_script module provides a powerful way to link parameters and actions of multiple modules. This gives you the ability for a sequence of events to be "triggered" as the result of one or more parameters changing.

The module requires a Python script be created, which runs when you "Add" triggers. Triggers are module parameters that might change and thereby cause the script to be run. The script can do just about ANYTHING.

In addition to the Triggers that you specify, there are 4 input (and output) ports that accept numbers (such as a plume level) that can be used in your script, and are more readily accessible without accessing the Python script.

Module Input & Output [Ports](#)

- **N1** [Number] Accepts a number
- **N2** [Number] Accepts a number
- **N3** [Number] Accepts a number
- **N4** [Number] Accepts a number

merge_fields

merge_fields combines the input fields from up to 4 separate inputs into a unified single field with any number of nodal data components, which can be output to virtually any filter or mapper module, OR directly to the viewer. This is useful when you want to slice through or otherwise subset multiple fields using the same criteria or object.

Module Input [Ports](#)

- **First Input Field** [Field] Accepts a data field.
- **Second Input Field** [Field] Accepts a data field.
- **Third Input Field** [Field] Accepts a data field.
- **Fourth Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with all inputs merged
- **Output Object** [Renderable]: Outputs to the viewer.

float_math

This module provides a simple means to perform mathematical operations on numbers coming from up to 4 input ports. By using multiple float_math modules, any number of values may be combined.

The panel for float_math is shown above. The default equation is $f1 + f2 + f3 + f4$ which adds all four input ports.

Pop-up Available Mathematical Operators here or [Jump to Available Mathematical Operators here](#)

Any of these operators may be used.

The output (rightmost output port) is the numeric value resulting from the equation. The value will update when any of the input values are changed unless the checkbox next to the input value is turned off.

Module Input [Ports](#)

- **Input Value 1** [Number] Accepts number 1
- **Input Value 2** [Number] Accepts number 2

- **Input Value 3** [Number] Accepts number 3
- **Input Value 4** [Number] Accepts number 4
- **Input String 1** [String] An input string

Module Output [Ports](#)

- **Output Value1** [Number] Outputs number 1
- **Output Value 2** [Number] Outputs number 2
- **Output Value 3** [Number] Outputs number 3
- **Output Value 4** [Number] Outputs number 4
- **Output String 1** [String] An input string
- **Result Value** [Number] The final output

scat_to_tin

The scat_to_tin module is used to convert scattered sample data into a three-dimensional surface of triangular cells representing an unstructured mesh.

"Scattered sample data " means that there are discrete nodes in space. An example would be geology or analyte (e.g. chemistry) data where the coordinates are the x, y, and elevation of a measured parameter. The data is "scattered" because there is not necessarily an implicit grid of data.

scat_to_tin uses a proprietary version of the Delaunay tessellation algorithm.

Module Input [Ports](#)

- **Input Points** [Field] Accepts a data field of points or uses the nodes (points) from lines

Module Output [Ports](#)

- **Output Field** [Field] Outputs the surface data field
- **Output Object** [Renderable]: Outputs to the viewer.

scat_to_unif

The scat_to_unif module is used to convert scattered sample data into a three-dimensional uniform field. Also, scat_to_unif can be used to take an existing grid (for example a UCD file) and convert it to a uniform field. scat_to_unif converts a field of non-uniformly spaced points into a uniform field which can be used with many of EVS's filter and mapper modules. "Scattered sample data " means that there are disconnected nodes in space. An example would be geology or analyte (e.g. chemistry) data where the coordinates are the x, y, and elevation of a measured parameter. The data is "scattered" because there isn't data for every x/y/elevation of interest.

scat_to_unif lets you define a uniform mesh of any dimensionality and coordinate extents. It superimposes the input grid over this new grid that you have defined. Then, for each new node, it searches the input grid's neighboring original nodes (where search_cube controls the depth of the search) and creates data values for all the nodes in the new grid from interpolations on those neighboring actual data values. You can control the order of interpolation and what number to use as the NULL data value should the search around a node fail to find any data in the original input.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field

Module Output [Ports](#)

- **Output Data** [Field] Outputs the volumetric uniform data field

material_to_cellsets

material_to_cellsets is intended to receive a 3D field into its input port which has been processed through a module like plume. If the original field (pre-plume) had multiple cell sets related to geologic units or materials the output of plume will generally have only two cell sets which comprise all hexahedron and all tetrahedron cells. The ability to control the visibility of the layer-cell sets is normally lost.

This module takes plume's output and recreates the cell sets based on nodal data. However, since each geologic layer will likely have two cell sets each (one for all hexahedron and all tetrahedron cells), the output tends to have twice as many cell sets as the original pre-plume field).

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the processed field.

loop

The loop module iterates an operation. For example, you could use a loop object to control the movement of an object in your application; such as incrementing the movement of a slider for a slice plane.

modify_data_3d

The modify_data_3d module provides the ability to interactively change data in 3D volumetric models. This is not a recommended practice since volumetric models created in EVS generally have underlying statistical measures of quality that will be meaningless if the data is modified in any way.

However, it is not unusual for a model to occasionally have regions where extrapolation artifacts cause shards of plumes to appear. This module provides a way to remove those.

The basic approach is to move the modification sphere to the problem region and set the size and shape of the ellipsoid before changing your data.

Module Input [Ports](#)

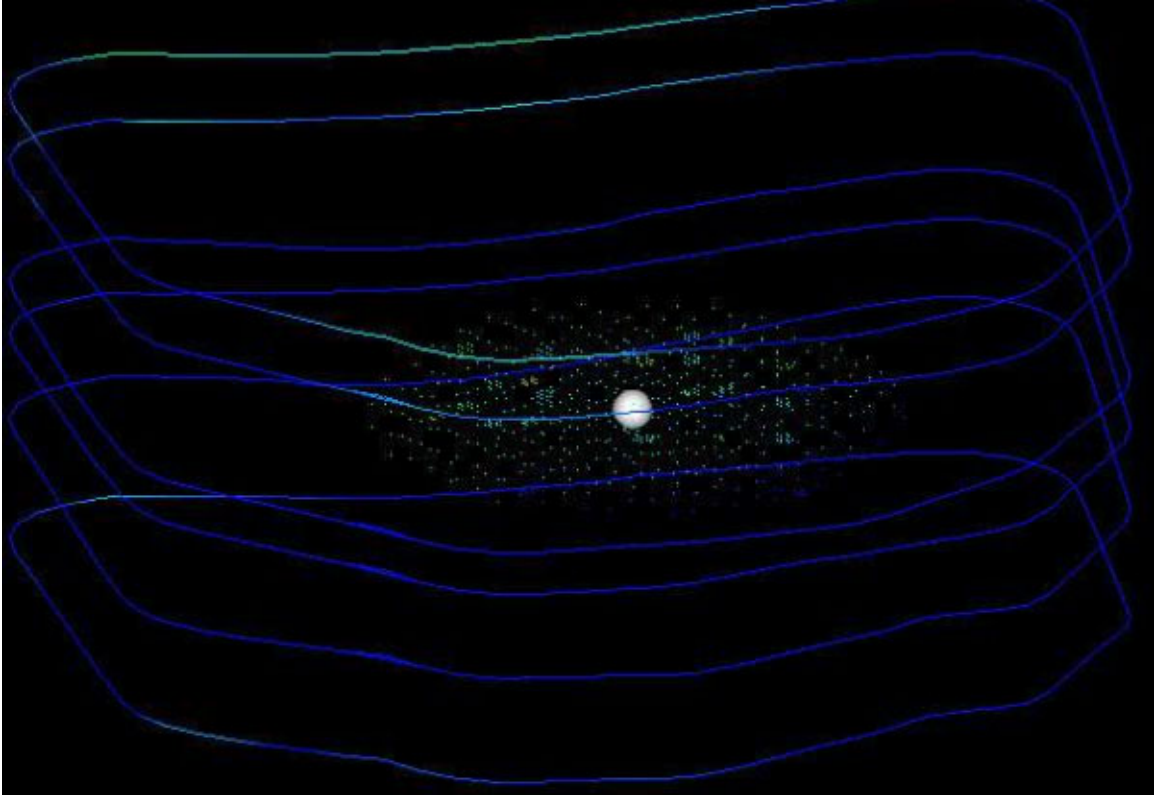
- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a data field from krig_3d or other similar modules.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with modified data
- **Sample Data** [Renderable]: Outputs to the viewer

The figure below shows the cloud of points display from this module. Note the adaptively gridded regions with clusters of nodes!

Note: This module does not modify the upstream data.



cell_computation

The cell_computation module is used to perform mathematical operations on cell data in fields. Unlike node_computation, it cannot affect coordinates.

Though data values can't be used to affect coordinates (x, y, or z), the cell center (average of nodes) coordinates can be used to affect data values.

Up to two fields can be input to cell_computation. Mathematical expressions can involve one or both of the input fields.

Cell data input to each of the ports is scalar.

If a data field contains more than one data component, you may select from any of them.

Module Input [Ports](#)

- **Input Field 1** [Field] Accepts a data field.
- **Input Field 2** [Field / minor] Accepts a data field.
- **Input Value N1** [Number / minor] Accepts a number to be used in the field computations.
- **Input Value N2** [Number / minor] Accepts a number to be used in the field computations.
- **Input Value N3** [Number / minor] Accepts a number to be used in the field computations.

- **Input Value N4** [Number / minor] Accepts a number to be used in the field computations.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the subsetting field as faces.
- **Output Value N1** [Number / minor] Outputs a number used in the field computations.
- **Output Value N2** [Number / minor] Outputs a number used in the field computations.
- **Output Value N3** [Number / minor] Outputs a number used in the field computations.
- **Output Value N4** [Number / minor] Outputs a number used in the field computations.
- **Output Object** [Renderable]: Outputs to the viewer.

Each cell data component from Input Field 1 is assigned as a variable to be used in the script. For example:

- An0 : First input data component
- An1 : Second input data component
- An2 : Third input data component
- An* : Nth input data component

The min and max of these components are also added as variables :

- Min_An0 : Minimum of An0 data
- Max_An0 : Maximum of An0 data
- Min_An* : Minimum of An* data

For Input Field 2 the variable names change to:

- Bn0 : First input data component
- Bn1 : Second input data component
- Bn2 : Third input data component
- Bn* : Nth input data component

cell_to_node

The cell_to_node module is used to translate cell data components to nodal data components. Cell data components are data components which are associated with cells rather than nodes. Most modules in EVS that deal with analytical or continuum data support node based data. Therefore, cell_to_node can be used to translate cell based data to a nodal data structure consistent with other EVS modules.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field with cell data.

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the field with cell data converted to nodal data
- **Output Object** [Renderable]: Outputs to the viewer.

node_to_cell

The node_to_cell module is used to translate nodal data components to cell data components. Cell data components are data components which are associated with cells rather than nodes. Most modules in EVS that deal with analytical or continuum data support node based data, and those that deal with geology (lithology) tend to use cell data. Therefore, node_to_cell can be used to translate nodal data to cell data.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field with nodal data.

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the field with nodal data converted to cell data
- **Output Object** [Renderable]: Outputs to the viewer.

shrink_cells

The shrink_cells module produces a mesh containing disjoint cells which can be optionally shrunk relative to their geometric centers. It creates duplicate nodes for all cells that share the same node, making them disjoint. If the shrink cells toggle is set, the module computes new coordinates for the nodes based on the specified shrink factor (which specifies the scale relative to the geometric centers of each cell). The shrink factor can vary from 0 to 1. A value of 0 produces non-shrunk cells; 1 produces completely collapsed cells (points). This module is useful for separate viewing of cells comprising a mesh.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the field with modified cells
- **Output Object** [Renderable]: Outputs to the viewer

cell_centers

cell_centers module produces a mesh containing Point cell set, each point of which represents a geometrical center of a corresponding cell in the input mesh. The coordinates of cell centers are calculated by averaging coordinates of all the nodes of a cell. The number of nodes in the output mesh is equal to number of cells in the input mesh. If the input mesh contains Cell_Data it becomes a Node_Data in the output mesh with each node values equal to corresponding cell value. **Nodal data is not output directly.** You can use this module to create a position mesh for the glyph module. You may also use this module as mesh input to the [interp_data](#) module, then send the same nodal values as the input grid, to create interpolated nodal values at cell centroids.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field.

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the field as points representing the centers of the cells.
- **Output Object** [Renderable]: Outputs to the viewer.

interp_cell_data

The interp_cell_data module interpolates cell data from one field to another using a Nearest Neighbor interpolation. Typical uses of this module are mapping of cell data from a 3D mesh onto a geologic surface or a 2D fence section. In these applications the 2D surface(s) simply provide the new geometry (mesh) onto which the adjacent cell values are interpolated.

Module Input [Ports](#)

- **Input Cell Data Field** [Field] Accepts a field with cell data.
- **Input Destination Field** [Field] Accepts a field onto which the data will be interpolated

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field Destination Field with new data
- **Output Object** [Renderable]: Outputs to the viewer.

viewer

The viewer accepts renderable objects from all modules with red output ports to include their output in the view.

Module Input [Ports](#)

- **Objects** [Renderable]: Receives renderable objects from any number of modules

Module Output [Ports](#)

- **View** [View / minor] Outputs the view information used by other modules to provide all model extents or interactivity

viewer Properties:

The user interfaces for the viewer are arranged in 10 categories which cover interaction with the scene, the characteristics of the viewer as well as various output options. The categories are:

1. **Properties:** includes the ability to set the view (Azimuth, Inclination, Scale, Perspective, etc.), pick objects and probe their data and control how the view scale reacts as new objects or data are added to the scene.

2. Window Size: sets the size of the viewer. The view has apparent size (the size of the visible window) and the true image size. Outputting a high resolution image involves setting a true image size to match your desired output dimensions.
3. [Output Image](#): includes the ability to export the view in PNG, BMP, JPG, or TIF format. Additional view scaling options are included.
4. Distance Tool: provides an interactive means to measure the distance between points in the viewer's scene and to export the line between two points in C Tech's ELF format.
5. Background: sets the style and colors for the background.
 1. The default, 2 color background will be saved in 4DIMs and will display in all output.
 2. Use *Unlocked Background* for VRML output. Please note that Unlocked Backgrounds are not inherited in a 4DIM and therefore the background can be changed.
6. View: provides controls for stereo viewing, updating and depth sorting.
7. Lights: provides the ability to control one or more lights in the scene and their properties.
8. Camera: provides detailed controls over the camera's interaction with the scene of objects.
9. Record 4DIM: provides the ability to [export the scene in C Tech 4DIM format](#).
10. Write_VRML: provides the ability to export the scene as [VRML for conversion to 3DPDF](#) or [3D printing](#).

Object Manipulation in the viewer

When the viewer is instanced, it opens a window in which objects connected to the viewer are rendered and can be manipulated. Objects can be transformed and scaled in the viewer window by using combinations of mouse actions and various keys on the keyboard.

- Rotation of objects in the viewer is accomplished by clicking and dragging on any portion of the viewer window with the left mouse button.
- Translation of objects in the viewer is accomplished by clicking and dragging on any portion of the viewer window with the right mouse button.
- Zooming of an object in the viewer is accomplished using the mouse wheel. Alternatively by depressing the Shift button while clicking and dragging the middle-mouse towards the upper right to zoom IN or lower left to zoom OUT.

Output Images

The View Scale parameter allows you to specify that your image to be output will be "n" times larger (or smaller if a fraction less than 1.0 is specified) than your current Window Size

When the **Autoscale FF Font** toggle is selected all Forward Facing fonts in the image will be scaled depending upon the size of the output image.

The *suffix* specified for the Image Filename determines the type of output.

- For PNG (portable network graphics), a compression slider is provided. The max value of 9 results in a very small increase in compute time for compressing the images. Since PNG is a LOSSLESS compression format, the quality of the image is not affected by this value.
- For JPEG, a quality parameter is provided. Higher qualities result in less LOSS to the image but create much larger files. We recommend using PNG instead of JPEG whenever possible. The PNG images are often smaller and are always higher quality than a JPEG image.

The Anti Aliasing option renders an image that is twice as big as the specified Width and Height. This high resolution image is then filtered and subsetting to the specified size. This process reduces the brightness (contrast) of fine lines but it also smooths the lines and dramatically reduces jaggies.

The *Mask Background* toggle allows you to create an image with a transparent background. In order to accomplish this, several things must be done:

- You must specify an image type that supports transparent backgrounds. PNG is recommended
- You must have a background color which is unique from any pixels in your objects which are rendered. This can be somewhat difficult if you have a rendered object with shading and specular highlights. Shading creates darker versions of the colors in your datamap and specular highlights creates less saturated (more white) versions of those colors. To avoid creating object colors that match your background, a masking background color should be selected which has a unique HUE not found in your datamap.
- Anti-Aliasing and filtering will intelligently detect the edges that are transparent and not mix in "pink" edges on your objects.

NOTE: There is no tolerance for matching the background color. The color must match the RGB value exactly.

TIP: The mask background function can be used to create transparent HOLES in your images. For example, a lake, which is rendered as a unique color could become a transparent hole in your rendered output. In order to accomplish this, the object which represents the lake must be colored to exactly match your mask color and it must have its surface rendering set to "Flat Shading". The **Select File** button is used to bring up a standard windows file browser for choosing the name and location of the file to create. The Accept Current Values push button begins creation of the file.

write_vrml

The write_vrml output in the viewer is able to output most graphics objects in the viewer to a VRML-formatted file.

VRML is a network transparent protocol for communicating 3D graphics. It has fallen out of favor on the web, though it is still a standard for 3D model output.

We provide VRML output for two primary purposes:

1. Export of 3D models for conversion to [3D PDF](#)
2. Export of 3D models for [3D Printing](#)

Known Issues

- Turn on the "Use Unlocked Background" option in the viewer->Background editor when writing VRML files, since the background is otherwise rendered as a small square at the origin.
- Always set your viewer to a Top View (180 Azimuth and 90 Inclination) before writing the VRML file.
- Do not use any modules which display in the 2D overlay. The 2D overlay is analogous to drawing on the glass on a TV or monitor. Items in the 2D overlay do not move, rotate or scale when you manipulate your 3D model. Examples are add_logo, Titles, and legend.
- Do not use volume rendering. These techniques are not supported.
- VRML does not support the full spectrum of data coloring supported in EVS.
 - Though both cell and nodal data coloring is supported, sometimes combinations of these cause problems.
 - Object colors (such as the red, blue, green grid lines of the axes module) often revert to white (uncolored). This can be problematic on a white background.
 - The [texture_colors](#) module is recommended for final output of most all colored objects to help avoid these issues.
- Trial and Error is often the only way to determine what combinations of rendering modes are supported, especially for 3D PDF and 3D printing. Remember these vendor's software all interpret the VRML files in slightly different ways. You will likely not be able to do everything you can do in a 4DIM or in EVS.
- VRML viewers: There is a list of VRML viewing software published by National Institute of Standards and Technology [here](#). We recommend Cosmo, though it is far from perfect. We have created VRML files which will not display correctly in any of the VRML viewers that we have tested (including Cosmo), but which DO convert to 3D PDF perfectly. Conversely, there are occasions when something will look ok in VRML and not convert properly to 3D PDF.

