AthosGeo View Users Guide Documentation

Release 2.2.0

Cornelis Bockemühl

Mar 31, 2023

PART I

This guide can be split into two volumes. **Introduction and Concepts** covers basics of geodata support with **AthosGEO View**.

Data Visualization and Analysis describes geodata visualization and analysis options available with **AthosGEO View**.

CHAPTER

ONE

ATHOSGEO VIEW INTRODUCTION AND CONCEPTS

1.1 Introduction and Availability

1.1.1 Introduction

AthosGEO View is an open-source visualizer for geodata, with some data editing and manipulation features. The focus is on block models and sampling sets, which can be complemented by background topography or any geometric shapes like buildings, geological units, faults, limits, boundaries, points with markers, and much more.

Did you know?

AthosGEO View is built on top of **ParaView** which is an open-source scientific data analysis and visualization tool. All of it's functionality is available also in **AthosGEO View**, and also the user interface and logic are the same.

Finally, **AthosGEO View** is also the base for **AthosGEO Blend**, which extends the abilities of the first with functions that are specifically interesting for the production of cement raw material. This is a commercially licensed tool.

In this Guide

This user's manual is designed as a guide for using the **AthosGEO View** application, dealing with the specific data set types, their visualization and some manipulation options.

Did You Know?

In this guide, we will periodically use these **Did you know?** boxes to provide additional information related to the topic at hand.

Common Errors

Common Errors blocks are used to highlight some of the common problems or complications you may run into when dealing with the topic of discussion.

For an introduction to basics of user interface, working pipeline, loading and saving data etc., please refer to the documentations for **ParaView** which are referenced in *First Steps*. This includes also advanced options such as the Python integration that are available in **AthosGEO View** in exactly the same way as in **ParaView**.

This current guide can be split into two volumes. *Introduction and Availability* to *Surfaces and Lines* will introduce the basic concepts of the different **AthosGEO View** specific data set types like *block models* and *sampling sets*, together with the various *attribute naming conventions*. More generic VTK data types like surfaces, lines, points and tabular data are discussed in the context of geodata processing. In *Manipulating Block Model Attributes* some advanced block model related subjects are handled, basically the interactive manipulation of attribute data.

The remaining chapters, from *Display a Block Model* to *Animations*, are about the different ways to visualize and analyze geodata sets like block models and sampling sets with **AthosGEO View**. Please remember that this can only cover some examples because with the means of the underlying **ParaView**, plenty of additional possibilities are available.

1.1.2 AthosGEO View is a Custom Software Derived from ParaView

ParaView is an open source, 3D data visualization and analysis software with a special focus on processing huge data sets, including the ability to process the data on several different server and client computers, running a variety of operating systems.

For more information, examples, software download etc., please refer to the **ParaView** website: https://paraview.org

Thanks to the fact that **ParaView** is open source, it is possible to download the sources, modify and/or extend the code and in this way derive an own custom software from it. **AthosGEO View** is exactly such a customized and extended variant of **ParaView**.

As a consequence, many features in **AthosGEO View** are identical to how they exist also in **ParaView**, like:

- The user interface (GUI)
- The concept of the processing pipeline with *Sources*, *Filters*, *Readers*, *Writers*, *Views* and *Representations*
- Many of the specific *Filters* etc. in **ParaView**, with exceptions that do not seem so relevant for the purposes of **AthosGEO View**
- The **Python** interface
- The ability to export views as *images* or generate *animations*

The big advantage of this setup is the fact that all these powerful features did not have to be rewritten for **AthosGEO View**. For certain desirable functionality this can of course also be a limitation. An example would be functions for the interactive manipulation of data as described in *Manipulating Block Model Attributes*.

Certain things that are available in **ParaView** are not or not fully made available in **AthosGEO View** so far, mostly because it was considered of less importance for geodata processing, like:

- So far, only a binary version with installation packages for **Windows** is available, while **ParaView** supports also **Linux** and **Mac** computers with binary packages.
- During the development of extensions, no effort was done to allow for distributed computing on several computers, because block model or sampling set data are normally not as large as some other data sets that are processed with **ParaView** in other fields.
- Some *filters* are not included because they are considered less important for geodata.

1.1.3 Availability of AthosGEO View

Binaries

Binaries of **AthosGEO View** can be downloaded from *cobo GmbH* (https://cobo.bockemuehl.ch). The only requirement is a free registration. Binaries are only available for Windows so far.

Sources

Sources of **AthosGEO View** are available through GitLab here:

https://gitlab.com/cobo-gmbh/athosgeo-view

Detailed instructions for building the software are still missing. So far it has been successfully done for both Windows and Linux.

Did you know?

AthosGEO View does not come with an automatic update mechanism for new versions and releases: It is up to the user to check whether there is something new available.

And more: There is not even an update installation available: Installing a new version or release will be like installing a new software on a computer.

This means that in order to do a full update, two steps are required:

1) Download and **install** the new version or release.
2) **Uninstall** the old version or release.

Is this a bug or a feature? It is actually a behaviour that is inherited from **ParaView**, and it is considered as a feature for users who want to have the full control of their tools: "Do I need that new release?" - "Just install and try!" - "And if I find that I do not want it?" - "No problem, you still have the old software, and you only remove it if you know what you really want!"

1.2 First Steps

First steps into **AthosGEO View** are basically first steps into **ParaView**, because it is most important to understand the underlying logic of *sources*, *filters*, *viewers* and more. Understanding this logic will give you the ability to find solutions for problems that are not covered in any user's manual or documentation on your own, while the same thing may be extemely difficult if you are looking for solutions that resemble those with other softwares that you may know already.

Anyway, humans tend to be impatient, and being sent to learning **ParaView** first instead of going directly into the applications that are of immediate interest is an advice that not many people will be ready to follow. For this reason, the following section will show how to *visualize a first built-in block model example*, just to give you an impression.

But then you will find a section that will show you the entry point to a step by step introduction - starting with learning the basics of **ParaView**.

If even this cannot convince you to take one step after the other, a third section will give you at least some few hints about the most basic concepts of **ParaView** and **AthosGEO View**.

1.2.1 A First Impression

Once the **AthosGEO View** software is installed and started, a demo model can be opened very easily through the *Help* menu: Selecting *Example Cases* will open a dialog as shown in Fig. 1.1, with the choice of two possible example visualizations.

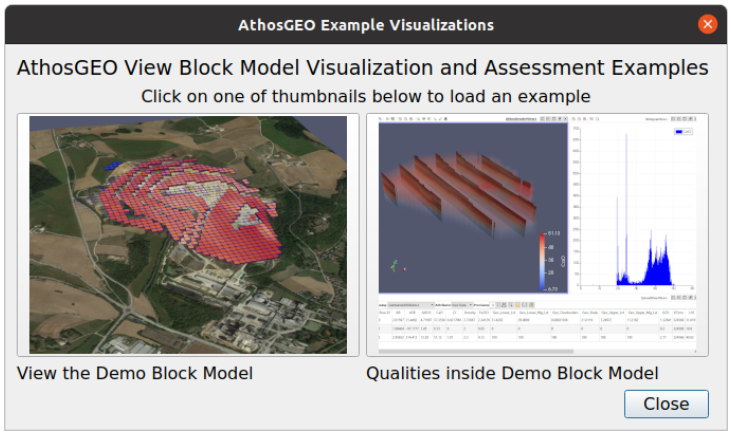


Fig. 1.1: A choice of two example visualizations are built into an installed **AthosGEO View** software

First Demo Example

The first example shows a demo block model, together with a background topography and aerial photo, as can be seen in Fig. 1.2.

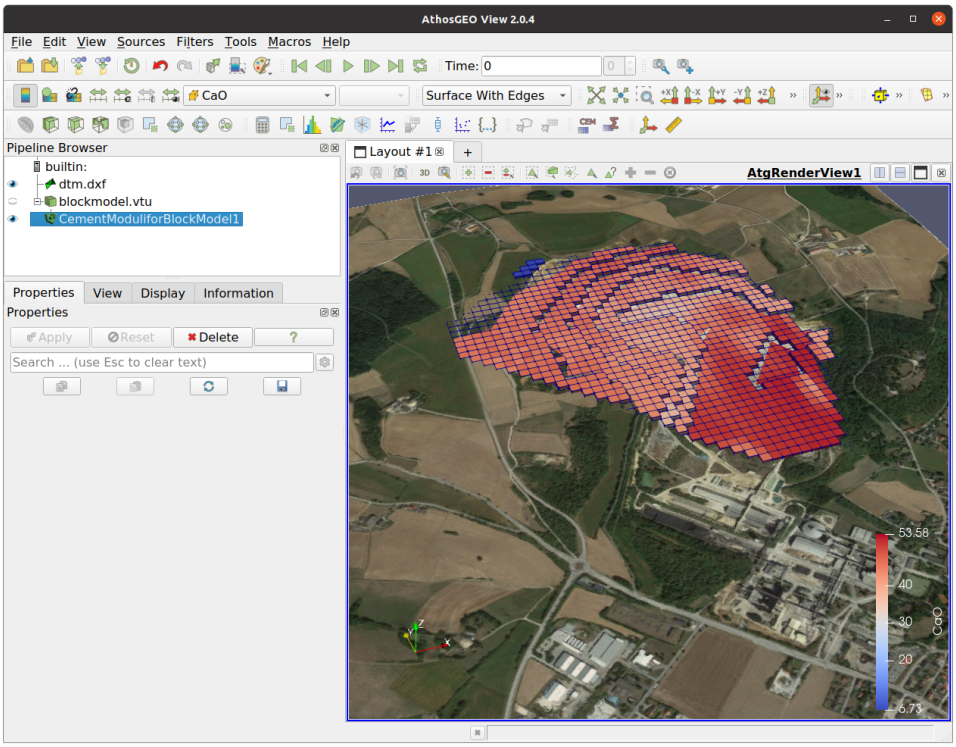


Fig. 1.2: Display of **Demo model** display with *topography* and *aerial photo* as a background.

The Render View

At the right side of the screen, in the **view area**, an *Athos Render View* shows a 3D display of the block model, together with a topography, that can be **zoomed** in and out, **rotated** and **panned** (moved) with the mouse interactively.

The blocks of the model are **colored** according to their content of **CaO**.

The Pipeline Browser

The **Pipeline Browser** is found at the upper left side of the view area. It shows the different *data objects* together with their relations. At the left side of the different items, an **eye symbol** is shown if the item can be displayed in the currently active view. Clicking on that eye will toggle it between *open* and *closed*, which toggles between *showing* and *hiding* the respective item.

Clicking on an item directly will make it **active**. Once a pipeline item is active, some view parameters can be adapted in the different combo boxes in the **toolbar** (see Fig. 1.3):

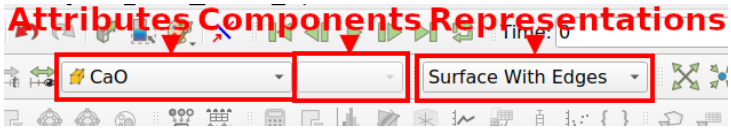


Fig. 1.3: The **Tool Bar Combo Boxes** are always referring to the **active pipeline item** in the current **active view**

- In the **Attributes** combo box, an attribute can be selected for coloring of the blocks in the model.
- The **Components** combo box is for attributes with several components, like a coordinate that has 3 components for the X, Y and Z direction.
- The **Representations** combo box allows to select a representation for a pipeline item, like *Surface*, *Surface with Edges*, *Feature Edges* and others.

The Properties Panel

The **Properties Panel** is at the lower left of the screen in the default configuration (which can be easily changed by the user). It comes with the following four tabs:

Properties

This tab allows to manage available **properties** of the currently **active pipeline item**. After doing some change, the user needs to press *Apply* to apply the changes.

View

The *View* tab is about properties of the **current view**, thus referring not only to one single pipeline item, but to the entire view. An example would be the display of **coordinate axes**. Changes will be applied immediately, without the need to explicitly press *Apply*.

Display

The *Dislay* tab refers to the **representation** of a currently **active pipeline item** within the **current view**. In the example it would apply e.g. to the block model, but not to the topo surface. Also the Display properties are immediately applied without further user action.

Information

This tab is of interest if the user wants to see what are the actual **data items** corresponding to the currently **active pipeline item**. This is for information purposes only, nothing to be changed by the user.

Did you know?

Like many other dialog windows within **AthosGEO View**, all these panels have a **normal** and an **advanced** mode that may show some less commonly used properties, and pressing the  button in the upper right part of the panels toggles between the two modes.

To the left of this button, there is also a **Search** entry field that assists with the management of large and sometimes a bit confusing panels: Just start typing the first letters of a property that you are looking for, and all properties that do not start with these letters will automatically be hidden.

Second Demo Example

The second example that is accessible through the *Example Cases* dialog in Fig. 1.1 is about different **view types** as shown in Fig. 1.4.

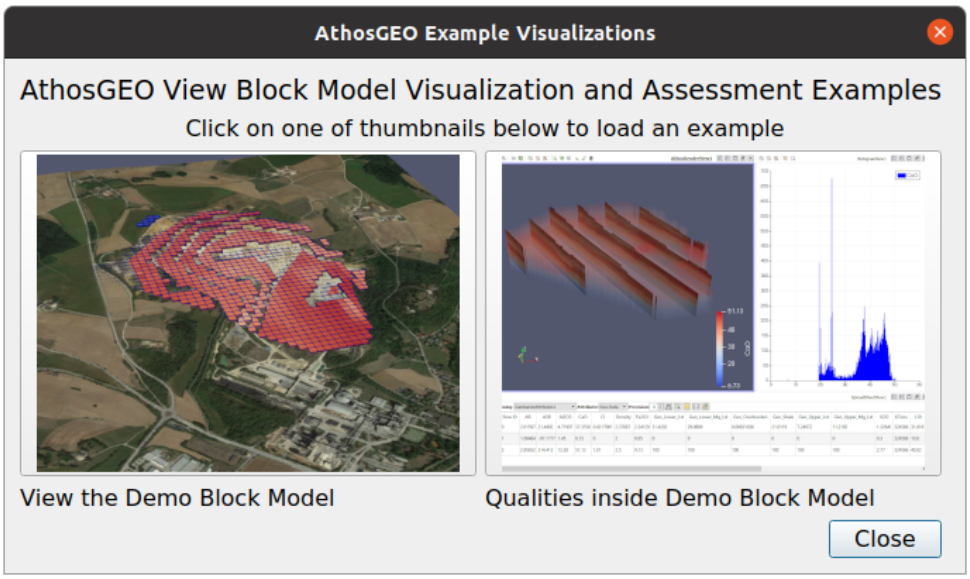


Fig. 1.4: This example shows 4 different ways to display the data of a block model

This second example shows several different ways to visualize the data of a block model:

- In the **Athos Render View**, the complete block model (*blockmodel.vtu* in the *pipeline browser* is displayed with a **Volume View** representation. This is a display where the blocks are not only colored, but have also a transparency assigned to it, with the highest values not only in red, but also with the highest opacity. This is a way how for example *lenses* of material with a higher content of some chemical compound can be visualized even “inside” the block model.
- In the same **Athos Render View**, the block model is visualized a second time, but after having cut into a series of slices by using the **Slice Filter** that is also visible in the *pipeline browser*.
- At the bottom of the *view area*, the same block model is displayed in a **Athos Summary View**. This is a tabular view where all the attributes are summarized according to their specific account type: tonnages are simply added, chemical compounds are tonnage weighted mean values of the entire model, etc.
- At the right side of the *view area*, a **Histogram View** shows the distribution of *CaO* contents in the block model. With the **Display** properties panel it is possible to select another attribute or chemical compound for display and adapt the number of bins.

1.2.2 Start with ParaView

After these first impressions from the example cases, it may be worth understanding the basic concepts of the **AthosGEO View** software a bit more, which happen to be the same as those of **ParaView**. For this reason it is recommended that you continue now with an introduction into **ParaView** for which you find an introduction here:

https://docs.paraview.org/en/latest/

Most of the things that are described in the **ParaView** documentation can be directly applied also in **AthosGEO View**. The easiest way to get also access to the **ParaView** example cases is an installation of that software which is described in the documentation.

Note

Some features from **ParaView** were disabled in **AthosGEO View** in order not to overload that software with features that are not applicable for geodata visualizations.

The **ParaView** documentation is not only of interest for the basic concepts, but also for more detailed information about advanced features like the **Python** support which exists also in **AthosGEO View** but not handled in this current documentation.

1.2.3 For Those who Cannot Wait

Pipeline Elements

With the **VTK**, data processing always works along a **pipeline of filters**, with *filters* being any kind of data processing module, as symbolized in Fig. 1.5. The power of this approach is it's enormous flexibility because filters can be added or removed at any time.

Fig. 1.5: The **VTK** data processing filter pipeline

ParaView and with it also **AthosGEO View** are visualizing this abstract concept in the **Pipeline Browser** that allows the user to add, remove or reconnect the *filters* within the *pipeline*.

So basically the elements of the pipeline are **filters** in general, with input and output **pipes**, but there are also special cases of filters:

Sources

These are filters without an input pipe because they are generating data from scratch. An example would be the *Sphere* filter that generates the geometry of a sphere with a specified center location and radius.

Readers

These are also filters without an input pipe because they are reading their input from a disk file.

Writers

As counterparts of the readers, these have no output pipe, bit instead they are writing the data into a disk file. Writers do not appear in the pipeline, but they are called with the *Save Data* command.

Views

Also the views are considered filters without an output pipe. Also they are not symbolized in the *Pipeline Browser*, but instead they are generating screen output either graphically or in tabular form.

Extractors

A new type of pipeline items are the *Extractors* (since **ParaView** version 5.9). They can be inserted into the pipeline in order to extract intermediate data on request and write them to files.

Summarizing, **Pipeline Items** as visualized in the *Pipeline Browser* can be considered as being **Filters** in the sense of the VTK, because by choosing a *filter* (or *source*) from the **ParaView** main menu will result in adding a corresponding icon.

But if you want, you can also consider them as representing the **data output** of a filter that can be analyzed by looking at the *Information* panel, written to a *file* or visualized in a *view*.

Did you know?

Online help for all the available filters, sources, readers and writers is available through the *Help* menu through the *Filter, Reader and Writer Reference* command. Find there a short description of the purpose, together with input, output and properties of all the filters.

Data and State Files

The *Save Data* command (or  from the *toolbar*) will save the output data of the currently active *pipeline item* to a data file.

The *Save State* (or ) will write a ***.pvsm** file that describes the current state of the **pipeline**, meaning: the *filters* with their *properties* and *connections*. However, it **does not write** the *data* that were loaded from disk files.

Did you know?

A **state file** contains also the full file names of the *data files* that are part of the current state of the pipeline (because these are actually *properties* of the *readers*), but sometimes it happens that these paths are not any more correct, like after moving data and state files to another computer. This is not a problem with **AthosGEO View** because on reading a state file, the program will ask whether to keep the current paths, look for all the files in a different path, or specify the correct path for each single data input file.

Points and Cells

When dealing with VTK type 3D geometries it is good to know that they are made up of **points** and **cells** that are in a way independent. Let's take a little triangulated surface as an example as shown in Fig. 1.6.

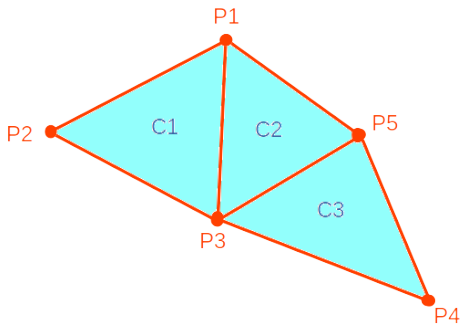


Fig. 1.6: A little triangulated surface, made up of 5 **points** and 3 **cells** of type **triangle**

Data of this little example will be stored in two tables, as shown in Table 1.1 and Table 1.2.

Table 1.1: **Points** table for triangulated surface in Fig. 1.6

PointID	X	Y	Z
1	x1	y1	z1
2	x2	y2	z2
3	x3	y3	z3
4	x4	y4	z4
5	x5	y5	z5

Table 1.2: **Cells** table for triangulated surface in Fig. 1.6

CellID	CellType	Points
1	triangle	1, 3, 2
2	triangle	5, 3, 1
3	triangle	4, 3, 5

In the case of a **block model**, cells are of type *hexahedron*, defined by 8 points. This knowledge about points and cells helps to better understand the following section.

Point, Cell, Field and Row Data

Geometries in VTK can have attached attributes of several different types:

Point Data

A table of data, with each data row assigned to a *point*.

Cell Data

A table of data, with each data row assigned to a *cell*.

Field Data

A table of data, with the rows not assigned to anything special.

In addition to these data types, there are still the

Row Data

These are basically tables of data which are not assigned to any geometry.

And Once More...

For more detailed and thorough information about all the concepts that were shortly introduced here please refer to the **ParaView** documentation as explained here in Section 1.2.2.

1.3 Block Model Basics

1.3.1 Definition and Purpose

In very general terms, a block model is a way to store spatially distributed data which can be numeric or any other data type. A typical application is in geology, mining and mines planning, and typical data would be qualities, like SiO2, MgO or S, but they can be everything that is related to the underground volume of interest, like names of geological units, permitting or whatever.

The blocks within a block model have typically the form of a cuboid or hexahedron and can be either all of the same size or different. Some functions of AthosGEO are only working properly if at least the base squares of the blocks are the same, with only the vertical dimension being variable.

1.3.2 Generating an Empty Block Model

An empty rectangular block model can be easily generated with **AthosGEO View** by choosing the **New Block Model** source: *Sources* → *Block Model* → *New Block Model*.

This command will open a **Properties** panel as shown in Fig. 1.7.

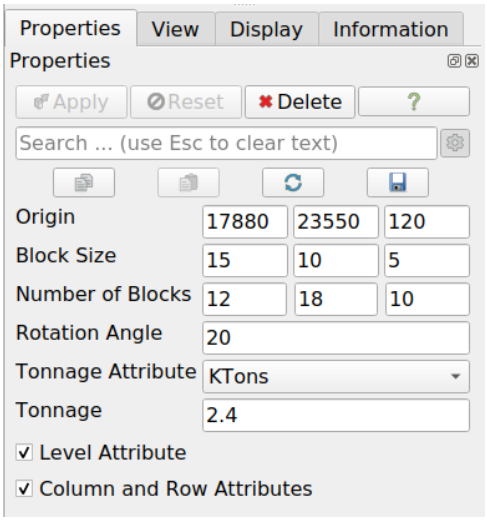


Fig. 1.7: Properties for defining an empty block model

On pressing *Apply*, the block model will be generated and shown in an *AthosGEO Render View* (3D View): see Fig. 1.8.

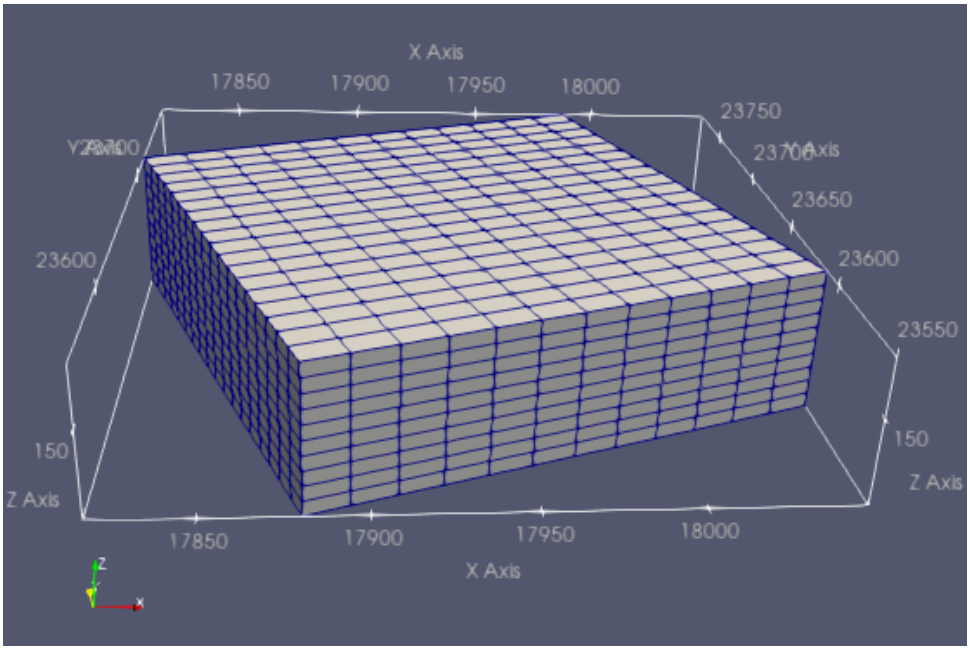


Fig. 1.8: An empty block model, meaning just blocks without attributes

It is now possible to manually generate attributes by using different filters, like:

- **Write Value** or **Write Multiple Values** for manually assigning single values to selected blocks
- **Read Attributes from a File and Assign to Unstructured Grid** if attributes are available in an existing file
- **Calculator** or **Calculator for Selection**, as soon as some other attributes do exist

The following filters will allow to clip the block model to a specific shape:

- **Clip Model with Surface** or **Clip Model with GeoTiff** for applying a top (topo surface) or bottom to the model.
- **Clip Model with Boundary** for applying a bounary to the model like a cookie cutter.

Note

The latter filters will keep blocks that are only partially inside the desired volume and add a **VolFactor** attribute to all blocks, indicating the percentage of the block that is considered to be inside of the model.

1.3.3 Reading a Simple Block Model from a CSV File

Normally a block model is only useful if it has attributes assigned to the single blocks. Such a block model with attributes can be generated in many different ways depending on the input data and available softwares, written into a CSV file and from there directly converted into a block model. A very simple example of such a CSV file could be a table like this one: Table 1.3

Table 1.3: Simple block model example data

X	Y	Z	dX	dY	dZ	Unit	CaO
1010	2010	205	20	20	10	LowerLst	52.0
1030	2010	205	20	20	10	LowerLst	52.3
1010	2030	204	20	20	8	LowerLst	51.9
1030	2030	204	20	20	8	LowerLst	52.1
1010	2010	213	20	20	6	LowerLst	52.3
1030	2010	213	20	20	6	HigherLst	48.7
1010	2030	211	20	20	6	HigherLst	48.6
1030	2030	211	20	20	6	HigherLst	48.7

In order to visualize this block model using **AthosGEO View**, you need to save it as a CSV file and open it with **AthosGEO View** with the block model reader, see Fig. 1.9

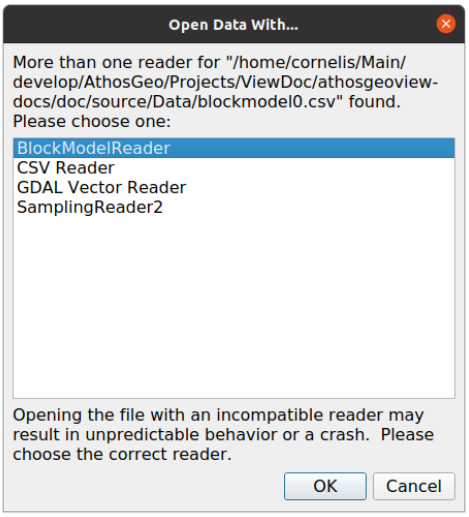


Fig. 1.9: On opening a CSV file, the user will see a choice of several possible readers

With this, the *Properties* panel will appear, see Fig. 1.10

The section of primary interest is about **Block Model Parameters**: *Block Size* (dX, dY, dZ) and *Angle*, which is an azimuth angle, counting in degrees from N via E to S and W. Both can be

- either specified within the **table**, such as above. In this case it is possible to specify block sizes or azimuth angles for every single block

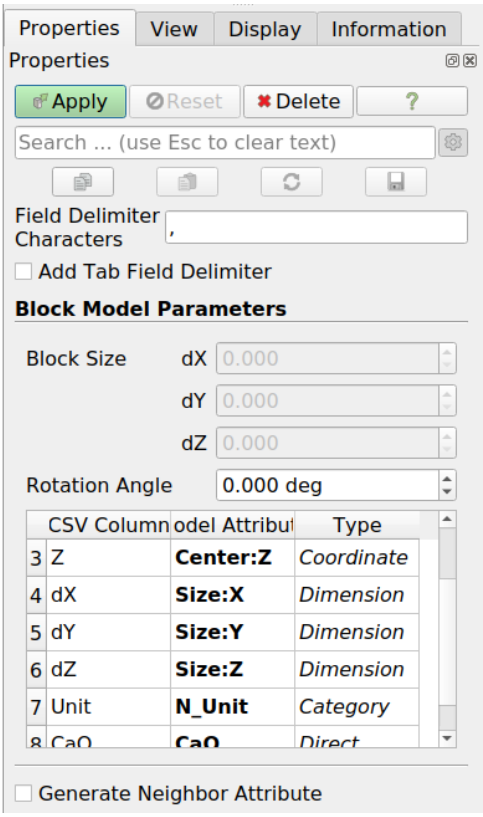


Fig. 1.10: Properties panel for the Block Model Reader

- or specified in the **panel**. In that case these are global values, to be applied to all the blocks equally.

At the bottom there is another property that can be activated: *Generate Neighbor Attribute*. This is only of interest for users of the cement specific modules of **AthosGEO Blend**. For users of **AthosGEO View** it is unimportant.

After pressing the *Apply* button, the block model can be displayed in the **AthosGEO Render View** with coloring according to the two defined attributes, as can be seen in Fig. 1.11 and Fig. 1.12

Hint

There are two ways to generate a block model from attributes in a CSV file (table), and you will use the one or the other depending on what kind of information you have in the table to specify the blocks:

- For each block, centroid coordinates, block dimensions and if required a rotation angle are present as attributes: *Reading a Simple Block Model from a CSV File*.
- For each block, a **BlockId** attribute is present: *Generating an Empty Block Model*, adding the actual attributes with the **Read Attributes from a File and Assign to Unstructured Grid** filter.

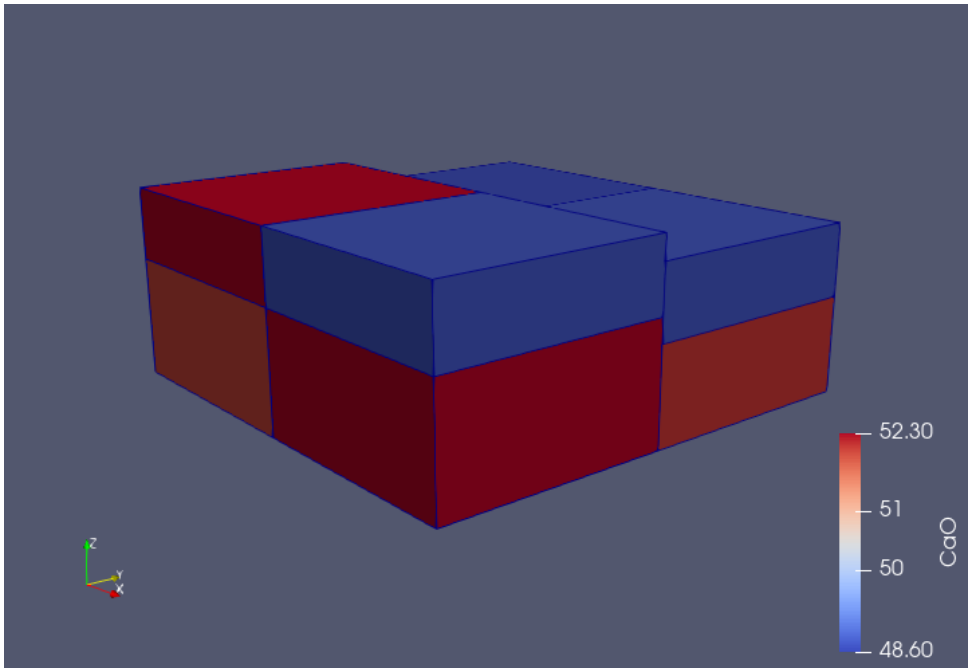


Fig. 1.11: Simple block model example displayed in an **AthosGEO View** Render View, with coloring according to the *CaO* attribute

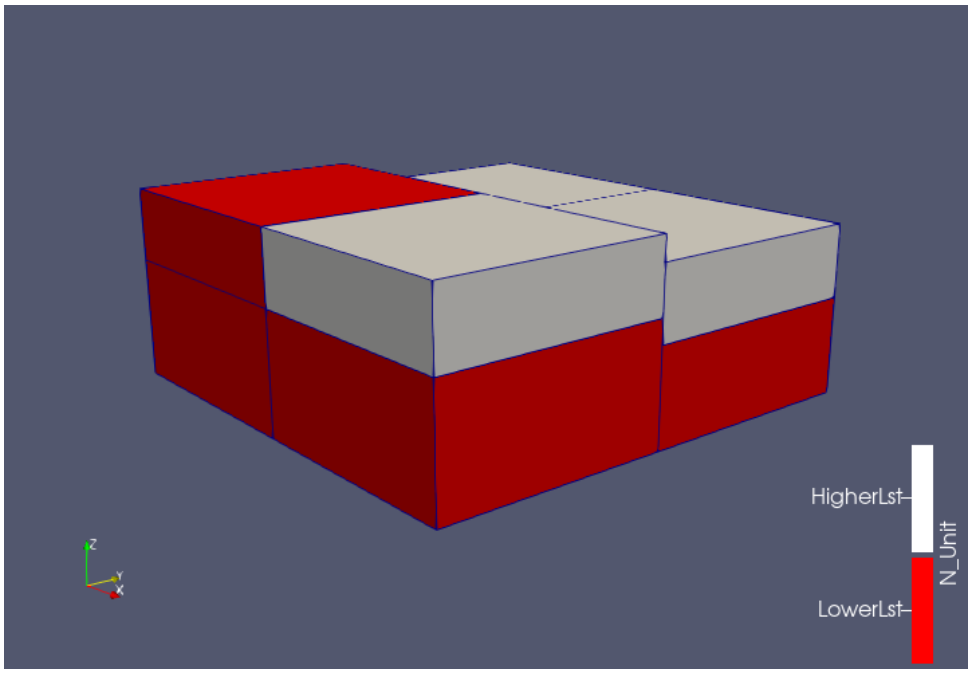


Fig. 1.12: Simple block model example displayed in an **AthosGEO View** Render View, with coloring according to the *Units* attribute

1.3.4 Block Model Attributes

For the naming of block model attributes in **AthosGEO View** conventions do apply, allowing the program to handle them accordingly, as coordinates, dimensions, tonnages, quality data, category names and many more. For the import from a CSV file, additional names are possible that are automatically converted into the standard names for convenience.

Please refer to *Attribute Conventions* for more details about this subject.

1.3.5 Saving a Block Model

Once a block model is imported as explained above, it can be saved in one of two possible ways: As a *CSV file* or as a VTK *Unstructured Grid* file (*.vtu). The following sections will explain the implications of these two options.

Many more formats are possible and can be relevant for the data exchange with other softwares.

Save as VTK Unstructured Grid (VTU) File

Saving as VTU file is the most straightforward way to save a block model in some kind of "native" **ParaView** format. A number of options can be chosen, as shown in Fig. 1.13

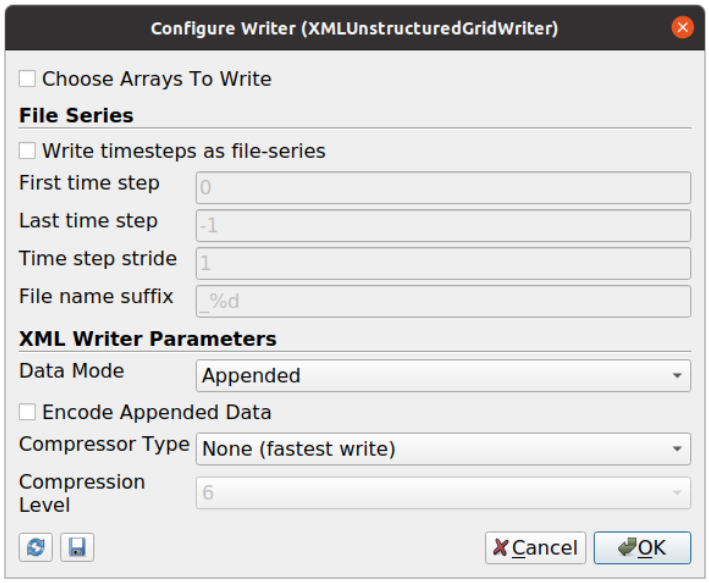


Fig. 1.13: Options for the Unstructured Grid Writer

The **XML Writer Parameters** are about compression. Go for Data Mode *Binary* and Compressor Type *LZMA* for smallest file size, or *Ascii* and *None* if you want a full plain text format, but if neither the one nor the other are of interest, this choice is not crucial.

In order to save an animation as a series of VTU files, the **Write timesteps as file-series** option needs to be checked.

Save as CSV File

A block model can be generated from a CSV file as described above in Section 1.3.3, and it can also be written back into a CSV file in such a way that reading it back into **AthosGEO View** is possible.

With this it is possible to manipulate the block model data with any *spreadsheet program*.

It is important to change the *Precision* in the **CSV Writer Parameters** dialog to something much larger than the default of 5, in order to get the centroid coordinates written with sufficient numeric precision: see Fig. 1.14

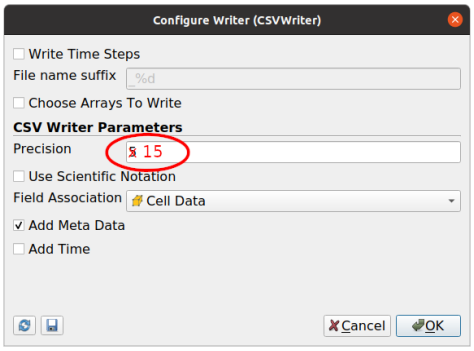


Fig. 1.14: Options for the CSV writer. In order to save a block model to a CSV file, *Precision* should be much more than 5 and *Field Association* must be **Cells**

The *Field Association* must be **Cells** because these are the actual blocks with the attributes.

Hint

Writing and reading a CSV file takes longer than the same operations for generic VTU files, so saving as CSV file makes only sense if manipulation is indeed the intention.

1.3.6 Viewing Block Models

Block models can be viewed with **AthosGEO View** in a 3D View (*AthosGeo Render View*) or as a numeric table (*AthogGeo Table View*), and analyzed with a great number of filters coming either from **ParaView** or from **AthosGEO View**.

Refer to *Display a Block Model* of the **Reference Manual** for more.

1.4 Manipulating Block Model Attributes

The following refers to manipulating attributes of **block models**, but it can be applied also to **sampling sets** (see *Display Sampling Set*) and to **triangulated surfaces** or **polylines** (see *Display a Topo*).

Preliminary Remark

The primary purposes of **ParaView** are **data visualization** and **data analysis**. This is also the focus of the underlying **VTK** (*Visualization Toolkit*), and **AthosGEO View** inherits this focus as well. The basic data and models (like the *block models*) have to be prepared with other tools. The **pipeline of filters** is a perfect concept for this kind of functionality.

However, the pipeline of filters is far less suitable for cases where data need to be interactively changed or edited within a software. Still it is in certain cases desirable or even required to do it in **AthosGEO View**. So in order to fit into the working principle of the software, each and every editing step has to be a **filter**, and if a second editing step follows the first, **another filter** must be added to the processing pipeline, and so on.

1.4.1 Calculator 

The **Calculator** filter takes a data set with attributes as input and calculates a *new attribute* by applying a formula that can be entered in the **Properties** panel (see Fig. 1.15), or it overwrites an *existing attribute*.