Module Input [Ports](#)

- **View** [View] Connects to the viewer to receive all objects in the view

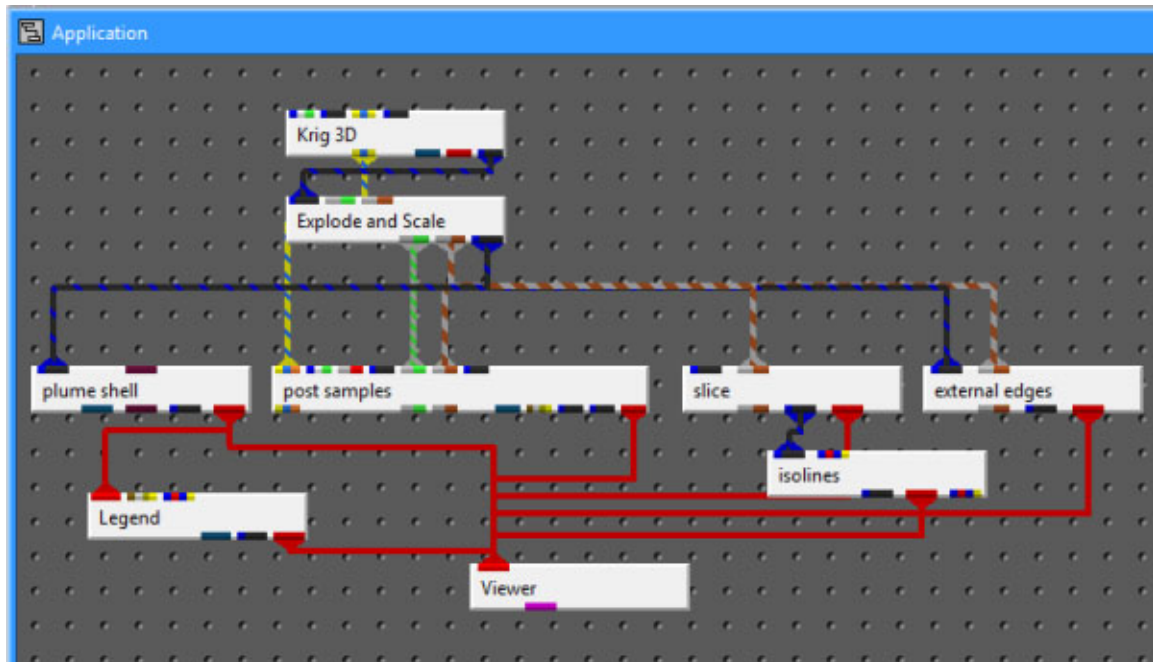
Guidelines for 3D PDF Creation

The following is a list of guidelines that must be considered when making EVS models that will be output as 3D PDF files using the *C Tech 3D PDF Converter*.

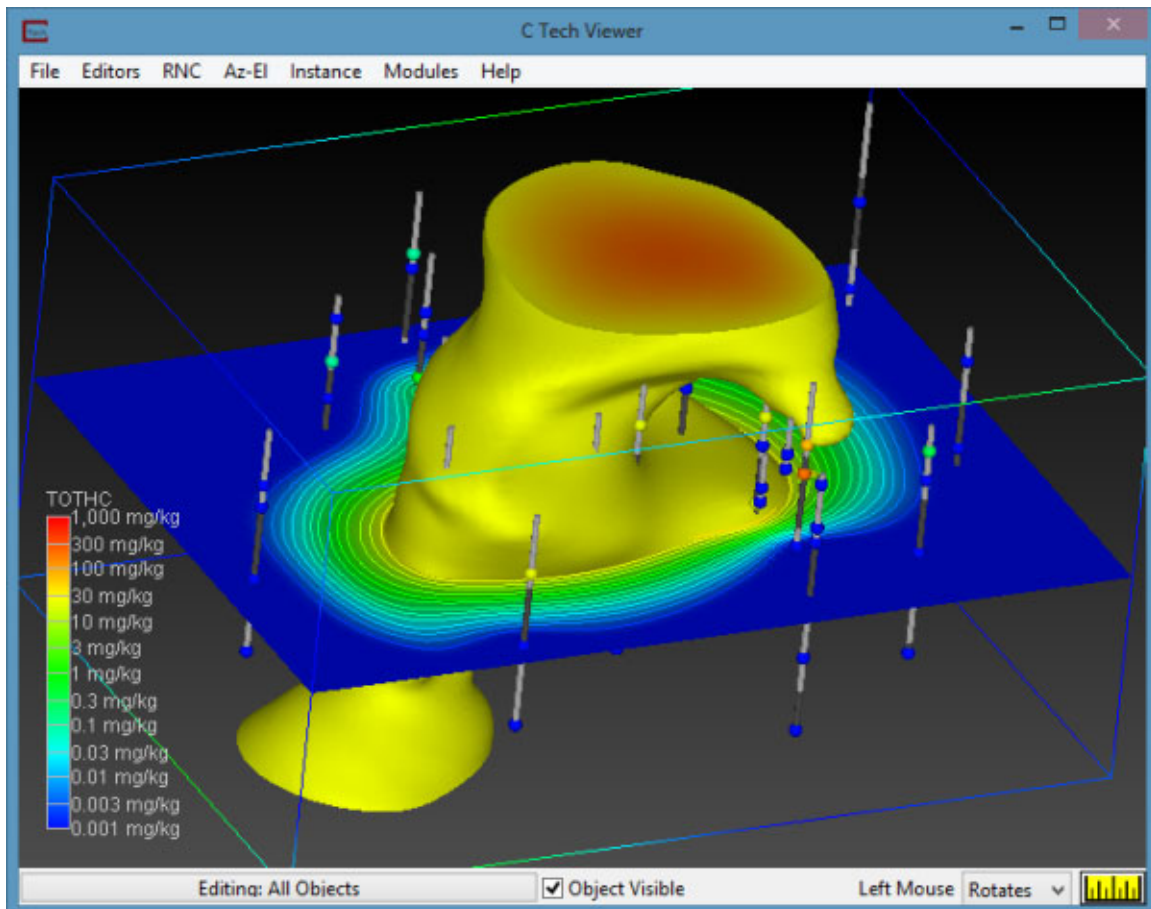
Note: The C Tech 3D PDF Converter is a separately purchased product not included with any other C Tech software licenses. Please see www.ctech.com for pricing.

EVS output from [write_vrml](#). You must follow the guidelines in write_vrml in addition to these additional guidelines.

Let's begin by building a simple application

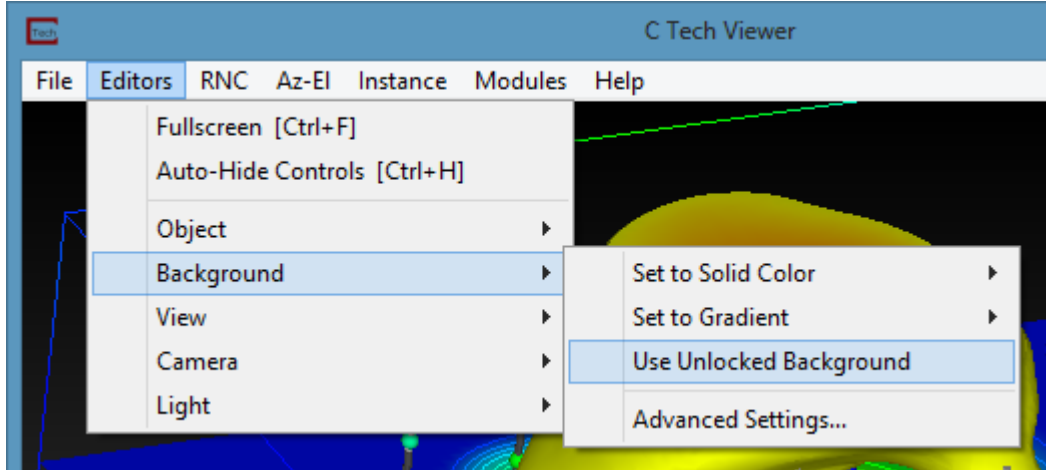


Whose output is:



The first things we MUST do for VRML output are to remove the legend and use an *Unlocked Background*. If you see a gradient background in your viewer, you

definitely aren't using an unlocked background. Once you use an unlocked background, you can still set a solid (single) background color.



Always set your viewer to a Top View (180 Azimuth and 90 Inclination) before writing the VRML file.

If we output this current model as VRML and convert to 3D PDF,

C Tech 3D PDF Converter

License

Maintenance Date: 7/9/2014 12:00:00 AM [Update License](#)

Input/Output

Output File: C:\CTech\Data\krig3d_plume_slice.pdf

Input Files:

C:\CTech\Data\krig3d_plume_slice-30.wrl

Add Remove Edit

Page Setup

Units: Inches (")

Size: Height: 8.5 Width: 11

Landscape: ☒

Margins: Top: 0.7 Left: 0.7 Right: 0.7 Bottom: 0.7

Top Background Color: Black

Bottom Background Color: Black

Model Setup

Units: Feet

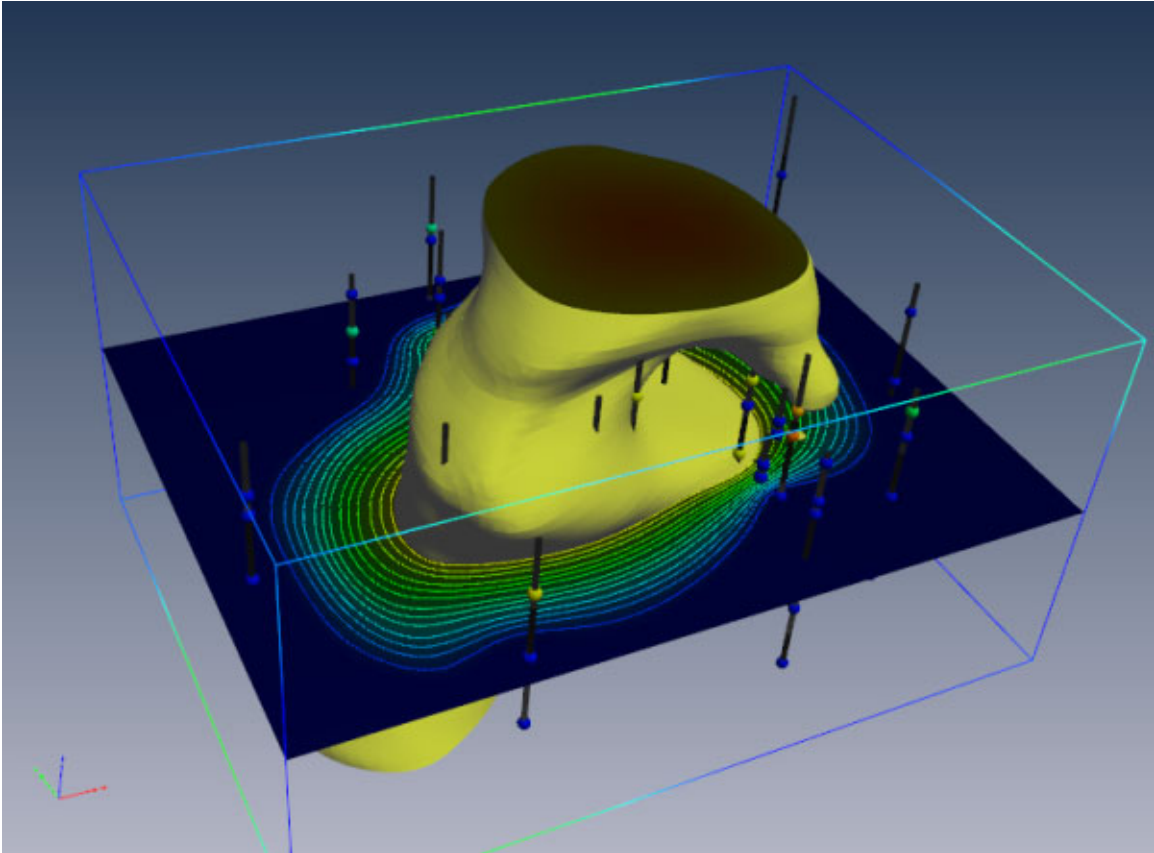
Illumination: CAD

Animation

Type: None

Convert

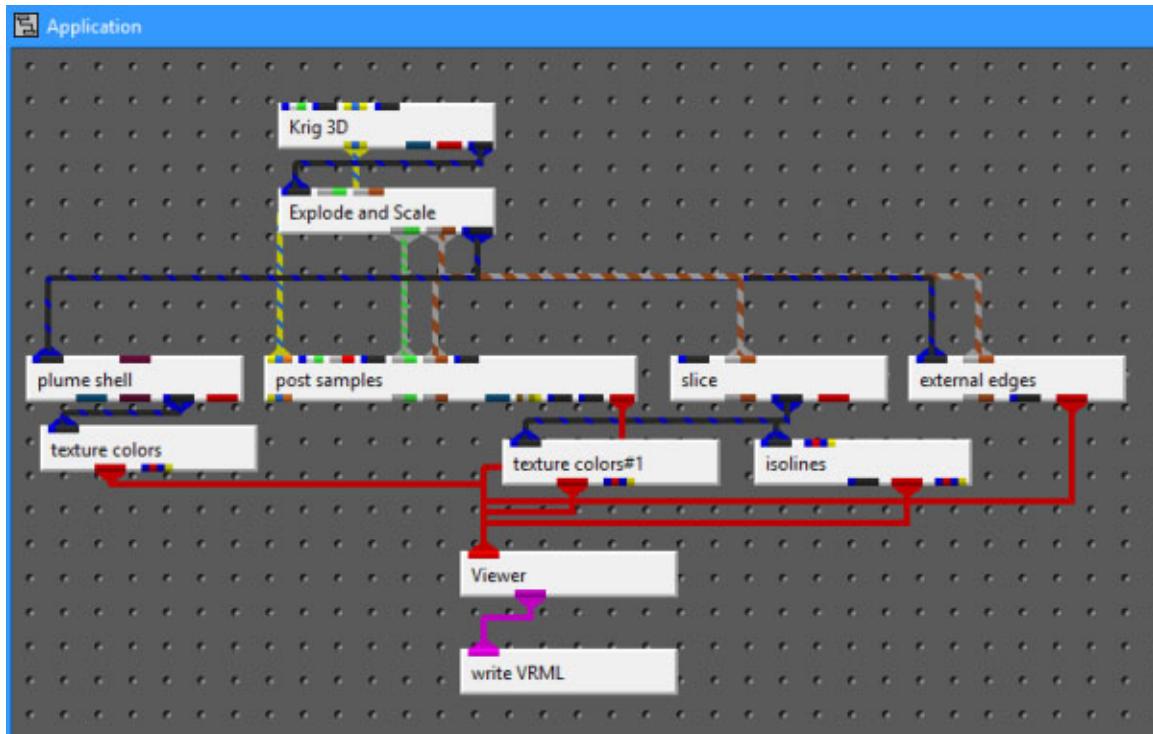
the results are less than wonderful:



The above 3D PDF has three obvious problems:

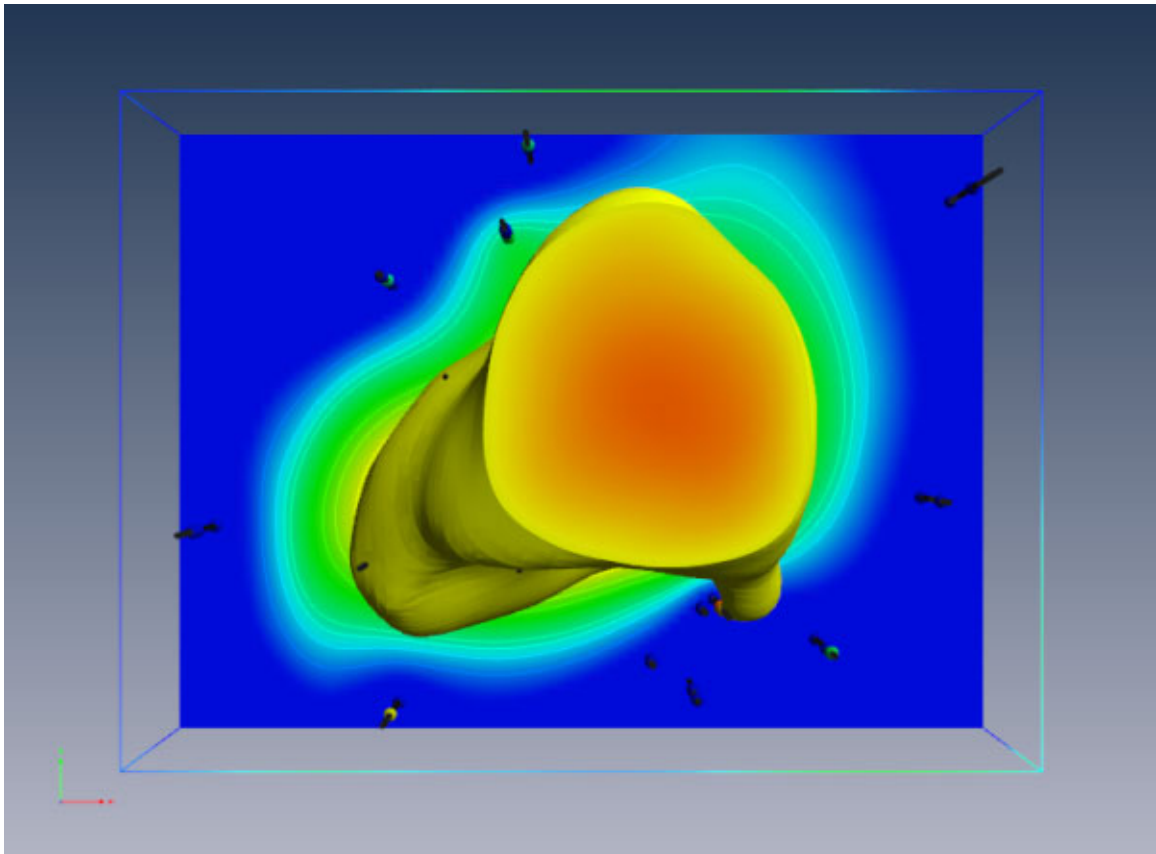
1. The top and bottom of the plume are very dark.
2. The slice is dark
3. post_sample's borings are dark.

We need to modify the application using two [texture colors](#) modules as follows:

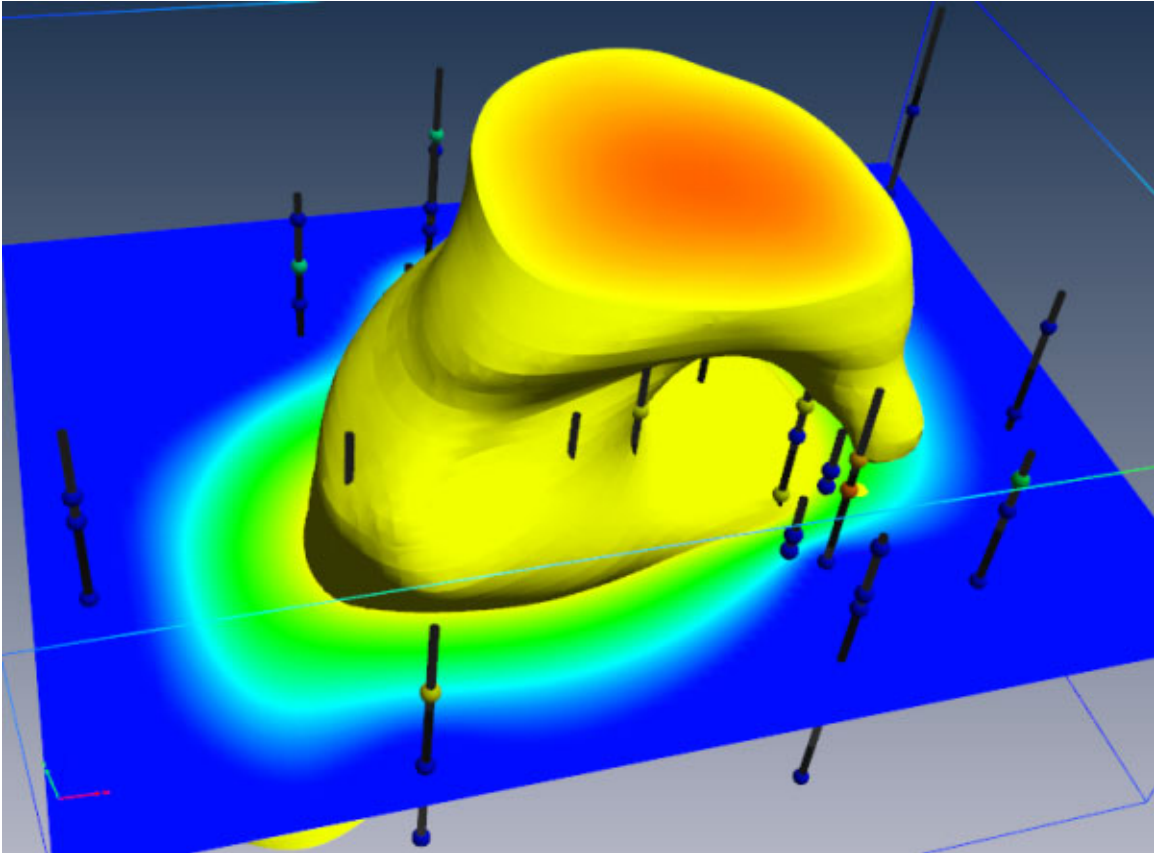


You'll notice that in the revised application, the output in the viewer is virtually identical. This will address the first two problems, however we expect to resolve the dark borings in an upcoming release.

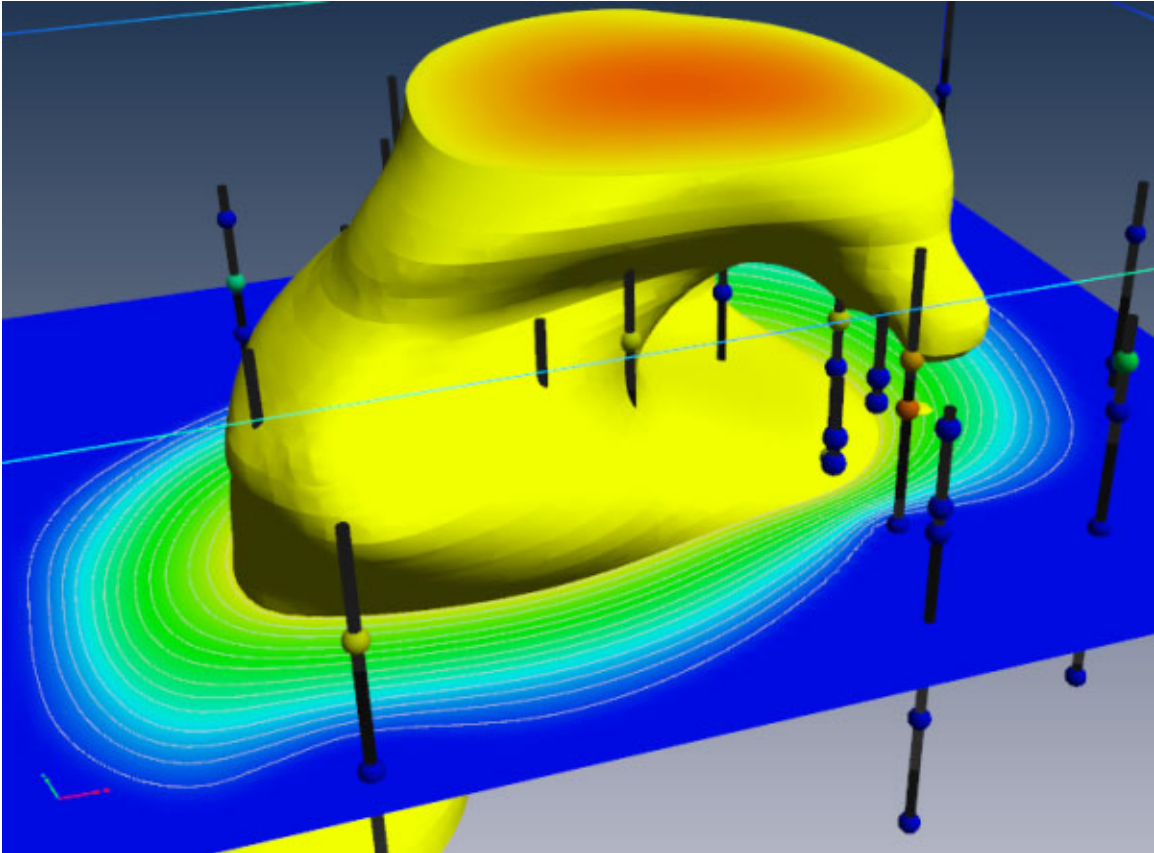
If we export this model to VRML and convert to PDF, the result is:



One other issue is that by default, we create isolines coincident with the surface(s) and resolve the coincidence in EVS using jitter. At some rotations you will notice that the isolines may disappear. This can be because jitter is not supported, but also because the underlying surface is so bright that the lines are not distinguishable.



This can be addressed using the `surface_offset` parameter in isolines. This will offset the lines from the surface (in one direction) and eliminate the coincidence. However, this will also mean that the lines will not be visible from one side of the slice. Making the lines uncolored is another option.



Guidelines for 3D Printing

The following is a list of guidelines that must be considered when making visualizations that will be printed using 3D Systems (previously Zcorp) technology. As of this software release, no other full color 3D printer has been successfully tested with output from [write_vrml](#). You must follow the guidelines in `write_vrml` in addition to these additional guidelines.

These guidelines are provided to minimize printing problems. Users should fully understand the issues below or they will likely not create VRML files suitable for 3D printing. Given the cost of the raw material it is best to do it right the first time!

Many of these issues (if not heeded) will be obvious when the model is viewed in Z Corp's ZPrint software. Make sure the model is carefully examined in ZPrint before actual printing.

1. **Internal Faces:** You must avoid internal External faces. This naturally occurs when we cut a hexahedral volumetric model with our older `plume_volume` module. The volumetric subset consists of hexahedron and tetrahedron cells. This creates surfaces that are internal to the model even though they represent the external faces of each set of cells. The real problem here is that the mating surfaces of each cell set are coincident (see 4 below). This major problem and many others are resolved by the `intersection_shell` module.
2. **Normals:** Must have all surface normals facing outward to define a solid volume for printing (handled by `intersection_shell` module)

3. **Coincident surfaces:** You **CANNOT HAVE** coincident surfaces. If two layers (or other objects) have coincident surfaces this will result in open parts and printing problems. You must separate the parts by a small amount (recommend 0.005 inches in final printed size) which should not be noticeable visually. Z-Print's process will fuse these parts together (because there isn't sufficient gap to keep them truly separate).
4. **Overlapping parts:** This is supported. It is possible to have two closed volumes overlap each other and Z-Print will sort it out so long as 1, 2 and 3 above are still valid.
5. **Surfaces:** *Must be extruded* or represented as a volumetric layer. Surfaces have no thickness and if placed coincident with the top of a volumetric object will result in leaving the volume OPEN (unclosed). This will cause serious problems.
6. **Cell Data:** Another limitation is the inability to mix nodal and cell data. Since we use nodal data for so many things you should always strip out the cell data and use nodal data exclusively. You must be aware of the following:
 - a. Ensure that there are no modules connected to the viewer that contain cell data. The safest way to ensure this is to pass questionable modules through `extract_mesh` with "Remove Cell Data" toggle ON. Normally you would want the "Remove Nodal Data" toggle OFF.
 - b. If you want your cell data (colors) to be displayed, pass the cell data through the `cell_to_node` module. However be aware that you'll still need to use `extract_mesh` afterwards because `cell_to_node` doesn't remove the cell data it just creates new nodal data from cell data.
 - c. Typical modules that have cell data are `read_vector_gis`, `indicator_geology`, `Solid_3D_Set`, `Solid_contour_set`, and most of the modules in the **Cell Data** library.
7. **Explode distance:** Need to ensure that there is sufficient gap between exploded layers (separate parts) so that they don't fuse together. Separation should be 1 mm (0.04 inches) minimum in the final print scale. Be aware that a 1 mm gap in the Z direction isn't equivalent to a 1 mm separation if the mating parts have high slopes. *If your mating surfaces have a 45 degree slope, the separation is reduced by $\cos(45)$ (~ 0.7). If you have higher slopes such as 80 degrees, the factor would be ~ 0.17 . This would mean that you would need a Z gap of nearly 6 mm to ensure a 1 mm separation between parts.*
8. **Disconnected pieces:** Although Z Print can print disconnected pieces, they won't retain their spatial position. Plumes that aren't connected to solid structure will just be loose pieces in the final print. This would also apply to `post_samples`' borings and spheres, unless they are connected by some common surface or geologic layer.
9. **Concepts that are NOT Supported:**
 - a. **Points and Lines:** Points and Lines cannot be printed (except as elements of an image used in a texture map). Lines must be converted to some 3D solid structure (such as closed tubes) and they must be of sufficient thickness to have some strength AND must not be disconnected pieces. Points should be represented as glyphs of sufficient size and not be disconnected.

- b. **Transparency:** Transparency as an object property cannot be supported since Z Print's ink is printed onto opaque plaster or starch powder. The illusion of transparency could be achieved by creating a texture map that was a blend (using the `image_transition` module) between two different images.
 - c. **Volume rendering:** This is a subset of Transparency and therefore is not supported at all.
 - d. **Jitter:** First, you must make sure that coincident surfaces are avoided anyway. Jitter is designed into EVS to allow preferential visualization of coincident objects. With Z Printing we cannot have coincidence in the first place! Offset the desired **primary** object to ensure that it is visible.
Remember no lines and no surfaces!
10. **Thin sections:** This is a somewhat subjective issue in that we really can't tell you the definition of *too fragile*?. We would recommend a minimum thickness of 0.5 mm, but depending on the width (total cross sectional area of the section) this may be too fragile or exhibit too much distortion during curing. We still want to have lenses pinch out, but if sections get very thin, the pieces may break.
11. **Top View:** You should write out the VRML file from a top view. If there are any truly flat (horizontal) surfaces, this keeps them flatter and smoother. Also, it helps to keep the models with the largest dimensions in the x-y plane (rather than z). This speeds up printing.

Recording (Capturing) 4DIM Files

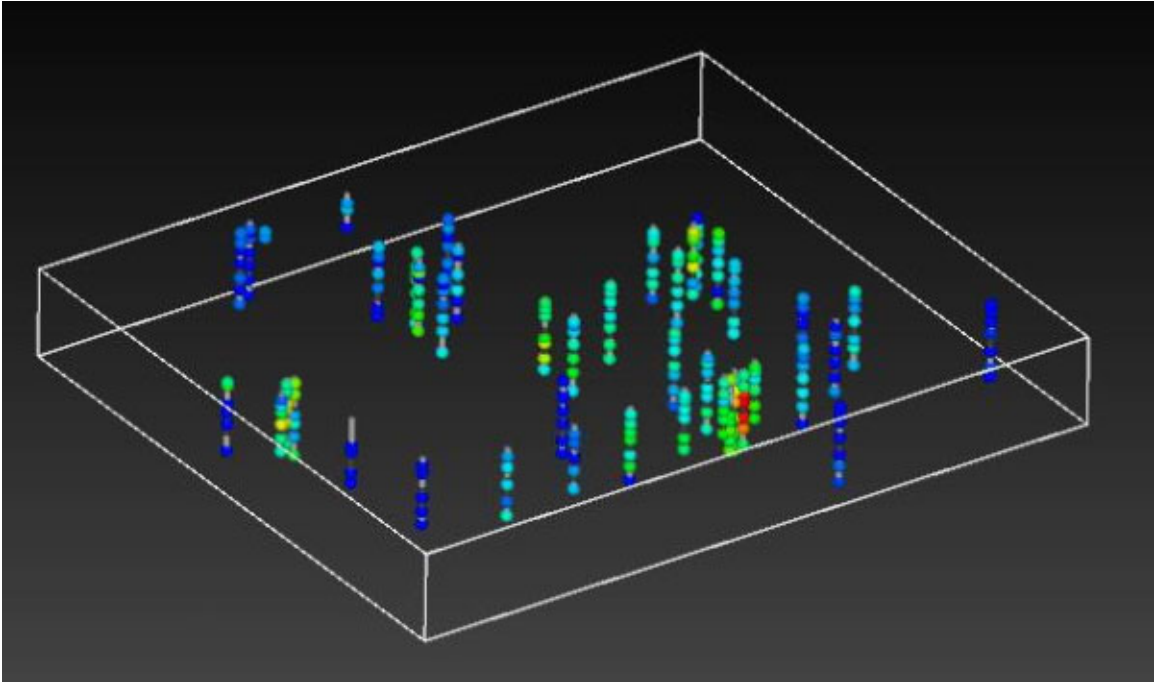
The Record 4DIM output option in the Viewer provides the ability to export in C Tech's proprietary 4DIM vector animation format.

Limitations

- In some circumstances `transform_group` cannot be used with 4DIMs. It can cause the 4DIM extents to be different than they were in the EVS viewer. This has been noted when doing rotations.
 - In most cases, the [transform_field](#) module can be used instead, however it does not allow for multiple objects to be connected to its input.
- [volume_renderer](#) is not compatible with 4DIMs
- 4DIM files will not record any object whose cache has been disabled. This occurs when large fields are connected to the viewer. When this occurs (for `external_faces` in this example), the following message appears in the Status Window:

```
--- Warning from: module: external_faces ---
Field is too big (140 MB) to be put into GDOBJECT's cache (128 MB).
Drawing the bounds only. Consider increasing the cache size or reducing
the field's complexity.
```

You will also know this has happened when you see an object in your viewer that is only the white bounds of what SHOULD be displayed. Such as:



When this occurs, the procedure to fix it is:

1. Select the object using the *Choose Object to Edit* button the viewer's Properties.
2. Increase the cache size from the default value of 128 (Mb) to a larger value.

Operation

When in Manual mode, frames (3D Models) are saved only when the "Record a Single Frame" button is depressed. When in Automatic mode, every time the *model is changed* a frame is appended the 4DIM animation. The definition of *model is changed* is not the same as the automatic mode in output_images. For this module, a change is defined as a change to one or more of the 3D objects in the viewer. Merely manipulating the view with Az-Inc or your mouse does not constitute a change. The reason for this is that recording frames that represent viewer manipulations is a waste. 4DIM files can be manipulated exactly the same way you manipulate the viewer. With 4DIM files we only want to save frames that represent changes to the **content** in the viewer.

Before the 4DIM file is written, you have the option of deleting the last frame (this can be done repeatedly) or clearing all frames. When creating small 4DIMs manually, this can be useful. What is saved?

Some geometries may not display properly when the animation is played back. In particular, volume rendering is not supported.

Geometry that does not change from frame-to-frame is not re-saved. Instead, a reference is made to the previous frame so that data does not need to be duplicated. Invisible objects (visible set to zero) are not captured.

View attributes will not be saved as part of the animation.

Attributes that can be saved

1. Visibility

2. Transparency
3. Most object modes (rendering modes and line modes)
4. Background color and background type
 1. If Locked 2 or 4 color backgrounds are used, they cannot be changed by the user in the 4DIM player

View, Light and Camera Attributes

The following lists the view attributes you can change.

You can change all view attributes.

All light attributes can be changed.

The following camera attributes can be changed:

lens

clipping plane

depth cueing

merge_fences

The merge_fences module is used to merge the output from multiple krig_fence modules into one data set (i.e., to merge cross sections into a fence diagram). This is useful for performing uniform data manipulation procedures on fence data from several krig_fence outputs. For example, if several krig_fence modules are used, they should all pass through a merge_fences module before being passed to explode and scale. Therefore, all fences will be exploded and scaled the same amount and only one dialog box is needed to control all fences. merge_fences should always be used when more than one krig_fence module is used.

Module Input [Ports](#)

- **First Input Field** [Field] Accepts a data field.
- **Second Input Field** [Field] Accepts a data field.
- **Third Input Field** [Field] Accepts a data field.
- **Fourth Input Field** [Field] Accepts a data field.