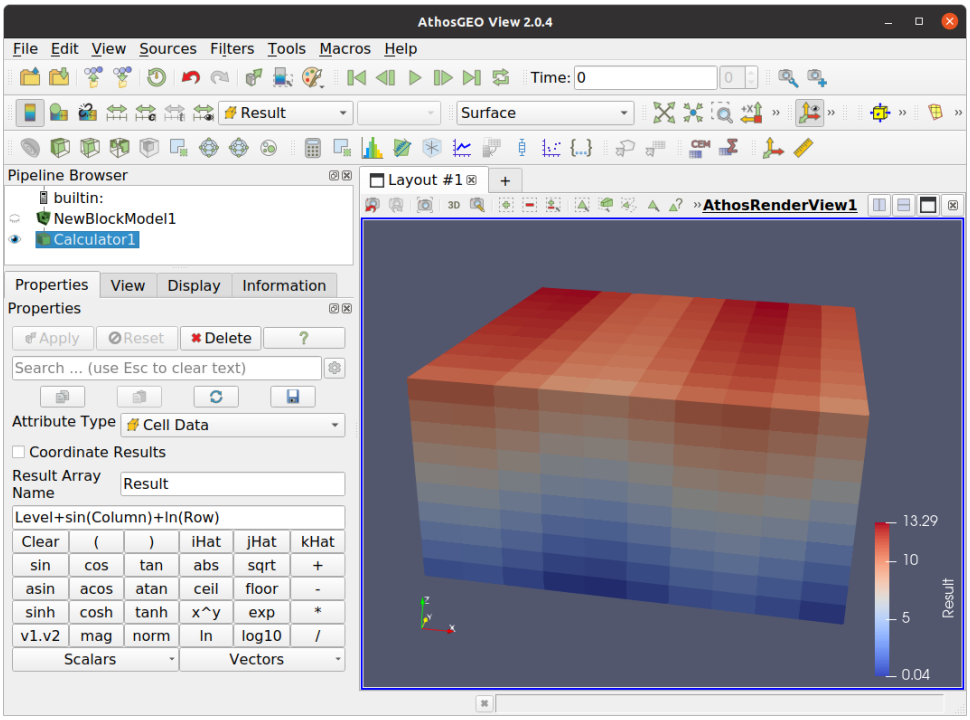


Fig. 1.15: **Properties** of the **Calculator** filter

In the example in Fig. 1.15, pressing *Apply* will generate a new attribute **Result** in the **Cell Data** of the data set.

1.4.2 Calculator for Selection 

Sometimes it is not desirable to apply a formula to all the blocks in a model. For this case, **AthosGEO View** comes with another calculator filter: **Calculator for Selection** - see example in Fig. 1.16.

In order to apply this **Calculator for Selection** filter, the following steps are required in this order. Regarding selections, see also *Selections for Attribute Manipulation*.

1) Choose the **Calculator for Selection** by pressing  in the *toolbar*.
2) **Select** the blocks to which the calculations should be applied. Please refer to Section 2.6 for more information about how to select blocks.
3) Enter the desired **result array name** and the **formula** and press the *Apply* button.

Note

As with the **Calculator** filter, also the **Calculator for Selection** can either generate a new attribute or overwrite an existing one. But what happens with the cells that are not selected in these two cases?

- If the result attribute is new, the **Replacement Value** will be assigned to all the non-selected model blocks.
- If the result array already exists, values for the non-selected model blocks will remain unchanged.

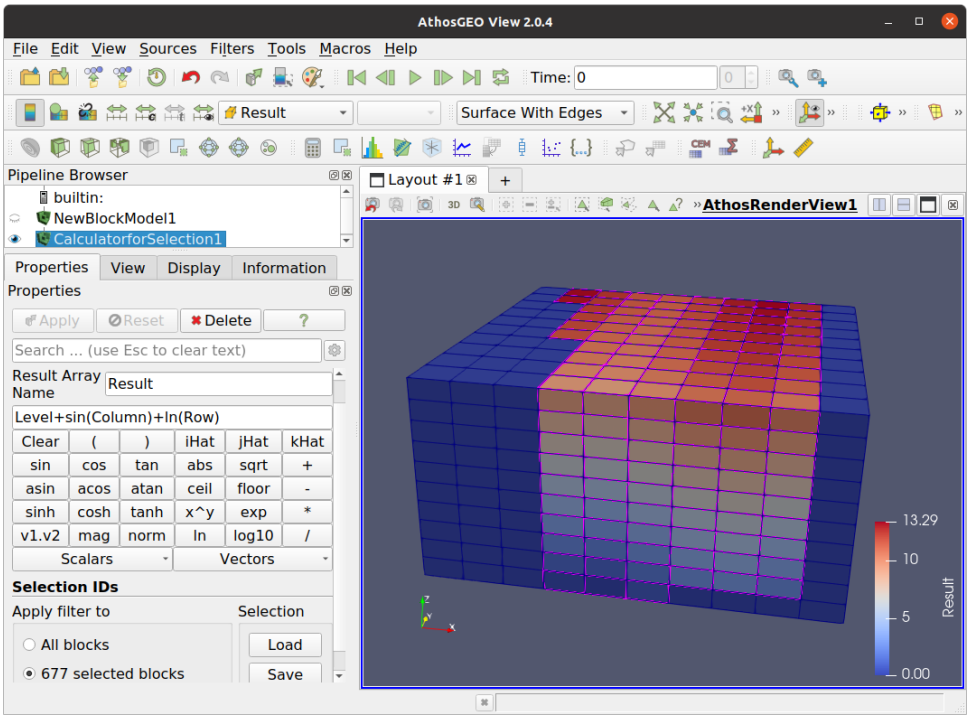


Fig. 1.16: With the **Calculator for Selection** filter, calculations are only applied to the selected blocks in the block model

Note that the **Replacement Value** is only visible in the **advanced mode** of the properties panel, and it exists for both calculator filters because that value will also be assigned where the formula results in an **error condition**, like a *division by zero*.

1.4.3 Write Value

The **Write Value** filter cannot do anything that the **Calculator for Selection** is not able to do as well (see Section 1.4.2). The only purpose is convenience for the case that the formula would be only one constant number. The functionality is very simple, as can be seen in Fig. 1.17: A new attribute is generated and a value is assigned to the selected blocks. Regarding selections, see also *Selections for Attribute Manipulation*.

If another value for the same new attribute has to be assigned to another set of blocks, a second instance of the **Write Value** filter must be appended to the pipeline, as shown in Fig. 1.18.

1.4.4 Write Multiple Values

Also the **Write Multiple Values** filter is a convenience filter that is able to do in one single step the two steps that are described for the **Write Values** (see Section 1.4.3). It also either generates a new attribute or manipulates an existing one, but it is able to write more than one single value, as shown in Fig. 1.19. Regarding selections, see also *Selections for Attribute Manipulation*.

Note

The **Write Multiple Values** filter needs to manage **multiple selections** in order to do it's job: Every row of the **Selections and Values** has one set of selected blocks, and making the one or other *active* will show or hide that selection in the view.

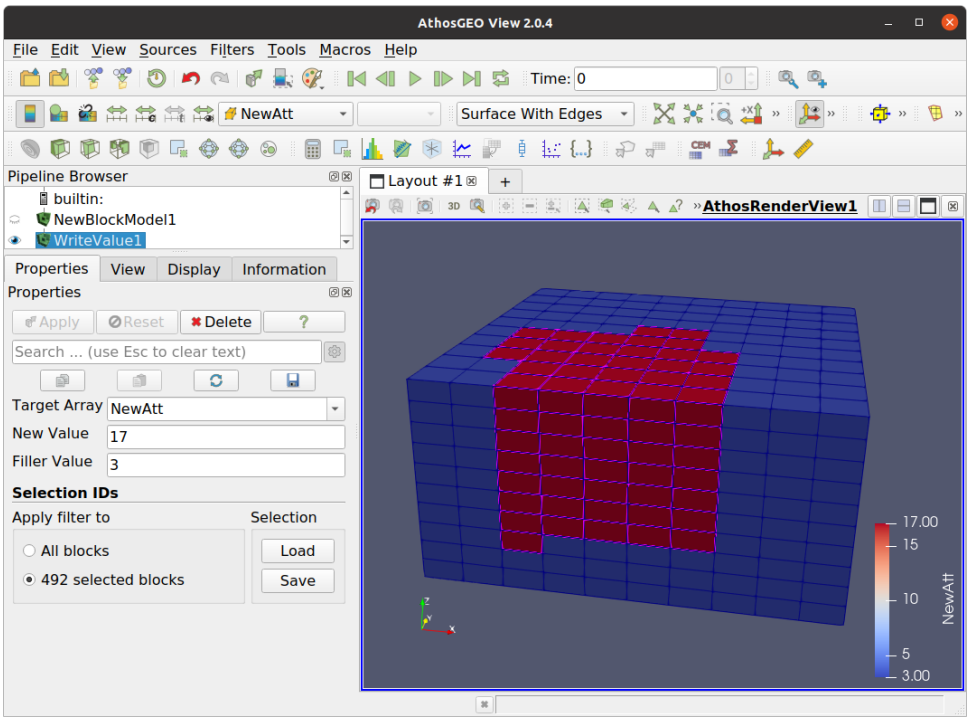


Fig. 1.17: Example of the **Write Value** filter: The *Target Array* is new, so the filter will generate it as a new attribute. The *New Value* is assigned to the selected blocks, and the *Filler Value* is assigned to all other blocks

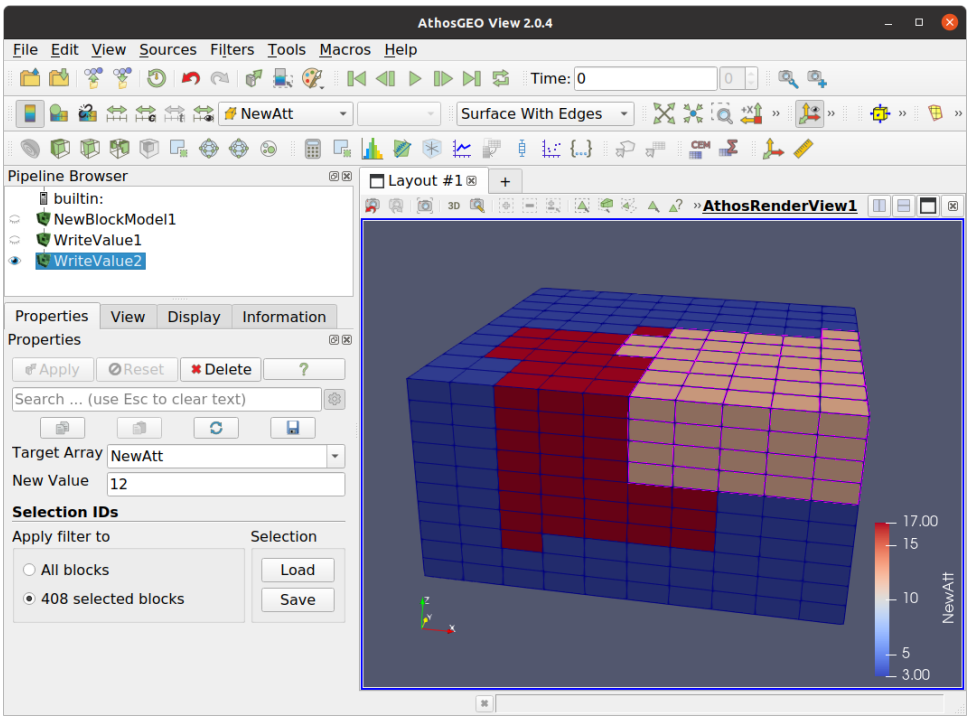


Fig. 1.18: For the second instance of the **Write Value** filter, the *NewAtt* attribute is not new any more, so the *properties* panel does not show a *Filler Value* property any more: All non-selected blocks will simply remain untouched. Selected blocks will receive the newly specified value.

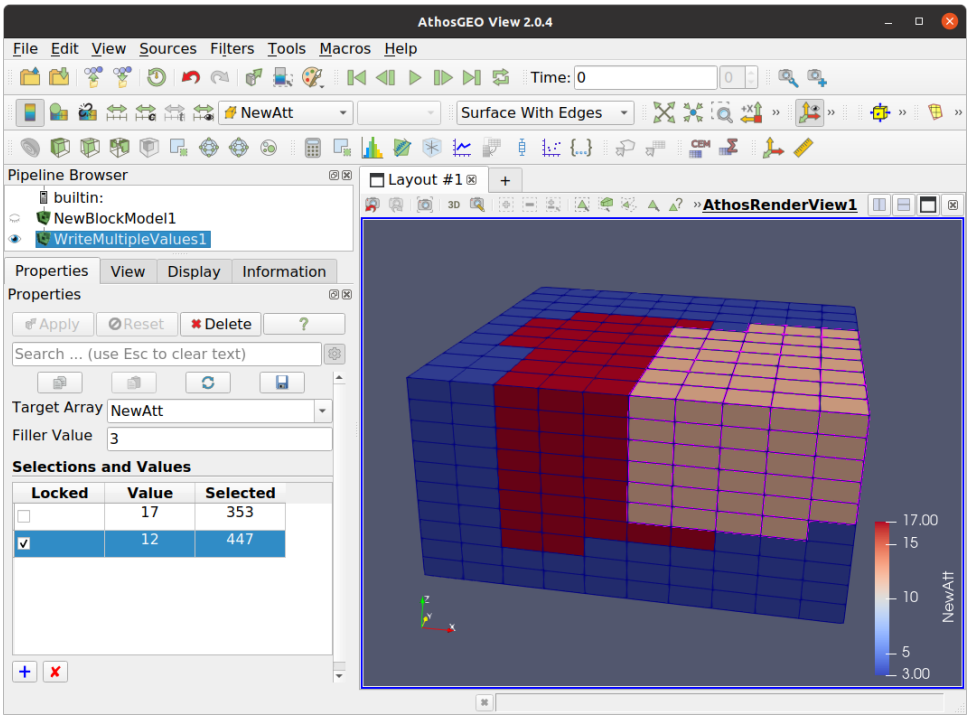


Fig. 1.19: With the **Write Multiple Values** filter, more than one value can be written into a new or an existing attribute.

1.4.5 Remove Data Arrays 

It's a bit confusing: If somebody looks in **ParaView** for a filter that *removes* the one or other not any more required *attribute*, nothing can be found! But this is only a question of wording: The name of the filter is **Pass Arrays**. The only difference to a non existing *Remove Arrays* filter is the fact that the user selects not the arrays to be *removed*, but the arrays to be *passed*. As a result, the effect is exactly the same!

1.4.6 How to Manipulate the Inner Blocks of a Model

In the **Surface** or **Surface with Edges** representation of a **Render View**, only the outer blocks of a block model are visible, but manual selection that is required for the different value manipulation filters described in this chapter (see Section 1.4.2, Section 1.4.3 and Section 1.4.4) is hardly possible for inner blocks.

This is where the **Category** or the **Category with Edges** representation will be helpful. First it is of course important to have a *Category* attribute in the model like **Level**. The procedure is explained with a series of figures from Fig. 1.20 to Fig. 1.26.

1.4.7 Models with Unequal Block Heights

Sometimes it makes sense to design a block model in such a way that blocks would follow geological units. This can be achieved with blocks always covering such a unit in vertical direction.

Note

A model with blocks following geological units is of special interest for users of **AthosGEO Blend** with a quarry where mining follows these units, in order to maximize the blending opportunities.

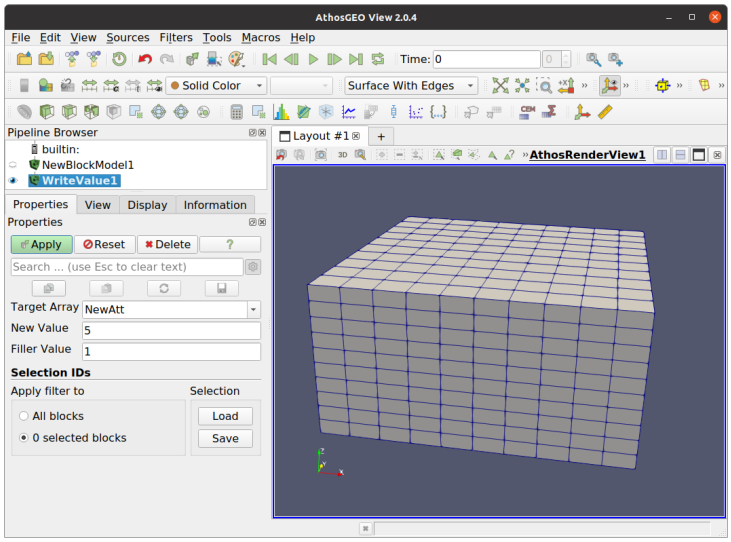


Fig. 1.20: As a first step in this example, the **Write Value** filter is activated.

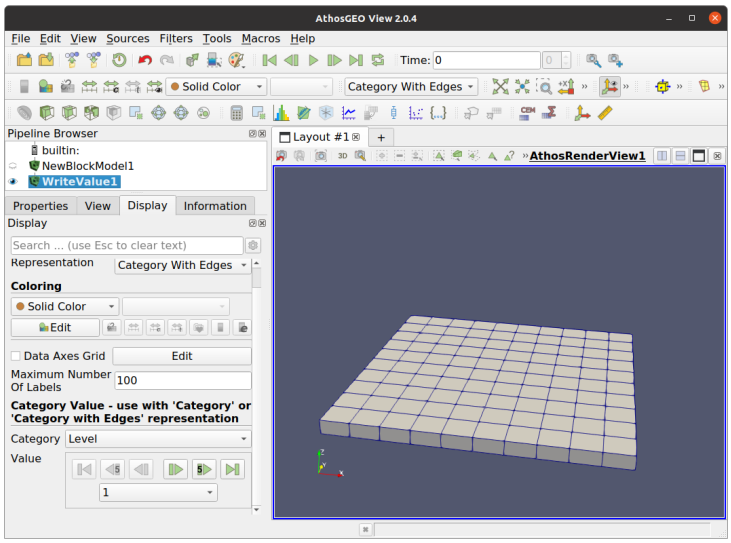


Fig. 1.21: Next, the **Category with Edges** representation is selected and in the **Display** tab of the *properties panel*, the *Level* attribute is selected for the display.

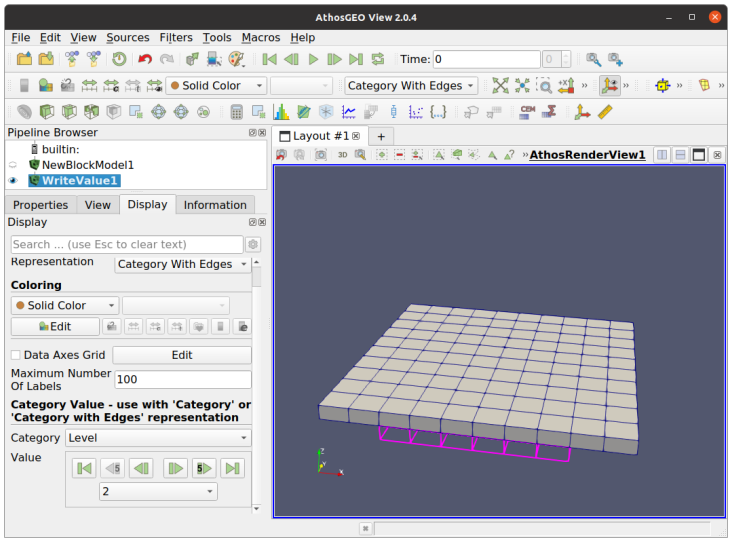


Fig. 1.22: A number of blocks were **selected at level 1**, then from the *Display* panel the *Level* was changed to **level 2**. Note that the selection at the lower level remains visible.

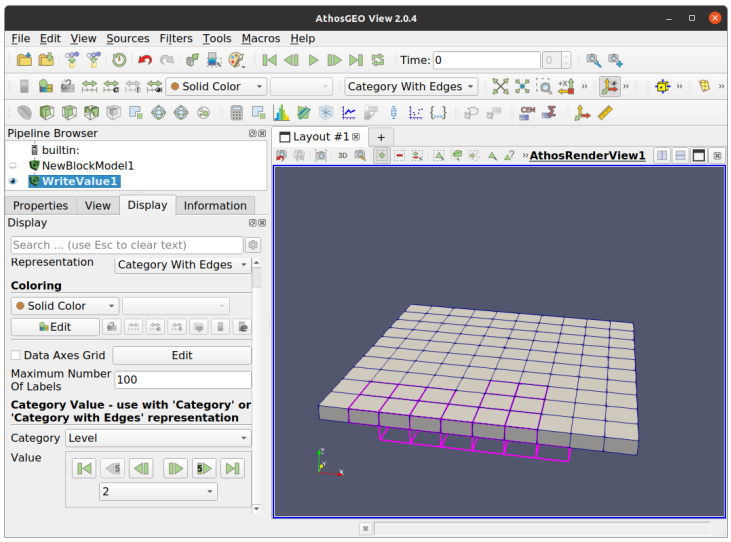


Fig. 1.23: With the + button turned on in the **selection tool bar** at the top of the view, another group of blocks is selected at level 2.

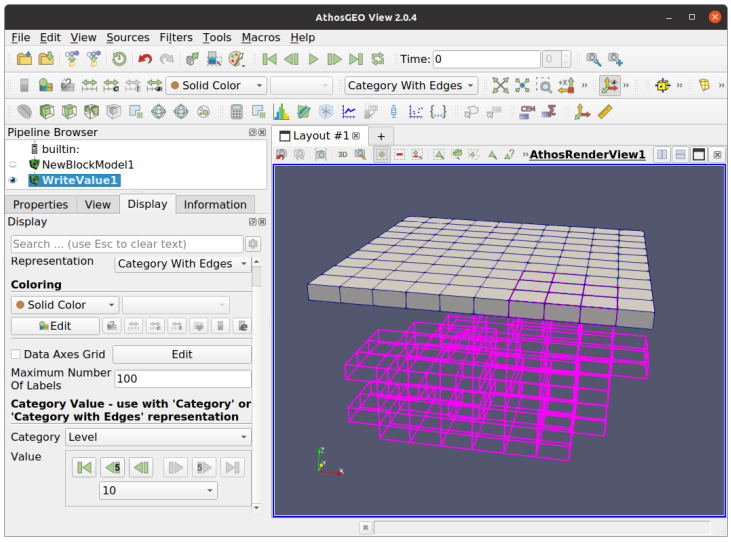


Fig. 1.24: A few levels later, an entire **selection of blocks in 3D space** is finished. **Hint**: Make sure to do this **from bottom to top** (in this specific case), because otherwise it is difficult to do the proper block selections "through" the already existing selection indicators.

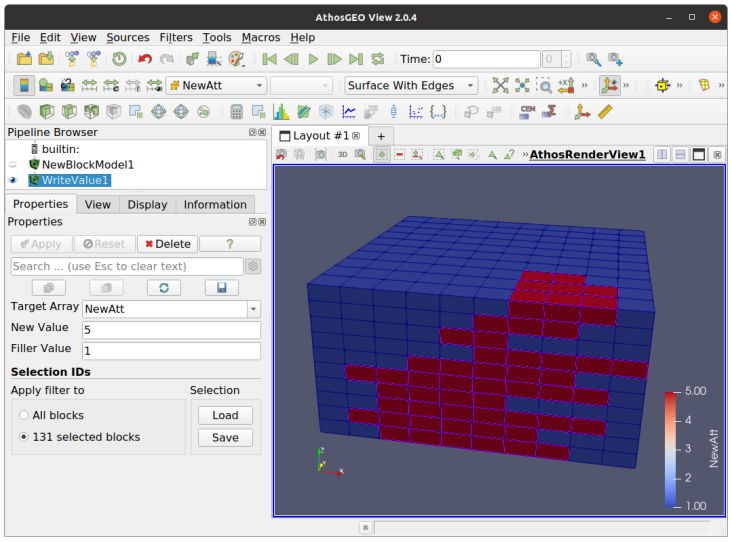


Fig. 1.25: After pressing *Apply* and switching back to a **Surface with Edges** representation shows the result of the action.

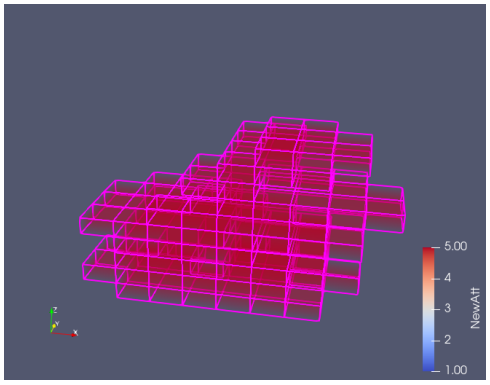


Fig. 1.26: The same thing as a **Volume** representation where the full spacial extent becomes visible.

With the following the generation of such a block model will be described. The starting point is illustrated with a schematic geologic cross section as shown in Fig. 1.27.

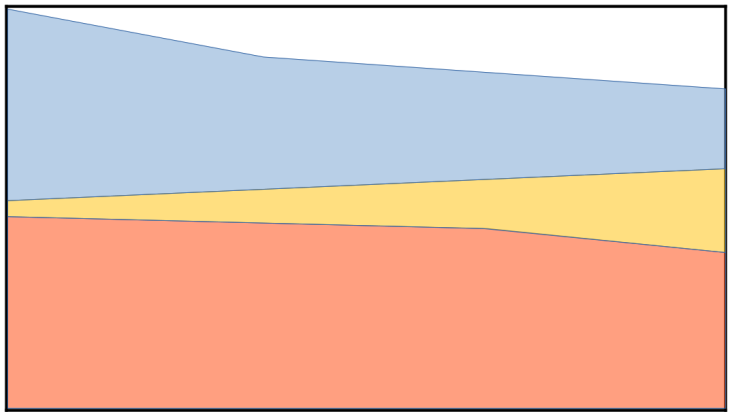


Fig. 1.27: Cross section with **geological section**. The intention is to follow these units during mining, and in order to reflect in the planning, we want the blocks of the model to follow the geology.

AthosGEO View and **AthosGEO Blend** are not able to do geostatistics for the generation of a block model. Other tools can do it, but they are normally not able to do what we are trying to achieve here. What can be done is the generation of a model with very "flat" blocks, like 1m or 2m only. And it is possible to assign some code for the geological units that are dominating within each one of the blocks, as shown in Fig. 1.28.

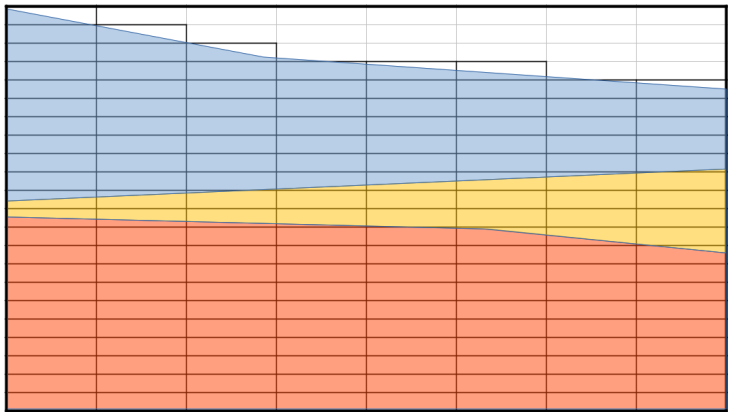


Fig. 1.28: Tools to calculate block models will normally not generate models that follow geological units, but it is possible to generate models with rather flat blocks, with a vertical dimension of only 1m or 2m, as a first step.

Before **AthosGEO View** can now merge blocks within the same geological unit vertically, theses blocks need to be marked accordingly, like with a **Code** attribute that assigns a number to every block, indicating which unit it mostly belongs to. This would normally still be part of the block model generatin step, and the result would be a block model as shown in Fig. 1.29.

Now is the time to apply the **Merge Blocks by Category** filter. In the *Properties* panel it will ask for selecting one available attribute of attribute type **category** (see Section 1.6 regarding attribute types). The result will be a block model with blocks that approximately follow the geological units, as shown in Fig. 1.30.

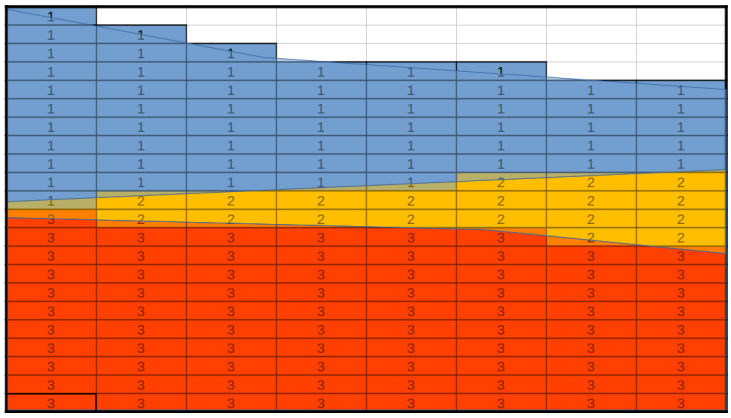


Fig. 1.29: In order to merge the flat blocks according to geological units, an attribute of **category** type is required, like **Code**, **Geology** etc.

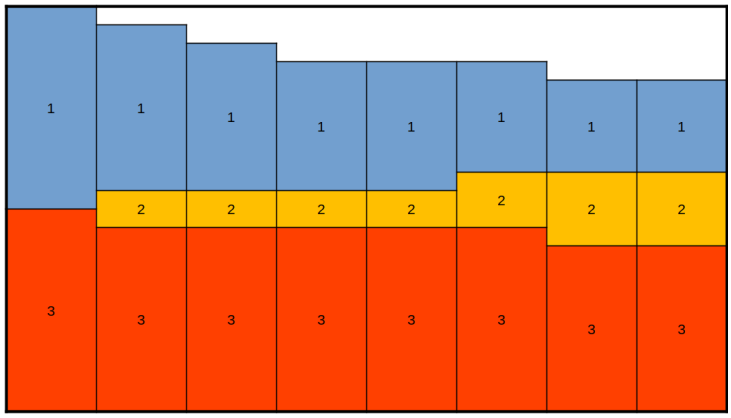


Fig. 1.30: After application of the **Merge Blocks by Category** filter, this will be the output. **Note** that block merging will only happen in vertical direction.

Attributes

So far this was about geometry. Next we still need to cover what happens with the **attributes**. Basically, attributes of the new merged blocks are calculated following the strategy of the **Summary Table**, as described here in Section 2.3.2.

However, there are two differences regarding attributes of type **category**:

- The **category type attribute** that is the base of the block merging is of course maintained in the output.
- **Additional attributes** will optionally be generated, indicating the percentage of material with a specific attribute value within every generated merged block.

The second feature will require some additional explanation, both about the purpose of this feature and the way how it is calculated. Both can best be done with an example.

The **properties** panel of the **MergeBlocksByCategory** filter allows to specify two properties (see Fig. 1.31):

- One category attribute that will guide the block merging - **Code** in the example
- Optionally one or several category attributes for the generation of new **category value attributes** - again **Code** in the example.

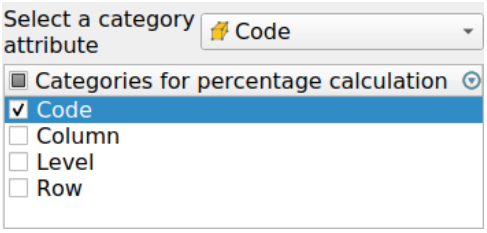


Fig. 1.31: The **Properties** panel of the **MergeBlocksByCategory** filter allows the selection of one category attribute that guides the merging. Additionally and optionally it is possible to select one or several category attributes for the generation of new category value attributes.

The result of the merging can be seen in Fig. 1.32:

- The block's vertical extent follows the **Code** values.
- Four new attributes were generated, for every existing value of the **Code** attribute:
  - C_Code_0
  - C_Code_1
  - C_Code_2
  - C_Code_3

  The values of this attribute can only be 0 or 100% in this case, because every new block is made up of always only one unique **Code** attribute value.

It is also possible to choose another category attribute for the generation of new percentage attributes: see Fig. 1.33 for an example where **Level** was selected.

Note

This last example with the level percentage is a bit artificial, with little practical use, but it illustrates the functionality.

The purpose of having such kind of percentage attributes is for mixing calculations. Assume that a subset of blocks within the model is selected, like the production of one year, such attributes will allow to directly see how many percent of the selected set is from Code value 1, 2, 3 etc., and the same in the above example also with Level value 1, 2, 3 etc. And if there was a category attribute **Quarry** in a multi-quarry model, also the proportion from Quarry A, B and C etc. can be directly seen.

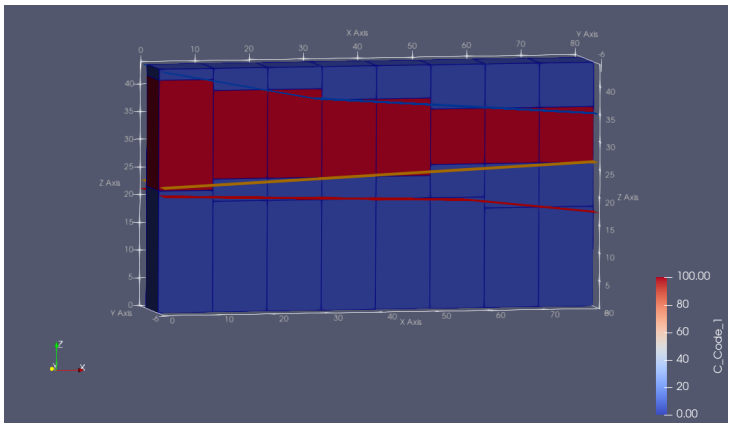


Fig. 1.32: Result of block merging by **Code** attribute values. Coloring shows all blocks in red that are 100% made up of material where this value is 1, while all others are blue, indicating 0%.

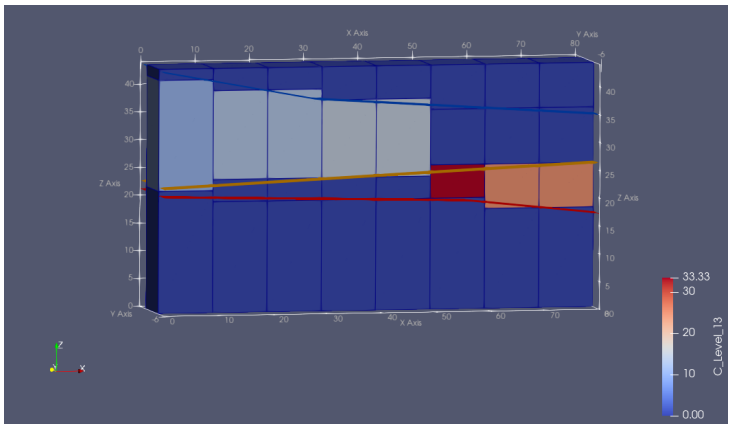


Fig. 1.33: This the result of merging the blocks by **Code** and calculating new attributes indicating the amount of material within a specific original **Level**. The coloring shows the percentage of each block that is from level 13.

Note

Use this function with care, because if a category attribute has many used values, it generates a huge number of attributes!

It is possible to achieve a slightly more precise result if the type of category value attributes is prepared already before the merging, e.g. during the block model generation. In Fig. 1.34 it is shown how already the "flat" blocks are having a C_Code_1 attribute that was calculated from the block fraction below and above the blue and yellow surfaces.

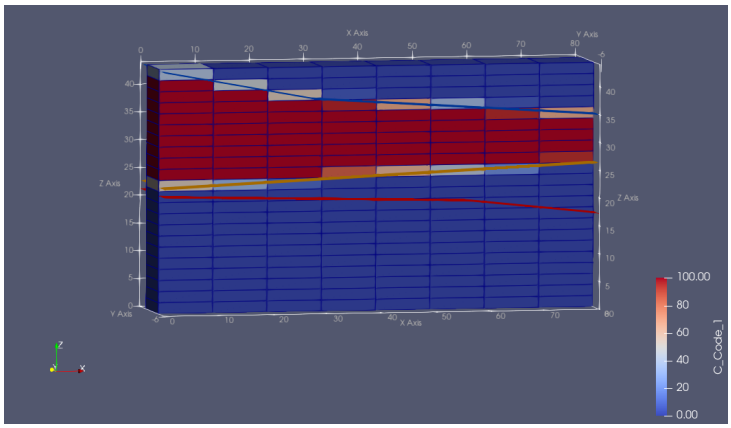


Fig. 1.34: In this model is already a **C_Code_1** attribute indicating the percentage of "Code = 1" material within every "flat" block (plus another attribute for each other Code value). Such attribuges can be generated in many ways, either within **AthosGEO View** or with other softwares.

In oder to use the existing **C_Code_n** attributes during the merging process, the **Code** attribute must **not** be checked in the **Categories for percentage calculation** list in the **Properties** panel. In this case, these attributes are dealt with according the rules for **Direct** type attributes: calculate the tonnage weighted mean. The result looks like shown in Fig. 1.35: Blocks are not any more either 0 or 100%, but can include "impurities" from the neighboring Code zone.

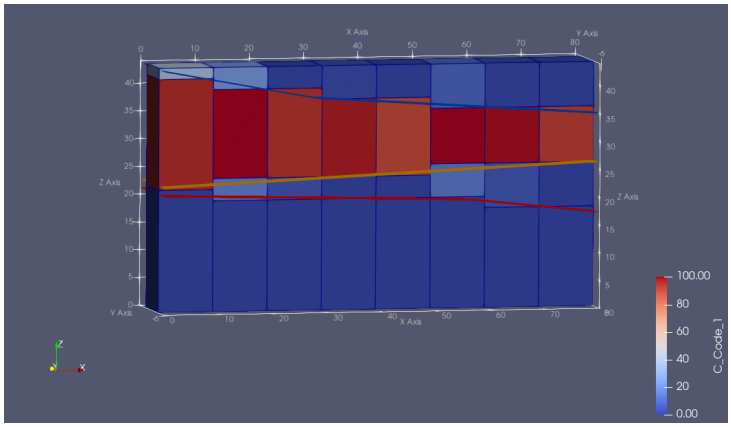


Fig. 1.35: In this variant, the **C_Code_n** attributes of the block merging result are indicating also little amounts of "contaminations" from neighboring Code value zones.

Note

For this last hint to work, it is of course not required to follow the naming scheme like **C_Code_n**, but any valid attribute name can be used, as long as it follows the pattern for **Direct** type attributes.

1.5 Sampling Set Basics

1.5.1 Definition and Purpose

In very general terms, a sampling set is

- A series of **georeferenced sample data**, i.e. samples with labels, laboratory results and coordinates (x, y, z).
- A data set from a **drilling campaign**, again with labels, laboratory results and coordinates, often stored in 3 tables:

  asset laboratory data with drillhole and sample labels and depth ranges within the drill hole

  collar list of drillholes with drillhole labels, coordinates of the hole top points and typically also characteristic of the drillhole, e.g. "linear"

  survey geometric drillhole descriptions with drillhole labels, depth, dip and azimuth angles

  In simple cases, the drillhole data can be delivered with less tables - details see below.

The intention of loading sampling sets into **AthosGEO View** is the display of point or linear symbols in order to visualize quality raw data of a survey.

Hint

Sampling sets are typically the input data for calculating **block models** (see *Block Model Basics*) with methods of geostatistics.

Please note that **geostatistics** is **NOT** part of the **AthosGEO View** software.

1.5.2 Reading Sample Data from a CSV File

In the case of plain georeferenced sample data, only one CSV file it required for input. Table 1.4 would be a minimal example.

Table 1.4: Sample data table

SampleID	X	Y	Z	SiO2	CaO
Sample 1	41226	25732	412	9.8	48.2
Sample 2	52332	20158	396	28.2	32.4
Sample 3	46298	29647	422	8.7	50.1

In order to visualize thess samples using **AthosGEO View**, you need to save this table as a CSV file and open it with **AthosGEO View** with the block model reader, see Fig. 1.36

With this, the *Properties* panel will appear as in Fig. 1.37

In the **Sampling Details** section it is possible to specify some parameters for the display of the samples.