Module Output [Ports](#)

- **Output Field** [Field] Outputs the field with all inputs merged

geologic_surfmap

This module is deprecated and replaced by [surfmap](#).

geologic_surfmap provides a mechanism to drape lines onto Geologic surfaces. It compares to surfmap, but lines are not subsetting to match the size of the cells of the surface on which the lines are draped. In other words, only the endpoints of each line segment are draped.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Geologic Field** [Field] Accepts a geologic field
- **Input Lines** [Field] Accepts a field with the lines to be draped

Module Output [Ports](#)

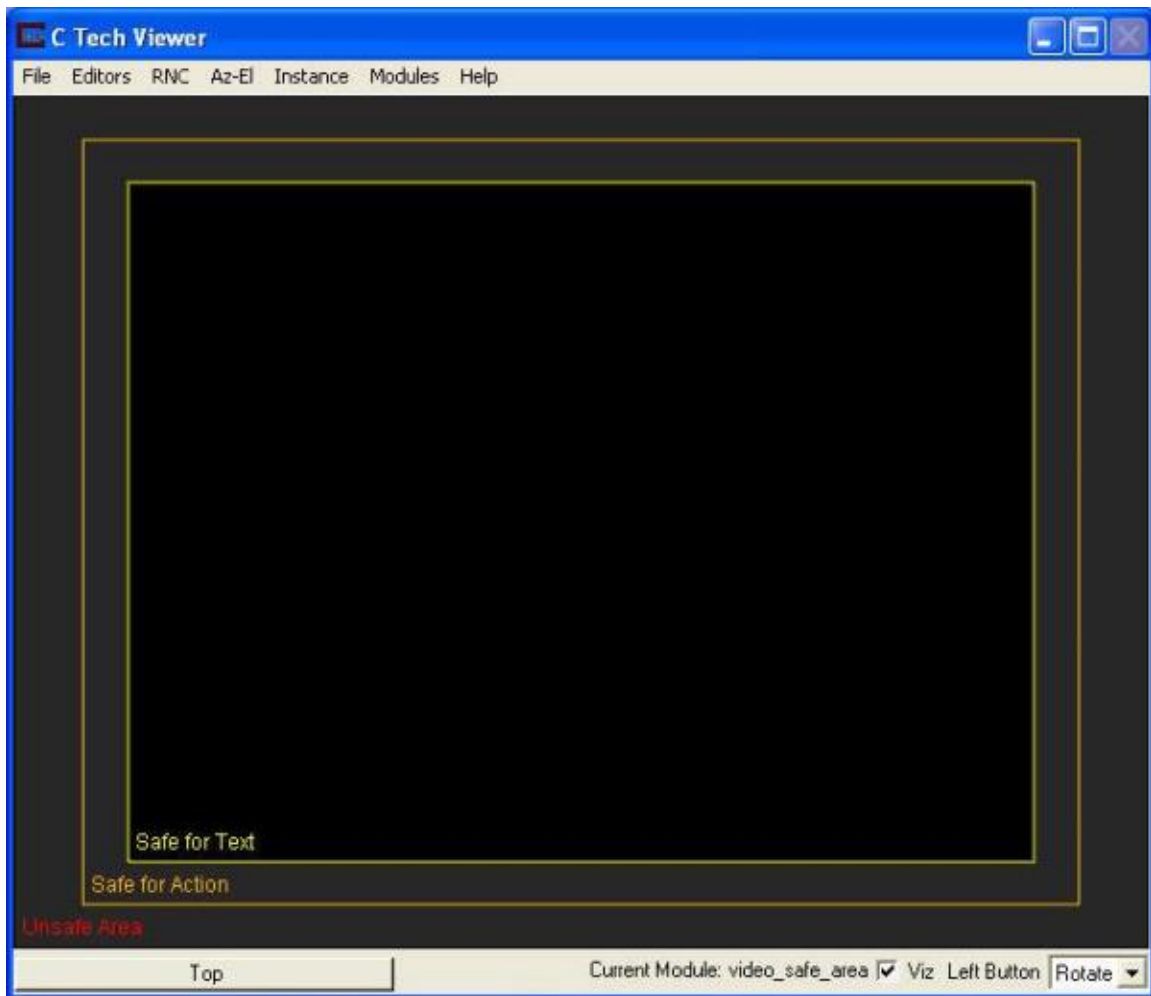
- **Z Scale** [Number] Outputs the Z Scale (vertical exaggeration).
- **Output Field** [Field] Outputs the draped lines
- **Surface** [Renderable]: Outputs the draped lines to the viewer.

time_field

The time_field module allows you to extract a field (grid with data) from a set of time-based fields. The time for the extracted field can be any time between the start and end of the set of fields. It will interpolate between adjacent known times.

video_safe_area

The video_safe_area module is used when creating an animation for DVD or Video. It displays the areas that are usable for both text and animation purposes for several standard video formats. This allows you to properly setup your animation in order to get the best possible output on multiple television sets.



The **VideoOutput Format** changes the safe areas in the viewer window to match the default width and height values for the selected video format.

The **Visible** toggle turns the safe area display on and off. This toggle should always be off when making the actual video so the safe areas are not recorded.

The **Move to Back** toggle will put the safe area display behind any graphics in the viewer.

The **Transparency** slider changes the opacity of the safe area mask.

The **Mask** toggle turns the safe area masks on and off. The mask is a visual tool to help visualize which graphics fall into which safe area.

The **Mask Text Area** toggle turns the masking surrounding the text area on or off.

Mask Color alters the color of the masking.

The **Lines** toggle turns the lines defining the safe areas on and off.

The **Labels** toggle turns the labels defining the safe areas on and off.

The **Action Border Color** button selects the color of the action border.

The **Text Border Color** button selects the color of the text border.

Selecting **Set viewer Res.** sets the resolution of the viewer to the default for the video format that has been selected.

If the **Preserve Width** toggle is selected when the Set viewer Res. toggle is chosen, the current resolution width of the viewer will be maintained while the resolution height of the viewer will be based upon the appropriate ratio for the video format that has been selected.

If the Preserve Width toggle is unselected the **Double Res** toggle can be selected.

The Double Res toggle will double the resolution of the viewer, while keeping the appropriate width-height ratio for the video format that has been selected. This should only be used while using the Screen Renderer output of output_images with the 4x4 anti-aliasing option.

The **Update viewer** button will set the viewer to the correct width and height if the Set viewer Res toggle has been selected.

advector

The advector module combines [streamlines](#) capability and a tool for sequential positioning of glyphs along the streamlines trajectory to simulate advection of weightless particles through a vector field (for example, a fluid flow simulation such as modflow). The result is an animation of particle motion, with the particles represented as any EVS geometry (such as a jet or a sphere). The glyphs can scale, deflect or deform according to the velocity vector it passes. At least one of the nodal data components input to advector must be a vector. The direction of travel of streamlines can be specified to be forwards (toward high vector magnitudes) or backwards (toward low vector magnitudes) with respect to the vector field. The input glyphs travel along streamlines (not necessarily visible in the viewer) which are produced by integrating a velocity field using the Runge-Kutte method of specified order with adaptive time steps.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field with vector data.
- **Input Starting Locations** [Field] Accepts a data field.
- **Input Glyph** [Field] Accepts a field representing the glyphs

Module Output [Ports](#)

- **Output Field** [Field] Outputs the glyphs
- **Output Streamlines** [Field] Outputs the streamlines field
- **Output Glyph** [Renderable]: Outputs the glyphs to the viewer.

- **Output Streamlines Object** [Renderable]: Outputs the streamlines to the viewer.

modpath_advvector

The modpath_advvector module combines MODPATH capability and a tool for sequential positioning of glyphs along the MODPATH lines trajectory to simulate advection of weightless particles through a vector field. The result is an animation of particle motion, with the particles represented as any EVS geometry (such as a jet or a sphere). The glyphs can scale, deflect or deform according to the velocity vector it passes. The direction of travel of streamlines can be specified to be forwards (toward high vector magnitudes) or backwards (toward low vector magnitudes) with respect to the vector field. The input glyphs travel along streamlines (not necessarily visible in the viewer) which are produced by integrating a velocity field using the Runge-Kutte method of specified order with adaptive time steps.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field with vector data.
- **Input Starting Locations** [Field] Accepts a data field.
- **Input Glyph** [Field] Accepts a field representing the glyphs

Module Output [Ports](#)

- **Output Field** [Field] Outputs the glyphs
- **Output Streamlines** [Field] Outputs the streamlines field
- **Output Glyph** [Renderable]: Outputs the glyphs to the viewer.
- **Output Streamlines Object** [Renderable]: Outputs the streamlines to the viewer.

create_spheroid

This module is deprecated and replaced by [place_glyph](#)

The create_spheroid module produces a 2D circular disc or 3D spheroidal or ellipsoidal grid that can be used for any purpose, however the primary application is as starting points for streamlines or advector.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a field to extract its extent

Module Output [Ports](#)

- **Output Field** [Field / Minor] Outputs the surface
- **Surface** [Renderable]: Outputs to the viewer

advect_surface

The advect_surface module combines [streamline_surface](#) capability and a tool for sequential positioning of glyphs along the streamlines trajectory to simulate advection of particles down a surface. The result is an animation of particle motion, with the particles represented as any EVS geometry (such as a jet or a sphere). The glyphs can scale, deflect or deform according to the velocity vector. The direction of

travel of streamlines can be specified to be downhill or uphill (for the slope case). The input glyphs travel along streamlines (not necessarily visible in the viewer) which are produced by integrating a velocity field using the Runge-Kutte method of specified order with adaptive time steps.

The `advect_surface` module is used to produce streamlines and particle animations on any surface based on its slopes. The direction of travel of streamlines can be specified to be downhill or uphill for the slope case. A physics simulation option is also available which employs a full physics simulation including friction and gravity terms to compute streamlines on the surface.

Module Input [Ports](#)

- **Z Scale** [Number] Accepts Z Scale (vertical exaggeration).
- **Input Field** [Field] Accepts a field with vector data.
- **Input Starting Locations** [Field] Accepts a data field.
- **Input Glyph** [Field] Accepts a field representing the glyphs

Module Output [Ports](#)

- **Output Field** [Field] Outputs the glyphs
- **Output Streamlines** [Field] Outputs the streamlines field
- **Output Glyph** [Renderable]: Outputs the glyphs to the viewer.
- **Output Streamlines Object** [Renderable]: Outputs the streamlines to the viewer.

fence_geology

The `fence_geology` module uses data in specially formatted .geo files to model the surfaces of geologic layers in vertical planes, or cross sections. Fence Geology essentially creates layers of quadrilateral (4 node) elements (in a vertical plane) in which each node (and element) is assigned to an individual geologic layer. The output of `fence_geology` is a data field, consisting of a 2D line with each layers elevation as nodal data elements, that can be sent to the `krig_fence` and `3d_geology_map` modules where the quadrilateral elements are connected to the element nodes in adjacent geologic surfaces to create layers along the fence.

Module Input [Ports](#)

- **Input Filename** [String] Receives the filename from other modules.
- **Input Line** [Field] Allows the user to import a line (path) to which all data will be kriged.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Line** [Field] Connects to `krig_fence`
- **Filename** [String / minor] Outputs a string containing the file name and path. This can be connected to other modules to share files.

file_output

The file_output module creates a formatted string based upon the values passed to it. This string is then written to the selected ascii text file. Certain modules such as krig_3d, krig_2d, and krig_fence output a formatted string for just this purpose.

adaptive_indicator_krig

adaptive_indicator_krig is an alternative geologic modeling concept that uses geostatistics to assign each cell's lithologic material as defined in a pregeology (.pgf) file, to cells in a 3D volumetric grid.

There are two methods of lithology assignment:

- Nearest Neighbor is a quick method that merely finds the nearest lithology sample interval among all of your data and assigns that material. It is very fast, but generally should not be used for your final work.
- Kriging provides the rigorous probabilistic approach to geologic indicator kriging. The probability for each material is computed for each cell center of your grid. The material with the highest probability is assigned to the cell. All of the individual material probabilities are provided as additional cell data components. This will allow you to identify regions where the material assignment is somewhat ambiguous. Needless to say, this approach is much slower (especially with many materials), but often yields superior results and interesting insights.

adaptive_indicator_krig is an extension of the technology in indicator_geology for several reasons:

1. Material assignments are done on a nodal versus cell basis providing additional inherent resolution
2. Gridding is handled by outside modules. This allows for assigning material data based on a PGF file after kriging analyte (e.g. chemistry) or other parameter data with krig_3d.
3. Though it does not provide material boundaries that are as smooth as krig_3d_geology, it does provide much smoother interfaces than indicator_geology's *Lego-like* material structures.

There are two fundamental differences between indicator_geology and adaptive_indicator_krig

1. Geology / Grid input:
 1. indicator_geology expects input from modules like krig_3d_geology (which is a set of surfaces) and it builds you grid for you just as krig_3d does.
 2. adaptive_indicator_krig is more like the "Kriging to an external grid" option in krig_3d. You need to create the 3D grid (which doesn't need to have any data) that it will use. It will take that grid as a starting point for material assignments and later smoothing.
2. Lithologic Material Assignment
 1. indicator_geology assigns whole cells to cell sets and sets CELL data which is Material_ID.

2. `adaptive_indicator_krig` takes the external grid and further refines it by splitting whole cells along all boundaries between two or more materials to create smoother interfaces.

Module Input [Ports](#)

- **Input Field** [Field] Accepts a data from `krig_3d`, `3d_geology_map` or other modules that have already created a grid containing volumetric cells. If the input field has data such as concentrations, it will be included in the output.
- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Refine Distance** [Number] Accepts the distance used to discretize the lithologic intervals into points used in kriging.

Module Output [Ports](#)

- **Geologic legend Information** [Geology legend] Supplies the geologic material information for the legend module.
- **Output Field** [Field] Contains nodal data and a refined grid representing geologic materials..
- **Filename** [String / minor] Outputs a string containing the file name and path. This can be connected to other modules to share files.
- **Refine Distance** [Number] Outputs the distance used to discretize the lithologic intervals into points used in kriging or displayed in `post_samples` as spheres.

Properties and Parameters

The Properties window is arranged in the following groups of parameters:

- Grid Settings: control the grid type, position and resolution
- Krig Settings: control the estimation methods
 - NOTE: Nearest Neighbor assigns the lithologic material cell data based on the nearest lithologic material (in anisotropic space) to your PGF borings. This is done based on the cell center (coordinates) and an enhanced refinement scheme for the PGF borings. *In general Nearest Neighbor should not be used for final results*

Advanced Variography Options:

It is far beyond the scope of our Help to attempt an advanced Geostatistics course. The terminology and variogram plotting style that we use is industry standard and we do so because we will not provide detailed technical support nor complete documentation on these features, which would effectively require a geostatistics textbook, in our help.

However, we have offered an online course on how to take advantage of the complex, directional anisotropic variography capabilities in `adaptive_indicator_krig`(which applies equally well to `indicator_geology` and `krig_3d`), and that course is available as a recorded video class. This class is focused on the mechanics of how to employ and refine the variogram anisotropy with respect to your data and the physics of your project such as contaminated sediments in a river bottom. The variogram is displayed as an ellipsoid which can be distorted to

represent the Primary and Secondary anisotropies and rotated to represent the Heading, Dip and Roll. Overall scale and translation are also provided as additional visual aids to compare the variogram to the data, though these do not affect the actual variogram.

We are not hiding this capability from you as the *Anisotropic Variography Study* folder of *Earth Volumetric Studio Projects* contains a number of sample applications which demonstrate exactly what is described above. However, we assure you that understanding how to apply this to your own projects will be quite daunting and really does require a number of prerequisites:

- A thorough explanation of these complex applications
- A reasonable background in Python and how to use Python in Studio
- An understanding of all of the variogram parameters and their impact on the estimation process on both theoretical datasets as well as real-world datasets.

This 3 hour course addresses this issues in detail.

krig_fence

krig_fence models parameter distributions within domains defined by the boundaries of the input data in 3D Fence sections which can "snake" around in the x-y plane and are parallel to the z-axis. krig_fence can also receive the geologic system modeled by Fence Geology. It creates a quadrilateral finite-element grid with kriged nodal values of any scalar property and its kriged confidence level, and outputs a geometry whose elements can be rendered to view the color scaled parameter distribution on the element surfaces. krig_fence provides several convenient options for pre- and post-processing the input parameter values, and allows the user to consider anisotropy in the medium containing the property.

Module Input [Ports](#)

- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Fence Geology Input** [Field] Accepts a field from krig_fence containing geologic layers.
- **Input External Data** [Field / minor] Allows the user to import a field contain data. This data will be kriged to the grid instead of using file data.

Module Output [Ports](#)

- **Filename** [String / minor] Allows the sharing of file names between similar modules.
- **Output Field** [Field] Outputs a 3D data field which can be input to any of the Subsetting and Processing modules.
- **Status Information** [String / minor] Outputs a string containing module parameters. This is useful for connection to save_evs_field to document the settings used to create a grid.

fence_geology_map

The fence_geology_map module creates 3-dimensional fence diagram from the 1-dimensional line contours which follow your geology produced by fence_geology, to

allow visualizations of the geologic layering of a system. It accomplishes this by creating a user specified distribution of nodes in the Z dimension between the top and bottom lines defining each geologic layer.

The number of nodes specified for the Z Resolution may be distributed (proportionately) over the geologic layers in a manner that is approximately proportional to the fractional thickness of each layer relative to the total thickness of the geologic domain. In this case, at least three layers of nodes (2 layers of elements) will be placed in each geologic layer.

Module Input [Ports](#)

- **Input Geologic Field [Field]** Accepts fence_geology output

Module Output [Ports](#)

- **Output Field [Field]** Outputs the field

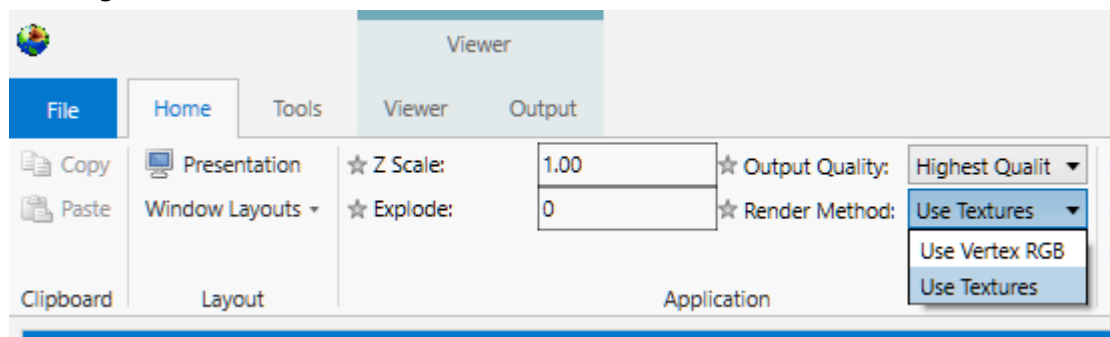
application_notes

The application_notes has been deprecated and replaced by the Annotation's "Notes"

texture_colors

This is a deprecated module

texture_colors functionality has been incorporated into all modules. On the Home tab, you have the Render Method selector where you can choose to use Vertex RGB coloring or Textures.



C Tech GMS Project File Converter

Note: This is unsupported technology. C Tech cannot provide tech support with issues of problems you may have with the use of this tool.

The GMS->EVS Converter takes as input a GMS project file (*.gpr) and parses it looking for parts that can be converted to an EVS file format. **This supports all version 4 GMS projects only.** To fully convert a MODFLOW, MT3DMS, or FEMWATER project make sure to run that simulation in GMS and save your project before attempting a conversion. There should only be one loaded solution for each model.

The following GMS Project and File types are convertible into their respective C Tech (EVS) file types:

1. **MODFLOW:** This converts into both EVS field files (*.eff, *.efz) and into an EVS TCF file (*.tcf) for animation.
2. **MT3DMS:** This converts into both EVS field files (*.eff, *.efz) and into an EVS TCF file (*.tcf) for animation.
3. **FEMWATER:** FEMWATER converts only into an EVS field file (*.eff, *.efz) at this time, because it is only handling steady state FEMWATER models. When this is rewritten to handle transient FEMWATER models, a TCF file (*.tcf) will also be added.
4. **IMAGE:** Image files (*.tif and *.jpg) are copied into the output directory and a georeferencing file (*.gcp) is created for image orientation.
5. **2D SCATTER:** These files are converted into EVS Geology Multi File format (*.gmf).
6. **3D SCATTER:** These files are converted into EVS 3D analyte (e.g. chemistry) files (*.apdv).
7. **2D Grid:** 2D Grid files are converted into the EVS Field file format (*. eff, *.efz) as well as EVS Geology Multi file format (*.gmf).
8. **TIN:** TIN files are converted into the EVS Field file format (*.eff, * .efz) as well as EVS Geology Multi file format (*.gmf).
9. **BOREHOLE:** BOREHOLE data is converted to the EVS Pre-Geology file format (*. pgf).

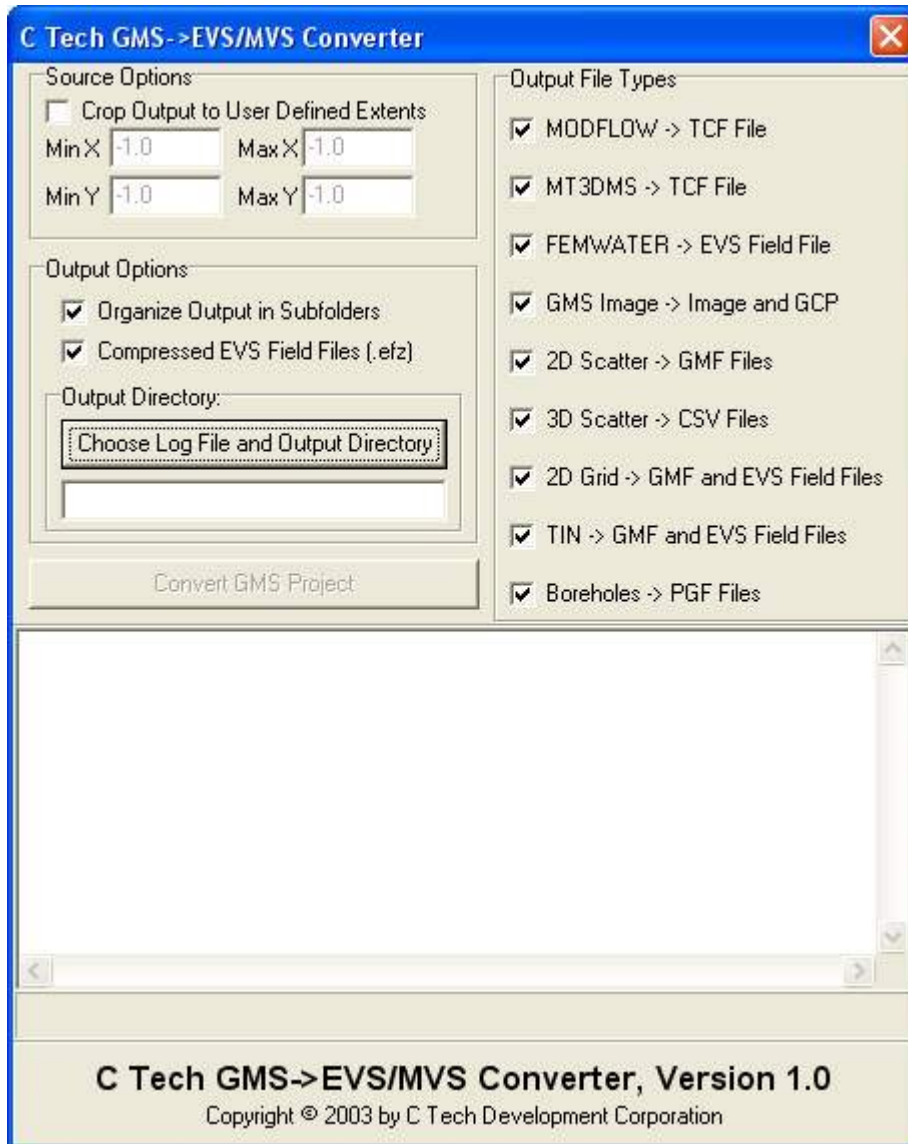
There is one source control option. By clicking on Crop Output to User Defined Extents the input from GMS can be focused on areas of interest to the user. These extents are assumed to be valid when entered by the user. No Error checking is done. This works only on MODFLOW, MT3DMS, and FEMWATER conversions.

There are two options for output:

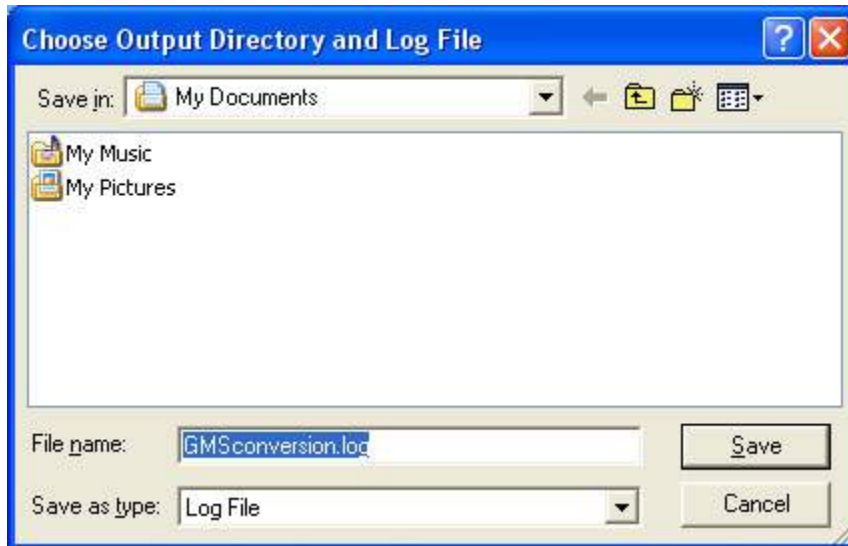
1. **Organize output to subfolders:** This allows the user to keep track of which specific part of the project was converted by placing the appropriate converted files in new subfolders based on the data type (e.g. MODFLOW, IMAGE, etc.).
2. **Compressed EVS Field Files(*.efz):** This option changes all EVS field files (*.eff) into their compressed form (*.efz) for space saving purposes.

To run the converter

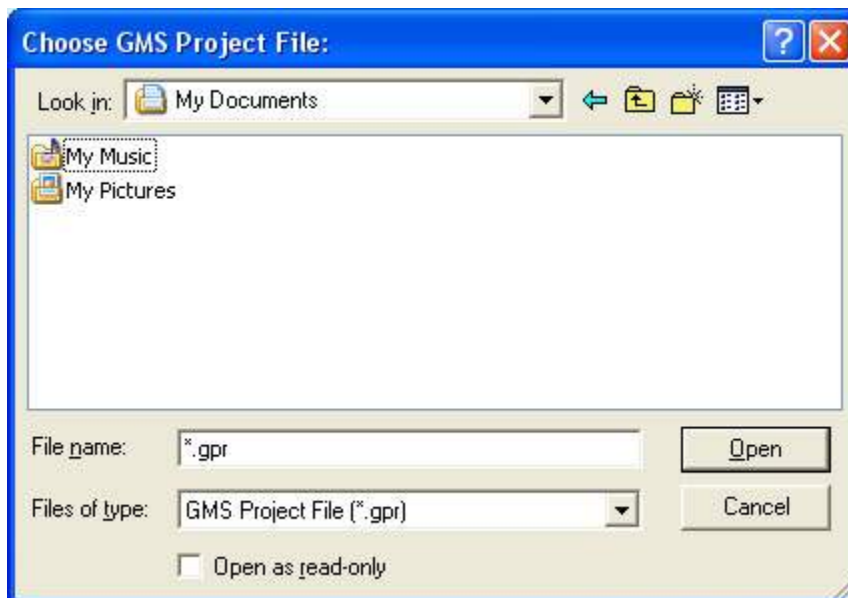
1. Select all appropriate options and file types to be converted.



2. Click the Choose Log File and Output Directory button to choose the destination for the log file and converted files/folders. The log file is a status report of the converter which keeps track of all errors and files created.

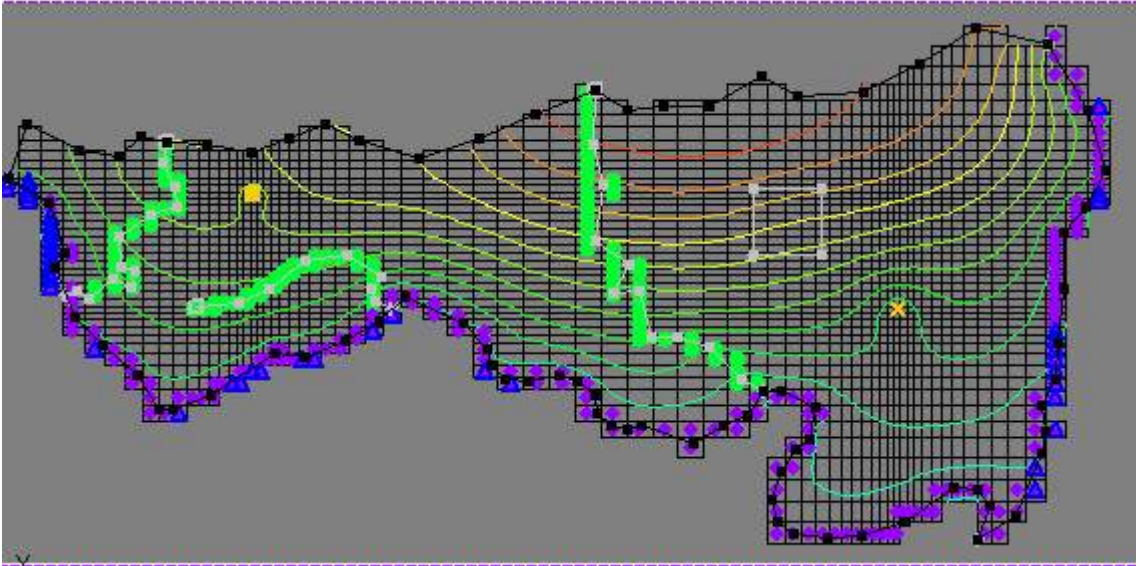


3. Next click on the Convert GMS Project button and select the GMS project file (*.gpr) to be converted. All files should be created and shown in the Status window, click Done to finish.

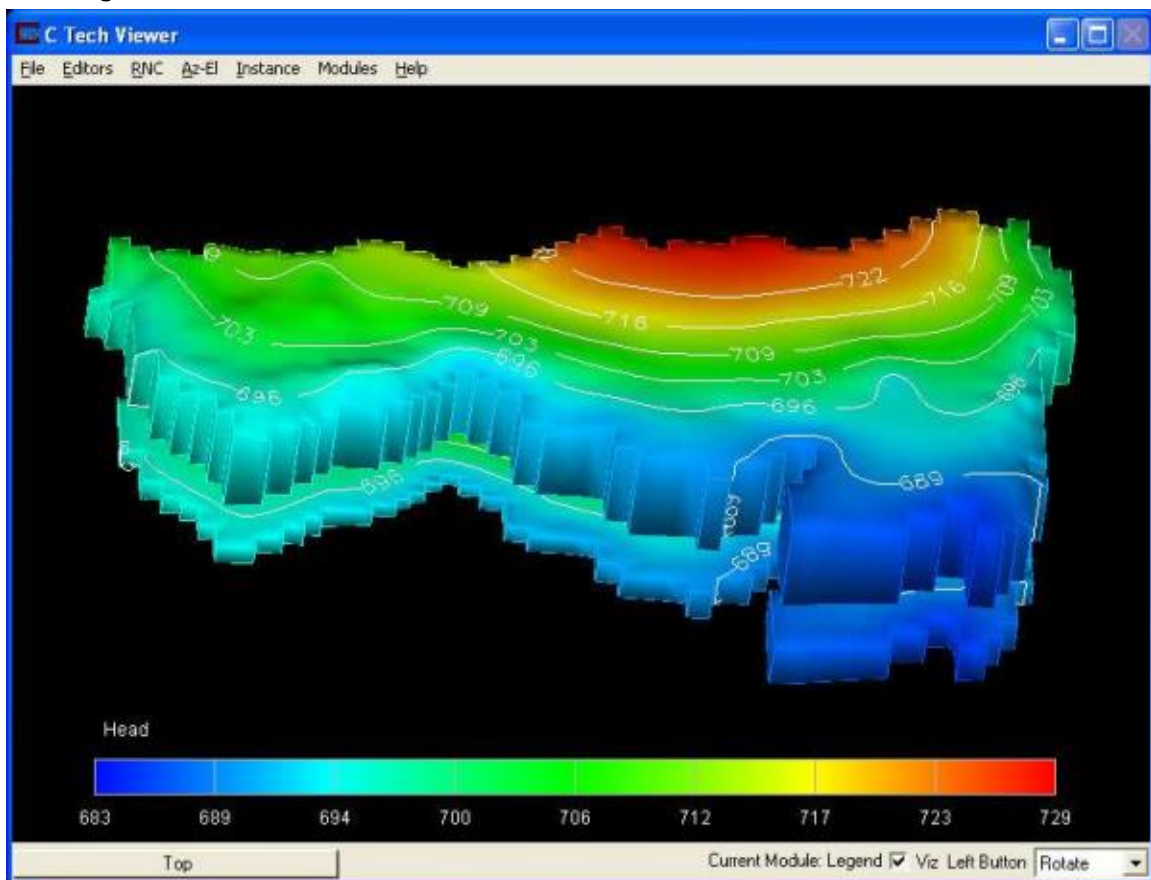


Example:

The following sample MODFLOW model in GMS was converted using the converter:

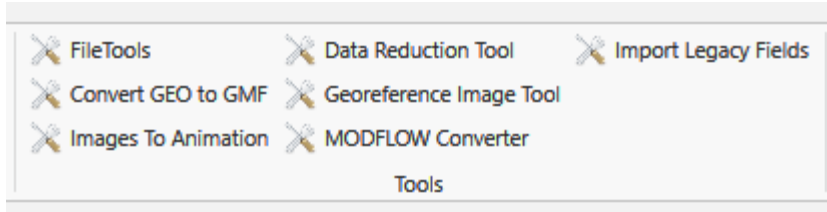


Upon running the GMS to EVS converter, a compressed EVS field file was created. This file can then be used in a variety of ways in EVS, as shown in the picture showing head values below:



Tools

The Tools section of the Tools tab provides the following functions:



These utilities are provided as tools vs. modules since none of them would have input or output ports to allow them to connect to other modules.

- [FileTools](#)
- **Convert Geo to GMF:** Allows you to quickly create a GMF file using a GEO file as input. This is useful if you want to replace a single surface in a GEO hierarchy (such as the ground surface) with more high-resolution data that is not synchronous with your .GEO borings.
- [Images To Animation](#)
- Data Reduction Tool: Used to get optimal results when kriging [Dense Data](#)
- [Georeference Image Tool](#)
- [MODFLOW Converter](#) Note: This is unsupported technology. C Tech cannot provide tech support with issues of problems you may have with the use of this tool.
- Import Legacy Fields: Reads older format files that can contain EVS Fields, such as UCD (.inp) and netCDF files and converts them to standard [EVS Field File](#) format.

4DIM Playback

4dim playback is integrated into Earth Volumetric Studio. You only need to open the file, just as you would open a Studio application.

C Tech GMS Project File Converter

Note: This is unsupported technology. C Tech cannot provide tech support with issues of problems you may have with the use of this tool.

The GMS->EVS Converter takes as input a GMS project file (*.gpr) and parses it looking for parts that can be converted to an EVS file format. **This supports all version 4 GMS projects only.** To fully convert a MODFLOW, MT3DMS, or FEMWATER project make sure to run that simulation in GMS and save your project before attempting a conversion. There should only be one loaded solution for each model.

The following GMS Project and File types are convertible into their respective C Tech (EVS) file types:

1. **MODFLOW:** This converts into both EVS field files (*.eff, *.efz) and into an EVS TCF file (*.tcf) for animation.
2. **MT3DMS:** This converts into both EVS field files (*.eff, *.efz) and into an EVS TCF file (*.tcf) for animation.
3. **FEMWATER:** FEMWATER converts only into an EVS field file (*.eff, *.efz) at this time, because it is only handling steady state FEMWATER models. When

this is rewritten to handle transient FEMWATER models, a TCF file (*.tcf) will also be added.

4. **IMAGE:** Image files (*.tif and *.jpg) are copied into the output directory and a georeferencing file (*.gcp) is created for image orientation.
5. **2D SCATTER:** These files are converted into EVS Geology Multi File format (*.gmf).
6. **3D SCATTER:** These files are converted into EVS 3D analyte (e.g. chemistry) files (*.apdv).
7. **2D Grid:** 2D Grid files are converted into the EVS Field file format (*.eff, *.efz) as well as EVS Geology Multi file format (*.gmf).
8. **TIN:** TIN files are converted into the EVS Field file format (*.eff, *.efz) as well as EVS Geology Multi file format (*.gmf).
9. **BOREHOLE:** BOREHOLE data is converted to the EVS Pre-Geology file format (*.pgf).

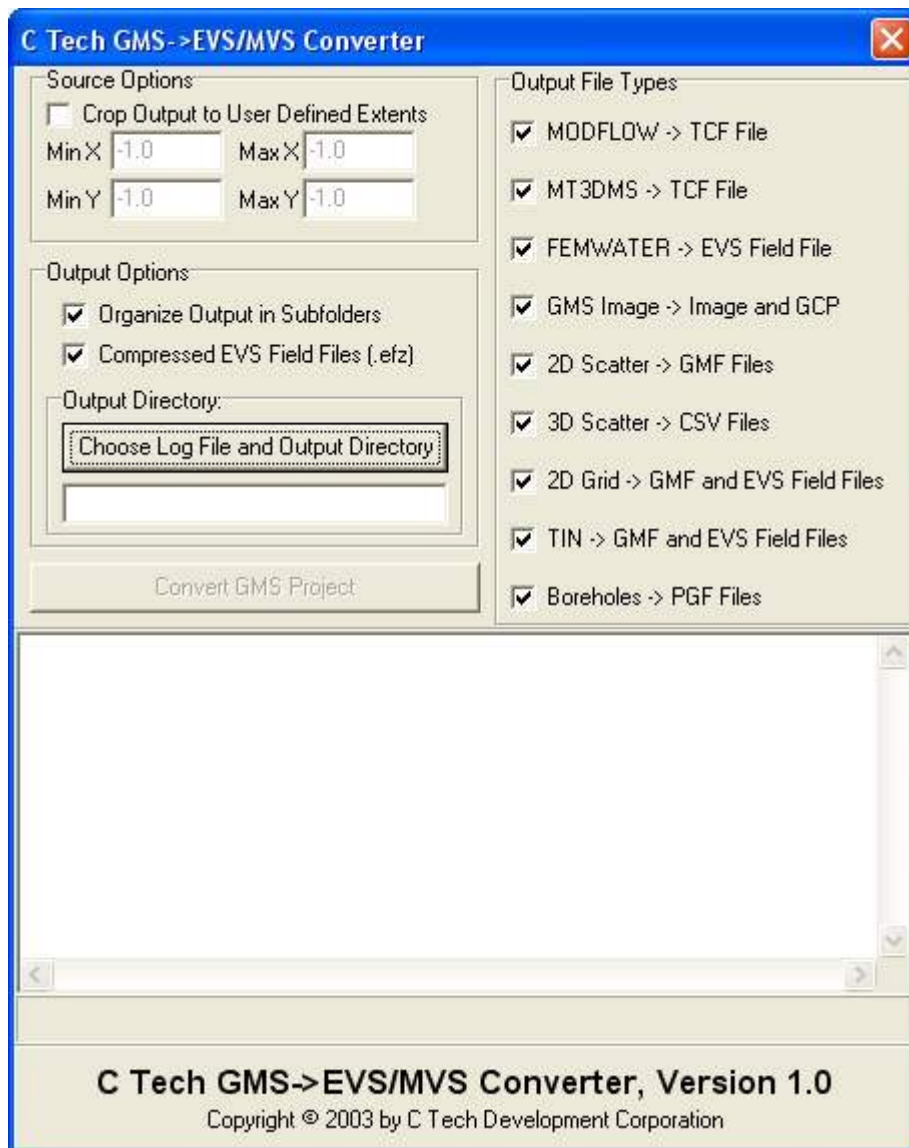
There is one source control option. By clicking on Crop Output to User Defined Extents the input from GMS can be focused on areas of interest to the user. These extents are assumed to be valid when entered by the user. No Error checking is done. This works only on MODFLOW, MT3DMS, and FEMWATER conversions.