If the input table does not come with a **Sample ID** column, a sample ID will be automatically generated from a prefix that can be specified in the *Sample ID Prefix* property and a numeric counter.

The table within this section is about display of the samples:

Type If the input table has a *Holetype* column, the different hole types will be listed, so they can be assigned different location markers and sizes. If such a column does not exist, it will be (all).

Decorator With a click into a cell within this column, a decorator or little 3D object can be chosen to mark the sample locations in space.

Size With this, the relative size of the decorator within the 3D view can be determined.

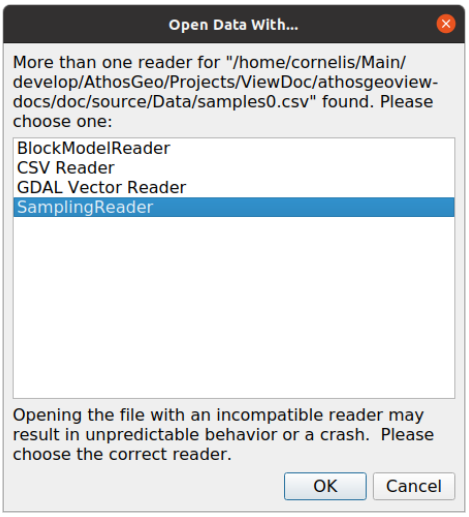


Fig. 1.36: The Samples Reader interprets the content of a CSV file as either a number of single samples or as the *assay* file of a drilling campaign

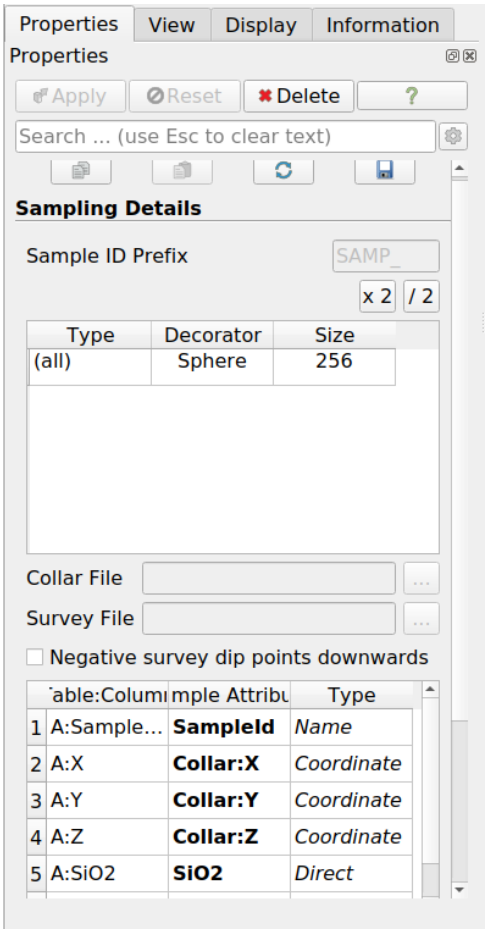


Fig. 1.37: Properties panel for the Samples Reader

Note

The *Type* feature in the **Sampling Details** table was introduced for drilling campaigns originally, which is also the reason for the naming of the column. See the list of possible point sample decorators in Fig. 1.38.

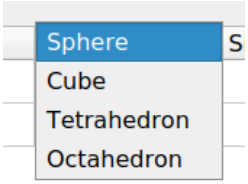


Fig. 1.38: Possible sample point decorators

Note

The two little buttons *x 2* and */ 2* can be used to increase or decrease all size values by a factor of 2.

The section of primary interest is about **Block Model Parameters**: *Block Size* (dX, dY, dZ) and *Angle*, which is an azimuth angle, counting in degrees from N via E to S and W. Both can be

- either specified within the **table**, such as above. In this case it is possible to specify block sizes or azimuth angles for every single block
- or specified in the **panel**. In that case these are global values, to be applied to all the blocks equally.

After pressing the *Apply* button, the samples can be displayed in the **AthosGEO Render View** with coloring according to the two defined attributes, see Fig. 1.39

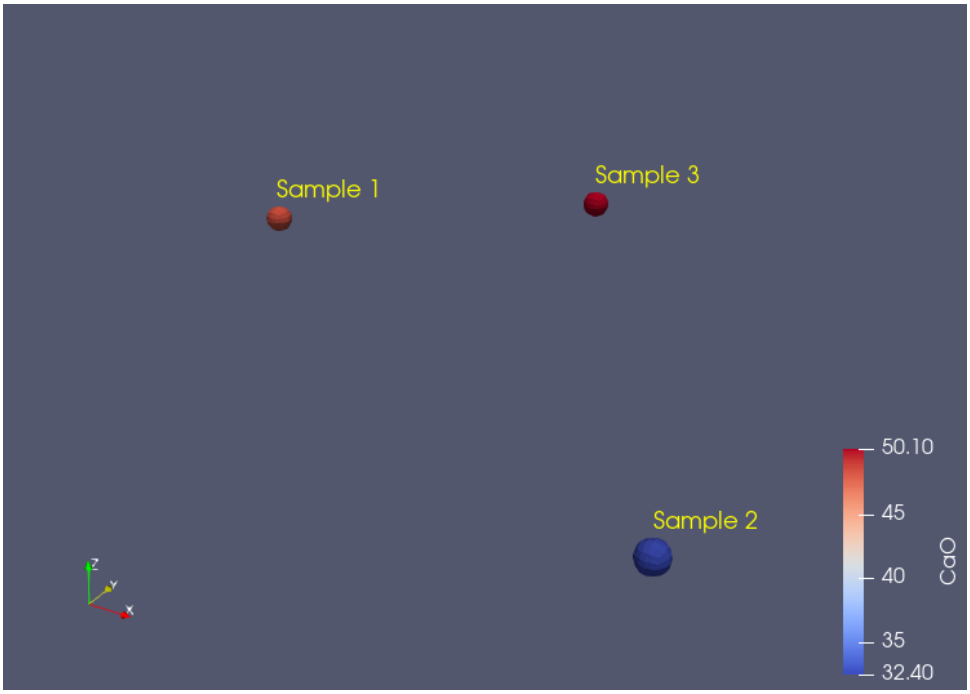


Fig. 1.39: Samples example displayed in an **AthosGEO View** Render View, with coloring according to the *CaO* attribute

Note

Sizing of the 3D symbols is not automatic, so if no 3D symbols do appear, the size should be increased manually. Again press *Apply* to see the effect.

Note

Normally, reading a data item in **AthosGEO View** will generate one display item in the **Pipeline Browser**, but the *SamplingReader* generates two, as shown in Fig. 1.40

Samplings The sampling data objects themselves, with attributes that can be chosen for coloring

Sampling Labels Labels next to the data objects that can be shown or hidden independently

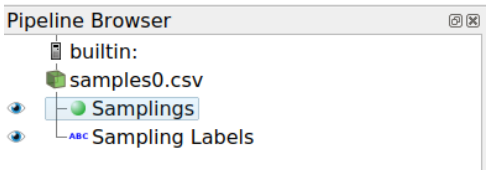


Fig. 1.40: Pipeline browser for a sampling, showing *Samplings* and *Sampling Labels*

1.5.3 Reading a Drilling Campaign from CSV Files

A typical drilling campaign data set comes with 3 tables, as specivied above (Section 1.5.1). In order to read these data into **AthosGEO View**, these tables need to be stored in CSV files, with the *assay* being mandatory and *collar* and *survey* being optional, if the required data are provided in the *assay* already.

The *assay* is what would be selected with the *Open* command, and a little example would look like shown in Table 1.5

Table 1.5: Assay data table

HoleID	SampleID	DepthFrom	DepthTo	CaO	SiO2
Hole 1	Sample 1	0	22	47.6	5.2
Hole 1	Sample 2	22	38	25.9	24.8
Hole 2	Sample 3	0	18	50.2	4.8
Hole 2	Sample 4	18	37	44.6	5.8
Hole 2	Sample 5	37	55	22.4	26.1

At this point, the **Properties** panel will appear as shown in Fig. 1.41. Within this panel, **Collar File** and **Survey File** need to be selected, as shown in Table 1.6 and Table 1.7.

Table 1.6: Collar data table

HoleID	X	Y	Z	HoleType
Hole 1	42662	16524	460	campaign 2008
Hole 2	42498	16766	455	campaign 2020

Table 1.7: Survey data table

HoleID	Azimuth	Dip
Hole 1	130	45
Hole 2	0	90

Note

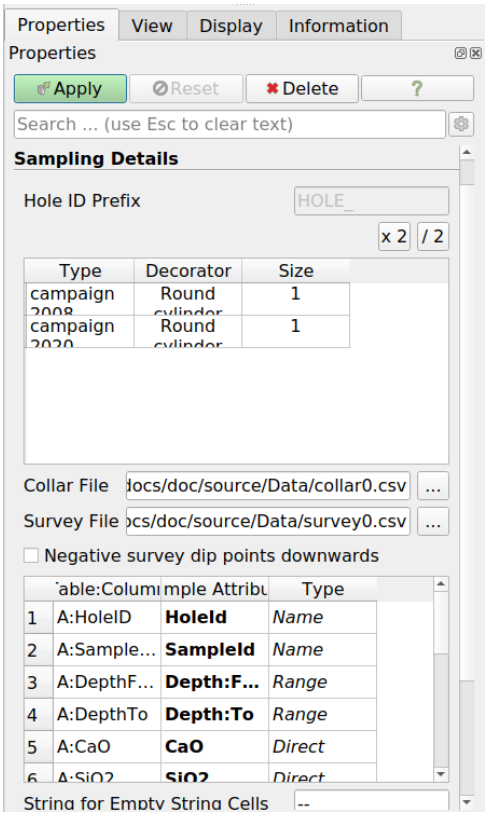


Fig. 1.41: Properties panel of the samples reader

The *Type* feature in the **Samplings Details** table lists the *HoleType* attribute from the *collar* table. To each one of these types, another *Decorator* can be assigned. See the list of possible drill hole decorators in Fig. 1.42.

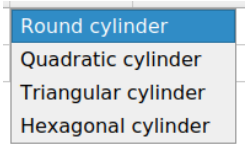


Fig. 1.42: Possible drillhole decorators

On pressing the *Apply* button, the *AthosGEO Render View* (3D view) will show the drillholes as colored cylinders as in Fig. 1.43

Did you know?

The way how **AthosGEO View** will distinguish between a **Set of Single Samples** and a **Drilling Campaign Data Set** is with the **HoleID** attribute: If it does not exist, it is not a drilling campaign.

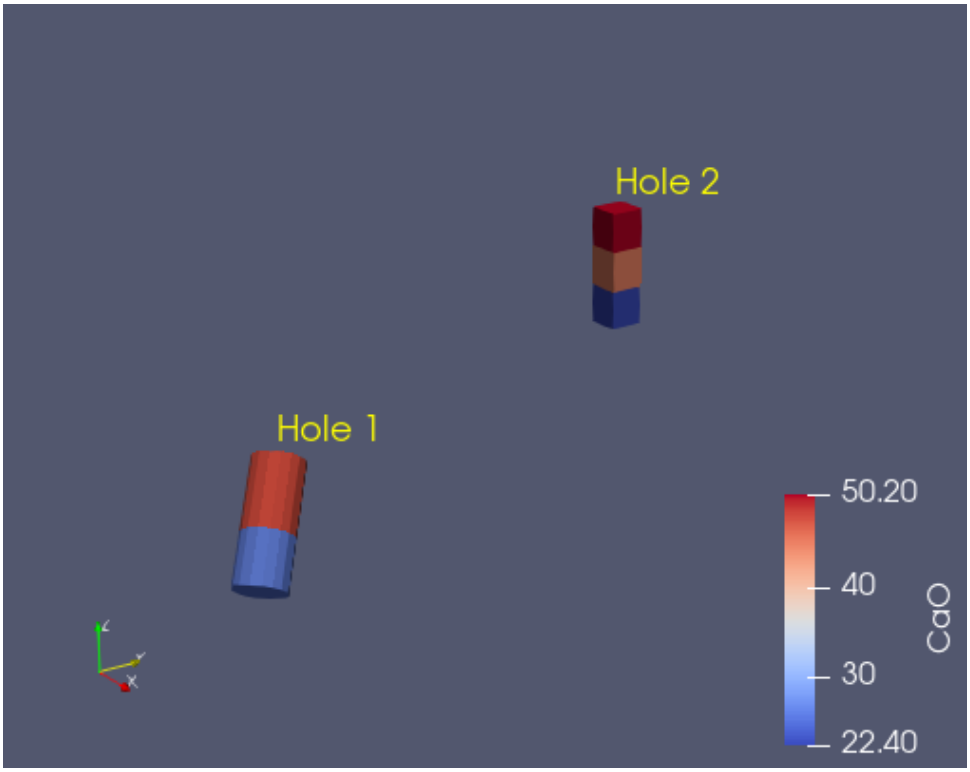


Fig. 1.43: Drillholes example displayed in an **AthosGEO View** Render View, with coloring according to the *CaO* attribute

1.5.4 Sampling Set Attributes

Please refer to the **Block Model Attributes** section of *Block Model Basics* which applies also to the attributes of sampling sets.

1.5.5 Viewing Sampling Sets

Sampling sets can be viewed with **AthosGEO View** in a 3D View (*AthosGeo Render View*) or as a numeric table (*AthogGeo Table View*), and analyzed with a great number of filters coming either from **ParaView** or from **AthosGEO View**.

Refer to *Display Sampling Set* of the **Reference Manual** for more.

1.6 Attribute Conventions

1.6.1 Attribute Category Example

For many functions of **AthosGEO View** and the derived software **AthosGEO Blend** it is makes things much easier if there is some clarity about the meaning of different attributes. This is done by assigning a specific *category* to each one of the attributes of a block model or sampling set. Instead of lengthy explanations, an example will be given in Table 1.8.

Table 1.8: Summarizing attributes of block model blocks

BlockId	KTons	SiO2	Al2O3	Fe2O3	CaO	LS	Geol
1	2.5	6.1	3.1	1.1	48.1	224.2	Limestone
2	3.0	4.2	2.5	1.8	52.3	329.3	Limestone
3	3.0	21.5	18.1	9.3	7.2	8.2	Marl
Total	8.5	10.9	8.2	4.2	35.1	82.1	

Referring to Table 1.8, different columns all need to be handled differently during summarizing, with the results shown in the last row:

KTons

Tonnages in terms of kilotons (1000 tons) are simply added:
$KTons_{total} = \sum_n KTons_n$

SiO2, Al2O3, Fe2O3, CaO

Chemical compounds are summerized as tonnage weighted means:
$CaO_{total} = \frac{\sum_n KTons_n CaO_n}{KTons_{total}}$

LS (lime saturation)

This is a cement specific modulus that is calculated from chemical compounts:
$LS = \frac{100CaO}{2.8SiO_2+1.18Al_2O_3+0.65Fe_2O_3}$
For the summary row, it is recalculated from the tonnage weighted means.

Geol, BlockId

These attributes cannot be summarized and will not appear in the summary attribute set

How is is possible that **AthosGEO View** always "knows" the correct category for each attribute? This is achieved by strictly standardizing the names of the attributes. How *attribute categories* are related to *summarizing operations* is explained in more detail in *Display Table (and other) Data* in *Summary Tables*.

1.6.2 Attribute Categories

Direct

These are proportions or percentages, like well known chemical compounds, cost per ton or other values that are related to tonnage.

For summarizing, tonnage weighted means will be calculated. In **AthosGEO Blend**, these attributes can be used in optimization targets (product constraints).

Any attribute that is not explicitly assigned to another attribute type, by filter rule or by containing string values, will be treated as being of this type, so it is the default attribute type.

Derived

Predefined cement specific derived attributes. They are calculated from chemical compounds (if available) with a special filter. They are also treated properly during summarization and can be used in constraints in **AthosGEO Blend**.

Tonnage

For summarizing, tonnages will be simply added. They can also be chosen as the tonnage attribute for the tonnage weighted mean calculation of *Direct* attributes.

Weight

Some filters can be applied to only some fraction (percentage) of a block, which can be specified with a *Weight* attribute.

Example: *Zone*. This attribute would specify how many percent of the block are inside a specific zone.

Category

This is slightly confusing: There is indeed an attribute category named *Category*. They are used to classify materials, so during summary calculation *Category* attributes will be dropped.

Category attributes can be either numeric or strings.

Note that in the **Summary View** it is possible to choose a *Category* attribute and then "scroll" from one value to the next.

Example: A *Category* attribute that corresponds to geological units can be either numeric codes (which is better for some filters) or strings (which is more human readable for example in a legend). With a **Summary View** it is then possible to scroll through the geological units, i.e. display them one by one.

Others

Some other categories are specified for different purposes within **AthosGEO View** or **AthosGEO Blend**:

- Coordinate
- Angle
- Dimension
- Index
- Period
- Range
- SpecTonnage

Note

In order to quickly check the types or categories of the different attributes within a model, the **Show AttributeTypes**  filter will display this information in a table.

1.6.3 Attributes during Import

If block models or sampling sets with attributes are read from CSV files, naming of the attributes is less strict than later on within the models. This is a convenience feature for the user.

Note that once all patterns are checked on importing an attribute name, any remaining attribute will remain unchanged and become category *Direct*.

Note

Attribute names must not contain spaces or other special characters. Accented characters may work, but there is no guarantee.

Case Conversion

Standardized names of attributes are case sensitive within the model, but not on import from CSV files. During the import, the names will be converted accordingly, like *cao* would be converted into *CaO*.

Alternative Attribute Names

During import, some alternative names are recognized and automatically renamed.

Example: The name *Tonnage* would also be recognized as meaning kilotons and would be renamed as *KTons* accordingly.

Attribute Name Patterns

In some cases, only part of the name needs to be standardized, while the user can attach some descriptive part related to a project.

Example: The attribute name *Zone_<..>* could become *Zone_LeaseA* or *Zone_Reserve* etc. and still be recognized as an attribute of category *Weight*.

Multi-Component Attributes

It is a special feature of **ParaView** that it allows attributes to have multiple components, the most typical being coordinates with *X*, *Y* and *Z* component. CSV or other table data formats do not support such a feature, so these attributes would be split into several component attributes with names like *Coord:X*, *Coord:Y*, *Coord:Z*, or else *Coord:0*, *Coord:1*, *Coord:2*. These two forms are equivalent in this case. Important is the colon (**:**) separator between attribute name and component.

Table of Categories and Attributes

Table 1.9: Categories and Attributes

Category	Attribute	Alternatives	Unit	Description
Direct	LOI		%	Loss on ignition
Direct	SiO2		%	
Direct	Al2O3		%	
Direct	Fe2O3		%	
Direct	CaO		%	
Direct	MgO		%	
Direct	SO3		%	
Direct	K2O		%	
Direct	Na2O		%	
Direct	TiO2		%	
Direct	Mn2O3		%	
Direct	P2O5		%	
Direct	Cr2O3		%	
Direct	Cl		%	
Direct	F		%	
Direct	Hg		%	
Direct	TOC		%	Total organic carbon
Direct	OfferedPerc		%	Offered percentage of a block
Direct	Offered-Perc_<product>		%	Offered percent of a block for a product

continues on next page

Table 1.9 – continued from previous page

Category	Attribute	Alternatives	Unit	Description
Direct	TakenPerc		%	Taken percentage of a block
Direct	TakenPerc_<product>		%	Taken percentage of a block for a product
Direct	WastedPerc		%	Wasted percentage of a block
Direct	WasteRatio		%	Waste ratio for block: wasted tonnage for this block related to taken tonnage
Direct	VolFactor	volumefactor, fillingdeg*	%	Volume fraction of the block that is not air
Direct	<any other>			
Derived	LS			Lime saturation
Derived	SR			Silica ratio
Derived	AR			Alumina ratio
Derived	ASR			Alkali sulfur ratio
Derived	NaEq		%	Sodium equivalent
Tonnage	KTons	tonnage	kt	kilotons = 1000 tons
Tonnage	Tons		tons	
Tonnage	Tons_<product>		*t/kt*[*]	
Tonnage	Offered		*t/kt*[*]	
Tonnage	Offered_<product>		*t/kt*[*]	
Tonnage	Taken		*t/kt*[*]	
Tonnage	Taken_<product>		*t/kt*[*]	
Tonnage	WastedForBlock		*t/kt*[*]	Wasted tonnage for extracted tonnage in this block
Tonnage	Wasted	waste, stripping	*t/kt*[*]	
Weight	Pit		%	Percentage of block inside of pit
Weight	Pit_<..>		%	Percentage of block inside of named pit
Weight	Zone		%	Percentage of block inside of zone (delimited by a closed boundary line)
Weight	Zone_<..>		%	Percentage of block inside of named zone
Category	Code			Categorizing by numeric or named code
Category	Code_<..>			Dto. with additional name
Category	Geology	geol<..>		Geological unit
Category	Class			Like Code
Category	Class_<..>			Dto. with additional name
Category	Product			Product name
Category	Level			Block level
Category	Level_<quarry>			Block level in specific quarry
Category	Column			Block column
Category	Column_<quarry>			Block column in specific quarry

continues on next page

Table 1.9 – continued from previous page

Category	Attribute	Alternatives	Unit	Description
Category	Row			Block row
Category	Row_<quarry>			Block row in specific quarry
Category	Line			Like Row
Category	I			Like Column
Category	J			Like Row
Category	K			Like Level
Category	Cat			Like Code
Category	Name			Classification name
Category	Start	begin, open, first		In **AthosGEO Blend** for specification of blocks where to start the scheduling
Category	Force			In **AthosGEO Blend** for specification of blocks blocks to be taken in any case for one of the products during optimization or scheduling
Category	HoleType	hole_type, sond*meth*, meth*sond*		Classification attribute for drillholes or sampling sets
Category	HolePath	hole_path		Type of drillhole path - so far ignored and always handled like “linear”
Category	N_<..>			On reading from CSV files, all columns with strings that are not otherwise classified are getting a N_ prepended to their name and with this they are classified as Category attributes
Angle	Angle	ang*, rot*	°	Orientation of block model, counting N-E-S-W
Angle	Azimuth		°	Orientation of inclined drillhole
Angle	Dip		°	Dip angle of inclined drillhole, 90° for vertically downwards
Coordinate	Center:X	center:0, center_0, cent*x, world*x, x*world, x, east*		Block center X coordinate
Coordinate	Center:Y	center:1, center_1, cent*y, world*y, y*world, y, north		Block center Y coordinate
Coordinate	Center:Z	center:2, center_2, cent*z, world*z, z*world, z, alt		Block center Z coordinate
Coordinate	Collar:X	collar:0, collar_0, collar*x, sondage*x		Drillhole collar X coordinate

continues on next page

Table 1.9 – continued from previous page

Category	Attribute	Alternatives	Unit	Description
Coordinate	Collar:Y	collar:1, collar_1, collar*y, sondage*y		Drillhole collar Y coordinate
Coordinate	Collar:Z	collar:2, collar_2, collar*z, sondage*z		Drillhole collar Z coordinate
Date	Date			ISO-Format: YYYY-MM-DD
Dimension	Size:X	size:0, size_0, size*x*, dx, width		Block size in X direction
Dimension	Size:Y	size:1, size_1, size*y*, dy, length		Block size in Y direction
Dimension	Size:Z	size:2, size_2, size*z*, dz, height		Block size in Z direction
Dimension	Slope:XNeg	slope:0, slope_0, slope:xneg, slopexneg		Mining slope angle in negative X axis *direction***
Dimension	Slope:XPos	slope:1, slope_1, slope:xpos, slopexpos		Mining slope angle in positive X axis *direction***
Dimension	Slope:YNeg	slope:2, slope_2, slope:yneg, slopeyneg		Mining slope angle in negative Y axis *direction***
Dimension	Slope:YPos	slope:3, slope_3, slope:ypos, slopeypos		Mining slope angle in positive Y axis *direction***
Dimension	Pref:XNeg	pref:0, pref_0, pref:xneg, prefxneg		Mining preference in negative X axis *direction***
Dimension	Pref:XPos	pref:1, pref_1, pref:xpos, prefxpos		Mining preference in positive X axis *direction***
Dimension	Pref:YNeg	pref:2, pref_2, pref:yneg, prefyneg		Mining preference in negative Y axis *direction***
Dimension	Pref:YPos	pref:3, pref_3, pref:ypos, prefypos		Mining preference in positive Y axis *direction***
Dimension	Pref:ZNeg	pref:4, pref_4, pref:zneg, prefzneg		Mining preference in negative Z axis direction
Dimension	NumItems	numitems		In summary tables the number of summarized items (blocks etc.)
Dimension	MaxDepth	depth, max_depth		Maximum length of a drillhole
Dimension	Volume	vol, volume		Block volume that is not air
Index	BlockId			
Index	Nb:XNeg	nb:0, nb_0, nb:xneg, nbxneg		Neighbor block ID in negative X axis *direction***
Index	Nb:XPos	nb:1, nb_1, nb:xpos, nbxpos		Neighbor block ID in positive X axis *direction***

continues on next page

Table 1.9 – continued from previous page

Category	Attribute	Alternatives	Unit	Description
Index	Nb:YNeg	nb:2, nb_2, nb:yneg, nbyneg		Neighbor block ID in negative Y axis *direction***
Index	Nb:YPos	nb:3, nb_3, nb:ypos, nbypos		Neighbor block ID in positive Y axis *direction***
Index	Nb:ZNeg	nb:4, nb_4, nb:zneg, nbzneg		Neighbor block ID in negative Z axis direction
Index	Nb:ZPos	nb:5, nb_5, nb:zpos, nbzpos		Neighbor block ID in positive Z axis direction
Name	Material	material, corr*		Material or corrective name
Name	RowType	rowtype		Row type in summary table
Name	HoleId	holeid, hole_id, dhid, dhnr, name*hole, sond*name, name*sond*		Unique ID of a drillhole
Name	SampleId	sampleid, samp*id, prob*name, name*prob*		Unique ID of a sample
Period	Period	miningperiod		
Period	FirstPeriod			First period of taking tonnage from a block
Period	WastePeriod			Period where wasted tonnage was taken
Range	Depth:From	depth:0, depth_0, depth*from, schicht*von		Depth range begin of sample within a drillhole
Range	Depth:To	depth:1, depth_1, depth*to, schicht*bis		Depth range end of sample within a drillhole
SpecTonnage	Density	dens*, sg, spec*		Density or specific gravity of material

* **t/kt** stands for either *tons* or *ktons* (1000 tons).

** **direction:** negative or positive X or Y direction refers to a model with **Angle** of 0°. Otherwise, everything is rotated accordingly.

1.7 Surfaces and Lines

Besides the visualization of block models and sampling sets, **AthosGEO View** is able to visualize all kinds of geometric items that also **ParaView** would handle. For the purpose of geodata visualization, it is often desirable to add items like:

- **Topography**, including a projected *aearial photo*, a *geological map*, a *topographic map* or any other kind of mapping.
- **Boundary lines** (closed polylines) like lease or any other aereal boundary, or any kind of other **polylines** showing important features.

Note

The following will give only a small impression of the possibilities to show additional geometries with **AthosGEO View**. Please have a look into the **ParaView** documentations for further instructions: *First Steps*.

1.7.1 Triangulated Surfaces

Reading from File

The native file format for triangulated surfaces in VTK is a *polydata file* (with .vtp extension). With **AthosGEO View** comes the additional possibility to read them also from .dxf files, and from **ParaView** comes the ability to read even more formats.

Reading from GeoTIFF File

AthosGEO View is able to read a topography from a TIFF file if it contains rastered altitude values in GeoTIFF format.

The displayed output will be a triangulated surface that is automatically generated and optionally reduced (decimated) from the input raster data.

Regarding the reduction options see Section 1.7.1.

Generating from Points

If a series of surface point coordinates (x, y, z) are available, a surface can be triangulated between them with **AthosGEO View**. The starting point could be a CSV file with a table like in Table 1.10.

Table 1.10: Topo points example

x	y	z
10200	24300	190
10200	24310	192
10200	24320	195
10200	24330	199
10200	24340	205
10200	24350	212
10200	24360	215
10200	24370	217
10200	24380	218
10200	24390	217
10200	24400	214
10210	24300	191
10210	24310	192
etc.		

In **AthosGEO View**, open the points CSV file and choose the **CSV Reader** for loading the data into a table as in Fig. 1.44.

On pressing *Apply*, a *SpreadSheet View* will open and show the points table as in Fig. 1.45.

Next, these tabular data need to be converted into geometric points in space by applying the **Table to Points** filter.

Did you know?

A quick way to get that **Table to Points** filter is through *Filters* → *Search...* and starting to type the word *Table*: The filter will immediately appear in the list and with a click and pressing *Enter* it is attached to the *filter pipeline*.

In the appearing **Properties** panel of the **Table to Points** filter, table columns for the x, y and z coordinates need to be specified, see Fig. 1.46.

Hint

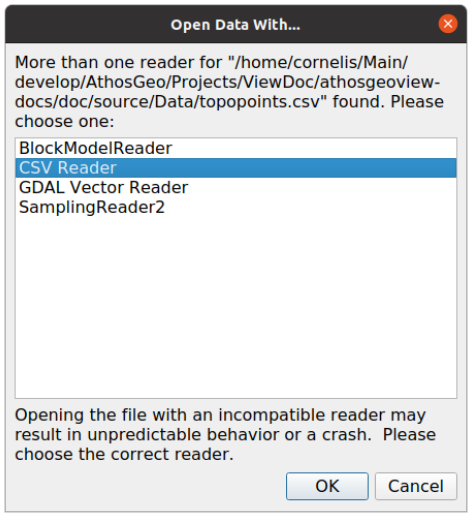


Fig. 1.44: The *CSV Reader* reads a CSV file and displays it as a table in a *SpreadSheet View*.

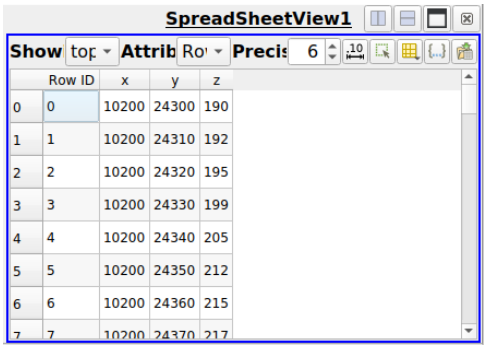


Fig. 1.45: Point coordinates displayed in a *SpreadSheet View*.

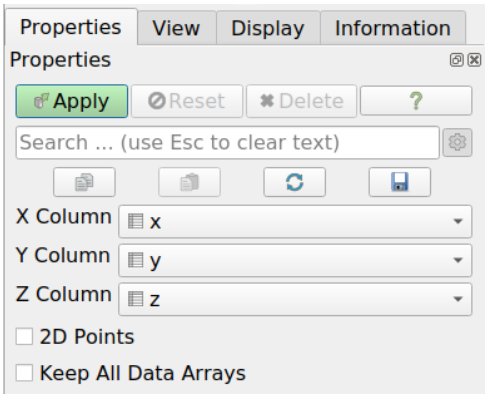


Fig. 1.46: The **Properties** panel of the *Tables to Points* filter.

Before pressing *Apply*, make sure that a *render view* (3D view) is activated: Only then you will see the new points immediately, while otherwise they will be displayed in the *SpreadSheet View* that was previously showing the table data.

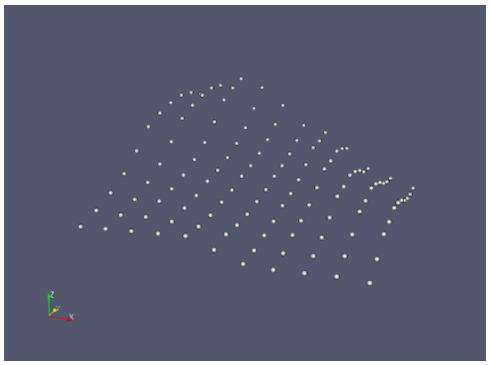


Fig. 1.47: Display of points in an *AthosGEO Table View*.

With this we have free points in space as shown in Fig. 1.47 and the next step would be to triangulate a topo surface between them using the **Delaunay 2D** filter.

Note

Two other filters look like they would do the same or similar things, but this is not true:

- The **Triangulate** filter does not triangulate a surfass between points, but converts surfaces that are made up of face elements (cells) that are more complex than simple triangles (like quads, polygons or triangle strips) into plain triangulated surfaces. This is a requirement for certain other filters that cannot handle these more complex surface elements.
- The **Delaunay 3D** filter might look like the adequate filter for our purpose, because we want to generate a surface in 3D space. However, This filter actually does not generate triangles, but tetrahedra, filling the void space inside a shell.

The **Properties** of the *Delaunay 2D* filter is shown in Fig. 1.48. xx

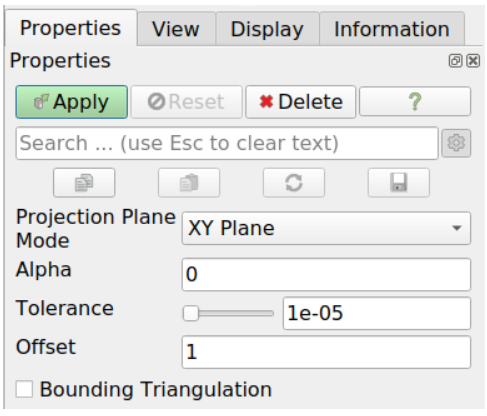


Fig. 1.48: The **Properties** panel of the *Delaunay 2D* filter that will triangulate a surface between a set of points.

Select the **XY Plane** option for generating a topo surface.

The **Tolerance** is the distance between points below which are considered identical and merged. For a topo with coordinates in meters, the default value might be too small, so better increase it to something like 0.01.

After pressing *Apply*, a surface will appear as shown in Fig. 1.49.

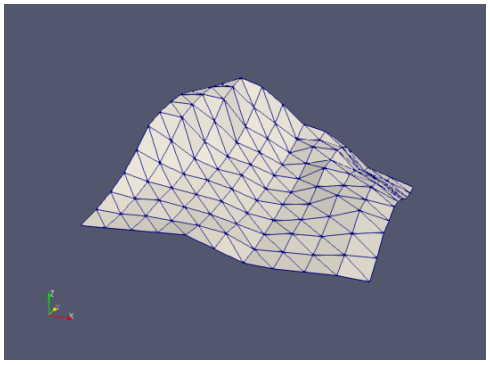


Fig. 1.49: A triangulated surface generated from input points using the *Delaunay 2D* filter.

At this point, a new triangulated surface is often too finegrained, with too many points and triangles to best represent the topo surface. A clever way to reduce the number of triangles in order to optimize the representation is described in Section 1.7.1.

Reduction of Triangulated Surfaces

The **Decimate** filter (to be found at *Filters* → *Geometry*) allows to reduce the number of triangles in a triangulated surface in an intelligent way, preserving morphologic features as much as possible within the limits of the selected parameters, see Fig. 1.50.

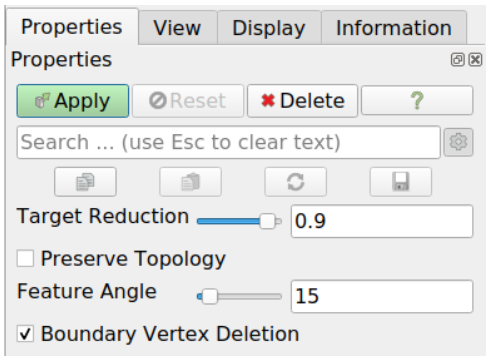


Fig. 1.50: The **Properties** panel of the *Decimate* filter allow to finetune the resulting output triangulation.

Did you know?

The **Decimate** filter works only for surfaces that are plain triangulated surfaces without any other surface elements, like quads, polygons etc. If this is not true for a given surface, the **Triangulate** filter needs to be applied first.

The two most important **Properties** of the *Decimate* filter are:

- **Target Reduction:** A value of 0.9 means that the number of triangles should be reduced by 90% if this is possible while respecting the other constraints of the reduction. If the reduction looks too extreme, reduce this value to retain more triangles.
- **Feature Angle:** Any common edge between two triangles where the angle between the triangles is more than the feature angle, must not be removed. So if the result of a decimation looks like retaining morphologic features not good enough, reduce that angle, or vice versa.

Example: The result of reducing the surface shown in Fig. 1.49 with the *Decimate* filter default properties as shown in Fig. 1.50 will produce a result as shown in Fig. 1.51. Visually, a much less radical reduction with *Target Reduction* of only **0.3** and a *Feature Angle* of **10** seems to generate a better result, as shown in Fig. 1.52.

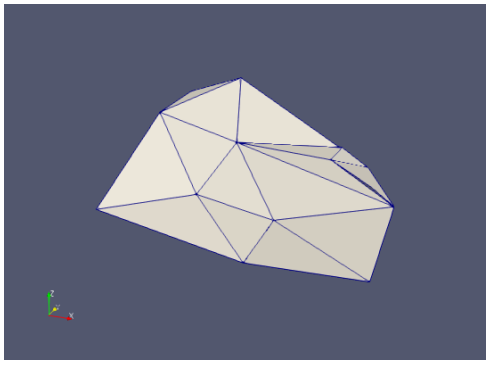


Fig. 1.51: The surface shown in Fig. 1.49 after application of the *Decimate* filter with default properties.

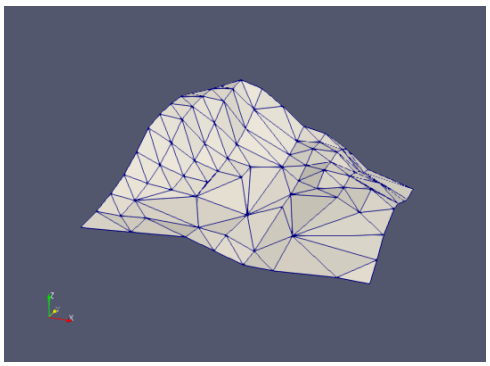


Fig. 1.52: The surface shown in Fig. 1.49 after application of the *Decimate* filter with a much more optimum set of properties (*Target Reduction* = 0.3, *Feature Angle* = 10).

The optimum set of properties for the *Decimate* filter will depend on the input triangulated surface and the requirements of retaining morphologic features with higher or lower precision.

Reduction of Polylines

The **Decimate Polyline** filter will reduce the number of points definint a polyline in such a way that the geometry will be affected as little as possible.

Writing Triangulated Surfaces and Polylines

The native VTK format for triangulated surfaces and polylines is a *Polydata File*, with **.vtp** extension.

Writing them to **.dxf** file is an extension of **AthosGEO View** to allow data exchange with softwares that prefer such a format.

A number of other formats are already supported by **ParaView**.

1.7.2 Projected Maps

Georeferencing

Topo surfaces are having coordinates from their points, but images with maps or aerial photos need to be xplicitly georeferenced in order to be correctly projected onto a topo. Two systems are recognized in **AthosGEO View**:

- ESRI worldfiles are small plain text files describing the positioning of the image relative to a geographical reference grid. If they exist, **AthosGEO View** will find and use them.x
- In the case of GeoTIFF, the georeferencing information is incorporated within the image file itself. If a TIFF image file contains such information, **AthosGEO View** will use it.

The *Texture Georeferenced* Filter

The **Texture Georeferenced** filter asks for the name of a georeferenced texture (image) file, as specified above.

Next, select the **Topo with Map Texture** representation, instead of the common **Surface** or **Surface with Edges** representations.

The example topo with a projected example map Fig. 1.53 would look like shown in Fig. 1.54.