There are two options for output:

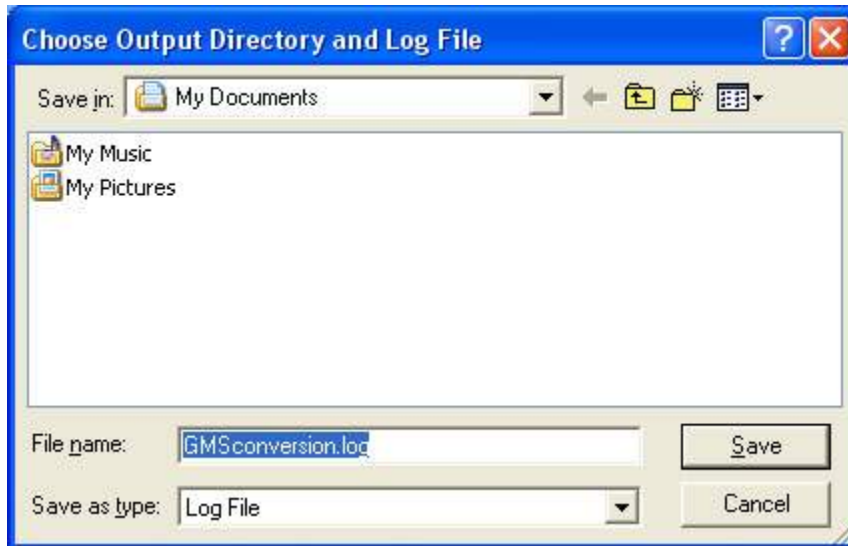
1. **Organize output to subfolders:** This allows the user to keep track of which specific part of the project was converted by placing the appropriate converted files in new subfolders based on the data type (e.g. MODFLOW, IMAGE, etc.).
2. **Compressed EVS Field Files(*.efz):** This option changes all EVS field files (*.eff) into their compressed form (*.efz) for space saving purposes.

To run the converter

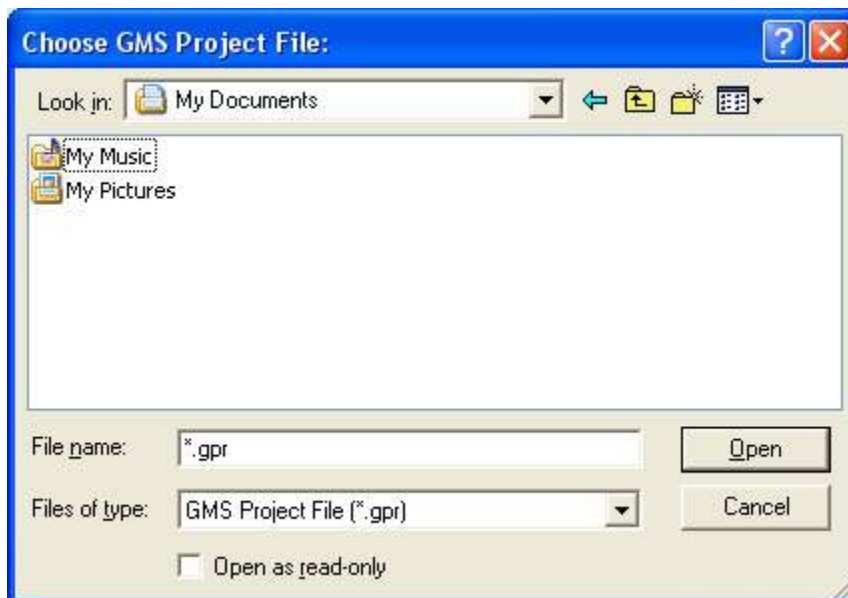
1. Select all appropriate options and file types to be converted.



2. Click the Choose Log File and Output Directory button to choose the destination for the log file and converted files/folders. The log file is a status report of the converter which keeps track of all errors and files created.

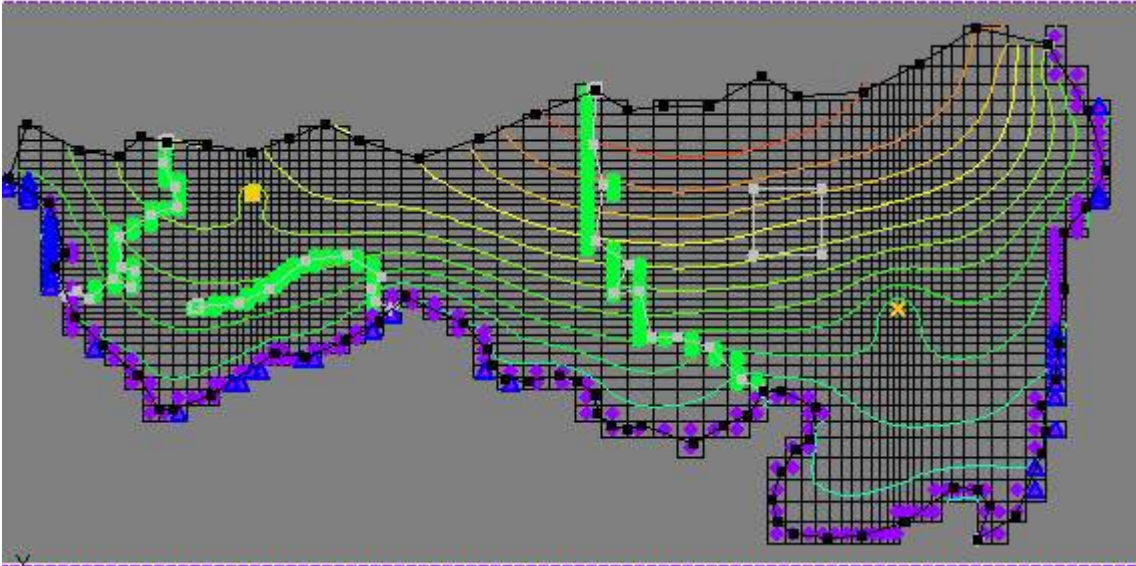


3. Next click on the Convert GMS Project button and select the GMS project file (*.gpr) to be converted. All files should be created and shown in the Status window, click Done to finish.

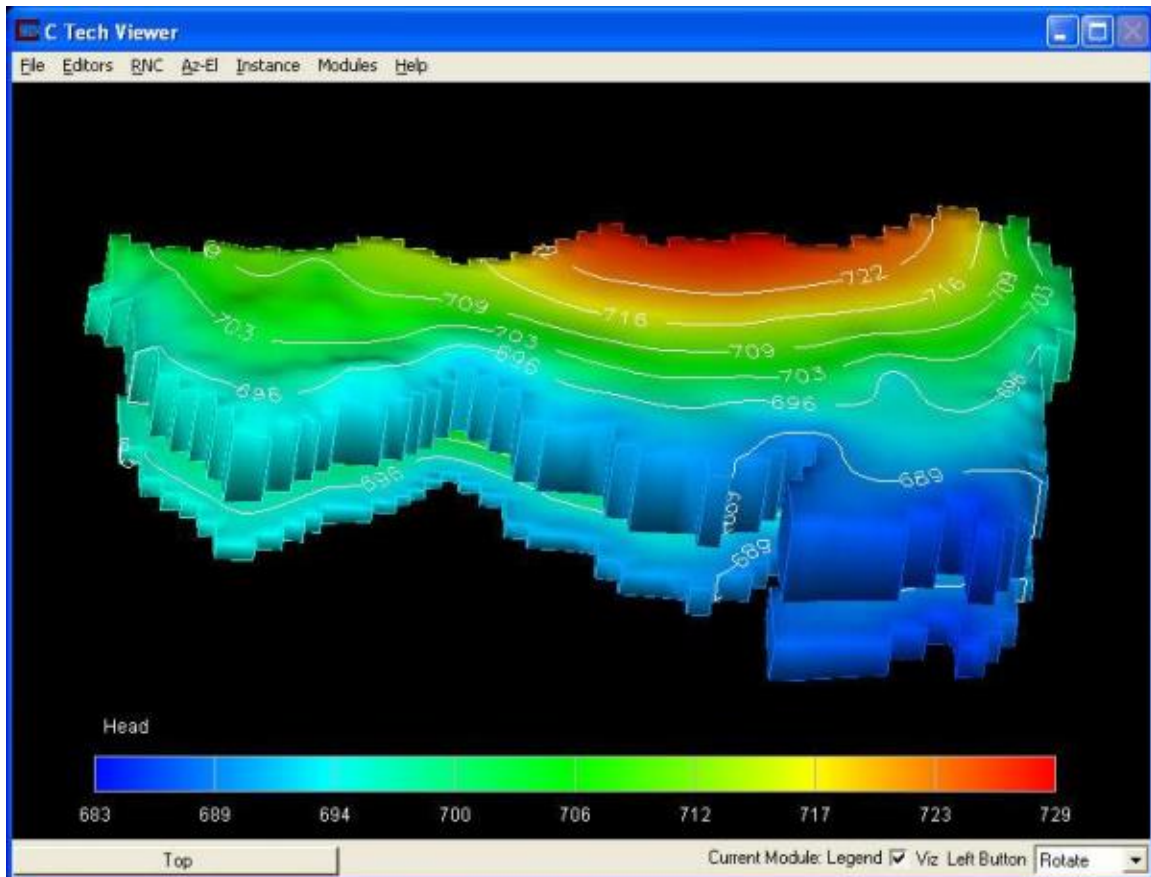


Example:

The following sample MODFLOW model in GMS was converted using the converter:



Upon running the GMS to EVS converter, a compressed EVS field file was created. This file can then be used in a variety of ways in EVS, as shown in the picture showing head values below:



VM_to_EVS

General ToolFunction

The VM_to_EVS tool can convert a Visual MODFLOW project to a several useful EVS file formats. This tool is not supported by C Tech Development Corporation, but is useful for visualizing VMOD projects.

ToolControl Panel

The control panel for the VM_to_EVS converter can be seen above. The first step in converting a project is to select a Visual MODFLOW project (*.VMG) with the **Select Visual MODFLOW Project Folder and VMG File button**. The converter will then locate all additional VMOD project files (as shown below). The user can also crop the extents of the project that has been read in by changing the values in the Starting Row/Column and Ending Row/Column fields.

Visual MODFLOW® to EVS Conversion

Visual MODFLOW® Project Folder and VMG File

Select Visual MODFLOW Project Folder and VMG File

C:\CTech\Data\VMOD\VMOD_Test.vmg

Visual MODFLOW® Files for Conversion to EVS

VMG File

C:\CTech\Data\VMOD\VMOD_Test.vmg

VMP File

C:\CTech\Data\VMOD\VMOD_Test.vmp

HDS File

C:\CTech\Data\VMOD\VMOD_Test.HDS

FLO File

C:\CTech\Data\VMOD\VMOD_Test.FLO

MODFLOW Model Dimensions

Rows 40 Columns 50 Layers 1

The next step is to select the type of output that is desired. The first (and recommended) option is to create a set of EFF/TCF/DWR files. To use the MODPATH module the cell components Head, Hydraulic Conductivity, Effective Porosity, and Groundwater Flux must be selected as shown below.

Select EVS File Type and Included Data

☒ EFF/TCF/DWR
 ☐ UCD
 ☐ GMF

☐ Head (node data)
☐ Model Layer (Required to Explode Visualization) (node data)
☐ Velocity in Length/Time Units (node data)
☐ Distance to Water Table (node data)
☒ Head (cell data)
☒ Hydraulic Conductivity (cell data)
☒ Effective Porosity (cell data)
☒ Groundwater Flux (CCF) in Length³/Time Units (cell data)
☐ Concentration (node and cell data)

EVS Files Base Name

VMOD_Test

EVS Files Folder

Select Destination Folder For EVS Files

Select Units Used in MODFLOW Model

Length	Time	Mass	Concentration
feet	day	pounds	micrograms/Liter

To use the streamlines module the Velocity component must be selected. The MODFLOW units should be checked to ensure that the right coordinates are used. The final step is select the Convert button (as seen below), when the progress bar has finished then select the Quit button to exit.

Progress in EFF/TCF File Creation

Log:

Convert

Quit

Images to Animation

images_to_animation incorporates the functionality of Gromada's VideoMach into an easy to use tool.

Animation CODECS try to default to TechSmith Screen Capture Codec, or, if that doesn't exist, to the HuffYUV codec (both are lossless).

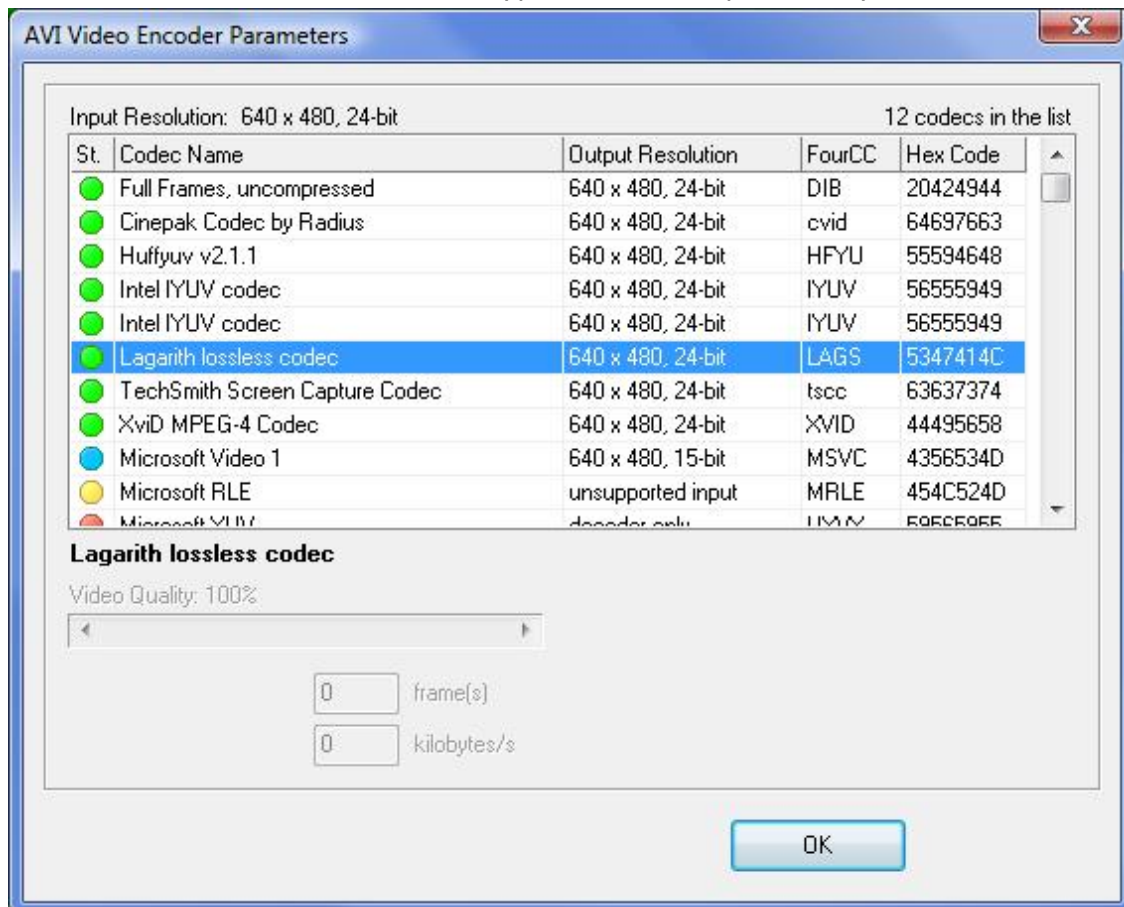
images_to_animation supports several animation file formats including:

1. AVI Windows Audio Video Interleaved (avi)
2. Windows Media Video - WMV produces some of the smallest files and this output is virtually guaranteed to run on any up to date Windows computer. However, the quality can be poor unless the data rate is increased.
3. MPG Moving Pictures Expert Group, MPEG-1 (mpg,mpeg)
4. HAV High quality Audio Video (hav) HAV is a format that can be played with the freeware program Imagen (formally HAV player). This format has some distinct advantages, specifically it uses lossless compression. This results in the highest quality output. Surprisingly, HAV files are often as small as or smaller than lower quality AVI or MPG files.

When you run, the first thing that will happen is a window will pop up which allows you to set the Output Animation Codec:

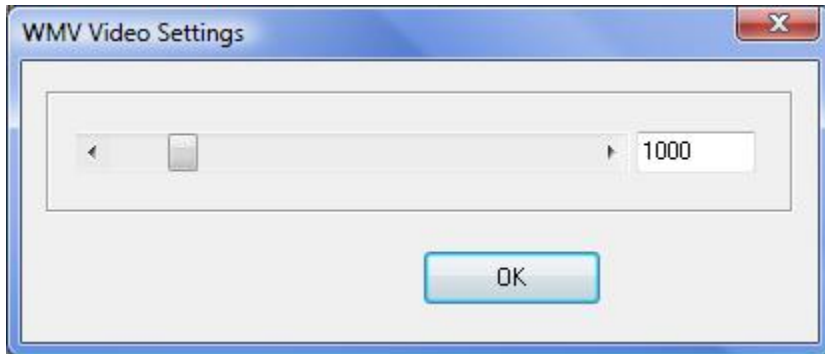
This shows the selected default CODEC for the type of animation file you've created. There are four supported bitmap animation file types which are AVI, WMV, MPG (MPEG), and HAV.

The AVI parameters window has the most options. The first thing to take notice of is the colored circles on the leftmost column. Only those colored GREEN are fully supported. Blue and yellow may work but will likely have issues and red is not recommended. With each CODEC type, additional options may be available.