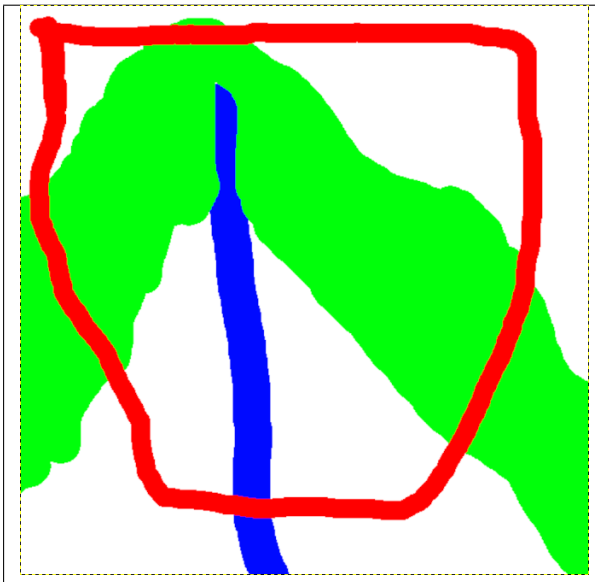


Fig. 1.53: The **map** to be projected on the **topo**

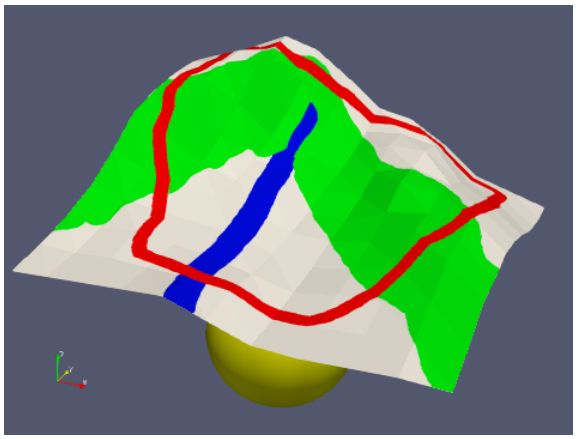


Fig. 1.54: An example **topo** with the projected example **map**.

Hints

The **Texture Georeferenced** filter actually adds the full texture file name and path to the *Field Data* of the triangulated surface (topo). That means, if the output of that filter would be saved to a ***.vtp** file, also that texture file name will be saved.

If now that ***.vtp** file is loaded back to **AthosGEO View**, if the representation is then switched again to **Topo with Map Texture**, and if finally the **texture file** is still present in the same path, the topo projection will be back even without applying the **Texture Georeferenced** filter again.

Textures with Transparency

By default, the **Ignore Alpha** property of the **Texture Georeferenced** is **on**, meaning: If a texture map file has an alpha channel, care will be taken that the entire texture map is interpreted as being **opaque**.

If that property is **off**, an alpha channel will be respected, with dramatic effects: It does not only affect texturing of the topo surface, but it makes the topo surface itself **fully or partially transparent**.

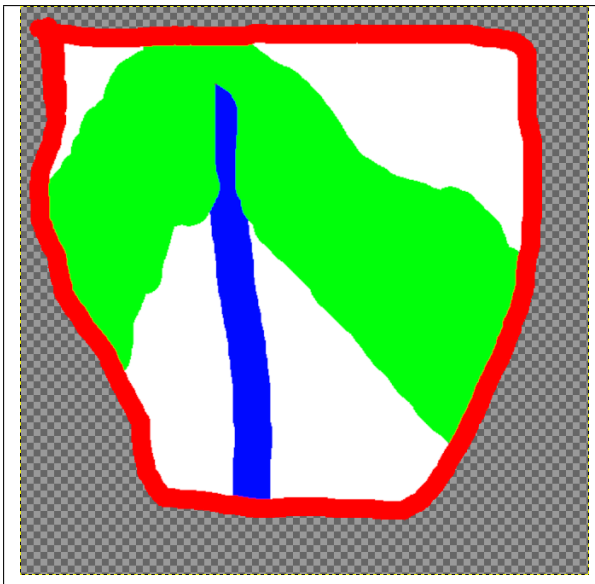


Fig. 1.55: This example **map** is **fully transparent** in the areas that are not of interest

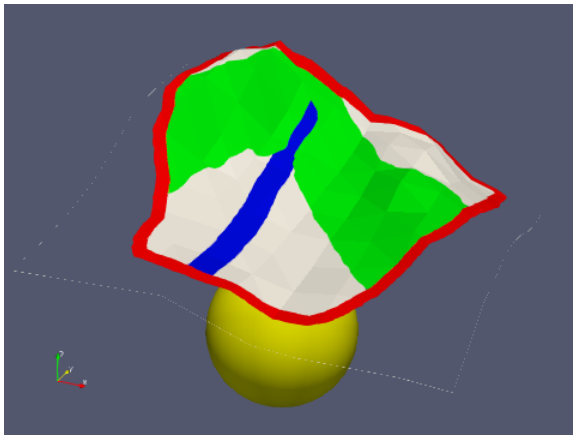


Fig. 1.56: The effect is like **clipping** the topo along the boundary of the range of interest

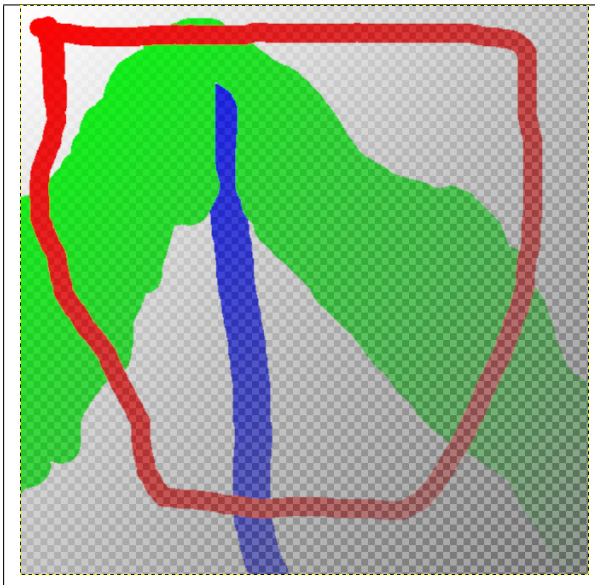


Fig. 1.57: This example **map** has a **transparency gradient** from the upper left to the lower right corner

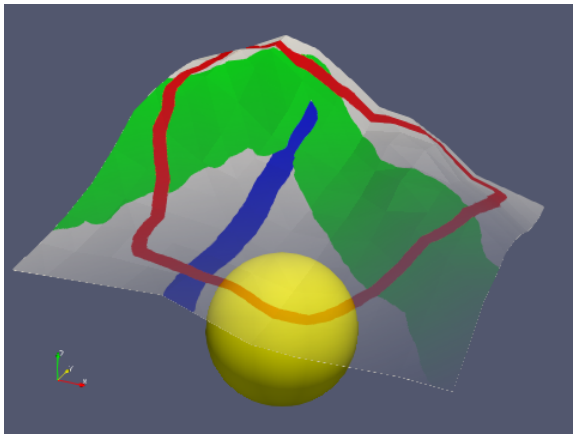


Fig. 1.58: The effect is a textured topo surface with **varying transparency** (The underlying sphere becomes increasingly visible)

1.7.3 Lines, Polylines and Polygons

Reading from File

The native file format for lines, polylines and polygons in VTK is a *polydata file* (with .vtp extension). With **AthosGEO View** comes the additional possibility to read them also from .dxf files, and from **ParaView** comes the ability to read even more formats.

Manually Generating Polylines or Polygons

With **AthosGEO View** it is possible to manually digitize lines, polylines and polygons using the **Poly Line Source**. It can be found at *Sources* → *Geometric Shapes*. With this, a **Properties** panel will appear as shown in Fig. 1.59.

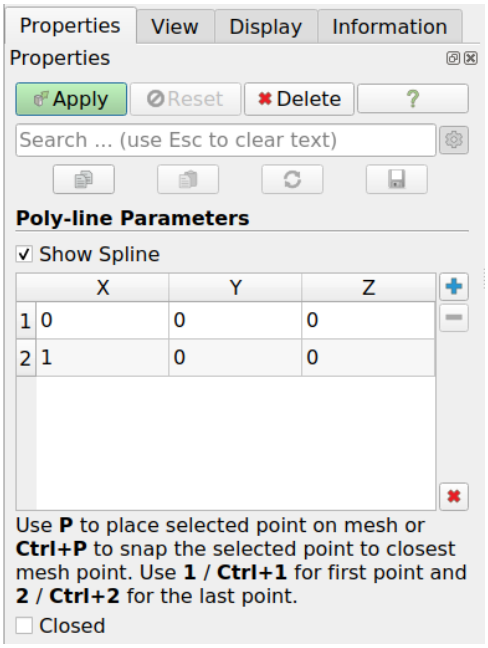


Fig. 1.59: The **Poly Line Source** *Properties* panel explains the different steps to manually digitize a *polyline*. If the **Closed** checkbox is checked, the polyline will be automatically closed, generating a *polygon*. With the +, - and *x* buttons, single points can be added or removed, or the entire list will be cleared.

With a map in the background, a drawn polygon can be digitized as a geometric *polygon* object for further processing or activities, as shown in Fig. 1.60.

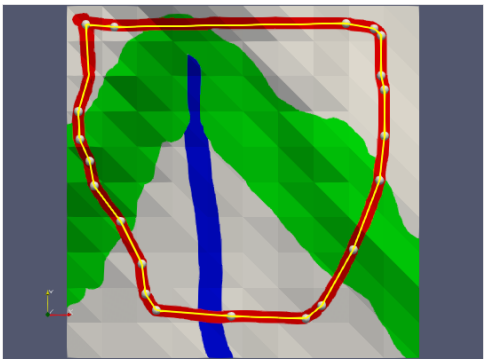


Fig. 1.60: The yellow *spline* was manually adapted to the *red outline* in the example map.

Hints

- Make sure that the projection is **parallel**, and this can be achieved by unchecking the *Camera Parallel Projection* checkbox in the *View* panel.
- Make sure that the view is exactly vertical, which can be achieved by pressing the -*Z* button in the toolbar.
- It is possible that the *spline* disappears behind the topo surface because the *Z coordinate* is too low. It is not possible to directly change the Z coordinate in the *Properties* panel, so some tricks need to be applied. One possibility:
  - After activating the *Poly Line Source* and ensuring the *parallel projection*, rotate to a side view, e.g. by pressing +*Y* in the tool bar.
  - Move the mouse to the *highest point* in the model (maybe a zoom in is required for the precision) and press both the *1* and the *2* buttons, one after the other, in order to bring both the start and the end point to that elevation.
  - Press the -*Z* button to change to the top view. Start and end point are now coinciding, but visible, at the highest possible elevation.
  - Now move the start and end points to where they should be: the *Z coordinate* will remain at the elevation of the top point of the topo, and the same is true for all points that are added from now on.

At the end, do not forget to press the *Apply* button, because the *spline* that can be manipulated with with the mouse, is not yet the *geometric object* (polyline, polygon) that can be visualized, saved or otherwise further processed.

1.8 Color Display

1.8.1 Color Mapping

For the 3D display (*Athos Render View* or *Render View*), attributes can be chosen for the visualization, with the effect that automatic color mapping will be applied:

- The full value range of the attribute will be calculated, and a predefined default color scale will be mapped to that value range.
- A legend will be shown that explains the value mapping to colors.

All the above can be adapted by the user, from the value range via the color palette to the visibility and location of the legend.

Hint

Automatic color mapping also works for attributes that are **categories** with names, like geological unit names. In this case, the value strings will appear in alphabetical sort order in the legend.

This may not always be the desired ordering, i.e. in the case of geological units that would better appear in an order from top to bottom. Right now the only way to achieve this is to prepend the names with a letter or number like this:

- A Upper Limestone
- B Upper Shale
- C Lower Limestone
- etc.

1.8.2 Direct Color Display

Much less obvious is the fact that colors can also be directly assigned to either points or cells (model blocks, topo triangles etc.). This works with attributes that have 3 components, representing the Red, Green and Blue color components.

If the attribute is of an integer data type (i.e. no fractions), these color component values are assumed to be in the value range between 0 and 255. If the attribute is of a floating point type (float or double), the value range must be between 0.0 and 1.0 to cover the full color space.

Finally it is important to turn off the **Map Scalars** option in the **Display** panel to get the desired color display - see Fig. 1.61.

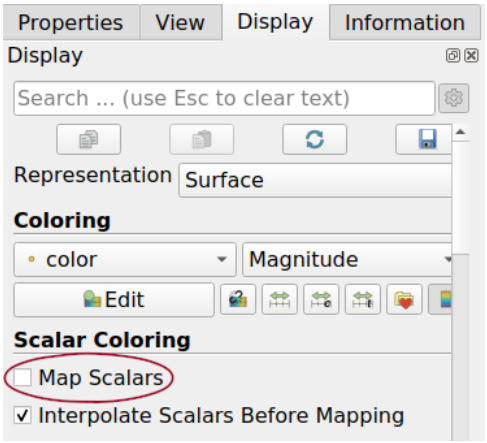


Fig. 1.61: By default, automatic attribute to color mapping is always active and very useful. However, if specific colors are coded in an attribute (3 components, representing red, green and blue), it has to be turned off explicitly in the **Display** panel.

Example: The *Fixed Colors from Categories* Filter

As mentioned above, the default display of **AthosGEO View** handles also attributes of **category** type, like codes, names or labels, assigning each one a unique color. However, if more control of the colors is required (and a legend is of less importance), here is a filter that assigns explicit colors to data items by category.

A possible use case: Some well known standard coloring for well known categories, like an internal standard definition. In such a case, a CSV file needs to be prepared that reflects this color standard, with the following columns:

- **Category** *(optional)*: name of the category to which a color definition applies
- **Value** *(mandatory)*: category value to which the color applies
- **Colors:red**, **Colors:green**, **Colors:blue** (or *Colors:0*, *Colors:1*, *Colors:2*): color components
- **ColorAlpha** (or *Colors:alpha* or *Colors:3*) *(optional)*: alpha value (transparency). Note that this is only for storing a required transparency value, but transparency display will will not happen that way (and is not possible to specify explicitly with the software, only as part of color mapping)

CHAPTER

TWO

ATHOSGEO VIEW DATA VISUALIZATION AND ANALYSIS

2.1 Display a Block Model

2.1.1 Block Model in 3D

Block models can be visualized in 3D with the **Athos Render View** or with the classical **Render View**. The following **Representations** are able to display a block model:

Surface, **Surface With Edges**

This is the most common representation of a block model and allows to color the model blocks (cells) by attributes, with or without the block edges.

Category, **Category With Edges**

This representation is like the **Surface** representations, but allows to "scroll" through the values of a **Category** attribute. Attribute and values can be selected in the **Display** tab of the properties panel.

Wireframe

Display only the block edges, with attribute coloring.

Feature Edges

Show only the overall shape of the block model with lines, without attribute coloring.

Volume

A representation that uses transparency to allow a view into the inner volume of a block model. The color scale is linked with a transparency scale, typically higher values more opaque and lower values more transparent. This is useful if a block model contains "lenses" or other shapes of higher concentrations of some chemical compound.

The other available representations (*Points* etc.) are less useful because the block model attributes are **cell data**, not **point data**, so the geometry can be visualized, but not the attributes with such representations.

2.1.2 Block Model as Points

For certain purposes it is helpful to display a block model not as *cells* (representing the blocks of the model), but as *points*. One thing could be the display with **Points**, **Point Gaussian** or **3D Glyphs** representation. Another one could be the generation of a **box plot** as described in *Box Plot*.

The filter for this purpose is **Cell Centers**: It generates a dataset of only points without cells, located at the centers of the cells, and assigns the *cell data* to the *point data*, so the display with coloring by attributes is now also possible with the points.

Be Careful

At first sight it looks like the **Cell Data to Point Data** filter would be the way to go if we need point data for display, but this is not entirely true. The difference between the two filters is the following:

Cell Centers filter

The center points of every cell, thus our model blocks, are calculated for every cell (block), then the cell data are assigned to the newly generated center points.

Cell Data to Point Data filter

For every existing point in the block model, i.e. all the points where cells (model blocks) are touching, the **average** values of the attributes of the adjoining cells are calculated and assigned to these points. **Note that with this the original information from the blocks is not any more represented.**

2.1.3 Block Model Data in Table or Chart Views

Block model data can also be displayed as tables or analyzed with charts, in pretty much the same way as other tabular data: see *Display Table (and other) Data*.

2.1.4 Filters Supporting Block Model Data Analysis

A number of filters can help to visualize and analyze the data of a block model from different points of view. Some of them will be presented in the following sections.

Clip and Slice Filters 

A first visual impression of the distribution of qualities within a block model can be gained by applying a clip or slice filter, as shown in Fig. 2.1.

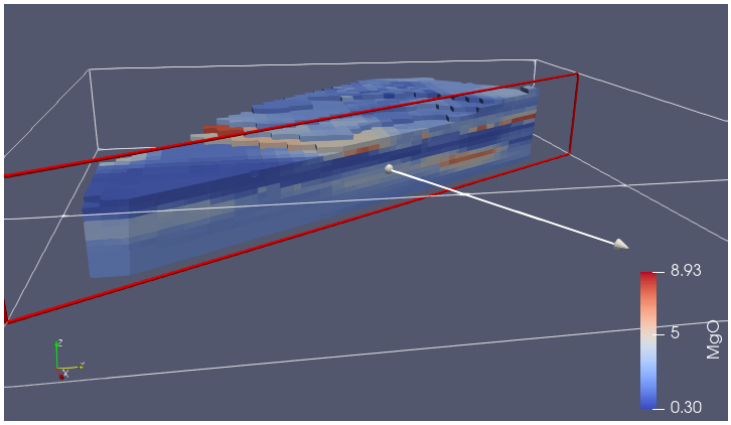


Fig. 2.1: The **Clip Filter** lets the user interactively move both the **clip plane** and the **normal vector**, and by pressing *Apply*, the model will be clipped along the current plane.

Fence Diagram

In geology and other earth sciences, **fence diagrams** are a common way to visualize the inside of a rock volume. With **AthosGEO View**, a fence diagram can be generated with the **Slice** filter with some advanced properties, so the first step is to make sure that the **Advanced** features are activated with the  button.

Now in the lower part of the **Properties** panel a list of values becomes visible that can be managed with the buttons at it's right. The effect is that not only one slice will be generated, but any number of slices, with equal or variable distances, as shown in Fig. 2.2.

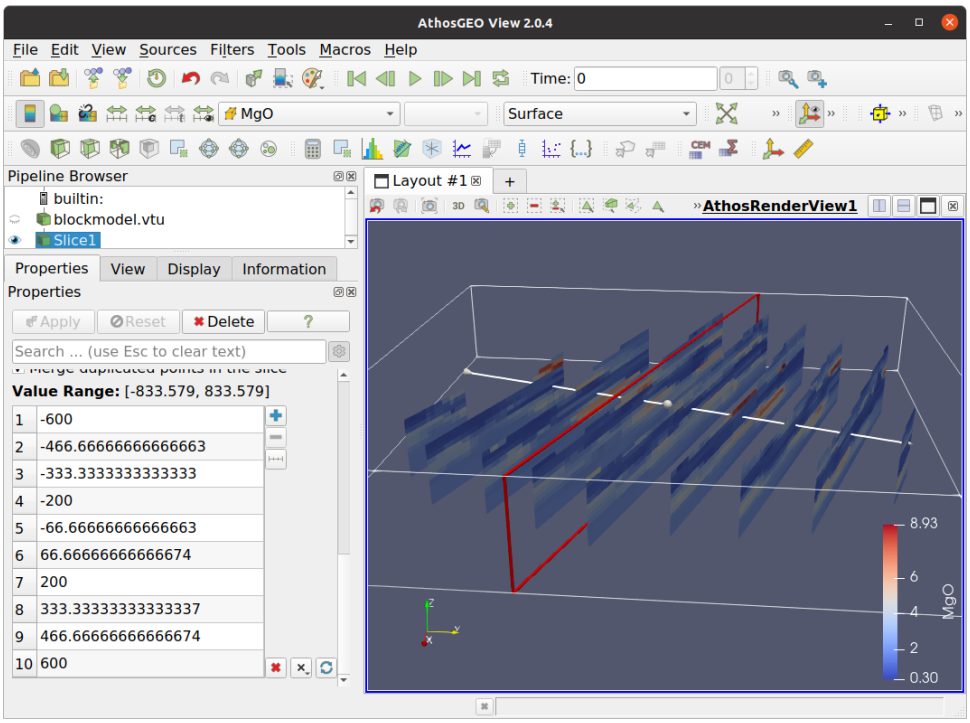


Fig. 2.2: The **Slice** filter allows to generate **fence diagrams** by defining a sequence of numerical offset values in the **advanced** properties of the **Properties** panel.

Threshold 

With the **Threshold** filter, blocks of the block model can be extracted that are within a specified **value range** of an **attribute**, as shown in Fig. 2.3.

Block Attributes I: Hover on Cells

A quick way to see a list of all attribute values of a model block is to press the  button at the top of the 3D (Render) view. With this turned on, the mouse can be moved over the block model, and wherever it stops, a text pops up with a list of the attributes, as shown in Fig. 2.4.

This very easy method hits it's limits in models with a high number of attributes, because in this case that function does not work properly: The list pops up, but if it is too large to fit properly on the screen, it will close immediately.

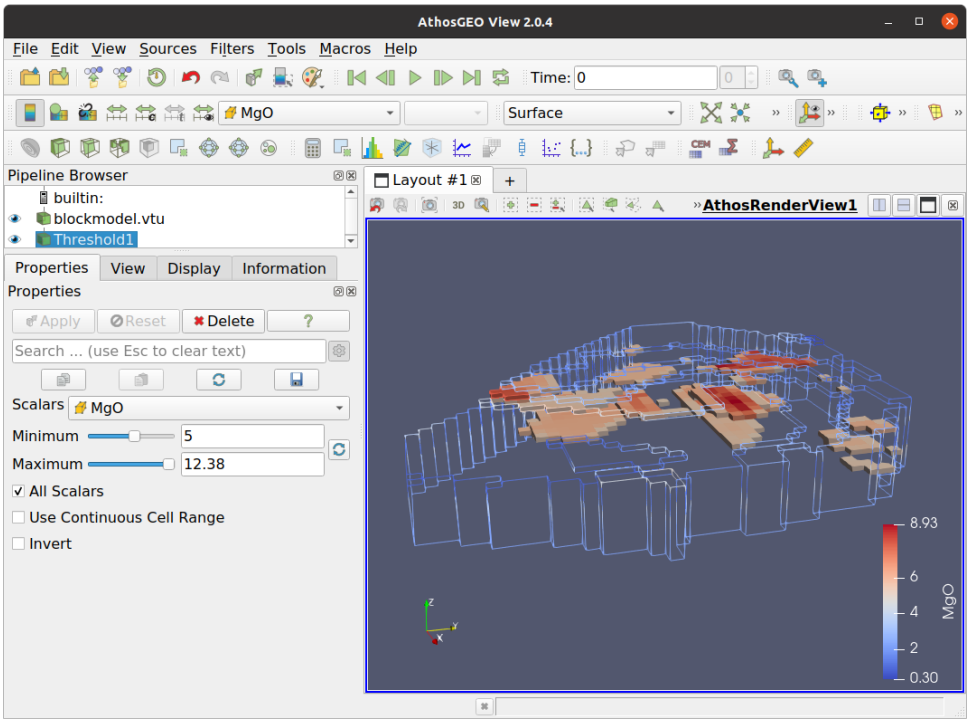


Fig. 2.3: In this example, **blocks with MgO above 5%** are extracted and shown inside a wireframe display of the entire block model. With this, zones of highest MgO content are easily recognized inside the rock volume.

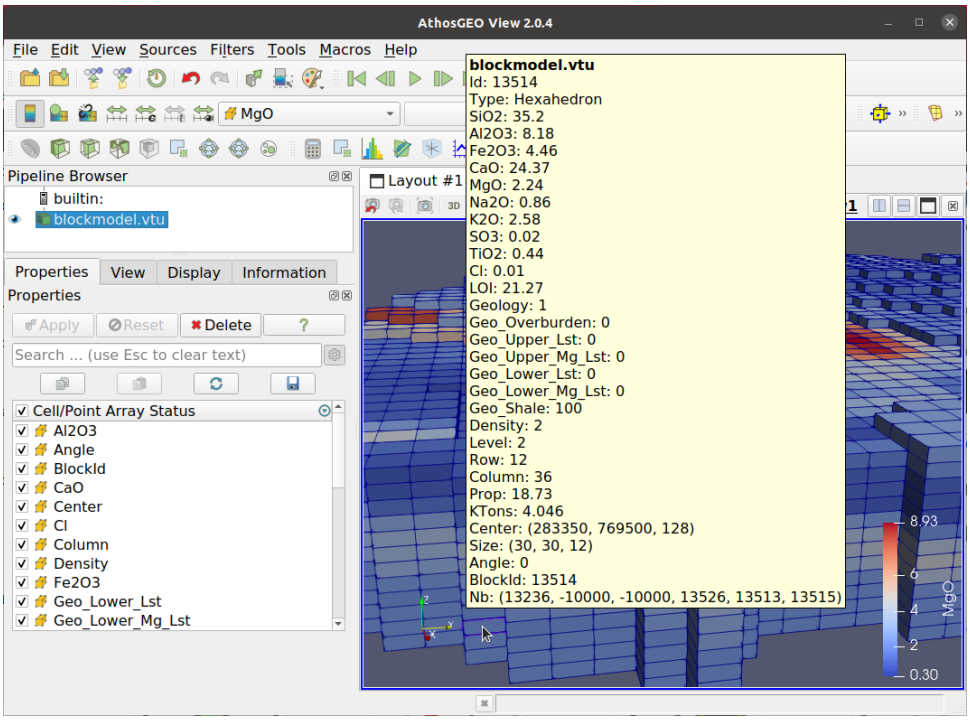


Fig. 2.4: With the  button above a **Render View** turned on, a list of all attributes will always pop up if the mouse remains above a block in the model for a short moment.

Block Attributes II: Probe Location 

The **Probe Location** filter  works also if the number of attributes is large.

On top of solving that problem, it also allows to write the numerical values of a block into a file. Basically, that filter allows to display the attributes of one single block in a **table view** (either **Athos Table View** or **SpreadSheetView**), as shown in Fig. 2.5.

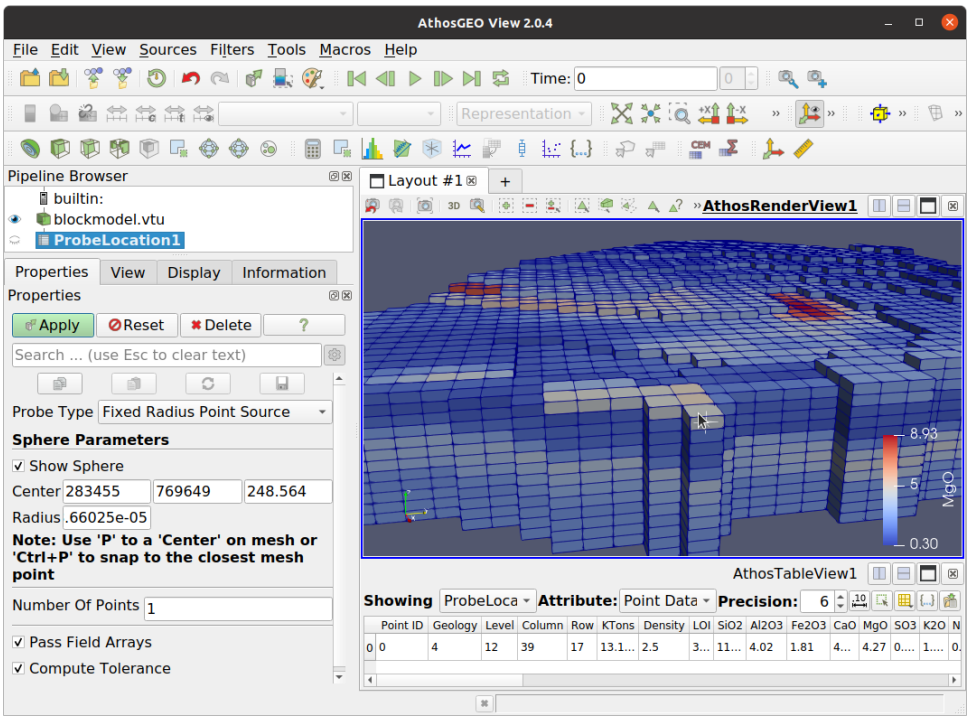


Fig. 2.5: With the **Probe Location** filter, the user moves the mouse to a block of interest and presses the letter *P* to mark a point in the model. Pressing *Apply* will open a **table view** that shows the attributes at that point (which is at the surface of a model block), from where it can be further processed, like written to a **CSV file**. In order to change the point of interest, move the mouse to another block, and press again *P* and *Apply*.

Block Attributes III: Selecting Blocks

A third way to show the attributes of a model block in numeric form is by making use of the fact that **selections** are shared between different views. For this purpose, open the same block model both in an **Athos Render View** and an **Athos Table View**. In the latter, activate the **Show only selected elements** option with the  button, and in the first, activate the **Interactive Select Cells On** option with the  button. Now you can click on cells to select or unselect them, while the attribute data of all selected cells will automatically be displayed in the table view, as shown in Fig. 2.6.

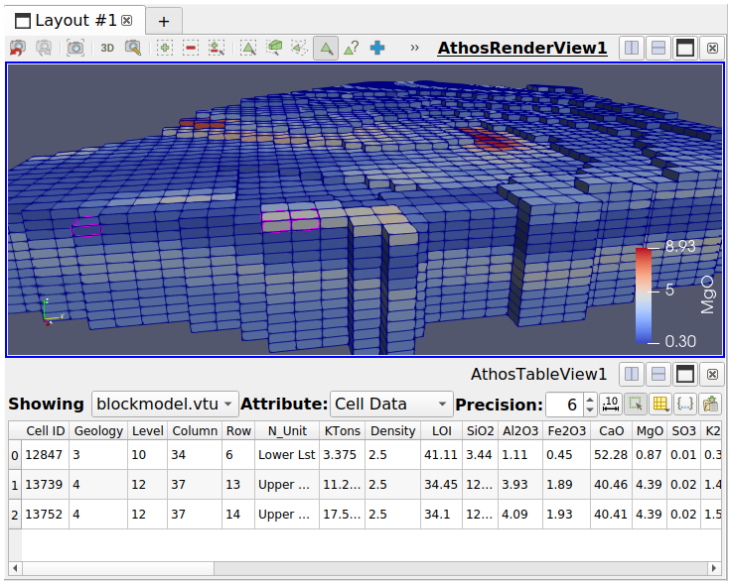


Fig. 2.6: In this example, the **Athos Table View** always shows the attribute data of those blocks that are **selected** in the **Athos Render View**.

Histogram

This filter will generate more or less the same output as the **Histogram View**, as explained in *Histograms*. The only difference is that in a first step the **Histogram** filter generates a data table with two columns (*bin_extents* and *bin_values*), which then is displayed in a **Bar Chart View**.

Plot over Line

The **Plot over Line** filter generates a plot that shows the variation of selected attributes along a line through the model, as shown in Fig. 2.7.

Scatter Plot

Selecting the **Scatter plot** filter and pressing *Apply* will generate a **Line Chart View** as shown in Fig. 2.8, which is probably not yet what we really want to see.

A scatter plot of one chemical compound vs. one or several others can be generated by adapting the properties in the **Display** tab of the **Properties** panel as follows:

- Uncheck the **Use Index for XAxis** option and select a **chemical compound** as the **X Array Name**.
- In the **Series Parameters** list, select the one or several compounds to be shown in the *Y axis*.
- At the very bottom of the panel, change the **Line Style** to *None* (because we do not want to connect the points with lines) and select a **Marker Style** and **Marker Size**, for each one of the selected compounds from the list.

The result would be a scatter plot like in Fig. 2.9.

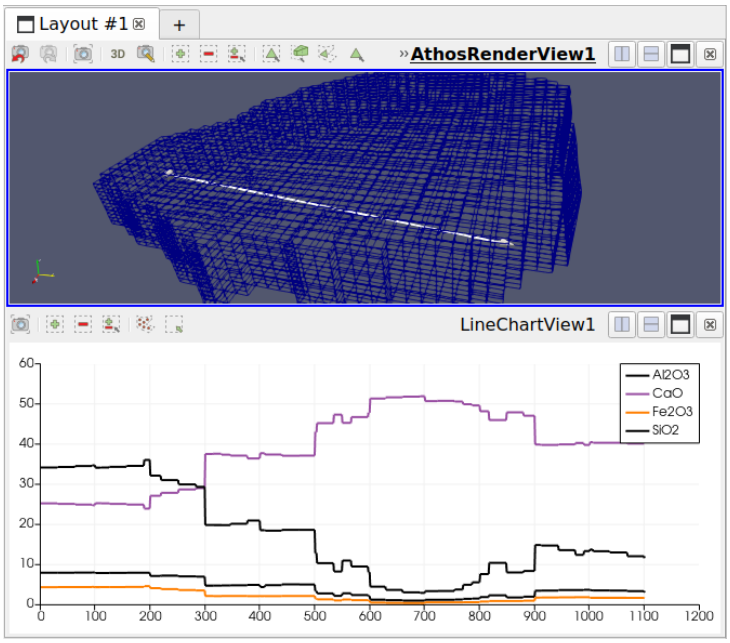


Fig. 2.7: For this **Plot over Line**, the **line** can be specified interactively or with coordinates in the **Properties** panel of the filter. The selection of the **attributes** is done in the **Display** panel of the **Line Chart View**.

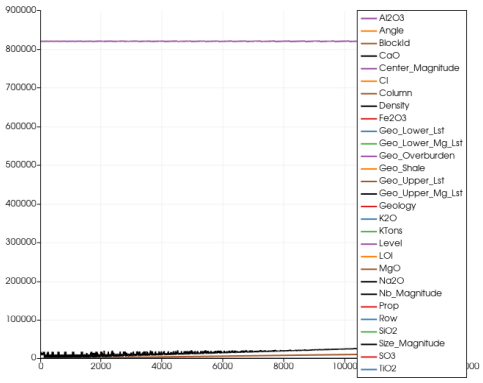


Fig. 2.8: In this initial **Line Chart View**, all attributes are plotted against the *BlockID* attribute of the block model - normally not very useful for a block model.

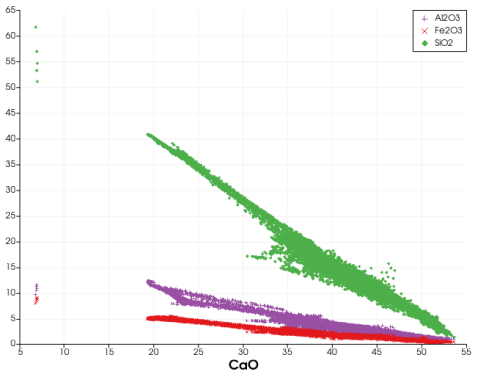


Fig. 2.9: A **scatter plot** showing **Al2O3**, **Fe2O3** and **SiO2** vs. **CaO**

Box Plot

Generating **box plots** is straight forward with the filter **Compute Quartiles**

The output will be as shown in Fig. 2.10.

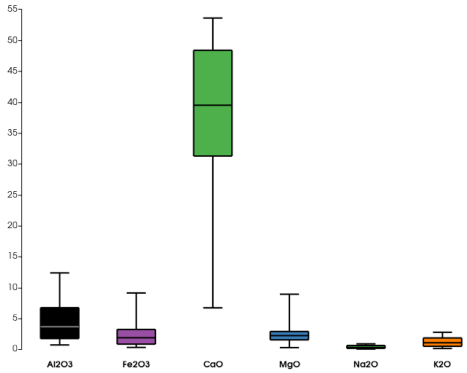


Fig. 2.10: The **properties** of this **Box Chart View** can be adapted in the **Display** tab of the properties panel.

2.2 Display Sampling Set

Technically, sampling sets (either single samples or drilling campaigns) are displayed as **VTK Unstructured Grid**, which means that what was explained about **block models** in *Display a Block Model* mostly applies to **sampling sets** as well.

This also refers to the possibilities to manually manipulate attributes with the **Write Value**, the **Write Multiple Values** and the **Calculator** filters (see *Manipulating Block Model Attributes*).

2.2.1 Single Samples in 3D

A series of single samples can be loaded from a CSV file containing a **sample lable** column, **coordinate** columns and **attributes**, as shown in Table 2.1.

Table 2.1: An example of a table with single samples

SampleId	SiO2	Al2O3	Fe2O3	CaO	N_Unit	Collar:X	Collar:Y	Collar:Z
samp1	12.06	2.55	1.76	42.85	Upper Lst	282630	770160	290
samp2	15.1	3.35	2.08	34.44	Upper Mg Lst	282960	769920	242
samp3	11.46	3.05	1.66	45.97	Upper Lst	282480	769860	290
samp4	14.04	3.74	1.84	40.67	Upper Mg Lst	282780	769680	266
samp5	5.21	1.23	0.55	50.91	Lower Lst	283050	769290	242

Loading this table with the **SamplingReader** will generate a display as shown in Fig. 2.11.

Properties for the display of the single samples can be specified in the **Properties Panel** as shown in Fig. 2.12.

The **Pipeline Browser** shows that the *Sampling Reader* actually generates two output data objects, which are the **Samplings** themselves and the **Sampling Labels**. This allows to show and hide the two independently.

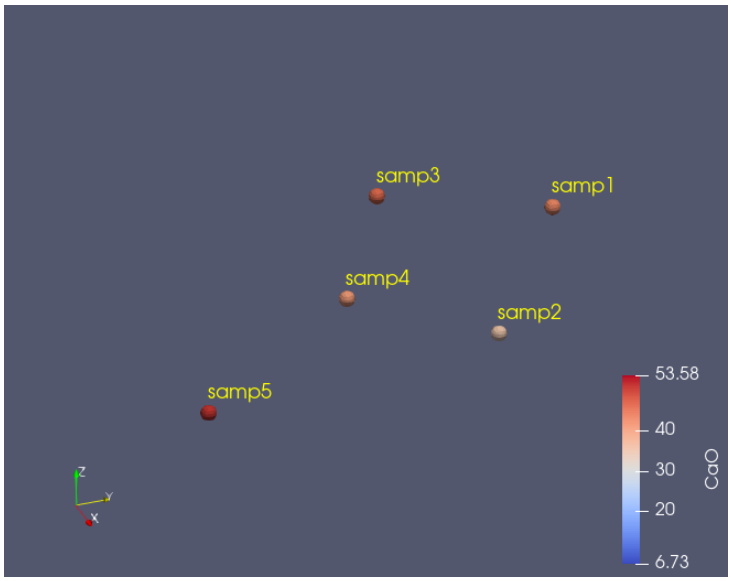


Fig. 2.11: Single **samples** are shown with a **3D Marker** (spheres in this case) and with their **sample label**

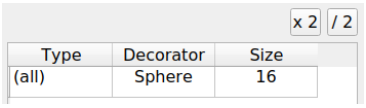


Fig. 2.12: The **Properties Panel** allows to specify the type and size of the **3D Marker** (sphere, cube, tetrahedron or octahedron) and it's size.

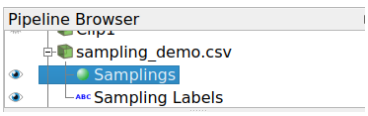


Fig. 2.13: The **Pipeline Browser** for a set of samples shows two objects that can be independently shown or hidden: **Samplings** and **Sampling Labels**.

2.2.2 Drilling Campaign in 3D

Drilling campaigns can be read from CSV files as explained in Section 1.5. The main difference to the display of sets of single samples as explained in Section 2.2.1 are the following:

Samplings

Display options are *round cylinder*, *quadratic cylinder*, *triangular cylinder* and *hexagonal cylinder*.

Sampling Labels

These represent the **HoleId** attribute (and not the **SampleId**).

Did you know?

With a **HoleType** attribute it is possible, to define different display **decorators** for different classes of drillholes, like different **drilling campaigns**, see Fig. 2.14. This actually works also for **sets of single samples**, with the same attribute name.

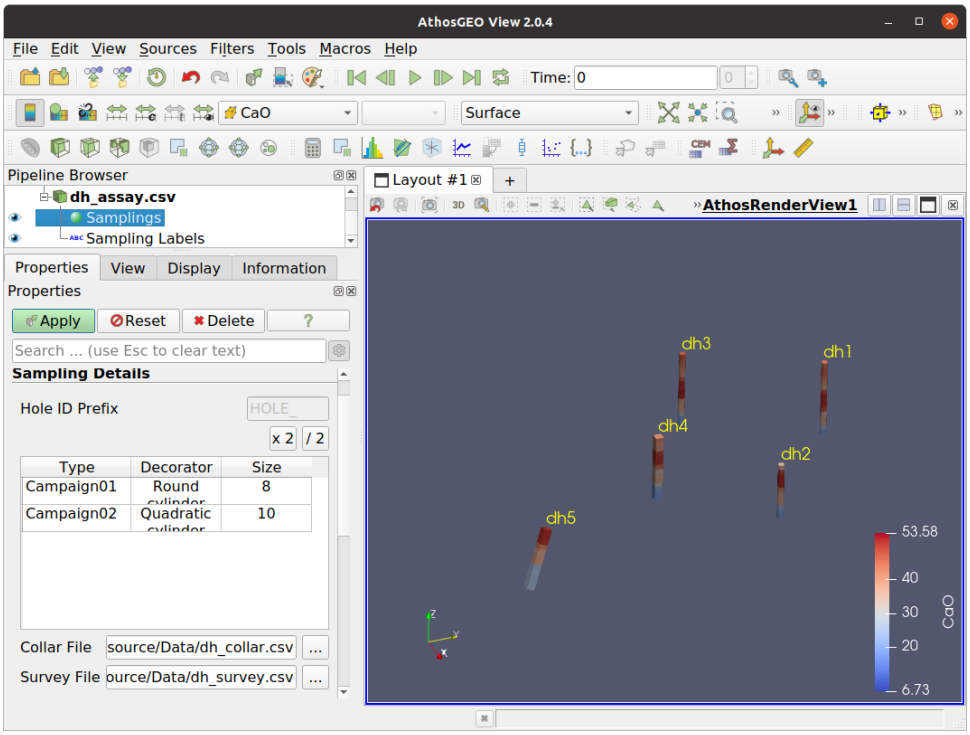


Fig. 2.14: Display example of a simple **drilling campaign**.

2.2.3 Sampling Set as Points

The **Cell Centers** filter allows to also transform either single samples or the samples that make up drillholes into center points, for further processing like generating charts etc.

2.2.4 Sampling Set Charts

In Fig. 2.15, a series of drillholes was first converted into center points with the **Cell Centers** filter and then shown using the **Box Chart** (or *Compute Quartiles*) filter in a **Box Chart View**.

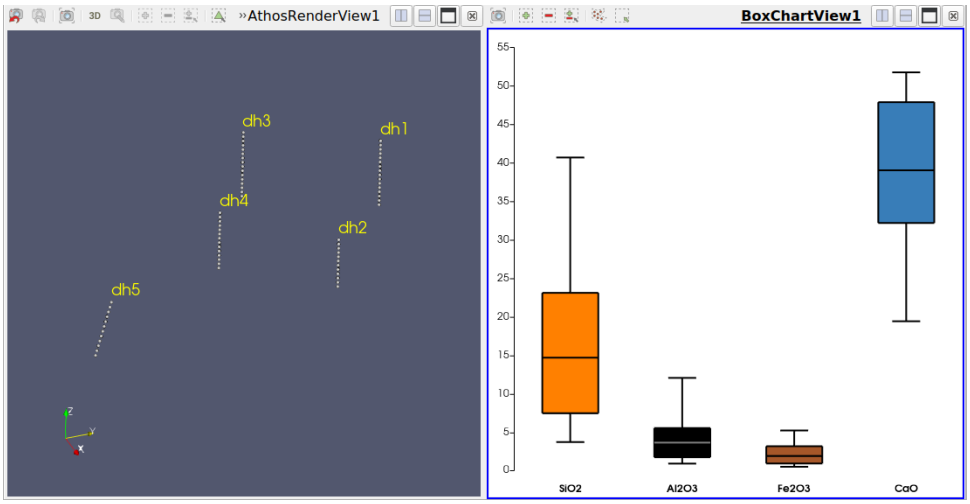


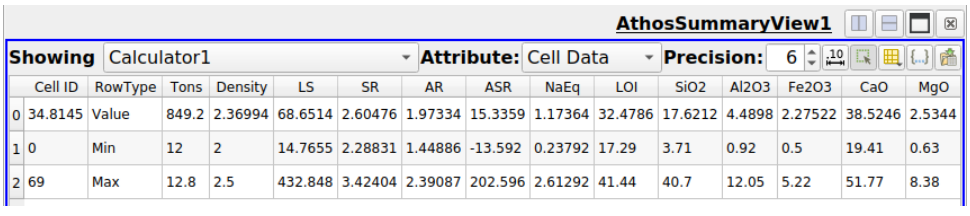
Fig. 2.15: In the **left** view, a set of drillholes was converted into center points, and in the **right** view, four chemical compounds are shown in a **box chart**. Note that all samples within the drillholes are having the same weight here, no matter what length of the drillhole they do represent.

2.2.5 Summary Table of a Sampling Set

Normally, a sampling set cannot be summarized with a **Summarize Attributes** filter or with an **Athos Summary Table**. The reason is the fact that they do not come with a **Tonnage** attribute that can be used to calculate the *tonnage weighted means* of chemical compounds and other attributes.

It might be reasonable to use the *length* of samples for the weighing, or if these are all the same, it could be simply the *number of samples*, and this can be achieved with a workaround as follows:

- Generate a **Tons** attribute by using the **Calculator** filter and simply calculating *Depth_To-Depth_From*.
- Fig. 2.16 shows the summary of an entire drilling campaign, weighted by drill hole lengths.
- Fig. 2.17 shows the same thing by geological units. The formula for calculating the **sample lengths** was changed to *(Depth_To-Depth_From)/5*, meaning that it is divided by the number of drillholes. The effect is that the **Tons** attribute would now represent the **average unit thickness** (except if a unit was not completely traversed by any one of the drillholes).



	Cell ID	RowType	Tons	Density	LS	SR	AR	ASR	NaEq	LOI	SiO2	Al2O3	Fe2O3	CaO	MgO
0	34.8145	Value	849.2	2.36994	68.6514	2.60476	1.97334	15.3359	1.17364	32.4786	17.6212	4.4898	2.27522	38.5246	2.5344
1	0	Min	12	2	14.7655	2.28831	1.44886	-13.592	0.23792	17.29	3.71	0.92	0.5	19.41	0.63
2	69	Max	12.8	2.5	432.848	3.42404	2.39087	202.596	2.61292	41.44	40.7	12.05	5.22	51.77	8.38

Fig. 2.16: Summarized quality data of a full drilling campaign. The **Tons** attribute is actually **length in meters**

AthosSummaryView1

Showing: Calculator1 Attribute: Cell Data Precision: 6

	Cell ID	RowType	N_Unit	Tons	Density	LS	SR	AR	ASR	NaEq	LOI	SiO2	Al2O3	Fe2O3	CaO
0	34.1246	Value	Lower Lst	48.46	2.5	304.591	2.60847	1.97468	2.97483	0.362039	40.4989	5.18114	1.31855	0.667726	50.2487
1	5	Min	Lower Lst	2.4	2.5	171.802	2.38889	1.77064	-10.1358	0.23792	38.52	3.71	0.92	0.5	46.43
2	69	Max	Lower Lst	2.56	2.5	432.848	2.92697	2.32	50.9726	0.58138	41.44	8.47	2.16	1.17	51.77
3	36.3636	Value	Lower Mg Lst	48.62	2.48042	62.9513	2.65464	1.94372	11.3837	1.23685	32.0319	18.4773	4.59591	2.3645	36.9503
4	2	Min	Lower Mg Lst	2.4	2.3	31.8171	2.41688	1.6982	-13.592	0.68034	24.96	10.17	2.72	1.37	29.53
5	66	Max	Lower Mg Lst	2.56	2.5	135.283	3.42404	2.30919	202.596	1.77388	37.23	28.82	8.29	3.59	44.07
6	36.7579	Value	Shale	36.76	2.01108	18.7533	2.55408	2.05101	139.899	2.49426	19.6847	37.3567	9.83234	4.79391	22.3759
7	0	Min	Shale	2.4	2	14.7655	2.38151	1.63675	40.3299	2.09076	17.29	33.21	7.66	4.29	19.41
8	62	Max	Shale	2.56	2.16	23.807	2.906	2.39087	163.136	2.61292	22.29	40.7	12.05	5.22	25.4
9	28	Value	Upper Lst	14.4	2.5	124.2	2.60744	1.78717	-0.294528	0.667223	35.9833	11.325	2.785	1.55833	44.7233
10	13	Min	Upper Lst	2.4	2.5	106.961	2.28831	1.44886	-7.83717	0.58822	34.64	9.78	2.21	1.32	42.85
11	43	Max	Upper Lst	2.4	2.5	148.429	2.79814	1.98795	9.53092	0.82588	37.48	12.81	3.3	1.76	45.97
12	34.1111	Value	Upper Mg Lst	21.6	2.5	86.844	2.691	1.82136	4.38497	0.942276	34.9278	14.2144	3.41	1.87222	39.1156
13	10	Min	Upper Mg Lst	2.4	2.5	72.3757	2.51613	1.54639	-9.54512	0.8222	34.39	13.39	3	1.74	34.44
14	57	Max	Upper Mg Lst	2.4	2.5	99.28	2.84211	2.05028	11.6173	1.02986	35.38	15.1	3.86	2.08	42

Fig. 2.17: Summarized quality data of drilling campaign by geological units. Here the **Tons** attribute represents **average unit thickness** of the respective geological unit (except for units that are not fully traversed by one or several of the drillholes).

2.3 Display Table (and other) Data

The **AthosGEO View** program inherits abilities to display and analyze tabular data from the **ParaView** software. Tabular data can be read directly from *CSV Files*, but also the attributes of geometric objects are basically tabular data and the following explanations apply to them as well.

Did you know?

For geodata processing with **AthosGEO View** the most important geometric objects are one of the two VTK data types:

- **Unstructured Grid** (block models, sampling sets) and
- **Poly Data** (surfaces and polylines, both closed or open, as well as point sets).

These data objects are made up of two elements that are internally kept separate (which makes data manipulation much more efficient):

- **Points** with their coordinates
- **Cells** that are everything else: *triangles* in the case of a triangulated surface, *line segments* in the case of polylines, *hexahedra* in the case of block models or many other 3D elements in the case of sampling sets. These *cells* have no coordinates, but refer via internal ID numbers to the *points*.

On top of that, all these data elements can be connected with attribute data, which are basically all data tables:

- **Point Data** are attached to the points. The point data table has the same number of rows as the point data set has points, so one data row refers to exactly one point. The number of columns is free, and the data types of the columns can be all different, numeric or strings.
- **Cell Data** are attached to the cells, and each cell is again related to one row in the cell data table. Block model attributes are actually cell data.
- **Field Data** are data tables that are attached to the entire data object, without a relation to single points or cells.

All the above tabular data are included in the following explanations about display and management of table data. One additional data table type is

- **Row Data** which are data tables that are not related to any geometric objects, possily read directly from a *CSV File*.

Note: This explanation of VTK data object types, points, cells etc. is by far not exhaustive. For more detailled information see: VTK File Formats.

2.3.1 Table Views

The most simple display for table data is one of the two available *table views* (see Fig. 2.18).

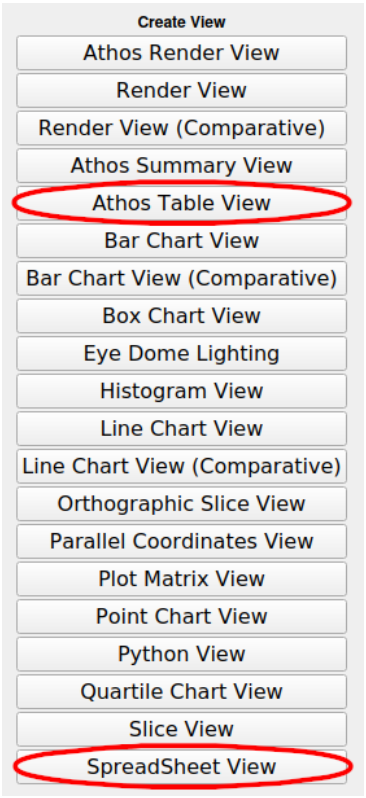


Fig. 2.18: Both the *Athos Table View* and the *SpreadSheet View* can display tabular data in a table format. The first is derived from the latter, and the difference is only the ordering of the columns.

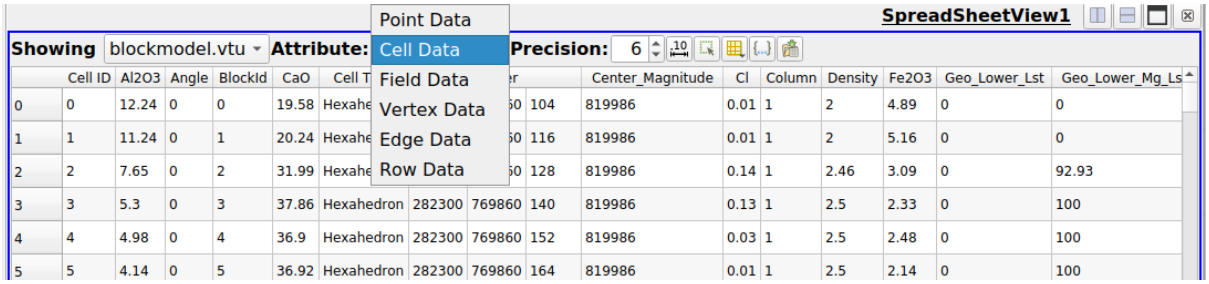
These are the available table views in **AthosGEO View**:

SpreadSheet View

This table view does not really offer the functionality of a spreadsheet, but shows the content of a data table in tabular form, including *numeric* and *string* data. By default, the *Point Data* of a geometric data object are shown, so for a block model or a sampling set, the display has to be manually switched to *Cell Data*: see Fig. 2.19. A plain table like it can be read from a *CSV File* is shown as *Row Data*.

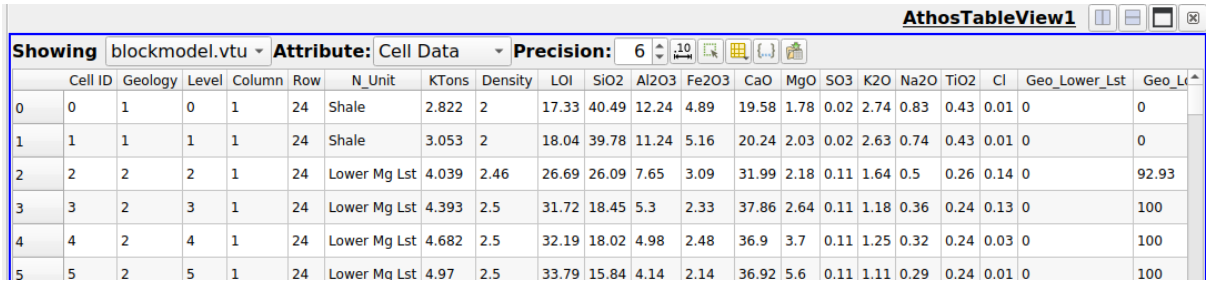
Athos Table View

This table view is derived from the *SpreadSheet View* and only the ordering of the columns is different: While the first orders the columns alphabetically, this view is tailor made for the needs of displaying *block model* and *sampling set* data in a more clearly arranged way for that purpose, as can be seen in Fig. 2.20. One more detail: The default display on opening an *Athos Table View* is *Cell Data* because only with this the user will see the data content of a block model or sampling set directly.



	Cell ID	Al2O3	Angle	BlockId	CaO	Cell T				Center_Magnitude	Cl	Column	Density	Fe2O3	Geo_Lower_Lst	Geo_Lower_Mg_Ls
0	0	12.24	0	0	19.58	Hexahe			104	819986	0.01	1	2	4.89	0	0
1	1	11.24	0	1	20.24	Hexahe			116	819986	0.01	1	2	5.16	0	0
2	2	7.65	0	2	31.99	Hexahe			128	819986	0.14	1	2.46	3.09	0	92.93
3	3	5.3	0	3	37.86	Hexahedron	282300	769860	140	819986	0.13	1	2.5	2.33	0	100
4	4	4.98	0	4	36.9	Hexahedron	282300	769860	152	819986	0.03	1	2.5	2.48	0	100
5	5	4.14	0	5	36.92	Hexahedron	282300	769860	164	819986	0.01	1	2.5	2.14	0	100

Fig. 2.19: The **SpreadSheet View** is the table view of **ParaView**. Please note the little buttons above the header line of the table: They allow to specify the number of displayed digits, switch between scientific and fixed point notation, show or hide table columns and either all rows or selected rows only and finally write the table data to a *CSV file*.



	Cell ID	Geology	Level	Column	Row	N_Unit	KTons	Density	LOI	SiO2	Al2O3	Fe2O3	CaO	MgO	SO3	K2O	Na2O	TiO2	Cl	Geo_Lower_Lst	Geo_Lc
0	0	1	0	1	24	Shale	2.822	2	17.33	40.49	12.24	4.89	19.58	1.78	0.02	2.74	0.83	0.43	0.01	0	0
1	1	1	1	1	24	Shale	3.053	2	18.04	39.78	11.24	5.16	20.24	2.03	0.02	2.63	0.74	0.43	0.01	0	0
2	2	2	2	1	24	Lower Mg Lst	4.039	2.46	26.69	26.09	7.65	3.09	31.99	2.18	0.11	1.64	0.5	0.26	0.14	0	92.93
3	3	2	3	1	24	Lower Mg Lst	4.393	2.5	31.72	18.45	5.3	2.33	37.86	2.64	0.11	1.18	0.36	0.24	0.13	0	100
4	4	2	4	1	24	Lower Mg Lst	4.682	2.5	32.19	18.02	4.98	2.48	36.9	3.7	0.11	1.25	0.32	0.24	0.03	0	100
5	5	2	5	1	24	Lower Mg Lst	4.97	2.5	33.79	15.84	4.14	2.14	36.92	5.6	0.11	1.11	0.29	0.24	0.01	0	100

Fig. 2.20: The **Athos Table View** shows first the *category* columns, followed by *tonnages* and *chemical compounds* and others. All the other functionality is inherited from the *SpreadSheet View*.

CSV Reader for Athos Table View

Opening a CSV file will first open a reader selection dialog (see Fig. 2.21).

Both the **CSV Reader** and the **CSV Reader for Athos Table View** will interpret CSV files as simple tables and display them accordingly:

- The **CSV Reader** will display the file in a **SpreadSheet View**
- The **CSV Reader for Athos Table View** will display the same file in an **Athos Table View**

Table Tool Buttons

At the upper right of an Athos Table View there is a little toolbar (see Fig. 2.22), with the following functionality:

1. **Precision**: enter the number of valid digits in the numeric output
2. **Representation**: scientific or fixed point
3. Show either all or only **selected** elements
4. Choose the **columns** to display in the table
5. **Cell connectivity** is not being used for **AthosGEO View**
6. The **Export Spreadsheet** function allows to write the currently shown table content to a CSV file. The difference to the **Save Data** function is the fact that it maintains column order and other display settings, while the latter always writes the entire table to a file.

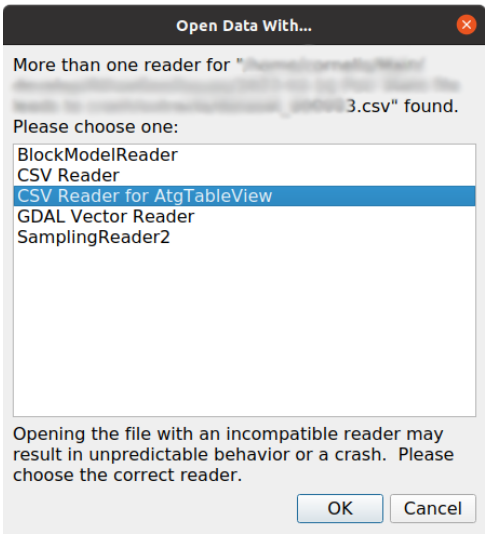


Fig. 2.21: Choice of readers for **CSV files**

Fig. 2.22: The **Athos Table View** tool bar buttons. They are the same as for the **Athos Summary View** (see next section).

2.3.2 Summary Tables

Summarizing quality data from block models means calculating tonnage weighted means of the quality attributes. However, other attributes require a different handling, and this is one of the main reasons for the introduction of the different *attribute categories* in **AthosGEO View** - see *Attribute Conventions*.

The display of a data table with the **Athos Summary Table** will result in an automatically summarized display of the data as shown in Fig. 2.23.

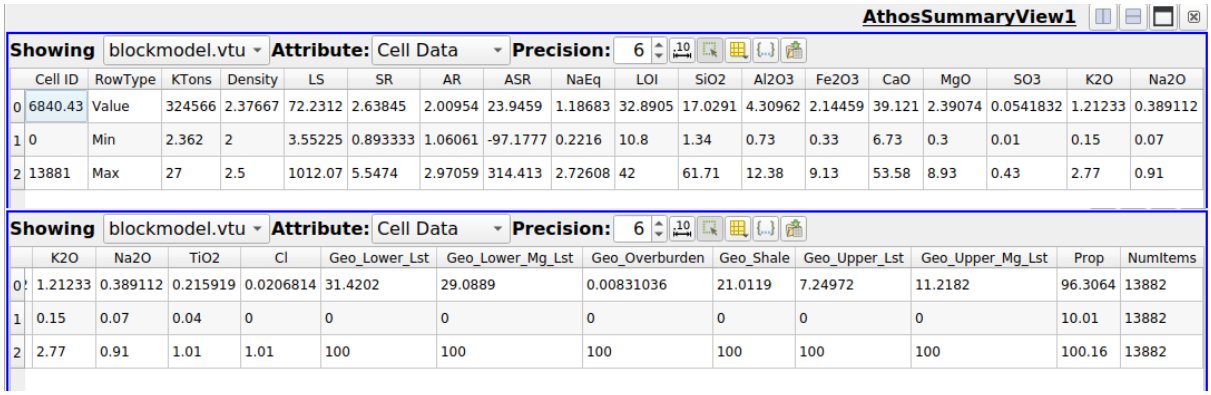


Fig. 2.23: **Athos Summary View** display of the same data as in Fig. 2.20.

The **Athos Summary View** shows 3 rows of data, as indicated in the *Row Type* column:

Value: The calculated summary value for a specific attribute.

Min: The lowest value of a specific attribute within the range of the summary.

Max: The highest value of a specific attribute within the range of the summary.

The applied operations for the different *attribute categories* are summarized in Table 2.2.

Table 2.2: Summarizing *operations* for different *attribute categories*

Attribute Category	Examples	Summarizing Operation
Direct	SiO2, CaO	Tonnage weighted means
Derived	LS, SR, AR	Recalculate from tonnage weighted means
Tonnage	KTons, Taken	Sum
Weight	Pit, Zone	Tonnage weighted means
Category	Code, Level	Skip
SpecTonnage	Density	Volume weighted means
Others	coordinates, block sizes	Skip

A number of *properties* can be adapted for the *Athos Summary View*, and this can be done in the **Display** tab as shown in Fig. 2.24.

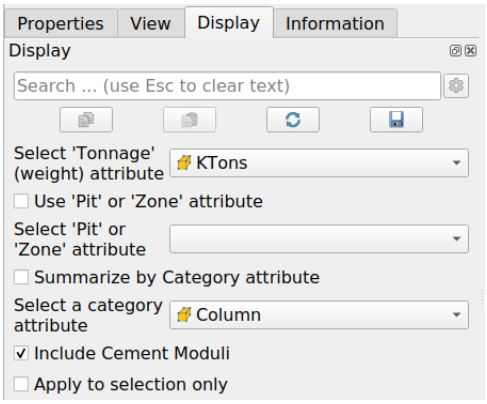


Fig. 2.24: Properties of the **Athos Summary View** can be adapted in the **Display** panel.

The following properties can be changed:

Tonnage Attribute

Every block of a block model must have at least one *Tonnage* type attribute, either *Tons* or *KTons*. But it is possible to have additional tonnages like *Taken*, which is the tonnage that was taken for a specific planning and which can be less than the total tonnage if only part of the block is required for the planning. If now the total taken tonnage is needed, *Taken* should be selected here.

Pit or Zone Attribute

In order to ignore all the tonnages outside of a specific pit or zone, it can be selected here. Do not forget to check the check box to make it happen.

Summarize by Category Attribute

Normally the summary table has exactly the 3 rows as explained in Fig. 2.23. However, if *Summarize by Category attribute* is on, for each existing value of the selected *Category* such a row triple will be calculated.

Include Cement Moduli

No matter if the input table had *cement moduli* or not, the output will have them calculated if this option is checked.

Apply to Selection Only

With this it is possible to select a number of blocks and automatically calculate the summary for these blocks only. If blocks are unselected or blocks added to the selection, the *Athos Summary View* will be updated on the fly.

Did you know?

Besides the **Athos Summary View**, summarizing of tabular data can also be done with the **Summarize Attributes** filter. Two things to remember:

- *Properties* of the summarizing operation are not adapted in the **Display** tab, but in the **Properties** tab.
- The output of the *Summarize Attributes* filter must be displayed with the **Athos Table View**, not with the **Athos Summary View**, because the latter would try to summarize once more the result of the summarizing.

2.3.3 Charts

ParaView has the ability to generate different types of **charts** from data tables and data sets, and **AthosGEO View** inherits them. In this section only a few examples for such charts are explained: Please have a look into the **ParaView** documentation for more - see *First Steps*.

Histograms

Histograms can be generated by simply showing the table data in a **Histogram View** as illustrated in Fig. 2.25.

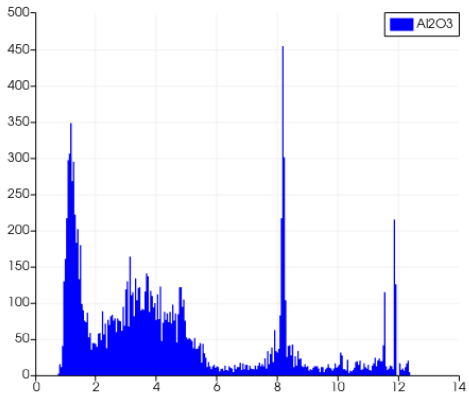


Fig. 2.25: A **Histogram** showing the *Al2O3* attribute of a data table.

The **Display** panel allows to finetune the *properties* of the *Histogram View* as shown in Fig. 2.26.

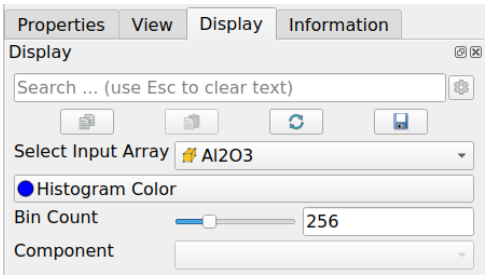


Fig. 2.26: In the **Display** panel of the *Histogram View*, an attribute for the display can be selected and other display settings.

On top of that, additional properties regarding the **X and Y axis** of the *Histogram View* can be adapted in the **View** panel as shown in Fig. 2.27.

Hint

The **View Properties** of a *Histogram View* are initially set in such a way that all the data are displayed, and they can be adapted in the *View Panel* as shown in Fig. 2.27. On top of that, the numeric display range can also be directly adapted with the mouse in the *Histogram View* display.

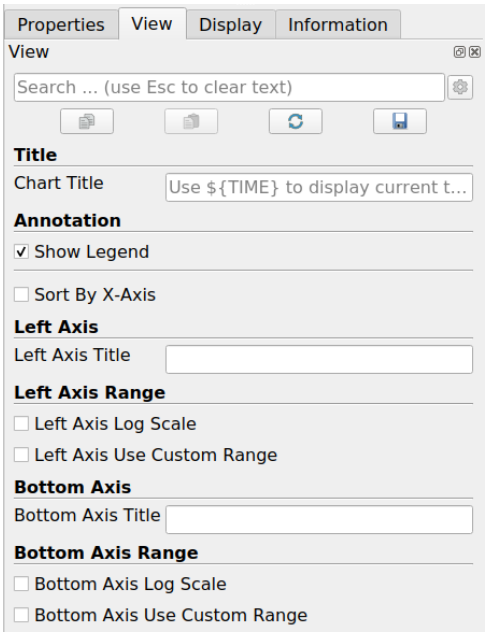


Fig. 2.27: In the **View** panel, properties regarding the axes and chart titles can ge adapted.

Such a histogram of the distribution of chemical compounds in a rock volume will give valuable insights about homogeneity or heterogeneity and other characteristics of of the materials.

Box Charts

Box Charts are a useful way for the visualization of more than one attribute - see Fig. 2.28.

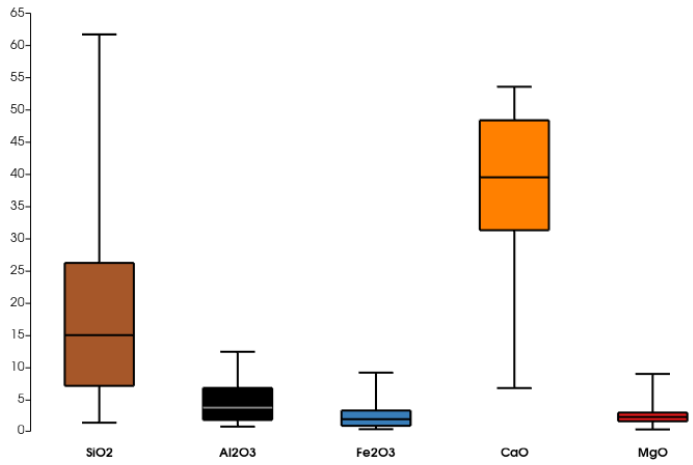


Fig. 2.28: Box chart example.

Generate a box chart by applying the **Compute Quartiles** filter to a **block model**, **sampling set**, **drillhole data** or all kinds of **data tables**.

Matrix of Correlation Charts

An even more powerful way to analyze the composition of rocks within a rock volume that is represented by a block model is the matrix of charts that can be displayed with the **Plot Matrix View** as shown in Fig. 2.29.

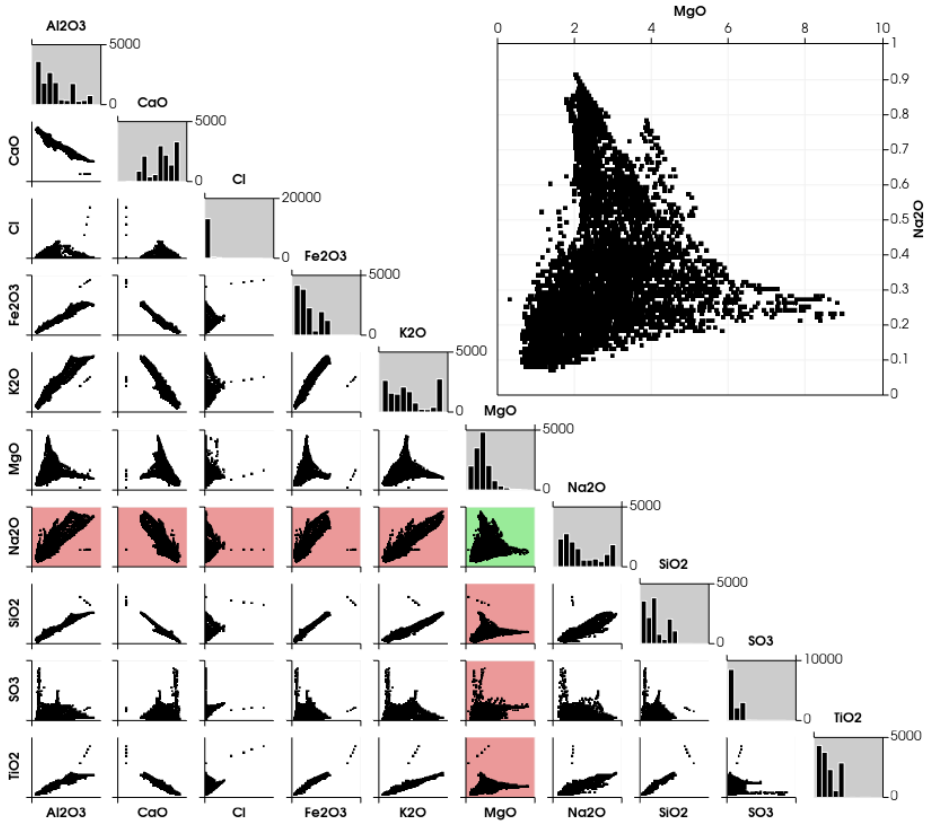


Fig. 2.29: The **Plot Matrix View** shows a matrix of charts

The following charts are shown in a **Plot Matrix View**:

- a diagonal of *histograms* for all the selected chemical compounds.
- in the lower left triangle, correlation diagrams for all chemical compounds with all others.
- in the upper left triangle, one enlarged correlation diagram that can be selected from the lower left diagrams with a mouse click.

Select the *chemical compounds* to be included in the chart in the **Display** properties panel as shown in Fig. 2.30.

Did you know?

With the little **camera button** above the upper left corner of the charts it is possible to copy the displayed chart directly into the clipboard for insertion into a report or other document. Pressing that button together with *Ctrl*, *Alt* or *Shift* will open a file dialog that allows to save the chart into an image file.

Did you know?

Chart display can be part of interesting **linked selections**, in two ways:

- automatically updated chart display on changing data items or selections in other views
- selecting items in a chart and automatically visualize them in a 3D (Render) or Table view

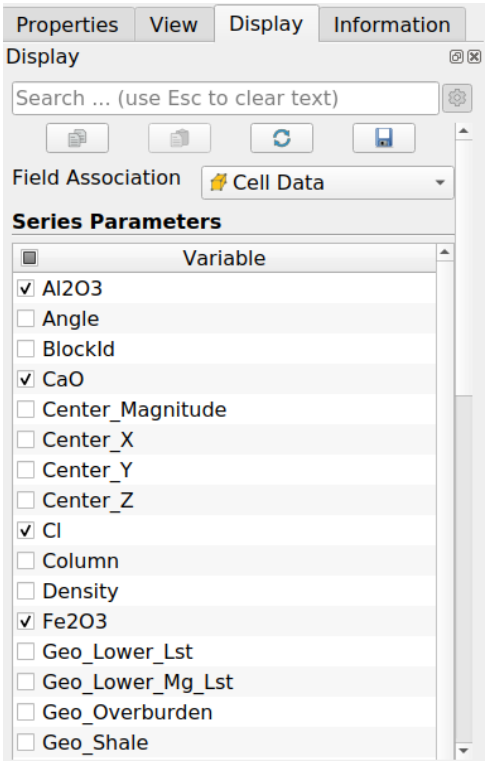


Fig. 2.30: In the **Display** properties panel, choose the compounds that should be included in the plots

For more about selections see *Selections*.

2.3.4 Conclusion

The above examples for data display and analysis with **AthosGEO View** do not cover all the possibilities that are available thanks to the fact that the software is built on top of **ParaView**. For more information, follow the links that can be found here in *First Steps*.

2.4 Display a Topo

Most of what a user needs to know about display of a topo is already explained in Section 1.7:

- **File types** (**.vtp*, **.dxf* and *GeoTIFF* files) and **VTK data types** (*polydata* for both triangulated surfaces or polylines/polygons).
- **Manually generating** surfaces and polylines.
- **Simplification** of triangulated surfaces and polylines with the **Decimate** and the **Decimate Polyline** filters.
- **Projection** of a georeferenced bitmap image on a topo (with either *ESRI World Files* or *GeoTIFF*).

This section will show a few more ways how to display and manipulate topography data - basically triangulated surfaces.

2.4.1 Color by Elevation

A plain triangulated surface does not have attributes, so coloring by attribute is not possible, but this can easily be changed as follows:

- Use the **Calculator** filter to generate an **Elevation** *point attribute* The formula would simply be *coordsZ*. This formula basically transfers the *Z* coordinate of the points to a new attribute that can be used for a color mapping display.
- Now it is possible to color the surface by **Elevation**, as shown in Fig. 2.31.

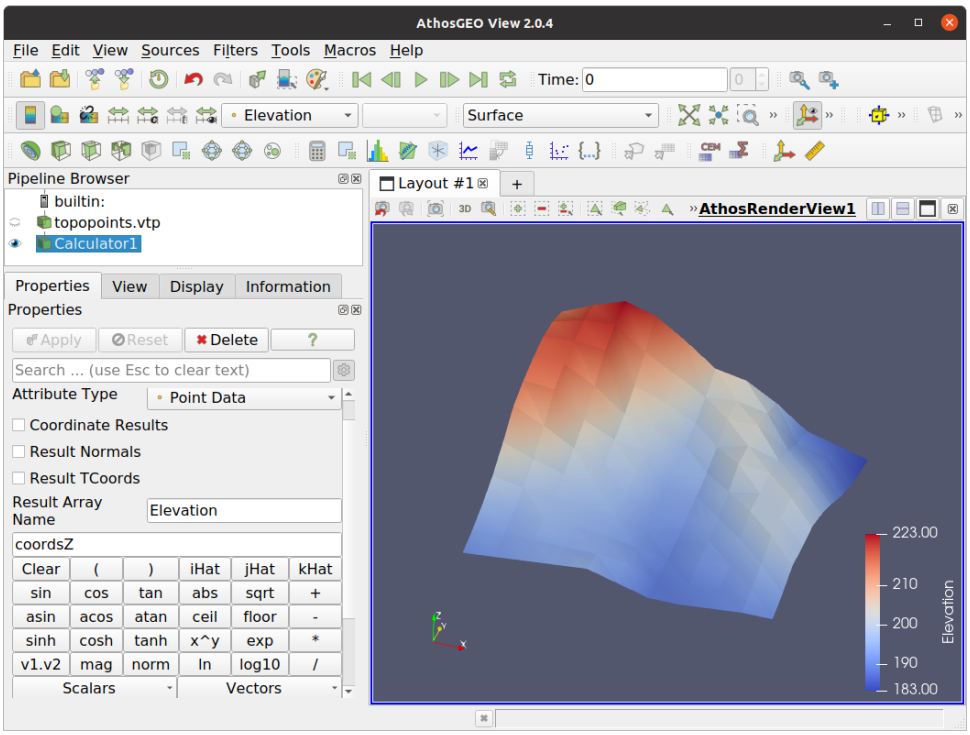


Fig. 2.31: Coloring a **triangulated topo surface** by a newly generated **Elevation** attribute.

Did you know?

As can be seen in Fig. 2.31, coloring appears nicely and smoothly interpolated. This is because the **Elevation** attribute is a **Point** attribute. See also Section 2.4.3 for more information about *point and cell attributes*.

2.4.2 Contour Lines

Once a triangulated surface has an **Elevation** point attribute as explained in Section 2.4.1, it is also possible to calculate **contour lines** for the topo surface with the **Contour** filter.