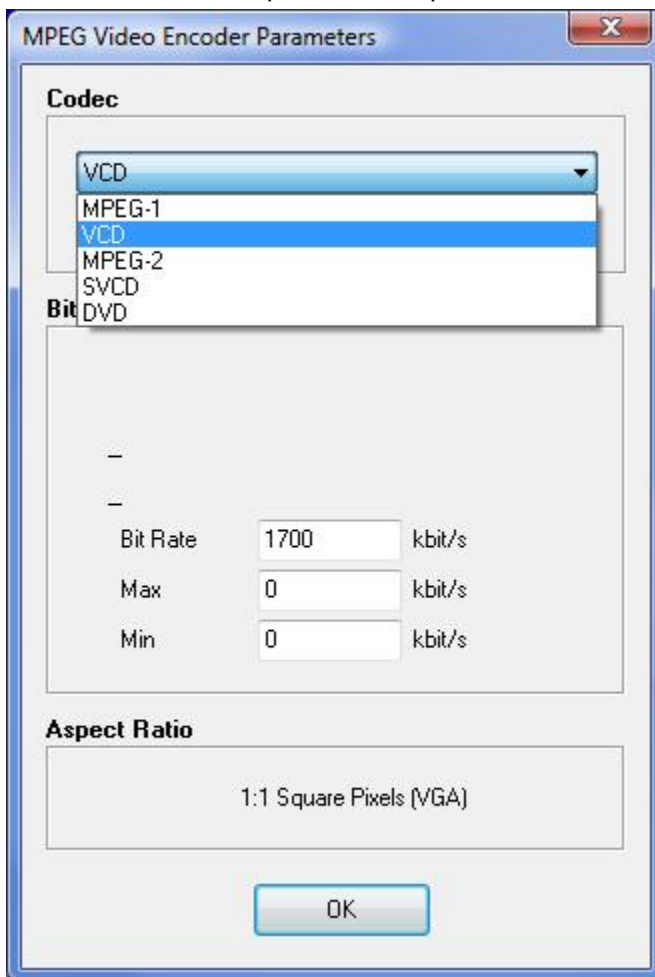


The Windows Media Video (WMV) CODEC has only one parameter which is not labeled (this is a Windows generated panel). The parameter is bit rate which defaults to

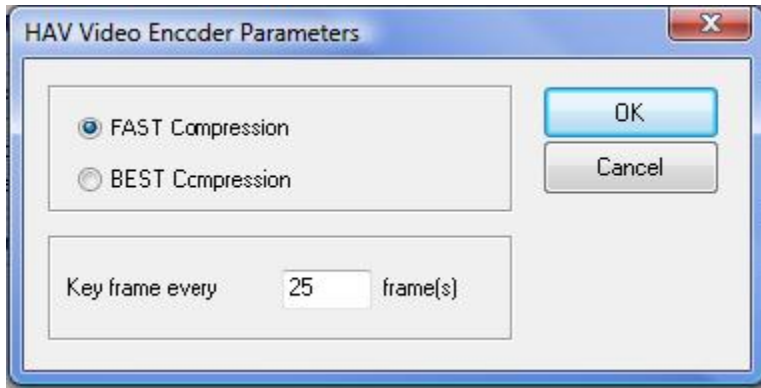
1000 and has a max value of 8,000. Higher values give higher quality and a larger file size.



The MPEG encoder parameters provides several useful options including DVD.



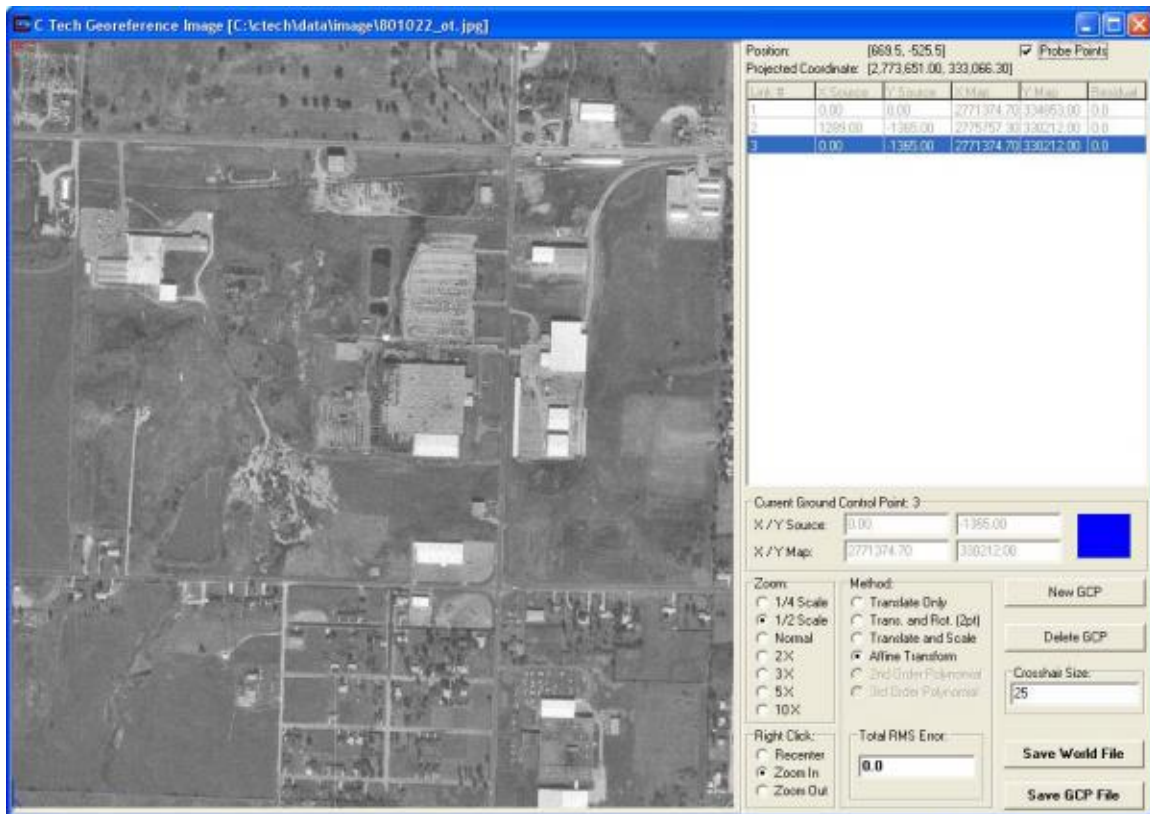
The HAV format has only two types and the ability to set the key frames.



Georeference Image

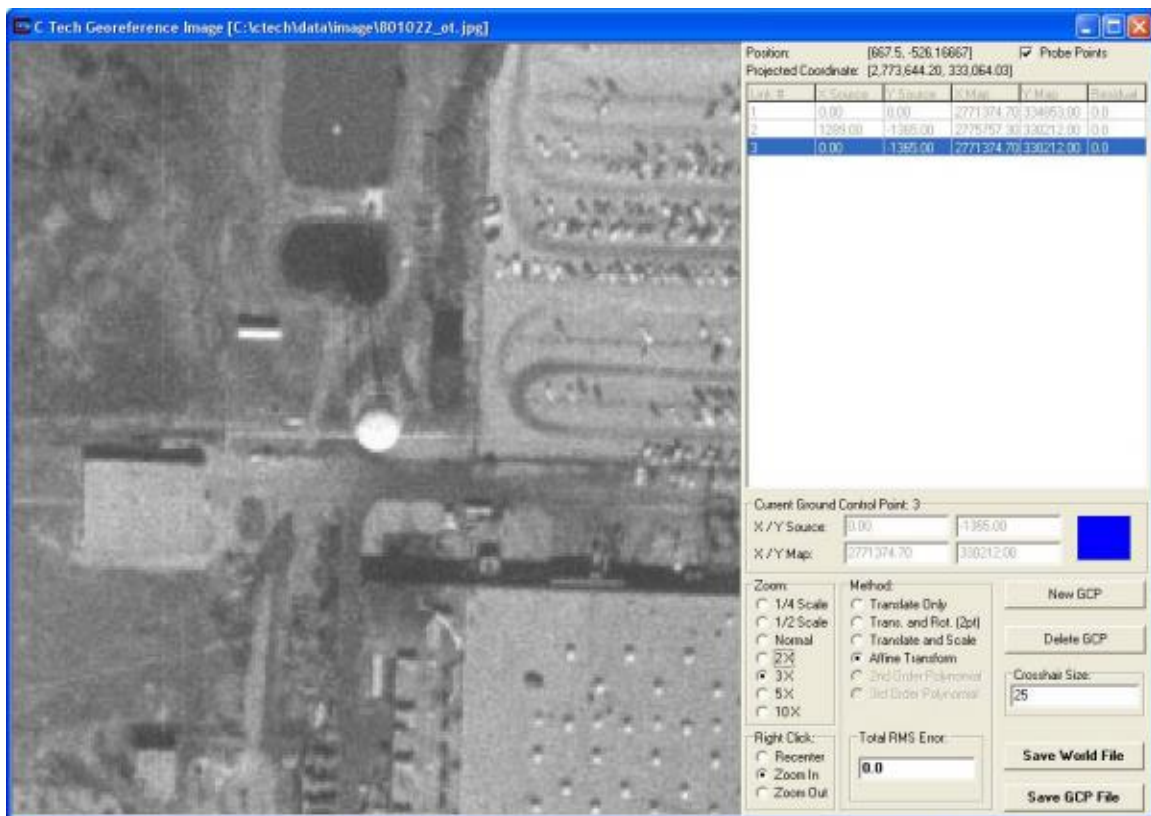
GeoreferenceImage is a standalone utility program which can create world files or .gcp (ground control point) files for images. It supercedes the georeference_image module. The .gcp files are compatible with ArcGIS image link files, but use a .gcp extension (ArcGIS defaults to .txt extension instead of .gcp, but these are compatible). **GeoreferenceImage** will allow you to view total root mean squared error given the number of Ground Control Points you have and the selected texture_map mapping option. This standalone utility can be run with a single button click from [Read Image](#) or by running from the *tools* folder in the C Tech program group.

When you run Georeference Image from the *tools* folder in the C Tech program group it will initially prompt you for an image to open. If run from Read_Image, it will open the image already selected in Read_Image. The example shown below uses the image 801022_ot.jpg which has a corresponding world file 801022_ot.jpw. Georeference Image automatically reads the world file and creates three ground control points (GCPs) as shown in the figure below.



If you have additional ground truth data, you can add additional GCPs to enhance the accuracy of the image projection.

When initialized with a world file, you can use Georeference Image to determine the x-y coordinates of any point on the image. By zooming in on the image (as shown below) it is possible to get accurate coordinates of individual pixels.



Index

.
.eff, 296, 319, 326
.efz, 297, 317, 326
.inp, 317

—
_2D_Overlay, 361

1

1.0e9, 156, 199, 202, 206, 207
 value, 157
1280, 128
 set, 128

2

2d, 344, 386
2D, 128, 133, 141, 152
2D mesh from 1D lines, 390
2D space, 270
2D surface of triangular polygons, 363

3

3d, 344, 386
 types, 140
3D, 127, 129, 133, 140, 141, 152, 156
3-D, 156
3-D, 210
3-D, 226
3D Adaptively Gridded Model, 140
3-D dataset, 156
3D Kriged Model, 153
 Slice Planes Passing Through, 152
3D mesh from 2D surfaces, 237
3D PDF, 371
3D Plume Showing Various Shading
 Methods, 149
3d_exp.geo, 211, 215

3d_geology_map, 237, 390
3D_plume, 289
3D_Plume, 286, 288, 290
3D-refinement, 140
3-space, 256

4

4D Interactive Model Player, 396

5

50by5.csv, 164, 166

A

adaptive, 388
Adaptive Gridding, 138
Adaptively Gridded Convex Hull Grid,
139
Adaptively gridding, 138
add, 270
add logo, 270
adjust_transparency, 256
advect_surface, 386
Advection, 258, 333, 385
advectord, 258, 333, 385, 386
aidv, 176
All Axis Variables, 268
Alpha Range Model, 256
Alt-left mouse action, 236
AN EXAMPLE FILE FOLLOWS, 186,
189, 201, 205
Analyte, 125
analytical data, 163
animate slider, 364
 cutplane or plume changes, 364
animation, 349, 350
Animation, 351, 361
Animation Time Steps, 258, 333
annotation, 270
apdv, 168

- appending to a gmf, 236
- Appendix 1.A, 125
- Appendix 1.B, 125
- Appendix 1.C., 125
- application_notes, 391
- area, 286, 335, 384
- area calculations, 264
- area_cut, 286
- area_integrate, 264
- areal, 286, 290, 335
- ASCII, 156
- ASCII Files, 125
- aspect, 259
- ASPECT, 285
- Assymmetric, 131
- AutoCAD, 313
- automatic incrementing, 364
- Average, 265
- avi, 157
- axes, 268
- Axes, 268
- Axis, 268
- Azimuth and Elevation, 39

B

- B, 125
- Back To Front (BTF), 256
- Background Shading, 149
- basemap, 210
 - use, 210
- bench cutting, 288
- Binary Output of an EVS/MVS Mesh, 325
- BLDG, 315
- Bldg file, 315
- BLUP, 149
- bmp, 157
- Boolean, 281, 282
- borehole, 165, 168, 173, 175, 179, 184, 199, 202
 - surface, 164, 166, 172, 174, 177, 182

- Borehole, 128, 236, 323
- Borehole Geology, 156
- boreholes, 164, 166, 177, 182, 202, 205
- boring, 206
 - C, 206
- Boring Tubes, 131
- borings, 156, 199, 202, 207, 208, 210
 - coordinates, 199, 202, 207
 - number, 156, 199, 202, 210
 - order, 156
 - posting, 156
 - represent, 128
 - representations, 128
- Borings, 125, 130, 249
- borings WEL-67, 201, 204
- BOT, 206, 208
- Both kriging, 149
- boundaries, 286
- boundary, 286
- bounding faces, 286
- bounding mesh box, 286
- bounds, 267, 268, 286
- Bounds grey, 267
- Bounds On, 267
- Bounds rendering, 361
- Box Axes, 268
- Bright-Saturate, 268
- BTF, 256
- BTF volume, 256
- buffer, 288
- Buildings, 315
 - Sample Buildings File, 316

C

- C, 125, 164, 166, 171, 174, 177, 182, 206, 207
 - boring, 206
- C Tech Development Corporation's Environmental Visualization System, 125
- C Tech's, 126
- C Tech's EVS, 128

- C1, 209
- C2, 209
- C3, 209
- CAD, 157, 327
- CAD Files, 327
- California, 129
- capture_zone, 333
- Cartesian, 125
- causes, 207
 - kriging, 207
- cell, 264
- Cell centroids, 367
- cell data, 296, 343, 366
- Cell Type, 132
- cell_centers, 367
- cell_to_node, 366
- cells, 367
- centroid of hexahedron, 367
- change_minmax, 255
- Characteristics, 128
- Chem Density, 261, 264
- chemistry, 368
- Chemistry, 156
- Chemistry Files, 156
 - Fence Sections, 156
- Chemistry Sections, 390
- Chemistry Time Files Example, 177, 182
- Clay Lens, 126
 - Geologic Hierarchy, 125
- closed, 340, 341
- CMY, 369
- CMYK, 369
- CMYU, 369
- Coast Facility Showing Contaminant Plume, 151
- Cokriging, 149
- color, 267
- Color, 151
- Color datamap, 267
- Color_legend, 267
- combine, 362
- combine_comp, 246, 294
- combine_components, 246, 294
- combine_geology, 246
- combine_vect, 331
- combined rendering, 361
- combining data, 246, 294
- combining data components, 246, 294
- combining Dx, 331
 - Dy and Dz, 331
- combining vectors, 331
- Common Cell Types, 133
- company, 270
- Complex Geologic Hierarchy, 127
- component, 259
- comprise, 35, 37
 - EVS Application, 35, 37
- concentration gradients, 332
- concentrations, 278
- Cone Minimum Radius, 289
- Cone Penetration Test, 249
- Connect the modules, 58
- Consistent Coordinate Systems, 159
- Continuous_sketch, 339
- contour labels, 278
- contour_data, 255
- contours, 278
- Control_Fly_Through, 349
- converter, 401
- Convex Hull, 137, 226
- Convex Hull Grid, 137
- Coord, 265
- coordinates, 199, 202, 207
 - borings, 199, 202, 207
- Coordinates, 265
- CorelDrawTM, 156
- Corner, 142
 - Figure 1.18, 142
 - Figure 1.19, 142
- cost, 266
- Covariates, 149
- CPT, 249
- create_fault, 344
- create_fault_surface, 344

- Creating, 128, 132, 141
 - high-resolution, 128
 - TINs, 141
 - two-dimensional, 133
- crop, 285
- crop_and_downsize, 285
- Cross Section A-A, 210
 - Geology File, 210
- Cross Section B-B, 210
 - Geology File, 210
- Cross Sections, 383, 387, 390
- cross_section_tubes, 341
- Cross3D, 333, 336, 343
- csv, 156
- Csv, 390
- csv file, 156, 164, 166, 171, 174, 177, 182
- Csv file, 390
- ctech\data\chemistry, 164, 166, 177, 182
- cut, 279, 287
- Cut 3D Kriged Model, 154
- cutting, 288
- cylinder, 288, 289, 315

D

- D.L., 149
- data, 264, 362, 364
- data components, 259, 332
- Data Content Requirements, 125
- data domain, 285
- Data Requirements, 79
- data values, 332
- data_translate, 296
 - translate: extrude, 296
- Datamap, 151, 267
- Datamap Editor, 368
- Datamaps, 361
- dataset, 156
- dBaseIV, 156
- decommissioning, 266
- Delaunay, 141

- form, 141
- DEM, 339
- describe, 156
 - 3-D, 156
- Desktop Area, 128
- detection limit, 158
- determines the slope and aspect of a surface, 259
- Dewatering, 131
- dimensionality, 279
- Dimensionality, 132, 152
- dip, 320, 321
- Dipping Strata, 207
- Direct Data Visualization, 128
- Discontinuities, 149
- displace_block, 248
 - translate: faults: tears: surf_cut, 248
- display, 321
- Display, 323, 361
- Display Image, 350
 - Display_Image: DisplayImages, 350
- Display nonvertical borings, 249
- DNAPL, 135
- domain, 285
- Domain, 351, 361
- draw_lines, 338
- drive_glyphs, 343
- DWG/DXF File button, 327
- DXF, 289
- DXF draping, 383
- DXF files, 157, 313, 327, 344, 383
- DXF Import, 327

E

- Eastings, 200, 204, 206
- edges, 276, 286, 339
- ELEV, 164, 166, 177, 182
- elevation/depth, 186, 188
- ELF, 318
- ELF format, 318
- Equal, 152

- isolevel, 152
- ESRI Shapefile, 156
- ESRI shapefiles, 156
- estimation, 226
- Evaluating, 152
 - subsetting, 152
- EVS, 34, 35, 125, 135, 137, 156, 164, 166, 172, 174, 177, 182, 199, 202, 207, 210, 211, 215
- EVS Application, 35, 37
 - comprise, 35, 37
- EVS Data Input/Output Formats, 156
- EVS Environment, 34, 35
 - Getting Familiar With, 34, 35
- EVS field, 296, 326
- EVS Lines File, 318
- EVS Main Window, 35, 36
- EVS Message Console, 185, 187
 - printed, 186, 188
- EVS viewer, 369
- EVS.SYM file, 323
- EVS/MVS geology, 236
- EVS's two-dimensional kriging, 146
- EXAMPLES, 202, 205
- excavation, 289
- excavation cutting, 287
- excavation volumes, 289
- exceed
 - isolevel, 152
- Exceed, 152
- Exit EVS, 56
- explode_and _scale functionality, 286
- explode_and_scale, 251
- expression, 284
- Extents, 265
- external, 276
- external_edges, 276, 286
- external_faces, 276, 280, 281
- external_kriging, 236
- extract_component, 259
- extrude, 343

F

- faces, 276, 315
- fade, 350
- fault, 344
- fault displacement, 287
- Fence Diagrams, 387, 390
- fence section, 368, 390
- fence sections
 - Chemistry Files, 156
 - Geology Files, 156
- Fence Sections, 156
- fence_geology, 387
- fence_geology_map, 390
- fences, 387
- Fences, 383
- field, 362, 363
- Field File Formats, 296, 325
- field math, 287
- field_math, 290
- fields, 362
- fields only, 285
- Fields only, 286
- Figure 1.1, 125
- Figure 1.13, 137
- Figure 1.14, 138
- Figure 1.16, 140
- Figure 1.18, 142
- Figure 1.19, 142
 - corner, 142
- Figure 1.2, 125
- Figure 1.20, 143
- Figure 1.21, 143
- Figure 1.22, 145
- Figure 1.23, 146
- Figure 1.24, 147
- Figure 1.26, 151, 152
- Figure 1.3, 127
- Figure 1.3., 127
- Figure 1.30, 153
- Figure 1.5, 129
- Figure 1.8., 132
- Figure 1.9., 133