On the **Properties** panel, the elevations are defined in the **Isosurfaces** section. The result will be as shown in Fig. 2.32.

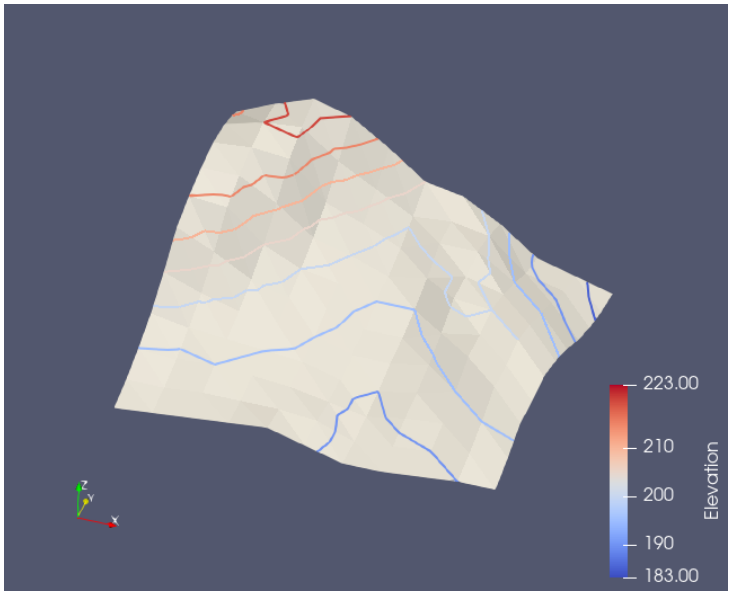


Fig. 2.32: Calculated **contour lines** for a triangulated topo surface. The lines are colored with the **Elevation** attribute, and the line width is increased in the **Display** panel to enhance visibility. The topo surface is shown as well, but could be turned off in the **Pipeline Browser**.

2.4.3 Point and Cell Attributes

While for *block models* and *sampling sets* it is clear that we are interested in *cell attributes* only, it makes a lot of sense for triangulated surfaces to care about both **point** and **cell attributes**, depending on the visual effect that is intended:

- **Point attributes** are interpolated between the points, which results in **smooth color transitions** between the points, as shown in Fig. 2.31.
- **Cell attributes** will result in a **precise coloring** where each triangle receives the color that corresponds to it's cell attribute, with hard edges, as shown in Fig. 2.33.

Two filters are available for *converting between point and cell attributes*:

- **Point Data to Cell Data** does what the name is saying: It converts *point attributes* into *cell attributes*. It does so by averaging all available point attributes of those points that are part of a specific cell. A result of applying this to the **Elevations** attribute that was generated in the previous section looks like shown in Fig. 2.33.
- **Cell Data to Point Data** is doing the inverse operation of the previous filter. Also in this case, attribute values are averaged from adjoining cells.

2.4.4 Direct Coloring

So far and by default, coloring is for an attribute by **color mapping**. However, it is possible to turn color mapping off and use color values that are directly stored as attributes:

- In the following, **Colors** will be used as the attribute name for colors, but actually the name is unimportant: It is only the **Map Scalars** property that counts - see Fig. 2.34.
- If a **Colors** attribute has **one component**, the values will be interpreted as **grey values**. If it has **three components**, the components will be interpreted as *red*, *green* and *blue* components.
- If the **data type** of the **Colors** attribute is *floating point values* (*float* or *double*), the value range is between 0.0 and 1.0. If it is *integer values*, the value range would be 0 to 255.

In Fig. 2.35 you find an example where a full color (RGB) attribute is generated with the **Calculator** filter. However, it is completely irrelevant how that attribute is generated, as long as the conditions given above are followed.

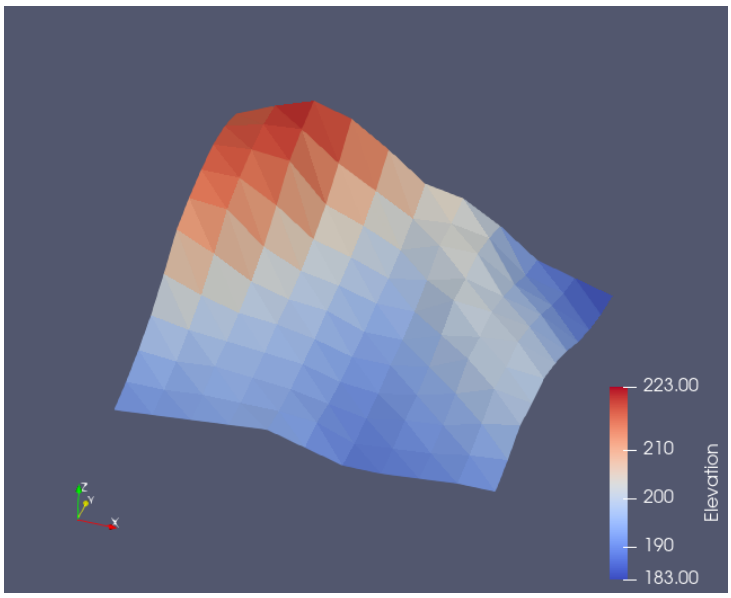


Fig. 2.33: Triangles are colored according to the **average elevation** of their adjoining points.

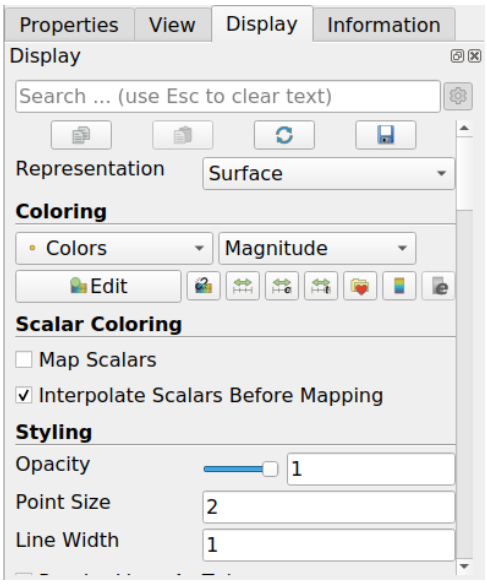


Fig. 2.34: The **Map Scalars** property is responsible for deciding whether attribute values are interpreted as **colors directly** (**off**) or colored via a **color map** (**on**, which is default) that can be manipulated with the **Color Map Editor**. Please note that below this property you find also the **Opacity** property that allows to make a surface **semi-transparent**.

In the example, the following formula is used to generate a **Colors** attribute:

iHat(coordsX-10200)/100+jHat*(coordsY-24300)/100+kHat*

Some explanatory hints about this formula:

- *iHat*, *jHat* and *kHat* are actually the unity vectors *(1,0.0)*, *(0,1,0)* and *(0,0,1)*, so by multiplying with them, we are generating a **3-component attribute**, not a single scalar only.
- From *coordsX* we are subtracting *10200*, which is the minimum X coordinate in the example, and we divide the result by *100*, which is the value range of X. The same is true for the Y coordinate, and Z we ignore in this case.

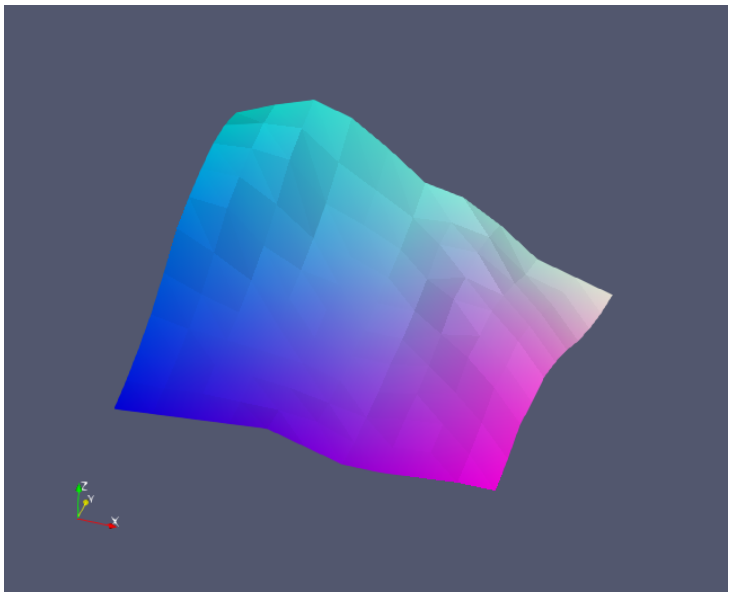


Fig. 2.35: In this example, coloring is the more *red* (or *magenta*) the higher the X coordinate, *green* is increasing with the Y coordinate, and where both are small, *blue* becomes the dominating color.

Another example is shown in Fig. 2.36 that shows an explicit example for greyscale coloring by elevation.

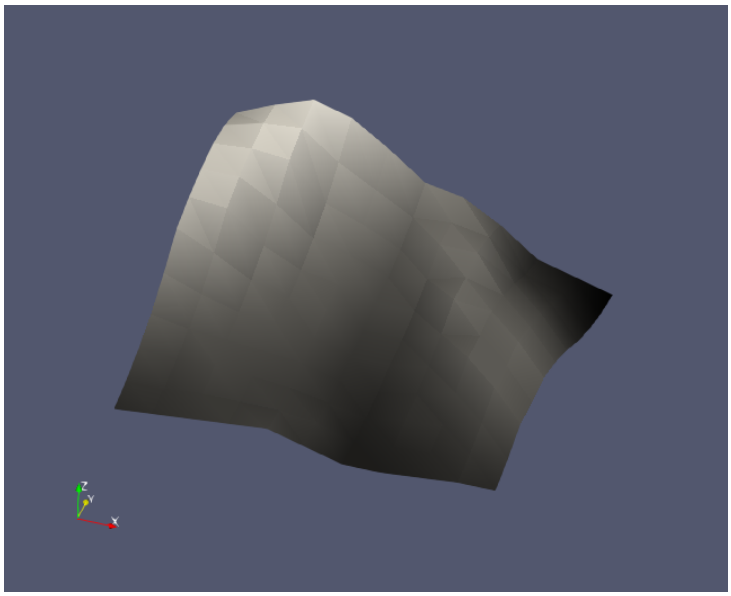


Fig. 2.36: Here greyscale coloring is directly from a **one-component attribute** with values going from 0.0 to 1.0.

2.4.5 Write Value and Write Multiple Values for Surfaces or Polylines

The **Write Value** and the **Write Multiple Values** filters are described in the sections *Write Value* and *Write Multiple Values* , both for a block models, but they can be equally well applied to triangulated surfaces or polylines where it will affect the **Cell attributes** of the triangles or line segments, respectively, as shown in Fig. 2.37, with results as shown in Fig. 2.38.

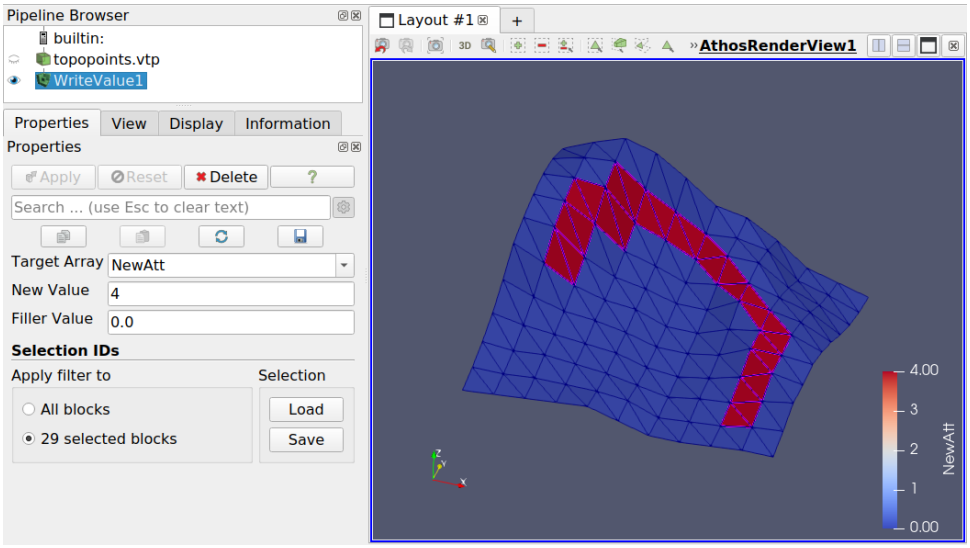


Fig. 2.37: The **Write Value** filter allows to **"paint"** values on a triangulated topo surface

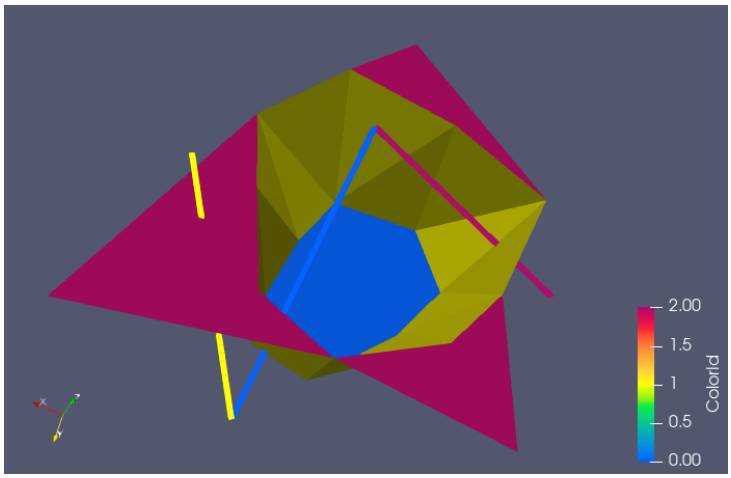


Fig. 2.38: Example with a **triangulated surface** and a **polyline** that are colored by their assigned attributes

---

Note

In the same way as with the **Write Value** filter, also the **Calculator**, the **Calculator for Selection** and the **Write Multi Values** filters can be applied to triangulated topo surfaces: see Section 1.4, where they are all explained for block models.

Hint

If the resolution of the triangle grid is too coarse grained for the assignment of a specific coloring, it can be refined using the **Subdivide** filter.

2.4.6 LAS (Laser Scanner) File Reader

AthosGEO View has a reader for laser scanner (LAS) files. The output is a point cloud which may have point attributes: see Fig. 2.39 for an example.

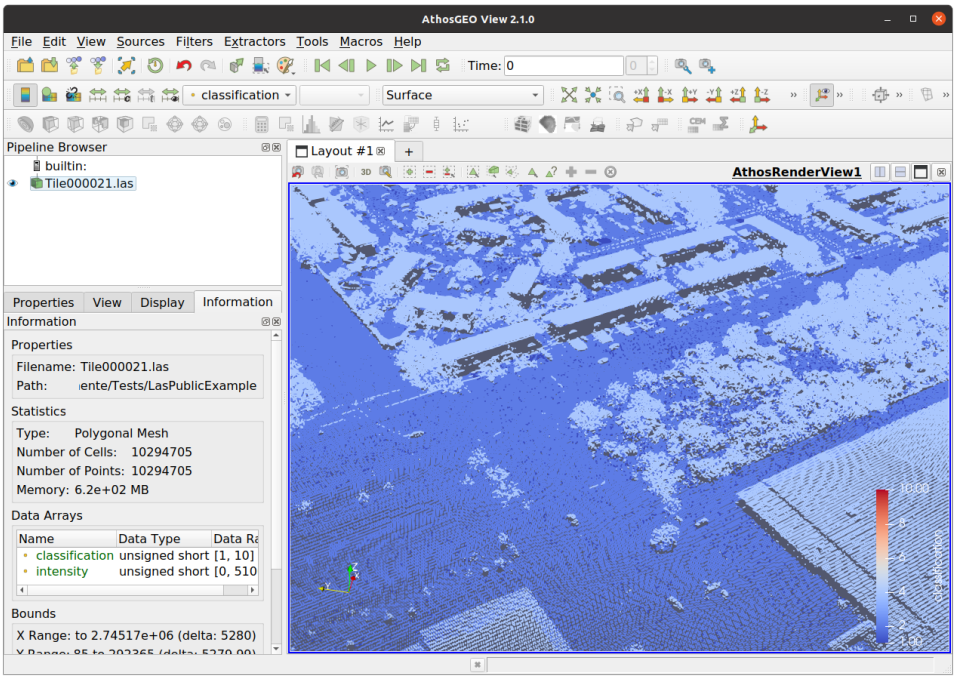


Fig. 2.39: Example of a point cloud from a laser scanner, with a **classification** attribute.

Hint

If the LAS data set has a **colors** attribute, it's values will be adjusted in such a way that RGB color can be directly displayed by turning the **Map Colors** off in the **Display** panel.

Note

LAS files can be very large, and whether **AthosGEO View** will be able to process the data depends on the memory of the computer system. This is not only true for plain reading and display: system overload can also happen if any filters are applied.

2.5 Clipping, Cutting, Merging and Extracting

2.5.1 Functions and Limitations

AthosGEO View is a viewer in the first place. In Section 1.4 it was described how attributes can be manipulated. In this current chapter it will be about geometric manipulations of both **block models** and **surfaces** such as topos.

Three types of manipulations will be described:

- **Extracting** parts of a geometry
- **Merging** geometries
- And the most complex subject: **clipping and cutting** geometries with other geometries

2.5.2 Extracting Parts of a Block Model or a Triangulated Surface

Extracting parts of a geometry is done in two steps:

- **First**, select the parts to be extracted. Selections are a science in it's own, so they are handled in an own chapter here: Section 2.6.
- **Second**, use the **Extract Selection**  filter

2.5.3 Merging Geometries

There are two filters that allow the merging of geometric objects, that are applicable depending on the input data type. In order to apply one of these two filters, we need to select the geometric objects to be merged in the **Pipeline Browser**, as shown in Fig. 2.40.

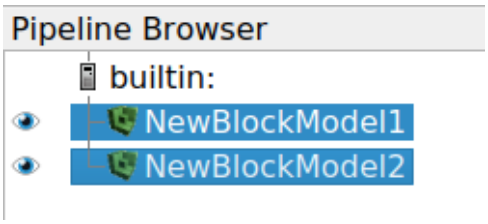


Fig. 2.40: More than one item can be selected in the pipeline browser by first clicking on the first, then hold down the *CTRL* key while clicking on additional items. Note that it is possible to select two or more items that way

Once a number of items are selected, the following two filters can be activated from the *Filters* → *Alphabetical* menu, depending on the VTK data or geometry types of the input data items:

- **Append Geometry**
- **Append Dataset**

Note

The *Information* panel is providing information about the **VTK data or geometry type** of a selected pipeline browser item.

Append Geometry

If all input items are of type **Polygonal Mesh**, it is possible to generate one merged polygonal mesh with the **Append Geometry** filter: see Fig. 2.41.

Append Dataset

If one or several of the input pipeline browser items are of type **Unstructured Grid**, then the **Append Dataset** filter is able to merge these items into one single unstructured grid: see Fig. 2.42.

Hint

To some degree, **Unstructured Grid** items can be "converted" into **Polygonal Mesh** items with the **Extract Surface** filter, but the result will not be a 1:1 equivalent. If the input unstructured grid is e.g. a block model, the output polygonal mesh will be a surface that represents the hull of the block model, without the volume filling.

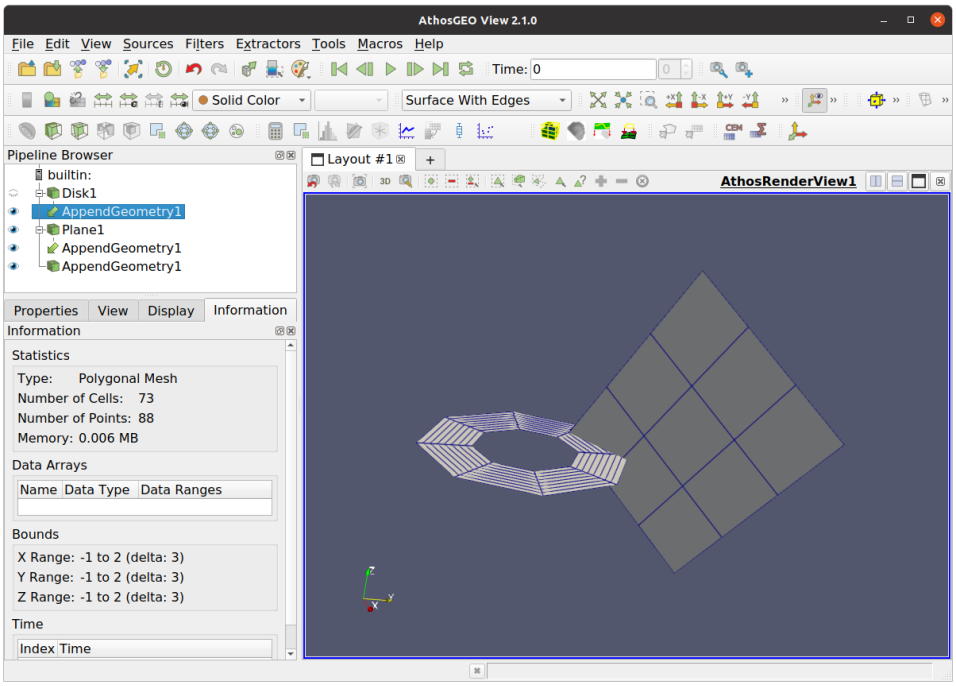


Fig. 2.41: Two or more **Polygonal Mesh** geometries can be merged into one single **Polygonal Mesh** with the application of the **Append Geometry** filter

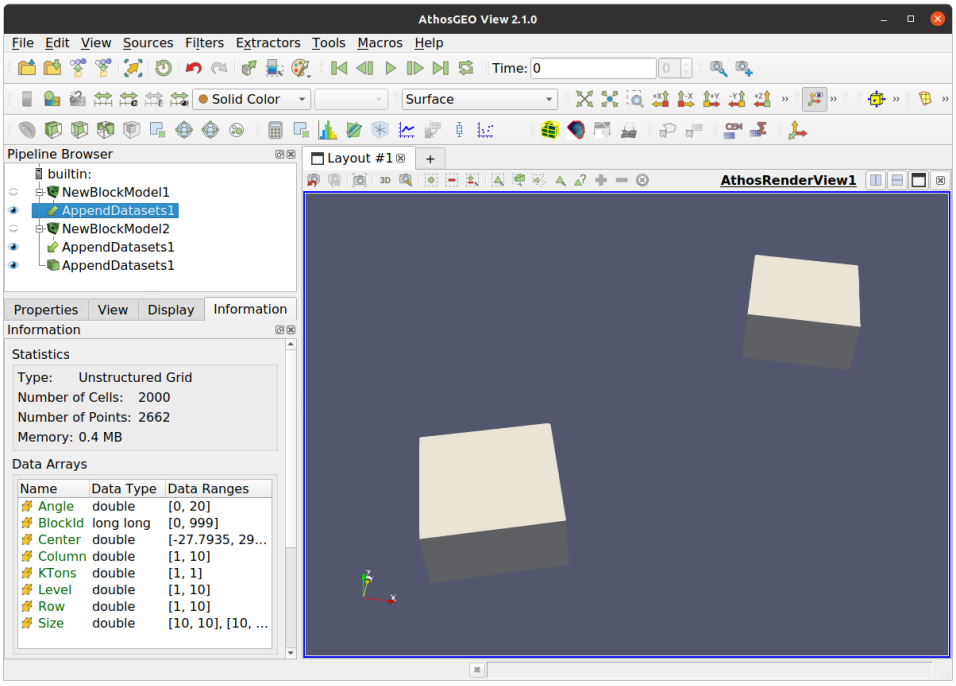


Fig. 2.42: Two or more **Unstructured Grid** and **Polygonal Mesh** geometries can be merged into one single **Unstructured Grid** with the application of the **Append Datasets** filter

Finalizing merged items

If merged surfaces or block models are actually adjoining each other, the merging process should be completed with the application of one of the **Cleaning** filters:

- The **Clean** filter is for **Polygonal Mesh** data. Apply it in order to e.g. merge coinciding points at the boundary between the previously merged geometries
- The **Clean Cells to Grid** filter does the same thing for **Unstructured Grid** objects like block models

---

Hint

The **Clean to Grid** filter does basically the same thing, but it will always generate an **Unstructured Grid** as output.

2.5.4 Clipping and Cutting

Clipping and cutting block models and surfaces are more complex operations, with different filters doing a similar job, but with a different focus. The following sections will deal with them in an order that is based on typical use cases:

Generic clipping of block models or surfaces

- Clip Geometry

Manually shaping a manually generated block model

- Clip Model with Boundary
- Clip Model with Surface
- Clip Model with GeoTiff

Cutting precise holes into a topo

- Topo Cutter
- Outline Block Model

Generic Clipping 

The **Clip Geometry** filter combines a number of functions into one filter, to allow clipping both *block models* and *topo surfaces* with both *surfaces* and *boundary lines*. The focus is on "getting it done" in a robust way, not on ensuring that the clipping is in all cases 100% perfact, no matter what the resolution (point density) of the input geometries is - see the examples below.

Clipping a Sphere (VTK type *Poly Data*) with a Plane (*Poly Data*)

The input geometries - sphere and plane - are shown in Fig. 2.43.

- Without any options, cells are not clipped, and only cells that are fully on one side of the clipping plane are kept in the output: Fig. 2.44.
- With the *Keep cells along boundary* option turned on, also all those cells that are cut by the clipping plane will be kept in the output: Fig. 2.45.
- The *Invert output* will invert the orientation of the clipping plane, while not inverting the effect of the *Keep cells along boundary* option: Fig. 2.46.
- The *Clip cells* option will actually clip also the cells, generating new ones: Fig. 2.47.

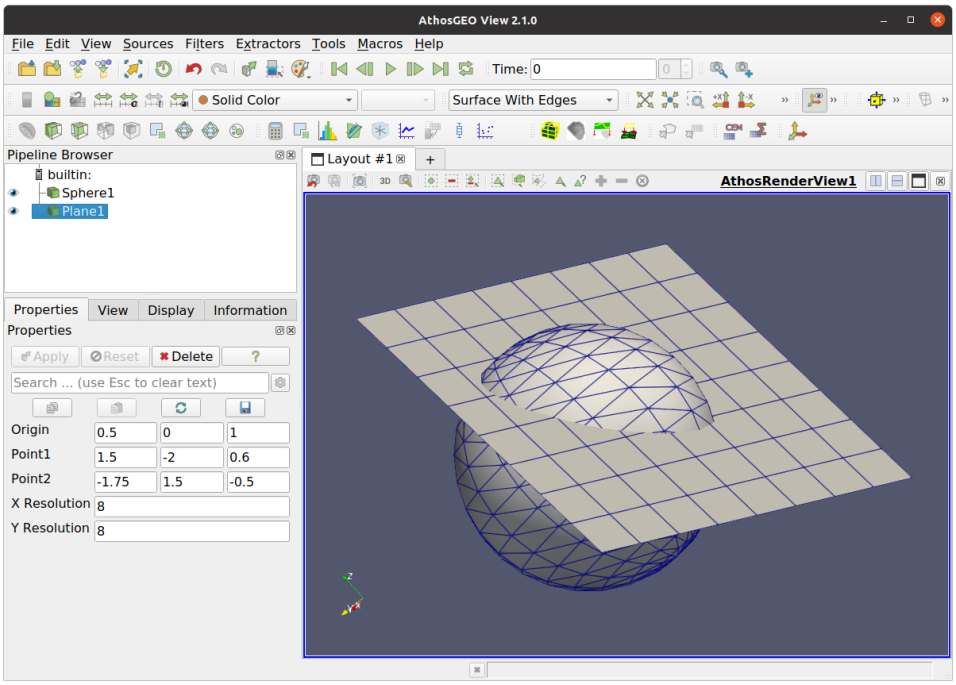


Fig. 2.43: A simple **sphere** that is intersected by a simple **plane**. The following images will illustrate the different options

- Note that the *Only cells along boundary* option does not have an effect if the input is of type *Poly Data*, like the sphere in this example.

Clipping the sphere can also happen with a **boundary line**. That boundary line will be extended along the Z axis, like a "cookie cutter":

- Clipping the sphere without further options: Fig. 2.48.
- The same thing with the **Clip cells** option turned on: Fig. 2.49.

Hint

If precise "cookie cutting" is important, please have a look into the *Precise Topo Cutting* section.

For the following examples, a small **block model** (VTK type **Unstructured Grid**) will be clipped instead of the sphere of the previous examples:

- All options turned off: Fig. 2.50
- With the **Keep cells along boundary** option, blocks (cells) that are cut by the clipping plane are retained in the output: Fig. 2.51
- The **Invert output** flips the orientation of the clipping plane (but not the effect of the **Keep cells along boundary** option): Fig. 2.52
- For block models, the **Keep only blocks along boundary** option will only keep those blocks in the output that are cut by the clipping plane: Fig. 2.53
- The **Clip cells** option, if applied to block models, will clip also those blocks (cells) that are cut by the clipping plane - with the effect, that they are not any more cuboids at the end: Fig. 2.54

Hint

The **Clip Geometry** filter clips a block model only geometrically, without taking **attributes** into account. If this

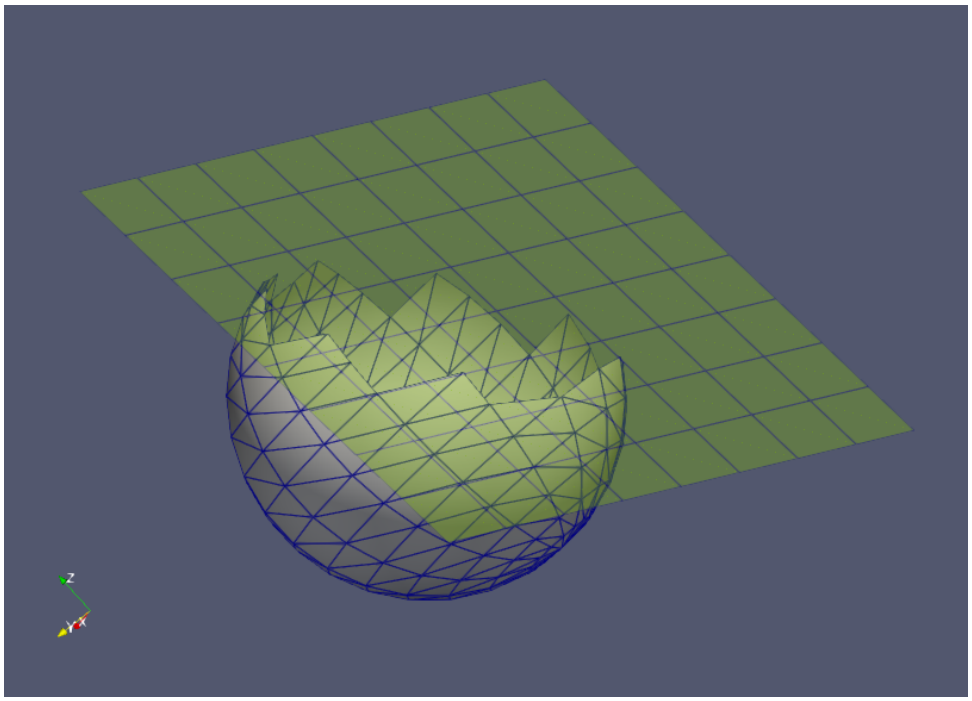


Fig. 2.44: All options are off

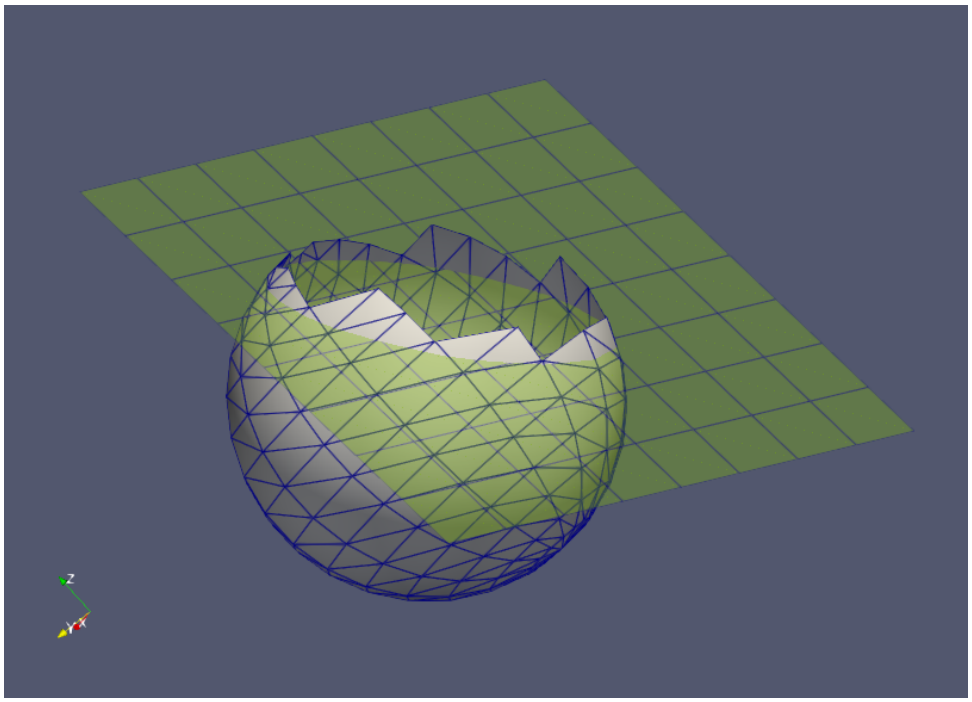


Fig. 2.45: With **Keep cells along the boundary** on, cells that are cut by the clipping plane, are kept in the output

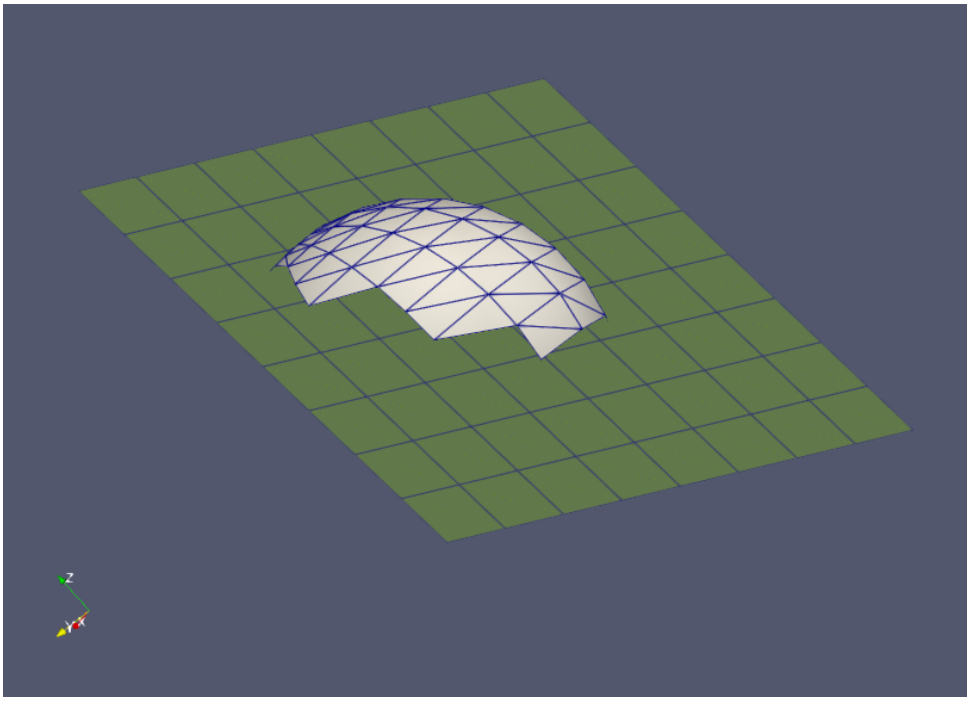


Fig. 2.46: The **Invert output** option will invert the orientation of the clipping plane. Note that the **Keep cells along the boundary** option is not affected by this option

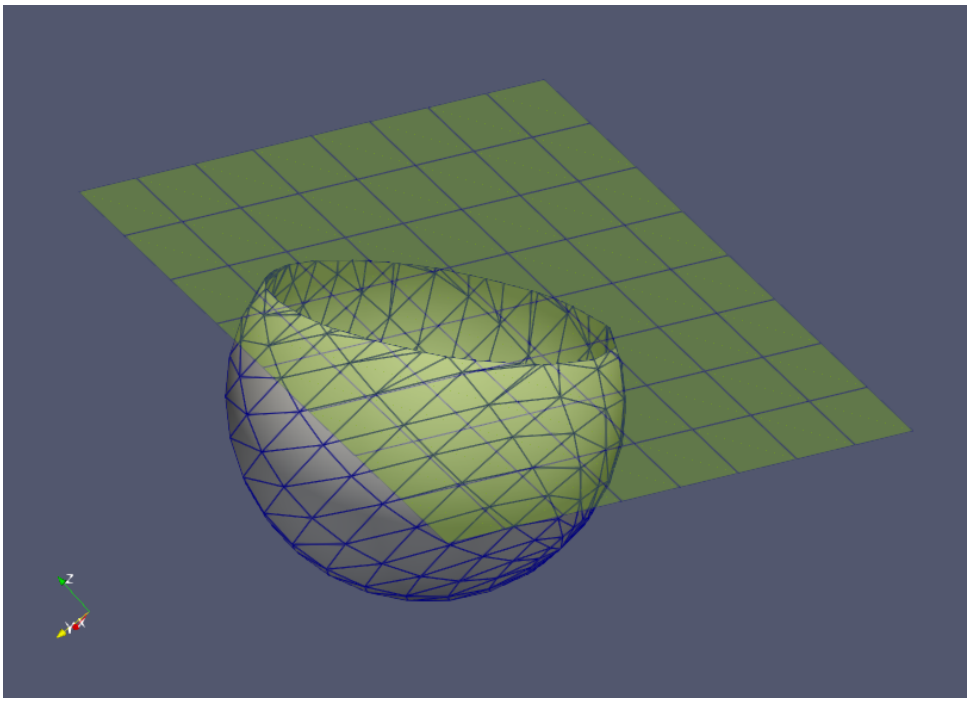


Fig. 2.47: The **Clip cells** option will clip the cells and generate new ones. However, it does not do an effort to ensure a 100% perfect cut along the clipping plane. The result largely depends on the resolution of the input geometries, so refining the triangulation will result in more precise clipping.

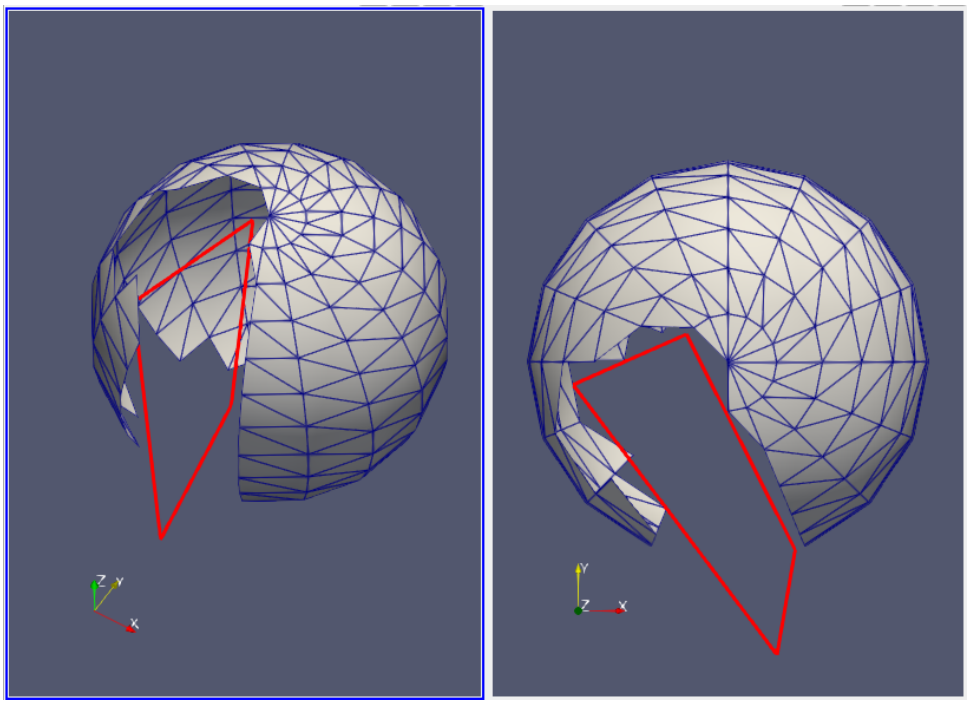


Fig. 2.48: Clipped the sphere with a boundary line results in a “cookie cutter” effect along the Z axis.