- Figure Y, 125
- FIGURES, 199, 202
- file, 388
- FILE FORMATS, 156
- file_output, 388
- Fill, 147
 - three-dimensional, 147
- find_port, 225
- Finite Difference, 135
- Finite Element Grid, 325
- Finite-difference, 226
- Finite-difference gridding, 226
- Flat Shading, 149
- Flat-Shaded Delaunay TIN, 142
 - Geologic Surface, 142
- Flat-Shaded Subdivided TIN, 144
 - Geologic Surface, 142
- float_math, 362
- flow, 256, 258, 329, 330, 331, 332, 333, 385, 386
- flow directions, 256, 258, 329, 330, 331, 333, 385, 386
- flow vectors, 256, 258, 329, 330, 331, 332, 333, 336, 343, 385, 386
- flow velocities animation, 258, 333, 385
- flowpath animations, 343
- fly, 349
- fly_through, 349
- fly-through, 349
- fly-thru, 349
- footprint, 283
- Form, 141, 148
 - Delaunay, 141
 - IDWA, 148
- fplane, 344, 386

G

- Generate Axes, 268
- geo, 156, 199, 202, 211, 215
- Geo, 226, 264
- geo file, 156, 199, 202, 210
- GEO File button, 226

- Geologic File Example, 206, 207
 - Outcrop of Dipping Strata, 207
 - Sedimentary Layers and Lenses, 206
- Geologic Hierarchy, 125
 - Clay Lens, 126
- Geologic Indicator Kriging, 128
- Geologic Input Files, 202, 205
 - Other Examples, 202, 205
- geologic layers, 226, 237, 238, 239, 364, 387, 391
- geologic model, 226, 237, 238, 239, 390
- Geologic Sections, 387, 390
- Geologic Surface, 142, 226, 237, 390
 - Flat-Shaded Delaunay TIN, 142
 - Flat-Shaded Subdivided TIN, 144
 - Gouraud-Shaded Subdivided TIN, 145
 - Solid Contour TIN, 146
- Geologic_Indicator_Kriging, 239
- Geologic_Surface, 239
- Geologic_Surfaces, 238
- geologic_surfmap, 383
- geology, 237, 238, 239, 368, 390
- Geology, 226, 387
- GEOLOGY FILE EXAMPLES & FIGURES, 199, 202
- Geology Files, 156, 210
 - Cross Section A-A, 210
 - Cross Section B-B, 210
 - Fence Sections, 156
- Geology Files for Production of a Fence Diagram, 210
- Geology Multi-File, 156, 211, 215
 - gmf, 211, 215
- Geology Multi-Files, 226
 - ctech_example.gmf, 219
- geology_to_raster, 328
- geology_to_structured, 237
- geology_to_vistas, 329
- Geometries, 333, 336, 343
- geophysics, 320, 321
- Georeference Image, 406

- georeferenced_output, 348
- Geostatistical Methods, 149
- Getting Familiar With, 34, 35
 - EVS Environment, 34, 35
- Getting Familiar With The EVS Environment, 34, 35
- GIF, 349
- GIK, 128, 185, 187
- Global Positioning Satellite, 125
- glyph, 336, 343
- Glyphs, 128, 333, 336, 343, 367
- gmf, 156, 199, 202, 211, 215
- Gmf, 226, 236
- gmf file, 156
- Gmf file, 226, 236
- GMS, 335, 336
- GMS Project File Converter, 392, 396
- Gou, 142
- Gouraud, 142, 149, 256
- Gouraud Shading, 149
- Gouraud-Shaded Delaunay TIN, 143
 - Geologic Surface, 142
- Gouraud-Shaded Subdivided TIN, 145
 - Geologic Surface, 142
- GPS, 125
- gradient, 332, 334
- gradient directions, 256, 258, 329, 330, 331, 333, 385, 386
- gradients, 332
- Gravel, 126
- Greyscale, 267
- grid, 285, 286, 344, 386
- grid boundary, 286
- grid cropping, 285
- grid math, 290
- Gridding, 128, 132, 134, 138, 146
- grids, 276
- ground water flow, 256, 258, 329, 330, 331, 332, 333, 385, 386
- groundwater, 163
- Groundwater Vistas, 335, 336
- GroupObject, 345, 361
- Guidelines for Z Print, 379

- gwc, 171, 174

H

- Handling Non-Detects, 158
- head, 333
- hierarchy, 245
- high order, 363
- High-resolution, 128
 - creating, 128
- Histogram, 265
- horizon_ranking, 245
- HTML, 370

I

- I-1, 134
- IDWA, 148
 - form, 148
- illuminated_lines, 256
- Image, 348
- Image Generation Manual, 369
- Image processing, 348
- image_transition, 350
- Images, 369
- images_to_animation, 403
- includes, 157
 - output_images, 157
- Increment, 364
- Indicator, 388
- Input GMF File, 236
- Instance Krig_Z, 57
- interactive, 271, 364
- interactive_labels, 271
- interp_cell_data, 368
- interp_data, 295
- interpolate values, 295
- interpolation, 351, 361, 363
- intersection, 280, 281
- Inverse Distance Weighted, 148
- irregular, 363
- irregular grids, 226
- isolevel

- equal, 152
- exceed, 152
- Isolevel, 152, 280, 281
- isolines, 156, 278
- Isolines, 145, 152
- Isopachs, 145, 154
- isosurfaces, 156, 280, 281
- isovolume, 156, 286, 288, 364
- Isovolume, 152
- isovolumes, 280, 281

J

- J, 133
- J-1, 134
- JPEG, 349

K

- K, 133
- K-1, 134
- krig_2d, 230
- krig_3d_Geology, 211, 215, 226, 247, 248
- krig_3d_Geology Module Hints, 226
- krig_fence, 200, 204, 387, 390
- Krige, 149, 226
- Kriged 2D Convex Hull Grid, 147
- kriging, 158, 207, 247, 248
 - causes, 207
- Kriging, 146, 149, 227, 388, 390

L

- Label Decade Rounding, 268
- Labeled Isolines, 153
- Labeled isolines on 10, 146
- labels, 271
- Lat-Lon, 125
- layer edges, 276
- layer_from_surface, 238
- Layering, 251
- layers, 237, 238, 239, 247, 248, 391
- legend, 267

- Lenses, 206
- Line, 199, 203, 206
- Line 3+n, 164, 166, 177, 182, 186, 189, 201, 205
- linear, 363
- lines, 276, 278, 286, 329, 340, 341
- Load Glyph, 322
- load_evs_field, 296, 317
- locked, 270
- Logarithmically, 125
- logo, 270
- loop, 364
- Lower Sand, 126
- Lsand, 201, 204
- Lsnd, 199, 202

M

- m, 199, 202
- magnitude, 332
- make_geo_hierarchy, 236
- make_single_layer, 248
- map DXF lines to a ground surface, 344, 383
- Map Spheres, 164, 166, 177, 182
- Map_Spheres, 164, 166, 177, 182
- mask_geology, 290
- Material Color, 208
- material color numbering, 236
- material_mapping, 246
- material_to_cellsets, 364
- math, 290
- math functions, 290
- math operations, 290, 362
- Max, 265
- Mean, 265
- merge, 362
- MERGE, 248
- merge_fences, 383
- merge_fields, 362
- Mesh, 317, 325
- Mesh visualization, 367
- Min, 265

- Mineshaft (as cylinder), 289
- Minimum Alpha, 256
- MIP, 163
- model boundary lines, 276
- model display, 276
- model grid, 276
- Model Subsetting, 152
- Modflow, 258, 333, 385
- MODFLOW, 131
- modflow_converter, 335, 336
- MODFLOW2000, 335, 336
- MODFLOW98, 335, 336
- modify_data_3D, 364
- modpath, 331, 386
- MODPATH, 386
- modpath_advector, 386
- Module Status Icons, 223
- Module Sublibrary, 222
- mouse interactive areas, 286, 290, 335
- mouse interactive geology building, 236
- mouse interactivity, 236
- Mouse Transformation, 38
- move
 - three-dimensional, 129
- Move, 129
- multi-valued in the z direction, 287
- MVS only, 287, 289, 339

N

- N, 164, 166, 177, 182
- Name, 206, 208
- Nd, 125
- NetCDF, 296, 317, 326
- NetCDF file, 296, 317, 325
- network, 363
- Network Editor, 35, 36
- nodal, 362
- Nodal Data, 265
- Nodal_Data, 265
- node data, 296, 343, 345, 366

- Nodes, 132
 - Number, 132
- Non-detects, 158
- Nonstationarity, 149
- north, 269
- North, 152
- North arrow, 269
- Northings, 200, 204, 206
- ntsc, 384
- NTSC, 156
- number, 156, 199, 202, 210
 - borings, 156, 199, 202, 210
 - Nodes, 132
- Number, 132

O

- Object Manipulation, 368
- offset, 345
- Offset, 137
- offset grid coordinates, 345
- on-screen titles, 270
- OpenGL, 256
- optimizing triangle size, 339
- order, 156
 - borings, 156
- Orthoslice, 286
- orthoslice, 286
- Other Examples, 202, 205
 - Geologic Input Files, 202, 205
- output, 369, 388
- output_images, 157, 369
 - includes, 157
- over, 349
- overburden, 289
- Overburden isocomponent, 289
- Over-excavation, 290
- overlay, 270
- overlay_aerial, 346
- overscan, 384

P

- particle flow, 258, 333, 385
- PBM, 348
- PGF, 194
- pgf file, 236
- Pgf file, 236
- PGF File Examples, 190
- photograph, 351
- Pickability, 271, 361
- place_glyph, 343
- place_text, 271
- Plume, 152
- Plume Visualization, 152
- plume_shell, 252
- Plumes, 281, 282
- Point sampling, 256
- Points, 323
- Polyline, 339
- polyline_spline, 339
- polylines, 340, 341
- posting, 156
 - borings, 156
- Postscript, 369
- Pre Geology File, 185, 187
- Pre_Geology, 185, 187, 236
- Pregeology, 128
- pre-geology file, 236
- Pregeology file, 128
- Print Quality Tips, 370
- printed, 186, 188
 - EVS Message Console, 185, 187
- priorities, 245
- project_field, 345
- Projecting File Coordinates, 159
- Property Files, 156

R

- Range Alpha, 256
- rankings, 245
- raster_to_geology, 315
- Ray-tracer, 256

- Ray-tracer volume, 256
- Read Field
 - EFF_File, 300
- Read_DXF, 383
- Read_Image, 348
- read_lines, 317, 318, 329
- Read_Multi_TCF, 353
 - TCF_File, 351, 354
- Read_netCDF, 296, 317
- Read_TCF, 351
 - TCF_File, 351, 354
- read_vector_gis, 315
- Real, 164, 166, 178, 183
- Recording (Capturing) 4DIM Files, 381
- rectangle, 288
- Rectilinear, 226
- Rectilinear Grids, 133
- regional_averages, 335
- regions, 286, 290, 335
- removal, 266
- Renderer Selection Software, 369
- rendering, 361, 367
 - characteristics, 141
- Rendering, 141
- rendering velocities isovolumes, 332
- Represent, 128
 - borings, 130
- Representations, 128
 - borings, 130
- resolution, 344, 386
- RGB, 370
- Rockhead, 128
- rotate, 345
- rotated, 288
- Run the Network, 63
- Runge-Kutte, 258, 333, 385
- Run-Length-Encoded, 369

S

- s, 156
- S, 133, 151, 152
- Sacramento, 129

- SAD, 321
- SAD file, 321
- safe, 384
- Sand, 126
- save, 329
- Save the network, 37
- save_evs_field, 325
- savings, 266
- scale, 369
- scale data, 267
- Scaling, 251
- Scanline images, 348
- scattered, 363
- Seamlessly, 127
- Section A-A, 210
- Section B-B, 210
- Sections, 383
- Sedimentary Layers, 206
- See Figure 1.10, 134
- seepage_velocity, 333
- select_cells, 286
 - cells, 286
- select_data, 259
- Semivariogram, 149
- set
 - 1280, 128
- Set, 128
- Set Contour Levels, 278
- SGI Image, 348
- shape_cut, 288
- shapefile, 296, 343
- Shift key, 368
- shows
 - two-dimensional, 137
- Shows, 138
- shrink_cells, 367
- Significant, 152
- Similarly BOR-72, 201, 204
- site
 - three-dimensional representations, 151
- SITE_ID, 164, 166, 177, 182
- slice, 278, 279, 286, 364
- Slice plane, 286
- Slice Planes Passing Through, 152
 - 3D Kriged Model, 153
- Slicing, 152
- SLOPE, 285
- slope_aspect_splitter, 284
- Smooth, 339
- smoothing lines, 339
- software rendering, 256
- Soil, 163
- Solid Contour TIN, 146
 - Geologic Surface, 142
- Solid Contours, 149
- special effects, 350
- Sphere Connectors/color tubes, 236
- Spheres, 249
- Splining, 148, 339
- Splining's, 148
- Start
 - EVS, 34, 35
- starting points, 344, 386
- Statistics, 265
- step through increments, 364
- streamline_surface, 330, 331, 386
- streamlines, 258, 329, 330, 331, 332, 333, 340, 341, 385, 386
- strike, 320, 321
- strike_and_dip, 320, 321
- subset, 247, 248, 280, 281
- subset of cells, 286
- subset_layers, 247, 248
- subsetting, 156, 285, 286, 290, 335
 - evaluating, 152
- Subsetting, 152
- sum ports, 362
- Sun Raster, 349
- surf_cut, 287, 289, 339
- surf_cut example discussion, 287
- surf_cut_examples, 288
- surface, 164, 166, 177, 182, 368
 - borehole, 165, 168, 179, 184
- surface operations, 290
- Surface rendering, 256

- surfaces, 236, 247, 248, 339, 344, 383
- Surfaces, 149
- surfmap, 344
- Sym, 323
- SYM file, 323
- symbols, 323
- Symobls, 325

T

- Table 1.1, 132
- targa, 156
- Targa, 369
- TEST_SYM.GEO, 325
- Tet, 132
- Tetrachloroethylene, 164, 166, 177, 182
- texture, 315, 351
- texture cylinder, 351
- Texture Mapping, 151
- texture maps, 351
- texture mesh, 315
- texture sphere, 351
- texture_colors, 391
- texture_geology, 348
- texture_walls, 347
- texture_wave, 258
- texturing, 351
- the Application window, 34, 35
- the slope, 259
- thin_fence, 277
- Three Dimensional Chemistry File Example, 164, 166
- Three Dimensional GroundWater Chemistry File Example, 171, 174
- three-dimensional
 - fill, 147
 - move, 129
- Three-dimensional, 125, 129, 133, 134, 147, 148, 150, 154, 226
- Three-Dimensional Cubic Glyphs, 130
- Three-Dimensional Finite Difference Grid, 136

- Three-Dimensional Glyphs, 131
- Three-Dimensional Glyphs Representing Vector Data, 132
- Three-Dimensional Rectilinear Grid, 134
- Three-dimensional representations, 151
 - site, 151
- Three-dimensions, 132
- Thresholded, 140
- Tick Marks, 268
- TIFF, 349
- Time, 351, 361
- Time Domain AIDV, 180
- Time Domain Analyte Data, 179
- time_data, 351, 361
- time_field, 384
- time_geology, 361
- time_loop, 361
- time_value, 355
- tin, 363
- TIN, 236, 287, 339
- TIN surface, 339
- TIN surface problems, 339
- TINs, 141
 - creating, 141
- titles, 270
- tools, 400
- TOP, 206, 207
- topographic surface, 288
- TOTHC, 164, 166, 171, 174, 177, 182
- transform, 345
- transform_field, 345
- transform_group, 345
- translate, 288, 345
- translate cell data to node data, 366
- translate coordinates, 345
- translation, 366
- tri_tool, 339
- triangular, 339, 363
- triangular irregular network, 363
- Triangular Irregular Network, 236
- Triangular Networks, 141

- triangulate_polygons, 339
- Triangulated Irregular Networks, 141
- Trichloroethylene, 164, 166, 177, 182
- tubes, 340, 341
- tunnel_cut, 289
- TVF File Format, 355
- Two-dimensional, 129, 133, 137, 148
 - creating, 132
 - shows, 138
- Two-Dimensional Glyphs, 129
- Two-Dimensional Rectilinear Grid, 133
- Two-Dimensional Rotated Finite Difference Grid, 136
- Types, 140
 - 3D, 140

U

- UCD, 164, 166, 177, 182, 296, 317, 326
 - writing, 177, 182
- unconformity, 288
- unconformity cutting, 287
- uniform, 363
- uniform field, 256
- union, 280, 281
- Upper, 126
- Upper Sand, 126
- use, 210
 - basemap, 210
- User Defined Bounds, 268
- USGS, 386
- USGS's MODFLOW, 135
- Usnd, 199, 202

V

- values, 157, 185, 188
 - 0.0, 186, 188
 - 1.0e9, 156
- Variogram, 149
- vector, 259, 331, 332
- vector data, 331
- vector magnitude, 332

- vectors, 327
- vertical discretization of nodes, 237, 390
- Vertical Exaggeration, 251
- VHS, 156
- video, 384
- video safe area, 384
- viewer, 368
- viewer window, 156
- viewer Window, 369
- Vinyl Chloride, 172, 174, 177, 183
- Vinyl_Chloride, 177, 182
- Visual MODFLOW, 335, 336, 401
- Visual Programming, 35, 36
- VisualModflow, 400
- vizbook_Index, 125
- VOC, 177, 182
- Volume, 256
- volume calculations, 261, 264
- Volume rendering, 256
- Volume Units, 262, 264
- volume_and_mass, 262, 264
- volume_integrate, 264
- volume_render, 256
- volumes, 280, 281, 289
- volumetrics, 264
- VRML, 370
- VRML-formatted file, 370

W

- W, 199, 202, 211, 215
- Wall Slope, 289
- Water Tables, 199, 203
- WEL-12, 202, 205
- well_decommission, 266
- Windows Audio-Visual Interleaved Data, 156
- Windows Bitmap, 156, 369
- Windows NT, 156
- wipe, 350
- Within C Tech's EVS, 152
- Workbook4, 387, 390

workbooks, 23
World Wide Web, 370
write, 329
Write netCDF, 325
Write netCDF File, 325
write_cad, 327
write_coordinates, 327
Write_netCDF, 296, 317, 326
write_ucd, 364
write_vector_gis, 328
write_vrml, 370
writing, 177, 182
 UCD, 164, 166, 177, 182

X

X-Coord, 164, 166, 171, 174, 177, 182

Y

y, 156, 164, 166, 171, 174, 177, 182, 185, 187, 199, 202, 206, 207, 211, 215
Y, 133, 141, 152
y coords, 211, 215
Y-Coord, 164, 166, 171, 174, 177, 182

Z

z, 156, 164, 166, 177, 182, 211, 215
Z, 125, 133
zoom, 38