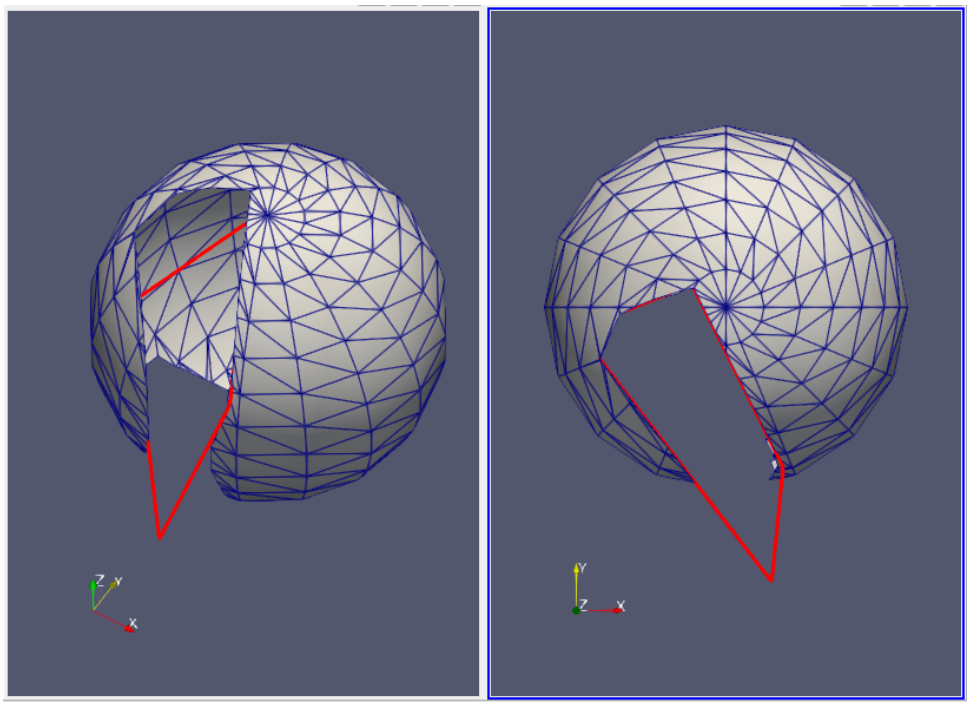


Fig. 2.49: Clipped the sphere with a boundary line results in a “cookie cutter” effect along the Z axis.

is the intention, please have a look into the *Shaping a Block Model* section.

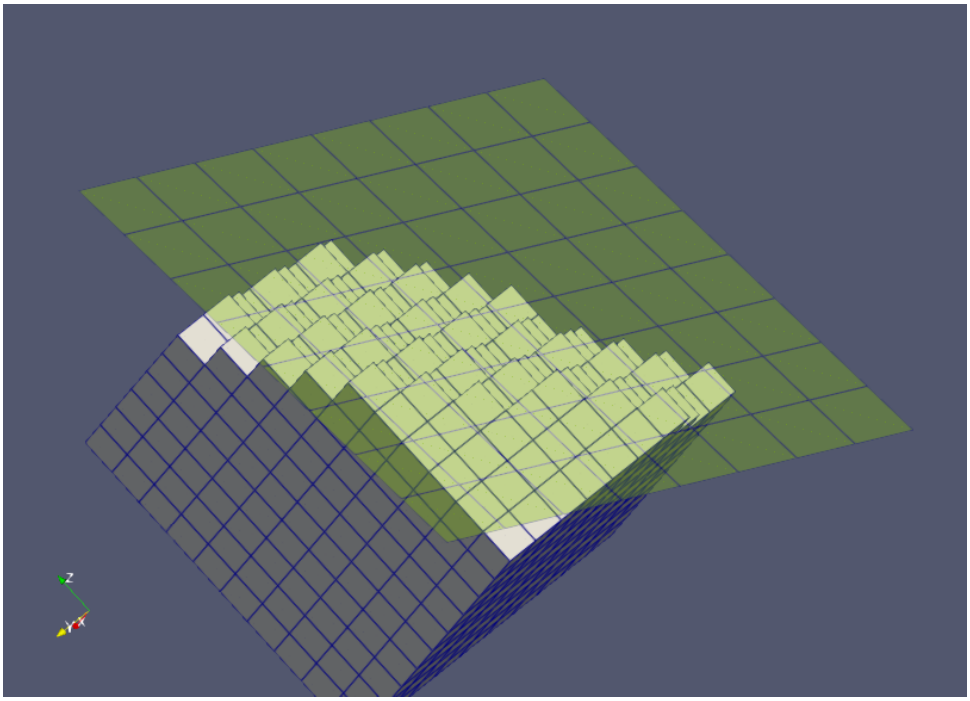


Fig. 2.50: Clipping a block model without further options

Shaping a Block Model

Normally, block models are assumed to be generated with other tools for further processing with **AthosGEO View** and **AthosGEO Blend**. However, it is possible to generate simple block models also "manually", with the help of **AthosGEO View**. This includes the initial generation with the **Block Model** source, and assigning attributes with the different attribute assignment filters.

On top of that, we want to adapt the shape of the model to the volume that we want to cover. However, the **Clip Geometry** filter only deals with the geometry, but not with the specifics of a block model with it's typical attributes and attribute types (see Section 1.6).

Example: Cutting a block model should keep the blocks intact, which we can achieve also with the **Clip Geometry** filter without the *Clip the cells* option. But along the boundary, we will certainly have blocks with reduced tonnage, and this cannot be handled with the **Clip Geometry** filter: This is the domain of the specific block model clipping filters.

Clip Model with Boundary 

The function of the **Clip Model with Boundary** filter is that of a "block model cookie cutter". Input is a block model and a closed boundary line (polyline) - see Fig. 2.55.

The result of the clipping can be seen in Fig. 2.56. Part of the resulting block model is also shown in tabular form in Fig. 2.57. You see there that not only the *KTons* attribute was adapted to the fraction of a block inside the boundary line, but also a new attribute *VolFactor* was generated that reflects the percentage of the block that is part of the model.

Hint

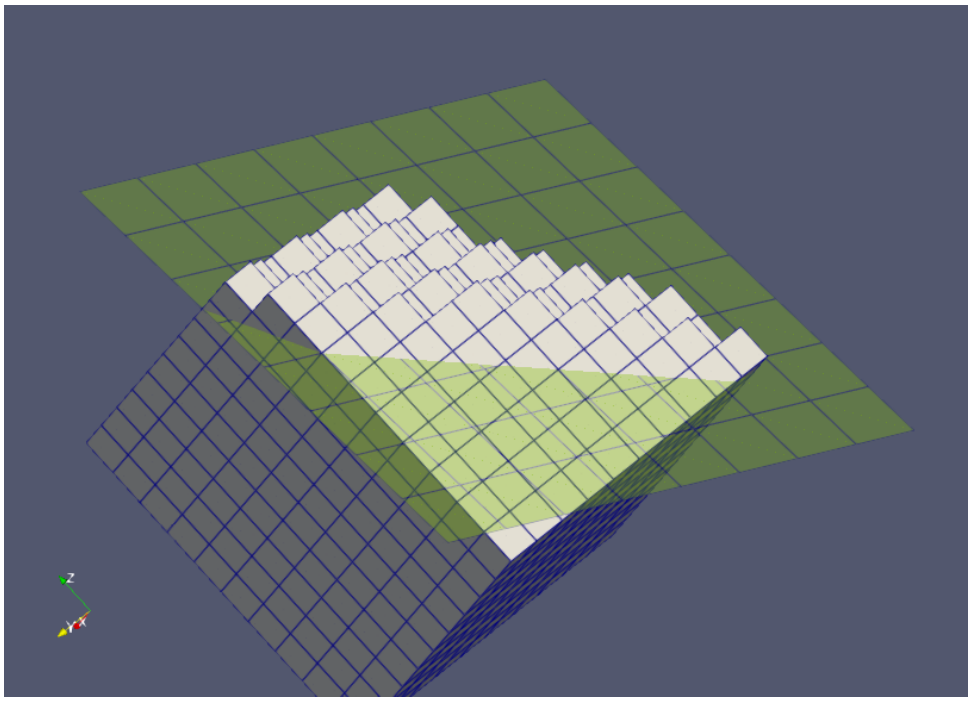


Fig. 2.51: Keep also the blocks (cells) that are cut by the clipping plane

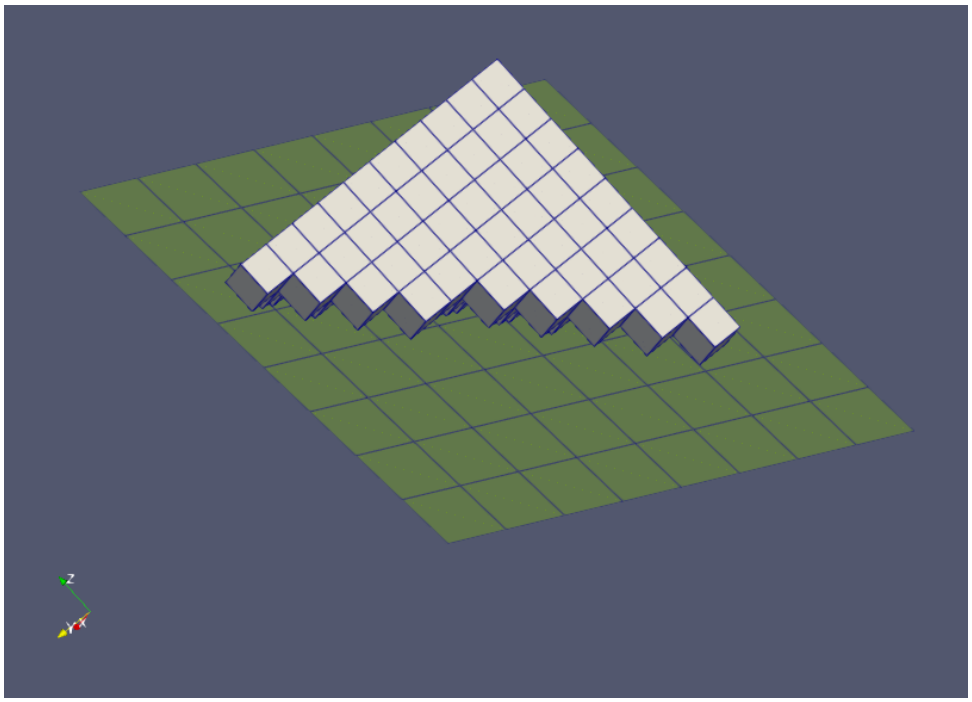


Fig. 2.52: Invert the orientation of the clipping plane

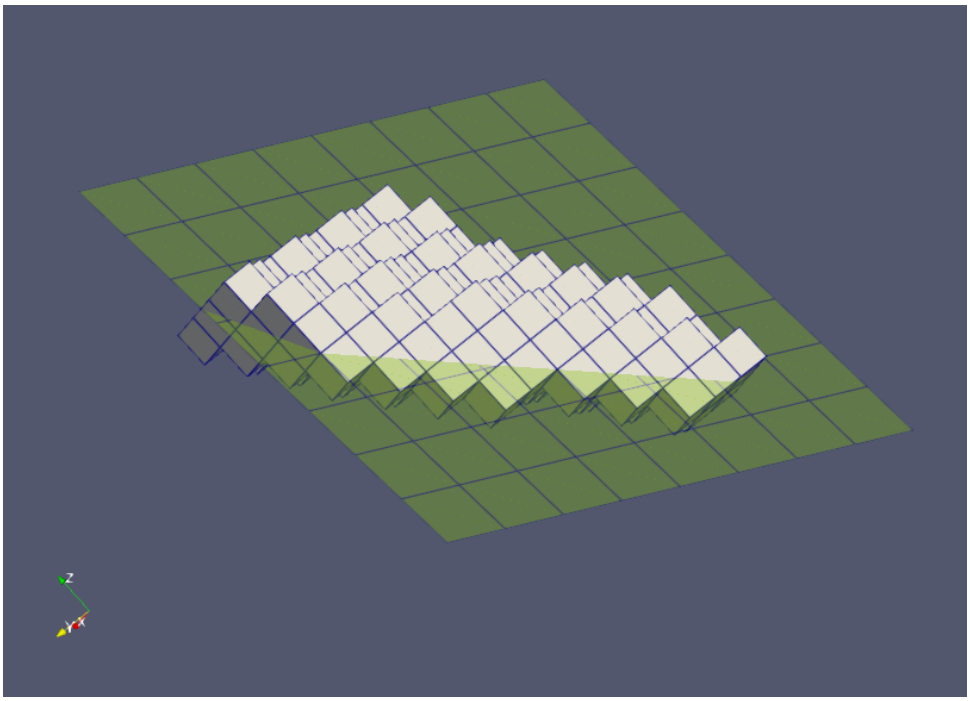


Fig. 2.53: Keep only those blocks (cells) that are cut by the clipping plane

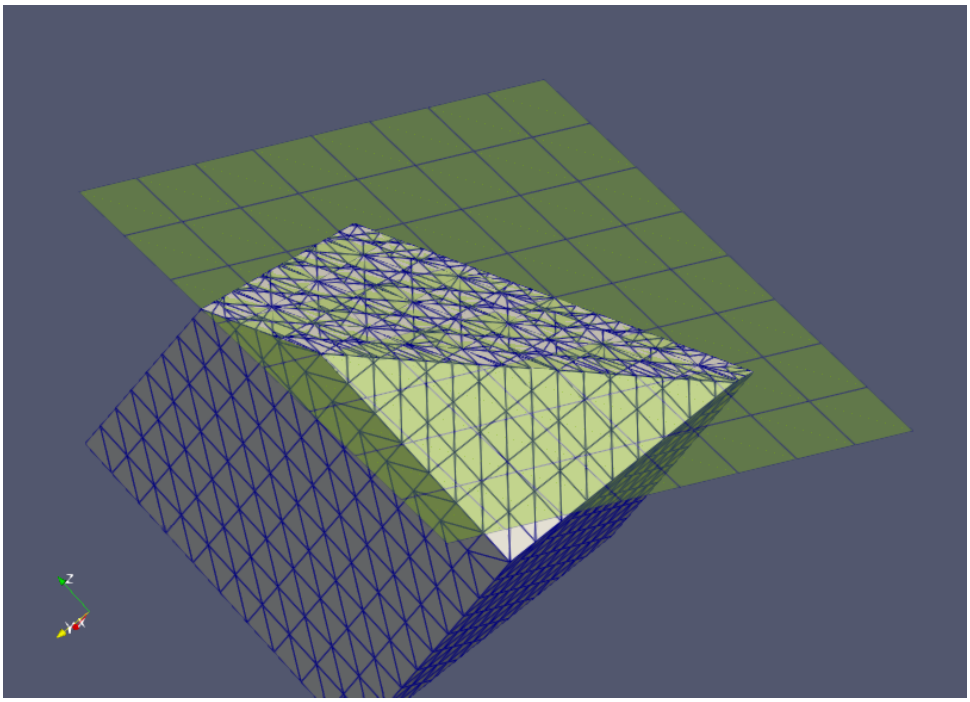


Fig. 2.54: Clip also the blocks (cells) of the model - with the effect that they are not any more cuboids

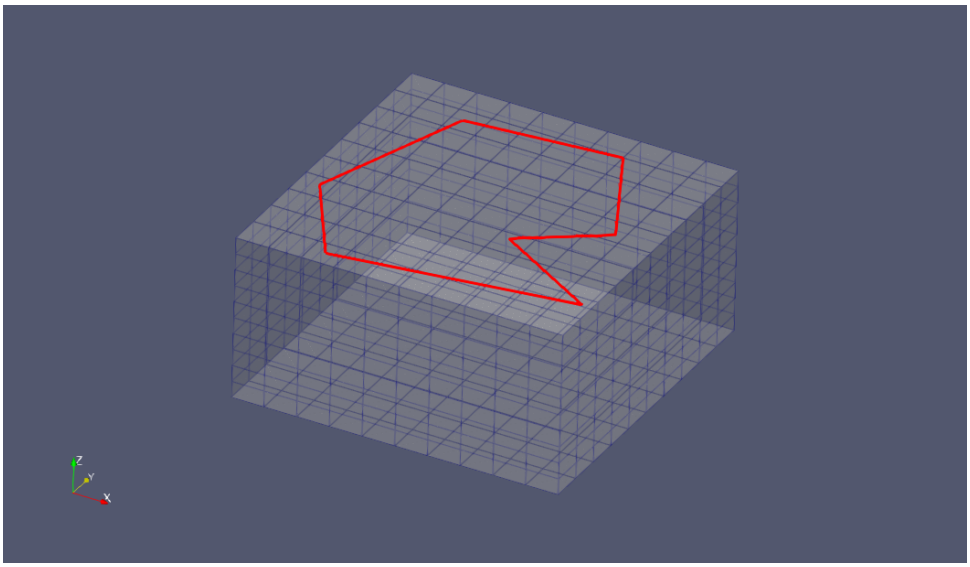


Fig. 2.55: An "empty" block model (in this case generated with the **Block Model** source) and a polyline (generated with the **Poly Line** source), to be used as input for the **Clip Model with Boundary** filter

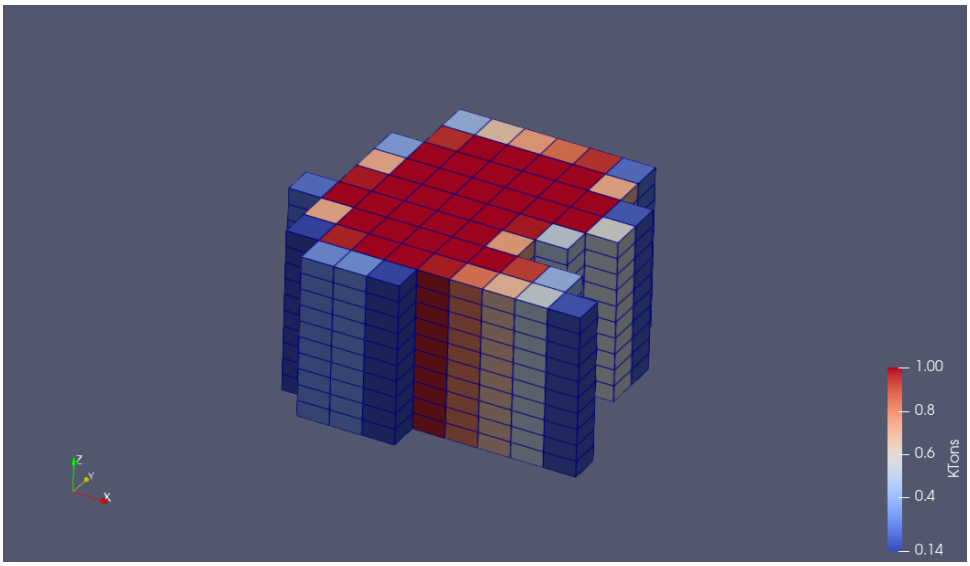


Fig. 2.56: Initially, all blocks in this artificial example model had a tonnage of **1 KTons**. After clipping with the boundary line, tonnage of each block was adapted in such a way to reflect only the tonnage that is inside the boundary line, while the shape of the blocks were not changed.

Cell ID	Level	Column	Row	KTons	VolFactor	Center_0	Center_1	Center_2	Size_0	Size_1	Size_2	BlockId	Angle
0	1	3	1	0.300377	30.0377	25	5	2.5	10	10	5	0	0
1	1	4	1	0.312743	31.2743	35	5	2.5	10	10	5	1	0
2	1	5	1	0.154013	15.4013	45	5	2.5	10	10	5	2	0
3	1	2	2	0.144973	14.4973	15	15	2.5	10	10	5	3	0
4	1	3	2	0.972687	97.2687	25	15	2.5	10	10	5	4	0
5	1	4	2	1	100	35	15	2.5	10	10	5	5	0
6	1	5	2	1	100	45	15	2.5	10	10	5	6	0
7	1	6	2	0.97773	97.773	55	15	2.5	10	10	5	7	0
8	1	7	2	0.836553	83.6553	65	15	2.5	10	10	5	8	0

Fig. 2.57: The tabular display of the clipping result illustrates the fact that existing attributes are handled according to their meaning, and an additional **VolFactor** attribute is added if it does not exist yet

The **Minimum Retained Volume Factor** option allows to discard those blocks that would contain only a very small fraction of the original block tonnage. Set this to 0 if such a behaviour is not allowed, or to some pragmatic low value like 10%.

Regarding the **Respect Preceding Horizontal Clip** option, please see the following section *Clip Model with Surface*

Clip Model with Surface 

The **Clip Model with Surface** filter is intended for adapting a block model to a real topo surface from above (with option **Keep Blocks Below the Surface** turned on), or cutting it from below at some underground limit (same option turned off). Input is a block model and a triangulated surface (in the sense of a "topo surfac" or "2.5D surface"), as shown in Fig. 2.58, and the result would be like shown in Fig. 2.59.

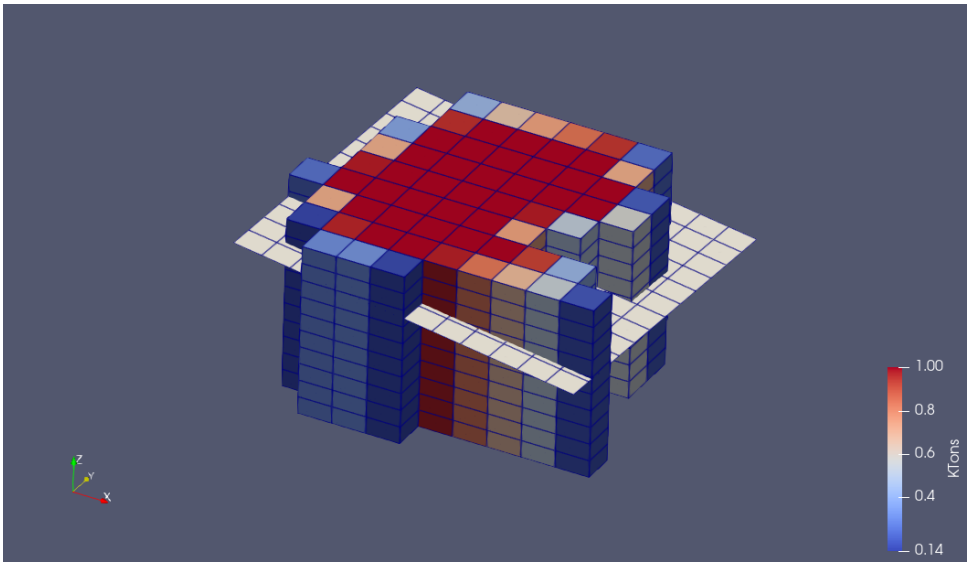


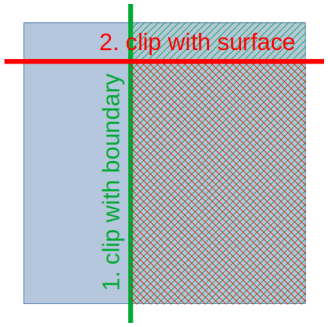
Fig. 2.58: The clipping surface of the model is supposed to be of type "2.5D", with no more than one value of Z referring to every (X, Y) coordinate pair

Hint

The **Respect Previous Vertical Clip** option has an effect if the input block model already has a **VolFactor** attribute from a previous clipping operation. The problem is that the filter cannot "know" from with side of a block any previously clipped volume was removed, so some simplifications must be made - and the user needs to give a hint in order to make the approximation a little bit more reliable.

The assumption is that we have two cases to distinguish:

- The volume factor reduction is the result of a previous vertical clip, i.e. the application of a **Clip Model with Boundary** filter:



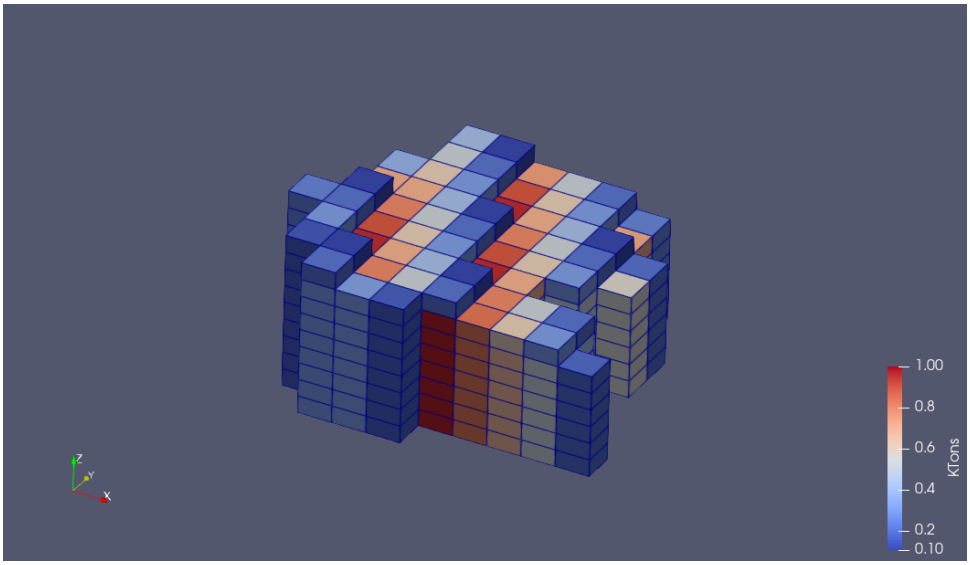
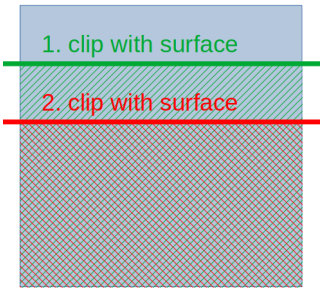


Fig. 2.59: Like with the *Clip Model with Boundary* filter, also this filter will handle attributes according to their meaning, like adapting the tonnage of a block to the volume fraction that remains inside the block model after clipping, while leaving the shape of the blocks unchanged

- The volume factor is an update of a previous horizontal clip, i.e. the application of a **Clip Model with Surface** filter:



In the first case, the resulting **VolFactor** can be found by a multiplication of the old with a new volume factor, while in the second case, the result will be simply the smaller value of the two

Clip Model with GeoTiff

This filter is similar to using the **Clip Model with Surface** filter, except for the fact that the filter asks directly for a **GeoTIFF** file as a property that describes the topo surface to be used. The result would be something like shown in Fig. 2.60.

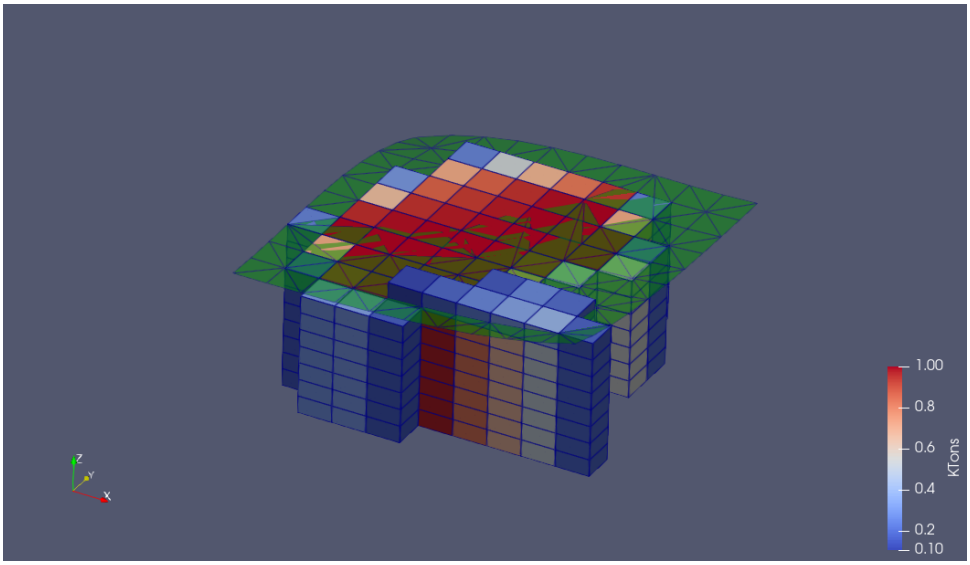


Fig. 2.60: This model is clipped directly with a **GeoTIFF** file. The same file has also been loaded with the **GeoTIFF Reader**, for illustration purposes: that step is not required for the functionality of the clipping filter.

Precise Topo Cutting 

The **Clip Geometry** filter described above in *Generic Clipping* is not designed to clip a top surface with maximum possible precision, as can be seen in Fig. 2.61 and Fig. 2.62.

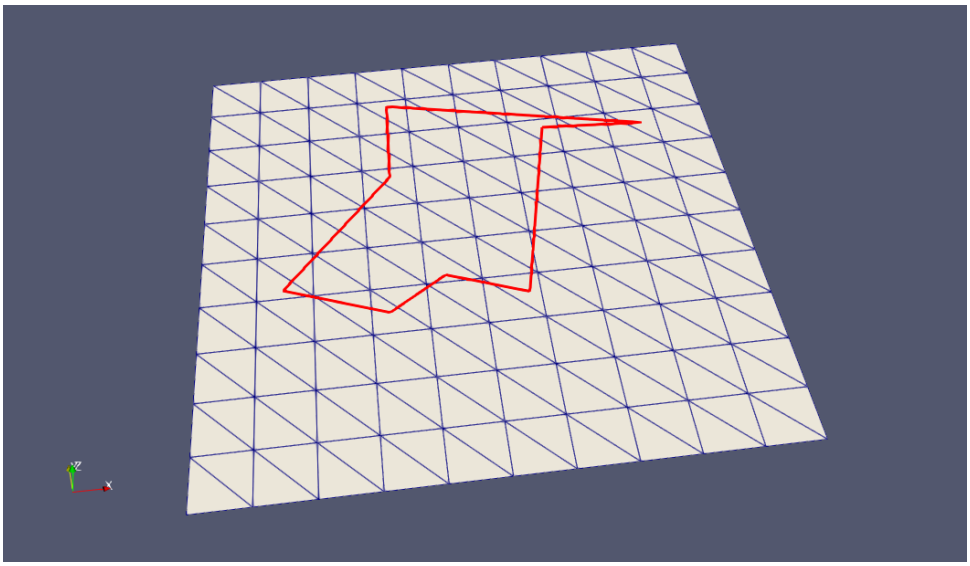


Fig. 2.61: A triangulated surface that symbolizes a **topo surface**, and a polyline **boundary line**. The purpose is to use the boundary to clip a "hole" into the topo surface, in a "cookie cutter" style

It is obvious that the punched hole is not following the boundary input line precisely. However, if the required geometric resolution of the topography is defined by the input triangles, the approximation does not look unreasonable.

If a higher resolution is needed, it is possible to apply the **Subdivide** filter to the topo, the geometric resolution will be increased, and the punched hole accordingly - as can be seen in Fig. 2.63.

With all the above background, it is immediately obvious how the **Topo Cutter** filter is working with a very different algorithm: see Fig. 2.64.

The difference in strategy can be summarized in two points:

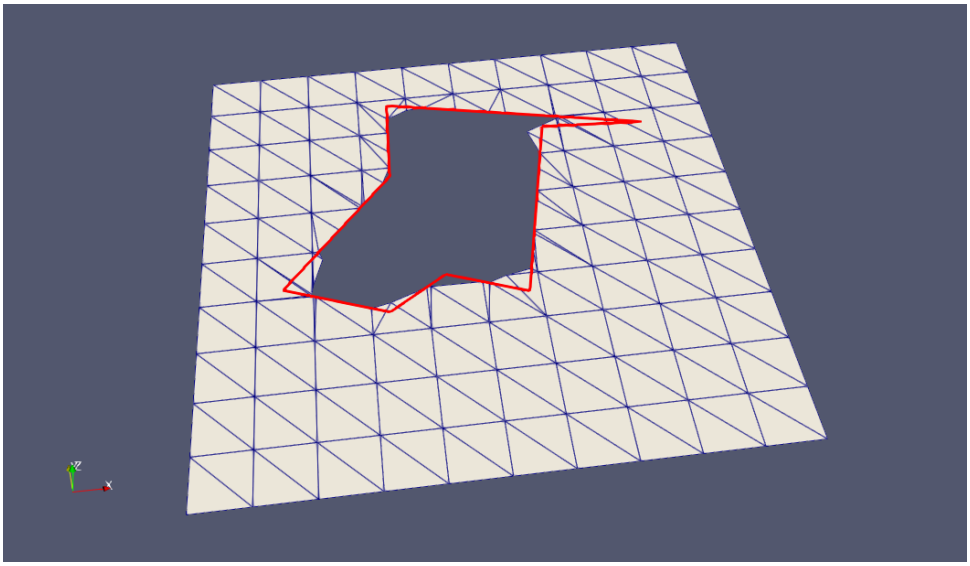


Fig. 2.62: The cookie cutting job as defined in Fig. 2.61 can be done with the **Clip Geometry** filter, and the result will be as shown here

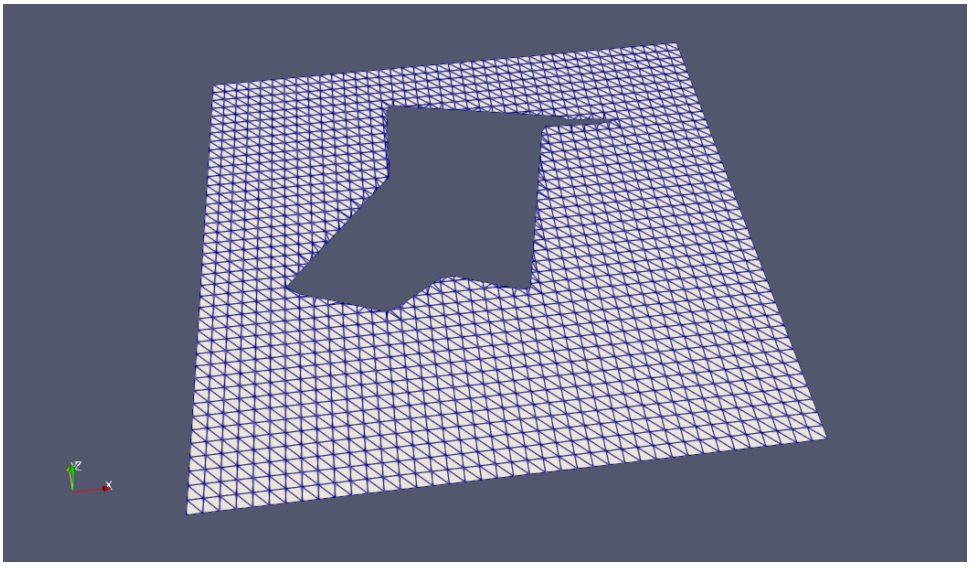


Fig. 2.63: After application of the **Subdivide** filter to the input topo, the **Clip Geometry** filter will generate a result with much higher geometric resolution than shown in Fig. 2.62

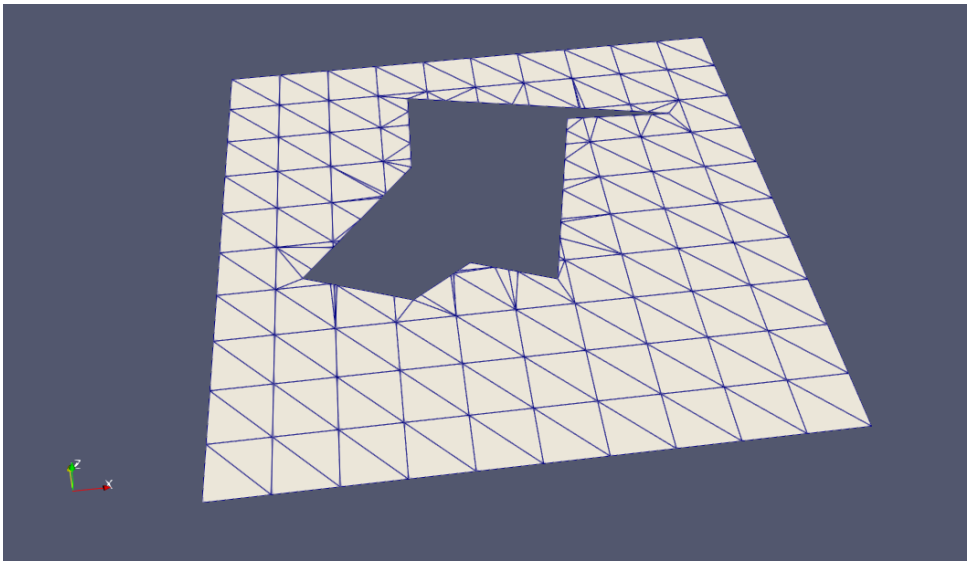


Fig. 2.64: The **Topo Cutter** filter cuts every single triangle in the best possible manner with the input boundary

- Clipping is done triangle by triangle in a way that the **boundary line is followed exactly**. This includes even cases where we have more than one closed boundary, and even if closed boundaries are nested within each other, as shown in Fig. 2.65. (Exception: self-intersections are not allowed.)
- No clipping algorithm will ever be 100% perfect in all situations. While **Clip Geometry** is not as precise as possible, it tends to be very robust. The **Topo Clipper**, on the other hand, tries to do it's best regarding precise clipping, but in some special cases it fails: Adapt the *Delaunay Offset* and/or the *Number of Points to Insert* properties.

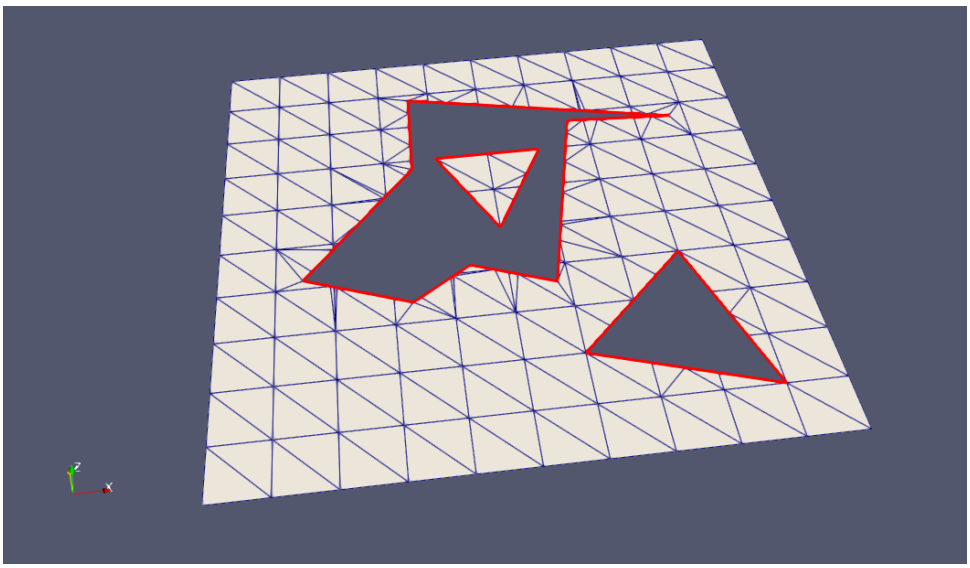


Fig. 2.65: With the **Topo Cutter**, a boundary can consist of several closed polylines that may even be nested. Only (self) intersections are not allowed

Outline Block Model

The **Outline Block Model** filter is a complement for the **Topo Cutter** filter described above if the purpose is to cut a hole into a topo surface that matches a block model precisely. The block model outline will be generated at a Z value of 0, as shown in Fig. 2.66. It is directly useable as input for the **Topo Cutter** filter.

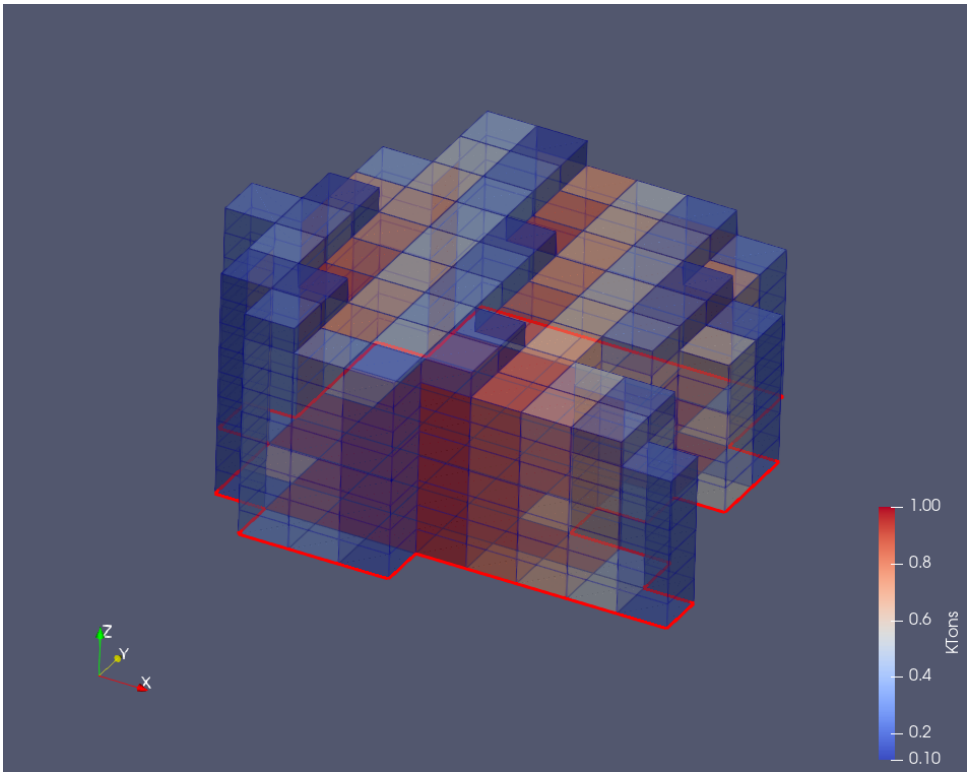


Fig. 2.66: The **Outline Block Model** filter.

2.6 Selections

Selecting blocks in a model is important for the function of many filters, but in order to successfully work with them, it is important to know two three things about them in the context of **AthosGEO View** and **ParaView**. With this it makes sense to dedicate an entire chapter of this manual only to selections.

2.6.1 Fundamentals about Selections in AthosGEO View

The selection system of **AthosGEO View** is inherited from **ParaView**, but **AthosGEO View** has to use it in a slightly uncommon way for some of it's filters. Some basics that need to be known:

- Within the entire **filter pipeline**, always **only one selection** can exist. This is important to know because many filters do rely on an active selection, and it is of course possible to have more than one filter in the pipeline that needs a selection as input.
- The filter pipeline can be saved into a **state file** (***.pvsm** file), but this does not include the active selection. However, the selection definition is retained as a **property**, so it will be recovered if a state file is loaded.

The effect of the latter is that a pipeline with several filters that require selections will work properly and can even be saved to and loaded from **state files**, as illustrated in Fig. 2.67 to Fig. 2.69. A problem arises if the user wants to go back in the pipeline and **modify** a selection.

The **"trick"** that **AthosGEO View** and **ParaView** applies to achieve the result as in Fig. 2.69 can be seen in the **Properties** panel: parameters of the selection were saved and recovered as properties of the filter. However, going

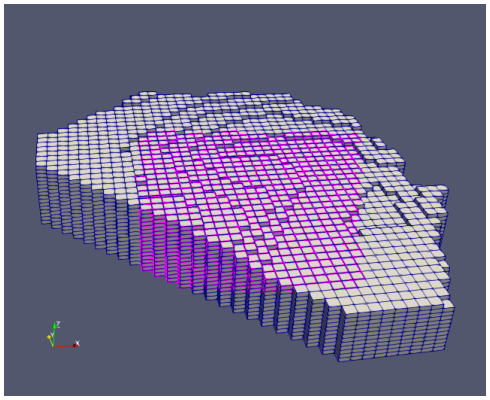


Fig. 2.67: A **demo model** with some **selected blocks**.

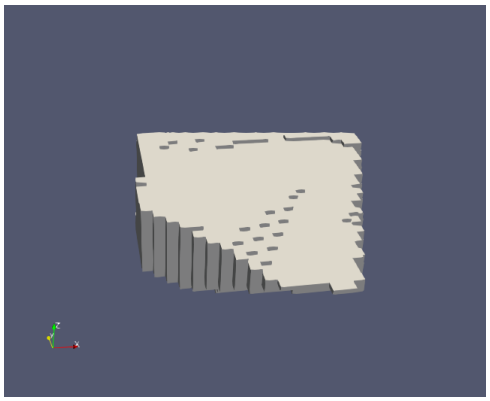


Fig. 2.68: An **Extract Selection** filter was appended to the filter pipeline and the extracted blocks were extracted.

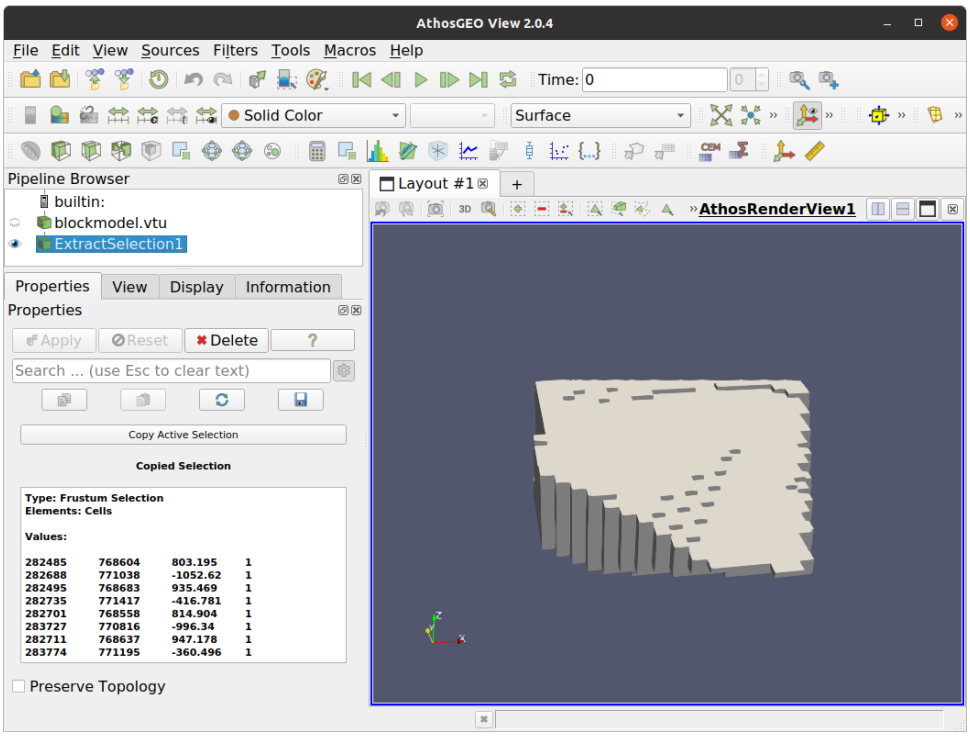


Fig. 2.69: The pipeline with **demo model** and **Extract Selection** filter was saved to a **state file** and loaded back into **AthosGEO View**: The render view shows the extracted blocks like after initially applying the filter.

back in the pipeline, the selection does not reappear: it does not exist any more.

What the user can do:

- Go back to the block model,
- Make a completely new selection,
- In the properties panel of the filter, press the *Copy Active Selection* button and
- Press also the *Apply* button.

As a result, the displayed **selection parameters** will appear in the **Properties** panel, and a new extraction is shown in the view.

Did you know?

The example shows the **typical order** of actions with selections that require a selection:

1) Do some **selection** on the pipeline item that **precedes** the pipeline item that represents the filter.
2) Add the **filter** that will recognize the existing selection and show it's parameters in the **Properties** panel.
3) If the user wants to **change** the selection now, he needs to switch between **input item** and **filter item** in the pipeline browser accordingly.

2.6.2 Selection Tools

Most of the selection tool buttons are found immediately above the **Render View** (3D view) panels.

Did you know?

In the following sections, only selection tools for cells will be explained!

The **Athos Render View** in **AthosGEO View** is a special version of the **Render View** that exists already in **ParaView**. The difference is the availability of **selection tool buttons**. Functionality is not added, but removed: The original render view has the same selection tools not only for **cells**, but also for **points**.

Why has this functionality been removed for **AthosGEO View**? It is because we have a very strong focus on **block models**, and in this special case the availability of point selection buttons is only confusing!

This is not a problem, because for all cases where the user needs point selection tools, the original **Render View** is still available.

Select Cells On 

Select a rectangle and all model blocks (cells) that are visibly partially or fully inside this rectangle will be selected. An example of this functionality is shown in Fig. 2.70.

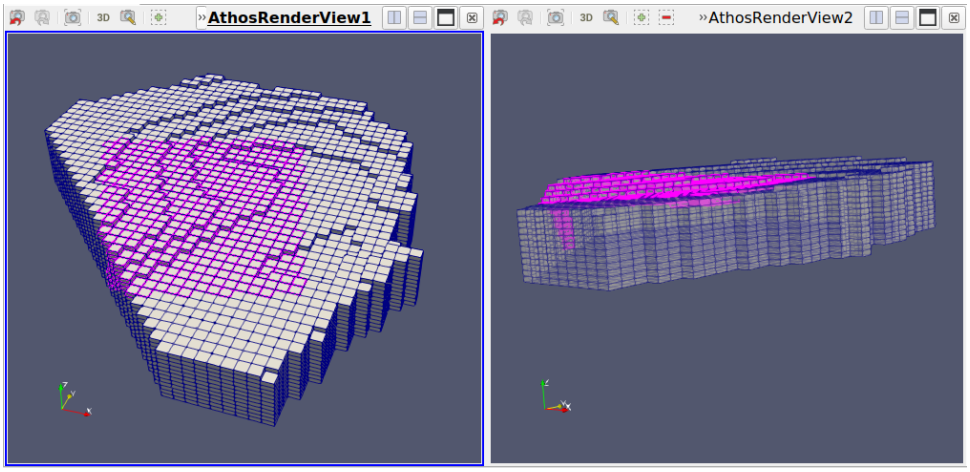


Fig. 2.70: With the **Select Cells On**  function, only visible surface blocks are selected within a rectangular view area.

Select Cells Through 

Same as above, but in this case also all those model blocks (cells) will be selected that are visibly and invisibly inside a rectangular prism, as seen in Fig. 2.71.

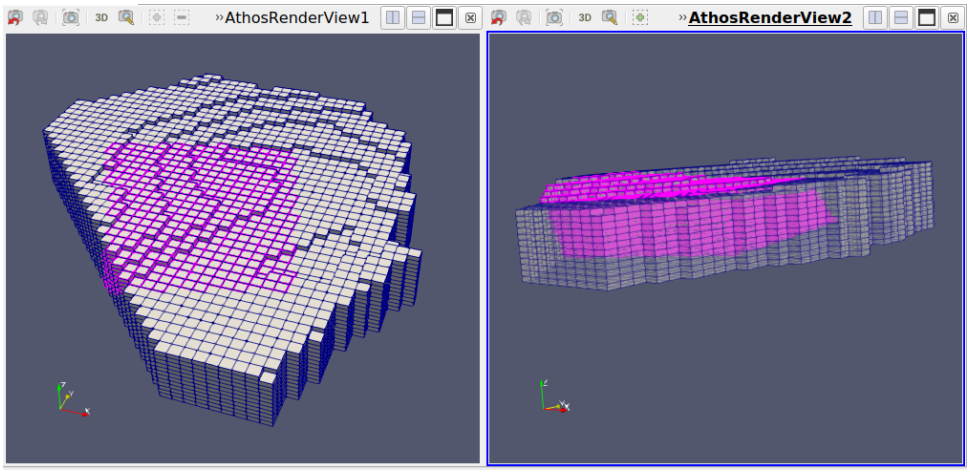


Fig. 2.71: The difference between **Select Cells Through**  and **Select Cells On**  is the fact that the first will select only those model blocks that are invisibly "behind" the visible selected rectangle.

Hint

The **Select Cells Through** tool can be pretty dangerous because it will select blocks that you do not even see because they are behind the visible surface blocks! There are two things that you can do to ensure best possible control of what is going on:

- In the **View** panel, check the **Camera Parallel Projection** option to gain more control about which model blocks will actually be affected.
- The **Set View Direction to…** buttons in the **toolbar** can be used to to align the view exactly in parallel to one of the main coordinate axes.

Select Cells with Polygon 

This tool allows to select blocks inside a more complex area than a simple rectangle. It selects only blocks at the visible surface, as shown in Fig. 2.72.

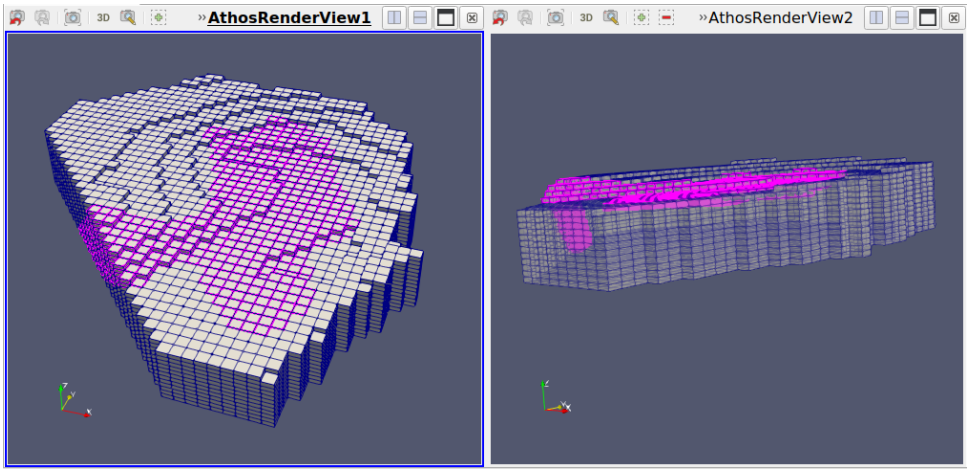


Fig. 2.72: In order to select surface blocks with the **Select Cells with Polygon** tool, press the left mouse button, move along the desired outline, and once the button is released, the polygon will be closed automatically.

Select Cells Interactive 

The **Select Cells Interactive** tool is a **toggle** that is turned on or off by pressing the  button. If it is on, clicking on a model block will select it.

Hint

Make sure to always turn this tool off after use!

There is no automatism that would turn the tool off automatically at any time. If it is not done manually, the behaviour of the view will be "strange", and in the worst case unintended things will happen.

Find Data 

The **Find Data** docking panel is certainly the most powerful selection function: It selects model blocks (*cells*) by criteria. The icon  cannot be found among the other selection tool buttons but in the main **toolbar** of **AthosGEO View**. Alternatively, the Find Data docking panel can also be opened through the **View** menu, and it will appear as shown in Fig. 2.73.

Note

The **Find Data** tool is very powerful regarding selections by attributes, but it handles only **numeric** attributes, not **strings**, as shown in Fig. 2.74.

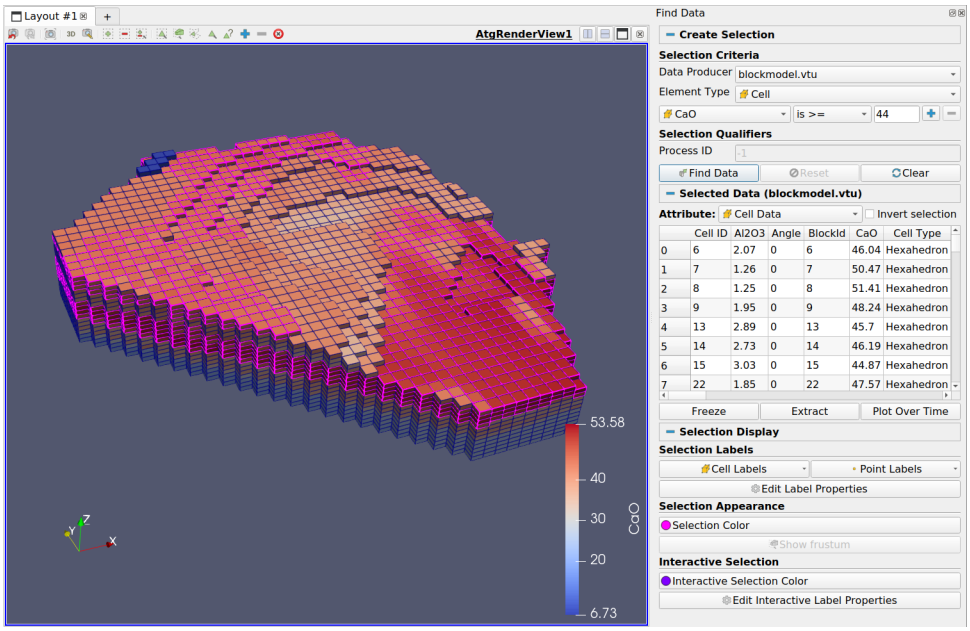


Fig. 2.73: The **Find Data** docking panel allows to specify criteria for the selection of model blocks

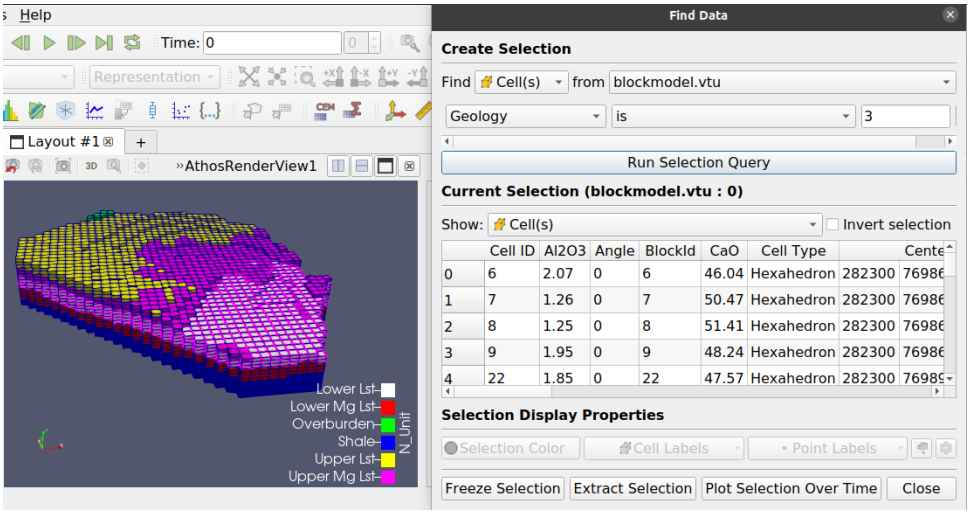


Fig. 2.74: In this example, **Find Data** is supposed to select blocks in the `Lower Limestone` geological unit. In the **N_Unit** attribute, these appear as `Lower Lst`, and in the 3D display they can be shown as such in the **legend**. However, **Find Data** can only handle **numeric** attributes, so another attribute had to be introduced, **Geology**, where the same units are represented by code numbers - 3 in this case.

2.6.3 Add, Subtract or Toggle Selection

By default, selection tools will always **replace** an existing selection. This behaviour can be changed by activating one of the following selection mode buttons at the top of a **Render View** panel:

Add Selection 

If this selection mode is turned on, any new selection will be **added** to the already existing selection.

Subtract Selection 

With this selection mode, all selection tools will actually **unselect** blocks.

Toggle Selection 

In this mode, selection tools will work as a **toggle**: selected blocks will be unselected and unselected blocks will be selected.

2.6.4 Grow and Shrink Selections

Two more handy functions, with tool buttons directly at the top of the **Render View** panels:

Grow Selection 

Pressing this button will try to **add** one more model block to the existing selection in all possible directions.

Shrink Selection 

This function will do the inverse of the previous function and try to **remove** model blocks from the border of the current selection.

Note however, that it will work only with model blocks that were previously added by the **Grow Selection** function!

2.6.5 Selections for Attribute Manipulation

The **AthosGEO View** software extends **ParaView** with some simple **selective attribute manipulation functions**. A problem is the fact that the **ParaView** selection system as described in Section 2.6.1 is not really made for such kind of functionality. As a consequence, these specific **AthosGEO View** plugins for selective attribute manipulation are dealing with selections in a somewhat different way than other plugins are doing.

Specifically, these explanations are about the following plugins:

- *Calculator for Selection*
- *Write Value*
- *Write Multiple Values*

For the user, the most important difference is the order of activation:

- **Common**
  1) **Select** blocks
  2) Activate the **filter**

- **Selective Attribute Manipulation**

  1) Activate the **filter**

  2) **Select** blocks

The important point is, that, other than described in Section 2.6.1, selections should be recovered as far as possible if the user goes back in the **Pipeline Browser** in order to **change a selection**. With the common setup, this works only as long as there is not more than one selection in the entire pipeline and if it is not re-loaded from a state file.

The series of screenshots from Fig. 2.75 to Fig. 2.78 will explain the point with an example, where first 3 values were defined in the block model using 3 instances of the **Write Value** filter, then the selection for the first of these filters was changed.

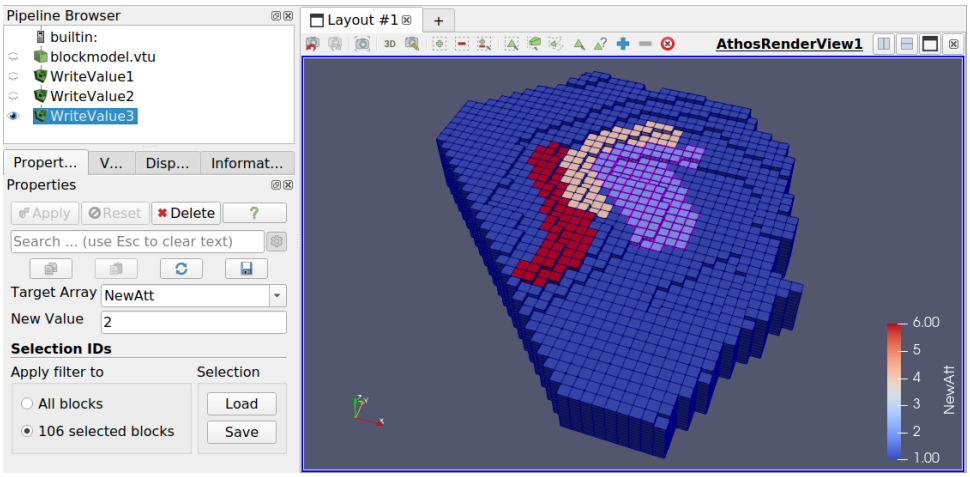


Fig. 2.75: A new attribute **NewAtt** was created in this block model and 3 different values were assigned with 3 instances of the **Write Value** filter.

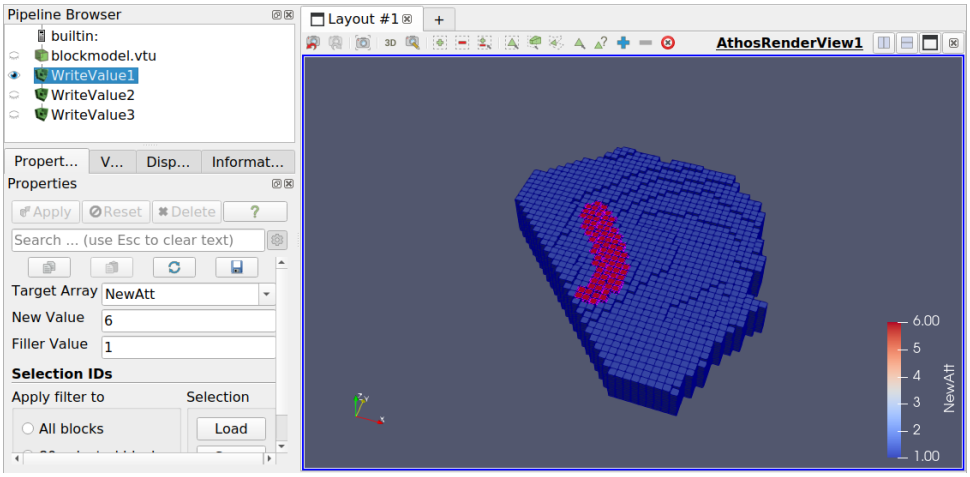


Fig. 2.76: **First** turn off the display of **WriteValue3**, in order to hide also the **selection** that is related to it. **Then** turn on **WriteValue1** by clicking on the **eye** symbol. With this, the **selection** that was related to this filter will be restored.

Note

The **order** of hiding and showing items in the pipeline can be important because of the rule that always **only one single selection** can really exist in the **entire pipeline**. If two pipeline items with selections are shown at the same time, the outcome may not always be predictable.

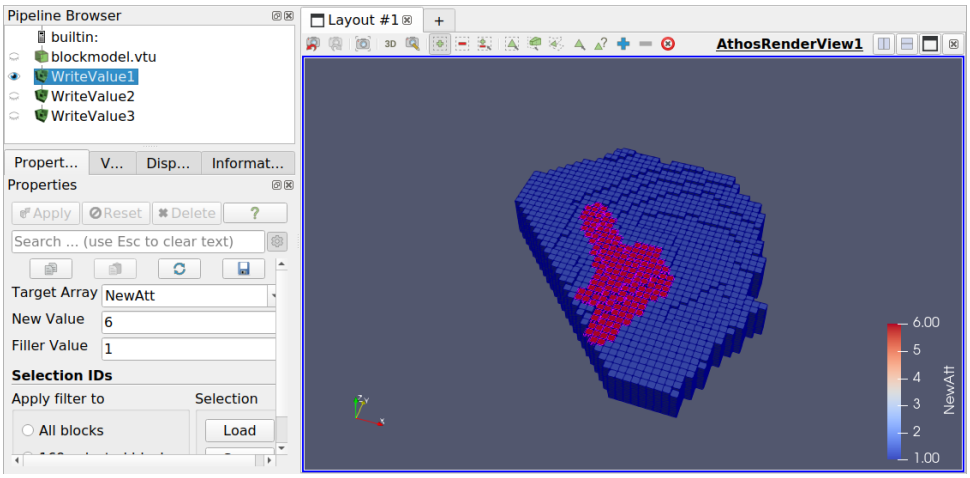


Fig. 2.77: After activating the **Add Selection** option in the toolbar above the view, an additional region was selected with the **Select Cells with Polygon** filter. Pressing *Apply* will extend the area with **NewAtt** value of 6.

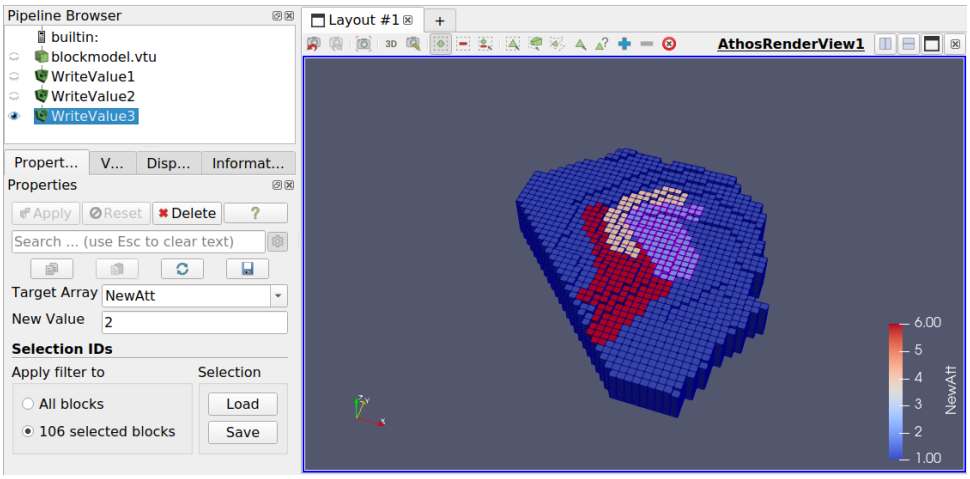


Fig. 2.78: Again **WriteValue1** should be unchecked first before showing **WriteValue3** again.

Hint

Please make use of the *Save* and *Load* buttons related to the **Selection IDs** in the **Properties** panel: If a complex selection should better not get lost accidentally, it should be saved to a file, so it can be easily restored without much effort.

Did you know?

Creating a new attribute and assigning 3 different values (plus a background value) as shown in Fig. 2.75 can also be achieved with only one instance of the **Write Multi Values** filter.

2.7 Animations

Animations are a nice feature of **ParaView** that is inherited also in **AthosGEO View**. Many of the **properties** of the filters in the pipeline can be animated.

Fig. 2.79 shows a screen where an animation is defined where the **camera** orbits around the block model around a vertical axis.

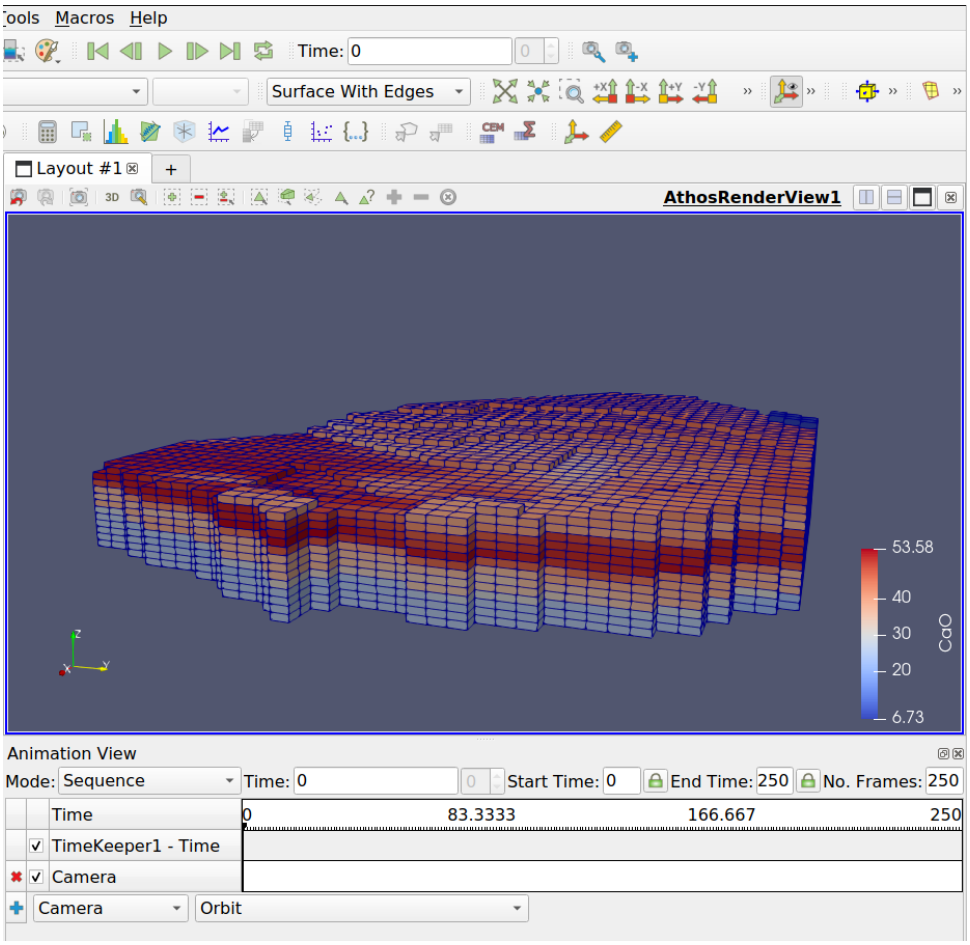


Fig. 2.79: In the **Animation View** panel, the parameters of the animation are specified. That panel can be shown or hidden from the *View* menu. The buttons for running the animation can be found in the **toolbar** .

2.7.1 Steps

The following steps will generate an animation where a **clip plane** moves through a block model, so a viewer will get a short glimpse of the internals of the model within about 10 seconds.

- As a first step, the **Animation View** panel is shown from the *View* menu, by default at the bottom of the views.
- Open a **block model** and add a **Clip** filter.
- In the **Properties** panel, our property of interest is the **Offset**. Finding the minimum and maximum values of interest for the animation is a question of trial and error.
- In the **Animation View**, select **Clip1** and **Clip Type - Offset** from the combo boxes at the bottom. Press the + button to add the offset to the animation parameters.
- Double-click on the new **Clip1 - Clip Type - Offset** button to open a dialog that allows to specify the value range for the offset within the animation, as shown in Fig. 2.80.
- Use the **animation toolbar** to run the specified animation.
- From the *File* menu, the **Save Animation** command is able to directly write the animation into a **video file**.

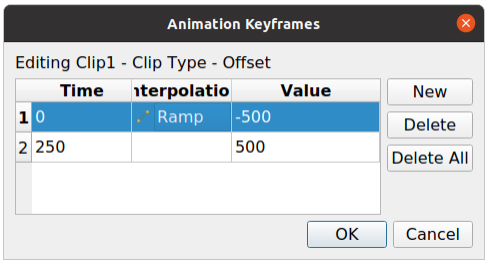


Fig. 2.80: The value range of interest for an animation can be specified in this dialog.

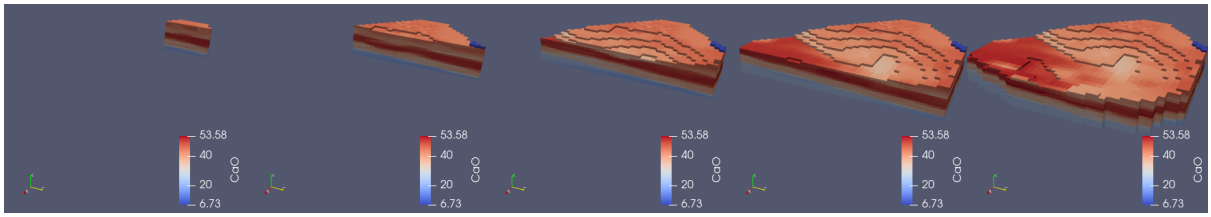


Fig. 2.81: In the resulting animation, a clip plane will move through the block model.

These steps must be good enough now for a user to get started with animations.

Loading an animation from files: Numbered sequences of files

A numbered sequence of files would be something like::

```
file01.csv
file02.csv
file03.csv
```

In the Open file dialog of **AthosGEO View**, this group would appear as such as shown in Fig. 2.82.

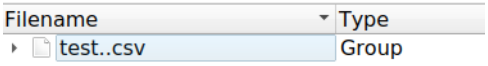


Fig. 2.82: In an **Open** file dialog, a numbered group of files will appear like this and can be selected as a whole.

File types canot only be **CSV**, but also all other file types that are known to **AthosGEO View**, like **VTU**, **VTD** and many others.

Once a file group is loaded, it will be shown in the related viewer, like a **Table View** or a **Render View**, once the user has pressed **Apply**.

At this moment, the tools of the **animation tool bar** are available to run step by step through the loaded files. What this means depends on the viewer:

- A **Render View** will visualize geometric objects one after the other
- A **Table View** will simply go through the different table data

Attaching filters to animations

Many **filters** can be attached to an animation. Once the animation is run, they will be applied to the different files one after the other, generating a resulting animation.

Saving animation data to files: Extractors

Animations cannot only be displayed, but also written to disk files. This can be the output of a filter that generates an animation by it's own, or it is the output of a filter that takes an animation as input, generating an animation also as output.

It is now possible to attach an **Extractor** to such an output port, and there are different extractors available, according to the generated data type: **CSV**, **VTU**, **VTD** or others.

One of the properties that have to be specified is an output filename, as shown in Fig. 2.83. The default default filename can be overwritten, but **make sure to keep a {timestep:...} section**: The output will not be a single file, but a numbered sequence of files, and the number will be generated at the location of this section!

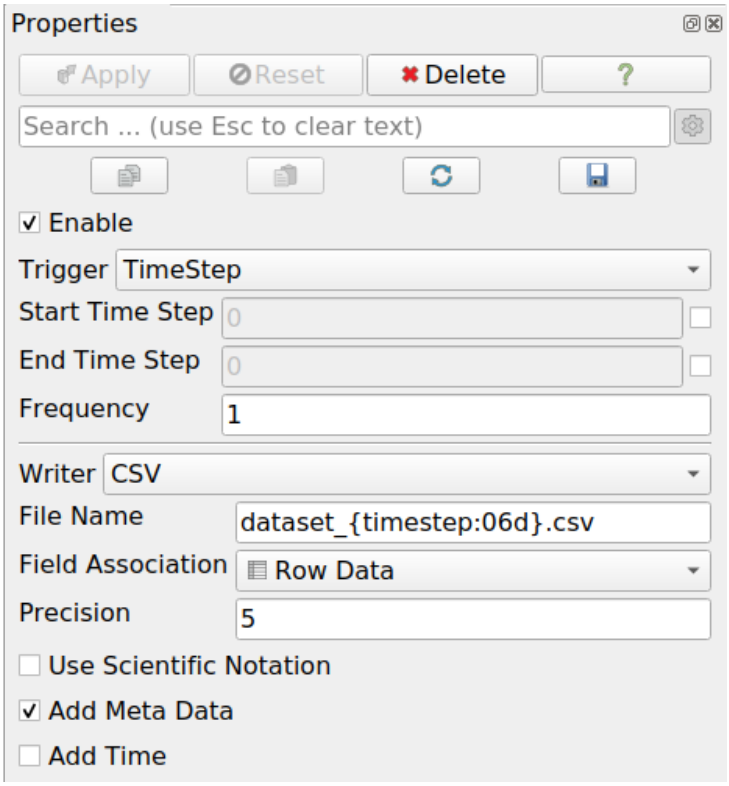


Fig. 2.83: The **Properties** of a **CSV Extractors** include a filename which needs to include a **{timestep:...}** section, with some format indicator: If this is **06d** for examples, it will be numbers in a 6 digit field, filled with zeroes like **filename000001.csv, filename000002.csv** etc.

Saving the file sequence will be initialized with the **Save Extracts** tool button  .

The user will be asked to choose a folder for the output files, then the animation will be run automatically, writing the output to the files as desired.

One way to further process such numbered file sequences is to load and visualize them as an animation, as described above.

INDEX

G

H

I

L

M

O

P

R

S

T

U

V

W